Call Center Mathematics

A scientific method for understanding and improving contact centers

Ger Koole

Version of January 26, 2007

Department of Mathematics, Vrije Universiteit Amsterdam, and CCmath consulting and software.
Internet: www.math.vu.nl/obp/callcenters and www.CCmath.com.
Contact: koole@few.vu.nl, +31 6 24922626.

Preface

This book is written for everybody who is dedicated to improving call center performance. It offers a scientific method to understanding and improving call centers. It explains all generic aspects of call and contact centers, from the basic Erlang formula to advanced topics such as skill-based routing and multi-channel environments. It does this without using complicated mathematical formulae, but by stressing the meaning of the mathematics. Moreover, there is a companion web page where many calculations can be executed. Next to understanding call center phenomena we show how to use this insight to improve call center performance in a systematic way. Keywords are data collection, scenario analysis, and decision support.

This book is also a bridge between call center management and those parts of mathematics that are useful for call centers. It shows the manager and consultant the benefits of mathematics, without having to go into the details of it. It also shows the mathematically educated reader an interesting application area of queueing theory and other fields of mathematics. As such, this book can also be used as additional material in an applied course for mathematics and industrial engineering students. Basic knowledge of call centers is assumed, although a glossary is added in case of omissions.

Ger Koole
Amsterdam/Sophia Antipolis, 2001–2005

Contents

Chapter 1

About this book

This book can be used on its own, but to get the most out of it it is advised to use the companion web site, www.math.vu.nl/~koole/ccmath. This site contains updates of the book, a list of typos, and, most importantly, the web tools in which you can do the calculations that illustrate the text.

1.1 No maths

This book is meant for call center managers and presumes no knowledge of mathematics whatsover. Not being familiar with higher mathematics does not mean that you cannot use maths to improve call center performance. For example, even without knowing the maths everybody can understand the Erlang formula. (And, vice versa, knowing the Erlang formula, does not mean that you understand it!) This book is directed towards understanding the quantitative aspects of call centers. Current computer-based systems allow us to gain this understanding without knowing the maths themselves. Only engineers that implement the math in our computer systems should know and understand the formulas. This is the second group for which this book might be helpful: engineers and mathematicians that know the mathematics, who want to be introduced to the field of call centers. For the mathematically interested readers some formulae are supplied in the appendices, where also an extensive list of references can be found to assist further study.

1.2 Why a web site?

In this book we give the results of many calculations. Most of these can also be executed by the reader through the web page of this book: www.math.vu.nl/~koole/ccmath. By putting them on a web site and not on a CD the reader has always access to the latest version of our tools.

1.3 Feedback

I'm still working on this book, and I'm open to all kinds of feedback and other questions. Do not hesitate to send me an email at koole@few.vu.nl.

1.4 Overview

Here we give a short overview of what can be expected in the remaining chapters of this book. The first chapters are of a general nature, discussing the benefits of call center mathematics and call center management objectives. Then we continue with those subjects that are relevant to every call center manager or planner: the Erlang C formula, forecasting, and staffing. The final chapters contain more advanced subjects, such as extensions to the Erlang C model and multiple skills. We conclude with a glossary, an annotated bibliography, and an appendix with the math of the Erlang C formula.

1.5 How to use this book

Sections with "*" in the title contain side topics, often of a more mathematical nature. They need not be read for understanding the main text. The same holds for remarks that are typeset in a smaller font. The many examples are put in *italics*. Quite often these examples include calculations involving one of the calculators on the web site. For a good understanding it is crucial to repeat the calculations yourself as well.

1.6 Acknowledgments

I would like to thank some people for their input and corrections: VU-alumnus Theo Peek, research assistant Arnout Wattel, and consultant Roger Rutherford. The tools on the web site were made by the graduate students Marco Bijvank and Auke Pot and by Arnout Wattel.

My thanks also go to the Maestro project at INRIA Sophia Antipolis for their hospitality during my visits in the last years that allowed me to write the bigger part of this book.

Chapter 2

What is Call Center Mathematics?

In this chapter we explain how call centers can benefit from a mathematical approach.

2.1 The subject of Call Center Mathematics

To manage call centers, or more generally, contact centers, effectively, one needs to have multiple skills. Roughly speaking there are those skills which are unique to the product that is delivered, and there are those skills that are needed in virtually any call center. Some of these latter skills are soft, such as training and motivating people. Other skills are of a more quantitative nature, and are related to service level and an efficient use of the main resource, the people that work in your call center. Mathematics can play an important role in getting the best out of the service level/cost trade-off. In a simple single-skill call center we see that the Erlang formula is used to determine the occupation level at any time of day. Scheduling algorithms are then used to determine shifts and to assign employees to shifts. In more complex environments mathematics is used to route calls, to decide how and when call blending is done, and so forth. Mathematics are an essential part of call center management.

2.2 Why should a call center manager know about mathematics?

Let us first put you at ease by stating that we do not think that managers in call centers should know the mathematics themselves. We do think that managers should know about the implications of mathematical theory for call centers. But if the mathematics are already implemented in the software, why know anything about it? Why should we understand the Erlang formula if it is readily available in many decision support systems? The answer to this lies in the name decision *support* system. No computer-based system can completely automate the complex scheduling and planning tasks in a call center. Human interaction is always needed, and this is only possible if the user *understands* the software. In this way, learning about call center mathematics increases the effectiveness of the available software.

On the other hand, certain tasks within a call center, such as call routing, are completely automated. But here the crucial decision was taken at the moment the routing algorithm was implemented. Again, only an understanding of the dynamics of call centers can help us to implement the right routing machanisms. Thus again, an understanding of call center mathematics will help us make better decisions. The same holds for long-term decisions about the structure of a call center. Mathematics can help us understand and quantify decisions related to the merging of call centers, call blending, multi-channel management, and so forth.

A better understanding will also improve the communication with other people, not in the least with the consultant who is trying to sell a model-based solution.

2.3 Two ways to call center improvement

Any business change raises questions about the effectiveness of the proposed change. Will it really work out the way it is foreseen? Given our understanding of call centers we often have an idea what the type of effect will be of certain changes. Direct implementation of the proposed changes, *on-line experimentation*, has the advantage of simplicity and low costs. But these costs remain only low if the effects of the changes are positive! For this reason one often likes to experiment first in a “laboratory” setting. Mathematics offers such a “virtual laboratory”. The important aspects of reality are described in a mathematical *model* and this model can be analyzed using mathematical techniques. This way different scenarios can be analyzed, hence the term *scenario analysis*. But mathematics can do more. It can generate solutions for you. This is what a workforce management tool does when it generates an agent schedule. This solution can be of varying quality, depending on the model that is implemented. In theory mathematics can generate solutions that are better than those that are thought of by a human, and in much less time. The mathematical model however is constructed by humans, and everything depends on its quality.

Merging two call centers leads to economies-of-scale advantages. However, the physical costs of such a merger can be high! Calculations based on the Erlang model can quantify the expected cost reduction. This way a reasonably accurate cost trade-off can be made.

There are in principle two ways to use mathematics to improve the performance of a call center. There is a ‘minimal’ approach in which the insights obtained from this book and other sources are used to give rough estimates of consequences of possible management decisions. This type of ‘back-of-the-envelope’ calculations take little time, probably involve the use of a spreadsheet and one or more call center calculators, some performance indicators that are already known, and perhaps contact with an in-house OR professional or mathematician. It is mainly to equip the manager or planner with the skills to execute this type of task that I wrote this book.

The second approach is the ‘standard’ Operations Research approach, but not necessarily the best. It consists of building a mathematical model of the call center, estimating all relevant parameters, and drawing conclusion from a thorough analysis of the model. This analysis often involves simulation of the whole system. This approach is time consuming, usually performed by external consultants, and whether it really gives good results is sometimes doubtful. It works best for operational problems of a repetitive nature.

Whatever approach is taken, improving your call center does not end with modeling (parts of) it once. Improving your call center is a continuous process in which the subsequent modeling steps (model construction, data collection and analysis, running scenarios, implementation) are followed again and again.

2.4 What to expect from call center mathematics

Mathematics can help you manage your call center. However, you should not expect miracles. Every modeling exercise implies simplifying the real situation first to fit it in the framework of the model. With this modeling step certain approximations are introduced, requiring a careful use of the outcomes. Modeling everything simply isn't possible, because of time constraints and because for example human behaviour cannot be modeled in all details. Sometimes you won't even be able to get all the information you want out of the ACD! This doesn't make modeling useless, but it requires an attitude in which outcomes of modeling studies are tested thoroughly before being implemented. Never implement a proposed solution until you are completely confident that it will work out the way it is intended!

Chapter 3

On call center management and its goals

In this chapter we discuss the overall goals of call center management. Starting from these overall goals, that hold for longer time periods, we formulate objectives for short periods. In the next chapters we make, in all detail, the translation back from short to long time intervals. We also discuss what types of decisions can be taken to fulfill these goals.

3.1 Cost versus service

A call center offers a service, delivered through telephone calls with clients. Service level can be defined as the degree of satisfaction of callers with the offered service. This service level consists of many different aspects, related to the quality of the answer, the waiting time of the customer, etc. Some of these are hard to quantify, such as friendliness of the agent, others are more easily quantified.

A help desk tries to answer 90% of all question correctly during the first call. Next to that they require that 80% of the calls is answered within 20 seconds waiting, and that no more than 3% of the calls abandons before getting a representative.

The manager of a call center tries to satisfy the service levels set by higher management, given its budget, and other constraints such as the number of work places (often called seats), the ICT infrastructure, and the available workforce. Of course, the higher the budget, the higher the service level can be, due to better training and more available resources. The main resource is the call center agent or representative, although communication costs can also be high, certainly for toll-free services. This means that the (infra)structure and processes of a call center should be such as to maximize the effective and efficient employment of the workforce.

The cost-service level trade-off thus has a central place in quantitative call center management. In general, when costs increase, then the service level (SL) increases. Thus we can draw a graph in which we show the SL as a function of the costs. This is called the *efficiency curve.* Every call center has its own efficiency curve. Where the effciency curve lies depends on the infrastructure and the processes. They often depend on long-term decisions, while the cost-SL game is played on a daily basis. For this reason we discuss next the various types and levels of decisions that influence the performance of a call center.

After that discuss costs and SL in more detail.

In certain situations the profit of each individual call can be measured in terms of money. In such a situation the average profit per handled call can be calculated, and instead of balancing cost and service level, we just maximize profit. We will pay attention to this business model in Chapter 8.

3.2 Types of management decisions

In this book we focus on decisions taken by call center managers, planners, and shift leaders. However, decisions relevant to call centers are also taken by other people and by software. We discuss all relevant types of decisions.

Strategic decisions Strategic decisions are made by upper management. They concern the role of the contact center in the company, the type of service that is to be delivered, etc. It imposes the framework in which the call center management has to work. Upper management also decides on the budget that is available to the call center.

Tactical decisions Tactical decisions are typically taken by the call center management, They concern how the resources are to be used. These resources consist of the budget, the existing ICT equipment, and the (knowledge of) the people working in the call center.

Decisions about structure (e.g., skill-based routing) and organization are taken at this level, as well as decisions about the hiring and training of agents.

Planning decisions At the operational level we can still distinguish between the time-horizon in which decision take effect, ranging from weeks to milliseconds. Usually on a weekly basis new agent schedules are make by a planner at the call center. This is called workforce management.

Daily control Every day decisions have to be taken to react to the current situation in the call center. Usually shift leaders monitor service levels and productivity and can react to that.

Real-time control Finally, certain decisions are taken real-time by software, usually the ACD. This concerns for example decisions about the assignment of calls to available agents. Sometimes these decisions involve complex algorithms, for example in the case of skill-based routing.

3.3 Costs and productivity

It has been noted that the main costs in a call center are salary costs of agents, and that therefore agents should work as effective and efficient as possible. The most important performance indicator for effectiveness is the First-time-resolution (FTR, also called first call resolution) percentage. However, together with the average holding time (AHT) of a call, it is also an important efficiency indicator: if the FTR is high, then few follow-up calls are needed. Increasing the FTR and reducing the AHT is a major objective in call centers. It is mainly obtained by proper training of agents.

The main indicator for efficiency is the productivity, measured over a certain period (for example, a week). It is usually given as the percentage of time that an agent is working of his or her total scheduled working time:

$$\text{Productivity} = \frac{\text{Total working time}}{\text{Total time working and time available}} \times 100\%.$$

The total working time is defined as the total talk time plus the wrap-up time. For the denominator it is less clear what we should take. Evidently, it consists of the time that the agent spends on calls plus the time that the agent is available for receiving calls. But should we count the time for breaks as well? And training? Depending on the definition of productivity very different number can occur.

An agent has a contract for 36 hours a week. Of this time she spends 3 hours on training and activities outside the call center, she takes breaks during 230 minutes, she is available waiting for calls during 265 minutes, and she is handling calls (talking plus wrap-up) during 1485 minutes.

If we do not count breaks and training then the productivity is $1485/(1485+265)\times 100 = 85\%$*, if we count brakes 75%, and if we count all the time she spends at work 69%.*

Which definition is best depends on the situation. If agents are free to take breaks whenever they like then it is probably better to include these in the denominator. In any case, all performance indicator should be considered together: a high productivity is useless if the first-time-resolution percentage is low! In fact, decreasing the first-time-resolution percentage decreases the idle time through an increase in calls and thus “improves” the productivity!

There are other obvious but interesting relations between the performance indicators. If one tries to decrease the number of performance indicators then one probably ends up considering the number of resolved calls. The disadvantage of this criterion however is that it is hard to measure.

3.4 Service level metrics

We saw that the goal of call center management is to obtain the right cost-service level trade-off. We also saw by who and by what types of decisions cost and service level can be influenced. We now go into more detail how this service level can be defined.

The service level (SL) obtained by a call consists of several different aspects. Several of these aspects are related to the handling of the calls themselves, such as the way in which the agents attend to the call, and the ratio of calls that need no need further calls, the first-time-fixed ratio. Others are related to the waiting process, notably the waiting times and the occurrence of abandonments. We focus on waiting times and abandonments, although other aspects of the service level can have a large impact on the waiting time and therefore also on the abandonments.

The help desk of an Internet Service Provider had a considerable rate of callers that phoned back after their call because the answer was not sufficiently clear to solve their problems. By improving scripts and documentation and by additional training this rate was reduced considerable. This not only improved the perceived service level, it also reduced the number of calls. This had a positive effect on the waiting times, and thus again on the service level.

The common way to define service level is by looking at the fraction of calls that exceeds a certain waiting time, which we will call the "acceptable waiting time" (AWT). The "industry standard" is that 80% of all calls should be answered in 20 seconds, but other numbers are possible as well. The SL can simply be calculated by dividing the number of calls handled before the AWT by the total number of calls.

Often we know the SL for short intervals (often giving by the ACD), and we want to compute the SL for longer intervals, for example in a spreadsheet to make a monthly report. The SL of a long period composed of several shorter of which we know the SL can be calculated be averaging in the right way service levels over the shorter periods. When averaging over a number of intervals the number of calls in these intervals should be taken into account. Consider the table below. At first sight the average service level is 75%, by averaging the four percentages, but now the differences in numbers of calls per week are not taken into account. The right way of calculating is to compute the *fraction* of calls in each interval first. For example, the fraction of calls in the first interval is $\frac{2000}{17000}$, 17000 being the total number of calls over the four weeks. Using these fractions a *weighted average* is calculated in the following way:

$$\frac{2000}{17000} \times 95 + \frac{7000}{17000} \times 55 + \frac{5000}{17000} \times 70 + \frac{3000}{17000} \times 80\% = 68.5\%.$$

Week	Number of calls	Answered within 20 s.	SL
1	2000	1900	95%
2	7000	3850	55%
3	5000	3500	70%
4	3000	2400	80%

This way of calculating averages corresponds to the answer in case the service level was computed directly for the whole month. Indeed, out of a total of 17000 calls 11650 were answered in time, thus a $\frac{11650}{17000} \times 100 = 68.5\%$ service level.

The difference between 68.5 and 75% is not that dramatic. This is because the number of calls in the different weeks are roughly of the same order of magnitude. If the number of calls in the intervals over which we average are very different, then the way of averaging can have an even bigger impact on the result. These big fluctuations typically occur during days. At peak hours we can easily have ten or twenty times as many calls per hour as during the night. Then the difference between ways of averaging can run into the tens of percents.

The percentage of calls that is answered in less than a certain fixed waiting time is sometimes called the *telephone service factor* (TSF). Another commonly used waiting time metric is the *average speed of answer* (ASA).

3.5 Interpreting the TSF*

The TSF is commonly used as SL metric in call centers, but it is of interest to pay attention to its interpretation, especially for the callers. We will consider the regular 80/20 TSF, where we assume that the number 20 is chosen such that the callers in general find the service 'good' when they wait less than 20 seconds and 'bad' when they wait more than 20 seconds. An 80/20 TSF means that 80% of the customers receive good service, and 20% bad.

Consider an individual customer that belongs to the 20% that received bad service. To this customer it is right now irrelevant if the SL was 50/20 or 80/20, in the former case there are just more unsatisfied customers. To the unsatisfied customer the SL becomes relevant when he or she tries to call again. If the TSF at that moment is again 80/20, then the probability of another bad experience is 0.2, or 20%. Almost 1 out of 100 customers have 3 consecutive bad experiences. And how many customers will try a third time after two bad experiences if they have alternatives? If the competition is strong then offering only an 80/20 TSF can lead to churn. Thus whether a 80/20 TSF or any choice is the right SL for a CC depends on the behavior of the callers. Will they call back after a bad experience, and is 20 seconds indeed the correct borderline between good and bas service? Things become even more complicated when we take abandonments into account, see the next section on this subject.

Choices related to SL become even more difficult when we consider call centers with multiple types of calls (see chapter 9 for more on this). Consider for example two types of calls for which we like to obtain both an 80/20 SL. Now what if we obtain 70/20 on one and 90/20 on the other? and if we have the choice, with the same means, between 70/20 and 90/20 or 75/20 and 80/20? The former has a better average SL (assuming an equal load), the latter shows less variance. The answer depends again on the behavior of callers and the nature of the service: will they mostly generate the same type of call, or do they change type? In the former case we should consider the types independently, in the latter case we should perhaps focus on the average SL.

The situations becomes even more complicated when we have different SL constraints for different call types, for example because we want the sales line to have a better SL than the after-sales line. Here we might have 90/20 and 70/20 constraints, and still be more satisfied when we realize 95/20 and 65/20, simply because we value individual sales calls higher than after-sales calls. A SL definition that corresponds with our intuitive notion of SL might be to have a constraint on the high-value calls of 90/20 and an average constraint of 80/20.

3.6 Service level and abandonments

A phenomenon that occurs in every call center is that callers *abandon* (or *renege*) while waiting in the queue. In general, this is considered to be something to avoid, although some callers abandon in less than the AWT. One way to deal with abandonments is by setting a separate service level constraint on abandonments, e.g., on average not more than 3% abandonments.

If the TSF is used, then there is also the possibility to integrate the abandonments in this way of choosing the SL. For this, we first have to decide how to count abandonments. It is clear that callers who abandon after the AWT have received bad service, and therefore these calls are added to the number of calls for which the service requirement was not met. For callers that abandon before the AWT there are different possibilities. The most reasonable is perhaps not to count these calls at all. This leads to the following definition of service level:

$$\text{SL} = \frac{\text{Number of calls answered before AWT}}{\text{Number of calls answered} + \text{Number of calls abandoned after AWT}} \times 100\%.$$

Another possibility is to count them as calls for which the SL was met.

A call center receives 510 calls during an hour. The AWT is set equal to 20 seconds. A total of 460 receive service, of which 410 are answered before 20 seconds. Of the 50 abandoned calls 10 abandon before 20 seconds. Therefore the service level is $\frac{410}{460+40} \times 100\% = 82\%$*. Not taking abandonments into account when computing the SL would lead to a SL of* $\frac{410}{460} \times 100\% = 89\%$*!*

These ways of calculating the service level are all easily done on the basis of observed waiting times of calls: one needs to remember the numbers of served and abandoned calls that get served or abandon before and after the AWT, in total four numbers per interval for which we want to know the SL.

Another way of defining the service level is to compute it from the waiting time of ‘test customers’ who have infinite patience. In general this leads to numbers very close to the definition in which we ignore customers who abandon before the AWT. This definition is attractive because it is independent of the patience of a caller. On the other hand, it is somewhat more complicated to derive from the observed statistics: just the four numbers as above do not suffice, one should really introduce virtual test customers and look what their waiting times would have been.

Service levels can be measured in two different scales: between 0 and 100 or between 0 and 1. We will use both. To go from one scale to the other we simply have to divide or multiply by 100. Mathematicians often prefer to measure between 0 and 1, because the results can be interpreted as fractions or probabilities. Although it will be clear usually, we will always use the ”%” sign when using the percentage scale.

We should also consider how to incorporate abandonments in the ASA, in case the ASA is used as service level metric next to or instead of the TSF. Defining the ASA in the case of abandonments is done by looking at the ASA of test customers with infinite patience.

3.7 A discussion of service level metrics*

When such a complex phenomenon as service level is reduced to a few numbers, then it is unavoidable that certain aspects are ignored.

As an example, take the waiting times of just 4 calls: 0, 10, 30, 100 seconds. Then the ASA is 35 seconds. However, the sequence 35, 35, 35, 35 has the same ASA. This shows that the ASA, by its proper definition, does not depend on the variability: is the ASA caused by many calls having a short waiting time or by a few calls having a very long waiting time? Both are possible!

This is a good reason to look for other service level metrics. Consider next the TSF, which is indeed, to a certain extent, sensitive to variability. However, in case of a bad SL (a low TSF) you can better have high variability, and in the case of a high SL a low variability! This can be seen from the following examples, each with AWT 20 seconds: 15, 15, 15, 15 (ASA 15, TSF 100), 0, 30, 0, 30 (ASA 15, TSF 50); 25, 25, 25, 25 (ASA 25, TSF 0), 15, 35, 15, 35 (ASA 25, TSF 50).

Another disadvantage of the TSF is that does not take into account the waiting time in excess of the AWT: the sequences 0, 10, 30, and 100 seconds and 0, 10, 24, and 30 seconds give the same TSF of 50, although there is a clear difference between the situations! The difference shows up if we vary the AWT. This however would lead to a SL metric consisting of multiple numbers, which has the disadvantage that it is harder to interpret and to compare.

Thus we find that neither the ASA nor the TSF represents the SL well. Focusing on one of these can lead to consequences that go against common sense: it motivates managers to take decisions that decrease the common perception of SL.

A call center has two types of calls: calls with a negociated SL in terms of a TSF that has to be met, and ”best effort” traffic where the revenue depends on the SL. Under high traffic conditions the TSF of the first type of calls cannot be met, even when priority is given to these calls. Therefore, the rational decision, given the contract*, is to give priority to best effort calls in case of high load and to give priority to fixed TSF calls when traffic is low to catch up with the SL. This is in complete contradiction with the intentions behind the SL contract. (Source: Milner & Olsen, Management Science, 2006.)*

It is common practice in call centers to answer the longest waiting call first. If the SL is only measured through the TSF, then this is not a good solution: calls waiting longer than the AWT should not be answered at all, instead the call that waits the longest among those that wait less than the AWT should be helped. Thus the TSF stimulates wrong behavior.

For the ASA the order in which calls are answered does matter at all. A possible solution would be to report both the ASA and the TSF. Still priority is given to calls waiting a little less than the AWT, but long-waiting calls eventually get served. An alternative SL metric consisting of a single number that motivates us to help long waiting calls first is as follows. It takes the AWT into account, and it penalizes waiting longer than the AWT by measuring the time that waiting exceeds the AWT. We call it the *average excess time* (AET). For the 0, 10, 30, 100 sequence the waiting times in excess of 20 seconds are 0, 0, 10, and 80, giving $90/4 = 22.5$ seconds as AET. For 0, 10, 24, 30 it gives 3.5, and for 35, 35, 35, 35 the AET is equal to 15. When using the AET is it clear that those calls that wait longer than the AWT get priority.

Chapter 4

The Erlang C formula

In the last chapter we saw that service levels, even for longer periods, could be derived from the service levels over shorter intervals. In this chapter we study call center performance over intervals that are short enough to assume that the characteristics do not change. The basic model for this situation is the Erlang model. This is what we study in this chapter in all detail.

4.1 The Erlang formula

In this section we introduce the famous Erlang C or Erlang delay formula, named after the Danish mathematician who derived the formula at the beginning of the 20th century. We have a call center with only one type of calls and no abandonments, thus every caller waits until he or she reaches an agent. The number of calls that enter on average per time unit (e.g., per minute) is denoted with the Greek letter λ. The average service time of calls or average holding time is denoted with β, measured in the same unit of time. We define the load a as $a = \lambda \times \beta$. The unit of load is called the *Erlang*.

Consider a call center with on average 1 call per minute, thus $\lambda = 1$*, and a service time duration of 5 minutes on average, thus* $\beta = 5$*. The load is* $a = \lambda \times \beta = 1 \times 5 = 5$ *Erlang. Note that it does not matter in which time unit* λ *and* β *are measured, as long as they are the same: e.g., if we measure in hours, then we get again* $a = \lambda \times \beta = 60 \times \frac{1}{12} = 5$ *Erlang.*

The offered traffic is dealt with by a group of s agents. We assume that the number of agents is higher than the load (thus $s > a$). Otherwise there are, on average, more arrivals than departures per time unit, and thus the number of waiting calls increases all the time, resulting in a TSF of 0%. (In reality this won't occur, as callers will abandon.) We can thus consider the difference between s and a as the overcapacity of the system. This overcapacity assures that variations in the offered load can be absorbed. These variations are not due to changes of λ or β, they originate in the intrinsic random behavior of call interarrival and call holding times. Remember that λ and β are averages: it occurs during short periods of time that there are so many arrivals or that service times are so long that undercapacity occurs. The strength of the Erlang formula is the capability to quantify the

TSF (and other waiting time measures) in this random environment with short periods of undercapacity and therefore queueing.

The Erlang C formula gives the TSF for given λ, β, s, and AWT. For the mathematically interested reader we give the exact formula, for $a < s$:

$$\text{TSF} = 1 - C(s,a) \times e^{-(s-a)\frac{\text{AWT}}{\beta}}.$$

Here e is a mathematical constant, approximately equal to 2.7; $C(s,a)$ is the probability that an arbitrary caller finds all agents occupied, the *probability of delay*. In case $a \geq s$ then TSF $= 0$. The formula itself is useful for those who *implement* it; see Appendix C.1 for details. For a call center manager it is more important to *understand* it, i.e., to have a feeling for the TSF as variables vary. For this reason we plotted the Erlang formula for some typical values in Figure 4.1. We fixed β, s, and AWT, and varied λ. In the figure we plotted λ on the horizontal axis, and the TSF on the vertical. The numbers in the figure can be verified using our Erlang calculator at www.math.vu.nl/~koole/ccmath/ErlangC.

With the numbers of the example above, $\lambda = 1$ and $\beta = 5$, we got a load of 5 Erlang. Let us schedule 7 agents, and assume that a waiting time of 20 seconds is considered acceptable, i.e., AWT$=$ 20 seconds. Filling in 1 and 5 and selecting "Number of agents" (20 is already filled in at start-up) gives after computation the TSF under "Service level". It is almost 72% (check this!). Increasing the number of agents to 8 already gives a TSF of 86%.

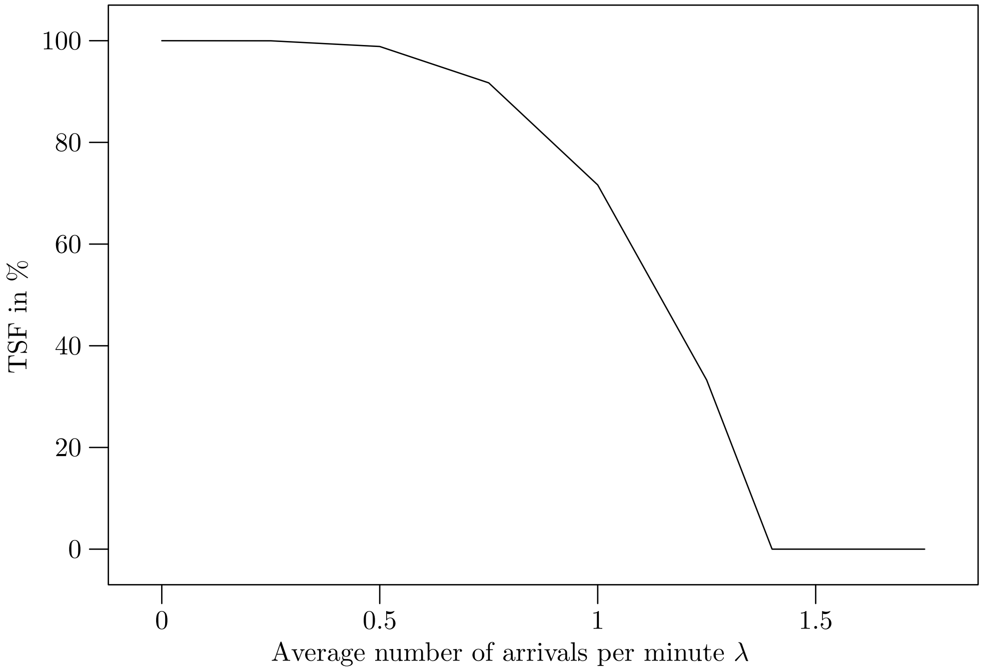

Figure 4.1: The TSF for $\beta = 5$, $s = 7$, AWT $= 0.33$, and varying λ.

We follow the curve of Figure 4.1 for increasing λ. Starting at 100%, the TSF remains close to this upper level until relatively high values of λ. As λ gets such that $a = \lambda \times \beta$

approaches s then the TSF starts to decrease more steeply until it reaches 0 at $\lambda = s/\beta = 7/5 = 1.4$. From that point on, as explained earlier, the TSF, as predicted by the Erlang formula, remains 0%.

Next to the SL in terms of the fraction of calls waiting longer than the AWT, the TSF, we can also derive the average speed of answer (ASA, also called average waiting time), the average amount of time that calls spend waiting. The overcapacity assures that the average speed of answer remains limited. How they depend on each other is given by the Erlang formula for the ASA. This formula is given by:

$$\text{ASA} = \frac{\text{Probability of delay} \times \text{Av. service time}}{\text{Overcapacity}} = \frac{C(s,a) \times \beta}{s-a}.$$

For the same input parameters as in Figure 4.1 we plotted the ASA in Figure 4.2. We see clearly that as λ approaches the value of $s/\beta = 1.4$ then the waiting time increases dramatically.

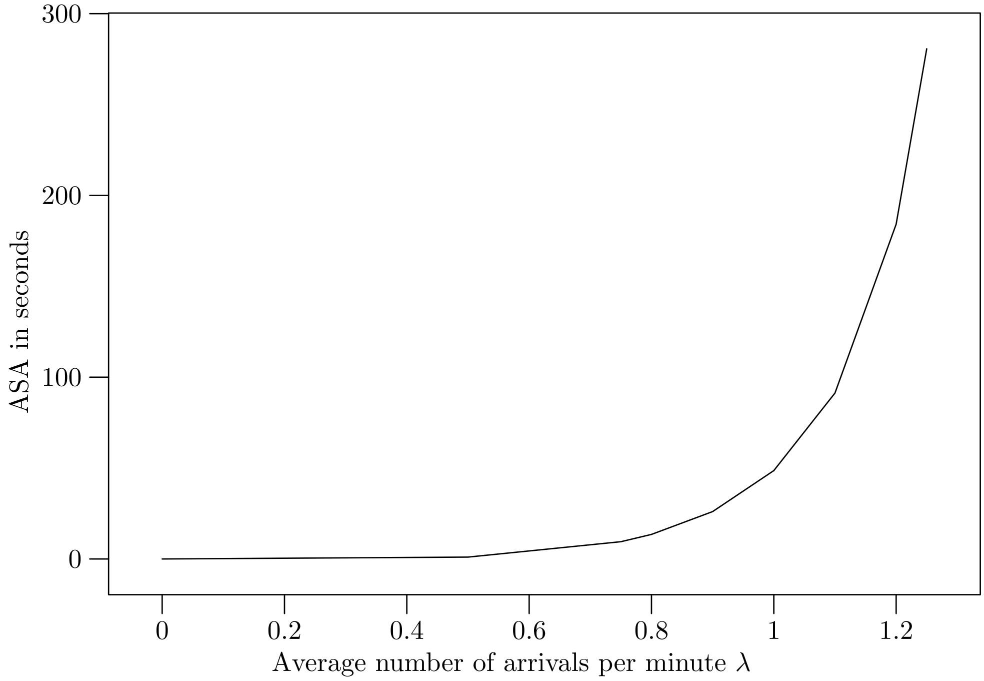

Figure 4.2: Values of the ASA for $\beta = 5$, $s = 7$, AWT $= 0.33$, and varying λ.

The probability of delay is not only an intermediate step in calculating TSF or ASA, it is also of independent interest: it tells us how many callers are put in the queue and how many find a free agent right away. The probability of delay can also be computed using an Erlang calculator. By computing the TSF for an AWT equal to 0 we find 100 minus the delay percentage. Dividing by 100 gives the probability of delay. Thus we need to fill in AWT = 0, and by noting that 100 × probability of delay = 100 − TSF.

Now we continue the example. We already saw that the load is 5 Erlang. Let us place 7 agents, then there is 2 Erlang overcapacity. Filling in 1, 5 and 7 we find that 68% waits *less than 0 seconds. Thus the probability of delay $C(s,a)$ is equal to 0.32. Now we can fill in the formula for the average waiting time, in seconds:*

$$ASA = \frac{C(s,a)\times\beta}{s-a} \approx \frac{0.32\times 300}{2} = 48 \textit{ seconds.}$$

This corresponds with the answers of the Erlang C calculator (be careful with the units, minutes or seconds!). Taking 8 agenten gives

$$ASA = \frac{C(s,a)\times\beta}{s-a} \approx \frac{0.17\times 300}{3} = 17 \textit{ seconds.}$$

Thus increasing the number of agents with 1 reduces the average waiting time with a factor 3.

Up to now we just discussed the service level aspects of the Erlang C system. Luckily, the agent side is relatively simple. Let us consider the case that $a < s$, thus $s - a$ is the overcapacity. Because every caller reaches an agent at some point in time, the whole offered load a is split between the s agents. This gives a productivity of $a/s \times 100\%$ to each one of them, if we assume that the load is equally distributed over the agents. If $a \geq s$ then saturation occurs, and agents get a call the moment they become available. In theory, this means a 100% productivity. In practice such a high productivity can only be maintained over short periods of time.

4.2 Using the Erlang formula

In the previous section we saw that the Erlang formula can be used to compute the average waiting time for a given number of agents, service times and traffic intensity. One would like to use the formula also for other types of questions, such as: for given β and s, and a maximal acceptable ASA or given SL, what is the maximal call volume per time unit λ that the call center can handle? Because of the complexity of $C(s,a)$ we cannot "reverse" the formula, but by trial-and-error we can answer these types of questions.

The question that is of course posed most often is to calculate the minimum number of agents needed for a given load and service level. This also can be done using trial-and-error, and software tools such as our Erlang calculator (to be found at `www.math.vu.nl/~koole/ccmath/ErlangC`) often do this automatically.

In our Erlang C calculator, fill in 1 and 5 at “Arrivals” and “Service time”, fill in “80” and “20” at “Service level” and select “Service level”. After pushing the “compute” button the computation shows that 8 agents are needed to reach this SL.

Most software tools will give you an integer number of agents as answer. This makes sense, as we cannot employ say half an agent. However, we can employ an agent half of the time. Thus when a software tool requires you to schedule 17.4 agents during a half an hour, then you should schedule 17 agents during 18 minutes, and 18 agents during 12 minutes. With 17 agents you are below the SL, with 18 you are above. Thus the ”bad” SL during 18 minutes is compensated by the better than required SL due to using 18 agents. In our Erlang C calculator we decided not to implement this, because we assume that the time interval is so short that a constant number of agents is required.

Let us continue the example. Selecting “Number of agents” instead of “Service level”, and pushing the “compute” button again shows that the actual service level is 86% instead of only 80% that was required.

”Garbage in = garbage out”. This well-known phrase holds also for the Erlang formula: the input parameters should be determined with care. Especially with the value of the expected call durations β one can easily make mistakes. The reason for this is that the entire time the agent is not available for taking a new call should be counted. For the Erlang formula the service starts the moment the ACD assigns a call to an agent, and ends when the agents becomes available, i.e., if the telephone switch has again the possibility to assign a call to that agent. Thus β consists not only of the actual call duration, but also of the reaction time (that can be as long as 10 seconds!), plus the wrap-up time (that can be as long as the call itself). Note that the reaction time is seen by the caller as waiting time. This should be taken into account when calculating the service levels, by decreasing the acceptable waiting time with the average reaction time.

In a call center the reaction time is 3 seconds on average, the average call duration is 25 seconds and there is no finish time. On peak hours on average 200 calls per 15 minutes arrive. An average waiting time of 10 seconds is seen as an acceptable service level. We calculate first the load without reaction time. The number of calls per second is $200/(15 \times 60) \approx 0.2222$ *($\approx$ means “approximately”), and the load is* $0.2222 \times 25 \approx 5.555$*. The Erlang formula shows that we need 7 agents, giving an expected waiting time of 8.2 seconds. This seems alright, but in reality there is an expected waiting time of no less than* 27.9 *seconds! This follows from the Erlang formula, with a service time of* $25 + 3 = 28$ *seconds (and thus a load of* $0.2222 \times 28 \approx 6.222$*), and 7 agents. The waiting time is then* 24.9 *seconds, to which the 3 seconds reaction time should be added. To calculate the right number of agents we start with a service time of 28 seconds, and we look for the number of agents needed to get a maximal waiting time of* $10 - 3 = 7$ *seconds. This is the case for 8 agents, with an average waiting time of 6.5 seconds. This way the average waiting time remains limited to 9.5 seconds.*

A possible conclusion of the last example could be that agents should be stimulated to react faster in order to avoid that an extra agent should be scheduled. However, these types of measures, aimed at improving the *quantitative* aspects of the call center, can lead to a decrease of the *quality* of the call center work, due to the increased work pressure. We will not deal with the human aspects of call center work; let it just be noted that 100% productivity is in no situation possible, and the overcapacity calculated by the Erlang formula is one of the means for the agents to get the necessary short breaks between calls.

4.3 Properties of the Erlang formula

Knowing the Erlang formula is one thing, *understanding it* is another. The Erlang formula has a number of properties with important managerial consequences. We will discuss these in this section.

Robustness One agent more or less can make a big difference in SL, even for big call centers. This is good news for call centers with a moderate SL: with a relatively limited effort the SL can be increased to an acceptable level. On the other hand it means that a somewhat higher load, necessitating an additional agent, can deteriorate the SL considerably. In general we can say that the Erlang formula is very sensitive to small changes in the input parameters, which are λ, β en s. This is especially the case if a is close to s, as we can see in Figures 4.1 and 4.2. The figures get steeper when λ approaches s/β, and thus small changes in the value of the horizontal axis give big changes at the vertical axis. This sensitivity can make the task of a call center manager a very hard one: small unpredictable changes in arrival rate or unanticipated absence of a few agents can ruin the SL. In Chapter 7 we discuss in detail the consequences of this sensitivity.

In our small call center with $\lambda = 1$, β and $s = 8$ we expect an ASA of around 17 seconds. However, there are 10% more arrivals (i.e., $\lambda = 1.1$). The ASA almost doubles to over 30 seconds!

Stretching time A second property is related to the absolute and relative values of the call characteristics, i.e., β and λ. Recall that the load is defined by $a = \beta \times \lambda$. If either λ or β is doubled, and the other is divided by two, then the load remains the same. This does not mean that the same number of agents is needed to obtain a certain service level.

A manager is working in a call center that merely connects calls, thus call durations are short. As a rule she uses a load to agent ratio of 80%. From experience with the call center she knows that this gives a reasonable service level. For parameters equal to $\beta = 32$ seconds and 15 calls per minute the load is $a = 8$ Erlang. Indeed, with 10 agents the average speed of answer is 6.5 seconds. After a promotion she is responsible for a telephone help desk with also a load of 8 Erlang, but with β approximately 5 minutes, more than nine times as much. She uses the same rule of thumb, to find out that the average waiting is now around 60 seconds!

When λ is multiplied by the same number (bigger than 1) as β is divided with, then the load remains the same but it is like the system goes slower. Evidently, the waiting time also increases. If AWT is multiplied by the same number then the TSF remains the same. The relationship between the ASA and stretching time is more complicated.

It is like saying that the load is insensitive to the ”stretching” of time. Certain performance measures depend only on s and a, but not on the separate values of λ and β. The probability of delay, $C(s, a)$, is a good example. It does not hold anymore for the TSF, here the actual value of β and λ play an important role. In fact, for given a and s, the service level depends only on AWT$/\beta$. Thus if time is stretched, and the acceptable waiting time is stretched with it, then the TSF remains the same. Of course, this is just theory, although we often see that the AWT is higher in call centers with long talk times compared to call centers with short talk times. For the ASA the effect of stretching time is simple: the ASA is stretched by the same factor.

Let us go back to the call center with $\lambda = 1$, $\beta = 5$, and $s = 8$. Then TSF $= 86\%$ for AWT $= 20$ seconds, and ASA $= 16.7$ seconds. Now stretch time by a factor 2, i.e., $\lambda = 0.5$ and $\beta = 10$. Then TSF $= 83\%$ for AWT $= 20$ seconds (a difference, but surprisingly small; the reason of this is explained below), TSF $= 86\%$ for AWT $= 2 \times 20 = 40$ seconds, and ASA $= 2 \times 16.7 = 33.4$ seconds.

Economies of scale Another well known property is that big call centers work more efficiently. This is the effect of the *economies of scale*: if we double s, then we can increase λ to *more* than twice its value while keeping the same service level, assuming that β and AWT remain constant.

A firm has two small decentralized call centers, each with the same parameters: $\lambda = 1$ and $\beta = 5$ minutes. With 8 agents the average waiting time is approximately 17 seconds in each call center. If we join these call centers "virtually", then we have a single call center with $\lambda = 2$ and 16 agents. The average waiting time is now less than 3 seconds, and employing only 14 agents gives a waiting time of only 13 seconds. An additional advantage is that there is more flexibility in the assignment of agents to call centers, as there is only a constraint on the total number of agents (although there will probably be physical constraints, such as the number of work places in a call center).

To give further insight in economies of scale, we plotted the two situations of the example above in a single figure, Figure 4.3. We consider the TSF, and take 7 and 14 agents. To make comparisons possible we put $\lambda \times \beta/s$ (the productivity) on the horizontal axis, and the TSF on the vertical axis. Because $\lambda \times \beta < s$, TSF gets 0 as soon as $\lambda \times \beta/s$ gets 1, no matter what call center we are considering.

In Figure 4.3 we see that the dashed line is more to the right: for the same productivity we see that a bigger call center has a higher TSF. Stated otherwise: to obtain a target SL, a big call center obtains a higher productivity. This is related to the steepness of the curve for productivity values close to 1, which is the sensitivity of the Erlang formula to small changes of the parameters, as discussed earlier in this section.

It is important to note that the relative gain of merging call centers (i.e., relative to the *size* of the call center) decreases as the size increases. The absolute advantage however (slowly) increases.

Consider four call centers, each with $\lambda = 10$ and $\beta = 2$ minutes, and 80% of the calls should be served within 20 seconds. If all call centers are separate then we need 24 agents in each call center, 45 in each when they are merged two by two, and 86 when we have one single call center. Merging two centers with arrival rate 10 saves 3 agents, merging two

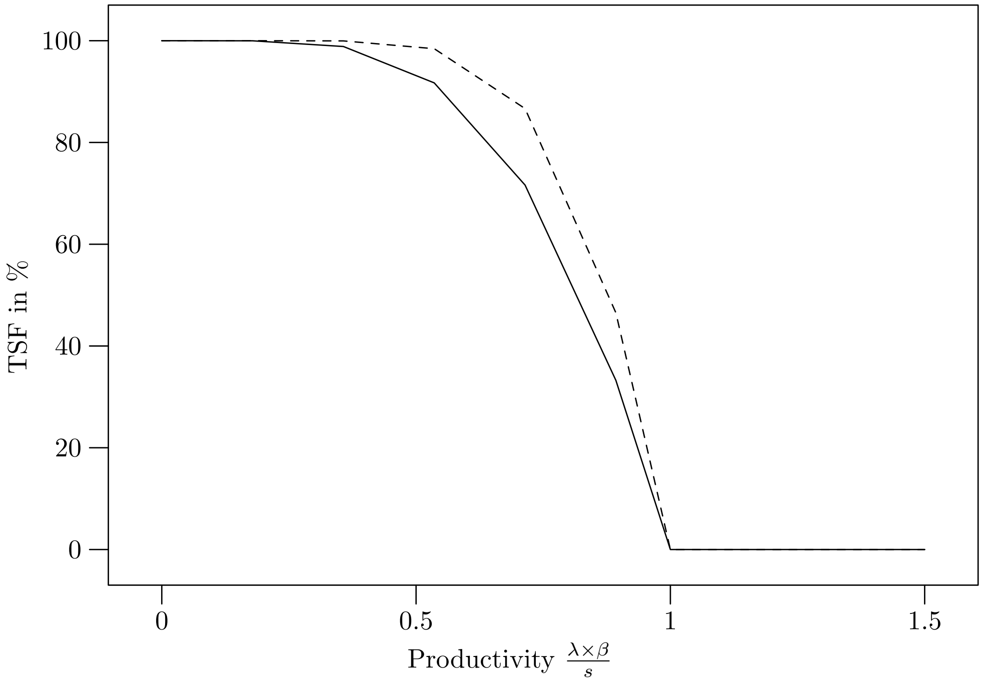

Figure 4.3: The TSF for $\beta = 5$, $s = 7$ (solid) and $s = 14$ (dashed), AWT = 0.33, and varying λ.

with arrival rate 10 saves 4. But divided by the arrival (that is, relative to the size), the economies are higher when the small centers are merged.

Variations in waiting times Consider two different call centers: one has parameters $\lambda = 1$, $\beta = 5$, and $s = 8$, the other has $\lambda = 20$, $\beta = 0.333$, and also $s = 8$. Both call centers have a TSF of around 86% for AWT = 20 seconds. Does this mean that the waiting times of both call centers are comparable? This is not the case. To make this clear, we plotted histograms of waiting times of both call centers in Figure 4.4. The level at the right of 100 denotes the percentage of callers that has a waiting time exceeding 100 seconds. We see that in the first call center, represented by the solid line, callers either do not wait at all or wait very long, there are hardly any callers that wait between 10 and 100 seconds. In the second call center (the dashed line) fewer calls get an agent right away, but very few have to wait very long.

There are two conclusions to be drawn from this example. In the first place: the TSF does not say everything. But more importantly, we see that depending on the characteristics of a call center there can be more or less variations in waiting times. Only a thorough investigation of for example the TSF for various AWTs can reveal the characteristics of a particular call center.

The remaining waiting time* When we enter a queue that we can observe (as in the post office or the supermarket) then we can estimate our remaining service time on the

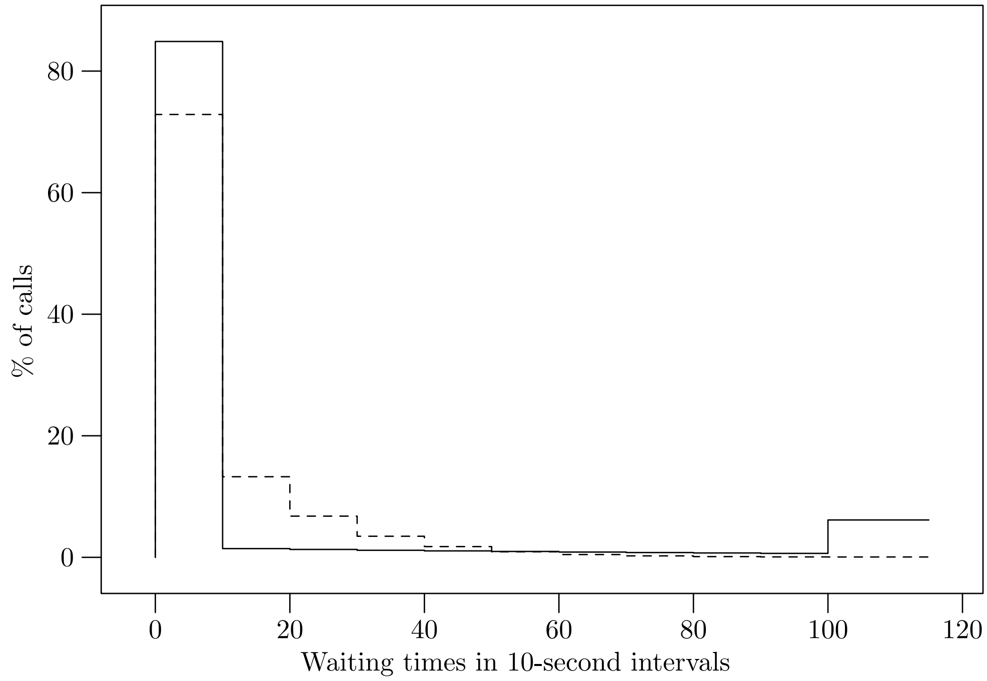

Figure 4.4: Histograms of waiting times for two different call centers.

basis of the number of customers in front of us. Usually our extimation of the remaining waiting will decrease while we are waiting as we see customers in front of us leaving. But how about the remaining waiting time in an *invisible* queue as we encounter in call centers? The mathematics show that under the Erlang C model the remaining waiting time is constant. Thus, no matter how long we have been waiting, the average remaining waiting time is always the same. How can this at first sight counterintuitive phenomenon be explained? As we enter the queue, we expect a certain number of calls to be waiting in front of us. As we wait a while, then we conclude that apparantly the queue was longer than expected. From the Erlang formula it follows that, as long as we are waiting, the expected number of customers remains always the same.

A possible consequence of this fact for customers is that one should not abandon while waiting: why hang up after 1 minute if your remaining waiting time is as long as when you started waiting? In practice however there are good reasons to hang up after a while, and good reasons to stay in line. A reason to hang up is the fact that customers do not know the call center's parameters, and therefore they do not know the average waiting time in the call center. The longer you wait, the more likely you entered a call center with unfavorable parameters, and thus your remaining waiting time does increase! On the other hand, the Erlang C formula does not account for abandonments. If your patience is longer than that of the customers 'in front of you', then they will abandon before you and you will eventually be served. In a system where calls abandon the average remaining waiting time decreases while waiting.

For call center managers it should be clear that, unless customers abandon quickly, very long waiting times can and will occur exceptionally. Theoretically there is no upper limit to the waiting time. To protect customers against unexpectedly long waiting times I think that it is good to inform customers on expected waiting times or numbers of customers waiting in the queue. Together with this the waiting customer could be pointed towards other channels to make contact such as internet (see also page 45).

4.4 The square-root staffing rule*

Up to now we saw that an increase of scale leads to advantages with respect to productivity and/or service level. These advantages can always be quantified using the Erlang formula. To obtain a general understanding we formulate a rule of thumb that relates, for a fixed service level, call volume and the number of agents. In a formula this relation can be formulated as follows:

$$\text{overcapacity in } \% \times \sqrt{s} = \text{constant}.$$

The constant in the formula is related to the service level, the formula therefore relates only overcapacity and the number of agents. The percentage overcapacity in the formula is given by $100 \times (1 - a/s)$. From the rule of thumb we obtain results such as: if the call center becomes four times as big, then the overcapacity becomes roughly halve as big. How we obtain this type type of results is illustrated by the following example.

A call center with 4 agents and $\lambda = 1$ *and* $\beta = 2$ *minutes has an average waiting time of a little over 10 seconds. For this call center the associated constant is* $100 \times (1 - 2/4) \times \sqrt{4} = 50 \times 2 = 100$*. If we multiply* s *by 4, than* $\sqrt{s}$ *doubles. Thus to keep the same service level (the same constant), we halve the overcapacity to 25%. Thus the productivity becomes 75%, and thus with* $s = 4 \times 4 = 16$ *this gives* $a = 12$ *and* $\lambda = 6$*. If we verify these numbers with the Erlang formula, then we find an average waiting time of a little over 6 seconds. Closest to 10 seconds is* $s = 15$*, with approximately 12 seconds waiting time. If we multiply* s *again with 4, then the overcapacity can be reduced to 12.5%. This means* $\lambda = 28$*, with 3.2 seconds waiting time. Closest to 6 seconds is* $s = 62$*, from which we see that the rule of thumb works reasonably well.*

From the example we see how simply we can get an impression of the allowable call volume if we change the occupation level. More often we prefer to calculate the number of agents needed under an increase in call volume. The calculation for this is more complex. If we denote with c the constant related to the service level divided by 100, then the formula for s is:

$$s = \left(\frac{c + \sqrt{c^2 + 4a}}{2} \right)^2 .$$

As in the previous example we start with 4 agents, $\lambda = 1$ *and* $\beta = 2$ *minutes. The number* c *is the constant divided by 100, thus* $c = 1$*. Filling in* $a = \lambda \times \beta = 2$ *and* $c = 1$*, then we find indeed* $s = 4$*. Assume that* λ *doubles. Then* $a = 4$*, and with* $c = 1$ *we find* $s \approx 6.6$*.*

This is a good approximation: $s = 7$ gives a waiting time under the 10 seconds, $s = 6$ above. If $\lambda = 10$, the we find $s = 25$ as approximation. An agent less would give a waiting time of 9 seconds. If λ doubles again, then we get 47 as approximation, with 45 as best value according to the Erlang formula. We see that for big values of λ doubling leads to doubling s.

If c is small with respect to a then we see that s is proportional to a. This means that the economies of scale become less for very big call centers, because it is already almost at the highest possible level. What "big" is in this context depends on the service level.

Using this rule of thumb should be done with care. It is only useful to relate λ and s. Next to that, one should realize that it is only an approximation, the results need to be checked with the Erlang formule before use in practice. This point was illustrated in the example.

4.5 How good is the Erlang formula?*

In this section we consider the weak points of the Erlang formula and its underlying assumptions. It will motivate some of the more sophisticated models that are discussed in Chapter 8.

It might come as a surprise that the ASA is bigger than 0 although there is overcapacity. The reason for this is the variability in arrival times and service durations. If all arrival times were equally spaced and if all call holding times were constant, then no waiting would occur. However, in the *random environment* of the call center undercapacity occurs during short periods of time. This is the reason why queueing occurs. The queue will always empty again if on average there is overcapacity. The Erlang formula quantifies the amount of waiting (in terms of ASA or TSF) for a particular type of random arrival and service times. The mathematical random processes that model the arrivals and departures are therefore nothing more than approximations. The quality of the approximation and the sensitivity of the formula to changes with respect to the different aspects of the model decide whether the Erlang formula gives acceptable results. We deal with the underlying assumptions one by one and discuss the consequences for the approximation.

Abandonments In a well-dimensioned call center there are few abandonments. Not modeling these abandonments is therefore not a gross simplification. However, there are call centers that show a completely different behavior than predicted by the Erlang formula because abandonments are not explicitly modeled. In general we can state that abandonments reduce the waiting time of other customers, thus it is good for the SL that abandonments occur! In call centers with a close to or even exceeding s it is crucial to model abandonments as well. Luckily this is possible. The corresponding model is discussed in Chapter 8.

Retrials Abandonments are relatively well understood and the Erlang C formula can be extended to account for abandonments without too much difficulty. This is no longer true if the customers who abandoned start to call again and thus generate retrials. Little is known about the behavior of customers concerning retrials and about good mathematical models. Unfortunately retrials are a common phenomenon in most call centers.

Peaks in offered load Formally speaking, the Erlang formula allows no fluctuations in offered load. However, in every call center there are daily changes in load. As long as these changes remain limited, and, more importantly, if there are no periods with undercapacity, then the Erlang formula performs well for periods where there are little fluctuations in load and number of agents. By using the Erlang formula for different time intervals we can get the whole picture by averaging (as explained in Chapter 3). However, as soon as undercapacity occurs then the backlog of calls from one period is shifted to the next. This backlog should be explicitly modeled, which is not possible within the framework of the Erlang formula. Therefore the Erlang formula cannot be used in the case of undercapacity. For a short peak in offered traffic (e.g., reactions to a tv commercial) straightforward capacity calculations ignoring the random behavior can give quite good results. See also Chapter 8.

Type of call durations The Erlang formula is based on the assumption that the service times come from a so-called *exponential distribution*. Without going in the mathematical details, we just note that all positive values are possible as call durations, thus also very long or short ones, but that most of the durations are below the average. Certain measurements on standard telephone traffic show that call durations are approximately exponential, although the results in the literature do not completely agree on this subject. A typical case where call durations are not exponential is when there are multiple types of calls with different call length averages, or if a call always takes a certain minimum amount of time. In these cases one should wonder what the influence is of the different service time distributions on the Erlang formula. We can state that this influence decreases as the call center increases in size. With some care it can be concluded that only the average call duration is of major importance to the performance of the call center.

Human behavior Up to now we ignored the behavior of the agents, apart from the time it takes to take up to phone. However, agent behavior is not as simple as that. Employees take small breaks to get coffee, to discuss things, etc. Modeling explicitly the human behavior is a difficult task; describing and quantifying the behavior is even more difficult! In most situations these small breaks are taken when there are no calls in the queue. It can therefore be expected that they are of minor importance to the SL. In other situations it has a bigger impact, and it can seriously limit the possibilities of quantitative modeling.

Chapter 5

Forecasting

The Erlang C formula or one of its generalizations is at the heart of call center management. It shows it many important properties that help us design optimally the call center, and it plays a crucial role in Workforce Management: it translates the call volume estimates into requirements on the number of agents that we need.

The step before executing the Erlang formula is forecasting call volume and other parameters. It is the subject of this chapter. The theory of forecasting is part of statistics, but we will assume no prior knowledge of this part of mathematics.

5.1 Attitudes towards forecasting

How should forecasts be made? And what should we expect from forecasts? In practice one encounters two types of attitudes towards forecasting in call centers. The first type of planner has the believe that forecasting can give exact results that are highly reliable. The outcomes of the forecasting module of the WFM tool are taken as exact and agent schedules are made with little room for later adjustments. Errors in the forecasts are due to the way in which the forecasts are generated and can in principle be completely avoided.

The second type of planner understands that the future can never be exactly predicted. However, for the near future we quantify the possible variation from single-number ‘point estimates’ of call volume and other parameters. These estimates plus their bandwiths will serve as the basis for further planning steps, where the call center should be designed such that there is the flexibility to deal with deviations from the point estimates (see Chapter 7). What makes forecasting more difficult than forecasting in most other situations is discussed in the next section.

5.2 The challenges of forecasting

Forecasting in call centers is not easy for a number of reasons:
- Forecasts have to be detailed, say one for every 15 or 30-munite period;
- Forecasts have to be precise;

- Forecasts depend on many known and unknown factors;
- There are many dependencies between call volume at different times;
- Business changes can have important consequences on call volume;
- Relevant data is often lacking.

Let us discuss these issues one by one.

Forecasts have to be detailed Calls usally have to be answered within less than a minute. To match the load with enough agent capacity as it is varying over the day we should not only estimate the daily call volume, but we should specify it up to the smallest interval that we distinguish in our schedules, often 15 or 30 minutes long.

Forecasts have to be precise In Figure 4.3 we saw that the TSF curve gets steeper and steeper as the productivity approaches one and as the size of the call center grows. This is equivalent to saying that the TSF is very sensitive to small changes in the forecast. Therefore the forecasts have to be very precise or other measures have to be taken to deal with unreliable forecast (see Chapter 7).

Forecasts depend on many factors Evidently, given what is said earlier, forecasts depend on the time of day. They also depend on the day of the week, and yearly fluctuations make that also the month plays a role. But that is not all: many other factors such as holidays, weather conditions, etc, can have a big impact. Some of these are known in advance, others are not. We will discuss this in detail.

There are many dependencies When call volume early on a day is high then experience shows that it will be high during the whole day. This means that there is a positive correlation between the call volume in different periods. This is an important observation that has to be exploited when reacting to changes deviations from forecasts occur.

Business changes have consequences on call volume This remark seems obvious, but it should that for example marketing decisions can have a major impact on call volume. Evidently the call center management should have the time to take measures; ideally they are involved in decisions that have a considerable impact on call volume.

Relevant data is often lacking Although in many call centers almost all transaction data is stored, data is sometimes less useful because business rules have changed since. Here we must think of the merging of call centers and the changing of scripts, changes in routing and skill groups in multi-skill call centers, or changes in products. Also changes in hardware and software play a role.

5.3 The objective

The ultimate goal of forecasting would be to come to per-interval estimates and 'confidence intervals' of all relevant parameters. Confidence intervals are intervals in which a parameter falls with say 90% statistical confidence.

On Monday May 17 8.30-9.00 the expected number of calls is 41.8, with 90%-confidence interval $[39.7, 43.8]$*; the next half hour the expected number is 117.1 with confidence interval* $[113.7, 120.4]$*. (These numbers are not made up but come from an actual call center.)*

However, deriving these confidence intervals is statistically involved and makes human intervention difficult. Compared to current practice it is already a big step forward for planners to realize that variations occur and to have a global idea of the size of these variations. As such, the objective of forecasting becomes the calculation of reliable per-interval estimates of all parameters that are needed for call-center management. The following numbers have to be estimated for each planning interval and for each call type:

- the number of inbound calls;
- the talk time;
- the wrap-up time.

The number of calls varies strongly over the day and between days, and has the highest variation. Talk time and wrap-up time vary as well, but much less than the number of inbound calls.

Next to these standard statistics it might also be useful to estimate the following quantities:

- the pick-up time;
- the patience of callers (i.e., the time they are willing to wait before they abandon);
- the retrial probability (i.e., the probability that after an abandonment the caller will call again) and the time until they retry.

Next to these properties of the offered traffic there are also a number of parameters that are related to the agents. An important number is the percentage of agents that are absent because of illness. On a longer time-scale it is important to estimate agent turnover. It is also important to measure variations in average talk time, pick-up time, etc., per agent. This can be useful for planning purposes, but it is mainly used for agent evaluation and training.

5.4 The standard forecasting method

In this section we concentrate on forecasting the offered numbers of calls. This is, together with the different call handling times, the most important number to estimate, and also one of the hardest.

Call volume depends on the year, the week of the year, the day of the week, and the moment of the day. The long-term changes in call volume are called the trend. Apart from the trend, it is usually observed that the fluctuations over the year follow a similar pattern:

e.g., few calls during the summer holidays, many calls just after it, etc. Similarly, every week also show a similar pattern: Tuesday, Wednesday, and Thursday look very much alike, but Monday morning shows more calls, Friday afternoon less. Every interval represents a fraction of the total weekly call volume: this is called the weekly profile. Starting from a weekly call volume estimate the estimate for any interval during the week can thus be obtained by multiplying the weekly estimate by the fraction for that particular interval. What remains is the estimation of the weekly call volume, which can be taken equal to the call volume in the same week the year before times the trend. The trend can be obtained by some standard statistical technique.

In the table below we see the weekly numbers of calls in an actual call center for 13 weeks in two successive years, together with the percential decreases. These decreases range from 4 to 20%. The simplest estimation of the number of calls in a certain week would be to decrease the number of year 2002 by the decrease of the week before. For example, for week 52 this would be $1916 * (1 - 0.19) \approx 1551$. The real value was 1596, an error of 3%, because the decrease with respect to 2002 appeared to be only 16%.

Week	40	41	42	43	44	45	46	47	48	49	50	51	52
# calls 2002	3412	3409	3313	3429	3538	3495	3494	3555	3430	3341	3537	3701	1916
# calls 2003	3086	2865	2660	2932	3124	3341	3250	2984	3306	2924	3015	2985	1596
decrease in %	9.6	16.0	19.7	14.5	11.7	4.4	7.0	16.1	3.6	12.5	14.8	19.3	16.6

As the monthly decrease is highly fluctuated, it is better to take the decrease of more months into account: hopefully the fluctuations will average out somewhat. Taking a *moving average* (the average over the last say 10 weeks) or using *exponential smoothing* (which gives an exponentially decreasing importance to previous weeks) are relatively simple ways to do this averaging. Exponential smoothing (with parameter 0.5) gives a very nice estimation for week 52: 1617, only a 1% error. It is computed as follows: the decrease of last week has a weight $\frac{1}{2}$, the decrease of the week before $\frac{1}{4}$, etc. With that we come to the following estimate of the trend for week 52:

$$\frac{1}{2} \times 19.3 + \frac{1}{4} \times 14.8 + \frac{1}{8} \times 12.5 + \cdots + \frac{1}{4096} \times 9.6 = 15.9\%,$$

and $1916 * (1 - 0.159) \approx 1617$. Over the whole period the error of exponential smoothing fluctuates from 1 to 10%.

When the increase in call volume over a number of weeks is about equal then it is statistically likely that this will stay this way. In such a case a point estimate of the trend, using one of the techniques discussed in the example, is relatively reliable. However, if the increase with respect to last year shows strong fluctuations, as in the example, then the estimate is usually not very precise.

The weekly profile consists of fractions for each interval. It can be obtained by computing the fraction of calls in every interval over a number of weeks, and then applying some statistical technique.

For some weeks and a particular interval the number of calls, and then average or exponential smoothing.

5.5 Predictable and unpredictable events

The procedure above usually works fine until there is some special event that influences call volume. These events might be a national holiday, some important econmical event, extreme weather conditions, school holiday dates that change from year to year, etc. These events can be classified in different categories: they are either *internal* or *external* to the company, and either *predictable* or *unpredictable*, depending on whether the future occurrence of the event is known by the time the forecast is made. Examples of external predictable events are holidays, implementation of new legislation, etc. Events such as marketing actions that generate calls are internal and (hopefully) predictable by the call center staff. Unpredictable internal events should be avoidable: they are often due to a lack of internal communication. Unpredictable external events are the hardest to deal with. Examples are bad weather generating calls to insurance companies and stock market crashes generating additional traffic to stock trading lines.

These events generate changes in offered load. We can forecast these changes as long as they are predictable. Predictable events are used twice for WFM. First to determine the expected call volume on a specific day or the week, and afterwards to update our knowledge concerning this type of event in order to improve future forecasts.

Unpredictable events need also be entered, although of course this knowledge cannot be used for forecasting these events. It is necessary however to ”filter” them out of the data, if we want that the forecasts represent regular days. The question how to deal with unpredictable events remains. In Chapter 7 we will discuss this issue in detail.

We saw that predictable events can explain part of the offered traffic, while the rest of the expected load is directly extrapolated from the historical data. To estimate the impact of the event historical data is used, but in an implicit way: the effect of the event is estimated using the data. Sometimes it is possible to base the entire estimation on the implicit use of historical data. This is for example the case if we base our estimation on the size of the customer base, and the types of customers. This can give more reliable forecasts: instead of estimating the load due to an increase in number of customers, the forecast can directly be based on the number and the average number of calls per customer. Further refinements can be introduced by diferentiating between new and old customers, the types of products that customers have, etc.

The question remains whether the forecasts are good enough for our purposes, keeping the steepness of the TSF as a function of the load in mind. It is a good habit to compare on a continuous basis the forecasts with the actual call volume. This shows whether the estimates that are generated are reliable or not, and the error that is made in the estimation can be quantified. This should be the starting point of a further analysis into the consequences of this error that should answer the question whether further measures are necessary. Some possible measures to deal with unpredictable deviations from forecasts are discussed in Chapter 7.

5.6 Forecasting in the presence of abandonments and retrials*

Data on numbers of calls and call holding times can usually easily be obtained from ACD reports or from WFM tools. Data on abandonments and retrials are much harder to get, they often require an analysis of the log files of the ACD at call level. These are files that store every event that occurs in a call center. From these files all waiting times can be obtained, including whether these ended with service or abandonment. By analyzing these files over longer periods also retrials can be identified, assuming that the calling numbers are given in the log files (which necessitates ANI).

After having obtained the data we have to estimate our parameters, startign with the average patience. This average patience is not simply the average time abandoned calls spent waiting. As an extreme example, assume that it never occurs that callers have to wait longer than 20 seconds; then no call abandon after more than 20 seconds, and 'thus' a patience of longer than 20 seconds would not exist! Estimating the patience of callers from previous data is a complicated task that requires sophisticated statistical analysis.

The simplest estimator for the average patience of callers over a certain period is as follows:

$$\text{Average patience} = \frac{\text{Sum of waiting times of all calls (abandoned and served)}}{\text{Number of abandoned calls}}.$$

As an example, suppose we had 5 calls, of which 2 abandoned, that spend 15, 20, 30, 60, and 70 seconds in queue. Then the estimated average patience is $(15 + 20 + 30 + 60 + 75)/2 = 100$ seconds. Note that none of the calls waited more than 100 seconds!

If the underlying assumptions are not valid then this estimator can give a considerable error. In this case more advanced statistical methods are necessary.

The hardest part of estimating retrial parameters (the retrial probability and the average time until retrial) is getting the data. The retrial probability is simply the fraction of calls that retry after abandonment, the average time until retrial can be obtained from the data. The retrial probability strongly depends on the type of call center; some studies have shown that the average time until retrial is usually quite short (the next retrial falls often within the ame interval).

5.7 Updating forecasts

Forecasting is done every couple of weeks when the new agent schedules are made. However, this is not the only moment that call volumes are considered and predictions are made. As all planning activities forecasting demands continuous monitoring and updating to react to possible fluctuations in offered load. If updates to forecasts are made early, then better and cheaper solutions can be found, for example by changing agent schedules. This process of regularly updating forecasts continues until the day itself. A recent study showed what most call center planners already knew: during a day, the offered load from interval to interval is positively correlated. This means that if the load is higher than expected in the beginning of the day, then this will probably stay so over the remainder of the day. This has important consequences for the way in which the call center needs to be managed.

5.8 Other forecasting methods

The forecasting methodology discussed up to know is based on historical time seies: on the basis of data of previous time intervals an interpolation to the future is made.

In certain call centers the call volume can also be estimated on current data, such as the number of customers a company has. Then the weekly call volume estimate is computed by multiplying the number of customers by the weekly average number of calls that a customer generates. It is straightforward to generalize this method to multiple customer and/or call types. In a completely analogous way the expected response to for example a direct-mail advertisement campaign can be estimated.

Chapter 6

Staffing

Agent scheduling, or staffing, deals with matching the offered call volume with agents. As such it is an important, and difficult, part of workforce management (WFM). Before going into the details of staffing we discuss WFM.

6.1 Workforce management

Workforce management deals with the optimal use of the main resource in a call center, the agents. For a given period (say a week), it starts a couple of weeks in advance with determining the agent schedules. As input it uses historic call center data on traffic loads and information on agent availability; the output consists of agent schedules.

Thus WFM can be split into several more or less separate steps. The first is forecasting traffic load. The second is determining staffing levels for each interval. After that we have to turn the staffing levels into agent schedules or rosters. Once the schedule is made then over time changes have to be made as additional information comes in changing the underlying assumptions. Here we can think of changing forecasts, agent availability, etc. Finally the day comes that the schedule is executed. During it it might be necessary to take additional measures to ensure that SL and productivity requirements are met. In this chapter we focus on the construction of the initial schedule.

Given the scale and objectives of WFM it would perhaps be better to call it workforce *planning*. Workforce management would then involve, next to workforce planning, also other issues such as longer horizon problems related to hiring and training, and issues that are of a less quantitative nature.

Decision support systems The different steps of WFM are implemented in a magnitude of WFM tools. Around their mathematical cores nice graphical user interfaces (GUI's) are built, adding many possibilities. We will not give web sites, they can easily be found using a search engine or a call center portal. Note however that the functionality of the tools varies enormously. In practice we see that many tools are only partly used, and that specially build tools for forecasting and scheduling are often used. WFM tools are mostly used for getting the data out of the PABX and for determining staffing levels. Other functionality (scheduling, rostering) is less used.

The main reason for this is that every call center is different. Of course, call centers have much in common, but every call center has its particularities which makes that a standard sofware solution does not fit. The choice is taking this for granted and buying a standard tool, or developing tailor-made software. As stated we often see compromises between the two, where standard tools are partly used.

6.2 Objectives of staffing

The construction of agent schedules is a process that often involves a lot of interaction between the planner and the agents, involving many constraints and preferences, some of which are not formalized and implemented in a scheduling tool. For this reason a computer system cannot do the planning completely by itself, but some form of human interaction is necessary. However, decision support systems can generate solutions that are of a high quality and that saves the planner a lot of time.

We concentrate on the way these initial schedules can be generated. The general objective of these scheduling tools is finding the best schedule under a number of constraints. 'Best' usually means cheapest, using as little agents as possible. But it can also be formulated as obtaining the highest SL with a given set of agents. Typical constraints concern the numbers of agents that are required for each interval to handle inbound calls, possibly split up in different skills. There might also be additional constraints concerning the other channels. The second set of constraints concern the possible schedules: which schedules are possible, how many of them can be scheduled, and so forth. Here we encounter an important difference in the way schedules can be made: either we determine a number of standard shifts that have to be fulfilled by the agents, and matching of agents to shifts is done separately, or schedules are right away created for the agents, on the basis of individual constraints. In the former case there is some (often web-based) procedure for agents to choose shift, sometimes in the order of seniority. It might also occur that the assignments are done by the planner, but in some ad hoc manner. The latter method, in which individual assignments are made, makes it possible to differentiate between agent properties and skills. As a consequence, the number of constraints is much bigger.

An intermediate method is making different groups of agents, for example depending on their contract type or skills. Formulated as such, there is no fundamental difference between the scheduling methods, only the number of groups and the number of agents per group vary.

6.3 The standard staffing method*

We give in more detail the objective and constraints of the standard staffing problem of a single-skill inbound call center. Later we discuss disadvantages of this method and possible extensions. Currently we concentrate on a single day.

We assume that point estimates are made of the call volume, and that they are translated into minimum numbers of agents per interval through the use of some call center model such as the Erlang C. On the other hand, we have a number of agent groups, each with a number of possible shifts, and minimum and maximum numbers of agents per group. For each possible shift, we know which intervals an agent doing this shift will be available. There are also costs for each shift. The objective is to find the schedule that minimizes the daily costs while satisfying all contraints. This is a so-called *integer programming problem*, for which powerful software tools exist. One is already freely available at many desktops, the solver included in the spreadsheet package Excel. Other exist as add-ins to Excel, as add-ins to other programming environments, or as stand-alone tools.

The integer programming formulation of this problem is as follows:

$$\min\left\{\sum_{k=1}^{K} c_k x_k \;\middle|\; \begin{array}{ll} \sum_{k=1}^{K} a_{tk}x_k \geq s_t, & t=1,\dots,T \\ l_n \leq \sum_{m\in S_n} x_m \leq u_n, & S_n \subset \{1,\dots,K\},\; n=1,\dots,N \\ x_k \geq 0 \text{ and integer}, & k=1,\dots,K \end{array}\right\}, \tag{6.1}$$

with x_k the decision variable denoting the number of shifts of type k and c_k the costs of shift k, $k=1,\dots,K$. Thus the objective is to minimize the total costs. The first constraint contains the following variables: $a_{tk}=1$ if shift k works in interval t, 0 otherwise; s_t, the number of agents required in interval t. The left-hand side of this equation is the number of agents in interval t, which should be bigger or equal to the number required. The next equation models the fact that a number of shifts has possibly a minimum and a maximum number of agents. This way agent groups with similar contracts can be modeled. The final constraint assures that an integer number of agents is scheduled.

Although this formulation looks quite complicated, it is straightforward for an Operations Research professional to implement it in a spreadsheet or optimization package.

6.4 Disadvantages and generalizations*

The standard staffing method in the previous sections has a number of disadvantages and allows for a number of generalizations that are discussed next.

Schedules for multiple days The method presented schedules a single day. This is fine, as long as one day's schedule is independent of the next. However, often this is not the case: think of a contract in which the number of hours that an agent works per day is to some extent flexible, but the number of hours per week is fixed. Then a schedule should be determined for a whole week at once, preferably using a single optimization model. The model can be extended to account for these weekly schedules.

In call centers that are open during weekends we also find dependencies between weeks, for example to avoid working more than a maximum allowed number of days consecutively. It is possible to make a single staffing model for multiple weeks, although the size of it becomes such that it is doubtful whether IP packages can solve the resulting problem.

Other methods, such as local search, might be a better choice. The alternative is to solve the staffing problem week by week, and to use the solution of one week as input to the next.

On staffing requirements The standard way in call centers to determine staffing is as described above: the required daily or weekly SL is required for every 15 or 30-minute interval. An interesting question is whether this is really required, or whether one allows fluctuations during the day, as long as the average SL (as we learned to calculate it in Section 3.4) is as required.

We will not go into this discussion; instead, we show what can be gained if we only require the daily average. For a simple numerical example, consider only two intervals that have to be scheduled: one with $\lambda = 10$ per minute, $\beta = 1$ minute, and AWT = 20 seconds, and the second interval with $\lambda = 1$ per minute. The difference of a factor 10 is not atypical, sometimes the difference between the busiest and least busy intervals are even bigger. In Table 6.1 we show the results of different staffing levels per interval and over the average of both intervals. In the first line we see the numbers of agents needed to obtain the required TSF (80%) in both intervals, resulting in an overall SL of more than 90%! Because the second interval has a small impact on the overall SL, we see that reducing s in the second interval still keeps the overall SL above 80%. Reducing s in the first interval would lead to a TSF under 80%. Interesting enough, moving one agent from interval 2 to interval 1 improves the overall TSF, as is shown in fourth line. Moving one agent more does not increase the SL. We conclude that letting go of the requirement that the SL requirements should be met for each interval can improve overall SL and reduce costs. As far as we know no WFM tool has implemented this. A simple tool that computes the minimal number of agent hours for a ten-interval period can be found at www.math.vu.nl/~awattel/agents.php.

s in interval 1	s in interval 2	TSF in interval 1	TSF in interval 2	overall TSF
13	3	89.51	95.33	90.04
13	2	89.51	76.12	88.29
13	1	89.51	0	81.37
14	2	95.41	76.12	93.66

Table 6.1: TSF for two intervals and their weighted average using the Erlang C model

Integrating steps There is another reason not to adher strictly to the interval requirements, but to consider only the daily SL requirements. This is the fact that when staffing is done based on a fixed staffing level for each interval, then sometimes considerable overcapacity occurs, because the length of shifts and relatively short peaks in traffic load cannot be matched. Thus the ”best” shift mixture not only satisfies the staffing level at all times, but around peaks it exceeds this level because we cannot hire agents for the peaks only.

A good solution is allowing a low SL for certain intervals, as long as the SL constraint is satisfied on average over the whole day.

More often than a restricted number of shifts with fixed length we see a multitude of different possible shifts with varying lengths. Then there are many different good solutions with different mixtures of shift lengths. Shift lengths are often part of the contracts that agents have. Which mixtures are possible depends therefore strongly on the preferences and contracts of the agents.

The other way around, the decision which type of contract to offer to an agent is an important decision with consequences for the scheduling step, but also consequences when it comes to costs. Small shift lengths make scheduling easier, and avoid unnecessary overcapacity. On the other hand, more agents have to be hired in total in the case of short shifts, and therefore overhead costs (such as training and monitoring costs) are higher. Very long shifts also reduce the efficiency of agents.

When two agents use car pooling to get to work they should have the same shifts. This fact should be taken into account in the shift determination step: one shift, with the proper requirements, should be chosen at least twice. We see that agent preferences, that usually come into play while making rosters, already play a role in the shift determination step. This calls for an integration of both steps. Also if the roster requirements are highly personalized then an integration of scheduling and rostering is called for.

But in general it is again a complicated task, in which we often cannot focus on a single day, as contracts often specify the weekly number of working hours. Again, mathematical solvers are excellent tools to find feasible rosters.

We saw that, except for forecasting, there are sometimes good reasons to integrate all WFM steps. A drawback is the complexity of the resulting integrated problem. However, computers and mathematical software are nowadays powerful enough to handle also these problems.

6.5 Workforce planning

On a longer time scale we have to consider how many agents to hire and when. Usually it takes a few months between the decision to hire agents and the moment they are operational, due to the hiring process and the training. At the moment the decision about the number to hire has to be taken it is unknown how many agents will have left by the time the agents become operational.

Suppose we have a call center with 500 agents. We have to hire agents right now, which will be operational after three months. We then need a total of 520 agents. Agent turn-over is estimated at 10% on average over a three-month period. How many agents to hire?

The obvious answer looks to be 70: if we do nothing then three months later 10% of the 500 agents will have left, leaving 450, thus 70 agents are needed to reach 520. However, in about 20% of the cases you need more than 75 agents, and in about 20% you need less than 65 agents. (This can for example be calculated using the Excel BINOMDIST function.)

We conclude that fluctuation occur and they cannot be predicted timely enough. Thus, unless other measures are taken, management should decide on which policy to follow: a service-oriented policy in which more agents than needed on average are hired, or a cost-oriented policy in which less agents are hired.

Chapter 7

Variations, uncertainty, and flexibility

In Chapters 4 to 6 the mathematical background is given for basic call center management. In this and the next chapters we discuss more advanced topics. This does not mean that they are less relevant: many call centers form a complicated environment that demands knowledge of the topics discussed here. We start with an in-depth discussion of the consequences of uncertainty for call centers.

7.1 Variations and the need for overcapacity

Every call center manager tries to combine a high service level with a high productivity. In the previous chapter we saw why this is not always possible, due to unavoidable variations in call holding times and interarrival intervals. The Erlang formula quantifies the influences of these variations, and shows what the variations cost in terms of additional personnel.

Consider a call center with $\lambda = 4$ and $\beta = 5$. The offered load is therefore 20 Erlang, and without any variations 20 agents would suffice to obtain a 100% SL. However, the Erlang formula shows that, under the usual variations, we need 5 additional agents to obtain a 20/80 SL, thus 25% overcapacity.

In the previous chapter we saw that increasing the scale smooths out these short-term unpredictable variations. Indeed, doubling the number of offered calls in the example reduces the needed overcapacity from 25% to 15%, as can easily be verified using the Erlang calculator at www.math.vu.nl/~koole/ccmath/ErlangC. Unfortunately, increasing the scale is not always possible, and even in large call centers some overcapacity is needed. Additionally, there are other types of variations that at first sight demand additional overcapacity.

Consider the average number of incoming calls λ. This number is the outcome of the forecasting procedure, and therefore an *estimation* of the real value. What if the estimation is 10% too low, that is, if λ is 4.4 instead of 4? In the example it leads to a 65% SL in the case of 25 agents. Now consider the case where the estimated offered load is twice as high, thus equal to 8. Again, we assume that the number of agents is such that a 80/20 SL can be obtained assuming that $\lambda = 8$. However, the SL under a 10% load increase is now only 40%! What we observed holds in general: the bigger the call center, the more important the consequences of load changes, the less *robust* the call center is. This is the flip side of the economies of scale.

There are more uncertain elements in call centers than just the offered load. Consider agent availability. Even when a constant number of agents is scheduled, then we see a variation in agent availability due to illness. Also meetings and other obligations outside the call center can lead to a reduced availability, especially when they are not planned but if they are part of the shrinkage. To assure the SL also under these circumstances we need to schedule additional overcapacity. E.g., we need two additional agents in the call center of the example to be able to cope with 10% illness. But what if not exactly two agents are absent? Again, we only know the average absence percentage, and the moment we make the agent schedule we cannot (completely) predict who will be available or not. At first sight there is not much more to do than to schedule two additional agents, hoping that no more than two agents will be ill.

Having scheduled this costly overcapacity, the following question remains: what to do if there is a 10% decrease in offered traffic? What if all scheduled agents are indeed available? In the example, a 10% decrease in offered load would require 23 agents, 2 less than the normal situation, and 4 less if we anticipated a 10% increase in traffic! In the next sections we discuss ways to deal with these unpredictable variations.

7.2 Averages versus distributions

The essence of entities such as the arrival rate and absence percentage is that they show fluctuations that cannot be predicted timely and entirely. When a heavy storm damages many houses it is too late for an insurance company to change the weekly schedule. The same holds when in the morning a larger than usual number of agents appears to be ill. And even for predictable events such as holidays, it is sometimes hard to estimate accurately their influence on the load.

It is well possible to estimate averages. This is what forecasting is about, and every call center manager can tell the average absence percentage. However, we also know that fluctuations around this average occur frequently: now and then fewer than the average number of agents are ill, and then again more than average. Similarly, in many call centers we also see that the offered load cannot be predicted accurately, no matter how much effort is put into it. Instead, we should accept that fluctuations around the average occur. Now the attention shifts to *quantifying* these fluctuations, and reacting accordingly.

The usual measure for the size of fluctuations is known from statistics as the variation. Based on the average and the variation a bell-shaped curve can be constructed representing the frequency table of for example the offered load. Under specific assumptions other approaches can be more appropriate. For example, if we assume that the illness of an agent has no relation with the illness of other agents, then it suffices to known the probability that an agent is ill. Given this it is straightforward mathematics to compute the fraction of days that a certain number of agents is ill. This can again be plotted in a frequency table. Such a frequency table is also known as a "distribution".

7.3 The need for flexibility

Having quantified the variability, we now have to take the appropriate measures. Our goal is to come to an acceptable cost-SL trade-off, in the presence of additional variations with which we are confronted after having scheduled the agents. Central are flexibility in the use of the workforce and reduction of the influence of variability. We start with the first.

By introducing flexibility at all time levels of the operation we can offer the required SL while keeping a high productivity at the same time. At the longest-term level we have flexibility in contracts. With this we mean that for certain agents we can decide on a very short notice (e.g., at the beginning of the day) whether we require them to work or not. Of course they get paid for being available, and often they are guaranteed a minimum number of working hours per week. This is an excellent solution to deal with variability in arrival rate and absence. For the latter this is obvious; for the former we have to realize that the arrival rate during the first hours of the day often gives a good indication of the load during the rest of the day. Thus early in the morning it can already be decided whether additional agents are needed.

When trying to quantify this, we start with a minimum number of fixed contract agent. This minimum is based on some lower bound on the arrival rate and a minimal absence. Then we assure that there are enough agents with flexible contracts such that we can get the number of agents equal to the number required in the case of a maximal arrival rate and maximal absence.

A call center has an arrival rate falls between 4 and 4.8, with 90% probability. For the lower bound 50 agents are needed, for the upper bound 9 more. Out of these 50 agents between 1 and 6 agents are absent, on average 3. Thus we schedule at least 51 agents, and in the "worst" case we have to hire 14 more, on average 6.

Introducing flexible contracts gives us the possibility to handle days with a higher than usual traffic load. If the peaks are shorter, in the order of an hour, then we cannot require agents to come just for this short period of time. Sometimes it is possible to mobilize extra workforce by having personnel from outside the office work into the call center.

Consider a bank with a stock-trading line. Waiting on the telephone can lead to huge costs for the clients in case of a stock crash, the value of the stock can go down considerably while the callers are waiting. Therefore it is absolutely necessary that the call center assures its SL even in case of a peak in offered traffic. They assure this by having people not being part of the call center on standby. They are mobilized in case of a peak in offered load.

Although this seems a simple solution for emergency cases, one should realize that the extra agents should be trained and that the telephony and IT equipment should be in place to accommodate all agents.

A simpler form of flexibility is work in overtime. The possibility and price of this depends on the contracts and union agreements, but this is often a relatively simple way to react to high workloads.

7.4 Multiple channels

A final type of flexibility is flexibility in task assignment. This is a method to react to load fluctuations that can even work at the finest level of fluctuations, that the Erlang formula accounts for. For this it is necessary that there are, next to the incoming calls, other tasks that have less strict service requirements. Examples are outgoing calls and faxes. Emails and messages entered on web pages fall into the same category. Nowadays they constitute a considerable part of the load in contact centers. They have service requirements that range from hours to days, thus of a totally different scale than the requirements of incoming calls, which are in the order of seconds. To be able to satisfy the service requirements for these so-called channels it suffices to schedule just enough agents to do the work. Scheduling overcapacity, as for incoming calls, is not necessary: here averages apply, we do not have to account for short-term peaks in offered load. It doesn't matter when outgoing calls or emails are handled, as long as they are handled in the required time interval. This makes it possible to use outgoing calls to fill in the gaps left by a low offered load, and allows in case of undercapacity agents originally scheduled for emails or outgoing calls to work on incoming calls. Thus instead of assigning in a fixed way agents to ingoing or outgoing calls, they are assigned dynamically (either by the supervisor or automatically) to a certain channel. This assignment should be done carefully. A free agent should obviously be assigned to a waiting incoming call if any are present. A way to maximize productivity is by assigning free agents to outgoing calls if there are no waiting incoming calls. However, then every incoming call has to wait for a free agent. In most situations this will lead to a very low SL. The solution is to keep a number of agents free for incoming calls when none are waiting. This rule works when changing from ingoing to outgoing calls takes relatively little time. It is known as call blending, as it was originally intended for call center dealing with inbound and outbound traffic. Simply *blending* seems a more appropriate name given the recent focus on communication over the internet. To experiment with blending a special calculator can be found at www.math.vu.nl/~koole/ccmath/blending. In the next example it is numerically illustrated how blending can improve productivity while keeping the SL acceptable.

A call center has $\lambda = 5$ per minute and $\beta = 5$ minutes. 30 Agents are dedicated to inbound calls to obtain a 80/20 SL, while 5 agents are working on outbound calls. The same performance can be obtained with 32 agents is blending is used, with the threshold, the number of agents held free for inbound calls, equal to 5. Thus blending saves almost 10% on the workforce.

A disadvantage of blending is that, in the case of a high load, the other channels do not meet their SL requirements anymore. This might even lead to additional incoming calls from people complaining that their emails are not answered! Overtime can be a good solution, certainly if it concerns outbound calls: later at the day people tend to be better reachable, thereby increasing the productivity.

Blending looks really good on paper, but in practice there are many disadvantages. Perhaps the most important is the loss of efficiency from an agent that regularly has to switch between types of tasks. For this reason it can be better to implement intermediate solutions, based on the general idea of blending. One such a solution is assigning some agents to outbound calls, and having them join the inbound group when the queue exceeds a certain level.

7.5 Reducing the impact of variability

In the previous sections we discussed ways to deal with variations by introducing flexibility in agent availability and task assignment. A different approach to dealing with variations is by reducing them, or by reducing their impact. Consider the following example.

To make reservations for international travel the Dutch railways have two options that can be done from your home. The first is calling the contact center on a 0900-number, meaning that the caller pays for the call. The second is entering your travel data and the moment at which you want to be called back (a 4 hour interval) on a web page. Potential travellers are thus financially stimulated to enter their data on the web page, thereby turning an inbound call into an outbound call. This allows the contact center to contact you at some quiet moment during your preferred time interval. Often the call takes little time as the agent already known the travel options, based on the data that you entered.

The example clearly shows the advantage of outbound or email contact over inbound calls. Instead of having to answer within 20 seconds after the arrival of a call, you can take the moment in a long interval that suits you best. In general, the same amount of work is done with less agents and at a higher SL. This is a direct consequence of the fact that outbound calls have a less strict service level requirement. Then we assume of course that call blending is being used, that agents are not assigned in a fixed way to either incoming or outgoing calls.

Another way to turn inbound into outbound that is especially effective in reducing peak loads is by offering callers a call-back service: their telephone numbers are registered and they are promised to be called back as soon as possible. Take the following example.

A manager of a free 0800 service is complaining, and with reason. Her SL is lousy and her telecom costs are going over the top. Due to the bad SL customers abandon: the call center is paying the telecom costs of customers that not even reach the call center! The bad SL is the result of increased customer attention that went too fast for the call center to cope with by hiring new agents. Thus the increase in customers does not lead to an increase in income, only to an increase in costs!

The answer to this kind of situation is limiting the number of callers that can wait simultaneously in queue. This can be done by asking people to call back or by asking them to leave their number so that they can be called back. This way callers that get no service do not wait in queue. Not only the costs are reduced (in case the call center pays for the communication), it is also customer friendly: you allow only those customers to enter for which you can offer a reasonable SL. Certainly if the offered load is high then there is no sense in making callers wait, there will always be new callers for free agents to handle. Calculating how to set the number of lines that can be used simultaneously requires extending the Erlang formula. This is the subject of Chapter 8.

7.6 Implications to outsourcing

Many companies use outsourcing to specialized companies to deal with at least part of the incoming calls. An issue that is always part of the contract between outsourcing company and outsourcer is the average volume that has to be dealt with by the outsourcer. A question that is much less addressed explicitly is how to split the variability in volume. In practice we see all possibilities.

The most often occuring situation is where the outsourcer has no variability in offered load: the assigment of incoming calls is adapted to the number of available agents. The only obligation for the outsourcer is to have the required number of agents in the call center, a number that has been set (between certain bounds) a few days in advance by the outsourcing client. In this situation the client has complete control over the traffic, but if it needs to react to short-term fluctuations it has to do this within its own call center. Thus the variability is higher, relatively to the volume treated in the client's call center. Evidently this is the cheapest solution to outsourcing.

The second situation is where a fixed daily volume for the outsourcer is determined. Again, this is done a number of days in advance, and again, within certain pre-fixed bounds. Together with that service level agreements have to be made, and penalties have to be negociated in case these are not met. Now the outsourcer has to do workforce management, but without forecasting. It can start right away with using the Erlang formula with the negociated volume as input. Forecasting is still being done by the outsourcing company, who has to do the whole wfm cycle because it has to operate an inhouse call that has to deal with the unpredictable variations in volume. This type of contract will be more expensive because of the increased responsabilities of the outsourcer, without offering many additional advantages compared to the seats occupied-based contract.

The final situation is appropriate in case the call center is completely outsourced. The outsourcer is made responsible for the complete wfm cycle, including the forecasting. The contract should allow for fluctuations in volume.

Chapter 8

Extensions to the Erlang C model

In the previous chapter we discussed several measures to improve the performance of the call center. To quantify the impact of these improvements with respect to the standard situation, represented by the Erlang C formula, we have to extend the underlying Erlang model. We also saw that the Erlang C needed improvements to better model reality: abandonments is a good example. We indicate all these extensions with Erlang X, from eXtension. This Erlang X model is the subject of this chapter.

8.1 Blocking

Theoretically speaking, the Erlang C model allows an unbounded number of queued customers. Not only will this never occur in practice because callers abandon, it is also impossible because the number of lines available to connect to a call center is always limited. Thus blocking can never be completely ignored. As we saw in the previous chapter it can even be advantageous to block customers when there are still lines available: this increases the blocking percentage, but decreases the waiting times of customers that are admitted. Unfortunately, it also decreases productivity.

To determine the best number of lines (or, equivalently, the maximum number of customers in the system) we have to calculate productivity, blocking percentage and waiting times for various numbers of lines. This allows us to see the trade-off between the three and make a justifiable choice. To calculate productivity and waiting times we have to make certain assumption about customer behavior in order to build the mathematical model. An important choice is related to the behavior of callers that are blocked. Either they are lost, they try to call again later, or they are called back as soon as the load permits. Each of these choices requires a different model and leads to a different performance.

Consider a call center with $\lambda = 10$, $\beta = 5$, and $s = 51$. Then the SL is equal to 21%, and the ASA is more than 4 minutes. Now assume that there are 58 lines, thus only 7 waiting places. Then SL = 80% and 1 out of 20 calls gets blocked. Note that we assume that blocked calls do not lead (immediately) to retrials.

In a fully occupied system on average every s/β time units a call is finished. If a call is nth in line, then its waiting time is the time until the nth service completion, which takes on average ns/β time units. For a big call center (or a long queue) this number is quite exact. This can be used to choose the number of lines. For instance, in the numerical example above, a call is finished every 6 seconds. If you prefer to block calls instead of letting them wait 1 minute or more, then we should have no more than 9 waiting places to avoid calls becoming 10th in line and having to wait 60 seconds. By taking 60 seconds as AWT we can indeed verify that less than 4% of the accepted calls spends more than 60 seconds in the queue.

In the above it became clear that blocking is an excellent way to protect callers against waiting too long. (Note also that customers who wait long tend to have longer call holding times!) An alternative is informing customers about long waiting times and advising them to call back another time.

8.2 Abandonments

As soon as waiting occurs it is inevitable that callers abandon. Some callers abandon as soon as they enter the queue (sometimes called balking); most abandon after having passed some time in the queue. Determining the patience of callers, i.e., the time that they accept to spend in the queue, is, mathematically speaking, a difficult task because (hopefully) most callers reach an agent before their patience is over; see page 32. After obtaining the average patience we can include abandonments in our analysis. Under certain statistical assumptions concerning the time until abandonments the SL calculations can be done. These assumptions boil down to two things:
- callers abandon the moment they are queued with a certain probability;
- callers in queue abandon within the next second with a probability that does not depend on the time they have already spent waiting.
Naturally, when doing numerical experiments we see the same behavior as in reality: abandonments decrease productivity somewhat (as less callers reach an agent), but also the waiting times decrease because some calls leave the queue. These experimentations can be done at our Erlang X calculator at www.math.vu.nl/~koole/ccmath/ErlangX.

An interesting subject is the psychology of abandonment and the ways to influence it. We measure the time at which people abandon, which we call their patience. It suggests that people base their decision only on the time spent in queue. This however is doubtful: why do people abandon at entering the queue? They have no patience at all? It also suggests that the caller's behavior cannot be influenced by given additional information such as expected waiting time, which is not realistic. Another way to explain abandonments is based on the idea that people make their decision on the time that they expect that they still have to wait. It explains that some callers abandon at entering the queue: they expect that their waiting time surpasses the time that they accept to wait. At first sight it does not explain why people abandon while waiting in queue: as they wait the remaining waiting time decreases, so why abandon? The reason is that they did not know their own waiting time, while waiting callers learn about their own waiting time. Surprisingly enough

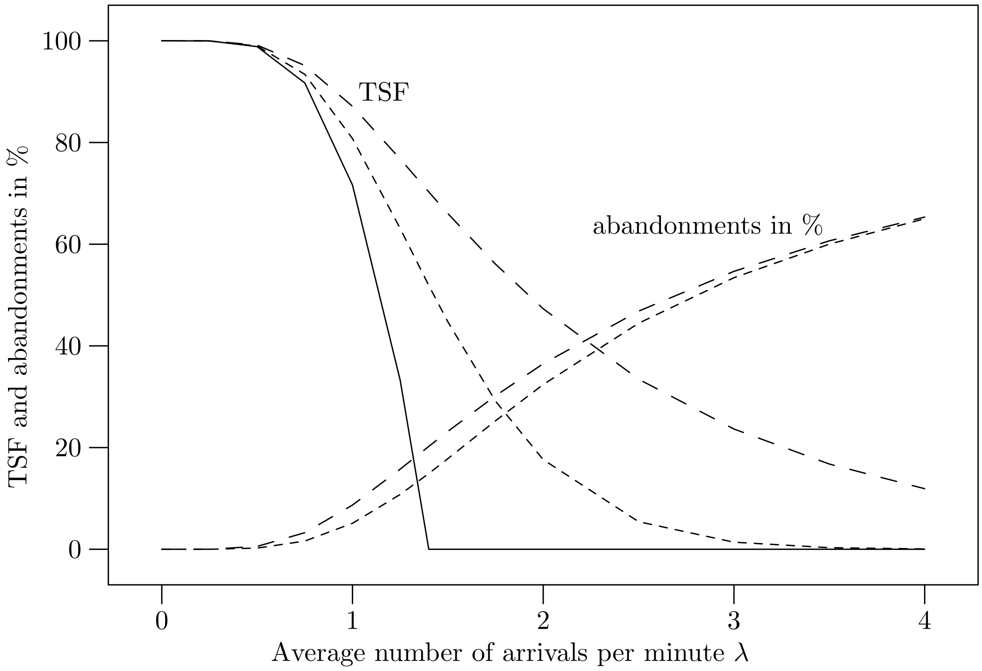

Figure 8.1: TSF and abandonment percentage for average patience ∞, 5 and 1 (from below), for $\beta = 5$, $s = 7$, AWT = 0.33, and varying λ.

it can be shown mathematically that the remaining waiting time in a standard call center, modeled by the Erlang C formula, stays always the same, no matter how long a caller has waited already. So, mathematically speaking, one would say that the rational caller has no reason to abandon while waiting. However, a waiting caller does not only learn about his own waiting time, he also learns about the situation the call center was in the moment he or she arrived. In other words: if you have to wait long, then you are probably calling to a badly managed call centers meaning that you probably still have to wait for a long time. And this is a good reason to abandon while waiting for some time.

In some call centers there is a direct connection between each call and the profit a company makes: a call means on average a certain income to the firm. It is hard to imagine a company with a business model that simple (even a mail order firm likes happy customers that phone back again), but it is an interesting exercise to pursue it. Using the Erlang model extended with abandonments we see that every additional agents increase less the productivity, and therefore a break-even point is reached at some number of agents. What the corresponding service level is depends of course strongly on the parameters.

8.3 Redials*

Consider the following numerical example.

A call center has $\lambda = 10$, $\beta = 5$, $s = 51$, *average patience 2, and* $AWT = 0.33$, *and all patience whos patience is exceeded abandon. Then* $SL = 83\%$, *as can be verified with the*

Erlang X calculator. About 6% of the calls abandon. Now assume that all these abandoned calls become retrials. Thus λ increases, and the increase should correspond to the number of abandoned calls. This is the case if we take $\lambda = 12$: the abandonment percentage is 17, and 17% of 12 is around 2. Now the SL is 51%.

In the example we saw that with 6% abandonments we had a great SL, but as soon as these abandonments became redials this SL decreased to around 50%. However, suppose that there were no retrials, and that all customers had infinite patience. Then the SL would be only 16%! Thus in the case with retrials, we see that waiting times are shorter due to calls leaving the queue. The price to pay is that a considerable portion of the customers will retry, some even multiple times.

8.4 Overload situations*

In Section 3.4 we saw how to calculate the service level using the Erlang formula by averaging the service levels of each interval. This approximation is called the *pointwise stationary approximation* (PSA), because it does not take into account the transitions from one interval to another. This is not problematic, as long as the load a is smaller than the number of servers s, in other words, if the agents can, on average, handle the traffic. If this is not the case for one or more intervals, then the PSA will give wrong results. Over the interval(s) where undercapacity occurs the Erlang formula will give SL 0%, which is often not the case: the SL deteriorates over the interval, but it is not 0%. On the other hand, as soon as the parameters change and we find ourselves again in a situation of overcapacity, then the Erlang formula predicts right away the SL belonging to these parameters, not taking into account the backlog of customers from the previous interval. This leads to a SL prediction that is too high for this interval. It is tempting to state that both errors will cancel out, but unfortunately there is no ground to assume that this is always the case. Alternative methods exist, but they are mathematically quite involved. An alternative approach is simulation, see page 67. But instead of pursuing this way we can also ask ourselves the question whether this is useful, especially because the above analysis is based on the Erlang C model. In reality callers abandon, which reduces the long-term correlation and makes that the backlog at any time remains limited. This calls for the PSA with the Erlang X system as performance model. However, then we overlook one issue: instead of waiting indefinitely in the Erlang C calls abandon, but they retry. This means that overload situations result in an increase in redials and thus also in load in consecutive intervals. Thus through the redial rate consecutive intervals are linked. How to model this exactly is a subject of scientific research.

Chapter 9

Multiple skills

When call centers increase in size, we speak of an increase in scale. Call centers can also increase in the number of tasks that they execute: multiple skills is the rule, not the exception. The obtained advantages are called economies of *scope*. How to get the maximum out of multi-skill call centers is the subject of this chapter.

9.1 The possible gains

When a call center grows in size or in the number of different types of tasks then the moment comes that management considers having agents specialized in only a subset of all tasks. There are several reasons for this: specialized agents are more efficient, need less training, and management becomes easier in certain aspects. But there is a price to pay. Specialized agents are less flexibility. Next to that, when a large call center is split up in multiple smaller, one for each skill, then we lack the economies of scale, as discussed on page 21, in the context of the Erlang C formula. Let us illustrate this numerically (see also the example on page 22).

Consider a call center with 2 skills or types of tasks and the regular 80/20 SL. The call handling time is $\beta = 5$. For $\lambda = 1$ for each skill we need, according to the Erlang C formula, 8 specialists, thus 16 in total. If we have only generalists then $\lambda = 2$ and filling in the Erlang C calculator tells us that we need 14 generalists. Thus cross-training agents saves two agents, which is an decrease in number of agents of $\frac{2}{16} \times 100 = 12.5\%$. For $\lambda = 5$ and 10 the required numbers of generalists and specialists are 57 and 60 respectively, an increase of only $\frac{3}{60} \times 100 = 5\%$.

Thus we see that the advantages of scale increase with the size of the call center (in the example from 2 to 3 agents less needed), but relative to the total number of agents they decrease (from 12.5% to 5%). What counts most depends on the call center. This calculation however, is under the assumption that there are no efficiency gains from specialization. This is an unrealistic assumption. In general specialized agents are more efficient than generalists. For this reason we should be careful with utilizing cross-trained agents for multiple types of tasks at the same time: the increase in productivity due to the increase in scale might not counterbalance the efficiency loss due to the lack of specialization. Because the relative gain of scale decreases with the scale, a full separation between skills might be interesting for big call centers. Let us illustrate the loss of efficiency due to a lack of specialization in an example.

Consider again a call center with 2 skills or types of tasks. Specialists of either skill still have $\beta = 5$*, but generalists have* $\beta = 5.5$ *(because they have to switch skill regularly). For* $\lambda = 2$ *we now need 15 generalists, thus cross-training agents saves only one agent. For* $\lambda = 5$ *we need in total 60 specialists. If we work with generalists then we need more agents: 62. Thus working only with specialists requires less agents in this situation, due to the efficiency of specialists.*

Up to now we only considered the advantages of scale related to multi-skill environments as compared to a complete separation of skills. An intermediate solution is often best, as we will see later in this chapter. A relatively small percentage of multi-skill agents gives the bigger part of the economies of scale.

However, cross-trained agents are not only useful in obtaining the economies of scale. They are also more flexible then specialists. Quite often this increase in flexibility is more important than possible gains from the increase in scale: when for example the loads fluctuate then generalists can be used for the skill for which they are needed most. This does not necessarily mean that the task assignment is done at the routing level, as in a real multi-skill environment, as this might lead to an efficiency loss. In many cases it is better to assign a generalist to a certain skill for a longer period of time. In case the load is well predictable then this can be part of the schedule, when the load has unpredictable fluctuations then the assignment should be a consequence of performance monitoring.

Let us consider again the numerical example above for $\lambda = 5$*. However, this estimation appears to be 10% off: one skill has* $\lambda = 5.5$*, the other 4.5. In the case of 30 specialists for each skill we have SLs of 54 and 94%: we lack flexibility. In the case of 62 generalists (and* $\beta = 5.5$*!) the SL remains 83%. However, if we have enough generalists to have 27 agents working on the low load class and 33 on the high load class, then again 60 agents is enough to obtain a 80% SL for each class.*

In conclusion, we find that a complete separation of skills leads to losses in flexibility and scale, but that our agents work most efficiently. Thus this might be a good solution under the following conditions: Few fluctuations (and thus no need for flexibility), and all separate skills already constitute big call centers (thus there are relatively little gains from an increase in scale). Having only generalists is only a good idea for very small call centers. The intermediate solution with specialists and generalists and some form of flexible task assignment is usually the best solution. Assigning tasks at the call level, called *skill-based routing*, or SBR), is good if there are no efficiency losses from multiple skills or if the loss in productivity is counterbalanced by the economies of scale. The latter can be the case if the load per skill is relatively small.

In this chapter we concentrate not only on (short-time) skill-based routing, but also on quantifying the need for flexible agents in situations where SBR is not used.

9.2 Multi-skill basics

Introducing multiple skills can only be done if the ACD can differentiate between the skills. There are different ways to obtain this differentiation. One way is installing a VRU (voice response unit) where callers have to choose; another way is communicating different numbers to the clients, depending on the required skill. Sometimes the PABX recognizes the calling number and differentiates between callers, for example between premium and regular customers, as to give the first group a better SL. The result is that there are different queues at the ACD for the different skills.

The term skill suggests that different agents can handle different skills. This is only partly true. In the first place, it is not uncommon for agents to be able to handle more than one skill. E.g., in a multi-lingual call center an agent might speak more than one language. However, often agents have a preferred skill, one in which they are best. Agents that have all the skills are called *generalists*; agents with only one skill are called *specialists*. Agents with more than one skill are said to be *cross-trained* (or *x-trained*). Although it often occurs that many agents have only one skill, we assume that this does not hold for all agents, and that some of these agents are ready to receive calls that require different skills.

Agents having the same set of skills belong to the same *skill group*. There can be many different skill groups. For example, for 3 skills, there are 7 different skill groups possible, with skills 1, 2, 3, 1/2, 1/3, 2/3, and 1/2/3. For 10 skills we have 1023 different possible skill groups!

In practice it is sometimes considered that the call types and the skills are not the same. For example, a call center might have high and low-value customers queueing at different lines, and two skills: for handling low-value only or for handling both. In what follows we assume that the call types coincide with the skills. This is no restriction: in the example we take as skill groups low-value and low-value/high-value and we have the situation as described above where call types and skills match.

Multi-skill call centers pose some challenging problems to the manager and the mathematician. All the problems of single-skill call centers, as discussed in earlier chapters, are also present in multi-skill call centers, but most of them become more complex. There are also problems that are unique to multi-skill call centers. Starting at the smallest time scale, we have the problem of how to execute skill-based routing. This problem is extremely complex, certainly in the case of many skills and many different groups of agents having different skill sets. The problem is to route calls in an optimal way, not only based on who is currently available, but also anticipating possible future arriving calls. This call routing problem has to be solved automatically thousands of times per day in many multi-skill call centers, and a really satisfying solution to this problem does not yet exist. However, several special cases have properties that make optimal solutions possible, and also for the general problem solution methods that perform reasonably well exist.

At a longer time scale we encounter the problem of scheduling agents. This also becomes extremely complex, due to the increase in possibilities that we have: which mixture of skills has the best cost-performance trade-off? And is this a feasible mixture in terms of the availability of agents? This holds both for the situation in which SBR is used, and in which multi-skill agents are always assigned to a single skill, but not always the same.

At a longer time scale, at the tactical level, the problem of hiring the right number of agents becomes one of hiring and training the right number of agents. Of course, also in a single-skill call center training is needed, but now we have the additional question which skill(s) to train. Learning new skills enables call center agents to have career paths in which one progressively acquires new skills. This is not always possible: in a multi-lingual call centers for example agents with the right language skills have to be hired, call centers usually cannot afford it to teach agents a language at the call center's expense. Planning education and hiring, while taking turn-over and fluctuating demand into account, is a challenging task.

Finally, having multiple skills makes the organization of the call center more complex. Management overhead should always be taken into account when making changes in the overall structure. In the next sections we deal with the problems of multi-skill call centers, starting from the short time scale.

9.3 Simulating a two-skill call center

On page 52 we studied a two-skill call center with generalists and/or specialists. Using the Erlang C formula (or one of its extensions) we could study the cases with only generalists or only specialists. However, our conclusion was that a mix might be the best. To evaluate these kind of complicated situations we cannot use the Erlang formulas anymore, instead we have to rely on mathematically involved and long computations, on approximations, or on simulation. In this section we discuss simulation in the context of call centers with multiple skills. For more background on simulation see page 67.

A simulation tool is available at `www.math.vu.nl/~koole/ccmath/sim2skill`. Its purpose is to illustrate aspects for multiple skills but for simplicity it is restricted to just two skills. Let us discuss the input. In contrast with some other tools you cannot choose the time unit per input parameter, we use minutes everywhere. Most input parameters will be clear. Entering a big number at "Average time until abandonment" approximates the case with infinite patience. "Waiting time factor" and "reservations" will be explained later, and can be set to 1 and 0, respectively. Now run example S1 at the website. We find the same answer as earlier computed with the Erlang C formula. However, now we can also test a combination of specialists and generalists. Some experimentation leads to the result that 58 agents is sufficient, for example 22 specialists for each skill and 14 generalists. Not that there are approximately 20% generalists, a rule of thumb found in the scientific literature.

To get used to the simulation tool and to quantify the advantages of having both specialists and generalists we analyzed up to now a simple symmetric situation (i.e., the skills have the same parameter values). Now we move to more complex asymmetric situations in which intelligent decisions concerning routing are required, thus skill-based routing.

9.4 Skill-based routing

The complexity of the routing problem in multi-skill call centers can differ enormously. Sometimes we see only a few skills (e.g., different lines for B2C and B2B), sometimes we see tens of different skills (e.g., a call center that has different product skills and language skills). Obviously, the routing problem is more difficult when there are more skills. However, optimal routing can be complex even when there are only a few skills. Next to that it is often the case that different skills have different SL requirements.

In skill-based routing there are two types of decisions. The first has to be taken when an agent becomes available, and there are calls of multiple types that can be dealt with by this agent. The second is when a call arrives and there are multiple agents available with different skill sets that contain the type of the call. To illustrate the complexities of skill-based routing we start with a simple example which involves a situation where only the first type of decision plays a role.

Consider a call center with two types of calls, B2B and B2C. There are two skill groups, for handling only B2C and for both (if an agent can handle B2B then (s)he can also handle B2C). Now consider the case where we have no assignment rule, we just allocate calls to a free agent with the right skill in order of arrival. Thus when a B2C specialist becomes available a queued customer call is assigned; when a generalist becomes available then simply the longest waiting call is assigned. In this situation consumer calls have more possible agents which can serve them, and therefore they get a better SL, contrary to what probably the cc objective is! This can be verified using the simulation tool available at `www.math.vu.nl/~koole/ccmath/sim2skill`*, for an example use the parameter values of example S2.*

From the example we learnt that simply assigning the longest waiting call with the right skill to an agent that becomes free can have undesirable effects, unless we are only interested in the average SL (see the discussion on page 11), and even then it is not always the case that simply serving the oldest call gives the highest average SL.

Take an example with only generalists and overload. The TSF is low for both groups. Not serving one group at all (by making the generalists specialists) gives a higher TSF for this group, and in many cases also a higher average TSF. (For example values, try example S3, and then move the generalists to one of the skills.)

The counterintuitive and unwanted effect from the example is caused by our definition of SL through the TSF: we do not care about calls that waited longer than the AWT. Ignoring one group of calls is not better if we take the AET as SL measure, which is a reason why the AET is preferable in certain situations (see page 13).

Assume that we are in a situation where we have a SL constraint on each call type, and that this constraint is not satisfied for one of the types. How can we change the assignment of calls to agents such that the SL of this call type increases?

A possible solution is making a third skill group with generalists that only take B2B calls. This however makes them less flexible. This can be an attractive solution if generalists are less efficient than specialists due to the multi-skilling. Let us assume that this not the case. Then it is not an attractive solution: it might occur that there are few B2B and many B2C calls, then the free agents in the B2B skill group cannot be assigned to the B2C calls. Better solutions keep the two skill groups, but reserve enough capacity for the B2B calls by either giving priority to them, by reserving capacity in a flexible way, or by doing both. How this can be done is discussed next. We start with discussing priorities.

At first sight we have two different choices: giving priority to B2B calls over B2C or giving no priority, i.e., first-come-first-served (FCFS). However, we can also see these as the two extremes of selection policy that uses *weights*. That is, within a skill we serve FCFS, but if we have a choice between two types we compare the waiting times of the longest waiting calls after we have multiplied them by a number that is specific for each call type, the *waiting time factor* (WTF). We select the call that has the highest product of waiting time and waiting time factor.

Take as waiting time factors 2 for B2B calls and 1 for B2C calls. Assume that the longest waiting calls have waited 5 for B2B and 15 for B2C. Then, because $2 \times 5 < 1 \times 15$*, the B2C calls is served first. If we would have looked 10 seconds later, then the B2B would have been served first because* $2 \times 15 > 1 \times 25$*.*

Note that if the waiting time factors are equal then FCFS is used for all calls; if the WTF is equal to 0 for a skill then this skill has the lowest priority, and calls of this type are only served if there are no high-priority calls waiting. The advantage of using this way to increase the SL of a skill compared to reserving agents for the skill is that the agents are used in the same flexible way, for the same skills as in the FCFS case.

Go back to example S2, but take waiting time factors 5 and 1. Now the SL for type 1 increases from 60 to around 70%. Note also how small the AET gets. Full priority to skill 1 (waiting time factor 0 for skill 2) leads to a SL of 77%.

It can occur however that even giving full priority to a skill does not lead to the desired SL for that skill. In that case we have to reserve capacity for the high-priority skill, meaning that agents are kept idle while low-priority calls are queued, waiting for high-priority calls. As we said earlier, this should not be done by reserving generalists to a single skill, but in a more flexible way, by keeping up to a maximum number of generalists idle when there only skill-2 calls waiting. We call this a threshold or reservation level.

Suppose the reservation level for skill 1 is 5. Then skill-2 calls can only be assigned to agents when there are more than 5 agents available, waiting for a call. Thus as soon as the 6th agent becomes available then a skill-2 call can be assigned. Skill-1 calls are assigned even when few agents are available.

By varying the reservation level all possibilities between full multi-skilled and full single-skilled can be obtained.

Continue with example S2, with full priority to skill 1. By setting "reservation skill 1" to 1 the SL for skill 1 already exceeds 80%.

The situation gets considerably more complex if we assume that multi-skilled agents work less efficiently. Then it pays to assign generalists to a single skill, and it is less clear which combination of measures is optimal.

If we set the service times of generalists in example S2 equal to 5.5, then we need full priority for skill 1 and a reservation level of 5 to get a 80% SL for skill. The SL for skill 2 is now less than 5%! It is better to combine full priority with 23 or 24 generalists dedicated to skill 1 (and no reservations for generalists), then the skill-1 SL gets also at 80%, but the skill-2 SL gets much higher, above 40%.

Up to now we studied a "simple" two-skill call center. What are the lessons to be drawn for call centers with more than two skills? A general conclusion is that it makes a big difference if there are efficiency gains from employing agents only for single skills. Then reservation policies are often less useful, and a limited number of multi-skill agents should be used to obtain the economies of scale.

In a setting where generalists are as efficient as specialists agents should be scheduled on as many skills as possible. Calls should be assigned to specialists first, only when they are all busy then using x-trained agents should be considered. In the multi-skill case it can both occur that an arriving call can be served by multiple x-trained agents and that an x-trained agent has multiple waiting calls to "choose" from. An often used routing policy is where arriving calls look in a list of skill groups for the first that has available agents. Although this can be effective, it is certainly not always the best solution, because it does not take the numbers of available agents into account. This routing policy can well be combined with a call-selection policy that uses weight. In case such a way of prioritizing is not sufficient to get the required SL, then also reservation policies should be used.

9.5 Staffing

Routing is an issue unique to multi-skill call centers (if we forget for the moment the fact that calls in a single-skill call center are "routed" to the longest waiting agent). The next step is the staffing question. How many agents, and with which skill sets, do we need to satisfy the SL requirements? For a single skill we could use Erlang C or one of its generalizations, for multiple skills we have to rely on trial and error. Essential input are the costs of agents with various skill sets. Already for two skills it is not clear how the staffing question should be answered in an optimal way. Of course, it is easy o find a configuration for which the SL constraints are satisfied. But how to check if this choice is also the most cost-effective one? Currently the only way to check this is by trying all possible solutions. For a few skills this is doable, for more than a few skill this quickly becomes unfeasable. Scientific research is being conducted on this subject.

9.6 Workforce planning*

Now suppose that we have some staffing vector for each time interval, specifying for each skill group how many agents should be there. The next step is finding suitable agent schedules, that is, finding the equivalent formulation as optimization problem (6.1), but now in a multi-skill setting. As in (6.1) "better" configurations, with more agents or with agents with more skills, are also allowed. Translating the notion of "better" into linear equation to obtain the equivalent of (6.1) is a complicated mathematical problem that only has been solved for two skills.

Appendix A

Glossary

abandoned call A call that is interrupted by the customer that initiated the call before contact with an agent was made.

ACD *Automatic Call Distributor*, a part of a PABX that can distribute calls that arrive on one or more numbers to extensions which are part of one or more groups that are assigned to that number.

AET *Average Excess Time*, the average time that a call waits longer than the AWT. See page 13.

agent An employee who works in a call center. Also called (call center) representative ('rep'), or CSR (customers sales representative).

ANI *Automatic Number Identification*, a technique used to identify customers by their telephone number. Used in combination with CTI to show right away customer information on the agent's computer screen.

ASA *Average Speed of Answer*, the average time a call waits before speaking to an agent.

AWT *Acceptable Waiting Time*, the target upper bound to the waiting time, very often equal to 20 seconds.

B2B, **B2C** *Business-to-business, -consumer*, relative to sales to customers that are businesses resp. individuals.

call blending A way of handling inbound and outbound calls at the same time by assigning them in a dynamic way to agents.

call center A collection of resources (typically agents and ICT equipment) capable of delivering services by telephone.

channel In the context of contact centers a means to have contact with customers. Examples are telephone, fax, and internet.

contact center A collection of resources (typically agents and ICT equipment) capable of delivering services through multiple communication channels.

cross-trained Denotes an agent who has more than one skill, who can therefore handle more than one type of call. A **generalist** is *fully* cross-trained. Also denoted as **x-trained**.

CTI *Computer-Telephony Integration*, the process that enables communication between and integration of telephone equipment and computer systems.

FCFS *First Come First Served*, refers to the orders in which queued calls are served: in the order of arrival.

generalist An agent who has all skills, i.e., (s)he can handle all types of calls.

ICT *Information and Communication Technology*, technology relative to computers and technology-assisted communication.

model Used in this book as *mathematical model*, a description in mathematics terms of part of a system, that allows an analysis of certain aspects of that system.

OR *Operations Research*, the science that uses mathematical models to improve business operations. Also known as *Management Science*, therefore sometimes called OR/MS. See also www.informs.org.

PABX *Private Automatic Branch eXchange*, the telephone switch local to the company.

predictive dialer Functionality of an ACD that allows outbound calls to be automatically initiated, anticipating future availability of agents.

redial The fact that a caller, after having abandoned, calls back after some time for the same service.

retrial Synonym to redial.

SBR *Skill(s)-based routing*, the fact that different types of calls are routed to different agent groups based on the type of the call and the skills of the agents.

skill group A group of agents all having the same skill set.

skill set The set of skills that an agent or group of agents have.

SL *Service Level*, an ambiguous term that can relate to all aspects of service (waiting time, abandonments, and so forth) or only to the TSF. See Chapter 3.

specialist An agent who can only handle one type of call.

TSF *Telephone Service Factor*, often called SL, the percentage of calls that are answered before AWT.

VRU *Voice Response Unit*, part of an ACD that allows a customer to enter information by responding to computer-generated and/or recorded instructions.

waiting time The time a call spends between the moment it enters the queue (often after a recorded message, or after having made a choice in a VRU) and the moment an agent is connected to the call.

WFM *Workforce Management* consists of all activities from forecasting and planning to online control that have to do with the employment of agents in call center.

WFM tool A computer tool that assists planners with their WFM tasks. It minimally consists of forecasting, Erlang C, and agent scheduling modules.

wrap-up time Time after the end of a call that the agent spends on the call. Consist usually of entering call-related data in a computer system.

Appendix B

Annotated bibliography

This annotated bibliography tries to assist the reader in delving deeper into the subject of call center mathematics. By no means it is our objective to be complete, we just try to give good starting points for further study.

Henk C. Tijms. *A First Course in Stochastic Models.* Wiley, 2003.
This graduate-level text book is one of the sources in which one can find a derivation of the Erlang C formula and the underlying mathematics, such as the exponential distribution and the Poisson process.

N. Gans, G.M. Koole, and A. Mandelbaum. Telephone call centers: Tutorial, review, and research prospects. *Manufacturing & Service Operations Management*, 5:79–141, 2003.
This paper gives the current state of the art concerning mathematical models relevant to call center management. It is written for academics, and assumes solid mathematical knowledge.

A. Mandelbaum. Call centers (centres): Research bibliography with abstracts. Electronically available as `ie.technion.ac.il/~serveng/References/ccbib.pdf`, 2001.
Avi Mandelbaum maintains a call center literature bibliography with hundreds of references.

P. Reynolds. *Call Center Staffing.* The Call Center School Press, 2003.
This book, meant for call center professionals, is an excellent introduction to workforce management.

B. Cleveland and J. Mayben. *Call Center Management on Fast Forward.* Call Center Press, 1997.
Also for call center professionals. Less of an overview, easier reading, but contains many interesting insights.

P. van Ladesteijn and L. Sterk. *Workforce Management*. BBP, Woerden, 2003.
This research report gives an overview of the functionality of 11 of the most popular WFM tools (in Dutch).

G. Jongbloed and G.M. Koole. Managing uncertainty in call centers using Poisson mixtures. *Applied Stochastic Models in Business and Industry*, 17:307–318, 2001.
In this paper a method is developed and used to derive confidence intervals for per-interval call volume estimates, see page 29.

A.N. Avramidis, A. Deslauriers, and P. l'Ecuyer. Modeling daily arrivals to a telephone call center. *Management Science*, 50:896–908, 2004.
This paper presents the state of the art in call center forecasting, especially when it comes to correlations between arrival counts in different intervals.

P. Chevalier, R.A. Shumsky, and N. Tabordon. Routing and staffing in large call centers with specialized and fully flexible servers. Submitted, 2004.
The authors show that under normal circumstances 20% multi-skilled agents is sufficient to obtain most of the profits from the economies of scale.

E. Zohar, A. Mandelbaum, and N. Shimkin. Adaptive behavior of impatient customers in tele-queues: Theory and emperical support. *Management Science*, 48:566–583, 2002.
This article contains some interesting ideas about abandonment behavior. Statistical more advanced methods than the one described on page 32 are also described, notably the Kaplan-Meier estimator for the distribution of the abandonment times.

D.Y. Sze. A queueing model for telephone operator staffing. *Operations Research*, 32:229–249, 1984.
Sze was the first to use the simple approximation of page 49 for redials or reattempts in the context of call centers.

Appendix C

Mathematical background

In this appendix we show how to implement the Erlang C formula and we discuss its extensions. The reader should be familiar with advanced mathematics to fully understand this appendix.

C.1 Implementing the Erlang C formula

In Chapter 4 we gave a formula for the ASA and the TSF in the Erlang C model. The probability of delay played an important role in this formula. This probability is given by the following formula:

$$C(s,a) = \frac{a^s}{(s-1)!\,(s-a)} \left[\sum_{j=0}^{s-1} \frac{a^j}{j!} + \frac{a^s}{(s-1)!\,(s-a)} \right]^{-1}.$$

Remember that s is the number of agents, and that a is the load: the product of arrival rate and average call handling time, thus $a = \lambda\beta$. The formula holds only if $s > a$, if $s \leq a$ then every call has to wait in queue and the probability of delay is thus equal to 1. This is not what the formula gives, therefore whether of not $s > a$ should be tested first. When $s > a$ then the formula above should be calculated. A direct calculation however often leads to numerical instabilities, and therefore completely incorrect results. For example, when calculating $a^j/j!$ (as part of the summation), for say $a = j = 50$, then we divide two extremely big numbers, often resulting in numerical problems. Avoiding this is crucial if we want to find the correct answer for the complete range of possible values for a and s.

To do so we rewrite the formula in such a way that when calculating it we avoid these instabilities. Straightforward math gives:

$$C(s,a) = \left[1 + \frac{s-a}{a} \sum_{j=0}^{s-1} \frac{(s-1)\cdots(j+1)}{a^{s-j-1}} \right]^{-1}.$$

Concentrate on the summation $\sum_{j=0}^{s-1}\frac{(s-1)\cdots(j+1)}{a^{s-j-1}}$. Another way to write this summation is

$$\Big(\cdots\Big(\Big(\frac{1}{a}+1\Big)\frac{2}{a}+1\Big)\cdots\Big)\frac{s-1}{a}+1.$$

This leads us to the following Javascript function for calculating the probability of delay in case $s>a$:

```
function C(s,a){
  var product=1.;
  for(var j=1;j<=s-1;j++)product=product*j/a+1.;
  return 1./(product*(s-a)/a+1);
}
```

Calculating the delay probability is the hardest part of the implementation of the Erlang formula. Once calculated the ASA and the SL can be found easily. We repeat the formula for the ASA or, in mathematical notation, $\mathbb{E}W$, given on page 17:

$$\mathbb{E}W=\frac{C(s,a)\beta}{s-a}.$$

The SL is slightly more complicated, it involves the mathematical constant e (see also page 16):

$$\mathbb{P}(W>t)=C(s,a)\mathrm{e}^{-(s-a)t/\beta}.$$

Here $\mathbb{P}$ should be read as "the probability that ...", and t plays the role of the AWT.

In call centers one is often interested in reverse calculations: what is the number of agents s that is needed to obtain a certain service level? This type of calculations can be done by calculating the SL repeatedly for different values of s. E.g., start with the minimal s such that $s>a$. Then, increase s by 1 until the SL exceeds the required level. Then you have found the right s.

On page 13 we discussed an alternative service level metric: the *average excess time* (AET). In mathematical notation the AET is given by $\mathbb{E}(W-t)^+$, with t again equal to the AWT. The AET is given by the following formula:

$$\mathbb{E}(W-t)^+=\frac{\beta C(s,a)\mathrm{e}^{-(s-a)t/\beta}}{s-a}.$$

C.2 The Erlang X formula and beyond

The attractiveness of the Erlang C system is the fact that a relatively simple closed-form expression exists. When considering extensions such as abandonments and a finite number of lines the expressions get more involved or simply do not exist anymore. Calculations now involve so-called *birth-death processes* (which are actually also used in the derivation of the Erlang C formula), and even then the calculation of the SL requires often advanced and only recently published mathematics. The mathematical details are beyond the scope of this book, we refer the interested reader to the references given in Appendix B. In the case of redials the situation gets even more complicated, and the general theory of *Markov processes* has to be used. These methods are often numerically quite demanding, but in the end they give the exact anwer. This in contrast with approximations or simulation, to be discussed next.

C.3 Simulation

For even more complicated situations, involving for example multiple skills, computational methods become infeasable due to the times that computers need to execute these methods: it easily becomes multiple days and longer, requiring many gigabytes of memory to store intermediate results. A solution is to use smart approximations, they are developed for example for certain problems with skill-based routing. The standard approach for these complex problems however is *discrete-event simulation.* Computer simulation in general is a method by which problems concerning uncertainty are "solved" by repeatedly doing an experiment on the computer.

The estimation of the call volume for future years often depends on estimations of economic growth percentages, market share, etc. For an estimation of the total call volume it suffices often suffices to know the average expected values of these quantities, but to answer questions about say the probability that total yearly call volume exceeds 500K we should estimate the possible fluctuations of the input values and quantify the influence on the outcome. Simulations consists of repeatedly drawing the input numbers, calculating the outcome for each sets of draws, and then averaging all outcomes to come to a final estimation. Mathematical techniques exists to estimate the accuracy of the outcome. This type of calculation can be done using a spreadsheet together with an appropriate add-in.

This type of simulation is also called *Monte-Carlo simulation.* The other main type of simulation is discrete-event simulation: it consists of simulating a system that evolves over time. For this type of simulation specialized visual tools exist, in which you can follows the events happening in the simulation. Alternatively, simulation can be implemented in any general programming language. This has certain adavantages, such as speed and versitality, the disadvantage is that usually there is no time to make a graphical user interface. For many simulation tools building blocks for call centers already exists, making it relatively easy to simulate call centers. Simulation is also used in certain workforce management tools, especially when it comes to multi-skilled operations, for which no simple formulas such as the Erlang C are appropriate.

Simulation is getting more popular, as computing power and the functionality of simulation tools increase. However, simulation time remains considerable, certainly for complex operations and if the results need to accurate. Reliable robust approximation methods are preferable if available. As for all tools based on mathematical models it holds that simulation can only give reliable answers if the system that is considered is well modeled. Therefore the simulation model should be thoroughly tested before conclusions are drawn from it. This step, called verification and validation, is more time-consuming than one would expect and is crucial to a proper use of simulation.

These ideas are well illustrated by the multi-skill simulation tool that can be found at `www.math.vu.nl/~koole/ccmath/sim2skill`. If you push the simulate button then it takes some time before the answers appear, even though this tool concerns a simple small model. Pushing the simulate button again gives indeed different answers. Certain tools give every time again the same answer. This is not a sign of accurancy, but comes from the fact that the simulation is re-initialized at every run.

One should be careful with interpreting results obtained by simulation, certainly when the load to the system exceeds 1 (i.e., when $s < a$ in the terminology of page 65) and there are no abandonments. To avoid that this problem can occur the tool works only with abandonments.