Quantifying Job Fairness in Queueing Systems

by
David Raz

Under the supervision of
Prof. Hanoch Levy and Prof. Benjamin Avi-Itzhak

Submitted to the Senate of Tel-Aviv University
in Partial Fulfillment of the Requirements for the Degree of

Doctor of Philosophy

October 2007

Tel-Aviv University
Raymond and Beverly Sackler Faculty of Exact Sciences
School of Computer Science
To my beloved wife and children: Ronit, Ofri and Lavi
ABSTRACT

Queueing models have long served as key models in a wide variety of fields and applications, both in Computer Science and in other areas. Fairness is widely accepted as a major issue in the operation of queues, perhaps the reason why queues were formed in the first place. There is also ample evidence that fairness is important to customers. Despite the above, little study has been conducted on the subject of queue fairness and how to quantify it. As a result, the issue of fairness is not well understood and agreed upon measures do not exist.

In this work we propose a measure and quantitative models for measuring fairness to customers or jobs. We study the measure’s main properties and show that it fits the intuition of both customers and practitioners, while agreeing with some other important properties.

We study and quantitatively compare different systems and settings using this measure: we compare different service policies under single server settings; we study common mechanisms used for managing multi-server systems such as using multiple queues, jockeying and queue joining policy; we study different classification and prioritization mechanisms such as priority queues and server dedication. In each of the areas we provide practical methods and results for quantitatively evaluating fairness.

We also study the important related subject of predictability. We propose a simple measure for predictability, and analyze it for several common systems.
ACKNOWLEDGMENTS

First and foremost, I wish to express my gratitude to my thesis advisors, Prof. Hanoch Levy and Prof. Benjamin Avi-Itzhak, for their dedication and hard work, but more than this, for provided exciting, invigorating, and insightful discussions and ideas throughout the course of this work. Their enthusiasm with the subject of Fairness has always been an inspiration to me. I’m also especially thankful for them lending me their vast experience in navigating the academic maze.

I would like to thank the people and organizations that funded my research. For one year I was fully funded by an IBM Ph.D. Fellowship, for which I wish to thank the IBM Corporation and my mentor for this fellowship, Dr. Dagan Gilat. The rest of the time I was partially funded by the Israeli Ministry of Science and Technology, grant number 380-801, by the EURO-NGI network of excellence, and by grants from the School of Computer Science. Other funding was supplied by the Deutsch Foundation travel grants, the KIPP Foundation travel grants and by a School of Computer Science prize for study achievements.

I wish to thank many editors and anonymous reviewers for their many insightful and valuable comments. Much of the material in this work was greatly influences by such comments.

Finally, on a personal note, I would like to thank my beloved family. I am thankful to my parents, who placed me on the beginning of the academic journey so naturally, and who keep supporting me on this road, and to my grandfather who is always proud of my achievements. But most of all I wish to thank my wife Ronit and my children, Ofri and Lavi, for their love, patience, dedication, encouragement and support on this long, and sometimes frustrating, adventure. I lovingly dedicate this thesis to them.
CONTENTS

Abstract ................................................................. iii
Acknowledgments ......................................................... iv
Contents ................................................................. v
List of Figures ........................................................... xi
List of Tables ............................................................ xiii

1. Introduction to Job Unfairness ................................. 1
   1.1 Thesis Outline .................................................. 5

2. Prior Art ........................................................................ 7
   2.1 Job Fairness Measures ........................................... 7
      2.1.1 Strict Seniority Based Fairness Measures .................. 7
      2.1.2 Slowdown Based Fairness Measures ......................... 9
      2.1.3 Discrimination Frequency Measures ....................... 12
   2.2 Fairness Measures in Related Areas ........................... 13
   2.3 Fairness Measures in General .................................. 16

3. Model and Notation .................................................... 18
   3.1 Customers and Scenarios ........................................ 18
   3.2 Classes of Service ................................................ 19
   3.3 General Notation Practices ..................................... 19
3.4 Common Service Policies .............................................. 20

4. Introducing the Fairness Measure: RAQFM .......................... 23
   4.1 RAQFM For Single Non-Idling Server ......................... 25
      4.1.1 Individual Customer Discrimination ...................... 25
      4.1.2 System Measure of Unfairness ............................ 26
      4.1.3 Unfairness of a Scenario ................................. 27
   4.2 Generalization of RAQFM for Multi-Server Systems .......... 28
   4.3 Generalization of RAQFM for Networks of Queues .......... 29
      4.3.1 The Global Method ........................................ 29
      4.3.2 The Local Method ........................................ 30
      4.3.3 The Decomposition Method ............................... 31

5. Basic Properties of RAQFM ............................................ 33
   5.1 Sensitivity of the Measure to Seniority and Service Requirement Differences 33
      5.1.1 The Mr. Short vs. Mrs. Long Example .................. 33
      5.1.2 Reaction to Differences in Seniority ..................... 35
      5.1.3 Reaction to Differences in Service Requirement ........ 38
   5.2 Absolute Fairness of PS ......................................... 40
   5.3 Zero Sum ........................................................... 42
   5.4 Bounds of RAQFM .................................................. 43
      5.4.1 Bounds On Individual Discrimination ..................... 43
      5.4.2 Bounds on Scenario and System Fairness ................ 44

6. The Locality of Measurement Property ............................... 47
   6.1 Locality of Reference and Comparison Set .................... 50
   6.2 Locality of Measurement, Locality of Variance and the Relation Between Them .............................................. 51
   6.3 Locally Measured Metrics ....................................... 54
   6.4 Explicit Evaluation of the Intra-Variance ..................... 60
### 6.5 Numerical Results
- 6.5.1 Numerical Example ........................................... 62
- 6.5.2 Simulation Results ........................................... 63

### 7. Computing RAQFM Under The Markovian Model
- 7.1 The Analysis Methodology ...................................... 65
- 7.2 Example: Analysis of Unfairness in a FCFS M/M/1 Queue ......... 68

### 8. Fairness in Single Server Systems
- 8.1 Conditional Discrimination in M/M/1 .......................... 71
  - 8.1.1 FCFS ........................................... 72
  - 8.1.2 LCFS ........................................... 76
  - 8.1.3 P-LCFS ........................................... 81
  - 8.1.4 RGS ........................................... 83
- 8.2 System Unfairness in M/M/1 .................................. 85
  - 8.2.1 FCFS ........................................... 85
  - 8.2.2 LCFS ........................................... 85
  - 8.2.3 P-LCFS ........................................... 86
  - 8.2.4 RGS ........................................... 86
  - 8.2.5 Numerical results and Discussion ......................... 87
- 8.3 The Tradeoff Between Seniority and Service Requirements, and RAQFM’s Sensitivity To It ........................................ 88
- 8.4 Going Beyond Exponential Service Requirement ................. 90
- 8.5 Summary ................................................................ 92

### 9. Fairness of Multiple Server Systems ................................. 93
- 9.1 Fairness of G/D/m Type Systems ............................... 94
  - 9.1.1 The Effect of Seniority on Fairness ....................... 94
  - 9.1.2 The Effect of Multiple Queues Operation Mechanisms on Fairness 98
  - 9.1.3 The Effect of Queue Joining Policy on Fairness .......... 103
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>11.4.3</td>
<td>LCFS and P-LCFS</td>
<td>172</td>
</tr>
<tr>
<td>11.4.4</td>
<td>SJF and P-SJF</td>
<td>173</td>
</tr>
<tr>
<td>11.4.5</td>
<td>LJF and P-LJF</td>
<td>175</td>
</tr>
<tr>
<td>11.4.6</td>
<td>LAS</td>
<td>175</td>
</tr>
<tr>
<td>11.4.7</td>
<td>SRPT</td>
<td>176</td>
</tr>
<tr>
<td>11.4.8</td>
<td>LRPT</td>
<td>178</td>
</tr>
<tr>
<td>11.4.9</td>
<td>RR</td>
<td>178</td>
</tr>
<tr>
<td>11.5</td>
<td>Discussion: Twin Measure Results</td>
<td>178</td>
</tr>
<tr>
<td>11.5.1</td>
<td>Classifying the Scheduling Policies</td>
<td>179</td>
</tr>
<tr>
<td>11.5.2</td>
<td>Comparing Predictability Criteria</td>
<td>181</td>
</tr>
<tr>
<td>11.5.3</td>
<td>Optimality under the Twin Measure</td>
<td>183</td>
</tr>
<tr>
<td>11.6</td>
<td>The Twin Measure in Multi-Server Systems</td>
<td>184</td>
</tr>
<tr>
<td>11.6.1</td>
<td>Single Queue</td>
<td>184</td>
</tr>
<tr>
<td>11.6.2</td>
<td>Multiple Queues</td>
<td>185</td>
</tr>
<tr>
<td>11.7</td>
<td>Extending the Twin Measure</td>
<td>187</td>
</tr>
<tr>
<td>11.7.1</td>
<td>Job Trains</td>
<td>187</td>
</tr>
<tr>
<td>11.7.2</td>
<td>Non-Simultaneous Twins</td>
<td>189</td>
</tr>
<tr>
<td>12.1</td>
<td>Online Algorithms and Competitive Analysis</td>
<td>190</td>
</tr>
<tr>
<td>12.2</td>
<td>Tail Analysis</td>
<td>192</td>
</tr>
<tr>
<td>12.3</td>
<td>Future Research</td>
<td>195</td>
</tr>
<tr>
<td>Bibliography</td>
<td>197</td>
<td></td>
</tr>
<tr>
<td>Appendices</td>
<td>212</td>
<td></td>
</tr>
<tr>
<td>Appendix A. Stronger Proof for Reaction to Differences in Seniority</td>
<td>213</td>
<td></td>
</tr>
<tr>
<td>Appendix B. Proof of the Locality of Reference Theorem</td>
<td>216</td>
<td></td>
</tr>
</tbody>
</table>
Appendix C. Analysis of Conditional Discrimination and Unfairness in an M/E_r/1 System
# LIST OF FIGURES

<table>
<thead>
<tr>
<th>Figure</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1</td>
<td>Classification According to the Expected Slowdown Fairness Criterion</td>
<td>11</td>
</tr>
<tr>
<td>5.1</td>
<td>Short and Long - Seniority vs. Service Requirement</td>
<td>34</td>
</tr>
<tr>
<td>6.1</td>
<td>“Heavy First Job” Scenarios under a Non-Preemptive Service Policy</td>
<td>59</td>
</tr>
<tr>
<td>6.2</td>
<td>Variance Vs. Intra-Variance of Sojourn Time</td>
<td>64</td>
</tr>
<tr>
<td>8.1</td>
<td>Conditional (normalized) Discrimination for M/M/1 under FCFS</td>
<td>75</td>
</tr>
<tr>
<td>8.2</td>
<td>Conditional (normalized) Discrimination for M/M/1 under LCFS</td>
<td>81</td>
</tr>
<tr>
<td>8.3</td>
<td>Conditional (normalized) Discrimination for M/M/1 under P-LCFS</td>
<td>83</td>
</tr>
<tr>
<td>8.4</td>
<td>Conditional (normalized) Discrimination for M/M/1 under ROS</td>
<td>84</td>
</tr>
<tr>
<td>8.5</td>
<td>System Unfairness (Variance of Discrimination) for M/M/1</td>
<td>87</td>
</tr>
<tr>
<td>8.6</td>
<td>System Unfairness For Discreet Distribution</td>
<td>89</td>
</tr>
<tr>
<td>9.1</td>
<td>Case 1 - Sequentially Served Customers By The Same Server</td>
<td>95</td>
</tr>
<tr>
<td>9.2</td>
<td>Case 2 - Partially Parallel Served Customers</td>
<td>96</td>
</tr>
<tr>
<td>9.3</td>
<td>Selection of Jockeying Customer, Equal Service Requirements, Case 2</td>
<td>101</td>
</tr>
<tr>
<td>9.4</td>
<td>Unfairness of Four Queue Strategies under the M/D/2 Model</td>
<td>104</td>
</tr>
<tr>
<td>9.5</td>
<td>Unfairness in the M/D/2 Model - Comparison of Queue Joining Policies</td>
<td>106</td>
</tr>
<tr>
<td>9.6</td>
<td>Unfairness of Four Queue Strategies for the M/M/2 Model</td>
<td>115</td>
</tr>
<tr>
<td>9.7</td>
<td>The Effect of the Queue Joining Policy on the System Fairness</td>
<td>119</td>
</tr>
<tr>
<td>9.8</td>
<td>Unfairness of Four Queue Strategies for High Variability Service Require</td>
<td>125</td>
</tr>
<tr>
<td>10.1</td>
<td>The Effect of Preemptive Priority on Four Classes of Customers</td>
<td>135</td>
</tr>
<tr>
<td>10.2</td>
<td>Single Server FCFS with Two Customer Classes: State Diagram</td>
<td>139</td>
</tr>
<tr>
<td>Figure</td>
<td>Description</td>
<td>Page</td>
</tr>
<tr>
<td>--------</td>
<td>-----------------------------------------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>10.3</td>
<td>Single Server FCFS System with Two Customer Classes, Exponential Distribution</td>
<td>141</td>
</tr>
<tr>
<td>10.4</td>
<td>Class Discrimination in a Single Server Priority System with Two Customer Classes, Exponential Distribution</td>
<td>147</td>
</tr>
<tr>
<td>10.5</td>
<td>Unfairness in a Single Server System with Two Customer Classes, Exponential Distribution</td>
<td>148</td>
</tr>
<tr>
<td>10.6</td>
<td>Unfairness in a Single Server Priority System with Two Customer Classes, Non-Exponential Distribution</td>
<td>149</td>
</tr>
<tr>
<td>10.7</td>
<td>Unfairness and Class Discrimination in a 12 server Dedication System</td>
<td>156</td>
</tr>
<tr>
<td>10.8</td>
<td>The Effect of Queue Dedication on the System Fairness</td>
<td>159</td>
</tr>
<tr>
<td>10.9</td>
<td>Comparison Between the Unfairness in Single Server and Dual Server Systems</td>
<td>165</td>
</tr>
<tr>
<td>11.1</td>
<td>Classification According to CRTC</td>
<td>181</td>
</tr>
<tr>
<td>A.1</td>
<td>Two Customers With Stochastically Identical Service Requirements</td>
<td>214</td>
</tr>
</tbody>
</table>
LIST OF TABLES

6.1 Variances of Discrimination Vs. Variance of Sojourn Time . . . . . . . . . 63

11.1 Comparing Predictability Criteria . . . . . . . . . . . . . . . . . . . . . . . 182
INTRODUCTION TO JOB UNFAIRNESS

Queueing models have long served as key models in a wide variety of fields and applications, including computer systems, telecommunication systems, and human services systems, e.g. Web servers, bank offices, etc., to mention just a few. In the context of computer systems, queues traditionally served a major role in operating systems. With the advance of technology and the shift of many services to computer systems, such as Web-based services and Call Centers, computer-managed queues are becoming an important part of many daily life applications.

In many applications, where an important question is how to operate the queueing system, the two issues most often considered are “efficiency”, which is mostly attributed to the mean waiting time in the system, and “fairness”. Efficiency has been extensively studied and is well understood, see the many textbooks on the subject e.g. Kleinrock [88, 85], Hall [61], Cooper [35], Daigle [39], Gross and Harris [60]. However, despite its great importance, which we soon discuss in detail, little study has been conducted on the subject of queue fairness and how to quantify it. As a result, the issue of fairness is not well understood and agreed upon measures do not exist.

Fairness is one of the cardinal issues in queueing systems. In fact, one may argue that one of the main reasons, perhaps the utmost reason, for using a queue in the first place, be it in a public office or at a computer system, is to provide some type of fairness to the jobs or customers being served. In this sense one could view a queue as a “fairness management facility”.

\footnote{The terms Customers and Jobs are used interchangeably throughout this work.}
Chapter 1. Introduction to Job Unfairness

The fairness factor associated with waiting in queues has been recognized in many works and applications; some of them are listed next. Larson [89] in his discussion paper on the dis-utility of waiting, recognizes the central role played by ‘Social Justice’, which is another name for fairness, and its perception by customers. This is also addressed by Rothkopf and Rech [129] in their paper discussing perceptions in queues. In that paper they bring an impressive list of quantifiable considerations showing that combining queues may not be economically advantageous, contrary to the common belief. At the end they concede however, that all these considerations may not have sufficient weight to overcome the unfairness perceived by customers served in a separate queues structure. Aspects of fairness in queues were also discussed earlier by quite a number of authors. Some of these are Palm [108] that deals with judging the annoyance caused by congestion, Mann [96] that discusses the queue as a social system, and Whitt [151] that addresses overtaking in queues.

Scientific evidence of the importance of fairness of queues was recently provided by Rafaeli et al. [112, 113]. That work uses an experimental psychology approach to study the reaction of humans to waiting in queues and to various queueing and scheduling policies. The studies revealed that for humans waiting in queues, the issue of fairness is highly important, sometimes even more important than the duration of the wait. For the case of common queue versus a separate one at each server, they found that the common queue was perceived as more fair. Probably for this reason we find separate queues mostly in systems where a common queue is physically not practical, e.g. traffic toll booths and supermarkets.

Recently, the issue of fairness is also discussed in the context of practical computer applications. For example, the issue of fairness in web servers is discussed by Harchol-Balter et al. [63], where a policy is shown to have reduced response times, but at the expense of unfairness to large jobs.

Considering the apparent importance of fairness of queues, there is very little published work providing quantitative results on job fairness. The objective of this work is to propose such measures and quantitative models.
But what are the major factors in measuring fairness in queues?

Traditionally, serving customers by order of arrival, i.e. First-Come-First-Served (FCFS), is considered the most fair queueing discipline. This probably derives from experience in exhaustible service systems where the total amount of service the system is able, or willing, to dispose is limited in some way e.g. a line at a gas pump at a time of energy crisis, a line for basic foods in a refugee camp, or less dramatic, a line for tickets for a show or a sports event. If you are not early enough in the queue, chances are you will never get the service, or product. Clearly, placing ahead of you a person who arrived after you can be regarded as grossly unfair, particularly if that person is not needier than you. Most present day queueing systems, however, are not of this type, and thus FCFS may not be as crucial in these systems. In this work we focus on such non-exhaustible servers.

Larson discusses the psychology of waiting. In the first part of that paper, dedicated to social justice in queues, the author brings several anecdotal actual situations, experienced by him and others, that strongly support the traditional claim of FCFS being the most socially just queue discipline. In fact he practically defines social injustice as violation of FCFS when stating

“customers may become infuriated if they experience social injustice, defined as violation of FIFO.”

So what would be a fair service order in a supermarket queue, an airport waiting line, or a computer system? Many people would instinctively embrace Larson’s view, responding that FCFS is the fairest order. Already Kingman pronounces that FCFS is

“in a sense the ‘fairest’ queue discipline.”

The underlying principle, or rationale, of this view can be expressed in one sentence: the one who has been waiting longest earned the right to be served first. But, recalling that the server is non-exhaustible, namely, it can serve forever, is FCFS undeniably the most fair discipline?

\(^2\) We use Typewriter-Style to denote scheduling policies. The description of the different policies can be found in Section 3.3.
To answer this question, consider a common situation which perhaps is best depicted in the supermarket queue setup: Mr. Short arrives at the supermarket counter holding only one item. In the line ahead of him he finds Mrs. Long carrying two fully loaded carts of items. Short says to Long “Excuse me, I only have one item. Would you mind if I go ahead of you?” Would it be fair to have Mrs. Long served ahead of Short and Short waiting for the full processing of Mrs. Long’s loaded carts? Or, would it be more fair to advance Short in the queue and serve him ahead of Long? This dilemma may cause some to “relax” their strong belief in the absolute fairness of FCFS. In fact, the dilemma brings to the discussion a new factor, that of service requirement. The basic intuition thus suggests that prioritizing short jobs over long jobs may also be fair, based on the underlying principle that the one who demands the least of the server’s time should be served first. The question of whether it is fair to serve Short before Long, and the dilemma associated with this question, is rooted in the contradicting factors of seniority difference, working to the benefit of Long, and service requirement difference, working to the benefit of Short. To further demonstrate the conflict we continue our scenario in two directions: (i) Long responds “Why don’t you go ahead of me. I have arrived just a few seconds ago and it is not fair that you will wait that long while your short service will delay me very little”. This is one possibility. Alternately, Long may be negative, saying (ii) “Look, I have been waiting in this line forever. If not for this lengthy wait I would have been out of here long before your arrival. You can patiently wait too”. Clearly, Long weighs their seniority difference against their service requirement difference in deciding what is the fair thing to do.

One may also note that the behavior of a queueing system is traditionally governed by these two major physical factors, job seniority and job service requirements. In traditional queueing analysis they serve, in the form of arrival times and service requirements, together with the server policy, to derive the system performance e.g. expected delay.

As we will show in Section 2.1, most of the measures available today don’t deal

---

This applies to practically all the measures available in the time when this research has started, in early 2003.
with both of these factors. Very intriguing and drastically contradicting results may be obtained if only one of the factors is accounted for. Thus, a measure that accounts only for seniority differences, as the one by Avi-Itzhak and Levy [7], will rank FCFS as the most fair policy and Last-Come-First-Served (LCFS) as the most unfair policy, in the case of equal service requirements and uninterrupted service. In contrast, a measure that accounts only for service requirement differences, such as the criterion developed by Wierman and Harchol-Balter [155], implies that Preemptive Last-Come-First-Served (P-LCFS) is always fair, while FCFS is always unfair. We will review both of these measures in detail in Section 2.1.

Our objective is therefore to propose a measure which deals with both job size and job seniority. Another important requirement is that it will be convenient to measure and work with by analysts. The measure is called RAQFM - a Resource Allocation Queueing Fairness Measure.

Using this measure, we study different queueing mechanism, focusing on evaluating the relative fairness of these mechanisms. Such analysis provides measures of fairness for these systems, that can be used to quantitatively account for fairness when considering alternative designs. The quantitative approach can enhance the existing design approaches in which efficiency (e.g., utilization and delays) is accounted for quantitatively, while fairness is accounted for only in a qualitative way.

1.1 Thesis Outline

The structure of this work is as follows. We first (Chapter 2) discuss prior work, both on the specific subject of queueing job fairness and on related areas. We then (Chapter 3) describe the general model and notation used throughout this work. Following this (Chapter 4) we introduce and describe the RAQFM measure itself, along with its quantitative model, for various settings. We then study the the basic properties of the RAQFM measure (Chapter 5). We dedicate another chapter, Chapter 6, to one specific property, namely to locality of measurement, due to its importance.
We then move on to apply RAQFM to different settings. But prior to that, we present
a general method for computing RAQFM in Markovian systems in steady state (Chapter
7). The method is demonstrated in detail for a single server FCFS system, and the same
method is later utilized to produce many of the results in this work. We are then free
to move to the actual application of RAQFM. We focus on several areas of interest. In
Chapter 8 we apply RAQFM to service policies in single server systems. In Chapter 9
we apply it to multiple server and multiple queue systems. In Chapter 10 we study the
fairness of classification and prioritization mechanisms. For each of these areas of interest
we discuss the relevant mechanisms used in theory and in practice, and compare them
quantitatively.

While the bulk of our research is about fairness, during our research we discovered that
the subject of fairness is closely related to that of predictability. If a system is unfair to
customers, it is sometimes also hard to predict the behavior of that system. We therefore
also researched the subject of predictability, although not to the same width and depth.
Chapter 11 is a self-contained chapter dealing with our research on predictability.

We finalize the discussion in Chapter 12 with mentioning some areas in which work is
in progress. In that chapter we also list many other open questions and suggestions for
future research.

The majority of the material in this thesis was previously published in a series of papers
and technical reports: Raz et al. [117, 118], Avi-Itzhak et al. [11], Raz et al. [120], Levy
et al. [93], Raz et al. [116, 119, 122, 121, 123]. We point the reader in the beginning of
each chapter to the relevant resources.
Chapter 2

PRIOR ART

In this chapter we survey prior work on job fairness. We start by surveying work dealing specifically with measures for job fairness in queues. We then provide some work done in closely related areas, such as flow fairness and fair CPU allocation. Lastly, we provide some pointers to general works which might be loosely related, but are of interest to the reader.

Major parts of this survey were published in Avi-Itzhak et al. [10]. See also a short survey by Wierman [153].

2.1 Job Fairness Measures

In this section we survey job fairness measures. As mentioned in the introduction, we believe a fairness measure should deal with both job size and seniority. We therefore also point out for each measure if it deals with job size, job seniority, or both.

2.1.1 Strict Seniority Based Fairness Measures

Several fairness measures were proposed which aim strictly at conserving seniority.

Maybe the first to deal with measuring fairness implicitly is Kingman [83], which shows that among all non-idling policies where the jobs are indistinguishable, FCFS has the lowest variance and is therefore

“in a sense the ‘fairest’ queue discipline.”
Thus, implicitly, the *Variance of Waiting Time* serves as a measure of fairness. This is of course a limited view, which deals only with the seniority of the jobs. We deal with the variance as a fairness measure in details in Chapter 6.

In Avi-Itzhak and Levy [7] a measure based on *Order of Service* has been devised for jobs with equal service requirements. The study deals with a specific sample path of the system $\pi$, and examines a realization of the service order with a fairness measure $F(\pi)$ defined on the service order. The paper assumes several axioms on the properties of $F(\pi)$. The major axiom is 1) Monotonicity of $F()$ under neighbor jobs interchange: If two neighboring jobs are interchanged then $F()$ is strictly monotone with the seniority difference between the jobs. The other axioms are 2) Reversibility of the interchange, 3) Independence on position and time, and 4) Fairness change is unaffected by jobs not interchanged. The results derived show that the quantity

$$c \sum_i a_i \Delta_i + \alpha,$$

where $a_i$ is the arrival epoch of customer $C_i$, $\Delta_i$ is the *order displacement* of $C_i$, i.e. the number of positions $C_i$ is pushed ahead or backwards on the schedule, compared to FCFS, and where $c > 0$ and $\alpha$ are arbitrary constants, satisfies the axioms. This quantity is the unique form satisfying the axioms applied to any feasible interchange, i.e. not necessarily of neighbors. Under steady state this quantity is equivalent to the variance of the waiting time, with a negative sign, up to a constant multiplier. The waiting time variance can thus serve as a surrogate for the system’s unfairness measure. By definition of the axioms, this approach deals with seniority differences only.

Gordon [58] proposes the *Slips and Skips* approach. It aims at quantifying the violation of social justice due to overtaking in the queue. The underlying rationale is that FCFS is just and customer overtaking causes injustice. The approach defines two types of overtaking events experienced by a tagged customer: 1) A Skip - when the tagged customer overtakes another customer, namely, it completes service before a customer that arrived ahead of it, and 2) A Slip - when the tagged customer is overtaken by another customer. While the
work suggests that counting the number of skips and slips can provide an indication of
the amount of injustice and analyzes these counts, it does not deal with how to use these
as the basis for a fairness measure. Several systems are studied, including: Two M/M/1
systems in parallel with random queue joining, two M/M/1 systems in parallel, where the
tagged customer, and only that customer, uses the “join the shortest queue” strategy, the
multi server system M/M/m, and the infinite-server system M/G/∞. For these systems
the probability laws of the number of skips and the number of slips experienced by an
arbitrary customer (denoted $N_{\text{SKIPS}}$ and $N_{\text{SLIPS}}$, respectively) are derived. Interesting
results are: 1) For every system $N_{\text{SKIPS}} = N_{\text{SLIPS}}$, 2) For most systems the distributions
of the two variables differ from each other, 3) Only one system is found by the author where
the distributions equals each other, the M/G/∞ system where the service requirement
distribution is symmetric around its mean, 4) Using the “join the shortest queue” strategy
by the tagged customer, when no one else uses it, reduces the number of slips and increases
the number of skips the tagged customer experiences.

2.1.2 Slowdown Based Fairness Measures

The Slowdown (a.k.a. stretch, normalized response time) was proposed as a metric of
unfairness in several works. The Slowdown of a job $j$ is defined as $T_j/s_j$ where $T_j$ is the
job’s response time (a.k.a. flow time, sojourn time, turnaround time; the time from when
the job first arrives at the system until it departs the system) and $s_j$ is the job’s service
requirement. The reason for looking at the slowdown metric is that it makes intuitive
sense that the mean response time experienced by a user should be proportional to the
service requirement of the job submitted by the user. For example, the PS policy was
observed by Kleinrock [85], Sec. 4.4 as being “eminently fair” since under PS (for M/G/1)
all jobs experience the same mean slowdown: $1/(1 - \rho)$, where $\rho < 1$ is the system’s load
(Sakata et al. [132, 133]). Note that inherently, slowdown based measures focus on the
size of jobs, and ignore the seniority, although the SQF measure from Avi-Itzhak et al. [9]
does account for seniority, as we will discuss later.

In Bender et al. [14] the max slowdown is proposed as a performance metric, and is
used as indication of unfairness. The authors provide theoretical results, both positive and negative, about algorithms for minimizing the max slowdown in different settings: offline (the job arrival times are specified ahead of time), online (jobs are released only at their arrival time and the algorithm must decide only based on already released jobs), nonpreemptive (jobs must be processed without interruption), and preemptive (jobs may be suspended and resumed latter with no cost). They also provide experimental results from HTTP and database logs using Shortest-Remaining-Processing-Time (SRPT), FCFS and their own heuristic policy DEDF.

One major problem with using the max slowdown as a fairness measure is that the slowdown of an arbitrary job can usually be arbitrarily large, even if in general jobs of that type are treated well. To avoid this problem Bansal and Harchol-Balter [13], when analyzing SRPT, propose to use the the expected slowdown. Let $T(x)$ be the steady-state response time for a job of size $x$. The slowdown seen by a job of size $x$ is defined as $S(x) \overset{def}{=} T(x)/x$ and the expected slowdown for a job of size $x$ is $\mathbb{E}\{S(x)\}$. In Harchol-Balter et al. [62] the expected slowdown as the job size goes to infinity is studied. It is shown that for any work conserving policy $\lim_{x \to \infty} \mathbb{E}\{S(x)\} \leq 1/(1-\rho)$. More specifically for PS (for which it is well known), SRPT, P-LCFS, LAS and LRPT this holds with equality.

In Wierman and Harchol-Balter [155] this is taken a step further (which is further discussed in Wierman [152]) when the expected slowdown is used for classifying M/GI/1 disciplines into three classes as follows.

**Definition 2.1.** Let $\mathbb{E}\{S(x)\}^P$ be the expected slowdown for a job of size $x$ under service policy $P$. Consider an M/GI/1 system with load $0 < \rho < 1$. A job size $x$ is treated fairly under policy $P$, service distribution $X$, and load $\rho$ if

$$\mathbb{E}\{S(x)\} \leq \frac{1}{1-\rho}.$$  

Otherwise, a job size $x$ is treated unfairly. A scheduling policy $P$ is fair under service distribution $X$ and load $\rho$, if every job size is treated fairly. Otherwise $P$ is unfair for that service distribution and load.
The choice of $1/(1 - \rho)$ is justified mainly because PS is considered fair, and thus any job size which is treated at least as good as the way it is treated under PS is treated fairly. Wierman [152] provides more formal reasoning.

**Definition 2.2.** Let $0 < \rho < 1$ in an M/GI/1 queue where $X$ is non-deterministic. A scheduling policy $P$ is: (i) Always Fair if $P$ is fair for all such $\rho$ and $X$; (ii) Sometimes Fair if $P$ is fair under some $\rho$ and $X$ and unfair under other $\rho$ and $X$; or (iii) Always Unfair if $P$ is unfair under all loads and service distributions.

The results of this classification for a wide set of scheduling policies are summarized in Figure 2.1 (the figure is from Wierman [152], brought with permission of the author).

![Figure 2.1: Classification According to the Expected Slowdown Fairness Criterion](image)

Another approach based on the slowdown is the Slowdown Queueing Fairness measure, SQF, proposed in Avi-Itzhak et al. [9]. SQF is based on the idea of equal slowdown, where deviation from equal slowdown is interpreted as unfair. Consider a system in equilibrium where service requirements are distributed as a random variable $B$ with probability density function (pdf) $b(x)$ and moments $b_k = \mathbb{E}\{B^k\}$. Let $T$ denote the equilibrium sojourn time and let $T(x)$ denote the conditional sojourn time for a customer whose service requirement
is $x$. Absolute fairness is obtained when $T(x) = cx$, for some constant $c$ which is chosen to in a way that guarantees zero sum. This leads to the following definition:

**Definition 2.3.** The slowdown queueing fairness measure is the second moment of $T(x)$ around $cx$:

$$SQF = \int \mathbb{E}( (T(x) - cx)^2 ) b(x) dx$$

where $c = \mathbb{E}(T)/b_1$.

The authors prove that unlike other fairness measures based on the slowdown, SQF accounts well for both seniority and size. For the $M/GI/1$ model the authors study non size-based policies and show that $SQF_{FCFS} < SQF_{ROS} < SQF_{LCFS}$ and $SQF_{PS} \leq SQF_{P-LCFS}$. For the $M/M/1$ and $M/D/1$ models they also show that $SQF_{LCFS} = SQF_{P-LCFS}$ and $SQF_{LCFS} < SQF_{P-LCF}$, respectively. For other policies they provide simulation results that suggest that the measure always increases when the variability increases. Further, when variability is very high, SRPT and PS are the most fair amongst the policies compared.

2.1.3 Discrimination Frequency Measures

Sandmann [136] presents an approach based on accounting for both service size and seniority explicitly, unlike other measures we surveyed which do so implicitly, if at all. Under that approach a customer is discriminated if either other customers are served earlier although they did not arrive earlier, or if the customer has to wait for customers with large service requirements. These two forms of discrimination are called *overtaking* and *large jobs*, respectively.

**Definition 2.4.** Let $n_i$, the amount of overtaking a job $J_i$ suffers from, be the number of jobs that arrived not earlier and complete service not later than $J_i$. Let $m_i$, the amount of large jobs a job $J_i$ suffers from, be the number of jobs not completely served upon arrival of $J_i$ that have at least as much remaining service requirements, and complete service not
later than \( J_i \). The discrimination frequency of a job is

\[
DF(i) \overset{\text{def}}{=} n_i + m_i.
\]

The fairness of a system is defined by taking expectations over the discrimination frequency of an arbitrary job in steady state, i.e. \( \mathbb{E}\{DF\} \).

It is easy to see that the measure accounts (explicitly) for both seniority and size, and this is further emphasized in Sandmann \[136\]. In Sandmann \[134\] the measure is evaluated for FCFS, LCFS and SJF. For the \( M/M/1 \) model it is analytically shown that \( \mathbb{E}\{DF\}^{SJF} < \mathbb{E}\{DF\}^{FCFS} < \mathbb{E}\{DF\}^{LCFS} \). Using simulation, the authors show that this is not the case with other service requirement distributions, where sometimes FCFS is more fair than SJF. In Sandmann \[135\] the author proposes a service policy which he shows by simulation to be more fair than both FCFS and SJF.

### 2.2 Fairness Measures in Related Areas

Fairness has been excessively treated in computer and communication science in contexts different than that of job fairness. When we relate to jobs we have specific customers in mind, arriving at the system, receiving service, and departing. This is not the case with the following related areas of research.

The first related area, where several fairness measures were proposed, is **flow control**. The best known notion in this area is that of Max-Min Fairness, starting with Jaffe \[70, 71\] and used by many afterwards. According to this concept a flow control scheme is fair if

“each user’s throughput is at least as large as that of all other users which have the same bottleneck.”

Gerla and Staskauskas \[54, 55\] achieve fairness by minimizing an objective function which was originally suggested by Gallager and Golestani \[52\], which includes a penalty for uneven allocation. Bharath-Kumar and Jaffe \[17\] state that any algorithm which allocates
zero throughput to any user is unfair; otherwise it is fair. Marsan and Gerla [98] present a quantitative fairness measure

\[ F = \min_{i,j} \frac{p_i}{p_j}, \]

where \( p_i \) and \( p_j \) are the powers (ratio of throughput and round trip delay) achieved by users \( i \) and \( j \) respectively. A protocol is called optimally fair if \( F = 1 \). Sauve et al. [137] contends that the fairness of a network should be measured by the variance of network delays of the different classes. This is revised to squared coefficient of variation of delay by Wong and Lam [158] and to a weighted squared coefficient of variation in Wong et al. [159]. Almost two decades later, the research area flourished again with the proposal of Proportional Fairness by Kelly [80], Kelly et al. [81]. Lately Balanced Fairness (Bonald and Proutière [20, 21]) and \((p, \alpha)\)-Proportional Fairness (Mo and Walrand [100]) were proposed. See survey of results in Roberts [126] and a comparative study in Bonald et al. [22]. The measures in this area assume some flow requirements exist in a network, and focus on the amount of bandwidth each flow should receive. The assumption is that the requirements and bandwidth assignments are permanent, and there is little significance to the order in which specific jobs arrive; only the overall bandwidth matters. As such they cannot be readily applied to the models we are interested in. An interesting recent paper on the subject, Briscoe [26], claims that comparing flow rates should not be used for claims of fairness in production networks. Instead, one should judge fairness mechanisms on how they share out the ‘cost’ of each user’s actions on others.

A second related area is that of stream fairness. Within that context, the literature has flourished on the subject of Fair Queueing, which received very much attention in the recent decade. Fair Queueing disciplines (starting with Demers et al. [43, 44]), and their variants, aim at the fair scheduling of packet streams within network devices. The most prominent measures in this area are the Absolute Fairness Bound (AFB) and Relative Fairness Bound (RFB). AFB, first used probably in Greenberg and Madras [59], is based on the maximum difference between the service received by a flow under the discipline being measured, and the service it would have received under the ideal PS policy. As AFB
Chapter 2. Prior Art

is frequently hard to obtain (see Keshav [82], ch. 9) RFB was proposed, first used probably by Golestani [57]. RFB is based on the maximum difference between the service received by any two flows under the policy being measured, see Zhou and Sethu [162] for relations between AFB and RFB. A similar criterion was suggested by Friedman and Henderson [50]. According to that criterion, a protocol $p$ is considered fair if it weakly dominates $PS$, namely no job completes later under $p$ than under $PS$, on any sample path. This criterion is similar to AFB in that it compares the protocol against $PS$, and it considers the worst case scenario, though it only classifies the protocol as fair or unfair. Most works in this area are based on the concept that General Processor Sharing (GPS) provides fair treatment to streams and thus policies “close” to GPS are fair as well. Some examples of early papers on the subject are Parekh [109], Parekh and Gallager [110], Rexford et al. [125], Bennet and Zhang [15]. Many other papers have been published on the subject.

Although these measures were originally meant to be used in studying flows and not specific jobs, they can be applied to jobs as well. For example, for AFB one can compute the maximum difference between the departure time of each job and the departure time it would have received under $PS$. However, when either of these measures is used for evaluating job fairness, the following emerges:

1. If job sizes are unbounded, these measures are unbounded (i.e. infinitely unfair) for all non-preemptive policies. In fact, the tightest bound possible for any non-preemptive policy is the size of the largest job, achieved by the Fair Queueing policy proposed in Demers et al. [43, 44].

2. Even if job sizes are bounded, it is easy to see that these measures do not differentiate between many non-preemptive service policies. For example, both FCFS and LCFS are equally and infinitely unfair, and so are most other non-preemptive policies.

Both cases above imply that these measures, based on a maximal-difference approach are not sensitive enough to differentiate between many popular scheduling policies which drastically differ from each other.
In the area of load balancing, Wang and Morris [149] propose the \(Q\)-factor. The \(Q\)-factor measures for a load \(\rho\) the mean response time of the worst customer source under the worst possible combination of stream loads, compared to the mean response time under global FCFS (namely FCFS among the customers of all sources). It therefore describes how closely a system comes to a multi-server FCFS system, as seen by every job stream.

Another area where related works are starting to appear is parallel job schedulers. Sabin et al. [131] propose a notion in which a job has been treated unfairly if its start time is delayed because of a later arriving job, not taking the job size into account. The results in this work are based on simulation. The work in Sabin and Sadayappan [130] provides another two unfairness metrics, one based on Gordon [58] (discussed above), and one based on RAQFM. The results in this work were also based on simulation. Vasupongayya and Chiang [148] use the variance, and other simple summary metrics, of some simplistic performance metrics e.g. the waiting time or the job rate. Like the above works, the results are based on simulation.

2.3 Fairness Measures in General

Fairness measures are also abundant in areas far from computer science.

In the field of economics, one measure worth mentioning is the well known Gini Coefficient, published as early as 1912 in Gini [56]. The Gini Coefficient is a measure of inequality of a distribution of income. It is formally defined as \(1 - 2 \int_0^1 L(x)dx\), where \(L(x)\) is the function representing the Lorenz curve (Lorenz [95]). In some cases it can also be calculated without direct reference to the Lorenz curve, see Gastwirth [54], Dorfman [45]. We have tried to apply the Gini Coefficient to our model, but as the “income” from the system is unclear, and varies with time this proved to be too complicated. However, the principles are not entirely unconnected.

Although strictly speaking the Fairness Index proposed in Jain et al. [73] was suggested in the context of computer systems, and was motivated by the research in the area of flow control, it is aimed at fairness of resource sharing in general. The four desired properties
of such an index, according to the authors, are (1) Population size independence, (2) Scale and metric independence, (3) Boundedness between 0 (totally unfair) and 1 (totally fair) and (4) Continuity. Assuming a vector of basic metrics $\mathbf{x} = (x_1, \ldots, x_n), x_i \geq 0$ their measure is

$$d(\mathbf{x}) = \frac{(\sum_{i=1}^{n} x_i)^2}{n \sum_{i=1}^{n} x_i^2}.$$  

By choosing the right basic metrics the same index can be applied to any setting. Algorithms for optimizing the Fairness Index are studied in Raftopoulou et al. [114].

Another field in which evaluating fairness is of interest is consumer reaction, where measurement is usually done through customer satisfaction surveys. Carr [29] proposes FAIRSERV - a model for evaluating consumer reactions to services based on a multidimensional evaluation of service fairness. Fairness is evaluated through five distinct fairness constructs, which are discussed in the fairness literature: distributive, procedural, interpersonal, informational, and systemic or overall fairness. All involve perceptions of the fairness of specific entities and activities.
Chapter 3

MODEL AND NOTATION

In this work we address quite a number of models, each requiring its own notation. To ease the reading, most of the notation related to each of the models is presented when the model is introduced. In this chapter we only summarize some of the common elements in these models, and the general notation practices we use. We also describe the common service policies investigated in this work.

3.1 Customers and Scenarios

Consider a system which is subject to the arrival of a stream of customers, \( C_1, C_2, \ldots \), who arrive at this order. Let \( s_i, a_i, d_i \), denote the service requirement measured in time units, the arrival epoch, and the departure epochs of \( C_i \), respectively. Specific series of values are called an arrival pattern (\( \{a_i\}_{i=1,2,\ldots,L} \)), an arrival and service pattern (\( \{a_i, s_i\}_{i=1,2,\ldots,L} \)), and a scenario (\( \{a_i, s_i, d_i\}_{i=1,2,\ldots,L} \)). Applying a service policy on an arrival and service pattern creates a scenario. We limit our discussion to arrivals which are independent of the system state and server decisions.

At each epoch \( t \) the system grants service at rate of \( s_i(t) \geq 0 \) to \( C_i \). The system is work-conserving, i.e. \( \int_{a_i}^{d_i} s_i(t) dt = s_i \). The total service rate given by the system is denoted \( s(t) \overset{\text{def}}{=} \sum_i s_i(t) \geq 0 \). Let \( N(t) \) denote the number of customers in the system at epoch \( t \).

Unless otherwise specified, all customers are “born equal”, belonging to the same class of customers, and thus no weights are assigned to them.

Throughout this work we use \( \rho \) in its common definition as the system utilization.
factor, which is \( \rho \overset{\text{def}}{=} \lambda \bar{x}/M \) where \( \lambda \) is the average arrival rate, \( \bar{x} \) is the average service requirement, and \( M \) is the number of servers (e.g. Kleinrock [88], Sec. 2.1). For stability, we usually assume \( \rho < 1 \).

3.2 Classes of Service

Define \( \Phi \) to be the class of non-preemptive, non-divisible service policies, i.e. service policies where once the server started serving a customer it will not stop doing so until the customer’s service requirement is fulfilled, and at most one customer is served at any epoch by each server.

Define \( \Phi^* \) to be a subclass of \( \Phi \) including only policies where the scheduler does not account for the actual values of the service requirements in the service decisions. Note that service policies may depend on the whole history of service, and particularly on how many jobs are in the system. On the other hand, if customers are assumed to all belong to the same class, policies belonging to \( \Phi^* \) cannot distinguish between customers at all based on size, not even by broad classifications such as ‘small’ and ‘large’.

3.3 General Notation Practices

We use **Typewriter-Style** to denote scheduling policies.

For convenience, we sometimes use the notation \( X^{(i)} \) to denote the \( i \)-th moment of \( X \), i.e. \( X^{(i)} \overset{\text{def}}{=} \mathbb{E}\{X^i\} \). We also use bar \( (\bar{X}) \) to denote expected value and hat \( (\hat{X}) \) do denote variance.

We use \( 1(\phi) \) to denote the indicator function, i.e.

\[
1(\phi) = \begin{cases} 
1 & \text{the expression } \phi \text{ is true} \\
0 & \text{otherwise}
\end{cases}
\]

We use **Bold – Style** to denote vectors, e.g. \( X \).

We use angled brackets, e.g. \( \langle a, b \rangle \), to denote a state of a system.
3.4 Common Service Policies

In this section we define some common service policies used in this work, and mention some commonly known performance results.

**PS: Processor Sharing** Under PS the processor is shared among all jobs currently in the system, in equal shares.

**FCFS: First Come First Served** Under FCFS the jobs are served in the order of arrival. Also sometimes referred to as FIFO for First In First Out.

**LCFS: Last Come First Served** Under LCFS the jobs are served in the opposite order of arrival, but without preemption. Also sometimes referred to as LIFO for Last In First Out.

**P-LCFS: Preemptive Last Come First Served** Under P-LCFS the jobs are served in the opposite order of arrival, and an arriving job preempts the currently served job.

**SRPT: Shortest Remaining Processing Time First** Under SRPT at every moment of time, the server is processing one of the jobs with the shortest remaining processing time (usually chosen in random if more than one such job is available). The SRPT policy is well-known to be optimal for minimizing mean response time (Schrage and Miller [139], Schrage [138]).

**LAS: Least Attained Service** Under LAS the job which received the least service gets the processor to itself. If several jobs all have the least attained service, they time-share the processor via PS. Also sometimes referred to as FB for Forward-Backward or FeedBack, or SET for Shortest Elapsed Time. See Nuyens and Wierman [103] for a recent survey of history and results.

**LRPT: Longest Remaining Processing Time First** Under LRPT at every moment of time, the server is processing the job with the longest remaining processing time. If multiple jobs in the system have the same remaining processing time, they time-share the processor via
PS. Note that this means that a job can only leave the system at the latest time possible, which is the end of the busy period. Thus it is probably the worst service policy possible in terms of efficiency.

**ROS**: Random Order of Service  Under **ROS** whenever the server becomes idle, one of the customers in the system is chosen for service in a uniformly random way.

**P-ROS**: Preemptive Random Order of Service  Under **ROS** whenever the server becomes idle, or a new customer arrives, one of the customers in the system is chosen for service in a uniformly random way.

**SJF**: Shortest Job First  Under **SJF** the jobs are served in order of service requirement, starting with the smallest, without taking into account the amount of service already given.

**LJF**: Longest Job First  Under **LJF** the jobs are served in order of service requirement, starting with the largest, without taking into account the amount of service already given.

**P-SJF**: Preemptive Shortest Job First  Under **P-SJF** the jobs are served in order of service requirement, starting with the smallest, without taking into account the amount of service already given. An arriving job preempts the currently served job if it has shorter service requirement.

**P-LJF**: Preemptive Longest Job First  Under **P-LJF** the jobs are served in order of service requirement, starting with the largest, without taking into account the amount of service already given. An arriving job preempts the currently served job if it has longer service requirement.

**RR**: Round Robin  Under **RR** customers are served in a cyclic manner, where in each cycle each customer receives a service quantum $\Delta$. To avoid system idling it is usually assumed...
that service requirements are integer multiples of $\Delta$. 
Chapter 4

INTRODUCING THE FAIRNESS MEASURE: RAQFM

In this chapter we introduce the fairness measure we propose, RAQFM: a Resource Allocation Queueing Fairness Measure.

Before proposing a fairness measure, one needs to ask the question, what is fairness? While almost every child, if asked, can tell you what is fair and what is not, it is quite a demanding undertaking to have a group of people agree on a common definition of fairness, much more so when it comes to defining a quantitative measure of the level of fairness.

Our approach in this is to consider the queueing system as a microcosm social construct. Its fairness should therefore conform to the general cultural perception of social justice in the particular society.

The issue of fairness and social justice has always been, and still is, a cardinal issue in all cultures. It is the cement holding the society together and as such it has been subject to debate by philosophers, prophets and spiritual leaders since the beginning of recorded history. Maybe one of the first relevant formulations of this issue is Aristotle’s idea in “Nicomachean Ethics”, that justice consists, at least in part, in treating equal cases equally, and unequal cases in proportional manner (Aristotle [4], Book V):

“Also, there will be the same equality between the persons and the shares: the ratio between the shares will be the same as that between the persons.”

In modern time, many economists and social scientists joined the ongoing debate. As is to be expected, there is a vast ocean of modern research and publications on this issue, mostly by philosophers, economists and social and behavioral scientists. It is of course be-
yond the scope of this work to review and interpret this literature, see some comprehensive reviews of the justice and fairness literature in Cropanzano [37], Beugré [16], Folger and Cropanzano [49], Cohen-Charash and Spector [33], Colquitt et al. [34], Cropanzano et al. [38]. However, a most prominent and comprehensive publication on this issue is Rawls’ book “A Theory of Justice” (Rawls [115]). In essence, Rawls’ general conception of social justice is (Rawls [115], p. 303):

“All social primary goods - liberty and opportunity, income and wealth, and the bases for self-respect - are to be distributed equally unless an unequal distribution of any or all of these goods is to the advantage of the least favored.”

We therefore see that throughout the debate, one prominent concept is that of fair resource allocation, or the idea that fairness is achieved when the resource ‘pie’ is appropriately divided between the consumers. But what is the pie in the case of a queueing system, and how should it be divided?

A very similar concept in the area of queueing is that of Processor Sharing, as embodied by the ideal PS policy, analyzed as early as Kleinrock [86, 87], Coffman et al. [32]. The root idea of this policy is that at every moment of time, the service rate is divided equally amongst the jobs present in the system.

This leads us to the basic principle, or belief, which is in the roots of our proposed measure:

At every epoch all jobs present in the system deserve an equal share of the system’s service rate. Deviations from it creates discriminations (positive or negative). Accounting for these discriminations and summarizing them yields a measure of unfairness.

The process for measuring RAQFM is therefore composed of two distinctive parts:

1. Discrimination: Each customer is given a single measure representing how well the customer was treated. A positive number means the customer was well treated, and a negative number means the customer was not treated well.
2. Fairness: A summary measure is taken over the discriminations. The (non-negative) result of this summary measure is the system’s unfairness, so a low measure means a more fair system.

4.1 RAQFM For Single Non-Idling Server

We start with the description of RAQFM for a single non-idling server, as first proposed in Raz et al. [117].

Observe a single server system, with a service rate of one unit. The server is non-idling, i.e. \( \forall t, N(t) > 0 \Rightarrow \sum_i s_i(t) = 1 \).

4.1.1 Individual Customer Discrimination

As the principle implies, at every epoch \( t \), all customers present in the system deserve an equal share of the system resources, or \( 1/N(t) \). We call this quantity the \emph{warranted service rate} of \( C_i \) at epoch \( t \), and denote it \( R_i(t) \). Integrating this for \( C_i \) yields \( R_i \equiv \int_{a_i}^{d_i} dt/N(t) \), the \emph{warranted service} of \( C_i \). The \emph{(overall) discrimination} of \( C_i \), denoted \( D_i \), is the difference between the warranted service and the granted service, i.e.

\[
D_i = s_i - R_i = s_i - \int_{a_i}^{d_i} \frac{1}{N(t)} dt
\]  

(4.1)

A positive (negative) value of \( D_i \) means that a customer receives better (worse) treatment than it fairly deserves, and therefore it is \emph{positively (negatively) discriminated}.

An alternative way to define \( D_i \) is to define the \emph{instantaneous discrimination rate} of \( C_i \) at epoch \( t \),

\[
\delta_i(t) \equiv s_i(t) - \frac{1}{N(t)}
\]

(4.2)

and then the overall discrimination of \( C_i \) is:

\[
D_i = \int_{a_i}^{d_i} \delta_i(t) dt.
\]

(4.3)
An important property of the discrimination is that it obeys, for every non-idling work-conserving system, and for every $t$: $\sum \delta_i(t) = 0$, that is, every positive discrimination is balanced by negative discrimination. This results from the fact that when the system is non-empty $\sum_i s_i(t) = 1$ (due to non-idling) and $\sum_i R_i(t) = N(t)(1/N(t)) = 1$. An important outcome is that if $D$ is a random variable denoting the discrimination of an arbitrary customer under steady state, then $\mathbb{E}\{D\} = 0$, namely the expected discrimination of an arbitrary customer under steady state is zero. A complete proof is given in Section 5.3.

The reason we use the total discrimination, rather than maybe the average instantaneous discrimination, i.e. the total discrimination divided by the sojourn time, is twofold. First, it represents better the effect a customer has on the system’s resources. A customer which takes a non-proportionally large share of the service rate for a long period of time, affects the system more than a customer which does the same for a short period of time. Second, it represents better the perception a customer has of the system. A customer receiving less than a fair share of the service rate for a long period of time is more likely to have a negative perception of the system than one receiving such unfair share for a short period of time.

4.1.2 System Measure of Unfairness

As mentioned above, to measure the unfairness of a system and of a service policy across all customers, that is, to measure the system unfairness, one would choose some summary statistics measure over $D$, where $D$ is a random variable denoting the discrimination of an arbitrary customer when the system is in steady state. Since fairness inherently deals with differences in treatment of customers, a natural choice is the variance of customer discrimination, $\hat{D}$. As $\mathbb{E}\{D\} = 0$, this equals the second moment, that is $\mathbb{E}\{D^2\}$, denoted $F_{D^2}$. Another optional measure is the mean of distances $\mathbb{E}\{|D|\}$, denoted $F_{|D|}$.

One can also choose to ignore customers with positive discrimination and use the mean value over customers with negative discrimination, $-\mathbb{E}\{D \mid D < 0\}$. However,
note that in some cases this might not be meaningful. For example suppose $D_1 = D_2 = \cdots = D_{19} = 1$ and $D_{20} = -19$, then $-\mathbb{E}\{D \mid D < 0\} = 19$. In other words, using the mean value over customers with negative discrimination might lead to very few customers influencing the unfairness in a major way. What might be more correct is the average negative discrimination, assuming positive discrimination does not count, but averaging over all customers, i.e. $\mathbb{E}\{\min(D,0)\}$ which in this case equals $19/20 = 0.95$. However, $\mathbb{E}\{\min(D,0)\} = \mathbb{E}\{\max(D,0)\} = \mathbb{E}\{|D|\}/2 = F_{\Delta}/2$.

Another approach is to use higher central moments, $\mathbb{E}\{(D - \mathbb{E}\{D\})^k\}$, which in this case are equal to the moments $\mathbb{E}\{D^k\}$. Alternatively one can use cumulant moments (see textbook Bailey [12] or examples of use Choudhury and Whitt [31], Duffield et al. [46], Matis and Feldman [99], Wierman and Harchol-Balter [154]). In fact, summary functions for variability or fairness have been suggested in other fields as well, see Chapter 2 e.g. the Fairness Index (Jain et al. [73]), and many of them will do. We believe that the choice of summary function is of less importance, and we choose to focus on the simplest ones, $F_{\Delta^2}$ and $F_{|D|}$. Throughout this work, the term “unfairness” refers to $F_{\Delta^2}$.

4.1.3 Unfairness of a Scenario

Recall that a scenario is defined as a specific series of values $(\{a_i, s_i, d_i\}_{i=1,2,\ldots,L})$. The unfairness of a scenario is also defined by some summary statistics measure over the set of individual discriminations $D_i, i = 1, \ldots, L$. A natural choice is the statistical variance of customer discrimination, and since the average of $D_i$ is zero, this equals the statistical second moment, that is $\frac{1}{L} \sum_{i=1}^{L} (D_i)^2$, denoted $F_{\Delta^2}$. Similarly other optional measure are $\frac{1}{L} \sum_{i=1}^{L} |D_i|$ (denoted $F_{|D|}$) or $\frac{1}{L} \sum_{i=1, D_i < 0}^{L} D_i$. We use the same notation used for system unfairness, $F_{\Delta^2}, F_{|D|}$, and the meaning will be obvious from the context.

Remark 4.1 (The relation between scenario unfairness and system unfairness). Note that system unfairness deals with the expectation over all scenarios, while scenario unfairness deals with the realization of a specific scenario. Thus, if for every arrival and service patterns policy $\phi_1$ has lower unfairness than policy $\phi_2$ then it also has lower system
unfairness.

### 4.2 Generalization of RAQFM for Multi-Server Systems

This following generalization was proposed in Raz et al. [122, 116, 121].

Consider a queueing system with $M$ servers. If we let $0 \leq \omega(t) \leq M$ denote the total service rate available at epoch $t$, then the fair share, or warranted service rate, is $R_i(t) = \omega(t)/N(t)$. We will discuss the choice of $\omega(t)$ momentarily.

Similarly to (4.2), the discrimination rate, $\delta_i(t)$ is

$$\delta_i(t) \triangleq s_i(t) - \frac{\omega(t)}{N(t)}, \quad (4.4)$$

and the total discrimination $D_i$, is the same as (4.3).

For work conserving systems we have, similarly to (4.1)

$$D_i = s_i - \int_{a_i}^{d_i} \frac{\omega(t)}{N(t)} dt.$$  

The choice of $\omega(t)$ depends on the specific application, and on what is perceived as the service rate available to customers. One simple option is to use $\omega(t) = M$, the number of servers. However, this option is less appropriate in many situation. Consider for example a multiple server system where only one job is present, and it is physically impossible to serve that job with more than one server. One can hardly say that the customer is negatively discriminated, yet if we use $\omega(t) = M$ we get $\delta_i(t) = 1 - M$. A second option is to use the total service rate physically granted to customers present at the system at that epoch, i.e. $\omega(t) = s(t)$. This choice has the benefit that it has zero expected value, as shown in Section 5.3 However, it might not account for all types of discrimination. Specifically it does not account for unfairness caused by explicitly changing the service rate. Consider for example a system where customers are always served alone, except some customers are served at full service rate while others at half service rate. If we take use $\omega(t) = s(t)$ we get $\delta_i(t) = 0$ for all customers, while clearly some customers are negatively discriminated.
Another option is to choose the number of servers which are physically capable of serving customers, even if they are currently idle, namely, \( \omega(t) = \min(M, N(t)) \). Of course this expression can change based on the physical capabilities of the system. While this might seem like an optimal choice it still isn’t always clear enough. For example, if one server is on maintenance, does it count as discrimination? As we would like to focus on the unfairness caused by the structure of the system, we choose to focus in this work on the second option, namely \( \omega(t) = s(t) \).

The choice of summary function is the same as for the measure for single server.

4.3 Generalization of RAQFM for Networks of Queues

While we don’t use the following generalization in this work, it is worth mentioning here for completeness. It can be used in future research on the fairness issues in networks of queues.

While the generalization in Section 4.2 is sufficient to address the operational mechanisms in which we are interested in this work, it cannot readily model more complex constructs such as tandem queues, which are quite common in the area of call and service centers. Networks of queues were used to model such complex structures of queues as early as the work of Burke [28] and Jackson [69], which considered networks of Exponential (Markovian) queues, see Kleinrock [88], Sec 4.8.

We propose three methods for extending RAQFM to networks of queues.

4.3.1 The Global Method

Using the global method we measure the instantaneous individual discrimination using the global state of the system, i.e. we use (4.4) where \( \omega(t) \) is the total amount of service given in the entire network at epoch \( t \), and \( N(t) \) is the total number of customers in the network at the same epoch. Customer discrimination is measured by integrating the instantaneous discrimination rate for the entire stay of the customer in the network, and system discrimination is measured by some summary function over the customer
discrimination.

While this seems like the obvious way to extend RAQFM note that it has some drawbacks. First, logically speaking, observe that in the simple multiple parallel server case, as described in Section 4.2, it made sense that the entire service rate is warranted to each customer, since the other queues/servers are generally “close by” and easy to observe. This is not the case in a large network of queues, and therefore it might not make sense that a customer in one corner of the network is warranted service which is given to customers in the far corner of the network. This is of course a matter of opinion, whether injustice that cannot be observed should be counted as such. Second, from a practical point of view, a customer is usually incapable of measuring the total number of customers in the system or the total rate of service given, so this measurement must be done by some overseeing entity. A customer is therefore incapable of knowing how well she was treated.

4.3.2 The Local Method

Using the local method we measure instantaneous individual discrimination using only the state of the queue in which the customer is currently residing, i.e. we use \( \omega(t) \) where \( \omega(t) \) is the amount of service given in the local queue in which the customer resides, and \( N(t) \) is the number of customers in the same queue. As is the case with the global method, customer discrimination is measured by integrating the instantaneous discrimination rate for the entire stay of the customer in the network, and system discrimination is measured by by some summary function over the customer discrimination.

The major benefits of using this method are exactly the drawbacks of the global method. As a customer is staying in a queue, it makes sense that said customer is warranted service given in that queue. It is also very practical for a customer to measure the amount of service given and the number of customers in the local queue.

The major drawback is that using this method several major forms of discrimination cannot be accounted for. Consider the following example:

**Example 1.** Consider a system with two parallel PS queues, \( q_1 \) and \( q_2 \). Assume customers
always arrive in bulks of three customers, and are assigned to the queues by the system. All customers have a service requirement of one unit, and both queues have one unit of service rate. Now observe a case where two customers were assigned to queue $q_1$ and one to $q_2$.

Clearly, the system is discriminating against the customers assigned to $q_1$. Using the global method, in the first unit of time, all customers are warranted a rate of $2/3$. Customers assigned to $q_1$ are granted a rate of $1/2$, and have a negative discrimination of $1/2 - 2/3 = -1/6$. The single customer at $q_2$ has a positive discrimination of $1 - 2/3 = 1/3$. Using the local method all customers have an instantaneous discrimination rate of zero, so this kind of discrimination isn’t accounted for.

Note that even the global method cannot always account for discrimination in the case where all customers are assigned to one queue, as this depends on the choice of $\omega(t)$, see discussion in Section 4.2 on the matter of the choice of $\omega(t)$.

### 4.3.3 The Decomposition Method

Using the decomposition method we assign each queue an unfairness measure in the way described in Section 4.1 or Section 4.2. We then attain the network’s unfairness through composition of the values. This can be achieved in the following way. We first assign each of the possible “walks” of a customer through the system an unfairness value, which is the sum of unfairness of the queues passed through on that walk. Then, we evaluate the expected value by weighing each walk with the probability of that walk. In another variant of the same method the weighing is based on the square of probabilities.

This method has one big advantage, which is that it is quite easy to evaluate stochastically, since evaluating the unfairness of a single server is not a very resource consuming process, as we will demonstrate in Chapter 8. Evaluating the network unfairness from this is quite straightforward.

However, as expected, this method does not account correctly for unfairness that a customer experiences when going through the system. First, observe that in Example 1
the decomposition method also assigns the network zero unfairness. To demonstrate the difference between the decomposition method and the local method, consider the following case.

**Example 2.** Consider a system with two queues, $q_1$ and $q_2$, working in tandem. $q_1$ is a FCFS queue with a service rate of one unit and $q_2$ is a P-LCFS queue with a service rate of $1/(1 + \epsilon)$, where $\epsilon \to 0$. All customers have service requirement of one unit from each of the servers, and they arrive in pairs every three units of time.

Starting with the local method, note that in every pair of customers one customer will be served first by $q_1$, w.l.o.g. say $C_1$. The discrimination accumulated in $q_1$ for $C_1$ and $C_2$ is 0.5 and $-0.5$, respectively. However, as $C_1$ arrives in $q_2$ it is served for one unit of time, with zero discrimination, and then preempted by $C_2$ with $\epsilon$ remaining service. It then collects a negative discrimination of $-(1 + \epsilon)/2$. The total accumulated discrimination for $C_1$ and $C_2$ is $-\epsilon/2$ and $\epsilon/2$, respectively, which tends to zero when $\epsilon \to 0$, so the network has zero unfairness when $\epsilon \to 0$. Using the decomposition method, $q_1$ and $q_2$ have an unfairness of $(0.5)^2 + (-0.5)^2 = 0.5$ and $(0.5 + \epsilon/2)^2 + (-0.5 + -\epsilon/2)^2 = 0.5 + \epsilon + \epsilon^2/2$ respectively. The total network unfairness is $1 + \epsilon + \epsilon^2/2 \xrightarrow{\epsilon \to 0} 1$. This demonstrates that the local method and the decomposition method can yield significantly different results. Note that the global method also attains zero unfairness in this case.
Chapter 5

BASIC PROPERTIES OF RAQFM

In this chapter we study the basic general properties of RAQFM. Most of these properties were published in Raz et al. [117], Avi-Itzhak et al. [11] and also appears in Raz et al. [118].

5.1 Sensitivity of the Measure to Seniority and Service Requirement Differences

In this section we demonstrate the sensitivity of RAQFM to both seniority differences and service requirement differences. As discussed in the introduction, we believe that reacting correctly to both seniority and service requirement differences is an important property for any fairness measure. We do this by first analyzing basic dilemma regarding the prioritization of Short vs. Long presented in the introduction and considering an example that is a special case of that problem. We then consider the same problem under more general conditions.

5.1.1 The Mr. Short vs. Mrs. Long Example

Consider the following arrival and service pattern. $C_0$, serving as a “bystander” for this analysis, arrives at the empty system at $a_0$ and leaves at $d_0$. At $a_0 < a_L < d_0$ arrives $C_L$ (Long) with a service requirement of $s_L$. At $a_L < a_S < d_0$ arrives $C_S$ (Short) with a service requirement of $s_S < s_L$. We now ask the question, whether it is more fair, under RAQFM, to serve $C_S$ ahead of $C_L$.

Let $X$ denote the length of the interval $(a_0, a_L) \ X = a_L - a_0$. Let $R$ denote the
remainder of the service requirement left for \( C_0 \) after \( a_S \), i.e. \( R = d_0 - a_S \). Let \( \Delta_a \) denote the difference in seniority between \( C_S \) and \( C_L \), i.e. \( \Delta_a = a_S - a_L \). Let \( \Delta_s \) denote the difference in service requirement between \( C_S \) and \( C_L \), i.e. \( \Delta_s = s_L - s_S \).

Figure 5.1 illustrates the two possible orders of service (scenarios). In the top half of the figure \( C_L \) is served before \( C_S \), and in the bottom half the opposite.

If \( C_L \) is served ahead of \( C_S \), then \( C_L \) leaves the system at \( d_L = d_0 + s_L \) and \( C_S \) leaves the system at \( d_S = d_0 + s_S + s_L \). The discriminations of the three customers are therefore

\[
D_0 = (X + \Delta_a + R) - \left( X + \Delta_a/2 + R/3 \right) \\
D_L = s_L - \left( \Delta_a/2 + R/3 + s_L/2 \right) \\
D_S = s_S - \left( R/3 + s_L/2 + s_S \right),
\]

and the total unfairness is \((D_0^2 + D_L^2 + D_S^2)/3\).

If \( C_S \) is served ahead of \( C_L \), then \( C_L \) leaves the system at \( d'_L = d_0 + s_S + s_L \) and \( C_S \) leaves the system at \( d'_S = d_0 + s_S \). The discriminations of the three customers are
therefore

\[ D'_0 = (X + \Delta a + R) - (X + \Delta a/2 + R/3) \]

\[ D'_L = s_L - (\Delta a/2 + R/3 + s_S/2 + s_L) \]

\[ D'_S = s_S - (R/3 + s_S/2) \]

and the total unfairness is \(((D'_0)^2 + (D'_L)^2 + (D'_S)^2)/3\).

The difference between the fairness values of the two possible scenarios is therefore

\[ \frac{(D'_0)^2 + (D'_L)^2 + (D'_S)^2 - D_0^2 + D_L^2 + D_S^2}{3} = \frac{(s_S + s_L)(\Delta a - \Delta s)}{6}. \]  (5.1)

Expression (5.1) now reveals that the unfairness difference in this case is monotonic in the difference between the seniority difference and the service requirement difference, that is in \(\Delta a - \Delta s\). The higher this difference is the more unfair it is to serve Short first; the lower it is, the more unfair it is to serve Long first. The point of indifference is exactly when these differences equal each other, that is \(\Delta a = \Delta s\).

It is clear from this scenario that RAQFM accounts for both seniority difference and service requirement difference, and in a proper manner.

5.1.2 Reaction to Differences in Seniority

We now move on to show that for a commonly encountered class of policies, RAQFM reacts well to seniority differences. We do this by showing that in the special case where service requirements are identical, either deterministically or stochastically, RAQFM “prefers” serving in order of seniority. The model we deal with is a single non-idling server.

**Theorem 5.1** (Preference of Seniority). Let \(a_1, a_2, \ldots, a_L\) be an arbitrary (deterministic) arrival pattern where \(a_j < a_k\), \(1 \leq j, k \leq L\). Let \(\{s_i\}\), \(i = 1, 2, \ldots, L\), \(i \neq j, k\) be an arbitrary (deterministic) set of service requirements. Let \(S_j\) and \(S_k\), the service requirements of \(C_j\) and \(C_k\), respectively, be random variables. Consider two possible orders of service.
In order of service $\phi$, $C_j$ is served ahead of $C_k$, and the opposite in $\phi'$. Let $F(\phi), F(\phi')$ denote the unfairness in order of service $\phi$ and $\phi'$ respectively. We assume that both of the schedules are feasible, i.e. that $C_j$ and $C_k$ reside in the queue when the first of them is picked for service, and thus are interchangeable. Then if $S_j$ and $S_k$ are i.i.d. then $\mathbb{E}\{F(\phi)\} \leq \mathbb{E}\{F(\phi')\}$, where expectations are taken over the joint distribution of $S_j$ and $S_k$.

Proof. Denote the service requirement of the first job to be served, either $C_j$ or $C_k$, by a random variable $S_1$ and the service requirement of the second one by $S_2$. Similarly, $h_1, h_2$ denote the times the first and second jobs go into service. Note that $h_2$ depends on the first service requirement $S_1$, so we denote it $h_2(S_1)$. Further note that the first job leaves at $h_1 + S_1$ and the second job at $h_2(S_1) + S_2$.

Let $\hat{F}(\phi)$ denote the total unfairness under $\phi$ due to $C_j$ and $C_k$, and $\hat{F}(\phi)$ denote the total unfairness under $\phi$ due to all customers other than $C_j$ and $C_k$. Then $F(\phi) = \hat{F}(\phi) + \hat{F}(\phi)$. Define similar notations for $\phi'$.

Note that since $S_1$ and $S_2$ are independent of the order of service of $C_j$ and $C_k$, the distribution of the unfairness of any of the other jobs is not influenced by that order, i.e. $\hat{F}(\phi) = \hat{F}(\phi')$ and thus $F(\phi) - F(\phi') = \hat{F}(\phi) - \hat{F}(\phi')$, where the equalities are in distributions.

Define $I(a, b) \overset{def}{=} \int_a^b \frac{1}{N(t)} \, dt$. Now note that

$$\hat{F}(\phi) = (S_1 - I(a_j, a_k) - I(a_k, h_1 + S_1))^2 + (S_2 - I(a_k, h_2(S_1) + S_2))^2,$$

and similarly,

$$\hat{F}(\phi') = (S_1 - I(a_k, h_1 + S_1))^2 + (S_2 - I(a_j, a_k) - I(a_k, h_2(S_1) + S_2))^2.$$

Thus,

$$\hat{F}(\phi) - \hat{F}(\phi') = -2I(a_j, a_k) (S_1 - S_2 - I(a_k, h_1 + S_1) + I(a_k, h_2(S_1) + S_2))$$

$$= -2I(a_j, a_k) (S_1 - S_2 + I(h_1 + S_1, h_2(S_1) + S_2)).$$
taking expectations, and since \( S_1 \) and \( S_2 \) are drawn from the same distribution, and \( I(\cdot) \) is always positive, \( \mathbb{E}\{F(\phi)\} - \mathbb{E}\{F(\phi')\} \leq 0 \).

**Remark 5.1.** In the special case where \( S_1 = S_2 \), i.e. both customers have equal service requirements, \( F(\phi) - F(\phi') \leq 0 \) deterministically, even without taking expectations.

**Remark 5.2.** Theorem 5.1 can also be proven using sample path arguments, considering every pair of mirroring realizations for \( S_j \) and \( S_k \). Such a proof is stronger, since it proves the theorem for every pair of mirroring realizations, and not only in expectation, i.e. it shows that for every realization \( S_j = \tau_1 \) and \( S_k = \tau_2 \) and mirror realization \( S_j = \tau_2 \) and \( S_k = \tau_1 \), the sum of the unfairness for the two realizations is always smaller when the order of service follows seniority. However, the above proof is sufficient for our needs, and is much shorter. We bring the somewhat stronger proof in Appendix A.

**Corollary 5.1 (Fairness of FCFS).** Consider a system with arbitrary arrivals.

If the service requirements of all customers are identical then: (i) For every arrival pattern FCFS (LCFS) will create the scenario with the lowest (highest) scenario unfairness in \( \Phi^* \) for that arrival pattern, and (ii) FCFS (LCFS) has the lowest (highest) system unfairness in \( \Phi^* \).

If the service requirements are i.i.d random variables with arbitrary known distribution. Then: (iii) FCFS (LCFS) is the service policy with the lowest (highest) expected system unfairness in \( \Phi^* \).

**Proof.** (i) follows from Remark 5.1 using the following contradiction argument. Assume for the contradiction that there exists an arrival pattern and a service policy \( \phi \in \Phi^* \), \( \phi \neq FCFS \), which is the least unfair policy in \( \Phi^* \) for this arrival pattern. Then the order of service created by \( \phi \) for this unfair pattern is different than the order of service created by FCFS, otherwise \( \phi \) is indistinguishable from FCFS.

Given this arrival pattern and the order of service created by \( \phi \), identify the first pair of customers which are adjacently served and are not served according to their order of arrival. Interchange the order of service between these two customers, which is certainly
possible since the service policy is non-preemptive. According to Remark 5.1, the result of this interchange is a decrease in the overall unfairness. Thus the resulting order of service is more fair than $\phi$, in contradiction to $\phi$ being the least unfair service policy for this arrival pattern.

The proof of (ii) follows from Remark 4.1. Similar arguments prove the claims for LCFS.

(iii) is immediate from Theorem 5.1 using a similar contradiction argument.

5.1.3 Reaction to Differences in Service Requirement

In this section we show that RAQFM reacts well to service requirement differences. We demonstrate this in the case where the arrival times of all customers are identical, i.e. all customers arrive simultaneously.

**Theorem 5.2** (Preference of Shorter Service Requirement for Simultaneously Arriving Customers). Let $C_i, i = 1, \ldots, L$ be $L$ customers arriving simultaneously (i.e. $\forall i, a_i = a$) at an empty single non-idling server. Assume that no arrivals occur between $a$ and the departure epoch of the last customer $a + \sum_1^L s_i$. Then, for any two customers $C_i, C_j$ such that $s_i < s_j$, it is more fair to serve $C_i$ before $C_j$.

**Proof.** For simplicity of presentation, and w.l.o.g, assume that customer index follows the customer’s service order, in opposite order, namely $C_i$ is served after $C_{i+1}$, $i = 1, 2, \ldots, L - 1$. The discrimination experienced by the $n$-th customer served is

$$D_n = s_n - \sum_{i=n}^{L} \frac{s_i}{i}.$$ 

The unfairness of the scenario is

$$F_{D2} = \frac{1}{L} \sum_{n=1}^{L} \left( s_n \frac{n-1}{n} - \sum_{i=n+1}^{L} \frac{s_i}{i} \right)^2.$$
To evaluate this expression we first evaluate the terms involving $s_n^2$. These yield

$$s_n^2 \left( \frac{n-1}{n} \right)^2 + \sum_{i=1}^{n-1} \left( \frac{s_n}{n} \right)^2 = \frac{n-1}{n} s_n^2.$$

Next consider the terms in the sum involving $s_n s_k$, $n < k$. These yield

$$-2 \frac{s_n(n-1)}{n} \frac{s_k}{k} + \sum_{i=1}^{n-1} \frac{2s_n s_k}{n-k} = 0.$$

To summarize, the unfairness of the scenario, reduces to

$$\frac{1}{L} \sum_{n=1}^{L} \frac{n-1}{n} s_n^2.$$

Thus, the proof follows from the fact that $\frac{n-1}{n}$ is monotone increasing in $n$. 

**Remark 5.3.** Note that from the proof above, if the service requirements are random variables, then serving the customers in increasing order of second moments of service requirement, yields the lowest unfairness in expectation.

**Corollary 5.2.** It follows immediately from Theorem 5.2 that for a single non-idling server and an arrival and service pattern consisting only of $N$ simultaneously arriving customers, the most fair policy in $\Phi$ is SJF and the least fair policy in $\Phi$ is LJF.

**Remark 5.4.** The advantage of serving a shorter service requirement customer $C_i$ ahead of a a longer service requirement customer $C_j$, as in Theorem 5.2 holds when arrival times of all customers are identical, and does not necessarily hold when only two customers arrive simultaneously, say $a_i = a_j$. For example, consider the arrival and service pattern \{(a_i, s_i)\}_{i=1}^{5} = \{(0,3), (1,1), (1,2), (3,1), (6,1000)\} and compare the service orders: (i) 1, 2, 3, 5, 4 (ii) 1, 3, 2, 5, 4. Though $a_2 = a_3$ and $s_2 < s_3$, the unfairness of service orders (i) and (ii) are roughly 83556 and 83528, respectively, namely the second is more fair.
To explain this behavior note that if one considers only the pair of customers exchanged, their total unfairness is reduced by serving the shorter first. However, other customers may also be influenced by this change, resulting in a total increase in unfairness. This can be viewed as a weakness of RAQFM.

5.2 Absolute Fairness of PS

Theorem 5.3 (Absolute Fairness of PS). In the single non-idling server case, for any arrival and service pattern, a scheduling policy has zero unfairness if and only if the departure epochs of all customers are identical to those under PS.

Remark 5.5 (PS Imitators). A preemptive policy can schedule its processing in a way that the departure epochs of all customers are identical to those in PS, even if the scheduling is not identical to PS at every epoch. We call such a policy a “PS Imitator”. We conjecture that in order to execute PS imitation a scheduler must know the exact service requirements and arrival epochs of the customers present in the system and future arrivals.

Proof of Theorem 5.3 First, PS has zero unfairness from the fact that for PS $s_i(t) = 1/N(t)$, thus according to the definition of $\delta_i(t)$ in (4.2) $\delta_i(t) = s_i(t) - 1/N(t) = 0$. Using (4.3)

$$ D_i = \int_{a_i}^{d_i} \delta_i(t) dt = 0 \Rightarrow E[D^2] = 0. $$

Second, to consider PS imitators, observe that given the arrival epochs $a_i$, each discrimination value, and therefore the unfairness, are functions only of the departure epochs $d_i$ and of $N(t)$. Thus, a scheduling policy that has departure epochs equal to that of PS for the same arrival and service pattern has the same discrimination values, and the same unfairness of PS.

Third, we prove the other direction of the theorem by way of contradiction. Assume for the contradiction that there exists an arrival and service pattern and a scheduling policy $\phi$, with departure epochs that are not equal to those of PS, and that the resulting
scenario has zero unfairness. Observe the first departure that is different from a departure according to PS, say the departure of \( C_k \). Denote the departure epoch according to PS and according to \( \phi \) by \( d_k \) and \( d'_k \) respectively, where \( d_k \neq d'_k \). Denote the discrimination of \( C_k \) according to PS and according to \( \phi \) by \( D_k \) and \( D'_k \) respectively. From the assumption \( \mathbb{E}\{(D')^2\} = 0 \) and thus we must have \( D'_k = 0 \). Denote the number of customers in the system at epoch \( t \) according to PS and according to \( \phi \) by \( N(t) \) and \( N'(t) \) respectively. We have from [4.1]

\[
D_k = s_k - \int_{a_k}^{d_k} dt/N(t) = 0.
\]

Suppose \( d_k > d'_k \), then all departures up to \( d'_k \) are the same for PS and for \( \phi \), and therefore \( \forall t < d'_k, N'(t) = N(t) \). Thus,

\[
D'_k = s_k - \int_{a_k}^{d'_k} dt/N'(t) = s_k - \int_{a_k}^{d'_k} dt/N(t) = s_k - \left( \int_{a_k}^{d_k} dt/N(t) - \int_{d'_k}^{d_k} dt/N(t) \right) = \int_{d'_k}^{d_k} dt/N(t) \geq 0,
\]

as \( N(t) \geq 1 \) in \( (d'_k, d_k) \) since \( C_k \) is in the system. Thus, the assumption is contradicted.

Now suppose \( d_k < d'_k \), then all departures up to \( d_k \) are the same for PS and for \( \phi \), and therefore \( \forall t < d_k, N'(t) = N(t) \). Thus,

\[
D'_k = s_k - \left( \int_{a_k}^{d_k} dt/N'(t) + \int_{d_k}^{d'_k} dt/N'(t) \right) = s_k - \left( \int_{a_k}^{d_k} dt/N(t) + \int_{d_k}^{d'_k} dt/N(t) \right) = -\int_{d_k}^{d'_k} dt/N'(t) < 0,
\]

again contradicting the assumption.

**Theorem 5.4** (Absolute Fairness in Multiple Server Systems). The above theorem also applies to (i) PS in multiple server systems and (ii) the G/G/\( \infty \) model.
Proof. (i) When applying the above theorem to multiple server systems one needs to first define its exact operation. One ideal way to define PS in a system with \( M \) servers is that if \( N(t) \leq M \) then each customer is served by one server. Otherwise, each customer receives a service rate of \( M/N(t) \). Thus

\[
s_i(t) = \begin{cases} 
1 & N(t) \leq M \\
\frac{M}{N(t)} & N(t) > M 
\end{cases},
\]

and

\[
\omega(t) = \begin{cases} 
N(t) & N(t) \leq M \\
M & N(t) > M 
\end{cases}.
\]

Using the definition of \( \delta_i(t) \) in (4.4)

\[
\delta_i(t) = s_i(t) - \frac{\omega(t)}{N(t)} = \begin{cases} 
1 - \frac{N(t)}{N(t)} = 0 & N(t) \leq M \\
\frac{M}{N(t)} - \frac{M}{N(t)} = 0 & N(t) > M 
\end{cases} = 0.
\]

(ii) in the G/G/\( \infty \) mode \( s_i(t) = 1 \) and \( \omega(t) = N(t) \). Using (4.4) \( \delta_i(t) = 1 - N(t)/N(t) = 0 \).

The rest of the proof, in both cases, is identical to the proof of Theorem 5.3.

Note that this does not depend on the system being non-idling or work-conserving.

5.3 Zero Sum

**Theorem 5.5** (Zero Expected Value). In a stationary system, the expected value of discrimination always obeys \( \mathbb{E}\{D\} = 0 \).

Proof. Observe that the total momentary discrimination rate at any epoch \( t \) is

\[
\sum_{\{i|a_i<t<d_i\}} \delta_i(t) = \sum_{\{i|a_i<t<d_i\}} \left( s_i(t) - \frac{\omega(t)}{N(t)} \right) = \omega(t) - N(t) \frac{\omega(t)}{N(t)} = 0, \tag{5.2}
\]

where the first equality is from the definition in (4.4) and the second is due to the fact that the sum of the service rates given to all the customers equals the total rate of service.
given, and the fact that there are exactly $N(t)$ customers in the system at epoch $t$, i.e. exactly $N(t)$ customers satisfy $a_i < t < d_i$. We have

$$\mathbb{E}\{D\} = \lim_{L \to \infty} \frac{1}{L} \sum_{i=1}^{L} D_i = \lim_{L \to \infty} \frac{1}{L} \sum_{i=1}^{L} \int_{a_i}^{d_i} \delta_i(t) dt = \lim_{L \to \infty} \frac{1}{L} \int_{0}^{\infty} \sum_{\{i|a_i < t < d_i\}} \delta_i(t) dt = 0,$$

where the first equality is due to the ergodicity assumption, the second is from (4.3), the third is due to changing the order of summation, and the forth is due to (5.2).

Note that this theorem applies to work conserving system, as well as systems which are not. It also does not depend on the system being non-idling.

5.4 Bounds of RAQFM

We now derive bounds on the discrimination and unfairness values of RAQFM in the single non-idling server setting. The importance of bounds is that they provide a scale of reference. Such scale of reference is useful when the measure is used to evaluate a system and some intuitive meaning of the fairness numbers is needed.

5.4.1 Bounds On Individual Discrimination

**Theorem 5.6.** For every scenario and every customer $C_i$, 

$$-\frac{W_i}{2} \leq D_i < s_i(1 - \frac{1}{N_{\text{max}}}),$$

where $W_i \overset{def}{=} d_i - a_i - s_i$, is the waiting time of $C_i$, and $N_{\text{max}}$ is the maximum number of customers allowed in the system. Both bounds are tight.

**Proof.** For the upper bound we have from (4.1),

$$D_i = s_i - \int_{a_i}^{d_i} \frac{1}{N(t)} dt < s_i - s_i \frac{1}{N_{\text{max}}} = s_i(1 - \frac{1}{N_{\text{max}}}).$$
To calculate the lower bound note that from (4.1) the discrimination is minimal when \( N(t) \) is minimal. The minimal possible \( N(t) \) is one for a period of length \( s_i \), when \( C_i \) is served with full service ratio, and two for the remaining sojourn time, i.e. for a period of length \( W_i \). Therefore
\[
D_i \geq s_i - (s_i + \frac{W_i}{2}) = -\frac{W_i}{2}.
\]

To show tightness of the upper bound consider a customer finding \( N_{\text{max}} - 1 \) customers in the system upon arrival, and being served without waiting, at full rate, as can happen in the P-LCFS policy. The discrimination rate of this customer is a constant \( 1 - 1/N_{\text{max}} \) and the discrimination is \( D_i = (1 - 1/N_{\text{max}})s_i \).

To show tightness of the lower bound consider the last customer in a FCFS busy period, who encounters exactly one customer in the system upon arrival.

Note that a customer may encounter a negative discrimination of \(-W_i/2\) whose value is unbounded, even if service requirements are all bounded. This occurs to a customer who arrives to a LCFS served system with a single customer in service, and who encounters a (possibly unbounded) sequence of arrivals occurring exactly at service completion epochs.

Note that both of these extremes tend to happen with LCFS or P-LCFS. This agrees with one’s intuition expecting these policies to have low fairness in some cases.

5.4.2 Bounds on Scenario and System Fairness

**Theorem 5.7.** Let \( N_{\text{max}} \) be the maximum number of customers allowed in the system. For every scenario let \( L \) and \( s_{\text{max}} \) be the number of customers and the maximal service requirement in that scenario, respectively. Then
\[
0 \leq F_{|D|} \leq 2s_{\text{max}}^*
\]
\[
0 \leq F_{D^2} < \frac{L}{2}(s_{\text{max}}^*)^2,
\]
where \( s^*_\text{max} \overset{\text{def}}{=} s_{\text{max}}(1 - 1/N_{\text{max}}) \). The lower bounds are both tight. The upper bound for \( F_{|D|} \) is tight when \( L \to \infty \).

Note that if one chooses the unit of time to be \( s_{\text{max}} \), the upper bounds are 2 and \( L/2 \).

**Proof.** The lower bounds, including their tightness, were shown in Theorem 5.3. The upper bounds can be derived by maximizing \( 1/L \sum_{i=1}^{L} |X_i| \) and \( 1/L \sum_{i=1}^{L} (X_i)^2 \) under the constraints

\[
-s_{\text{max}} \frac{L - 1}{2} \leq X_i \leq s^*_\text{max},
\]

\[
\sum_{i=1}^{L} X_i = 0.
\]

Constraint (5.3) reflects the bounds on the individual discriminations according to Theorem 5.6 and constraint (5.4) reflects the equality \( \mathbb{E}\{D\} = 0 \) from Section 5.3.

For \( L = 1 \), \( F_{|D|} = F_{D^2} = 0 \). For \( L = 2 \) the maximal absolute value of the discrimination is \( s_{\text{max}}/2 \) and the upper bound follows. For \( L \geq 3 \) one of the global maxima is achieved at \( X_1 = -(L - 1)s^*_\text{max}/2, X_2 = -(L - 3)s^*_\text{max}/2, X_i = s^*_\text{max} \) for \( i > 2 \) (other global maxima exist, for example, due to symmetry between the variables, or for \( F_{|D|} \) every \( 0 \leq X_1, X_2 \leq N - 1 \) where \( X_1 + X_2 = -(N - 2) \)). The maximum value for \( F_{|D|} \) is

\[
\max |D| = \frac{1}{L} \left( (L - 1) \frac{s^*_\text{max}}{2} + (L - 3) \frac{s^*_\text{max}}{2} + (L - 2)s^*_\text{max} \right) = (2 - 4/L) s^*_\text{max} < 2s^*_\text{max},
\]

and the maximum value for \( F_{D^2} \) is

\[
\max F_{D^2} = \frac{1}{L} \left( (L - 1) \frac{s^*_\text{max}}{2} \right)^2 + \left( (L - 3) \frac{s^*_\text{max}}{2} \right)^2 + (L - 2)(s^*_\text{max})^2
\]

\[
= \left( \frac{L}{2} - 1 + \frac{1}{2L} \right) (s^*_\text{max})^2 < \frac{L}{2} (s^*_\text{max})^2.
\]

To prove the tightness of the upper bound of \( F_{|D|} \), consider a scenario as follows. All customers in this busy period have a service requirement of one unit of time, i.e.
s_i = s_{max} = 1. The scenario starts with the simultaneous arrival of N customers (say \(C_1, C_2, \ldots, C_N\)) at the empty system. The first customer to be served is \(C_N\). As soon as \(C_N\) finishes service, a new customer (say \(C_{N+1}\)) joins the system and gets served ahead of \(C_1, \ldots, C_{N-1}\). Just prior to the service completion of \(C_{N+1}\), \(C_{N+2}\) arrives and gets served ahead of \(C_1, \ldots, C_{N-1}\), and so on, until \(C_L\) is served. At the service completion of \(C_L\), the first \(N-1\) customers are served together using PS, and all leave the system \(L\) units of time after the beginning of the scenario. Note that \(N_{max} = N, s_{max}^* = 1 - 1/N\). Analyzing the above scenario we find \(L - N + 1\) customers with a positive discrimination of \(1 - 1/N\) and \(N - 1\) customers with negative discrimination of

\[
1 - \frac{L - N + 1}{N} - \frac{N - 1}{N - 1} = - \frac{L - N + 1}{N}.
\]

The total unfairness is therefore

\[
\frac{1}{L} \left( (L - N + 1) \frac{N - 1}{N} + (N - 1) \frac{L - N + 1}{N} \right) = 2 \frac{(L - N + 1)(N - 1)}{NL},
\]

and in the limiting case

\[
F|D| \xrightarrow{L \to \infty} 2 \frac{N - 1}{N} = 2s_{max}^*.
\]

**Remark 5.6 (Lower Bound for the Unfairness Under Φ).** As shown in Theorem 5.3 only PS and PS imitators can achieve zero unfairness. It is easy to see that there are no PS imitators in Φ, and therefore the tightness of the lower bound does not hold for Φ. The issue of bounding the unfairness under Φ is left for future research.
Chapter 6

THE LOCALITY OF MEASUREMENT PROPERTY

The material in this chapter was published in Levy et al. [93] and also appears in Raz et al. [120].

The major motivation for studying the locality of measurement property comes from trying to use the waiting time variance, or the sojourn time variance, as a metric of system unfairness. Recall from Section 2.1.1 that the waiting time variance was implicitly used as a metric of system unfairness as early as Kingman [83], where it is shown that among all non-idling policies where the jobs are indistinguishable, FCFS has the lowest variance and therefore is “in a sense the ‘fairest’ queue discipline”. In Avi-Itzhak and Levy [7] the waiting time variance was shown to be a surrogate metric for evaluating the order-displacement in the queue as well as a metric that measures the deviation of waiting times between jobs. This also supports its use as a metric for queue unfairness. The question of why waiting time variance should not serve as a simple queue unfairness metric was also posed to us in some personal communications and some referee reports. Published work on evaluating the variance is quite abundant, usually when studying second moments. For example, second moment relationships in M/G/1 systems were studied by Fuhrmann [51] and second moments in some specific $M^X/G/1$ systems were studied in Rosenberg and Yechiali [128].

To demonstrate the issue of locality of measurement let us examine the following simple example exposing some weakness in using the sojourn time variance as a queue unfairness metric.
Example 3. Consider a single-server system where service requirements are deterministic of one unit and arrivals occur in bulks consisting of either 2 in a bulk (type A) or 4 in a bulk (type B), with equal probability for either bulk. Assume that the inter-arrival times are larger than 4 units, and thus each busy period consists of exactly one arrival. Now suppose that the server processes the jobs in a PS mode. The sojourn time of all jobs are therefore 2 units and 4 units for type A and type B, respectively. The average sojourn time is $3\frac{1}{3}$ and the sojourn time variance is $\frac{8}{9}$.

This reflects significant variability of sojourn time which is an indication of unfairness. Nonetheless, an examination of the system reveals that all the jobs that are present concurrently in the system receive exactly the same treatment and thus there is no discrimination in the system. That is, the system is fully fair, which contradicts the measure supplied by the sojourn time variance as an unfairness metric.

A careful examination of the system reveals the source for the difficulty in this example: the system variance accounts for inequity between the treatment of type A and type B jobs, namely jobs in different busy periods. Nonetheless, from unfairness point of view, each job cares only about other jobs that can affect its service. Thus, since a type A job cannot be affected by the service given to type B jobs, as they are in different busy periods, it should not be compared to them for the sake of evaluating unfairness. The use of the variance of sojourn time, therefore, yields some inaccuracy, since it involves a comparison of jobs that are not related to each other.

A second weakness in this approach is exposed by the following example.

Example 4. Consider the same system as above with the same arrival and service pattern. Now suppose that the server processes the jobs in a FCFS manner. The average sojourn time is 2.167. Now focus on the job served second in a type A busy period. Its sojourn time is 2 compared to the average sojourn time of 2.167, so this job receives better service than the average, and can be considered “positively discriminated”. However, closer examination reveals that this is not the case - this job is negatively discriminated as it is served behind the only other job in the busy period.
This reveals a second source of difficulty: a performance metric such as the sojourn time does not provide enough information to determine whether a specific job is positively or negatively discriminated.

In this chapter we approach the issues demonstrated in Example 3 and Example 4 in a general way. We first (Section 6.1) introduce the concept of “Locality of Reference”, and show that there is a good reason to compare each job only to other jobs in the same busy period. We then (Section 6.2) define two desired properties, “Locality of Measurement” and “Locality of Variance”, designed for dealing with these two difficulties. Following that (Section 6.3) we investigate a rather large class of performance metrics and show that within this class only a small subclass of performance metrics, all related to RAQFM, have these desired properties; neither the sojourn-time variance, nor the waiting-time variance belong to this subclass, and thus suffer from those difficulties. In Section 6.4 we consider a second approach one can use for devising a fairness metric, namely explicit evaluation of the intra-variance, to be defined later. In Section 6.5 we provide some numerical examples.

For this chapter we assume the service policy can rely on any of the jobs properties, including, but not limited to, the order of arrival, the service requirement, the time already spent in the system, the amount of service already given to the job, etc. We limit our discussion to non-idling service policies, and for this section we define $\Phi$ 1 – The class of work conserving, non-idling service policies and $\Phi$ 2 – The class of work conserving, non-idling, non-preemptive service policies, $\Phi$ 2 $\subset$ $\Phi$ 1 $\subset$ $\Phi$. Recall that: Work conserving requires that jobs receive exactly the amount of service they require, no less, no more. Non-idling requires that while there are jobs in the system, the server will grant service at full service rate. Non-preemptive requires that once the server started serving a job, it will not stop doing so until the job’s service requirement is fulfilled. It also requires that at most one job be served at any epoch.
6.1 Locality of Reference and Comparison Set

As demonstrated above, while it is natural to use the variance of a performance metric (e.g. the sojourn time) as an indication of service inequity, that is of queue unfairness, it is highly critical over which population such variance will be taken. The set of jobs over which the variance is taken is denoted the *comparison set*. The first question we address, therefore, is what comparison set should be used for evaluating unfairness among jobs. Intuitively speaking, Example 3 and Example 4 demonstrate that the performance of a job should be compared only to that of jobs that are time-wise “close to it”, which we call “Locality of Reference”. In other words, a job should be compared to any other job whose service scheduling can improve or worsen its own treatment. Naturally, such an improvement or worsening is done by shifting resources between the two jobs. We now establish when such a resource shift is possible.

For a single-server non-idling work-conserving system we claim that the performance metrics of two jobs should be compared to each other, that is, the jobs should be in the same comparison set, if and only if the two jobs are served in the same busy period. The reason is that the server can shift processing resources from one job to another if and only if the two jobs are processed in the same busy period.

This is established in the following theorem:

**Theorem 6.1** (Locality of Reference). For any service policy in $\Phi_1$, consider two arbitrary jobs $C_a$ and $C_b$. The server can shift resources between $C_a$ and $C_b$ if and only if they reside in the same busy period.

This theorem may seem intuitive, but requires detailed treatment. We thus provide the exact definition of resource shifting and the proof of this theorem in Appendix B (the theorem is given in Theorem B.1). In broad terms, we define resource shifting as the existence of an alternative assignment of service processing times in which a period exists for which service previously given to $C_a$ is now given to $C_b$ and vice versa, and the rest of the jobs are not affected.
This theorem suggests that the proper comparison set consists of the jobs of a busy period, which is quite intuitive, rather than the whole population.

6.2 Locality of Measurement, Locality of Variance and the Relation Between Them

Having proposed to use the variance of a performance metric as an indication of unfairness, and having realized that for the sake of unfairness evaluation it is desired to have the comparison set of the variance consisting of the jobs of a busy period, we now define two desired performance metric properties, designed to address the two difficulties raised in Example 3 and Example 4.

Let $C$ be a job. Let $X$ be a random variable denoting a performance metric of $C$ (e.g., waiting time or sojourn time) when the system is in steady state. Let $Y$ be a random variable denoting the value of the performance metric $X$ averaged over the jobs participating in the same busy period as $C$, when the system is in steady state.

Property 6.1 (Locality of Measurement). A performance metric is said to be Locally Measured if for every service policy in $\Phi_1$, and for every sample path, $Y$ is constant (not necessarily the same constant for all service policies).

To understand this property, note that for many performance metrics $X$ the corresponding $Y$ random variable is not constant. For example, if $X$ is the waiting time of a job then the average waiting time of the busy period is not constant and will usually vary from one busy period to another.

Intuitively speaking, if a performance metric is locally measured it provides a simple manner of determining whether a specific job is positively or negatively discriminated, by comparing its performance to the above mentioned constant (specific to the service policy). This provides a remedy to the difficulty demonstrated in the Example 4.

To address Example 3 we first define the concepts of inter-variance and intra-variance. The intra-variance of $X$, $\hat{X}_{\text{intra}}$, is the second moment of $X$ around the average of $X$ at the same busy period, namely around $Y$. This is in contrast to the regular variance $\hat{X}$,
which we denote global variance, in which the second moment is computed around the expected value of \( X \) over the whole job population, i.e. \( \mathbb{E}\{(X - \bar{X})^2\} \). In formal terms, let \( b \) be an instance of a busy period determined by the number of jobs, \( n_b \), their relative arrival times, \( a_1, ..., a_{n_b} \), and their service requirements, \( s_1, ..., s_{n_b} \). Note that by using the indices \( 1, ..., n_b \) to denote the indices of the jobs in the busy period we do not lose generality. Let \( f_B(b) \) be the density distribution function of the busy period. Let \( C \in b \) denote the event that \( C \) belongs to a busy period of instance \( b \). The density of this event is \( n_b f_B(b) / (\int n_b f_B(b) db) \).

For the specific busy period instance \( b \), the performance metric \( X \) over its jobs is given by specific values \( x_1, ..., x_{n_b} \). Then the intra-variance is computed by first computing

\[
\mathbb{E}\{(X - Y)^2 | C \in b\} = \frac{1}{n_b} \sum_{i=1}^{n_b} (x_i - \frac{1}{n_b} \sum_{j=1}^{n_b} x_j)^2,
\]

denoted \( \hat{X}_{\text{intra}}^b \), and then un-conditioning on \( C \in b \), i.e. unconditioning the fact that the measure was taken only on jobs of the specific instance \( b \). This is done via

\[
\hat{X}_{\text{intra}} = \frac{\int \hat{X}_{\text{intra}}^b n_b f_B(b) db}{\int n_b f_B(b) db}.
\]

The inter-variance \( \hat{X}_{\text{inter}} \) is defined as the second moment of \( Y \) around the expected value of \( X \). In formal terms, let \( Y_b \) denote the value of \( Y \) conditioned on the event \( C \in b \), that is \( Y_b = \mathbb{E}\{X | C \in b\} = \frac{1}{n_b} \sum_{j=1}^{n_b} x_j \). The inter-variance is computed by taking

\[
\mathbb{E}\{(Y - \bar{X})^2 | C \in b\} = (Y_b - \bar{X})^2,
\]

which we denote \( \hat{X}_{\text{inter}}^b \), and then un-conditioning on \( C \in b \), in the same manner as above.

Intuitively speaking, the intra-variance measures the dispersion of a measure from its average values in each busy period, while the inter-variance measures the dispersion of the busy period average values.

The connection between the global variance, the inter-variance and the intra-variance is established in the following theorem:
**Theorem 6.2.** $\hat{X} = \hat{X}_{\text{intra}} + \hat{X}_{\text{inter}}$, or, in words, the global variance equals the sum of intra-variance and inter-variance.

**Proof.** We start with conditioning $\hat{X}$ on $C \in b$, which we denote $\hat{X}^b$:

$$
\hat{X}^b = \mathbb{E}\{(X - \bar{X})^2 \mid C \in b\} = \mathbb{E}\{(X - Y) + (Y - \bar{X})\}^2 \mid C \in b\)
$$

$$
= \mathbb{E}\{(X - Y)^2 + (Y - \bar{X})^2 + 2(X - Y)(Y - \bar{X}) \mid C \in b\}
$$

$$
= \mathbb{E}\{(X - Y)^2 \mid C \in b\} + \mathbb{E}\{(Y - \bar{X})^2 \mid C \in b\} + \mathbb{E}\{2(X - Y)(Y - \bar{X}) \mid C \in b\}.
$$

We now focus on the third term in the last line. Note that conditioned on $C \in b$, the second multiplicand is constant and therefore

$$
\mathbb{E}\{2(X - Y)(Y - \bar{X}) \mid C \in b\} = 2(Y_{|b} - \bar{X})\mathbb{E}\{X - Y \mid C \in b\}
$$

$$
= 2(Y_{|b} - \bar{X}) (\mathbb{E}\{X \mid C \in b\} - \mathbb{E}\{Y_{|b} \mid C \in b\}) = 2(Y_{|b} - \bar{X})(Y_{|b} - Y_{|b}) = 0,
$$

and thus

$$
\hat{X}^b = \mathbb{E}\{(X - Y)^2 \mid C \in b\} + \mathbb{E}\{(Y - \bar{X})^2 \mid C \in b\} = \hat{X}_{\text{intra}}^b + \hat{X}_{\text{inter}}^b.
$$

Un-conditioning on $C \in b$ we get $\hat{X} = \hat{X}_{\text{intra}} + \hat{X}_{\text{inter}}$. 

We can now define the second property:

**Property 6.2 (Locality of Variance).** A performance metric is said to have its variance local if its global variance equals its intra-variance for every service policy in $\Phi_1$, and for every sample path, i.e. $\hat{X} = \hat{X}_{\text{intra}}$.

Such a performance metric does not suffer from the difficulty presented in Example 3, as there are no differences between busy periods, and thus the only differences measured by the variance are those within busy periods.

Properties 6.1 and 6.2 are strongly related:
Theorem 6.3. A performance metric is locally measured if and only if its variance is local.

Proof. If a performance metric is locally measured then $Y$ is constant, i.e. $X$ averaged over the jobs participating in a busy period is the same for all busy periods. Therefore $Y$ is constant, and clearly, $Y_b = \bar{X}$, so the inter-variance is zero. From Theorem 6.2 it follows that the variance of the performance metric is local.

For the other direction, if a performance metric has its variance local then from the definition of Property 6.2 and from Theorem 6.2 it follows that it has zero inter-variance. From the definition of inter-variance, this means that $(Y_b - \bar{X})^2 = 0 \Rightarrow Y_b = \bar{X}$, and thus $Y_b$ is the same constant for all busy periods, and clearly $Y$ is constant, satisfying the requirement for being locally measured.

6.3 Locally Measured Metrics

We now investigate for which performance metrics the properties defined above hold.

We consider a class of performance metrics $\xi$ which is relatively large. Let $N(t)$ be the number of jobs in the system at time $t$. Each performance metric $X \in \xi$ is identified by two functions, the warranted service function $f(N)$ and the utility function $g(S)$. The performance metric is

$$X_i = g(s_i) - \int_{a_i}^{d_i} f(N(t)) dt.$$  

Intuitively, $X_i$ is the net amount of utility job $C_i$ receives, which is the total utility $g(s_i)$ minus the utility the job is warranted due to its stay in the system, which is momentarily determined by $f(N(t))$.

Note that class $\xi$ is quite wide and includes a variety of performance metrics. For example, the waiting time and the sojourn time metrics both belong to $\xi$: for the waiting time $g(S) = 1, f(N) = 1$ while for the sojourn time $g(S) = 0, f(N) = 1$. For the discrimination defined in (4.1) $g(S) = 1, f(N) = 1/N$.  

We further define a class of performance metrics \( D \subset \xi \) where \( f(N) = \alpha/N \) and \( g(S) = \alpha S + \beta \), where \( \alpha \) and \( \beta \) are constants and \( \alpha, \beta \geq 0 \). The performance metric is

\[
X_i = \beta + \alpha s_i - \int_{a_i}^{d_i} \frac{\alpha dt}{N(t)}.
\]

Intuitively speaking, this generalization serves for a case where the job gets a certain fixed benefit \( \beta \) from being served, and an additional constant benefit of \( \alpha \) per unit of service requirement. Choosing \( \alpha = 1 \) and \( \beta = 0 \) yields the discrimination function.

Note that since we are dealing with work conserving systems \( s_i = \int_{a_i}^{d_i} s_i(t) dt \) where \( s_i(t) \) is the rate of service given to job \( i \) at epoch \( t \). We can therefore write the performance metric in the following way

\[
X_i = \beta + \int_{a_i}^{d_i} \alpha \left( s_i(t) - \frac{1}{N(t)} \right) dt. \tag{6.1}
\]

We now move on to show the importance of the class \( D \)

**Theorem 6.4.** For every performance metric \( X \in D \) both Property 6.1 and Property 6.2 hold.

**Proof.** Let \( X \) be a performance metric such that \( X \in D \). Let \( Y \) be a random variable denoting the value of the performance metric \( X \) averaged over the jobs participating in its busy period, when the system is under steady state.

Consider an arbitrary busy period \( b \) with \( n_b \) jobs. Assume, without loss of generality that the job indices are \( 1, 2, ..., n_b \) and let \( d_{last} \) denote the last departure epoch of the busy period. The expected value of \( X \) in this busy period, \( \bar{X}^b \), is

\[
\bar{X}^b = \frac{1}{n_b} \sum_{i=1}^{n_b} \left( \beta + \int_{a_i}^{d_i} \alpha \left( s_i(t) - \frac{1}{N(t)} \right) dt \right) = \beta + \frac{\alpha}{n_b} \int_{a_1}^{d_{last}} \sum_{i \in (a_i, d_i)} \left( s_i(t) - \frac{1}{N(t)} \right) dt
\]

\[
= \beta + \frac{\alpha}{n_b} \int_{a_1}^{d_{last}} \left( \sum_{i \in (a_i, d_i)} s_i(t) - \sum_{j \in (a_j, d_j)} \frac{1}{N(t)} \right) dt, \tag{6.2}
\]
where we evaluate the performance metric using (6.1).

Consider the first sum in the integral. Note that the system is non-idling, and thus the total amount of service given in any epoch is unity. Considering the second sum, we note that the condition \( i \mid t \in (a_i, d_i) \) holds for exactly \( N(t) \) jobs and thus

\[
\sum_{i \mid t \in (a_i, d_i)} \frac{1}{N(t)} = N(t) \frac{1}{N(t)} = 1.
\]

Substituting the above into (6.2) we get

\[
\beta + \frac{\alpha}{n_0} \int_{a_1}^{d_{\text{last}}} (1 - 1) dt = \beta.
\]

Since \( \beta \) is constant, \( Y \) is a constant and Property 6.1 holds, and from Theorem 6.3 so does Property 6.2.

We now move on to show the uniqueness of \( D \), that is:

**Theorem 6.5.** Let \( X \in \xi \). If either Property 6.1 or Property 6.2 holds for \( X \), then \( X \in D \).

**Proof.** We show that if Property 6.1 holds for \( X \in \xi \), then \( X \in D \). The parallel claim for Property 6.2 will follow from Theorem 6.3.

Note that for Property 6.1 to hold for \( X \), then for every service policy \( \phi \in \Phi_1 \) the measure \( Y \) of an arbitrary job must be constant, though, of course, this constant can be different for different service policies. For a performance metric \( X \in \xi \) we will assume that \( Y \) is a constant for \( \text{PS} \) and show that \( X \in D \).

Recall that \( Y \) is a measure attributed to an arbitrary job \( C \) and is equal to the average of \( X \) over the jobs participating with \( C \) in the same busy period. Let \( Z \) be the average of \( X \) taken over the jobs of an arbitrary busy period. It follows that if \( Y \) is a constant, then \( Z \) is a constant as well. We will now construct a busy period, and as \( X \) satisfies Property 6.1 the average of \( X \) taken over the jobs in this busy period must be a constant.

Consider a busy period, say \( j \), that starts with the simultaneous arrival of \( N^j \) identical jobs, each with the same service requirement \( s^j \). According to \( \text{PS} \) all the jobs are served concurrently for duration \( N^j s^j \). As all the jobs have identical service requirements and all
reside at the system exactly at the same epochs, the performance measures for all these jobs are equal, that is $X_i = X_k$ for every $i, k$ and are exactly

$$g(s^j) - \int_{a_i}^{d_i} f(N^j(t))dt = g(s^j) - N^j s^j f(N^j),$$

since $N^j(t) = N^j$ is constant, and $d_i - a_i = N^j s^j$. As all individual measures are equal this also equals $\bar{X}^j$.

As $X$ satisfies Property 6.1, this value is constant for every busy period, say $c$, and thus

$$g(s^j) - N^j s^j f(N^j) = c \Rightarrow N^j f(N^j) = \frac{g(s^j) - c}{s^j}.$$  (6.3)

Note that no restriction was set on $N^j$ and $s^j$, and therefore this must be true for every value of $N^j$ and $s^j$, and with the same constant $c$. Therefore, the left hand side of the equation cannot depend on $N^j$ and the right hand side cannot depend on $s^j$. This can only be satisfied if $g(s^j) = c + ds^j$ and $f(N^j) = d/N^j$, where $d$ is a constant. We can now replace $d$ by $\alpha$, $c$ by $\beta$, $s^j$ by $S$ and $N^j$ by $N$ to arrive at the exact definition of $D$, i.e. $f(N) = \alpha/N$ and $g(S) = \alpha S + \beta$.

Note that our proof is correct because of the definition of Property 6.1, namely that a constant should be provided for every service policy in $\Phi_1$. Consider now a weaker definition of Property 6.1 in which it is required that a constant be provided for some service policy in $\Phi_2$ (recall that $\Phi_2 \subset \Phi_1$) instead of every service policy in $\Phi_1$:

**Property 6.3** (Weak Locality of Measurement). A performance metric is said to be Weakly Locally Measured if there exists a service policy $\phi \in \Phi_2$ such that for every sample path, $Y$ is constant.

Similarly

**Property 6.4** (Weak Locality of Variance). A performance metric is said to have its variance weakly local if its global variance equals its intra-variance for some policy in $\Phi_2$, and for every sample path.
And of course

**Theorem 6.6.** A performance metric is weakly locally measured if and only if its variance is weakly local.

The proof is similar to the proof of Theorem 6.5.

For these definitions, a stronger claim can be proved:

**Theorem 6.7.** Let $X \in \xi$. If either Property 6.3 or Property 6.4 hold for $X$, then $X \in \mathcal{D}$.

Intuitively speaking, a performance metric satisfying Property 6.1 can be used to measure and compare every service policy in $\Phi_1$. According to Theorem 6.5 within $\xi$, only performance metrics belonging to $\mathcal{D}$ satisfy this requirement. However, some performance metric may exist that can be used to measure and compare only a subset of $\Phi_1$, and not the entire class. Theorem 6.7 shows that if this subset includes even a single policy in $\Phi_2$ (and recall that the most common policy FCFS, belongs to $\Phi_2$), then again, this performance metric must be in $\mathcal{D}$.

**Proof.** Consider an arbitrary service policy $\phi \in \Phi_2$. Assume that Property 6.3 holds for a performance metric $X \in \xi$ for this arbitrary service policy, and for every sample path. We show that $X \in \mathcal{D}$.

Consider a specific type of busy periods, which we will name the *Heavy First Job* scenario. Consider a busy period, say $j$, consisting of $N^j$ jobs. The first job to arrive at the system has a “heavy” service requirement of $N^j s^j$ units of time. Following this job are $N^j - 2$ jobs, each with a service requirement of $s^j$ units of time. These jobs arrive every $s^j$ units of time, starting $s^j$ units of time after the beginning of the busy period. In addition, one job arrives at the system $\epsilon$ units of time before the last job completes service with a service requirement of $\epsilon$, where $\epsilon \to 0$. As this is a non-preemptive service policy, the first job will be served until finished. Following this, the other jobs will be served in some arbitrary order defined by the service policy. However, as those jobs are identical in their service requests, the order of service does not influence $N(t)$, which is depicted in Figure 6.1(a).
We now evaluate $X_j$. It is useful at this point to introduce an alternative way of calculating $X_j$. Let $d_{last}$ denotes the last departure epoch of the busy period, then

$$X_j = \frac{1}{N_j} \sum_{i=1}^{N_j} \left( g(S_i) - \int_{a_i}^{d_i} f(N(t)) dt \right)$$

$$= \frac{1}{N_j} \left( \sum_{i=1}^{N_j} g(S_i) - \int_{a_1}^{d_{last}} \left( \sum_{i|a_i \leq t \leq d_i} f(N(t)) \right) dt \right). \quad (6.4)$$

Note that the warranted service is derived by integrating $N(t)f(N(t))$ over the entire busy period.

The value of (6.4) for the specific arrival and service pattern we consider is:

$$X_j = \frac{1}{N_j} \left( (N_j^2 - 2)g(s^j) + g(N_j s^j) + g(\epsilon) - 2s^j \sum_{i=1}^{N_j-1} i f(i) - \epsilon f(2) \right),$$
which after substituting \( \epsilon \to 0 \) yields

\[
X^j = \frac{1}{N_j} \left( (N_j^j - 2)g(s^j) + g(N_j^j s^j) + g^* - 2s^j \sum_{i=1}^{N_j^j-1} i f(i) \right),
\]

where \( g^* = \lim_{S \to 0} g(S) \), which is constant for every function \( g(S) \).

Now consider a similar busy period, say \( k \), except that on this busy period instead of the last job (the one with the service requirement of \( \epsilon \) units of time) a regular job with a service requirement of \( s^j \) units of time arrives \( (N_j^j - 1)s^j \) units of time after the beginning of the busy period. Figure 6.1(b) depicts \( N(t) \), where the gray squares form the difference between the busy periods.

For this busy period

\[
\overline{X}^k = \frac{1}{N_j} \left( (N_j^j - 1)g(s^j) + g(N_j^j s^j) - 2s^j \sum_{i=1}^{N_j^j-1} i f(i) - s^j N_j^j f(N_j^j) \right).
\]

As Property 6.3 holds for \( X \), \( \overline{X}^j = \overline{X}^k \) leading to

\[
\overline{X}^k - \overline{X}^j = \frac{1}{N_j} \left( g(s^j) - g^* - s^j N_j^j f(N_j^j) \right) = 0 \Rightarrow N_j f(N_j^j) = \frac{g(s^j) - g^*}{s^j}.
\]

This bears a striking resemblance to (6.3) as \( g^* \) is constant for \( g(S) \). We follow the same reasoning used in the proof of Theorem 6.5 to complete the proof.

### 6.4 Explicit Evaluation of the Intra-Variance

This chapter started with explaining why using the waiting time variance, or the sojourn time variance, as a measure of unfairness, can lead to difficulties, and that only the intra-variance matters for the purpose of fairness. Our discussion led to the uniqueness of \( D \) within \( \xi \), in having a variance equal to the intra-variance, thus avoiding this problem.

However, one could attempt to devise a fairness metric by choosing a performance metric, say the sojourn time, and explicitly evaluating its intra-variance as an unfairness
metric. To demonstrate how this solves the difficulty let us readdress Example 3. Recall that arrivals occur in bulks consisting of either 2 jobs in a bulk (type A) or 4 jobs in a bulk (type B), with equal probability for either bulk, and the server processes the jobs in a PS mode. The sojourn time of all jobs are therefore 2 units and 4 units for type A and type B, respectively. To calculate the intra-variance, we evaluate $\mathbb{E}\{(X-Y)^2 \mid C \in b\}$ for every $C \in b$. In all cases this equals zero as $X = Y_{|b}$. Unconditioning on $C \in b$ yields an intra-variance of zero, or no unfairness, which is proper for this system.

While this method seems viable it has two major drawbacks. First, for the customer, it does not address the issue of locality of measurement at all, i.e., it does not provide a customer with a scale of reference to determine the level, or even the sign, of its discrimination. To address this issue one needs to know the average measure in each busy period in an on-line manner, while the information is only available at the end of busy period.

A second drawback is to the analyst. While the intra-variance can be obtained through simulation, it makes analytical results hard to obtain. To compute the intra-variance through simulation, one needs to compute for every job its performance metric, say waiting time, and for every busy period its average performance metric, say average waiting time, derive the difference squared, and average over all jobs. This might require a memory large enough to include the entire busy period. Deriving the intra-variance analytically is even harder. One needs to compute $\mathbb{E}\{(X-Y)^2 \mid C \in b\}$ and then uncondition on $C \in b$. Note that

$$\mathbb{E}\{(X-Y)^2 \mid C \in b\} = \mathbb{E}\{X^2 + Y^2 - 2XY \mid C \in b\} = \mathbb{E}\{X^2 \mid C \in b\} + \mathbb{E}\{Y^2 \mid C \in b\} - 2\mathbb{E}\{XY \mid C \in b\} = \mathbb{E}\{X^2 \mid C \in b\} + (\mathbb{E}\{Y \mid C \in b\})^2 - 2\mathbb{E}\{Y \mid C \in b\}\mathbb{E}\{X \mid C \in b\} = \mathbb{E}\{X^2 \mid C \in b\} - (\mathbb{E}\{Y \mid C \in b\})^2$$

where the third line results from the fact that conditioned on $C \in b$, $Y$ is a constant. While the unconditioning of the first term simply yields the second moment of $X$, the
second term does not yield to analysis in most practical cases of randomly sampled arrival times and service requirements. We leave this as a challenging open problem.

6.5 Numerical Results

In this section we present numerical results from a simple scenario and from simulating some simple stochastic systems.

6.5.1 Numerical Example

In this section we provide a simple numerical example that shows that using different metrics can lead to contradicting results. We compare the (global) variance of the sojourn time to the (global) variance of the discrimination and show that the results are contradicting. We then compare the (global) variance of the sojourn time to its intra-variance and show again that the results contradict.

Consider a system where there are two types of customers $A$ and $B$, with service requirements $s_A = a$ and $s_B = b$, respectively. The arrival rate is very low ($\rightarrow 0$) and each arrival consists of two customers of the same type, with probability of 0.5 to any of the two types. We examine two service policies, $PS$ and $FCFS$. Note that in this specific case all non-preemptive policies behave the same, so $FCFS$ can be treated as a representative of $\Phi_2$.

Table 6.1 provides a comparison of the variance of the discrimination to that of the sojourn time. Note that while there appear to be two “classes” of customers, the measures are taken over the entire population, without taking this into account. Using the variance of discrimination as a fairness measure, clearly, $\frac{1}{5}(a^2 + b^2) > 0$, thus $PS$ is more fair than $FCFS$. Using the variance of the sojourn time we see that $\frac{1}{10}(11a^2 - 18ab - 11b^2)$ is not always larger than $(a - b)^2$. In fact, for every choice of $a$, if we choose $b > \frac{1}{5}(7 + 2\sqrt{6})a \approx 2.38a$ it yields as a result that $FCFS$ is more fair than $PS$. Specifically, if we choose $a = 1, b = 3$, the variance of the sojourn time is 3.5 for $FCFS$ and 4 for $PS$. The variance of discrimination, in contrast, is 1.25 for $FCFS$ and 0 for $PS$ and 1.25 > 0. As for the intra-variance of the
sojourn time we see that $\frac{1}{8}(a^2 + b^2) > 0$, meaning that according to the intra-variance of the sojourn time, PS is more fair than FCFS, in agreement with the variance of discrimination.

<table>
<thead>
<tr>
<th></th>
<th>Discrimination</th>
<th>Sojourn Time</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Global Variance</strong></td>
<td>FCFS: $\frac{1}{8}(a^2 + b^2)$</td>
<td>$\frac{1}{16}(11a^2 - 18ab + 11b^2)$</td>
</tr>
<tr>
<td></td>
<td>PS: 0</td>
<td>$(a - b)^2$</td>
</tr>
<tr>
<td><strong>Intra-Variance</strong></td>
<td>FCFS: $\frac{1}{8}(a^2 + b^2)$</td>
<td>$\frac{1}{8}(a^2 + b^2)$</td>
</tr>
<tr>
<td></td>
<td>PS: 0</td>
<td>0</td>
</tr>
<tr>
<td><strong>Inter-Variance</strong></td>
<td>FCFS: 0</td>
<td>$\frac{9}{16}(a - b)^2$</td>
</tr>
<tr>
<td></td>
<td>PS: 0</td>
<td>$(a - b)^2$</td>
</tr>
</tbody>
</table>

*Table 6.1: Variances of Discrimination Vs. Variance of Sojourn Time*

As a side note, notice that as expected from Theorem 6.2, the (global) variance equals the sum of the intra-variance and the inter-variance, $\frac{1}{8}(a^2 + b^2) + \frac{9}{16}(a - b)^2 = \frac{1}{16}(11a^2 - 18ab - 11b^2)$.

In conclusion, we see that using the variance of the sojourn time as a fairness measure can lead to FCFS being more fair than PS. This is caused by the sojourn time not having a local variance. Using either the variance of discrimination, or explicitly using the intra-variance of the sojourn time leads to the opposite, and perhaps more appropriate, result.

6.5.2 Simulation Results

Figure 6.2 depicts simulation results comparing the (global) variance of the sojourn time to its intra-variance. Figure 6.2(a) and Figure 6.2(b) present results for exponential service requirement distribution and constant (unit) service requirements, respectively. In each case we compare the FCFS policy to the LCFS one. Each of the plotted points is the result of simulation of at least $10^6$ jobs. The results demonstrate that indeed, the variance and the intra-variance differ significantly. This is more prominent for FCFS, and can reach relative differences of 100 percents or more, but is evident in both service policies and for both service requirement distributions.
Chapter 6. The Locality of Measurement Property

Figure 6.2: Variance Vs. Intra-Variance of Sojourn Time

(a) Exponential Service Requirement Distribution  (b) Constant Service Requirements
Chapter 7

COMPUTING RAQFM UNDER THE MARKOVIAN MODEL

In this chapter we provide a methodology for computing RAQFM for Markovian type systems in steady state. We then demonstrate this method in detail by carrying out the computation for the single-server FCFS system under M/M/1.

The material in this section appears in Raz et al. [118]

7.1 The Analysis Methodology

To facilitate the mathematical analysis of Markovian type systems, arrival and departure epochs are labeled event epochs, and time is viewed as being slotted by these event epochs. The \( l \)-th time slot, \( l = 1, 2, \ldots \), spans between the \( l \)-th and the \((l + 1)\)-th event epochs. Clearly, the number of customers in the system is constant during each slot \( l \), and is denoted by \( N_l \).

We limit the analysis to systems where a service decision is made only at arrival and departure epochs. Thus, the rate of service given to each customer in the system is constant during each slot. We denote the rate at which service is given to \( C_i \) at the \( l \)-th slot by \( \sigma_{l,i} \). When a tagged customer is observed, the rate of service is denoted \( \sigma_l \). Note that the state of the system does not change during a slot, as we assume that system state can only change because of either an arrival, a departure, or combined with a service decision made at such epochs.

Let the arrival and departure rates during the \( l \)-th slot be \( \lambda_l \) and \( \mu_l \) respectively. Let \( T_l, l = 1, 2, \ldots \) be the duration of the \( l \)-th slot. Then \( T_l, l = 1, 2, \ldots \) are independent
random variables exponentially distributed with parameters $\lambda_l + \mu_l$, and first two moments $t^{(1)}_l = 1/(\lambda_l + \mu_l), t^{(2)}_l = 2/(\lambda_l + \mu_l)^2 = 2(t^{(1)}_l)^2$.

We now move on to the methodology itself. We enumerate the steps in the analysis for reference in the rest of this work. Consider a tagged customer $C$. We first (step 1) define the state $S \in \mathbb{S}$ that $C$ observes in the system at any given slot, where $\mathbb{S}$ is the field of all possible states, i.e. the state space. We will alternatively say that ‘$C$ is at state $S$’. The state must include enough information for determining the number of customers in the system, the rate of service given to the tagged customer and the total service rate, the departure rate, the arrival rate, and for statistically predicting the following state.

Next (step 2), we represent the momentary discrimination in a slot in terms of the state $S$ seen at that slot, and denote it $\delta(S)$. Using (4.4) we have $\delta(S) = s(S) - \omega(S)/N(S)$, where $s(S), \omega(S), \text{and} N(S)$ are the service rate $C$ receives at that slot, the total service rate at that slot, and the number of customers in the system at that slot, respectively. We also express the moments of the slot length $t^{(1)}(S)$ and $t^{(2)}(S)$ using $\mu(S)$ and $\lambda(S)$, the departure rate and the arrival rate at that slot.

Let $D(S)$ be a random variable denoting the discrimination experienced by the tagged customer, during a walk starting at state $S$, i.e., its future discrimination starting when it is at state $S$ and ending when it exits the system.

Next (step 3), assume that $C$ is at state $S_k \in \mathbb{S}$ at some slot. $C$’s state can change only at the end of the slot, as a result of the event ending that slot. For each $S_k \in \mathbb{S}$ we enumerate the events possible at the end of the slot, their probabilities and the state observed at the next slot. We denote by $p_{k,j}$ the transition probability from $S_k$ to $S_j$ for all $S_k, S_j \in \mathbb{S}$. Of course $\forall k \mid S_k \in \mathbb{S} : \sum_{\{j \mid S_j \in \mathbb{S}\}} p_{k,j} = 1$.

Note that a departure of $C$, which depending on $S_k$ might be a possible even at the end of the slot, results in an absorbing state denoted $S_\infty$, and conveniently $D(S_\infty) \overset{\text{def}}{=} 0$. 

\[ \forall k \mid S_k \in \mathbb{S} : \sum_{\{j \mid S_j \in \mathbb{S}\}} p_{k,j} = 1. \]
Chapter 7. Computing RAQFM Under The Markovian Model

Assuming $C$ is in state $S_k$ in the $l$-th slot, then

$$D(S_k) = T_l \delta(S_k) + D(S_j), \text{ with probability } p_{k,j}. \quad (7.1)$$

Remark 7.1. In the equation above the sum is a sum of two independent random variables. This is so since the future discrimination, $D(S_j)$, is independent of the length of the current slot, $T_l$, and $\delta(S_k)$ is a constant.

We now let $d(S)$ and $d^{(2)}(S)$ be the first and second moments of $D(S)$. Taking expectations over $(7.1)$ yields the following set of linear equations:

$$d(S_k) = t^{(1)}(S_k) \delta(S_k) + \sum_{\{j: S_j \in S\}} p_{k,j} d(S_j), \quad \forall S_k \in S. \quad (7.1)$$

Similarly (step 4), we can take the second moments in $(7.1)$. In doing so we take advantage of the fact that $\delta(S_k)$ is a constant, and that the sum represents a sum of independent random variables due to Remark 7.1, and thus the expected value of the product equals to the product of the expected values. This leads to the following set of linear equations:

$$d^{(2)}(S_k) = t^{(2)}(S_k)(\delta(S_k))^2 + \sum_{\{j: S_j \in S\}} p_{k,j} d^{(2)}(S_j) +

2t^{(1)}(S_k) \delta(S_k) \sum_{\{j: S_j \in S\}} p_{k,j} d(S_j), \quad \forall S_k \in S. \quad (7.1)$$

These expressions can be used, via numerical recursion, to compute the values of $d^{(2)}(S_k)$ to any desired accuracy.

Finally (step 5), let $P_S$ be the steady state probability that an arriving customer finds itself in state $S$. Then

$$F_{D^2} = \sum_{S \in S} P_S d^{(2)}(S).$$

The probabilities $P_S$ can usually be computed. In particular, when the arrival process is Poisson, the PASTA property (see Wolff [157]) simplifies the computation.
Chapter 7. Computing RAQFM Under The Markovian Model 68

Note that if state $S$ is not possible upon arrival, then $P_S = 0$, and for convenience $d(S) \overset{def}{=} d^{(2)}(S) = 0$.

Remark 7.2 (Conditioned Expected Discrimination and Second Moment). For a Poisson arrival process, for every $S \in \mathcal{S}$ that is possible upon arrival, $d(S)$ is the expected discrimination conditioned on being in state $S$ upon arrival. The same is true for $d^{(2)}(S)$, the conditioned second moment. These values can be used to gain insight into the behavior of discrimination and fairness for individual customers, conditioned on their arrival state. For example, we evaluate $d(S)$ in Section 8.1 to gain insight into the discrimination observed under different service policies.

Remark 7.3 (Going Beyond the Exponential Case). The approach used above can be easily generalized for any phase-type service requirement distribution, i.e. the M/PH/$\cdot$ model. This was done by Brosh et al. [27] in the context of single server, see Section 8.4. There is a large body of literature on the topic of approximating general distributions by phase-type ones, for example Johnson and Taaffe [76, 77], Bobbio and Telek [19], Horváth and Telek [66], and many more. Lately Osogami and Harchol-Balter [105, 106, 107], Osogami [104] have addressed the issue again, see for review of many methods and approaches. Since any M/GI/$\cdot$ system can thus be approximated by a M/PH/$\cdot$ system this provides a way to approximate M/GI/$\cdot$ type systems.

7.2 Example: Analysis of Unfairness in a FCFS M/M/1 Queue

We now carry out the analysis needed for computing unfairness in a FCFS M/M/1 queue, with arrival rate $\lambda$ and mean service length $1/\mu$.

(1) Let us consider a tagged customer $C$. At every slot let $a \in \mathbb{N}^0$ denote the number of customers ahead of $C$ including the served customer and $b \in \mathbb{N}^0$ the number of customers behind $C$. Due to the memoryless properties of the system, the state $\langle a, b \rangle$ captures all that is needed for predicting the future discrimination of $C$.

(2) The number of customers in the system at a slot where $C$ observes the state $\langle a, b \rangle$ is $N(a, b) = a + b + 1$. The rate of service given to $C$ at that slot is $s(a, b) = 1(a = 0)$.
and the total service rate is a constant \( \omega(a, b) = 1 \). The rates of arrival and departure are constant at \( \mu(a, b) = \mu \) and \( \lambda(a, b) = \lambda \).

The momentary discrimination at state \( \langle a, b \rangle \), denoted \( \delta(a, b) \), is therefore given by

\[
\delta(a, b) = 1(a = 0) - \frac{1}{a + b + 1}
\]

and the moments of the slot length are,

\[
t^{(1)}(a, b) = \frac{1}{\lambda + \mu}, \quad t^{(2)}(a, b) = \frac{2}{(\lambda + \mu)^2} = 2(t^{(1)})^2, \quad a, b \geq 0,
\]

as long as the system is non-empty, which is always true for the slots of interest, since \( C \) resides in the system. As the moments are independent of the state we can denote them \( t^{(1)} \) and \( t^{(2)} \).

1. A customer arrives into the system. The probability of this event is \( \tilde{\lambda} \overset{\text{def}}{=} \lambda/(\lambda + \mu) \).

   Afterwards \( C \)'s state will change to \( \langle a, b+1 \rangle \).

2. A customer leaves the system. The probability of this event is \( \tilde{\mu} \overset{\text{def}}{=} \mu/(\lambda + \mu) \). If \( C \) is not being served \( (a > 0) \) \( C \)’s state will change to \( \langle a - 1, b \rangle \); otherwise \( C \) will leave the system.

This leads to the following recursive expression:

\[
d(a, b) = t^{(1)} \delta(a, b) + \tilde{\lambda}d(a, b + 1) + 1(a > 0)\tilde{\mu}d(a - 1, b). \tag{7.2}
\]

(4) Similarly, the equations for \( d^{(2)}(a, b) \) are

\[
d^{(2)}(a, b) = t^{(2)}(\delta(a, b))^2 + \tilde{\lambda}d^{(2)}(a, b + 1) + 2t^{(1)}\delta(a, b)\tilde{\lambda}d(a, b + 1) +
\]

\[
1(a > 0)\left(\tilde{\mu}d^{(2)}(a - 1, b) + 2t^{(1)}\delta(a, b)\tilde{\mu}d(a - 1, b)\right). \tag{7.3}
\]
(5) The states possible upon arrival are \((k, 0), k = 0, 1, \ldots\), where \(k\) is the number of customers seen on arrival. Therefore

\[
F_{D^2} = \sum_{k=0}^{\infty} p_k d^{(2)}(k, 0),
\]

where

\[
p_k = (1 - \rho) \rho^k
\]

is the steady state probability of encountering \(k\) customers in the system.
Chapter 8

FAIRNESS IN SINGLE SERVER SYSTEMS

In this chapter we use RAQFM to analyze the unfairness of several service disciplines in single server systems, namely FCFS, LCFS, P-LCFS and ROS. We start with the M/M/1 paradigm. We derive the expected discrimination experienced by an arriving customer as a function of the number of customers it finds in the system upon arrival. The results shed light on customer discrimination as a function of the queue situation it encounters. We then derive the system unfairness, measured via the variance of discrimination, for these four policies. Most of this material in this section was published in Raz et al. [117].

Following that, we study other service requirement distributions. We use simulation to study a bi-valued distribution with high variability. We then present results from Brosh et al. [27] that studied general distributions, with focus on phase type ones.

8.1 Conditional Discrimination in M/M/1

In this section we consider an M/M/1 system with arrival rate $\lambda$ and mean service length $1/\mu$. We derive $\mathbb{E}\{D \mid k\}$, the expected customer discrimination, conditioned on the number of customers it finds in the system upon arrival. While $\mathbb{E}\{D \mid k\}$ is not used to derive the system unfairness, it is still an interesting measure that will serve to shed light on the situations at which systems are subject to high discrimination. The general method for this evaluation was already detailed in the previous section, see Remark 7.2.
8.1.1 FCFS

The analysis of FCFS was detailed in Section 7.2 and (7.2) provides a set of equations for evaluating \( d(a, b) \). We can use \( \mathbb{E}\{D \mid k\} = d(k, 0) \) to evaluate the conditional distribution.

Remark 8.1. Note that the quantity \( \mathbb{E}\{D \mid k\} \) depends on \( \lambda \) and \( \mu \) only through their ratio \( \rho = \lambda/\mu \), and not through their individual values. This is true for all the policies studied in this chapter.

Alternatively, one can compute \( \mathbb{E}\{D \mid k\} \) using the result of the following theorem.

**Theorem 8.1.** Let \( P(a, b \mid k) \) be the probability that a walk visits \( \langle a, b \rangle \) given that the customer sees \( k \) other customers upon arrival and let \( G(a, b \mid k) \) be the number of times \( \langle a, b \rangle \) is reached in such a walk. Then (a)

\[
P(a, b \mid k) = \mathbb{E}\{G(a, b) \mid k\} = \tilde{\lambda}^b \tilde{\mu}^{k-a} \left( \begin{array}{c} k-a+b \\ b \end{array} \right) = \frac{\rho^b}{(1+\rho)^{k-a}} \left( \begin{array}{c} k-a+b \\ b \end{array} \right),
\]

and therefore (b)

\[
\mathbb{E}\{D \mid k\} = t(1) \sum_{b=0}^{\infty} \sum_{a=0}^{k} P(a, b \mid k) \delta(a, b),
\]

(8.1)

Proof. Moving along a walk \( b \) is monotone non-decreasing and \( a \) is monotone non-increasing. Furthermore, either the value of \( a \) or the value of \( b \) must change by one unit in each step. Therefore, if \( \langle a, b \rangle \) is visited, it cannot be visited again, and thus the range of \( G(a, b \mid k) \) is \( \{0, 1\} \) and

\[
\mathbb{E}\{G(a, b) \mid k\} = P\{G(a, b \mid k) = 1\} = P(a, b \mid k).
\]

This proves the first equality of (a), and (b) follows immediately.

To prove the rest of (a) we note that a walk will do this if and only if the first \( k - a + b \) events in the walk consist of exactly \( b \) arrival events and \( k - a \) departure events. The probability of such a walk is given by the binomial distribution:

\[
\mathbb{E}\{G(a, b) \mid k\} = \tilde{\lambda}^b \tilde{\mu}^{k-a} \left( \begin{array}{c} k-a+b \\ b \end{array} \right) = \frac{\rho^b}{(1+\rho)^{k-a}} \left( \begin{array}{c} k-a+b \\ b \end{array} \right),
\]
where for the second equality recall that \( \tilde{\lambda} \overset{\text{def}}{=} \frac{\lambda}{(\lambda + \mu)} \), \( \tilde{\mu} \overset{\text{def}}{=} \frac{\mu}{(\lambda + \mu)} \).

**Special Cases**

\( \mathbb{E}\{D | 0\} \), the expected discrimination for a customer arriving at an empty system can be easily derived from (8.1):

\[
\mathbb{E}\{D | 0\} = t^{(1)} \sum_{b=0}^{\infty} \tilde{\lambda}^b (1 - \frac{1}{b+1}) = t^{(1)} \left( \frac{1}{1-\tilde{\lambda}} - \sum_{b=0}^{\infty} \frac{\tilde{\lambda}^b}{b+1} \right)
\]

The second part of this sum yields:

\[
\sum_{b=0}^{\infty} \frac{\tilde{\lambda}^b}{b+1} = \frac{1}{\lambda} \sum_{b=0}^{\infty} \frac{\tilde{\lambda}^{b+1}}{b+1} = \frac{1}{\lambda} \sum_{b=0}^{\infty} \frac{\tilde{\lambda}^b}{\lambda} d\tilde{\lambda} = \frac{1}{\lambda} \int_{\tilde{\lambda}}^{\infty} \frac{1}{\lambda - \tilde{\lambda}} d\tilde{\lambda}
\]

and so

\[
\mathbb{E}\{D | 0\} = t^{(1)} \left( \frac{1}{1-\tilde{\lambda}} + \frac{1}{\lambda} \ln(1 - \tilde{\lambda}) \right) = t^{(1)} \left( 1 + \rho + \frac{1 + \rho}{\rho} \ln \frac{1}{1 + \rho} \right) = t^{(1)} \left( 1 + \rho - \frac{1 + \rho}{\rho} \ln(1 + \rho) \right).
\]

Observe that \( \mathbb{E}\{D | 0\} \) is a monotone-increasing function in \( \rho \) and thus its maximum value is reached when \( \rho \to 1 \) and its minimum value is reached when \( \rho \to 0 \), the high and low traffic bounds. For the high traffic bound

\[
\mathbb{E}\{D | 0\} \xrightarrow{\rho \to 1} t^{(1)}(2 - 2 \ln 2) \approx 0.613706 t^{(1)},
\]
which is the maximum expected discrimination for a user in the FCFS service policy. For the low traffic bound

\[
\mathbb{E}\{D \mid 0\} = t^{(1)} \left( 1 + \rho - \frac{1 + \rho}{\rho} \ln(1 + \rho) \right)
\]

This is expected since when \( \rho \to 0 \) there are almost no arriving customers, thus an arriving tagged customer finding the system empty will be served immediately, and leave without experiencing arrivals during its stay. Thus, its discrimination approaches \( D = 0 \).

Another value easily derived is the limit of \( \mathbb{E}\{D \mid k\} \) for \( \rho \to 0 \). If \( \rho \to 0 \) it follows immediately that \( \tilde{\lambda} \to 0 \) and \( \tilde{\mu} \to 1 \) and thus \( \mathbb{E}\{G(a, b) \mid k\} = 0 \) for \( b > 0 \) and so \((8.1)\) becomes

\[
\mathbb{E}\{D \mid k\} \xrightarrow{\rho \to 0} t^{(1)} \sum_{a=0}^{k} \delta(0, a) = 0 + t^{(1)} \sum_{a=1}^{k} \frac{1}{a + 1} = -t^{(1)} H_k,
\]

where \( H_n \) is the \( n \)-th harmonic number.

For the numerical computation of other values using \((8.1)\), the summation must be stopped at some large number, justified by the low probability of having very large values of \( b \), i.e. \( \mathbb{E}\{G(a, b) \mid k\} \xrightarrow{b \to \infty} 0 \). For the computation using \((7.2)\) the same applies as \( d(a, b) \xrightarrow{b \to \infty} 0 \). This can be simply done by setting \( d(a, b) \) to be zero for large values of \( a \) and \( b \). A second way to justify these is to look at a system with a large finite queue.

**Numerical results and properties**

Figure \(8.1\) depicts the value of the conditional discrimination normalized by \( t^{(1)} \), \( \mathbb{E}\{D[k] \}/t^{(1)} \), as a function of \( k \), for some values of \( \rho \). We use the normalized discrimination due to Remark \(8.1\) which enables us to create a single plot that covers every value of both \( \lambda \) and \( \mu \), by creating a plot of \( \rho \). We will follow this convention throughout this section.
All the special cases presented above can be verified in the figure. In addition we note the following properties:

1. The worst (most negative) discrimination a customer may experience is when the load approaches zero and the customer finds a very long queue; in this case the expected discrimination monotonically decreases in the queue length $k$ and is unbounded. This seems to agree with common feelings where perhaps the most disappointing queue state one can encounter is a long queue when the load is very small.

2. The best (most positive) discrimination a customer may experience is to find an empty queue when the load is very high. This, again, seems to agree with common customer feelings.

3. Negative discrimination appears to monotonically increase with the queue size encounters. This seems to fit our intuition as well.

4. Normalized discrimination seems to monotonically decrease with the load $\rho$.

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{expected_fifo2.png}
\caption{Conditional (normalized) Discrimination for M/M/1 under FCFS}
\end{figure}
8.1.2 LCFS

Following the same steps:

1. Let \( a \in \mathbb{N}^0 \) denote the number of customers ahead of \( C \) at the queue (and thus to be served after \( C \)). Let \( b \in \mathbb{N}^0 \) denote the number of customers behind \( C \) at the queue (and thus to be served before \( C \)).

(3) Possible events are:

1. A customer arrives into the system. The probability of this event is \( \tilde{\lambda} \). If \( C \) was in service (\( b = 0 \)) \( C \)'s state will change to \( (a + 1, b) \), otherwise to \( (a, b + 1) \).

2. A customer leaves the system. The probability of this event is \( \tilde{\mu} \). If \( C \) was in service, \( C \) will leave the system, otherwise \( C \)'s state will change to \( (a, b - 1) \).

This leads to the following set of equations:

\[
d(a, b) = t^{(1)} \delta(a, b) + \begin{cases} \tilde{\lambda}d(a, b + 1) + \tilde{\mu}d(a, b - 1) & b > 0 \\ \tilde{\lambda}d(a + 1, b) & b = 0 \end{cases}.
\] (8.2)

Observe that an arriving customer starts either with state \( \langle 0, 0 \rangle \) when the system is empty, or with \( \langle 1, k - 1 \rangle \) when it is not empty, thus

\[
\mathbb{E}\{D \mid k\} = \begin{cases} d(k - 1, 1) & k > 0 \\ d(0, 0) & k = 0 \end{cases}.
\] (8.3)

These equations can be solved numerically to evaluate \( \mathbb{E}\{D \mid k\} \).

The following is an alternative, more efficient, approach. We need Theorem 8.2 and Lemma 8.1 to prove Lemma 8.2 which will provide the actual method of evaluation.

**Theorem 8.2.** Consider the infinite series \( a_n, b_n, n = 1, 2, \ldots \) defined iteratively as follows:

\[
a_1 = 0 \quad b_1 = 1
\]

\[
a_2 = t^{(1)}c_{1,k} + \tilde{\mu}d_{k}^* \quad b_2 = \tilde{\lambda}
\]
\[ a_n = \frac{a_{n-1} + t(1)b_{n-1}c_{n-1,k} - b_{n-1}\tilde{\mu}\frac{a_{n-2}}{b_{n-2}}}{1 - \frac{b_{n-1}\tilde{\mu}}{b_{n-2}}} = \frac{b_{n-2}(a_{n-1} + t(1)b_{n-1}c_{n-1,k}) - \tilde{\mu}b_{n-1}a_{n-2}}{b_{n-2} - b_{n-1}\tilde{\mu}} , n > 2 \]

\[ b_n = \frac{\tilde{\lambda}b_{n-1}}{1 - \frac{b_{n-1}\tilde{\mu}}{b_{n-2}}} = \frac{\tilde{\lambda}b_{n-1}b_{n-2}}{b_{n-2} - b_{n-1}\tilde{\mu}} , n > 2 \]  

(8.4)

where \( c_{n,k} \overset{\text{def}}{=} \delta(k - 1, n) \) and 

\[ d^*_{k} \overset{\text{def}}{=} t(1) \left( 1 + \rho - \sum_{a=0}^{\infty} \frac{\rho}{1+\rho} a^a \frac{1}{a+k} \right) \]

Then (i) \( d(k - 1, 0) = d^*_{k} \) and (ii) for every \( n \geq 1 \) and \( k > 0 \)

\[ \mathbb{E}\{D \mid k\} = a_n + b_n d(k - 1, n) , \]  

(8.5)

Proof. (i) \( d(k - 1, 0) \) can be evaluated according to (8.2) as follows

\[ d(k - 1, 0) = t(1)\delta(k - 1, 0) + \tilde{\lambda}d(k, 0) = t(1)\delta(k - 1, 0) + \tilde{\lambda}(t(1)\delta(k, 0) + \tilde{\lambda}d(k + 1, 0)) \]

\[ = \ldots = t(1)\sum_{a=k}^{\infty} \tilde{\lambda}^{a-k}\delta(a - 1, 0) = t(1)\sum_{a=k}^{\infty} \frac{\rho}{1+\rho} a^{a-k}(1 - \frac{1}{a}) \]

\[ = t(1) \left( 1 + \rho - \sum_{a=0}^{\infty} \frac{\rho}{1+\rho} a^a \frac{1}{a+k} \right) = d^*_{k} . \]

(ii) For \( n = 1 \), follows immediately from (8.3).

For \( n = 2 \), from (8.3) and (8.2)

\[ \mathbb{E}\{D \mid k\} = d(k - 1, 1) = t(1)\delta(k - 1, 1) + \tilde{\lambda}d(k - 1, 2) + \tilde{\mu}d^*_{k} , \]

1 \( d^*_{k} \) can also be written as \( 1 + \rho - \frac{1}{x} F(k, 1; k + 1; \frac{\rho}{1+\rho}) \) where \( F(a, b; c; x) \) is Gauss’s Hypergeometric series, which might yield to even further reduction, see e.g. Jeffrey [75], Sec. 22.17, Abramowitz and Stegun [1], Sec. 15.1.
Next we prove the theorem for $n \geq 3$ by induction.

**Induction base:** correctness for $n = 1, 2$ was proven above.

**Induction assumption:** $\mathbb{E}\{D \mid k\} = a_m + b_m d(k - 1, m)$ for $m = 1, 2, \ldots, n - 2, n - 1$.

For brevity denote $d_n \overset{\text{def}}{=} d(k - 1, n)$ and $c_n \overset{\text{def}}{=} c_{n,k} = \delta(k - 1, n)$. Note that using this notation the assumption can be written as $d_1 = a_m + b_m d_m$.

**Induction step:** we now prove the correctness for $m = n$.

From the assumption for $m = n - 2$

$$d_1 = a_{n-2} + b_{n-2} d_{n-2} \Rightarrow d_{n-2} = \frac{d_1 - a_{n-2}}{b_{n-2}}. \quad (8.6)$$

Note that using the abbreviated notation (8.2) for $a = k - 1$ can be written as follows

$$d_n = t(1)c_n + \lambda d_{n+1} + \mu d_{n-1}.$$

Evaluating this at $n - 1$ yields

$$d_{n-1} = t(1)c_{n-1} + \lambda d_n + \mu d_{n-2}. \quad (8.7)$$

From the assumption for $m = n - 1$

$$d_1 = a_{n-1} + b_{n-1} d_{n-1}.$$ 

Substituting $d_{n-1}$ from (8.7) yields

$$d_1 = a_{n-1} + b_{n-1}(t(1)c_{n-1} + \lambda d_n + \mu d_{n-2}).$$

Substituting $d_{n-2}$ from (8.6) yields

$$d_1 = a_{n-1} + b_{n-1}(t(1)c_{n-1} + \lambda d_n + \mu \frac{d_1 - a_{n-2}}{b_{n-2}}).$$

This leads to

$$d_1(1 - \frac{\mu b_{n-1}}{b_{n-2}}) = a_{n-1} + t(1)b_{n-1} c_{n-1} - \mu b_{n-1} \frac{a_{n-2}}{b_{n-2}} + \lambda b_{n-1} d_n.$$
and then
\[
d_1 = \frac{a_{n-1} + b_{n-1}c_{n-1} - \bar{\mu}b_{n-1}\frac{a_{n-2}}{b_{n-2}}}{1 - \frac{b_{n-1}}{b_{n-2}}} + \frac{\tilde{\lambda}b_{n-1}}{1 - \frac{b_{n-1}}{b_{n-2}}}d_n,
\]
and by comparison to (8.5) the theorem is proved. \qed

**Lemma 8.1.** The series \( b_n \) given above satisfies
\[
0 < b_n/b_{n-1} < \rho.
\]
In other words, \( b_n \to 0 \) as \( n \to \infty \).

**Proof.** The proof is by induction.

**Induction base:** for \( n = 2 \)
\[
b_2/b_1 = \tilde{\lambda}/1 = \frac{\rho}{1+\rho} < \rho,
\]
and of course \( \tilde{\lambda} > 0 \).

**Induction assumption:** \( 0 < b_{n-1}/b_{n-2} < \rho \)

**Induction step:** From (8.4), \( b_n \) can be written as
\[
b_n = \frac{\tilde{\lambda}b_{n-1}}{1 - \frac{b_{n-1}}{b_{n-2}}} = \frac{\rho/b_{n-1}}{1 - \frac{b_{n-1}}{b_{n-2}}} = \rho b_{n-1} - \frac{1}{1 + \rho - \frac{b_{n-1}}{b_{n-2}}}.
\]

From the assumption
\[
b_{n-1}/b_{n-2} < \rho \Rightarrow 1 + \rho + \frac{b_{n-1}}{b_{n-2}} > 1 \Rightarrow 0 < \frac{1}{1 + \rho - \frac{b_{n-1}}{b_{n-2}}} < 1.
\]
Substituting this in (8.8) yields
\[
0 < b_n < \rho b_{n-1}
\]
proving the lemma. \qed
Lemma 8.2. \( a_n \xrightarrow{n \to \infty} \mathbb{E}\{D \mid k\} \)

Proof. From Theorem 8.2 we have \( \mathbb{E}\{D \mid k\} = a_n + b_n d(k - 1, n) \). From Lemma 8.1 \( b_n \xrightarrow{n \to \infty} 0 \) while \( d(k - 1, n) \) is bounded. Thus \( a_n \xrightarrow{n \to \infty} \mathbb{E}\{D \mid k\} \)

This provides the actual way to evaluate \( \mathbb{E}\{D \mid k\} \) iteratively, simply by computing \( a_n \) iteratively up to the required accuracy.

Special Cases

First note that for \( k = 0 \), all non-preemptive service policies yield the same results, thus

\[
\mathbb{E}\{D \mid 0\} = t^{(1)} \left( 1 + \rho - \frac{1 + \rho}{\rho} \ln(1 + \rho) \right), \tag{8.9}
\]

\[
\mathbb{E}\{D \mid 0\} \xrightarrow{\rho \to 1} t^{(1)}(2 - 2 \ln 2) \approx 0.613706, \tag{8.10}
\]

\[
\mathbb{E}\{D \mid 0\} \xrightarrow{\rho \to 0} 0. \tag{8.11}
\]

It is also simple to evaluate \( \mathbb{E}\{D \mid k\} \) when \( \rho \to 0 \). In this case an arriving customer will almost surely wait for the current customer to be served, then be served herself, i.e.

\[
\mathbb{E}\{D \mid k\} \xrightarrow{\rho \to 0} t^{(1)} \left( \frac{1}{k+1} + 1 - \frac{1}{k} \right) = \frac{t^{(1)}(k^2 - k - 1)}{k(k+1)}. \]

Numerical results and properties

Figure 8.2 depicts the conditional discrimination normalized by \( t^{(1)} \), \( \frac{\mathbb{E}\{D\mid k\}}{t^{(1)}} \), as a function of \( k \), for some values of \( \rho \). We note the following properties:

1. The highest value of negative discrimination is achieved at very high load and when \( k = 1 \). In this case the customer arrives and finds only the customer in service ahead of him, but then he ends up waiting for long time due to customers arriving after him. Thus a very high negative discrimination fits the natural intuition.
2. For all load values the highest value of negative discrimination is achieved when $k = 1$, which is, again, intuitive.

3. Relatively large values of positive discrimination are achieved when $k$ is large. In these cases the arriving customer gets positively discriminated since it is served ahead of $k - 1$ of those customers.

8.1.3 $P$-LCFS

Following the usual steps, we use the same state definition as in Section 8.1.2.

(3) The possible events are

1. A customer arrives into the system. The probability of this event is $\bar{\lambda}$. $C$’s state will change to $(a, b + 1)$

2. A customer leaves the system. The probability of this event is $\bar{\mu}$. If $C$ was in service, $C$ will leave the system, otherwise $C$’s state will change to $(a, b - 1)$.
This leads to the following set of equations:

\[ d(a, b) = t^{(1)} \delta(a, b) + \lambda d(a, b + 1) + \mathbb{1}(b > 0) \bar{\mu} d(a, b - 1). \]

Observe that an arriving customer starts service immediately and thus

\[ \mathbb{E}\{D \mid k\} = d(k, 0). \quad (8.12) \]

**Special Cases**

It is simple to evaluate \( \mathbb{E}\{D \mid k\} \) when \( \rho \to 0 \). In this case an arriving customer will almost surely be served in full and leave the system, thus

\[ \mathbb{E}\{D \mid k\} \xrightarrow{\rho \to 0} t^{(1)} \left( 1 - \frac{1}{k + 1} \right) \xrightarrow{k \to \infty} t^{(1)}. \]

**Numerical results and properties**

Figure 8.3 depicts the conditional discrimination normalized by \( t^{(1)} \), \( \frac{\mathbb{E}\{D \mid k\}}{t^{(1)}} \), as a function of \( k \), for some values of \( \rho \). We note the following properties:

1. The highest value of negative discrimination is achieved at very high load and when \( k = 0 \). In this case the customer arrives and finds no customer in service ahead of him, but then he has a high chance of ending up waiting for long time due to customers arriving after him. Thus a high negative discrimination fits the natural intuition.

2. For all load values the highest value of negative discrimination is achieved when \( k = 0 \), which is, again, intuitive.

3. Relatively large values of positive discrimination are achieved when \( k \) is large. In these cases the arriving customer gets positively discriminated since it is served ahead of those \( k \) customers.
8.1.4 ROS

(1) Let $a \in \mathbb{N}^0$ denote the number of customers in the system other than $c$. Let $q \in \{0, 1\}$ be a boolean variable equaling 1 if the customer is in service and 0 if it is waiting.

(2) From the state definition $N(a, q) = a + 1$, $s(a, q) = q$ and

$$\delta(a, q) = \mathbb{1}(q = 1) - \frac{1}{a + 1}.$$

(3) Possible events are:

1. A customer arrives into the system. The probability of this event is $\tilde{\lambda}$. $C$’s state will change to $\langle a + 1, q \rangle$.

2. A customer leaves the system, and $C$ is chosen to receive service next. The probability of this event is $\tilde{\mu}/a$. $C$’s state will change to $\langle a - 1, 1 \rangle$.

3. A customer leaves the system, and $C$ is not chosen to receive service next. The probability of this event is $\tilde{\mu}(a - 1)/a$. $C$’s change will change to $\langle a - 1, 0 \rangle$. 

Figure 8.3: Conditional (normalized) Discrimination for M/M/1 under P-LCFS
This leads to the following set of equations

\[ d(a, q) = t^{(1)} \delta(a, q) + \lambda d(a + 1, q) + \mathbb{1}(q = 0) \frac{\mu}{a} \left( d(a - 1, 1) + (a - 1)d(a - 1, 0) \right). \]

Observe that a customer arrives at the system either at \((0, 1)\) when the system is empty, or at \((k, 0)\) when it is not, thus

\[ \mathbb{E}\{D | k\} = \begin{cases} d(0, 1) & k = 0 \\ d(k, 0) & k > 0 \end{cases}. \] (8.13)

**Special Cases**

Again, for \(k = 0\) all non-preemptive service policies yield the same results \(8.9\)-\(8.11\).

**Numerical results and properties**

Figure 8.4 depicts the conditional discrimination normalized by \(t^{(1)}\), \(\frac{\mathbb{E}\{D(k)\}}{t^{(1)}}\), as a function of \(k\), for some values of \(\rho\). We note the following properties:

![Figure 8.4: Conditional (normalized) Discrimination for M/M/1 under ROS](image)
1. The discrimination values are significantly lower (both positive and negative) compared to LCFS. This demonstrates the fact that ROS violates the natural service order (FCFS) significantly less than LCFS.

2. For all load values, the discrimination reaches zero when $k$ is large. Recall that for PS discrimination value is constantly zero. This means that for large values of $k$ ROS behaves like PS. We conjecture that this is even more so for preemptive variants of ROS.

8.2 System Unfairness in M/M/1

In this section we derive the system unfairness $F_{D^2}$ for the same four policies, under the M/M/1 paradigm. We will follow steps (4) and (5) from the method in Section 7.1.

8.2.1 FCFS

The analysis $F_{D^2}$ for FCFS was already detailed in Section 7.2. (7.3) provides the set of equations for evaluating $d^{(2)}(a, b)$, (7.4) provides the arrival states and (7.5) provides their probabilities.

Remark 8.2. Note that similarly to Remark 8.1 $\frac{\mathbb{E}[D^2|k]}{\mathbb{E}[D^2]}$ also depends on $\lambda$ and $\mu$ only through their ratio $\rho$ and thus can be computed from the knowledge of $\rho$ alone. This is true for all the policies studied in this chapter.

8.2.2 LCFS

(4) We get the following set of equations:

$$d^{(2)}(a, b) = t^{(2)}(\delta(a, b))^2$$

$$+ \begin{cases} \tilde{\lambda} d^{(2)}(a, b + 1) + \tilde{\mu} d^{(2)}(a, b - 1) + 2t^{(1)}(a, b) \left( \tilde{\lambda} d(a, b + 1) + \tilde{\mu} d(a, b - 1) \right) & b > 0 \\ \tilde{\lambda} d^{(2)}(a + 1, b) + 2t^{(1)}(a, b) \tilde{\lambda} d(a + 1, b) & b = 0. \end{cases}$$
(5) Similarly to (8.3)

\[ \mathbb{E}\{D^2 \mid k\} = \begin{cases} d^{(2)}(k - 1, 1) & k > 0 \\ d^{(2)}(0, 0) & k = 0 \end{cases}, \]

and

\[ F_{D^2} = \sum_{k=0}^{\infty} p_k \mathbb{E}\{D^2 \mid k\} \]

\[8.2.3 \quad P-LCFS\]

(4) We get the following set of equations:

\[ d^{(2)}(a, b) = t^{(2)}(\delta(a, b))^2 + \tilde{\lambda}d^{(2)}(a, b + 1) + 2t^{(1)}\delta(a, b)\tilde{\lambda}d(a, b + 1) \]

\[ + \mathbbm{1}(b > 0) \left( \tilde{\mu}d^{(2)}(a, b - 1) + 2t^{(1)}\delta(a, b)\tilde{\mu}d(a, b - 1) \right). \]

(5) Similarly to (8.12)

\[ \mathbb{E}\{D^2 \mid k\} = d^{(2)}(k, 0). \]

\[8.2.4 \quad ROS\]

(4) We get the following set of equations:

\[ d^{(2)}(a, q) = t^{(2)}(\delta(a, 0))^2 + \tilde{\lambda}d^{(2)}(a + 1, q) + 2t^{(1)}\delta(a, b)\tilde{\lambda}d(a + 1, q) \]

\[ + \mathbbm{1}(q = 0) \left( \frac{\tilde{\mu}}{a}d^{(2)}(a - 1, 1) + \frac{\tilde{\mu}(a - 1)}{a}d^{(2)}(a - 1, 0) \right. \]

\[ + 2t^{(1)}\delta(a, b) \left( \frac{\tilde{\mu}}{a}d(a - 1, 1) + \frac{\tilde{\mu}(a - 1)}{a}d(a - 1, 0) \right) \right). \]

(5) Similarly to (8.13)

\[ \mathbb{E}\{D^2 \mid k\} = \begin{cases} d^{(2)}(k, 0) & k > 0 \\ d^{(2)}(0, 1) & k = 0 \end{cases}. \]
8.2.5 Numerical results and Discussion

![Graph showing system unfairness as a function of \( \rho \)](image)

Figure 8.5: System Unfairness (Variance of Discrimination) for M/M/1

Figure 8.5 depicts \( \frac{F_{D^2}}{t^2} \) as a function of \( \rho \) for the policies studied. The figure demonstrates the following properties:

1. In terms of unfairness the policies may be ranked as

   \[
   \text{P-LCFS} > \text{LCFS} > \text{ROS} > \text{FCFS} > \text{PS}. \tag{8.14}
   \]

   That is, P-LCFS is the most unfair and PS is the most fair. This dominance is observed for all values of \( \rho \), except for \( \rho = 0 \).

2. At \( \rho = 0 \) the unfairness measures of all policies converge, as expected, and all policies experience full fairness, \( F_{D^2} = 0 \). It is important to note that this results from the fact that most customers find an empty system upon arrival and thus are subject to no discrimination. Nonetheless if one conditions the discrimination on the system not being empty the discrimination is not zero.
3. The system unfairness seems to be monotonically increasing in the system utilization $\rho$ for all policies. For FCFS and ROS the increase is modest, and the values at $\rho = 0.9$ are relatively low. In fact the slopes of their curves suggest that when $\rho \to 1$ their values may converge to a finite number (in contrast, for example, to the expected delay, which blows up at $\rho = 1$). For LCFS and P-LCFS the increase is very drastic at high loads.

At this point it is interesting to compare these results with other queueing fairness criteria, in particular the results derived in Wierman and Harchol-Balter [155], see Section 2.1.1 and specifically Figure 2.1. In that work PS and P-LCFS are shown to be “always fair”, while all non-preemptive, non-sized based policies, including FCFS, LCFS and ROS, are shown to be “always unfair”.

Our results are in agreement regarding PS, which serves as the departure point for both measures, and regarding LCFS, which we find quite unfair. However, we differ significantly on FCFS and P-LCFS. The reason seems to be that the $E\{T(x)/x\}$ classifier deals with the expected performance of a size-$x$ job, averaged over all these jobs over all situations. The classifier emphasis is therefore on service requirements and it does not account directly for relative seniority. Thus P-LCFS whose seniority treatment is extremely unfair, as it discriminates against old jobs, is classified as fair, while FCFS whose seniority treatment is extremely fair is classified as unfair, maybe mostly because it is non-preemptive.

8.3 The Tradeoff Between Seniority and Service Requirements, and RAQFM’s Sensitivity To It

The analysis of the M/M/1 system demonstrated that when the seniority differences dominate the service requirement ones, RAQFM properly ranks the policies by their seniority preferences. Our aim in this section is to demonstrate the sensitivity of RAQFM to service requirement discrepancies and show that when this factor dominates the seniority factor, RAQFM reacts properly.

To this end we next consider a case where the arrivals remain Poisson while the vari-
ability of service requirements increases drastically. This is achieved by a bi-valued service requirement whose values are \( s = 0.1 \) with probability \( p \) and \( s' = 10 \) with probability \( 1 - p \). The value of \( p \) is selected to be \( p = \frac{90}{99} = 0.9009 \) so as to have mean service requirement of 1, identical to the previous numerical example. The variance of this service requirement is \( ps^2 + (1 - p)s'^2 = 9.1 \), in comparison to the variance of the M/M/1 case which is \( \frac{1}{\mu^2} = 1 \). This system is analyzed via a simulation program, which was run on each evaluated point for at least \( 10^6 \) customers. Figure 8.6 depicts \( F_{D^2} \) as a function of \( \rho \) for the FCFS, LCFS and P-LCFS policies. The figure demonstrates that over the range

\[
\rho = (0, 0.55)
\]

RAQFM ranks P-LCFS as the most fair among the three policies, in contrast to its ranking in the M/M/1 case. As such, it concurs, in this case, with the ranking of the slow-down fairness approach in Wierman and Harchol-Balter [155] and is in contrast with the order fairness approach in Avi-Itzhak and Levy [7].

To understand this, note that due to the large variability of service requirements, large discriminations (and unfairness values) are formed when a large job is served and many small jobs queue behind it. In such cases preemption of large jobs from service can
alleviate this problem. P-LCFS achieves this since it tends to preempt the large jobs with high probability. Thus, in the case of high variability service requirements, where service requirement differences dominate seniority differences, P-LCFS can be more fair than FCFS due to giving preferential treatment to short jobs over long jobs, despite its preferential service to less-senior jobs over more-senior jobs.

In summary, the examples presented in Figures 8.5 and 8.6 demonstrate how RAQFM accounts for the tradeoffs between seniority differences and service requirement differences. It should be noted that in the second example the lower unfairness of P-LCFS does not hold for the whole range of utilizations. For high load situations P-LCFS becomes again the most unfair policy. This may possibly be attributed to the fact that at high loads the queue size tends to be large and thus the magnitude of order discrepancies may increase sharply (similarly to the waiting time variance).

8.4 Going Beyond Exponential Service Requirement

Lately, Brosh et al. [27] has researched RAQFM for single server systems, in more general settings, for the major purpose of researching the effect of variability on RAQFM. We bring here the major results of this work for completeness.

Let $S$ be a random variable denoting the service requirement of customer $C$, with first two moments $s^{(1)}$ and $s^{(2)}$, variance $\sigma_s^2$ and coefficient of variation $\gamma_s \eqdef \sigma_s/s^{(1)}$.

First, they analyze P-LCFS under a general M/G/1 model, and show that

$$
\mathbb{E}\{D \mid k\} = s^{(1)} \left[ \left( 1 - \frac{1}{k+1} \right) + \lambda \mathbb{E}\{R \mid k+1\} \right],
$$

where $R$ is the warranted service, recall from Section 4.1 and $R \mid k$ is the warranted service conditioned on seeing $k$ customers upon arrival, which they show to equals

$$
\mathbb{E}\{R \mid k\} = - \frac{s^{(1)}}{\rho^{k+1}} \sum_{i=0}^{\infty} \frac{\rho^{k+i+1}}{k+i+1},
$$
The unfairness is
\[ F_{D^2} = s^{(2)} \sum_{k=0}^{\infty} (1 - \rho) \rho^k \left( 1 - \frac{1}{k+1} + \lambda \mathbb{E}\{R \mid k+1\} \right)^2 + \rho \sum_{k=0}^{\infty} (1 - \rho) \rho^k \mathbb{E}\{R^2 \mid k+1\}, \]

where \( \mathbb{E}\{R^2 \mid k\} \) can be evaluated using

\[ \mathbb{E}\{R^2 \mid k\} = s^{(2)} \left( \frac{-1}{k+1} + \lambda \mathbb{E}\{R \mid k+1\} \right)^2 + \lambda s^{(1)} \mathbb{E}\{R^2 \mid k+1\}. \]

This shows that the unfairness of P-LCFS under M/G/1 is dependent only on the first two moments of the service requirement.

For the rest of the service policies, FCFS, LCFS, ROS and P-ROS, the service distribution is approximated by a Coxian distribution using a second order moment matching technique (see Adan and Resing [2], Chap. 2). In particular, a phase-type distribution is fitted, either Coxian or Erlangian, to the mean \( s^{(1)} \) and to the coefficient of variation \( \gamma_s \). A distinction is made between two cases: (a) When \( 0 < \gamma_s < 1 \) the service requirement is matched to a \( k \)-stage Erlang distribution \( E_k \) and (b) When \( \gamma_s > 1 \) a Coxian-2 distribution is used which is composed of two exponential stages.

The analysis of the system unfairness follows a method very similar to the one described in Section 7.1. We bring the analysis for a M/Er/1 system under FCFS as an example in Appendix C, compare with Section 7.2.

Next, the effect of a long service customer, \( C_L \), on the discrimination in a non-preemptive system is discussed. In the model observed \( C_L \) has an exponential service requirement with parameter \( \mu_2 \) such that \( \lambda \gg \mu_2 \). Time is slotted by arrivals and departures, and the probability of a slot ending with an arrival of another customer, rather than the departure of \( C_L \), is \( p = \lambda / (\lambda + \mu_2) \). It is shown that the expected negative discrimination observed by a small customer is at most

\[ \approx -\frac{1}{\lambda + \mu_2} \ln\left( \frac{1}{1-p} \right) \approx -\frac{1}{\lambda} \ln\left( \frac{1}{1-p} \right) \approx -\frac{1}{\lambda} \ln\left( \frac{\lambda}{\mu_2} \right), \]
while the expected positive discrimination of $C_L$ is at most

$$\approx \frac{1}{\lambda + \mu_2} \left( \frac{1}{1 - p} - \ln\left(\frac{1}{1 - p}\right) \right) \approx \frac{1}{\lambda} \left( \frac{\lambda}{\mu_2} - \ln\left(\frac{\lambda}{\mu_2}\right) \right).$$

Interestingly the positive discrimination of $C_L$ is proportional to $1/\mu_2$ while the negative discrimination of a customer is only proportional to its logarithm.

Lastly, numerical results show the effect of variability. While for low variability the unfairness follows $P\text{-}LCFS > LCFS > ROS > P\text{-}ROS \approx FCFS > PS$, which is identical to (8.14), for high variability $LCFS \approx ROS \approx FCFS > P\text{-}LCFS > P\text{-}ROS > PS$. The authors conclude that preemption is an instrumental tool for increasing fairness when job sizes variability is large.

### 8.5 Summary

We studied four service disciplines under the $M/M/1$ regime: FCFS, LCFS, P-LCFS and ROS. For each of the service policies we derived the expected discrimination experienced by an arriving customer as a function of the number of customers it finds in the system upon arrival. We examined this behavior in detail for each of the service policies and the results were used to shed light on the behavior of the discrimination under these policies.

Under the same settings, we derived the system unfairness, measured via the variance of discrimination, for the same four policies. Our results indicate that under exponential distribution the unfairness is monotonically increasing with the system load $\rho$, and the order of unfairness is $P\text{-}LCFS > LCFS > ROS > FCFS > PS$.

We studied the behavior of the fairness for a bi-valued service distribution with large variability. The results showed that for this distribution, and under some loads, typically the not too high ones, $P\text{-}LCFS$ can become more fair than $LCFS$ and even $FCFS$, mainly due to preemption. We presented results from Brosh et al. [27], which studied general distributions and came to similar conclusions. Their study is an example of how to extend our analysis methods beyond the exponential service case.
Chapter 9

FAIRNESS OF MULTIPLES SERVER AND MULTIPLE QUEUE SYSTEMS

In this chapter we turn to investigate the fairness considerations in operating multiple server and multiple queue systems. We limit our discussion to servers with the same service rate, for simplicity a rate of one. Most of the material in this chapter was published in Raz et al. [122] and also appears in Raz et al. [119].

Multiple server queueing models have been used in a large variety of applications, including computer systems, call centers and Web servers, as well as human waiting lines, for example in airports. The configuration and operation of multi-server systems involves a variety of operational mechanisms that must be chosen. These include the number of queues and their dedication, the service policy, the queue joining policy and queue jockeying rules.

An important motivation, perhaps the most important one, behind these multi-server mechanisms, is the wish to provide fair service to the jobs. The importance of these multi-server mechanisms, at least in the eyes of customers, was reinforced by psychological experimental queueing studies Rafaeli et al. [112], [113]. These experimental studies have further shown that the single-queue system is perceived to be more just (fair) compared to the multi-queue system.

Our main purpose is to approve, in a quantitative manner, several rules of thumb that have been widely believed by practitioners and theoreticians but for which no analytic quantitative support was given. Particular design issues that we address are: 1) Whether to combine queues or not, 2) Whether to allow queue jockeying, and in what form, and 3) Queue joining policy.
Note that this is to our knowledge the first attempt to provide quantitative analysis of fairness for multi-queue multi-server systems; Most other fairness studies were limited to single server systems (the only exception we are aware of is Gordon [58]).

The structure of this chapter is as follows. We start our analysis (Section 9.1) by considering G/D/m type systems, that is, systems with a general arrival process and deterministic service requirements. We study the queuing mechanisms mentioned above, i.e. combining queues, jockeying, and queue joining policy. We then (Section 9.2) turn to study M/M/m type systems, namely systems with Poisson arrivals and exponential service requirements. We provide an exact fairness analysis for the same mechanisms, which demonstrates that in general the same properties shown for the G/D/m case hold also for the M/M/m one. We then (Section 9.3) turn to examine G/G/m type systems. We show, via some counter-examples, that the results derived in Section 9.1 and Section 9.2 do not necessarily hold for an arbitrary G/G/m model. We provide sufficient conditions under which some of them hold for the G/GI/m model.

Another multi-server mechanism is that of combining servers. Although this issue could have been studied in this chapter, we chose to study it in the context of multiple class systems in Section 10.4, see there.

9.1 Fairness of G/D/m Type Systems

In this section we analyze the fairness of G/D/m type systems, i.e. the customers arrive according to a general arrival process, the service requirements of all customers are identical, and the system has M servers.

9.1.1 The Effect of Seniority on Fairness

We start with analyzing the effect of customer seniority on fairness, and show that under the G/D/m model serving customers in order of seniority increases system fairness.

For simplicity we choose to demonstrate this for adjacently served customers. Consider two customers $C_j$ and $C_k$ that are adjacently served, i.e. they are scheduled to begin
service adjacently, with arrival times $a_j < a_k$ and equal service requirements, i.e. $s_j = s_k$.

Observe two possible cases: In the first case (Figure 9.1) they are served sequentially, by the same server. In the second case (Figure 9.2) they are served by two different servers in a partially-parallel manner.

We have two possible schedules: In schedule (a), (Figure 9.1(a), Figure 9.2(a)), the order of seniority is preserved, i.e. $C_j$ is served before $C_k$. In schedule (b), (Figure 9.1(b), Figure 9.2(b)), the order of service of $C_j$ and $C_k$ is interchanged and thus the order of seniority is violated. We assume that both of the schedules are possible, i.e. that $C_j$ and $C_k$ reside in the queue together for some time.

![Diagram](image)

**Figure 9.1: Case 1 - Sequentially Served Customers By The Same Server**

**Lemma 9.1** (Preference of Seniority Between Adjacently Served Customers with Equal Service Requirement). Let $C_j$ and $C_k$ be two customers, where $a_j < a_k$ and their service requirements are equal, i.e. $s_j = s_k = s$. Then the unfairness (measured as $F_{D2}$) of the seniority preserving schedule (a) is smaller than the unfairness of the seniority violating
schedule (b), for every arrival pattern, regardless of the service requirements of the other customers.

Proof. For simplicity we first prove this for adjacently served customers.

We need to consider both cases. We note that the first case (Figure 9.1) was proven in Theorem 5.1, see Remark 5.1. We therefore only need to prove the second case (Figure 9.2).

Let $D_i^a$ and $D_i^b$ denote the discrimination of $C_i$ under schedule (a) and schedule (b) respectively. Let $F_{D2}^a$, $F_{D2}^b$ denote the unfairness in the respective schedules, and let $N^a(t)$, $N^b(t)$ denote the number of customers in the system at epoch $t$, respectively.

Let $\hat{F}$ denote the total unfairness to all customers other than $C_j$ and $C_k$, and let $\hat{\hat{F}}$ denote the total unfairness to $C_j$ and $C_k$. Then $F_{D2} = \hat{F} + \hat{\hat{F}}$.

Define $\Delta F_{D2}$, $\Delta \hat{F}$, and $\Delta \hat{\hat{F}}$ to be the change, due to the interchange between $C_j$ and

![Diagram of Seniority Preserving and Violating Schedules](image)
Chapter 9. Fairness of Multiple Server Systems

97

in the values of $F_{D^2}$, $\hat{F}$, and $\tilde{F}$ respectively. We need to prove that $\Delta F_{D^2} \geq 0$, where

$$\Delta F_{D^2} = \Delta \hat{F} + \Delta \tilde{F} = \frac{1}{L} \left[ (D_{j}^h)^2 + (D_{k}^h)^2 - (D_{j}^a)^2 - (D_{k}^a)^2 \right]$$

$$+ \frac{1}{L} \left[ \sum_{i \neq j,k} (D_{i}^h)^2 - \sum_{i \neq j,k} (D_{i}^a)^2 \right], \quad (9.1)$$

where $L$ is the number of customers in the observed scenario.

We divide the time interval $(a_j, \max(d_j, d_k))$ (namely from the arrival of $C_j$ until both $C_j$ and $C_k$ depart) into five sub-intervals, where interval $i, i = 1, 2, 3, 4, 5,$ is $(E_i, E_{i+1})$, and where $E_1 = a_j; E_2 = a_k; E_3$ is the first point in time where service to either $C_j$ or $C_k$ starts; $E_4$ is the point in time where service to the second customer starts; $E_5 = E_3 + s; E_6 = E_4 + s$ (see Figure 9.2). As customers are adjacently served, $E_5 > E_4$.

We note that the number of customers in the system under schedule (a), $N^a(t)$, and the number of customers in the system under schedule (b), $N^b(t)$, are identical for every $t$. Therefore, the warranted service in interval $i$ (of any customer) is the same in both schedules, and we denote it $R_i$. As the warranted service in each interval is the same in both schedules, and as other customers’ departure epochs are also the same in both schedules, the interchange of $C_j$ and $C_k$ affects only $C_j$ and $C_k$, i.e. $D_{i}^a = D_{i}^b, i \neq j, k$.

Therefore $\Delta \hat{F} = 0$, and from (9.1) it follows that

$$\Delta F_{D^2} = \Delta \tilde{F} = \frac{1}{L} \left[ (D_{j}^h)^2 + (D_{k}^h)^2 - (D_{j}^a)^2 - (D_{k}^a)^2 \right]$$

$$= \frac{1}{L} \left[ (s - (R_{(1)} + R_{(2)} + R_{(3)} + R_{(4)} + R_{(5)}))^2 + (s - (R_{(2)} + R_{(3)} + R_{(4)}))^2 \right.$$

$$\left. - (s - (R_{(1)} + R_{(2)} + R_{(3)} + R_{(4)}))^2 - (s - R_{(2)} + R_{(3)} + R_{(4)} + R_{(5)})^2 \right] = \frac{2}{L} R_{(1)} R_{(5)} > 0,$$

where the inequality is due to $R_{(i)} > 0, i = 1, 2, 3, 4, 5$.

This completes the proof for adjacently served customers. To address non adjacently served customers one has to address a third case, similar to case 1 where the service of the
second customer starts after the service completion of the first customer. The correctness of this case is very similar to that of case 1.

\textbf{Theorem 9.1} (Fairness of FCFS and LCFS under G/D/m). \textit{If the service requirements of all customers are identical, then for every arrival pattern, FCFS is the least unfair service policy in }\Phi\textit{ and LCFS is the most unfair one.}

Note that by FCFS we mean global FCFS, i.e. serving customers by order of arrival. The proof is exactly the same as the proof of Corollary 5.1.

\subsection*{9.1.2 The Effect of Multiple Queues Operation Mechanisms on Fairness}

In this section we study the effect multiple queues mechanisms have on fairness, by observing several design and service decision considerations and examining their effect on the system fairness.

One such decision is commonly referred to in the literature as the “Combining Queues” problem, in which a system consisting of \( M \) queues, each having service rate \( \mu \) and arrival rate \( \lambda \), and denoted the \textit{multi-queue system} or the \textit{separate queue system}, is compared to a system with one queue served by \( M \) servers with same service rate \( \mu \), and a corresponding arrival rate \( M\lambda \), denoted the \textit{single queue system} or the \textit{combined queue system}. It is widely known that the single queue system is more efficient, i.e. it has a lower mean delay (a proof is given in Smith and Whitt [143]). However, if jockeying is allowed when a server is idle, in a manner that is nondiscriminatory with respect to the service requirement, the systems have the same mean waiting time. In other cases, for example when jockeying favors short jobs, the single queue system might even become less efficient than the separate queue system, see Rothkopf and Rech [129] for a full discussion in the matter. As mentioned above, in Rafaeli et al. [112, 113] an experimental psychology study showed that a single queue system is perceived to be more fair than a multiple queue one. Our goal is to quantitatively compare the fairness of the combined queue G/D/m system to that of the separate queue one. We show that indeed the combined (single) queue system is more fair than the separate (multiple) queue system.
A second issue, assuming multiple queues are used, is jockeying. In some systems jockeying is possible and we would like to find the effect it has on the system fairness. We focus on jockeying done while a server is idle, in a manner that is nondiscriminatory with respect to the service requirement. We leave other forms of jockeying for future research.

The third issue, assuming jockeying is possible, is whether jockeying should be done from the head of the queue or from its tail. Note that from the efficiency (mean delay) point of view there is no difference between the two manners, as long as the jockeying is nondiscriminatory with respect to the service requirement.

For the sake of illustration, we analyze four systems. All four systems have $M$ servers. The first system has one common queue. A customer arriving at the system joins the tail of the queue, and when a server becomes free it serves the customer at the head of the queue, thus the system is non-idling\(^1\). The customers are therefore admitted into service according to their order of arrival in a global FCFS manner. We refer to this system as the single queue system.

The second system has $M$ queues, where each of the servers serves only customers joining its assigned queue, in a FCFS manner. Customers arriving at the system are assigned to one of the queues in a uniformly random manner, and jockeying between the queues is not permitted at any time. We refer to this system as the standard multiple queue system. In Section 9.1.3 we address systems where queue joining isn’t done in a random manner.

The third and fourth systems have $M$ dedicated FCFS queues, and jockeying is allowed at epochs when a server is idle. Again, arriving customers are assigned to one of the queues in a uniformly random manner, however, if one or more of the servers is idle, an arriving customer is assigned to one of the idle servers, in a uniformly random manner. Following that, if a server becomes idle, a customer from one of the non-empty queues is immediately assigned to that server. The queue is chosen in a uniformly random manner, and the customer is chosen either from the head of the queue or from the tail of the queue.

\(^1\) A multi-server system is non-idling if the number of busy servers (and thus the actual service rate) is always $\min(N,M)$, where $N$ is the number of customers in the system.
We refer to the first as the head-jockeying-on-idle multi-server system and to the second as the tail-jockeying-on-idle system.

As mentioned above, both jockeying-on-idle systems are known to be as efficient as the single queue system, while the standard system is known to be less efficient than the other three systems.

**Theorem 9.2** (Single Queue is More Fair Than Multiple Queue for G/D/m). If the service requirements of all customers are equal, then for every arrival pattern, the single queue system has the lowest unfairness among the systems mentioned above.

**Proof.** The proof is immediate since serving in a single queue means serving in a global FCFS manner, which is proven in Theorem 9.1 to be the most fair policy. □

**Theorem 9.3** (Head-Jockeying-On-Idle is More Fair Than Tail-Jockeying-On-Idle for G/D/m). If service requirements of all customers are equal, and the choice of queue from which jockeying is done is identical, then for every arrival pattern the head-jockeying-on-idle system has lower unfairness than the tail-jockeying-on-idle system.

**Proof.** We prove that if the choice of queue from which jockeying is done is identical, the policy with the lowest unfairness is one where jockeying is always done from the head of the queue. Specifically, this proves that head-jockeying is more fair than tail-jockeying.

We prove this by way of contradiction. Assume for the contradiction that there exists a jockeying policy $\psi$ by which the jockeying customer is not always the customer at the queue head, and that $\psi$ is the policy with the lowest unfairness for every arrival pattern. We will show that a policy with lower unfairness can be constructed.

Consider an arrival pattern where there exists an epoch $e$ in which a server was idle, and in which the customer admitted to the idle server by $\psi$ was the $n$-th customer in some non-empty queue, and $n > 1$. Say there are $N$ customers in that queue, and denote the customers in that queue $C_1, C_2, \ldots, C_N$, according to their order of arrival, with $C_1$ being the first to arrive. We show that admitting $C_n, n > 1$ to the idle server has higher unfairness than admitting $C_{n-1}$. 
Chapter 9. Fairness of Multiple Server Systems

Say $C_n$ was admitted to the idle server at epoch $e$, and $C_{n-1}$ was served at some later epoch $e' > e$. We construct a policy $\psi'$ that exactly imitates $\psi$, except it exchanges the roles of $C_n$ and $C_{n-1}$, i.e. $C_{n-1}$ is admitted at epoch $e$ and $C_n$ is served at epoch $e'$. Since the customers are adjacent in the queue and have equal service requirements no other customers are influenced by this exchange. Therefore, using the same notations and arguments used in Lemma 9.1 \( \Delta F = \hat{\Delta} F \) and we need only to prove that $\hat{\Delta} F > 0$. For brevity and comparability with the proof of Lemma 9.1 we denote $j = n - 1, k = n$.

Let the service requirement of $C_j$ and $C_k$ be $s$. Two cases should be considered: either $e' - e < s$ or $e' - e \geq s$. For $e' - e < s$ observe that the situation is exactly the one depicted in Figure 9.2 and as shown in the proof of Lemma 9.1 $\hat{\Delta} F > 0$ in this case.

For the second case, $e' - e \geq s$, the situation is depicted in Figure 9.3. We divide the time interval $(a_j, e' + s)$ (namely from the arrival $C_j$ until both $C_j$ and $C_k$ depart) into five sub-intervals where interval $i, i = 1, 2, 3, 4, 5$, is $(E_i, E_{i+1})$, where $E_1 = a_j; E_2 = a_k$;
$E_3$ is the jockeying epoch; $E_4 = E_3 + s$; $E_5$ is the epoch where the second customer is served, $e'$; $E_6 = E_5 + s$, see Figure 9.3.

Let the service warranted in interval $i$ be $R(i)$. Using (9.1) the difference in unfairness is

$$\Delta \hat{F} = \frac{1}{L}[\left(s - (R(1) + R(2) + R(3) + R(4) + R(5))\right)^2 + \left(s - (R(2) + R(3))\right)^2$$

$$- \left(s - (R(1) + R(2) + R(3))\right)^2 - \left(s - (R(2) + R(3) + R(4) + R(5))\right)^2] = \frac{2}{L}R(1)(R(4) + R(5)) > 0.$$ 

We therefore showed that for every service policy $\psi$ that performs jockeying not always from the head of the queue for at least one arrival pattern, a better policy $\psi'$ exists, in contradiction to $\psi$ being the policy with the lowest unfairness for every arrival pattern. Therefore, the policy with the lowest unfairness for every arrival pattern must be one where the jockeying customer is always the one at the queue head.

Remark 9.1. We have also shown that the worst policy is a policy in which the customer admitted is always the one at the queue tail.

Conjecture 9.1 (Jockeying-On-Idle is More Fair Than Standard Multi-Queue). If the service requirements of all customers are equal, then for every arrival pattern the standard system is less fair than the jockeying-on-idle systems.

We base this conjecture on multiple simulations and on intuition. Intuitively speaking, allowing jockeying allows more servers to work simultaneously, thus distributing the service rate between more customers.

Remark 9.2 (Non-Equal Service Rate). As mentioned above we limit our discussion to servers with the same service rate, for simplicity a rate of one. It can be easily shown that when this is not the case, the properties discussed above are not necessarily true. For example, a more fair policy than FCFS might hold a senior customer in order to serve it by a faster server.
9.1.3 The Effect of Queue Joining Policy on Fairness

In this section we address the effect of the queue joining on fairness. To this end we analyze the head-jockeying-on-idle system with three queue joining policies: (a) customers join queues at random (RAND) (b) customers always join on of the shortest queues (SQ), and (c) customers join queues in a round-robin manner (RR).

The efficiency aspects of this issue were thoroughly studied, starting with Winston where the Shortest Queue (SQ) policy is shown to be optimal for exponential arrivals and service requirements. Further work was done on extending the results for more general systems, for example Ephremides et al. where two identical exponential servers are considered, with similar results, and Hordijk and Koole where a more general system is considered. Some cases where SQ is not optimal are given in Whitt.

Note that in practice selection of the queue joining policy can be affected by several physical factors of the system. These include who is making the queue joining decision, the customer or a queue manager, and whether queue size information is available to the decision maker.

**Theorem 9.4** (Relative Fairness of Queue Joining Policies under the G/D/m Model). If all service requirements are equal, then the unfairness in SQ equals that in RR, and both have the lowest unfairness.

**Proof.** To prove that the unfairness for SQ equals that of RR note that if service requirements are equal, SQ and RR are in fact identical, since using round-robin always directs a customer to the shortest queue.

Second, note that using SQ (or RR) yields an order of service which is in fact identical to global FCFS, and from Theorem 9.1 global FCFS is the most fair order of service.

9.1.4 Numerical Results for the M/D/2 Model

In this section we provide numerical results from simulation on a system with two servers (M/D/2) to demonstrate the issues mentioned in the previous sections.
Figure 9.4 depicts numerical results from simulating the M/D/2 model as a function of the system utilization factor $\rho = s\lambda/2$ for the four systems discussed above, where the queue to join is chosen randomly. Service requirement was set as one unit ($s = 1$). Each point is the result of simulating the passage of at least $10^6$ customers through the system. The figure demonstrates the following properties:

1. Except for one system and one point, single queue system at high load, the unfairness is monotone increasing with the system utilization factor $\rho$. This monotonic behavior was already demonstrated for single server systems. The reason for the decrease in unfairness at high loads for the single queue system is still unclear and further study is required.

2. The single queue system is more fair than all the multiple queue systems, for every system load, as expected from Theorem 9.2. The ratio between the unfairness of the single queue system and that of the head-jockeying-on-idle system increases with $\rho$, and reaches a ratio of more than 1 : 2 for $\rho = 0.9$. Recall that as long as jockeying is allowed only in a manner that is nondiscriminatory with respect to the service.
requirement, as is the case with the multiple queue systems analyzed, the single queue system is also at least as efficient as the multiple server systems.

3. Among the multiple queue systems, the jockeying-on-idle systems are more fair than the standard (idling) system, for every system load. The ratio between the unfairness of the standard system and that of the head-jockeying-on-idle system decreases with $\rho$, starting with a high ratio of more than $1:8$ for $\rho = 0.1$ and reaching a low of $1:1.9$ for $\rho = 0.9$. This agrees with Conjecture 9.1.

4. Among the jockeying-on-idle systems, head-jockeying is more fair than tail-jockeying for every system load, as expected from Theorem 9.3. The difference between them increases with $\rho$ and is as high as $25\%$ for $\rho = 0.9$. Recall that these two systems are equally efficient.

In Figure 9.5 the three queue joining policies discussed in Section 9.1.3 (head-jockeying-on-idle) are compared to each other, as a function of the system utilization factor $\rho$, for the M/D/2 model. Service requirement was set as one unit. Each point is the result of simulating the passage of at least $10^6$ customers through the system. The figure demonstrates the following properties:

1. SQ and RR are identical, as expected from Theorem 9.4. In fact, they are identical to the single queue M/D/2 system.

2. RAND has higher unfairness than SQ and RR for every system load, as expected from Theorem 9.4. The difference between them increases with $\rho$ and is as high as $90\%$ for $\rho = 0.9$. Recall that these two systems are equally efficient.

9.2 Fairness of M/M/m Type Systems

In this section we analyze the fairness of M/M/m type systems. We follow the method for analysis of Markovian systems presented in Section 7.1 and analyze the same systems presented in Section 9.1.
Chapter 9. Fairness of Multiple Server Systems

Figure 9.5: Unfairness in the M/D/2 Model - Comparison of Queue Joining Policies

9.2.1 The Effect of Multiple Queues Operation Mechanisms on Fairness

In this section we provide quantitative fairness analysis, under the M/M/m model, for the four systems described in Section 9.1.2, namely the single queue system, the standard multiple queue system, and the two jockeying-on-idle systems - head-jockeying-on-idle and tail-jockeying-on-idle. Recall that for all four systems the order of service within each queue is FCFS, except for jockeying.

Fairness in Multiple Server Single Queue Systems under the M/M/m Model

Following the method in Section 7.1

(1) Let \( a = 0, M, M + 1, \ldots \) denote the number of customers ahead of \( C \), including served customers, and \( b \in \mathbb{N}^0 \) the number of customers behind \( C \). If \( C \) is in service, \( b \) also includes customers served by other servers. Note the unavoidable jump occurring in the value of \( b \) when \( C \) enters service, and the fact that the values \( 1 \leq a < M \) are invalid. Due to the Markovian nature of the system the state \( (a, b) \) captures all that is needed to predict the future of \( C \).
(2) The number of customers in the system at a slot where \( C \) observes the state \( \langle a, b \rangle \) is \( N(a, b) = a + b + 1 \). The rate of service given to \( C \) at that slot is \( \mathbb{1}(a = 0) \) and the total service rate is a constant \( \omega(a, b) = \sigma(a, b) \), where \( \sigma(a, b) = \min(M, a + b + 1) \) is the number of active servers. The rates of arrival and departure are \( \mu(a, b) = \sigma(a, b)\mu \) and \( \lambda(a, b) = \lambda \).

The momentary discrimination at state \( \langle a, b \rangle \), denoted \( \delta(a, b) \), is therefore given by

\[
\delta(a, b) = \mathbb{1}(a = 0) - \frac{\sigma(a, b)}{a + b + 1}
\]

and the moments of the slot length are,

\[
t^{(1)}(a, b) = \frac{1}{\lambda + \sigma(a, b)\mu}, \quad t^{(2)}(a, b) = \frac{2}{(\lambda + \sigma(a, b)\mu)^2} = 2(t^{(1)})^2, \quad a, b \geq 0.
\]

(3) Possible events are:

1. A customer arrives to the system. The probability of this event is \( \tilde{\lambda}(a, b) \overset{\text{def}}{=} \lambda / (\lambda + \sigma(a, b)\mu) \). \( C \)’s state changes to \( \langle a, b + 1 \rangle \).

2. For \( a > 0 \): A customer leaves the system. The probability of this event is \( M\tilde{\mu}(a, b) \)
   where \( \tilde{\mu}(a, b) \overset{\text{def}}{=} \mu / (\lambda + \sigma(a, b)\mu) \). If \( a > M \), \( C \)’s state changes to \( \langle a - 1, b \rangle \). Otherwise \( (a = M) \), \( C \)’s state changes to \( \langle 0, a + b - 1 \rangle \).

3. For \( a = 0, b > 0 \): A customer other than \( C \) leaves the system. The probability of this event is \( (\sigma(a, b) - 1)\tilde{\mu}(a, b) \). \( C \)’s state changes to \( \langle a, b - 1 \rangle \).

4. For \( a = 0 \): \( C \) leaves the system. The probability of this event is \( \tilde{\mu}(a, b) \).

Note that here and throughout the chapter \( \tilde{\lambda} \) refers to the probability of an arrival of any customer, while \( \tilde{\mu} \) refers to the probability of a departure of a customer from one specific queue. This seeming inconsistency is required for mathematical brevity.
This leads to the following set of linear equations

\[
d(a, b) = t^{(1)}(a, b) \delta(a, b) + \tilde{\lambda}(a, b)d(a, b+1) + \tilde{\mu}(a, b)\begin{cases}
Md(a-1, b) & a > M \\
Md(0, a + b - 1) & a = M \\
(\sigma(a, b) - 1)d(a, b - 1) & a = 0, b > 0
\end{cases}
\]

(4) Similarly, the equations for \( d^{(2)}(a, b) \) are

\[
d^{(2)}(a, b) = t^{(2)}(a, b)(\delta(a, b))^2 + \tilde{\lambda}(a, b)d^{(2)}(a, b+1) + \tilde{\mu}(a, b)\begin{cases}
Md^{(2)}(a-1, b) & a > M \\
Md^{(2)}(0, a + b - 1) & a = M \\
(\sigma(a, b) - 1)d^{(2)}(a, b - 1) & a = 0, b > 0
\end{cases}
\]

(5) Note that a customer can arrive at either the states \( \langle k, 0 \rangle, k = M, M + 1, \ldots \) or the states \( \langle 0, k \rangle, k = 0, \ldots, M - 1 \) where \( k \) is the number of customers seen on arrival, i.e.

\[
\mathbb{E}\{D^2 \mid k\} = \begin{cases}
d^{(2)}(k, 0) & k \geq M \\
d^{(2)}(0, k) & k < M
\end{cases}
\]

Therefore

\[
F_{D^2} = \sum_{k=0}^{\infty} p_k \mathbb{E}\{D^2 \mid k\},
\]

where \( p_k \) is the steady state probability of encountering \( k \) customers in the system, which is

\[
p_k = \begin{cases}
p_0 \frac{(M\rho)^k}{k!} & 1 \leq k \leq M \\
p_0 \frac{\rho^k M^M}{m!} & k \geq M
\end{cases}, \quad p_0 = \left[ \frac{(M\rho)^M}{M!(1-\rho)} + \sum_{k=0}^{M-1} \frac{(M\rho)^k}{k!} \right]^{-1}, \quad (9.2)
\]

(e.g. Kleinrock [88], Sec. 3.5).
In the case \( m = 2 \) these are equal to

\[
p_k = p_0 2 \rho^k \quad k \geq 1, \quad p_0 = \frac{1 + \rho}{1 - \rho}.
\]

**Fairness in Multiple Server Multiple Queue Systems**

We now analyze the standard multiple queue system, as described in Section 9.1.2. Consider an arbitrary tagged customer \( C \). When \( C \) arrives into the system it joins a queue at random. We refer to the queue that the customer joined as the *local* queue and to the rest of the queues as the *other* queues. For the analysis we index the local queue 1, and the other queues 2, \ldots, \( m \), in some arbitrary way.

(1) Let \( a \in \mathbb{N}^0 \) be the number of customers ahead of \( C \) in the local queue, including the served customer. Let \( b \in \mathbb{N}^0 \) be the number of customers behind \( C \) in the local queue. Let \( l_i \in \mathbb{N}^0 \) be the number of customers in the \( i \)-th queue, \( i = 2, \ldots, m \), including any served customers, and let \( \mathbf{l} \) denote the vector \((l_2, \ldots, l_m)\). Due to the Markovian nature of the system, the state \((a, b, \mathbf{l})\) captures all that is needed to predict the future of \( C \).

(2) Define \( \Sigma_v \), the sum of the elements of a vector \( v \). Let \( \sigma(\mathbf{l}) \) denote the number of active servers, and since the server at the local queue is active as long as \( C \) is in the system

\[
\sigma(\mathbf{l}) = 1 + \sum_{i=2}^{M} \mathbb{1}(l_i > 0).
\]

Since \( N(a, b, \mathbf{l}) = \Sigma_l + a + b + 1, \omega(a, b, \mathbf{l}) = \sigma(\mathbf{l}) \) and \( s(a, b, \mathbf{l}) = \mathbb{1}(a = 0) \) we have the momentary discrimination

\[
\delta(a, b, \mathbf{l}) = \mathbb{1}(a = 0) - \frac{\sigma(\mathbf{l})}{\Sigma_l + a + b + 1}.
\]

Since \( \mu(a, b) = \sigma(\mathbf{l}) \mu \) and \( \lambda(a, b) = \lambda \) we have

\[
t^{(1)}(a, b, \mathbf{l}) = \frac{1}{\lambda + \sigma(\mathbf{l}) \mu}, \quad t^{(2)}(a, b, \mathbf{l}) = \frac{2}{(\lambda + \sigma(\mathbf{l}) \mu)^2} = 2(\delta(1))^2.
\]
And since they are independent of $a$ and $b$ we use $t^{(1)}(l)$ and $t^{(2)}(l)$ to denote them.

(3) Define $I_i$, the vector $(l_2 = 0, l_3 = 0, \ldots, l_i = 1, \ldots, l_m = 0)$. Also define

$$\tilde{\lambda}(l) \overset{def}{=} \frac{\lambda}{\lambda + \sigma(l) \mu}, \quad \tilde{\mu}(l) \overset{def}{=} \frac{\mu}{\lambda + \sigma(l) \mu}.$$

Possible events are:

1. A customer arrives at the system, joining the $i$-th queue. The probability of this event is $\tilde{\lambda}(l)/M$, for every value of $i = 1, \ldots, M$. If $i = 1$, $C$’s state changes to $\langle a, b + 1, l \rangle$. Otherwise, $C$’s state changes to $\langle a, b, l + I_i \rangle$.

2. A customer leaves the system from the local queue. The probability of this event is $\tilde{\mu}(l)$. If $a > 0$ $C$’s state will change to $\langle a - 1, b, l \rangle$. Otherwise $C$ leaves the system.

3. A customer leaves the system from one of the other queues. The probability of this event is $\tilde{\mu}(l)$, for each $i = 2, \ldots, M$ such that $\mathbb{1}(l_i > 0)$. $C$’s state changes to $\langle a, b, l - I_i \rangle$.

This leads to the following set of linear equations

$$d(a, b, l) = t^{(1)}(l) \delta(a, b, l) + \frac{\tilde{\lambda}(l)}{M} \left( d(a, b + 1, l) + \sum_{i=2}^{M} d(a, b, l + I_i) \right)$$

$$+ \tilde{\mu}(l) \left( \sum_{i=2}^{M} \mathbb{1}(l_i > 0)d(a, b, l - I_i) + \mathbb{1}(a > 0)d(a - 1, b, l) \right).$$
(4) The set of equations for $d^{(2)}(a, b, l)$ is

$$
d^{(2)}(a, b, l) = t^{(2)}(l)(\delta(a, b, l)) + \frac{\lambda(l)}{M} \left( d^{(2)}(a, b + 1, l) + \sum_{i=2}^{M} d^{(2)}(a, b, l + I_i) \right)$$

$$+ \hat{\mu}(l) \left( \sum_{i=2}^{M} \mathbb{1}(l_i > 0) d^{(2)}(a, b, l - I_i) + \mathbb{1}(a > 0) d^{(2)}(a - 1, b, l) \right)$$

$$+ 2t^{(1)}(l) \delta(a, b, l) \left( \frac{\lambda(l)}{M} \left( d(a, b + 1, l) + \sum_{i=2}^{M} d(a, b, l + I_i) \right) \right)$$

$$+ \hat{\mu}(l) \left( \sum_{i=2}^{M} \mathbb{1}(l_i > 0) d(a, b, l - I_i) + \mathbb{1}(a > 0) d(a - 1, b, l) \right) \right).$$

(5) Let $k = k_1, \ldots, k_M$ denote a queue occupancy, where $k_i$ is the number of customers at the $i$-th queue, including the customer in service. Let $\hat{k}_i$ denote the vector $k$, whose $i$-th element is omitted, i.e. $\hat{k}_i = (k_1, \ldots, k_{i-1}, k_{i+1}, \ldots, k_M)$. Using this notation we can write

$$\mathbb{E}\{D^2 \mid k\} = \frac{1}{M} \sum_{i=1}^{M} d^{(2)}(k_i, 0, \hat{k}_i).$$

Using this

$$F_{D^2} = \sum_{k_1=0}^{\infty} \sum_{k_2=0}^{\infty} \cdots \sum_{k_M=0}^{\infty} \mathbb{E}\{D^2 \mid k\} P_k,$$

where $P_k$ be the steady state probability that the occupancy upon arrival is $k$, which in this case consists of $M$ independent M/M/1 queues, where each is utilized a fraction $\rho = \lambda/(\mu M)$ of the time and therefore

$$P_k = (1 - \rho)^M \rho^{\sum k}.$$
Fairness in Multiple Server Multiple Queue Systems With Jockeying-On-Idle

The analysis of this system is similar to the one in Section 9.2.1. Steps (1) and (2) are identical, except for the more direct $\sigma(l) = \min(\sum_l + 1, M)$, although \(9.3\) holds as well.

(3) Using the same definitions of $\tilde{\lambda}(l)$ and $\tilde{\mu}(l)$, possible events are:

1. For $\sigma(l) < M$: A customer arrives at the system, joining one of the $M - \sigma(l)$ empty queues. The probability of this event is $\tilde{\lambda}(l)/(M - \sigma(l))$, for every value of $i = 2, \ldots, M$ such that $1(l_i = 0)$. $C$’s state changes to $\langle a, b, 1 + I_i \rangle$.

2. For $\sigma(l) = M$: A customer arrives at the system, joining the $i$-th queue. The probability of this event is $\tilde{\lambda}(l)/M$, for every value of $i = 1, \ldots, M$. If $i = 1$, $C$’s state changes to $\langle a, b + 1, 1 \rangle$. Otherwise, $C$’s state changes to $\langle a, b, 1 + I_i \rangle$.

3. A customer leaves the system from the local queue. The probability of this event is $\tilde{\mu}(l)$. If $a > 0$ $C$’s state will change to $\langle a - 1, b, l \rangle$. Otherwise $C$ leaves the system.

4. A customer leaves the system from one of the other queues. Regularly the probability of this event is $\tilde{\mu}(l)$, for each $i = 2, \ldots, M$ such that $1(l_i > 0)$. However, note that if a customer leaves a server with an empty queue ($1(l_i = 1)$), a customer from another non-empty queue $j$ ($1(l_j > 1)$), or from the local queue ($1(a + b > 1)$), will take its place. If this isn’t the local queue, from $C$’s point of view this will look as if the customer left queue $j$. Define $\sigma^*(a, b, l)$ the number of non-empty queues:

$$\sigma^*(a, b, l) \stackrel{def}{=} 1(a + b > 1) + \sum_{j=2}^{M} 1(l_j > 1).$$

The probability that a specific queue, including the local queue, will benefit from another server being idle is $\tilde{\mu}(l)\sigma^0(a, b, l)$ where

$$\sigma^0(a, b, l) \stackrel{def}{=} \sum_{j=2}^{M} 1(l_j = 1) / \sigma^*(a, b, l).$$
The only possible case in which a queue becomes empty is when \( \sigma^*(a, b, 1) = 0 \). Using these:

If \( l_i > 1 \), \( C \)'s state changes to \( \langle a, b, 1 - I_i \rangle \), with probability \( \tilde{\mu}(1)(1 + \sigma^0(a, b, 1)) \). If \( l_i = 1, \sigma^*(a, b, 1) = 0 \), \( C \)'s state changes to \( \langle 0, 0, 1 - I_i \rangle \), with probability \( \tilde{\mu}(1) \).

In all the following cases, a customer from the local queue moves to an empty queue, with probability \( \tilde{\mu}(1)\sigma^0(a, b, 1) \).

If \( a = 0, b > 0 \), \( C \)'s state changes to \( \langle a, b - 1, 1 \rangle \). If \( a \geq 1 \), \( C \)'s state changes according to the exact jockeying mode:

a) In the head-jockeying-on-idle system: If \( a > 1 \) then \( C \)'s state will change to \( \langle a - 1, b, 1 \rangle \). If \( a = 1 \) then \( C \) will change queues, and thus \( C \)'s state will change to \( \langle 0, 0, 1 + bI_i \rangle \).

b) In the tail-jockeying-on-idle system: If \( b > 0 \) then \( C \)'s state will change to \( \langle a, b - 1, 1 \rangle \). If \( b = 0 \) then \( C \) will change queues, and thus \( C \)'s state will change to \( \langle 0, 0, 1 + (a - 1)I_i \rangle \).

After some minor substitutions this leads to the following sets of equations: For head-jockeying-on-idle

\[
d(a, b, 1) = l^{(1)}(a, b, 1)\delta(a, b, 1) + \lambda(I) \left\{ \sum_{i=2}^{M} 1(l_i = 0) \frac{d(a, b + 1, l_i)}{M} + \sum_{i=2}^{M} 1(l_i = 1) \frac{d(a, b + 1, l_i)}{M} \sigma(I) < M \\
+ \tilde{\mu}(1) \left( 1(a = 1)\langle a - 1, b, 1 \rangle + 1(\sigma^*(a, b, 1) = 0) \sum_{i=2}^{M} 1(l_i = 1) d(0, 0, 1 - I_i) \right) \\
+ \tilde{\mu}(1)(1 + \sigma^0(a, b, 1)) \left( \sum_{i=2}^{M} 1(l_i > 1) d(a, b, 1 - I_i) + 1(a > 1)\langle a - 1, b, 1 \rangle \right) \\
+ \tilde{\mu}(1)\sigma^0(a, b, 1) \right\} \begin{cases} \\
(9.5)
\end{cases} \\
\begin{array}{ll}
d(a, b - 1, 1) & a = 0, b > 0 \\
d(0, 0, 1 + bI_i) & a = 1
\end{array}
\]
and for tail-jockeying-on-idle

\[
d(a, b, 1) = t^{(1)}(a, b, 1) \delta(a, b, 1) + \tilde{\lambda}(l) \begin{cases}
\sum_{i=2}^{M} \mathbb{1}(l_i = 0) \frac{d(a, b, 1 + l_i)}{M} & \sigma(1) < M \\
\frac{d(a, b, 1)}{M} + \sum_{i=2}^{M} \frac{d(a, b, 1 + l_i)}{M} & \sigma(1) = M
\end{cases}
\]

\[
+ \tilde{\mu}(l) \left( \mathbb{1}(a > 0)(a-1, b, 1) + \mathbb{1}(\sigma^*(a, b, 1) = 0) \sum_{i=2}^{M} \mathbb{1}(l_i = 1)d(0, 0, 1 - l_i) \right)
\]

\[
\quad + \tilde{\mu}(l)(1 + \sigma^0(a, b, 1)) \sum_{i=2}^{M} \mathbb{1}(l_i > 1)d(a, b, 1 - l_i)
\]

\[
\quad + \tilde{\mu}(l)\sigma^0(a, b, 1) \begin{cases}
\frac{d(a, b - 1, 1)}{M} & b > 0 \\
\frac{d(0, 0, 1 + (a - 1)l_i)}{M} & a \geq 1, b = 0
\end{cases}
\]

(4) Writing the sets of equations for \(d^{(2)}(a, b, 1)\) is a straightforward, though lengthy process. We omit them for brevity.

(5) Again, let \(k\) denote a queue occupancy vector, and \(\hat{k}_i\) denote the vector \(k\), whose \(i\)-th element is omitted. Using this notation we can write

\[
\mathbb{E}\{D^2 \mid k\} = \begin{cases}
\frac{1}{M} \sum_{i=1}^{M} d^{(2)}(k_i, 0, \hat{k}_i) & \Sigma_{k} \geq M \\
\frac{1}{M - \Sigma_{k}} \sum_{i=1}^{M} \mathbb{1}(k_i = 0)d^{(2)}(0, 0, \hat{k}_i) & \Sigma_{k} < M
\end{cases}
\]

Note that some occupancy vectors are impossible, namely every \(k_i = 0, k_j > 1, i, j = 1, \ldots, M\). For completeness in these cases we define \(\mathbb{E}\{D^2 \mid k\} \overset{df}{=} 0\).

Using this, (9.4) holds, and \(P_k\), the steady state probability that the occupancy is \(k\), can be numerically calculated using the system balance equations.

**Numerical Results for the M/M/2 Model**

Figure 9.6 depicts the results from evaluating the sets of equations provided in Section 9.2.1 for \(M = 2\). The results are also supported by simulation. For all plotted points \(\mu = 1\) while \(\lambda\) changes according to \(\rho\). The figure demonstrates the following properties:
1. For all systems, the unfairness is monotone increasing with the system utilization factor $\rho$.

2. Similarly to the situation in the M/D/2 model, the single queue system is more fair than all the multiple queue systems, for every system load. The difference in unfairness between this system and the best multiple server system, in our analysis the head-jockeying-on-idle system, is around 10%. Recall that as long as jockeying is allowed only in a manner that is nondiscriminatory with respect to the service requirement, as is the case in the systems analyzed, the single queue system is at least as efficient as the multiple queue systems.

3. Similarly to the situation in the M/D/2 model, among the multiple queue systems, the jockeying-on-idle systems are more fair than the idling system, for every system load. The difference between these two types of systems is quite substantial. At lower loads, the ratio between them can be more than 1:10.

4. Similarly to the situation in the M/D/2 model, among the jockeying-on-idle systems,
head-jockeying is more fair than tail-jockeying for every system load. The difference between them is around 10%. Recall that these two systems are equally efficient.

To summarize, all three properties discussed in Section 9.1.2 for the G/D/m model (and demonstrated to be true for the M/D/2 model in Section 9.1.4) are demonstrated to be true for the M/M/2 model.

9.2.2 The Effect of Queue Joining Policy on Fairness

In this section we analyze the queue joining policies discussed in Section 9.1.3 for M/M/m type systems. The RAND policy was analyzed in Section 9.2.1 and thus we only analyze the SQ and RR policies, with head-jockeying-on-idle.

Fairness in the Join The Shortest Queue Policy

The analysis of the SQ policy is very similar to that of the RAND policy given in Section 9.2.1 as the difference between the two policies is only noticeable on new customer arrivals. Steps (1) and (2) are identical. In step (3) the only difference is that the first and second events, dealing with a customer arriving into the system, are replaced with the following

1. A customer arrives at the system, joining one of the shortest queues. Let

\[ l_{\text{min}}(a, b, 1) \overset{\text{def}}{=} \min(a + b + 1, \min 1) \]

be the minimum queue length, and let

\[ N_{\text{min}}(a, b, 1) \overset{\text{def}}{=} \mathbb{1}(a + b + 1 = l_{\text{min}}(a, b, 1)) + \sum_{i=2}^{M} \mathbb{1}(l_i = l_{\text{min}}(a, b, 1)) \]

be the number of queues with minimal length. The probability of this event is \( \tilde{\lambda}(1)/N_{\text{min}}(a, b, 1) \), for every such queue. If \( i = 1 \), \( C \)'s state changes to \( (a, b + 1, 1) \). Otherwise, \( C \)'s state changes to \( (a, b, 1 + I_i) \).
This of course changes the second additive term of $\mathcal{L}$ to

$$\frac{\hat{\lambda}(1)}{N_{\min}(a, b, l)} \left( d(a, b + 1, l) + \sum_{i=2}^{M} \mathbb{1}(l_i = l_{\min}(a, b, 1))d(a, b, 1 + l_i) \right),$$

and similarly in step (4).

In step (5) if we use $N_{\min}(k)$ to denote the number of minimal queues in the system occupancy vector $k$

$$N_{\min}(k) \overset{def}{=} \sum_{i=1}^{M} \mathbb{1}(k_i = \min k)$$

then (9.6) changes to

$$\mathbb{E}\{D^2 \mid k\} = \begin{cases} \frac{1}{N_{\min}(k)} \sum_{i=1}^{M} \mathbb{1}(k_i = \min k)d^{(2)}(k_i, 0, \hat{k}_i) & \Sigma_k \geq M \\ \frac{1}{M - \Sigma_k} \sum_{i=1}^{M} \mathbb{1}(k_i = 0)d^{(2)}(0, 0, \hat{k}_i) & \Sigma_k < M \end{cases}$$

and the same occupancy vectors are impossible and defined $\mathbb{E}\{D^2 \mid k\} \overset{def}{=} 0$.

Note that $P_k$, the steady state probability that the occupancy upon arrival is $k$, is numerically calculated using different system balance equations.

**Fairness in the Round-Robin Queue Joining Policy**

Again, we use Section 9.2.1 as the base of our analysis.

We use the $\oplus$ symbol to denote increment modulo $M$ i.e. $x \oplus \overset{def}{=} (x + 1) \mod M$.

In step (1), let $r$ denote the index of the queue that the next arriving customer will join, $r = 1, \ldots, M$. The state $\langle a, b, l, r \rangle$ now captures all that is needed to predict the future of $C$. Step (2) is identical. Note that though $r$ is part of the state, it does not effect anything but the joining process. In step (3), a customer will always join queue $r$ first, and $r$ will be incremented. However, note that if server $r$ is busy, and another server is idle, the arriving customer will move immediately to one of the idle servers. Thus the first and second events, dealing with a customer arriving into the system, are replaced with
3. For \( \sigma(l) < M \): A customer arrives at the system. If \( r = 1 \) or \( l_r > 0 \) the customer joins one of the \( M - \sigma(l) \) empty queues. The probability of this event is \( \tilde{\lambda}(l)/(M - \sigma(l)) \), for every value of \( i = 2, \ldots, M \) such that \( \mathbb{1}(l_i = 0) \), and \( C \)'s state changes to \( \langle a, b, l + I_i, r_\oplus \rangle \). If \( r \neq 1 \) and \( l_r = 0 \) the customer joins queue \( r \). The probability of this event is \( \tilde{\lambda}(l) \) and \( C \)'s state changes to \( \langle a, b, l_+ I_r, r_\oplus \rangle \).

4. For \( \sigma(l) = M \): A customer arrives at the system, joining the \( r \)-th queue. The probability of this event is \( \tilde{\lambda}(l) \). If \( r = 1 \) \( C \)'s state changes to \( \langle a, b + 1, l, r_\oplus \rangle \). Otherwise, \( C \)'s state changes to \( \langle a, b, l_+ I_r, r_\oplus \rangle \).

For all the rest of the events, we just need to add \( r \) to the state.

This changes the second additive term of (9.5) to

\[
\tilde{\lambda}(l) \begin{cases} 
\sum_{i=2}^{M} \mathbb{1}(l_i = 0) \frac{d(a, b, l_+ I_i, r_\oplus)}{M - \sigma(l)} & \sigma(l) < M, (r = 1 \text{ or } l_r \neq 0) \\
\frac{d(a, b, l_+ I_r, r_\oplus)}{M - \sigma(l)} & \sigma(l) < M, r \neq 1, l_r = 0 \text{ or } \sigma(l) = M, r \neq 1 \\
d(a, b + 1, l, r_\oplus) & \sigma(l) = M, r = 1 
\end{cases}
\]

and \( r \) is added to every \( d(\cdot) \). Similar changes are made in step (4).

(5) Let \( k^* = k_1, \ldots, k_M, r \) denote the queue occupancy, appended by the queue joining slot, i.e. \( k_i \) is the number of customers at the \( i \)-th queue, including the customer in service, and \( r = 1, \ldots, M \) is the queue the next customer will join. Let \( \hat{k}_i^* \) denote the vector \( k^* \), whose \( i \)-th element is omitted, and without the last element \( r \), i.e. \( \hat{k}_i^* = (k_1, \ldots, k_{i-1}, k_{i+1}, \ldots, k_M) \). Using this notation we can write

\[
\mathbb{E}\{|D^2| \mid k^*\} = \begin{cases} 
\frac{d(2)(k_r, 0, \hat{k}_r^*, r_\oplus)}{M - \Sigma_k} & \Sigma_k \geq M \text{ or } k_r = 0 \\
\frac{1}{M - \Sigma_k} \sum_{i=1}^{M} \mathbb{1}(k_i = 0) d(2)(0, 0, \hat{k}_i^*, r_\oplus) & \Sigma_k < M, k_r > 0 
\end{cases}
\]

and

\[
F_{D^2} = \sum_{k_1=0}^{\infty} \sum_{k_2=0}^{\infty} \cdots \sum_{k_M=0}^{\infty} \sum_{r=1}^{M} \mathbb{E}\{|D^2| \mid k^*\} P_{k^*},
\]

where \( P_{k^*} \), the steady state probability that \( k^* \) is encountered upon arrival, can be numerically calculated using the appropriate system balance equations.
Numerical Results

Figure 9.7 depicts the unfairness of the three joining policies analyzed - RAND, RR, and SQ, for the M/M/2 model, under head-jockeying-on-idle, as a function of the system utilization factor $\rho$. They were computed using the equations derived above. For comparison we also plot the results for the single queue system and the results for the tail-jockeying-on-idle system with random queue joining. The results are also supported by simulation. The figure demonstrates the following properties:

1. All three queue joining policies are less fair than the single queue system, and more fair than the tail-jockeying system.

2. The unfairness of the system in the three queue joining policies is very similar. In fact, the unfairness is within a 1% interval, which might be caused by inaccuracies in our computation. This means that according to our analysis, in the case of head-jockeying-on-idle M/M/2 systems, the queue joining policy has little effect on the fairness of the system.

Figure 9.7: The Effect of the Queue Joining Policy on the System Fairness
Remark 9.3. The latter insensitivity to the queue joining policy can be explained by the fact that the jockeying alleviates potential discrimination caused by unfair queue joining policies.

9.3 Fairness of G/G/m Type Systems

In this section we consider G/G/m type systems, i.e. systems where the customers arrive according to a general arrival process, the service requirements are generally distributed and the system has multiple servers. We first show, via some counter-examples, that results similar to those derived in the analysis of the G/D/m system (Section 9.1.2) and demonstrated to be true for M/M/m type systems (Section 9.2.1) do not necessarily hold for an arbitrary G/G/m system. We provide sufficient conditions under which some of these properties hold for the G/G1/m mode, namely where the service requirements are generally distributed and independent of each other and of the arrivals.

In the following example we show that the single queue system is not necessarily more fair than a multiple queue system with no jockeying. Consider the following arrival and service pattern, involving 4 customers, $C_1, C_2, C_3$ and $C_4$: $\{(a_i, s_i)\}_{i=1,2,3,4} = \{(0, s + 1), (0, 2\epsilon), (\epsilon, s + 1), (1, \epsilon)\}$, where $\epsilon \to 0$ and $s \gg \epsilon$ is some finite service requirement. Let $M = 2$ and compare the case where customers are served in a single queue to one where customers are served in two queues. If customers are served in a single queue, then $C_4$, which has a very short service requirement, has to wait $s$ units of time to be served. As $C_2$ and $C_4$ are served immediately upon arrival and their service requirements are very small, we can disregard these intervals of service in our numerical calculation. This leads to a relatively high unfairness, namely $2 \times (s \times 1/3)^2 + (s \times (-2/3))^2 = 2s^2/3$. Consider now the two queues ($q_a$ and $q_b$) case where jockeying is not allowed. Assume $C_1$ joins $q_a$ and $C_2$ joins $q_b$. $C_3$ joins the system at epoch $\epsilon$, finding both serves busy, and joins $q_a$, using round-robin joining policy, randomly, or even using shortest queue joining. $C_4$ can then be served upon arrival. As $C_2$ and $C_4$ are served immediately upon arrival and their service requirements are very small, we can disregard these intervals of service in our
numerical calculation. As $C_3$ only needs to wait for one customer, namely $C_1$, compared to two customers for which $C_4$ needs to wait for in the previous case, the unfairness is only $(s \times 1/2)^2 + (s \times (-1/2))^2 = s^2/2 < 2s^2/3$. Note that even if queue joining if random, and one is interested in the expected unfairness, where expectation is taken over the queue joining random variable, the multiple-queue system is still more fair, with an expected value of $(2s^2/3 + s^2/2)/2 = 7s^2/12 < 2s^2/3$.

This same arrival and service pattern can also be used to show that the standard (idling) system is not always less fair than the jockeying-on-idle systems. Again, if $C_3$ joins $q_a$ and does not jockey, the unfairness is $s^2/2$ while in a jockeying system it is always $2s^2/3$. The same argument about expected unfairness also applies.

The following arrival and service pattern shows that the head-jockeying-on-idle system is not always more fair than the tail-jockeying on-idle one. Consider $\{(a_i, s_i)\}_{i=1,2,3,4,5} = \{(0,1 + 2\epsilon), (0,1 + 3\epsilon), (1,s), (1,s), (1 + \epsilon, \epsilon)\}$, where again $\epsilon \to 0$, and assume that $C_1$ joined $q_a$, $C_2$ joined $q_b$, and $C_3, C_4$ and $C_5$ joined $q_a$. At epoch $1 + 2\epsilon$ jockeying is allowed since $q_a$ is empty. If head jockeying is done then either $C_3$ or $C_4$ will be served, followed by the other at $1 + 3\epsilon$, when the server for $q_b$ becomes free. Thus the system will have high unfairness, as $C_5$ waits in the system unserved. However, if tail jockeying is allowed, then $C_5$ will be served first, leading to almost zero unfairness. Again, this is true not only for this specific queue joining sequence, but also in expectation over the queue joining.

The same arrival and service pattern can also be used to show that joining the shortest queue, or using round robin, are less fair than joining a queue randomly. Note that if customers join the shortest queue or use round robin, $C_5$ will have to wait for either $C_3$ or $C_4$, creating large unfairness. The only possible scenarios in which $C_5$ does not wait are those where queue joining is done in a random manner.

We now move on to consider the $G/GI/m$ model. Define the residual warranted service of customer $i$ at epoch $\tau$, $R_i(\tau)$, as the integral of warranted service rate from epoch $\tau$ onwards, i.e. $R_i(\tau) = \int_{\tau}^{d_i} \omega(t)/N(t)dt, a_i < \tau < d_i$.

**Definition 9.1 (Forward Dominance).** Let $g_i$ denote the epoch in which $C_i$ enters service.
Let $C_j, C_k$ be a pair of customers with i.i.d. service requirements $S_j, S_k$.

A system is said to be Forward Dominant if for every such pair $C_j, C_k$, every arrival pattern, every set of service requirements ${s_i}, i = 1, 2, \ldots, L, i \neq j, k$ and every service policy where $g_j < g_k$, the expected residual warranted service obeys $\mathbb{E}\{R_j(g_j)\} < \mathbb{E}\{R_k(g_j)\}$, where expectations are taken on the joint distribution of $S_j, S_k$.

In a forward dominant system it is, in essence, always better to be served first, at least by expectation. For example, it is easy to see that if all customers have identical service requirements, the system is Forward Dominant, as is the case for single server systems. In fact in these cases it is true for every pair of service requirements $s, s'$ and not only in expectation (for comparison, Theorem 5.1 is proven by expectation, while in Appendix A a stronger theorem is proven for every pair of service requirements). However, in the general multiple server case it is easy to see that this is not always the case, as demonstrated by the examples above.

Note that since we consider $\phi \in \Phi$, then the condition $g_j < g_k$ does not depend on the actual values of $S_j$ and $S_k$, and thus taking the expectation over the full joint distribution of $S_j$ and $S_k$ is well defined.

**Theorem 9.5** (Preference of Seniority for G/GI/m). Consider a Forward Dominant system. Let $a_1, a_2, \ldots, a_L$ be an arbitrary (deterministic) arrival pattern where $a_j < a_k$, $1 \leq j, k \leq L$. Let $\{s_i\}, i = 1, 2, \ldots, L, i \neq j, k$ be an arbitrary (deterministic) set of service requirements. Let $S_j$ and $S_k$, the service requirements of $C_j$ and $C_k$, respectively, be i.i.d. random variables, with realizations $s_j$ and $s_k$ respectively. Let $\phi \in \Phi$ be a service policy that serves $C_k$ ahead of $C_j$. Then there exists a service policy $\phi' \in \Phi$ such that $\phi'$ has lower expected overall unfairness than $\phi$, where expectation is taken over the joint distribution of $S_j$ and $S_k$, and $\phi'$ serves $C_j$ ahead of $C_k$.

This theorem is very similar to the one proven in Appendix A except that it is stated for Forward Dominant systems rather than a single server system. The proofs also have much in common.
Proof. First we construct \( \phi' \) from \( \phi \): Until \( g_k \) \( \phi' \) is identical to \( \phi \). At \( g_k \) it chooses to serve \( C_j \) (whose service requirement is \( s_j \)). From \( g_k + s_j \) and on \( \phi' \) imitates what \( \phi \) would do in the case that \( C_k = s_j \), with the exception that when \( \phi \) serves \( C_j \) \( \phi' \) serves \( C_k \). Since \( \phi \in \Phi \) and \( \phi' \) imitates it, \( \phi' \in \Phi \).

Next we compare the unfairness of \( \phi \) and \( \phi' \). We do this for any sample path, and then take expectation over the service requirements of the two customers. For \( s_j = s_k \), \( \phi' \) serves all customers in the same order as \( \phi \), except that it interchanges \( C_j \) and \( C_k \). From Lemma \( \ref{lem:order} \) it follows that the unfairness of \( \phi' \) is lower in this sample path.

For \( s_j \neq s_k \) we need to track two sample paths for each policy: For every pair of service requirements \( \tau_1, \tau_2 \) (w.l.o.g. assume \( \tau_1 < \tau_2 \)) we consider one path where \( S_j = \tau_1 \) and \( S_k = \tau_2 \) (sample path (a)) and one path where \( S_j = \tau_2 \) and \( S_k = \tau_1 \) (sample path (b)) and apply \( \phi \) on both. Let \( F(a), F(b) \) denote the unfairness for \( \phi \) in the respective sample path.

We then construct \( \phi' \) and apply it on the same sample paths. Let \( F(c), F(d) \) denote the unfairness for \( \phi' \) on the sample path (b), (a) respectively.

Note that the probability of \( S_j = \tau_1, S_k = \tau_2 \) is equal to that of \( S_j = \tau_2, S_k = \tau_1 \), since \( S_j \) and \( S_k \) are i.i.d. We can therefore show that the expected overall unfairness under \( \phi \) is larger than that under \( \phi' \), by showing that \( F(a) + F(b) - F(c) - F(d) > 0 \).

Let \( \hat{F}(a) \) denote the total unfairness under (a) due to \( C_j \) and \( C_k \), and \( \tilde{F}(a) \) denote the total unfairness under (a) due to all customers other than \( C_j \) and \( C_k \). Then \( F(a) = \hat{F}(a) + \tilde{F}(a) \). We define similar notations for (b), (c) and (d). Note that due to the way we constructed \( \phi' \), for any customer other than \( C_j \) and \( C_k \), (a) and (c) are identical, and so are (b) and (d). Therefore \( F(a) = \hat{F}(c) \) and \( F(b) = \hat{F}(d) \). Thus \( F(a) + F(b) - F(c) - F(d) = \hat{F}(a) + \hat{F}(b) - \hat{F}(c) - \hat{F}(d) \).

Let \( R(1), R(2) \) denote the warranted service in the intervals \((a_j, a_k)\) and \((a_k, g_k)\) respectively. These are the same for both paths for both policies. Let \( R_j^\phi(g_k), R_k^\phi(g_k) \) denote the residual warranted service from epoch \( g_k \), for customer \( C_j, C_k \) respectively, under \( \phi \) for
scenario (a). Denote similar variables for for scenario (b).

\[
\hat{F}(a) + \hat{F}(b) - \hat{F}(c) - \hat{F}(d) = \\
\frac{1}{L} \left[ (\tau_1 - (R_{(1)} + R_{(2)} + R_{j}^{a}(g_k)))^2 + (\tau_2 - (R_{(2)} + R_{k}^{a}(g_k)))^2 \\
+ (\tau_2 - (R_{(1)} + R_{(2)} + R_{j}^{b}(g_k)))^2 + (\tau_1 - (R_{(2)} + R_{k}^{b}(g_k)))^2 \\
- (\tau_2 - (R_{(1)} + R_{(2)} + R_{k}^{a}(g_k)))^2 - (\tau_1 - (R_{(2)} + R_{j}^{a}(g_k)))^2 \\
- (\tau_1 - (R_{(1)} + R_{(2)} + R_{j}^{b}(g_k)))^2 - (\tau_2 - (R_{(2)} + R_{j}^{b}(g_k)))^2 \right] \\
= \frac{2}{L} R_{(1)}(R_{j}^{a}(g_k) + R_{j}^{b}(g_k) - R_{k}^{a}(g_k) - R_{k}^{b}(g_k)).
\]

We now take expectation of this expression over the joint distribution of \(S_j\) and \(S_k\). Note that \(R_{(1)}\) is constant in this context, thus the sign of the expected value of this expression is determined by the sign of the expected value of \(R_{j}^{a}(g_k) + R_{j}^{b}(g_k) - R_{k}^{a}(g_k) - R_{k}^{b}(g_k)\), which is positive due to the system being forward dominant.

**Corollary 9.1.** If a G/GI/m system is forward dominant, then the properties in Theorem 9.1, Theorem 9.2 and Theorem 9.3 hold, with the modification that they hold in expectation over the service requirement distributions.

The corollary is immediate from Theorem 9.5.

**Remark 9.4 (Necessary Conditions).** While these conditions are sufficient, there is no reason to assume they are necessary. Finding necessary conditions remains and open problem and is left for future research.

**Conjecture 9.2.** The above properties hold for the M/GI/m model.

We base our conjecture on simulations we conducted for several service requirement distributions with Poisson arrival times.
One example is a case where the variability of service requirements is very large. This is achieved by a bi-valued service requirement whose values are \( s = 0.1 \) with probability \( p \) and \( s' = 10 \) with probability \( 1 - p \). The value of \( p \) is selected to be \( p = \frac{90}{99} = 0.9009 \) so as to have mean service requirement of 1, identical to the mean service requirement used in previous numerical example (Section 9.1.4, Section 9.2.1). The variance of service requirement is \( p s^2 + (1 - p) s'^2 = 9.1 \), in comparison to a variance of zero for M/D/2 and a variance of \( 1/\mu^2 = 1 \) for M/M/2.

Figure 9.8 depicts \( F_{D^2} \) as a function of \( \rho \) for the four systems described in Section 9.1.2. In this figure each point is the result of simulating the passage of at least \( 10^6 \) customers through the system. The figure demonstrates that the properties hold.

**Figure 9.8:** Unfairness of Four Queue Strategies for High Variability Service Requirements

### 9.4 Summary

We investigated several fairness considerations in operating multiple server and multiple queue systems, under different settings.

For the G/D/m and M/M/m systems we showed the following properties:
• Combining queues is more fair than splitting them. This agrees with customer intuition.

• Allowing jockeying when a server is idle improves the fairness.

• Jockeying from the head of the queue is the most fair.

For the issue of queue joining policy we showed that for the G/D/m case joining the shortest queue or doing round robin is the more fair than joining a queue in random. For the M/M/m case the queue joining policy has little effect on fairness.

For G/G/m systems we showed that these properties don’t necessarily hold. We provided sufficient conditions under which the properties hold for a GI/GI/m system and conjectured that they hold for M/GI/m systems. We demonstrated this for a bi-valued service distribution with high variability.
In this chapter we turn to investigate the fairness considerations involved in classification and prioritization mechanisms. Most of the material in this chapter appears in Raz et al. \[116\] [121]

Customer classification and prioritization are common mechanisms used in a large variety of daily queueing situations. Examples of daily applications include classifications of passengers to alien and non-alien in airport customs queues, gender classification in public facilities bath-room queues, and prioritization of jobs in computer systems. Classification of jobs is done based on many characteristics. For example in computerized call centers and Internet Web servers customers might be classified and prioritized by IP address range, transaction type, etc.

More specifically, the classification of customers based on their service requirements and the prioritization of such classes is a very common mechanism used for providing preferential service and for controlling the service in a queueing system. One common example is the priority mechanism used in computer systems where short jobs receive higher priority over long jobs. Another very common application is the queueing structure in supermarkets and other stores where short jobs, i.e. customers with a few items in hand, receive preferential service through special servers (cashiers) dedicated to them.

Two common mechanisms used in queueing systems to grant preferences to different classes are: a) Prioritization, in which the classes are ordered and priority, either preemptive or non-preemptive, is given to customers belonging to higher priority classes over those belonging to lower priority classes, and b) Resource dedication, in which each class
has a server, or a set of servers, and a queue dedicated to it. Our focus will be on studying these two mechanisms.

There is a large body of literature on such priority schemes (e.g. Avi-Itzhak and Naor [8], Avi-Itzhak [5, 6], Kella and Yechiali [78, 79], Davis [40], Jaiswal [74], Takagi [145]) where the focus in evaluating system performance is on the expected waiting time, or the mean waiting cost, under linear cost parameters. Optimization of the system with non-preemptive priorities, based on this performance objective, shows (e.g. Cox and Smith [36], pp. 84-85) that the optimal scheduling policy is to provide a higher priority to jobs with smaller mean service requirements, or when costs are involved, apply the $\mu C$ rule. Such priority may, however, result with long jobs waiting for the completion of many short jobs who arrive behind them, and thus, possibly, to unfair treatment by the system. Thus, system operation that accounts both for efficiency and fairness, might have to resort to a different scheduling.

In this chapter our purpose is to study the general fairness aspects of such prioritization and classification mechanisms in a quantitative manner, using the RAQFM measure. We limit our discussion to systems where job classification is based only on service characteristics. Systems where classes have other attributes, such as different economic values, can be treated by weighting mechanisms, which are out of the scope of this study, and are to be dealt with in future work.

We first Section 10.1 introduce class discrimination, which is a variant of RAQFM that accounts for the expected discrimination experienced by the customers of a certain class. Using this measure, in Section 10.2 we study class prioritization and in Section 10.3 we study resource dedication.

A less related issue, that of combining servers is studied in Section 10.4. While this subject could have been studied in the context of a single class system in Chapter 9, it was in effect studied first in the context of multiple classes, and we therefore bring it here.
10.1 Class Discrimination

Recall that customers are indexed \( C_1, C_2, \ldots \), and arrive according to this order. Each customer belongs to one of \( U \) classes, indexed \( 1, 2, \ldots, U \). The arrival rate of class \( u \) customers is denoted \( \lambda_u \) where \( \sum_{u=1}^{U} \lambda_u = \lambda \). An order of priorities is assigned to the classes, where lower class index means higher priority.

In our discussion, when we mention the Preemptive Priority class of scheduling policies, the order of service within each class of customers is FCFS, and preempted customers return to the head of the queue of their class. For discussion of this, and other variants, see Takagi [145], Sec. 3.4.

For a class \( u \) the discrimination \( D \) experienced by an arbitrary customer \( C \), when the system is in steady state, is a random variable denoted \( D_{(u)} \overset{def}{=} D \mid C \in u \). Our interest will be in the expected discrimination experienced by \( u \)'s customers, namely \( \mathbb{E}\{D_{(u)}\} \), termed Class Discrimination.

A second useful notion is that of class discrimination rate. The instantaneous discrimination rate of class \( u \) at time \( t \) is the sum of discriminations over all \( u \)'s customers present in the system at time \( t \). Let \( \hat{D}_{(u)}(t) \overset{def}{=} \sum_{l \in u} \delta_l(t) \) denote this variable and let \( \hat{D}_{(u)} \overset{def}{=} \lim_{t \to \infty} \hat{D}_{(u)}(t) \) be a random variable denoting the instantaneous discrimination rate of class \( u \) when the system is in steady state. Taking expectation of this variable we get the class discrimination rate \( \mathbb{E}\{\hat{D}_{(u)}\} \).

We observe that the relationship between the variables \( D_{(u)} \) and \( \hat{D}_{(u)} \) is analogous to the equilibrium relationship between the variables customer delay, i.e. the delay experienced by an arbitrary customer and number of customers in the system, i.e. the number of customers present at an arbitrary moment, in a stationary queueing system. While the former is more appropriate to describe the customer’s perception, the latter might be more appropriate to describe the system’s state. We therefore choose to focus on the former.

However, we recall from Little’s Theorem that \( N = \lambda T \), where \( T \), \( \lambda \) and \( N \) denote expected customer delay, arrival rate and expected number in the system, respectively.
(Little [92]). Using an analogy to Little’s Theorem, and the same machinery used in its proof, one can now derive a “discrimination version” of Little’s Theorem, namely:

\[
\mathbb{E}\{\hat{D}(u)\} = \lambda_u \mathbb{E}\{D(u)\}.
\] (10.1)

As a side note, applying this rule to the whole population, rather than to a certain class, results in \(\mathbb{E}\{D\} = 0\).

**Theorem 10.1.** The class discrimination of class \(u\) is bounded from above by the overall system unfairness as follows:

\[
\frac{\lambda_u}{\lambda} |\mathbb{E}\{D(u)\}| \leq \sqrt{\mathbb{E}\{D^2\}}.
\]

**Proof.** Since \(\mathbb{E}\{D(u)^2\} - (\mathbb{E}\{D(u)\})^2 \geq 0\) we have

\[
\frac{\lambda_u}{\lambda} |\mathbb{E}\{D(u)\}| \leq \frac{\lambda_u}{\lambda} \sqrt{\mathbb{E}\{D(u)^2\}}.
\]

But

\[
\frac{\lambda_u}{\lambda} \sqrt{\mathbb{E}\{D(u)^2\}} \leq \sqrt{\frac{\lambda_u}{\lambda} \mathbb{E}\{D(u)^2\}} \leq \sqrt{\sum_{i=1}^{U} \frac{\lambda_i}{\lambda} \mathbb{E}\{D(i)^2\}} = \sqrt{\mathbb{E}\{D^2\}}.
\]

\[\square\]

**Corollary 10.1.** Consider an arbitrary system with \(U\) customer classes. If the system unfairness obeys \(\mathbb{E}\{D^2\} = 0\) then for every class \(1 \leq u \leq U\) the class discrimination obeys \(\mathbb{E}\{D(u)\} = 0\).

The proof is immediate from Theorem 10.1.

**Theorem 10.2.** Consider a system with \(U\) classes. Assume that the class discrimination of each class \(u\) obeys \(\mathbb{E}\{D(u)\} = 0\). Then the system unfairness, \(\mathbb{E}\{D^2\}\) can still be positive.
Chapter 10. Fairness of Prioritization Systems

Proof. (by example). Consider a system with two classes, A and B. Assume that the service requirement is one unit for all customers and the arrival process is in pairs, one customer of each type. Assume that the inter-arrival time is given by \( x > 2 \) and that for half the pairs the server serves A first and for the other half it serves B first. One can easily observe that half of the customers experience positive discrimination of 0.5 and half experience negative discrimination of -0.5. Thus \( \mathbb{E}\{D^2\} = 0.25 \). Nonetheless the expected class discrimination is zero for both classes.

The implications of these results are: 1) If one maintains very low system unfairness it guarantees that the class discrimination of large population classes (classes with relatively high arrival rates) will be very small, while the discrimination of a lightly populated class can still be very high. 2) Maintaining low class discrimination to all classes does not guarantee a fair system, since there could be unfairness in treatment of customers within a class.

10.2 Class Prioritization

In this section we study the effect class prioritization has on the system unfairness. We first show that generally speaking, prioritizing short jobs is justified, since otherwise these jobs are negatively discriminated. We then show the effectiveness of class prioritization, and that while prioritization can guarantee positive discrimination to the class with highest priority and negative discrimination to the class with lowest priority, it cannot guarantee monotonicity in discrimination. Lastly we provide the analysis of unfairness and class discrimination in single server systems with Markovian distributions.

10.2.1 Prioritizing Short Jobs is Justified

Definition 10.1 (Stochastic Dominance Between Random Variables). Consider non negative random variables \( X_1, X_2 \) whose distributions are \( F_{X_1}(t) = \mathbb{P}\{X_1 \leq t\}, F_{X_2}(t) = \mathbb{P}\{X_2 \leq t\} \). We say that \( X_1 \) stochastically dominates \( X_2 \), denoted \( X_1 \triangleright X_2 \), if \( F_{X_1}(t) \leq F_{X_2}(t) \) \( \forall t \geq 0 \).
Theorem 10.3. Let $C_l$ be a customer with service requirement $s_l$. Consider a $G/G/M$ system with service policy $\phi \in \Phi^*$. Let $D_l^{(s_l)}$ be a random variable denoting the discrimination of $C_l$, when it arrives at the system in steady state. Then $D_l^{(s_l)}$ is monotone non-decreasing in $s_l$, namely if $s'_l > s_l$ then $D_l^{(s'_l)} \succ D_l^{(s_l)}$.

Proof. Consider service requirements $s'_l > s_l$ and observe a customer $C_l$. Under any non-preemptive service policy, $C_l$ waits until epoch $q_l$, when it enters service, and stays in service until its departure. \((4.3)\) can thus be written as

$$D_l = \int_{q_l}^{q_l + s'_l} \delta_l(t) dt + \int_{q_l}^{d_l} \delta_l(t) dt. \quad (10.2)$$

The first term in this sum is independent of the service requirement. In the second term $d_l - q_l = s_l$.

To prove the monotonicity we consider a specific sample path $\pi$ and compare the values of $D_l^{(s_l)}$ and $D_l^{(s'_l)}$ for this path, denoted by $D_{l,\pi}^{(s_l)}$ and $D_{l,\pi}^{(s'_l)}$. From \((10.2)\) we have

$$D_{l,\pi}^{(s'_l)} - D_{l,\pi}^{(s_l)} = \int_{q_l + s}^{q_l + s'} \delta_l(t) dt \geq 0, \quad (10.3)$$

where the inequality is due to $\delta_l(t) \geq 0$, which follows from \((4.4)\), when the customer is in service. Since \((10.3)\) holds for every sample path $\pi$, the proof follows.

This simple theorem, stated in terms of deterministic service requirements, can also be stated using stochastic service requirements, i.e. if the customer’s service requirements are stochastic variables $S_l$ and $S'_l$, and $S'_l \succ S_l$ then $D^{(S_l)} \succ D^{(S'_l)}$ and clearly $\mathbb{E}\{D^{(S'_l)}\} \geq \mathbb{E}\{D^{(S_l)}\}$.

Similarly, using class notation, if $S_x$ is the service requirement distribution of class $x$ customers. Then $S_u \succ S_{u'} \Rightarrow \mathbb{E}\{D(u)\} \geq \mathbb{E}\{D(u')\}$.

In conclusion, we have shown that service policies that do not give preferred service to shorter jobs, actually discriminate against those jobs. This provides one more justification for prioritizing shorter jobs.
Remark 10.1. Using the same arguments it can be shown that Theorem 10.3 also holds in the case of a preemptive system, providing that the preemption of a customer with service $s' > s$, during the period at which it receives the first $s$ units of service, is unchanged, i.e. preemptions are not determined by the length of the service required by the customer.

10.2.2 The Effect of Class Prioritization

We now move on to study how class prioritization affects class discrimination.

Theorem 10.4. In a $G/G/M$ system with $U$ classes, if the scheduling policy belongs to the class of preemptive priority scheduling policies, then $\mathbb{E}\{D(1)\} \geq 0$ and $\mathbb{E}\{D(U)\} \leq 0$.

Proof. Let $N_u(t)$ be the number of class $u$ customers in the system at epoch $t$. As the scheduling policy belongs to the class of preemptive priority scheduling policies, if $N_1(t) \leq M$, then all $N_1(t)$ customers are served at epoch $t$. Otherwise, $M$ out of them are served. Thus

$$
\tilde{D}(1)(t) = \begin{cases} 
N_1(t) - \frac{N_1(t)}{N(t)} & N_1(t) \leq M \\
N_1(t) & N_1(t) > M
\end{cases} \quad \text{where } N_1(t) \leq M,
$$

which is greater or equal to zero since $\omega(t) N_1(t) \leq N(t)$ and $N_1(t) \leq N(t)$.

Thus, $\tilde{D}(1)(t) \geq 0 \Rightarrow \tilde{D}(1) \geq 0 \Rightarrow \mathbb{E}\{\tilde{D}(1)\} \geq 0$, and from (10.4), $\mathbb{E}\{D(1)\} \geq 0$.

Note that (10.4) also provides the only epochs in which $\tilde{D}(1)(t) = 0$, namely when either $N_1(t) = N(t)$, i.e. all the customers in the system are of class 1, or $N(t) < M$, i.e. there are less than $M$ customers in the system, or $N_1(t) = 0$. In fact, for every class $u$, $\tilde{D}(u)(t) = 0$ when either $N_u(t) = N(t)$, or $N(t) < M$, or $N_u(t) = 0$.

As for $\tilde{D}(U)(t)$, it equals zero when either $N_U(t) = N(t)$, or $N(t) < M$, or $N_U(t) = 0$. Otherwise there are two cases, either $N(t) - N_U(t) \geq M$ or $N(t) - N_U(t) < M$. In the first case there are more than $M$ customers of higher priority in the system, and thus no class $U$ customers are being served. Therefore, $\tilde{D}(U)(t) = -N_U(t)M/N(t)$ which is less than or equal to zero. In the second case there are some class $U$ customers being served.
In this case let $\omega_U(t)$ be the number of class $U$ customers served at epoch $t$. Using this notation

$$\tilde{D}(U)(t) = \omega_U(t) - \frac{N_U(t)M}{N(t)} = \frac{\omega_U(t)N(t) - N_U(t)M}{N(t)}.$$  \hspace{1cm} (10.5)

To prove that this value is less than or equal to zero, let $N'(t) = N(t) - M$ denote the number of customers waiting at epoch $t$, all of whom must be of class $U$. We can write $N(t) = M + N'(t)$, $N_U(t) = \omega_U(t) + N'(t)$. Substituting into (10.5) yields

$$\tilde{D}(U)(t) = \frac{\omega_U(t)(M + N'(t)) - (\omega_U(t) + N'(t))M}{N(t)} = \frac{(\omega_U(t) - M)N'(t)}{N(t)} \leq 0,$$

since $\omega_U(t) \leq M$. Thus, $\tilde{D}(U)(t) \leq 0 \Rightarrow \tilde{D}(U) \leq 0 \Rightarrow \mathbb{E}\{\tilde{D}(U)\} \leq 0$, and from (10.1), $\mathbb{E}\{D(U)\} \leq 0$.

The important thing about Theorem 10.4 is that the most prioritized class has greater or equal to zero discrimination, even if the customers are extremely small. This means that at least for the first priority class, certain discrimination can be guaranteed.

Having shown that discrimination of the most prioritized class is always non-negative, and that of the least prioritized class is always non-positive, one might expect that the discrimination is monotonic with the class priority. However, as the following example shows, this is not the case. Consider a 4-class $M/M/1$ type system with preemptive resume priority. All four classes have an arrival rate of 0.01, and all but class 2 have a mean service requirement of 10 ($\mu = 0.1$). For class 2 we will consider $\mu = 0.1, 0.2, 0.3, 0.4, 0.5$.

Figure 10.1 depicts the class discrimination for the four classes. The results were achieved through simulation, although, as shown in the next section, similar results can be achieved through numerical analysis.

Observe that when the service requirement of class 2 is equal to that of the other classes, the class discrimination is monotonic with the class priority. However, when class 2 has smaller service requirement, this is not the case. This means that class prioritization is limited in its effect in multiple class systems.
One interesting case is the case where there are two classes, and the prioritized class has infinitesimal service requirements (i.e. \( 1/\mu_1 \to 0 \)). It is easy to see that in that case \( \mathbb{E}\{D_{(1)}\} \to 0 \). Interestingly if \( \lambda_1/\lambda_2 \to \infty \) it is also easy to see that \( \mathbb{E}\{D_{(2)}\} \to 0 \). This can be seen immediately if one considers the following simple to derive conservation law:

\[
\frac{\sum_{u=1}^{U} \lambda_i \mathbb{E}\{D_{(i)}\}}{\sum_{u=1}^{U} \lambda_i} = 0.
\]  

(10.6)

It is interesting to draw the similarity of this law to the pseudo-conservation law regarding the waiting time of customer classes in a single server systems (see Kleinrock [85], Chap. 3.4).

10.2.3 Exact Analysis in the Single Server Markovian Distribution Case

We now move on to provide exact analysis of the fairness in the single server case with Markovian distribution, using the methodology in Section 7.1.

We first analyze a multiple class system with exponential distribution, FCFS policy and
no prioritization, to study class discrimination and the effects classes themselves have on the unfairness.

Analysis of a Single Server FCFS System with Multiple Classes

Consider a single server system, with $U$ classes of customers, where class $u$ arrivals follow a Poisson process with rate $\lambda_u$, and their required service requirements are i.i.d. exponentially with mean $1/\mu_u$, $u = 1, 2, \ldots, U$. The total arrival rate is denoted by $\lambda \overset{\text{def}}{=} \sum_{u=1}^{U} \lambda_u$ and for stability it is assumed that $\rho = \sum_{u=1}^{U} \lambda_u/\mu_u < 1$.

(1) Let $a \in \mathbb{N}^0$ denote the number of customers ahead of $C$ in the queue, including the customer in service, $b \in \mathbb{N}^0$ denote the number of customers behind $C$ and $u$ denote the class of the customer currently in service. Due to the Markovian nature of the system the state $\langle a, b, u \rangle$ captures all that is needed to predict the future of $C$.

(2) Since $N(a, b, u) = a + b + 1$, $s(a, b, u) = 1(a = 0)$ and $\omega(a, b, u) = 1$, the momentary discrimination at $\langle a, b, u \rangle$ is simply $\delta(a, b, u) = \mathbb{1}(a = 0) - \frac{1}{a+b+1}$

Since $\mu(a, b, u) = \mu_u$ and $\lambda(a, b, u) = \lambda$ the moments of the slot size are

$$t^{(1)}(a, b, u) = \frac{1}{\lambda + \mu_u}, \quad t^{(2)}(a, b, u) = \frac{2}{(\lambda + \mu_u)^2} = 2(t^{(1)})^2, \quad a, b \geq 0.$$

(3) Let $D_j(a, b, u)$ be a random variable denoting the discrimination of customer of class $j$ doing a walk starting at state $\langle a, b, u \rangle$, with first moment $d_j(a, b, u)$ and second moment $d^{(2)}(a, b, u)$.

Let $p_u$ be the probability that a customer is of class $u$, $p_u = \lambda_u/\lambda$.

The possible events after a slot in which $C$ (which is of class $j$) observed the state $\langle a, b, u \rangle$ are

1. A customer arrives to the system. The probability of this event is $\tilde{\lambda}_u \overset{\text{def}}{=} \lambda/(\lambda + \mu_u)$. $C$’s state changes to $\langle a, b + 1, u \rangle$. 

Chapter 10. Fairness of Prioritization Systems 136
2. For \( a > 1 \): A customer leaves the system, and the next customer to be served is of class \( v \). The probability of this event is \( p_v \tilde{\mu}_u \) where \( \tilde{\mu}_u \overset{\text{def}}{=} \mu_u / (\lambda + \mu_u) \) for every \( v = 1, \ldots, U \), and \( C \)’s state changes to \( \langle a - 1, b, v \rangle \).

3. For \( a \leq 1 \): A customer leaves the system. The probability of this event is \( \tilde{\mu}_u \). If \( a = 1 \) \( C \)’s state changes to \( \langle 0, b, j \rangle \). Otherwise \( C \) leaves the system.

This leads to the following set of linear equations

\[
d_j(a, b, u) = t(1)(a, b, u)\delta(a, b, u) + \tilde{\lambda}_u d_j(a, b + 1, u) + \tilde{\mu}_u \begin{cases} \sum_{v=1}^U p_v d_j(a - 1, b, v) & \text{if } a > 1 \\ d_j(0, b, j) & \text{if } a = 1 \\ 0 & \text{if } a = 0 \end{cases}
\]

(10.7)

(4) Similarly, the equations for \( d_j^{(2)}(a, b, u) \) are

\[
d_j^{(2)}(a, b, u) = t(2)(a, b, u)(\delta(a, b, u))^2 + \tilde{\lambda}_u d_j^{(2)}(a, b + 1)
\]

\[+ \tilde{\mu}_u \begin{cases} \sum_{v=1}^U p_v d_j^{(2)}(a - 1, b, v) & \text{if } a > 1 \\ d_j^{(2)}(0, b, j) & \text{if } a = 1 \\ 0 & \text{if } a = 0 \end{cases}
\]

\[+ 2t(1)(a, b, u)\delta(a, b, u) \left( \tilde{\lambda}_u d_j(a, b + 1, u) + \tilde{\mu}_u \begin{cases} \sum_{v=1}^U p_v d_j(a - 1, b, v) & \text{if } a > 1 \\ d_j(0, b, j) & \text{if } a = 1 \\ 0 & \text{if } a = 0 \end{cases} \right).\]

(10.8)

(5) Let \( \mathbb{E}\{D_j \mid k, u\}, k \in \mathbb{N}, u = 1, \ldots, U \) denote the expected value of discrimination of a class \( j \) customer, given that the customer sees \( k \) customers on arrival including the one being served, and the one being served is of class \( u \). Let \( \mathbb{E}\{D_j \mid 0, 0\} \) denote the expected value of discrimination when the customer finds an empty system. For completeness, \( \mathbb{E}\{D_j \mid 0, u \neq 0\} \overset{\text{def}}{=} 0, \mathbb{E}\{D_j \mid k \neq 0, 0\} \overset{\text{def}}{=} 0 \). Similarly define \( \mathbb{E}\{D_j^2 \mid k, u\} \).
Clearly

\[ \mathbb{E}\{D_j \mid k, u\} = \begin{cases} d_j(k, 0, u) & k > 0 \\ d_j(0, 0, j) & k = 0 \end{cases} \]

\[ \mathbb{E}\{D^2_j \mid k, u\} = \begin{cases} d_j^{(2)}(k, 0, u) & k > 0 \\ d_j^{(2)}(0, 0, j) & k = 0 \end{cases}. \]

For calculating the class discrimination

\[ \mathbb{E}\{D(j)\} = \sum_{k=0}^{\infty} \sum_{u=0}^{U} \mathbb{E}\{D_j \mid k, u\} P_{k,u}, \]

Where \( P_{k,u}, k \in \mathbb{N}, u = 1, \ldots, U \) is the steady state probably of seeing \( k \) customers on arrival, including the one being served, and the one being served is of class \( u \); \( P_{0,0} \) is the steady state probably of finding an empty system and \( P_{k,u} \) \( = 0 \) for \( k = 0, u \neq 0 \) or \( k \neq 0, u = 0 \).

To calculate the unfairness let \( D_j \) be a random value denoting the discrimination observed by a customer of class \( j \), then

\[ \mathbb{E}\{D^2_j\} = \sum_{k=0}^{\infty} \sum_{u=0}^{U} \mathbb{E}\{D^2_j \mid k, u\} P_{k,u} \]

and

\[ F_{D^2} = \sum_{j=1}^{U} p_j \mathbb{E}\{D^2_j\}. \quad (10.9) \]

The steady state probabilities, \( P_{n,u} \), can be numerically calculated, for example, from
the balance equations (see the state diagram for $U = 2$ in Figure 10.2):

$$
(\lambda + \mu_u)P_{n,u} = \sum_{v=1}^{U} \mu_v P_{u+1,v} \left\{ \begin{array}{ll}
\lambda P_{n-1,u} & n > 1 \\
\lambda_u P_{0,0} & n = 1, \quad n > 0
\end{array} \right.
$$

(10.10)

$$
\lambda P_{0,0} = \sum_{v=1}^{U} \mu_v P_{1,v}
$$

(10.11)

$$
\sum_{i=1}^{\infty} \sum_{u=1}^{U} P_{i,u} + P_{0,0} = 1.
$$

(10.12)

which can be evaluated numerically to any required accuracy. Note that steady state

![Figure 10.2: Single Server FCFS with Two Customer Classes: State Diagram](image)

probabilities of multiple class systems were also recently studied in Sleptchenko [141], van Harten and Sleptchenko [147], Sleptchenko et al. [142], Harchol-Balter et al. [64] and more, and are available for use. Alternately, one may possibly want to use the Laplace-Stieltjes transforms derived for $M/M/c$ and $M/G/1$ with vacations, with priorities, see Kella and Yechiali [78, 79].

**Computational Aspects** (10.10)-(10.12) provide a set of linear equations for calculating $P_{n,u}$. Once one decides on a maximum relevant number of customers that can be seen on
arrival, say $K$. This leads to a sparse set of $UK + 1$ linear equations. See, for example, Anderson et al. [3], Davis [11], for efficient methods of solving such a set.

(10.7) and (10.8) provide a recursive method for calculating $d_j(a, b, u)$ and $d^2_j(a, b, u)$. Specifically, if one sets $d_j(a, b, u) = d^{(2)}_j(a, b, u) = 0$ for $a > K$ and for $b > K$, one can start with $b = K, a = 0$, calculating $d_j(a, b, u)$ for $a = 1, 2, \ldots, K$, then moving on to $b = K - 1, a = 0$ and so on. After the calculation of $d_j(a, b, u)$ is completed, one can do the same for $d^2_j(a, b, u)$. This leads to a time complexity of $O(UK^2)$. Since the computation can be done in a column by column fashion, each column depending only on the column preceding it, the space complexity is linear in $K$.

Our experience shows that choosing $K = 30$ for $\rho = 0.8$ yields values very close to those achieved through simulation.

Remark 10.2 (The size of $K$). One way to approximate the size of $K$ is to note that in a single class system, the steady state probability of having more than $K$ customers in the system is $\rho^{K+1}$. The selection of $K$ may be determined by the value one wants to allow for $\rho^{K+1}$. Suppose $\rho^{K+1} = 10^{-3} \Rightarrow K = -3/\log \rho - 1$, which for $\rho = 0.8$ yields $K = 30$. While this is only an approximation of the system analyzed, practice shows that it is not a bad estimate.

System Evaluation: Numerical Results We now present unfairness properties of the FCFS M/M/1 system with two classes for a variety of parameter sets. Of particular interest is to examine these properties as a function of the service difference between the customer classes, expressed by the mean service requirement ratio $\mu_1/\mu_2$. As demonstrated in Chapter 8, the unfairness of the system is sensitive to the utilization $\rho$. Therefore, we maintain a constant $\rho = 0.8$, independently of $\mu_1/\mu_2$. For simplicity, the evaluation is done for equal arrival rates of $\lambda_1 = \lambda_2 = 0.1$. Figure 10.3 depicts the results of the analysis. Figure 10.3(a) depicts the class discrimination while Figure 10.3(b) depicts the unfairness. We note the following properties:

1. Class discrimination for the class with larger mean service requirement (smaller $\mu$),
is positive, at the expense of the other class. This is due to Theorem 10.3 and (10.6).

2. The discrimination is monotone-increasing with the service requirement, as expected from Theorem 10.3.

3. The difference in class discrimination between the two classes increases as the ratio between the mean service requirements increases.

4. The unfairness observed by a specific class of customers is largest when that class has a large service requirement. A similar property was observed for a single class $M/M/1$ (Figure 8.5 Observation 3).

5. The system unfairness increases as the ratio between the service requirements increases. The reason for this is twofold, both an increase in “inter-class” unfairness, as stated in Observation 2 above, and increasing differences between the discriminations of customers within the same class, or “intra-class” unfairness, as stated in Observation 4.
Chapter 10. Fairness of Prioritization Systems

Analysis of a Single Server Priority System with Multiple Classes

We now move on to analyze the single server preemptive priority system with Markovian distributions.

Assume $C$ is of class $j$. Note that customers of classes $j + 1, \ldots, U$ are never ahead of $C$ in the queue.

(1) Let $a = (a_1, \ldots, a_j)$ be a vector representing the number of customers ahead of $C$ per class, i.e. $a_i \in \mathbb{N}^0$ denotes the number of customers ahead of $C$ in the queue from class $i$, including the customer in service. Let $b \in \mathbb{N}^0$ denote the number of customers behind $C$. Due to the Markovian nature of the system the state $\langle a, b \rangle$ captures all that is needed to predict the future of $C$.

(2) Since $N(a, b) = \Sigma a + b + 1$, $s(a, b) = \prod_{i=1}^{j} \mathbb{I}(a_i = 0)$ and $\omega(a, b) = 1$, the momentary discrimination at $\langle a, b \rangle$ is

$$\delta(a, b) = \prod_{i=1}^{j} \mathbb{I}(a_i = 0) - \frac{1}{\Sigma a + b + 1}$$

Let $u(a)$ be the class of the customer currently in service, i.e.

$$u(a) = \begin{cases} \min(i \mid a_i > 0) & \Sigma a > 0 \\ j & \Sigma a = 0 \end{cases}.$$ 

Then $\mu(a, b) = \mu_{u(a)}$ and $\lambda(a, b) = \lambda$, so the moments of the slot size are

$$t^{(1)}(a, b) = \frac{1}{\lambda + \mu_{u(a)}}, \quad t^{(2)}(a, b) = \frac{2}{(\lambda + \mu_{u(a)})^2} = 2(t^{(1)})^2, \quad a, b \geq 0.$$ 

(3) Define $I^j_i$, the vector of length $j$ with all zeroes and one in the $i$-th location, $(a_1 = 0, \ldots, a_i = 1, \ldots, a_j = 0)$. The possible events after a slot in which $C$ (which is of class $j$) observed the state $\langle a, b \rangle$ are
Chapter 10. Fairness of Prioritization Systems

1. A customer of class $i$ arrives to the system. The probability of this event is $p_i\tilde{\lambda}(a)$, where $\tilde{\lambda}(a) \overset{\text{def}}{=} \lambda / (\lambda + \mu(a))$, for each $i = 1, \ldots, U$. If $i < j$ C’s state changes to $\langle a + I_j^i, b \rangle$. If $i \geq j$ C’s state changes to $\langle a, b + 1 \rangle$.

2. A customer leaves the system. The probability of this event is $\tilde{\mu}(a) \overset{\text{def}}{=} \mu(a) / (\lambda + \mu(a))$. If $\Sigma > 0$ C’s state changes to $\langle a - I_j^j, b \rangle$. Otherwise C leaves the system.

This leads to the following set of linear equations

$$d_j(a, b) = t^{(1)}(a, b)\delta(a, b) + \tilde{\lambda}(a) \left( \sum_{i=1}^{j-1} p_i d(a + I_j^i, b) + d(a, b + 1) \sum_{i=j}^U p_i \right)$$

$$+ \mathbb{1}(\Sigma > 0) \tilde{\mu}(a) d(a - I_j^j, b).$$

(4) The equations for $d_j^{(2)}(a, b)$ are

$$d_j^{(2)}(a, b) = t^{(2)}(a, b)(\delta(a, b))^2 + \tilde{\lambda}(a) \left( \sum_{i=1}^{j-1} p_i d^{(2)}(a + I_j^i, b) + d^{(2)}(a, b + 1) \sum_{i=j}^U p_i \right)$$

$$+ \mathbb{1}(\Sigma > 0) \tilde{\mu}(a) d^{(2)}(a - I_j^j, b)$$

$$+ 2t^{(1)}(a, b)\delta(a, b) \left( \tilde{\lambda}(a) \left( \sum_{i=1}^{j-1} p_i d(a + I_j^i, b) + d(a, b + 1) \sum_{i=j}^U p_i \right) \right).$$

$$+ \mathbb{1}(\Sigma > 0) \tilde{\mu}(a) d(a - I_j^j, b)$$

(5) Let $k$ be a class occupancy vector, i.e. $k_u \in \mathbb{N}^0, u = 1, \ldots, U$ is the number of
customers in the system of class $u$. Then

$$
\mathbb{E}\{D_j \mid k\} = d_j(k^{(j)}, \sum_{u=j+1}^{U} k_u)
$$

$$
\mathbb{E}\{D_j^2 \mid k\} = d_j^{(2)}(k^{(j)}, \sum_{u=j+1}^{U} k_u),
$$

where $k^{(j)}$ denote the vector $k$ truncated to be of size $j$, i.e. $k^{(j)} \overset{\text{def}}{=} (k_1, \ldots, k_j)$.

For calculating the class discrimination

$$
\mathbb{E}\{D(j)\} = \sum_{k_1=0}^{\infty} \cdots \sum_{k_U=0}^{\infty} \mathbb{E}\{D_j \mid k\} P_k,
$$

where $P_k, k_i \in \mathbb{N}^0$ is the steady state probability of seeing $k_i$ customers of class $i$ upon arrival, including the one being served.

To calculate the unfairness

$$
\mathbb{E}\{D_j^2\} = \sum_{k_1=0}^{\infty} \cdots \sum_{k_U=0}^{\infty} \mathbb{E}\{D_j^2 \mid k\} P_k
$$

and (10.9) holds.

The balance equations for calculating $P_k$ are

$$(\lambda + 1(\Sigma_k > 0)\mu_{u(k)})P_k = \sum_{v=1}^{U} 1(k_v > 0)\lambda_v P_{k-1}^U + \sum_{v=1}^{U} 1(u(k + I_v^U) = v)\mu_v P_{k+I_v^U},$$

$$
\sum_{k_1=0}^{\infty} \cdots \sum_{k_U=0}^{\infty} P_k = 1,
$$

where we use a simpler $u(k) = \min(i \mid k_i > 0)$. 
Chapter 10. Fairness of Prioritization Systems

Computational Aspects  The balance equations lead to a sparse set of $K^U$ linear equations. It is therefore quite obligatory to use other methods for finding the steady state probabilities for large values of $U$. However, for $U = 2$ which we used it is still manageable.

Evaluating the equation sets for $d_j(a, b)$ and $d_j^{(2)}(a, b)$ is computationaly hardest for $j = U$. One method to do this is to iterate the equalities starting with an initial guess, say zero, until the required relative accuracy is reached. For example, one can keep iterating until the relative change in the sum of absolute values is smaller than some small constant $\alpha$, say $\alpha = 10^{-10}$. If $I$ iterations are needed to reach the required relative accuracy, the time complexity is $O(IK^{U+1})$. As the computation requires keeping a copy of the values of $d(a, b)$ for one iteration, the space complexity is $O(KU^{U+1})$.

Our experience shows that for $U = 2, \rho = 0.8$, choosing $K = 30, \alpha = 10^{-10}$ leads to $I = 200$ at the worst case. The results achieved were very close to those reached by simulation.

In order to make the algorithm practical when the number of classes $U$ is large the following approximation methods can be utilized:

1. A lower (upper) bound of $\mathbb{E}\{D_j\}$ and $\mathbb{E}\{D_j^2\}$ can be calculated by treating all classes of higher priority than $j$ as one class, with the expected service requirement of the class with the lowest (highest) expected service requirement between those classes, and an arrival rate which is the sum of the arrival rates. Note that all classes with lower priority are treated as one class anyway. This leads to $U = 3$ which is usually practical to analyze.

2. An approximation of $\mathbb{E}\{D_j\}$ and $\mathbb{E}\{D_j^2\}$ can be calculated by treating all classes of higher priority than $j$ as one class, with an expected service requirement which is the weighted (by arrival rate) average of their expected service requirements, and an arrival rate which is the sum of the arrival rates. As before, all classes with lower priority are treated as one class anyway. This also leads to $u = 3$. How far this result is from the exact one depends on the variability across the service requirements of these classes.
3. A closer approximation of $\mathbb{E}\{D_j\}$ and $\mathbb{E}\{D_j^2\}$ can be calculated by treating all classes of higher priority as if they are on one queue. Whenever some customer of that queue is to receive service, its class is randomly chosen based on a distribution which can be estimated as follows. Assume there are $k$ customers of higher classes in the system. The probability of service a class $i$ customer in the next slot is the probability of having no customers of classes $1, 2, \ldots, i-1$ in the system, given that there are $k$ customers in the system. Finding the exact distribution of this might also not be an easy task, but it might be more easily approximated.

**System Evaluation: Numerical Results**

One specific case of interest is the following: suppose that a call center services two types of customers, one requiring only a brief approval, and one requiring the full attention of a service person for several minutes. It is common to suggest, due to fairness reasons, that customers with shorter service requirement should be served ahead of other customers. For simplicity assume that the rates of arrival of both customer classes are equal.

It might be true that this suggestion is indeed fair. This, however, may depend on the parameters, and it is reasonable to predict that the shorter the service requirements of the priority class are, the greater are the fairness benefits relative to FCFS. One can therefore predict that there is some minimum ratio between the mean service requirement of the “preferred” class to that of the rest of the population, below which the priority schedule is more fair than FCFS.

As we did in the FCFS case, we maintain constant utilization $\rho = 0.8$, independently of $\mu_1/\mu_2$ and $\lambda_1 = \lambda_2 = 0.1$. Figure 10.4 depicts the class discrimination in the priority queue. We observe that:

1. Class 1 is always positively discriminated, and class 2 is always negatively discriminated, as expected from Theorem 10.4. Indeed, the discrimination of class 1 customers is at the expense of class 2 customers, because of (10.6).

2. The positive (negative) discrimination is monotone-increasing (decreasing) in the
Chapter 10. Fairness of Prioritization Systems

expected service requirement of class 1 customers, as expected from Theorem 10.3, see the discussion after that theorem.

Figure 10.5 depicts the system unfairness in the priority queue, and for comparison, in the FCFS queue analyzed above. For the prioritized system one may observe that:

1. The highest system unfairness is observed at the left part of the figure. This is the case where very long jobs (class 1) receive priority over the short jobs. This behavior is naturally expected.

2. In the right side of the figure, where the shorter jobs receive priority, the system unfairness slightly increases with the service requirement ratio. This slight increase is perhaps a result of unfairness between class 2 customers and themselves, which increases at this region due to the increased variability in service requirement.

Comparing the systems, we observe that:
3. When class 1 customers have longer expected service requirement it is less fair, system-wise, to give them priority.

4. When class 1 customers have shorter expected service requirement and the ratio is over 2 : 1 it is more fair, system-wise, to use two queues and give priority to the shorter jobs.

Next we study the effect the variability of service distributions has on these results. As discussed in Section 8.4, our analysis methodology can be easily generalized to the Phase-type distribution case. Following Brosh et al. [27] we used Erlang-10 and Coxian-2 distributions (see e.g. Adan and Resing [2], Chap. 2) with the same expected values to study the smaller and larger variability cases, respectively.

For the Erlang-10 distribution we have a coefficient of variation of $1/\sqrt{10}$ (The coefficient of variation of a random variable $X$ is $\text{stdev}\{X\}/\mathbb{E}\{X\}$), while for the Coxian-2 distribution we use the settings suggested by Marie [97], to achieve a coefficient of variation of $\sqrt{10}$: to achieve an expected value of $S$ with a coefficient of variation of $C$ use
\( \mu_1 = 2/S, p_1 = 1/(2C^2), \mu_2 = 1/(C^2S) \). Figure 10.6 depicts the results. One may observe that the behavior is very similar to the behavior observed for Exponential distribution. One difference is that when prioritizing the shorter class, in the right side of the figures, for the Erlang-10 distribution there is almost no increase in system unfairness, and in the Coxian-2 there is a large increase. This agrees with our expectations since for Erlang-10 there is very little variability within the class, and the opposite for Coxian-2.

To conclude this section, we observe that each distribution is characterized by a threshold. If the ratio between the mean non-prioritized job size and the mean prioritized job size is below this threshold, it is more fair to serve the customers in FCFS manner. Otherwise, the priority manner is more fair. The results seem to agree with common intuition. For example regarding the call server example - it is less fair to prioritize a specific class of customers over another class, unless the service requirement of the prioritized customers is small enough compared to the others.
10.3 Resource Dedication to Classes

In this section we deal with the dedication of resources to classes. We consider systems where each class is assigned a dedicated set of servers and a FCFS queue. We focus on analyzing the class discrimination in these systems.

Perhaps the most common approach for dedicating resources to classes is by assigning each class a set of one or more servers and associating each class with a single FCFS queue. These are very common in human-service facilities, including airport passport control systems which are customarily divided to alien and non-alien classes, and public restrooms. Since in some of these systems customers are classified based on personal properties, e.g. gender or nationality, fairness aspects of these systems are highly important. An operational question of interest is whether to allocate equal amount of resources to the different classes or to grant more resources to the class with the larger service requirement. The answer to this question is not immediate since one of the basic principles of the RAQFM fairness measure is that short jobs should get preference over long jobs.

We start by showing that for a wide set of conditions the answer to the question posed above is indeed to grant more resources to the class with the larger service requirement, as under equal resource allocation the more heavily loaded class is subject to negative discrimination; nonetheless, we show that this does not always hold and there are counter examples.

We then provide an algorithmic approach for deriving the class discrimination for more general systems. The algorithm we propose exploits the structure of the problem and yields results in polynomial complexity despite the exponential size of the state space. Lastly, we show numerical examples.

10.3.1 Dominance Results for 2 Class Systems

**Theorem 10.5.** Consider a dual class system where each class behaves like a GI/M/1 system. Let $A_u, S_u, u = 1, 2$ be random variables denoting the interarrival time and the service requirement, respectively, of class $u$ customers. Then, if either (i) $A_1 \prec A_2$ and
1/\mu_1 \geq 1/\mu_2$, or (ii) $A_1 \leq A_2$ and $1/\mu_1 > 1/\mu_2$ then $\mathbb{E}\{D_{(1)}\} < \mathbb{E}\{D_{(2)}\}$.

**Proof.** Let $\tilde{D}_{(u)}$ be a random variable denoting the total instantaneous discrimination rate to class $u$ customers at steady state. $\tilde{D}_{(1)}$ can be derived by conditioning on the system state and examining several cases:

1. server 1 is idle: In this case no class 1 customers are present in the system and thus $\tilde{D}_{(1)} = 0$.

2. server 2 is idle and server 1 is busy: In this case the total warranted service to class 1 customers is 1 and the granted service to the class is also 1. Thus $\tilde{D}_{(1)} = 0$.

3. server 1 and server 2 are busy: Let $n_i > 0$ be the number of customers present at the system of class $i$. Then the total warranted service to class 1 customers is given by $2n_1/(n_1 + n_2)$ while the granted service is 1. The total discrimination is $\tilde{D}_{(1)} = 1 - 2n_1/(n_1 + n_2) = (n_2 - n_1)/(n_1 + n_2)$.

Let $p(n_1, n_2)$ be the probability that at an arbitrary epoch there are $n_1, n_2$ customers in the system. Then the above leads to:

$$\mathbb{E}\{\tilde{D}_{(1)}\} = \sum_{n_1=1}^{\infty} \sum_{n_2=1}^{\infty} p(n_1, n_2) \frac{n_2 - n_1}{n_1 + n_2}.$$  

The expected discrimination for a customer of class 1, $\mathbb{E}\{D_{(1)}\}$ can be derived from (10.1). Further, note that in the case of server dedication $N_1$ is independent of $N_2$ and thus $p(n_1, n_2) = p_1(n_1)p_2(n_2)$ where $p_i(n_i)$ is the probability that $N_i = n_i$. These lead to:

$$\mathbb{E}\{D_{(1)}\} = \frac{1}{\lambda_1} \sum_{n_1=1}^{\infty} \sum_{n_2=1}^{\infty} p_1(n_1)p_2(n_2) \frac{n_2 - n_1}{n_1 + n_2}$$  

$$\mathbb{E}\{D_{(2)}\} = \frac{1}{\lambda_2} \sum_{n_1=1}^{\infty} \sum_{n_2=1}^{\infty} p_1(n_1)p_2(n_2) \frac{n_1 - n_2}{n_1 + n_2}.$$
Now the difference between these values is:

\[
\mathbb{E}\{D_{(2)}\} - \mathbb{E}\{D_{(1)}\} \geq \frac{1}{\lambda_1} \sum_{n_1=1}^{\infty} \sum_{n_2=1}^{\infty} p_1(n_1)p_2(n_2) \frac{2(n_1 - n_2)}{n_1 + n_2}.
\]

where we used \(\lambda_1 \geq \lambda_2\), which holds for both (i) and (ii). Note that when \(n_1 = n_2\) the term inside the sum is zero. We can therefore sum just for \(n_1 \neq n_2\) in the following way:

\[
\mathbb{E}\{D_{(2)}\} - \mathbb{E}\{D_{(1)}\} \\
\quad \geq \frac{1}{\lambda_1} \left( \sum_{n_1=1}^{\infty} \sum_{n_2=1}^{n_1-1} p_1(n_1)p_2(n_2) \frac{2(n_1 - n_2)}{n_1 + n_2} + \sum_{n_2=1}^{\infty} \sum_{n_1=1}^{n_2-1} p_1(n_1)p_2(n_2) \frac{2(n_1 - n_2)}{n_1 + n_2} \right) \\
\quad = \frac{1}{\lambda_1} \sum_{n_1=1}^{\infty} \sum_{n_2=1}^{n_1-1} (p_1(n_1)p_2(n_2) - p_1(n_2)p_2(n_1)) \frac{2(n_1 - n_2)}{n_1 + n_2}.
\]

We now require that \(\mathbb{E}\{D_{(2)}\} - \mathbb{E}\{D_{(1)}\} > 0\). Since \(n_1 > n_2\), a sufficient requirement is that

\[
\frac{p_1(n_1)}{p_1(n_2)} > \frac{p_2(n_1)}{p_2(n_2)}
\]

for any \(n_1 > n_2 \geq 1\), and we move on to show when this condition holds.

For the GI/GI/1 model with P-LCFS, where interarrival and service requirements are distributed as \(A\) and \(S\) respectively, it is known that the steady state probability of having \(n\) customers in the system at arbitrary times is geometric, given by \(p(k) = \rho(1-\sigma)\sigma^{k-1}, k = 1, 2, \ldots\) where \(\rho = \mathbb{E}\{S\}/\mathbb{E}\{A\} = \lambda\mathbb{E}\{S\} < 1\), \(\sigma = (\mathbb{E}\{B\} - 1)/\mathbb{E}\{B\}\), and \(B\) is the steady state number of customers served in one busy period (see Núñez-Queija [102] for a review of the literature on this subject).

In the GI/M/1 case with FCFS, where \(S\) is exponentially distributed with mean \(1/\mu\), the same applies, and we have \(\mathbb{E}\{B\} = (1 - A^*(\mu))/(1 - 2A^*(\mu))\) where \(A^*(s), s \geq 0\) is the Laplace transform of \(A\). This is immediately obtained when noticing that \(B\) is 1 with probability \(1 - A^*(\mu)\) and is distributed as \(B_1 + B_2\), where \(B_1\) and \(B_2\) are i.i.d as \(B\), with probability \(A^*(\mu)\). Therefore \(\sigma = A^*(\mu)/(1 - A^*(\mu))\).
Using the geometric forms of \( p_1(n) \) and \( p_2(n) \) we get that (10.13) is true iff \( \sigma_1 > \sigma_2 \) which is true iff \( A^*_1(\mu_1) > A^*_2(\mu_2) \) where \( A^*_i(s), s \geq 0 \) is the Laplace transform of \( A_i, i = 1, 2 \). Since \( A^*_i(s) \) is monotone non-decreasing this holds when either (i) or (ii) holds.

\[ \square \]

Remark 10.3 (Some Comments on Theorem 10.5).

1. In the \( M/M/1 \) and \( D/M/1 \) cases conditions (i) and (ii) take the form \( \rho_1 > \rho_2 \).

2. Conditions (i) or (ii) are sufficient but not necessary. In fact \( A^*_1(\mu_1) > A^*_2(\mu_2) \) combined with \( \lambda_1 \geq \lambda_2 \) is also satisfactory.

3. The final part of the proof (proving that (10.13) holds when either (i) or (ii) holds) can be achieved in several other ways, e.g. utilizing the fact that \( \sigma = A^* (\mu - \mu \sigma) \) and that in our case \( A^*_1(s) < A^*_2(s) \). However, we find that the proof above is the more elegant one, and requires the least limitations.

Conjecture 10.1. Consider a dual class system where each class behaves like a \( M/GI/1 \) system. Then, if either (i) \( \lambda_1 > \lambda_2 \) and \( S_1 \succeq S_2 \), or (ii) \( \lambda_1 \geq \lambda_2 \) and \( S_1 \succ S_2 \) then \( \mathbb{E}\{D_1(1)\} < \mathbb{E}\{D_2(2)\} \).

We base this conjecture on the fact that the steady state occupancy probabilities in an \( M/GI/1 \) type system, \( p(n) \), can be expressed using the following recursion (see Neuts [101]):

\[
p(0) = 1 - \rho \\
p(k + 1) = \frac{1}{a_0}[\alpha_{k+1}p(0) + \sum_{v=1}^{k} \alpha_{k-v+2}p(v)],
\]

where \( a = \{a_j\}_{j=0}^{\infty} \) is the probability function of the number of arrivals during a customer’s service requirement and \( \alpha_j = \sum_{k=j}^{\infty} a_k \). Let \( a^{(i)} = \{a^{(i)}_j\}_0^{\infty}, i = 1, 2 \) denote the probability function for class \( i \), and similarly define \( \alpha^{(i)}_j \). Then obviously both (i) and (ii) imply that \( a^{(1)}_i \geq a^{(2)}_i, i \geq 1 \) and \( a^{(1)}_0 \leq a^{(2)}_0 \), implying \( \alpha^{(1)}_i \geq \alpha^{(2)}_i, i \geq 1 \) and hinting that \( p_1(n_1)/p_1(n_2) > p_2(n_1)/p_2(n_2) \) for \( n_2 > n_1 \).
Chapter 10. Fairness of Prioritization Systems

The claim of Theorem 10.5 does not necessarily hold if one demands only that \( \rho_1 > \rho_2 \) and considers an arbitrary G/G/1 system. Consider for example a system where the service requirements of both classes are deterministic, equaling one unit, the arrivals of class 1 are deterministic at intervals of one unit (D/D/1) and the arrivals to class 2 occur in bulks of size \( k \) at inter-arrival time of \( m > k \) units. The instantaneous discrimination of class 1 is given by

\[
\tilde{D}(1) = \frac{1}{m} \sum_{i=1}^{k} (1 - \frac{2}{i+1})
\]

and that of class 2 is given by

\[
\tilde{D}(2) = \frac{1}{m} \sum_{i=1}^{k} (1 - \frac{2i}{i+1}).
\]

It is easy to see that \( \tilde{D}(1) > 0 \) and \( \tilde{D}(2) < 0 \) and thus \( D(1) > 0 > D(2) \).

10.3.2 Analysis of Class Discrimination in Systems with Many Classes

Consider a system with \( U \) classes, indexed 1, \ldots, U, each directed to a dedicated server with a single queue and served according to FCFS. We assume that the arrival process and service requirements of class \( i \) are independent of that of class \( j \) (1 \( \leq \) \( i \neq j \leq U \)). Thus, the steady state occupancy (number in system) of class \( i \) is independent of that of class \( j \).

Let \( p(i)(n) \) denote the probability that the number of customers of class \( i \) present in the system is \( n \). Since class \( i \) forms an independent queue, the values of \( p(i)(n) \) can be derived from the literature for a wide class of systems. For example, for an M/M/1 type queue \( p(i)(n) = (1 - \rho_i)\rho_i^n \). For an M/G/1 queue one can take the Pollaczek-Khinchin formula of the Laplace-Stieltjes transform (LST) of the queue occupancy and use standard numerical procedures to derive from it the values of \( p(i)(n) \). We will therefore assume that these values are given and show how to derive from them the class discriminations.

Below we demonstrate how to compute the discrimination experienced by class \( u \). Let \( p_{(i)}(1,2,\ldots,k)(n) \) denote the steady state probability that the system of classes 1,2,\ldots,\( k \) contains together \( n \) customers and \( l \) of their servers are busy. Obviously, one should consider only \( 0 \leq l \leq k \) and \( n \geq l \). Let \( p_{(i)}(n) \) denote the same probability for a system consisting of class \( i \) servers only. Using these two probability vectors we can compute
Chapter 10. Fairness of Prioritization Systems

\( p^{(1,2,...,k,k+1)}(l) \) from \( p^{(1,2,...,k)}(l) \) and \( p^{(k+1)}(l) \) as follows:

\[
p^{(1,2,...,k+1)}(n) = p^{(1,2,...,k)}(n)p^{(k+1)}(0) + \sum_{i=1}^{n} p^{(1,2,...,k)}(n-i)p^{(k+1)}(i)
\]

\[ 1 < l \leq k \] \hspace{1cm} (10.14a)

\[
p^{(1,2,...,k+1)}(1) = p^{(1,2,...,k)}(1)p^{(k+1)}(0) + p^{(1,2,...,k)}(0)p^{(k+1)}(1)
\]

\[ l = 1, \] \hspace{1cm} (10.14b)

\[
p^{(1,2,...,k+1)}(0) = p^{(1,2,...,k)}(0)p^{(k+1)}(0) = \prod_{i=1}^{k+1} (1 - \rho_i)
\]

\[ l = 0. \] \hspace{1cm} (10.14c)

Note that the actual convolution is performed in Equation (10.14a), and we denote this convolution in vector term as \( p^{(1,2,...,k)}(l) \ast p^{(k+1)}(l) \). Let \( N \) be the number of probability elements one keeps for each vector. Then the computational complexity of performing this convolution is \( O(N^2) \). Since \( 1 < l \leq k \), exactly \( k - 1 \) such convolutions are required and the overall complexity is \( O(kN^2) \).

Applying the above procedure in a recursive mode can yield \( p^{(1,2,...,U-1)}(l) \) from \( p^{(1)}(l) \), \( l = 0, 1, i = 1, \ldots, k \) leading to an overall complexity of \( O(k^2N^2) \).

Now, the expected instantaneous discrimination rate for class \( U \) can be computed from the vectors \( p^{(1,2,...,U-1)}(l) \) and \( p^{(U)}(l) \) as follows:

\[
\mathbb{E}\{\hat{D}(U)\} = 0 \cdot p^{(U)}(0) + \sum_{i=1}^{N} p^{(U)}(i) \sum_{l=0}^{U-1} \sum_{j=l}^{N} p^{(1,2,...,U-1)}(j) \left( 1 - (l + 1) \frac{i}{l + n} \right).
\]

The computational complexity of this expression, plus the complexity of recursively computing \( p^{(1,2,...,U-1)}(l) \) is \( O(U^2N^2) \).

If one wishes to compute the expected value of the instantaneous discrimination rate for all \( U \) classes the computational complexity is \( O(U^3N^2) \) steps.
Finally, the expected discrimination for each class can be derived from (10.15) and from the discrimination version of Little’s Theorem (10.1).

10.3.3 Numerical Results

Several questions need to be addressed in this context of resource dedication to classes. 1) How servers should be assigned to each class as to lead to maximally fair scheduling. 2) How fair are current practices. 3) How high is the class-discrimination experienced under various server assignments.

We start with a simple stochastic service example. Consider a system where the Poisson arrival rates of the two classes are identical $\lambda_1 = \lambda_2 = 0.005$ and the service requirements are exponential where the mean of class 1 doubles that of class 2: $\mu_1 = 0.12, \mu_2 = 0.24$. We consider a system consisting of 12 servers and evaluate system unfairness and class discrimination as a function of the server assignment policy (dedication of $k : 12 - k$ where $k$ is the dedication to class 1 and $12 - k$ to class 2). Figure 10.7 depicts these results.

![Figure 10.7: Unfairness and Class Discrimination in a 12 server Dedication System](image)

The figure demonstrates that the discrimination of a class increases with the number of servers allocated to it. It also demonstrates that equal assignment of servers negatively
(and drastically) discriminates class 1 and that the best operation point (smallest unfairness value and smallest absolute values of class discriminations) is to assign the servers proportionally to the mean service requirement of the class. Note that this agrees with an argument that was made in some public debates regarding public restroom assignment. The argument is that since the service requirement of women is approximately double that of men, women should be assigned twice the number of restrooms. Nonetheless, the class discrimination experienced at the optimal operation point (8:4) is not exactly zero, resulting from the slight differences in behavior between the 8 server system and the 4 server system.

We repeated this examination for uniformly distributed service requirements and found similar results.

We now move on to another interesting scenario, the rush hour scenario. In some cases customers appear over a short period of time, the “rush hour”, and are served from that moment until all customers leave the system. One example for such a system is a computerized call centers or an Internet Web server, which expects a large influx of customers arriving concurrently, due to a major TV advertisement. Such a system may be expecting two customer classes, previously registered members and new members, with different service requirements. A second example is the queue for the restrooms in theaters, at the beginning of the theater break. For simplicity we assume that all customers arrive concurrently.

For the sake of presentation, and to obtain tractable results, we consider a simple deterministic example. We assume a total number of 12 servers and $12j$ customers in each class, $j$ even. The service requirements of class 1 and class 2 are 1 and 2 time units respectively. We consider two intuitive and common server assignment policies: 1) Proportional to the service length, that is 4 and 8 servers to class 1 and 2, respectively, and 2) Equal, namely 6 servers to each class.

We track the system in the proportional case along time slots of one time unit. Since $12j$ jobs of each type are present at time zero, the number of slots is $3j$. The number of short jobs present, counting from the last slot backwards, is given by $4i$, $i = 1, \ldots, 3j$, and
the number of long jobs is 8, 8, 16, 16, 24, ..., 12j, 12j. Thus the overall warranted service of class 1 (along the 3j slots) is given by

\[ 12 \sum_{i=1}^{3j/2} \frac{2i}{4i} + \frac{2i - 1}{4i - 1}. \]

Recalling that the total granted service to class 1 is 12j, and that there are 12j customers, the class discrimination of class 1 is given by

\[ \mathbb{E}\{D_{(1)}\} = 1 - \frac{3}{4} - \frac{3j/2}{j} \sum_{i=1}^{3j/2} \frac{2i - 1}{4i - 1} \approx 1 - \frac{1}{4} - \frac{1}{j} \int_{x=1}^{3j/2} \frac{2x - 1}{4x - 1} dx = \frac{1}{4} - \frac{6j + \ln 3 + \ln(6j - 1) - 4}{8j}, \]

which for large values of j tends to \(-1/2\), while \(\mathbb{E}\{D_{(2)}\} = -\mathbb{E}\{D_{(1)}\} = 1/2\).

For the equal allocation case a similar analysis yields the overall warranted service of class 1 to be:

\[ 12 \sum_{i=1}^{j} \frac{2i - 1}{j + 3i - 1} + \frac{2i}{3i + j}, \]

which yields the class discrimination:

\[ \mathbb{E}\{D_{(1)}\} \approx 1 - \frac{1}{9j} \left( -12 + 12j - 2j \ln 4 + 2j \ln \frac{3 + j}{j} + (1 + 2j) (\ln(2 + j) - \ln(4j - 1)) \right) \]

which for large values of j tends to \((\log 256 - 3)/9 \approx 0.28\).

The analysis reveals that under our fairness model, neither proportional assignment nor equal assignment is most fair. Under proportional assignment the short jobs are negatively discriminated due to the relatively small number of servers allocated to them, out of proportion to their part in the population. Under equal assignment the long jobs are negatively discriminated due to the considerable amount of time during which they form a large majority of the presence, as most short jobs are gone earlier, while receiving only half of the resources. The values of discrimination under both allocations are considerably high. The “optimal” point of operation is therefore at the 7:5 allocation in which a simulation shows that \(\mathbb{E}\{D_{(1)}\} \approx 0.06\).
Both of the above numerical examples showed that it is indeed more fair to dedicate more resources to the more heavily utilized class. But what if the two classes are equally utilized? Would it then be more fair to dedicate servers to classes according to their service requirement, or would it be preferred to assign them to queues randomly?

In the system analyzed for Figure 10.8 $U = 2, M = 2$ and there are two queues. Both classes have exponentially distributed service requirements with utilizations $\rho_1 = \rho_2 = 0.45$. For class 2 customers $\mu_2 = 0.09, \lambda_2 = 0.1$. For class 1 customers, $\mu_1$ varies from 0.09 to 0.09*0.8 and $\lambda_1$ from 0.1 to 0.8. In the non-dedicated case customers are assigned to the queues randomly. In the dedicated case each queue is dedicated to one of the classes. In any case, jockeying from the head of the queue is allowed when servers are idle (see Section 9.1.2). The figure demonstrates the following properties:

1. Dedicating a queue to each of the classes is more fair than mixing the populations, for every ratio of service requirements.

2. As the ratio between service requirements grows, the ratio between the fairness values grows, reaching a peak of 1.3 at a ratio of 8:1.
10.4 Combining Servers

In this section we study the effect multiple serves have on the unfairness, in the context of a multi-class system. As mentioned in the beginning of this chapter, this issue might have fitted better in Chapter 9, but as it was first studied in Raz et al. [116] in the context of multiple class systems we bring it here.

Our purpose is to address the practice of combining servers. For example, observe the check-in line for flights, where most airline companies use a single queue multiple server setup. Assume that the rate at which customers can be checked in at each post can be doubled by assigning to each post one person for baggage check-in, and another for flight seat assignment. Would doing this, while halving the number of serving posts, increase or decrease the system fairness?

It is well known that if servers are combined while maintaining the processing rate fixed, the efficiency improves. This is quite obvious since when servers are not combined, there is loss of system service rate when the number of customers in the system is lower than the number of servers, see Kleinrock [84], Theorem 4.2. However, as we will show, combining servers increases the unfairness, and this increase, depending on service variability, may be quite meaningful.

We do the analysis in the case of exponential service requirements, using the methodology in Section 7.1. We analyze and compare three systems: 1) a single server multiple class system. This was already analyzed in Section 10.2.3. 2) a multiple server multiple class system. 3) A ‘hybrid’ or ‘hypothetical’ system, where there are multiple servers, however, when the number of customers is lower than the number of servers, the servers can combine efforts in such a way that the total service rate remains the same. We denote this system ‘hypothetical’ since in most applications it is impractical to implement. We bring it here mainly for the sake of comparison to the single server system, since they are equally efficient.
10.4.1 Analysis of Multiple-Server Multiple-Class FCFS system

The analysis is similar to the analysis in Section 10.2.3 with the addition of multiple servers artifacts, similar to the ones used in Section 9.2.1.

(1) Let \( a = 0, M, M + 1, \ldots \) denote the number of customers ahead of \( C \), including served customers, and \( b \in \mathbb{N}^0 \) the number of customers behind \( C \). If \( C \) is in service, \( b \) also includes customers served by other servers. Note the unavoidable jump occurring in the value of \( b \) when \( C \) enters service, and the fact that the values \( 1 \leq a < M \) are invalid. Let \( u \) denote the classes of the customers currently in service, i.e. \( u_i = 1, \ldots, U, i = 1, \ldots, M \) is the class of the customer served by the \( i \)-th server, and \( u_i = 0 \) indicates that the \( i \)-th server is idle. Due to the Markovian nature of the system the state \( \langle a, b, u \rangle \) captures all that is needed to predict the future of \( C \).

(2) Since \( N(a, b, u) = a + b + 1, s(a, b, u) = \mathbb{1}(a = 0) \) and \( \omega(a, b, u) = \min(a + b + 1, M) \), the momentary discrimination at \( \langle a, b, u \rangle \) is

\[
\delta(a, b, u) = \mathbb{1}(a = 0) - \frac{\min(a + b + 1, M)}{a + b + 1}
\]

The departure rate at a slot in which the state is \( \langle a, b, u \rangle \) is

\[
\mu(a, b, u) \overset{\text{def}}{=} \mu_u = \sum_{i=1}^{M} \mathbb{1}(u_i \neq 0) \mu_u.
\]

as \( \lambda(a, b, u) = \lambda \) the moments of the slot size are

\[
t^{(1)}(a, b, u) = \frac{1}{\lambda + \mu_u}, \quad t^{(2)}(a, b, u) = \frac{2}{(\lambda + \mu_u)^2} = 2(t^{(1)})^2.
\]

(10.16)

(3) Let \( u_i' \) be the vector \( u \) where the \( i \)-th location is replaced by \( v \). The possible events after a slot in which \( C \) (which is of class \( j \)) observed the state \( \langle a, b, u \rangle \) are

1. A customer arrives to the system. The probability of this event is \( \lambda(u) \overset{\text{def}}{=} \lambda/(\lambda + \mu_u) \). \( C \)'s state changes to \( \langle a, b + 1, u \rangle \).
2. For \( a > M \): A customer leaves the system from the \( i \)-th queue, and the next customer to be served is of class \( v \). The probability of this event is \( p_v \bar{\mu}(u_i, u) \) where 
\[
\bar{\mu}(u, u) \overset{\text{def}}{=} \mu_u / (\lambda + \mu_u),
\]
for every \( i = 1, \ldots M \) and \( v = 1, \ldots, U \). C’s state changes to \( \langle a - 1, b, u^v_i \rangle \).

3. For \( a = M \): A customer leaves the system. The probability of this event is \( \bar{\mu}(u_i, u) \) for every \( i = 1, \ldots M \). For simplicity and w.l.o.g. we rearrange the servers such that \( C \) is always served by the first server. C’s state changes to \( \langle 0, b + M - 1, (u_v^y)^1_i \rangle \).

4. For \( a = 0, b \geq M \): A customer other than \( C \) leaves the system, and the next customer to be served is of class \( v \). The probability of this event is \( p_v \bar{\mu}(u_i, u) \) for every \( i = 2, \ldots M \) such that \( u_i \neq 0 \) and every \( v = 1, \ldots, U \). C’s state changes to \( \langle 0, b - 1, u^v \rangle \).

5. For \( a = 0, b < M \): A customer other than \( C \) leaves the system. The probability of this event is \( \bar{\mu}(u_i, u) \) for every \( i = 2, \ldots M \) such that \( u_i \neq 0 \). C’s state changes to \( \langle 0, b - 1, u^0_v \rangle \).

6. \( C \) leaves the system.

This leads to the following set of linear equations

\[
d_j(a, b, u) = t^{(1)}(a, b, u) \delta(a, b, u) + \tilde{\lambda}(u)d_j(a, b + 1, u)
\]  
\[
+ \begin{cases} 
\sum_{i=1}^M \sum_{v=1}^U p_v \bar{\mu}(u_i, u)d_j(a - 1, b, u^v_i) & a > M \\
\sum_{i=1}^M \bar{\mu}(u_i, u)d_j(0, b + M - 1, (u^v)^1_i) & a = M \\
\sum_{i=2}^M p_v \bar{\mu}(u_i, u)d_j(0, b - 1, u^v) & a = 0, b \geq M \\
\sum_{i=2}^M \mathds{1}(u_i \neq 0) \bar{\mu}(u_i, u)d_j(0, b - 1, u^0) & a = 0, b < M
\end{cases}
\]

(4) We omit the equations for \( d_j^{(2)}(a, b, u) \) for brevity.

(5) Let \( \mathbb{E}\{D_j \mid k, u\} \), \( k \in \mathbb{N} \), \( u_i = 0, \ldots, U \), \( i = 1, \ldots, M \) denote the expected value of discrimination of a class \( j \) customer, given that the customer sees \( k \) customers on
arrival, including those being served, and the one being served on the \( i \)-th server is of class \( u_i \), or if none is served \( u_i = 0 \). For completeness, \( \mathbb{E}\{D_j \mid k, u\} \overset{\text{def}}{=} 0 \) for every \( \sum_{i=1}^{M} \mathbbm{1}(u_i \neq 0) \neq \min(M, k) \). Similarly define \( \mathbb{E}\{D_j^2 \mid k, u\} \).

We have

\[
\mathbb{E}\{D_j \mid k, u\} = \begin{cases} 
  d_j(k, 0, u) & k \geq M \\
  \frac{1}{M-k} \sum_{i=1}^{M} \mathbbm{1}(u_i = 0) d_j(0, k, (u_i^{u_i^{(1)}})^2) & k < M 
\end{cases}
\]

\[
\mathbb{E}\{D_j^2 \mid k, u\} = \begin{cases} 
  d_j^2(k, 0, u) & k \geq M \\
  \frac{1}{M-k} \sum_{i=1}^{M} \mathbbm{1}(u_i = 0) d_j^2(0, k, (u_i^{u_i^{(1)}})^2) & k < M 
\end{cases}
\]

For calculating the class discrimination

\[
\mathbb{E}\{D(j)\} = \sum_{k=0}^{\infty} \sum_{i=1}^{M} \sum_{u_i=0}^{U} \mathbb{E}\{D_j \mid k, u\} P_k, u,
\]

Where \( P_k, u, k \in \mathbb{N} \quad u_i = 0, \ldots, U \quad i = 1, \ldots, M \) is the steady state probability of seeing \( k \) customers on arrival, including those being served, and the one being served by the \( i \)-th server is of class \( u_i \); \( P_k, u \overset{\text{def}}{=} 0 \) for \( \sum_{i=1}^{M} \mathbbm{1}(u_i \neq 0) \neq \min(M, k) \).

To calculate the unfairness let \( D_j \) be a random value denoting the discrimination observed by a customer of class \( j \), then

\[
\mathbb{E}\{D_j^2\} = \sum_{k=0}^{\infty} \sum_{i=1}^{M} \sum_{u_i=0}^{U} \mathbb{E}\{D_j^2 \mid k, u\} P_k, u
\]

and (10.29) holds.

The steady state probabilities, \( P_{n,u} \), can be numerically calculated, for example, from the balance equations, which we omit for brevity.

Note that this entire analysis was done assuming each server has one unit of service rate. However, we would require that each server has \( 1/M \) units of service rate, so it is comparable with the single server case. This influences only the momentary discrimination
and the slot size moments. For the momentary discrimination \( s(a, b, u) = \mathbb{1}(a = 0)/M \)
and \( \omega(a, b, u) = \min(a + b + 1, M)/M \) so
\[
\delta(a, b, u) = \frac{\mathbb{1}(a = 0)}{M} - \frac{\min(a + b + 1)}{M(a + b + 1)}.
\]
The departure rate at a slot in which the state is \( (a, b, u) \) is
\[
\mu(a, b, u) \overset{\text{def}}{=} \mu_u = \frac{\sum_{i=1}^{M} \mathbb{1}(u_i \neq 0)\mu_{u_i}}{M},
\]
and with this new definition of \( \mu_u \) (10.16) holds.

10.4.2 Analysis of Multiple-Server Multiple-Class ‘Hybrid’ FCFS system

The analysis is very similar to the analysis above for the case where each server has 1/M units of service rate, except \( s(a, b, u) = \mathbb{1}(a = 0)/\min(M, a + b + 1) \) and \( \omega(a, b, u) = 1 \)
\[
\delta(a, b, u) = \frac{\mathbb{1}(a = 0)}{\min(M, a + b + 1)} - \frac{1}{a + b + 1}.
\]
The departure rate at a slot in which the state is \( (a, b, u) \) is
\[
\mu(a, b, u) \overset{\text{def}}{=} \mu_u = \frac{\sum_{i=1}^{M} \mathbb{1}(u_i \neq 0)\mu_{u_i}}{\min(M, a + b + 1)},
\]
and (10.16) holds. Of course the steady state probabilities also change.

10.4.3 Comparative Results

Figure [10.9] depicts the unfairness in the three systems, single server, dual server and ‘hypothetical’ for the case \( M = 2, U = 2 \), as a function of the mean service requirement ratio. The comparison shows that the dual server configuration is consistently more fair than the single server system. The figure also demonstrates that the unfairness measure of the hypothetical system is even lower than that of the dual server system, though only
slightly. This suggests that the lower measure observed for the dual-server system can be attributed to its more fair operation, and not due to its inefficiency.

The higher fairness of the multiple server configuration can be explained from the resource allocation point of view. In the single server system, the resources are always allocated to one customer at a time. In the multiple server system, the resources are allocated among more customers at any epoch, and therefore the resource allocation is more fair. In fact, as the number of servers grows, the system approaches the processor sharing model, which is the most fair system (see Theorem 5.3); of course under such conditions the system efficiency decreases and customer delays increase leading to potential customer dissatisfaction due to poor performance. The question whether the use of $k$ servers is always more fair than the use of a single server remains open for future research.

10.5 Summary

We studied the effect classification and prioritization mechanisms have on fairness and the discrimination, as observed by customers of different classes.
We first studied full prioritization. We first showed that it is justified to give short customers priority, since otherwise these customers are naturally negatively discriminated. However, we showed that prioritization can only guarantee positive discrimination to the class of highest priority. Regarding the unfairness, we demonstrated that there is some threshold value for the ratio between class job sizes, under which prioritization does not increase fairness. If the ratio is over this threshold, prioritization increases the fairness.

We then studied resource dedication. We first showed that under a wide set of conditions it is more fair to assign more resources to classes which are more heavily loaded. Nonetheless, we showed that this is not always the case. We demonstrate that in some cases the most fair assignment of servers is proportional to the load, but not in all cases. We also demonstrated that if loads are equal, queue dedication might still increases the fairness, if only because short jobs do not need to wait for longer jobs.

Finally we addressed combining servers. We showed that combining servers while keeping the service rate increases unfairness, and it is more fair to split the servers if this is possible.
Chapter 11

THE TWIN MEASURE FOR PREDICTABILITY

The material in this chapter was published in Raz et al. [123].

11.1 Introduction to Predictability

How does one measure the predictability of a queueing system?

The importance of the issues of predictability is widely recognized in many works and applications.

Experiments and experience show the increasing the predictability of the response time is sometimes more important to users than reducing the response times themselves because waiting much longer than expected causes far more user frustration than simply waiting longer on average. Taylor [146] shows that uncertainty about waiting time influences overall service satisfaction in a thorough way. Leclerc et al. [91] show that consumers have an aversive reaction to not knowing how long they will have to wait. Dellaery and Kahn [42] state that

“if a consumer is uncertain about how long they will have to wait before accessing the web site of interest there will be negative affect, and this negative affect will affect the evaluation of the web site reading event.”

Such effects and others are also reported by Zhou and Zhou [161], Hui and Tse [68] and more.

The issue of predictability is also strongly related to the issue of Fairness which is studied in depth in the preceding chapters. If the system isn’t predictable, it is in many
cases because some jobs are treated unfairly.

Despite its importance, there is little prior art trying to quantitatively capture the notion of predictability. One recent exception is Wierman and Harchol-Balter [154]. We discuss the metrics proposed in that work in detail in Section 11.5.2.

Our purpose in the chapter is therefore to propose a measure for system predictability in queueing systems, that can serve as a required feature in designing such systems, while being simple to measure and analyze.

11.2 Additional Useful Notation

In this chapter we deal with the M/GI/m model.

Assume service requirements are sampled independently with probability density function (pdf) \( b(x) \) and cumulative distribution function (cdf) \( B(x) \). \( B_x \) is the probability of a service requirement of exactly \( x \), and of course for continuous \( b(x) \), \( \forall x, B_x = 0 \). However, for many practical distributions there is at least one value for which there is an accumulation of probability and \( B_x \neq 0 \).

The service requirement has expected value \( \bar{x} \), and a second moment \( \bar{x}^2 \). The load (utilization) of the server is defined as

\[
\rho \overset{\text{df}}{=} \frac{\lambda \bar{x}}{s} = \frac{\lambda}{s} \int_0^\infty tb(t)dt,
\]

and for stability we require \( \rho < 1 \).

One useful quantity is the load made up by the jobs of size less than or equal to \( x \), denoted \( \rho(x) \), which is

\[
\rho(x) \overset{\text{df}}{=} \frac{\lambda}{s} \int_0^x tb(t)dt.
\]

We also define the load made up by the jobs of size strictly less than \( x \), denoted \( \rho(x^-) \), which is

\[
\rho(x^-) \overset{\text{df}}{=} \frac{\lambda}{s} \int_0^{x^-} tb(t)dt.
\]
Chapter 11. The Twin Measure for Predictability

Note that for a continuous pdf \( b(t) \), \( \rho(x) = \rho(x^-) \).

We use the notation \( X \sim BP(x) \) to denote that \( X \) is distributed as a busy period starting with a job of size \( x \). \( X \sim BP_{y}(x) \) and \( X \sim BP_{y^−}(x) \) denote the same, except the busy period is only composed of jobs of size not larger than \( y \) and smaller than \( y \), respectively.

We use the notation \( X \preceq F(t) \) to denote that a random variable \( X \) stochastically dominates \( F(t) \), i.e. \( \mathbb{P}\{X \leq t\} \leq F(t) \ \forall t \geq 0 \). We use the notation \( \prec \) in a similar way.

11.3 Introducing the Twin Measure

Naturally, the issue of predictability deals mainly with how well one is able to forecast the system response, based on available information. The more information one has, the more accurate the prediction can be. For example, knowing the system utilization \( \rho \) one can predict the expected waiting time \( \mathbb{E}\{W\} \). The unpredictability results from the variability in the system. Supplying more information, such as the service requirement of the customer and the state of the system (the number of customers, their service requirement) enables a better prediction, with lower variability.

But what if no information is available? In modern day systems such as call centers and computer systems, where the actual workings of the system are hidden from the customer, there is very little information available for such a prediction, other than the customer’s service requirement.

One practical way to measure and check for system predictability is to launch a pair of identical jobs with deterministically identical service requirements (Twins). For example, it is very natural that a Web user who submits two concurrent identical, or close to identical, requests to the same Web site, will appreciate the site predictability based on their relative response times. We propose a measure that is based on this notion. Obviously, a low “twin measure” is not a sufficient requirement for customer satisfaction. In fact, we do not even claim that it is sufficient for guaranteeing system predictability, as this issue is too complicated to be captured by this simple notion. However, we do claim
that it is a required feature, and is simple to measure and analyze.

Let $C_1$ and $C_2$ be identical jobs, with service requirements $x$ and $x + \delta$ respectively. We call such jobs twins. Let the arrival, departure, and first service epochs of twin $i$, $i = 1, 2$ be $a_i$, $d_i$, and $s_i$ respectively. Assume that $C_1$ arrives when $(a_1)$ the system is in steady state. Assume that the twins arrive $\epsilon$ time units apart, that is, $a_2 - a_1 = \epsilon$, $\epsilon > 0$.

**Definition 11.1** (Twin Measure). Define the random variable $Z(x, \epsilon, \delta) = |d_2 - d_1|$, given $x, \epsilon,$ and $\delta$. Let $z^n(x, \epsilon, \delta)$ be the $n$-th moment of $Z(x, \epsilon, \delta)$, i.e. $z^n(x, \epsilon, \delta) = E\{Z(x, \epsilon, \delta)^n\}$.

For scheduling policy $\phi$ and job size $x$, the $n$-th twin measure $T^\phi_n(x)$ is defined as the limit, when $\epsilon$ and $\delta$ tend to zero, of $z^n(x, \epsilon, \delta)$, assuming a single limit exists. Namely $T^\phi_n(x) = \lim_{\epsilon \to 0, \delta \to 0} z^n(x, \epsilon, \delta)$.

The shortened term twin measure, denoted $T^\phi(x)$, is used to describe the first twin measure, namely $T^\phi(x) = T^\phi_1(x) = \lim_{\epsilon \to 0, \delta \to 0} E\{Z(x, \epsilon, \delta)\}$. While we will focus in this work on the first twin measure, we find it useful to define it in the scope of higher moments, for future research.

**Remark 11.1.** The $n$-th twin measure is only defined when a single limit exists for both $\delta \searrow 0$ and $\delta \nearrow 0$, while $\epsilon \to 0$. One can devise a policy for which there is no such single limit. For example, consider a policy $\phi$ that serves jobs in a FCFS manner, unless the second job in the queue has a smaller size than the first one, in which case it servers the first two jobs in a PS manner. In this case for $\delta \searrow 0$ the twins are served in a PS manner and $T^\phi(x) = 0$, while for $\delta \nearrow 0$ the twins are served in a FCFS manner, and $T^\phi(x) = x$.

The twin measure in such a case can be chosen to be the maximum of all the limits, the mean value, or it can remain undefined, as appropriate for the application.

Definition 11.1 has the benefit that it applies to service distributions for which same size arrivals are impossible. It also avoids a pitfall that a policy $\phi$ can artificially serve equal sized jobs in a different manner than non-equal sized ones. As in some size distributions equal sized jobs are extremely rare, this will not hinder the expected performance of $\phi$, yet allow it to have an artificially low twin measure. However, this definition makes the analysis tedious so for the sake of analysis we use a simpler definition:
Chapter 11. The Twin Measure for Predictability

Definition 11.2 (Simplified Twin Measure). Let $\delta = 0$, i.e. both jobs have the same size. Define the random variable $Z(x, \epsilon) = |d_2 - d_1|$, given $x$ and $\epsilon$. Let $z^n(x, \epsilon)$ be the $n$-th moment of $Z(x, \epsilon)$, i.e. $z^n(x, \epsilon) = \mathbb{E}\{Z(x, \epsilon)^n\}$.

For scheduling policy $\phi$ and job size $x$, the $n$-th twin measure $T^n_\phi(x)$ is defined as the limit, when $\epsilon$ tend to zero, of $z^n(x, \epsilon)$, assuming a limit exists. Namely $T^n_\phi(x) = \lim_{\epsilon \to 0} z^n(x, \epsilon)$.

The shortened term twin measure, is again used to describe the first twin measure, namely $T^\phi(x) = \lim_{\epsilon \to 0} \mathbb{E}\{Z(x, \epsilon)\}$.

Whether we use the original or the simplified definition will be clear from the context.

One can also choose to normalize the twin measure, the obvious normalization factor being $x$. This does not change the results in any significant manner.

11.4 Analyzing Common Scheduling Policies for Single Server Systems

11.4.1 PS

When $C_1$ departs, $C_2$ can have at most $\epsilon + \delta$ remaining service requirement. Therefore $z(x, \epsilon, \delta) \leq (\epsilon + \delta)\bar{N}$ where $\bar{N}$ is the mean number of jobs in the system. Thus

$$T^{PS}(x) = \lim_{\epsilon \to 0, \delta \to 0} z(x, \epsilon, \delta) \leq \lim_{\epsilon \to 0, \delta \to 0} ((\epsilon + \delta)\bar{N}) = 0.$$  

Intuitively, both jobs are identical and arrive simultaneously, so they will receive exactly the same service, and leave the system simultaneously.

11.4.2 FCFS

Let $W(x, \epsilon, \delta)$ be a random variable denoting the time elapsing between $d_1$ and $s_2$. For any non-preemptive scheduling policy we have $Z(x, \epsilon, \delta) = W(x, \epsilon, \delta) + x + \delta$. For FCFS $W(x, \epsilon, \delta)$ is the amount of work arriving in the interval between the arrival epochs of the twins and it has an expected value of $\rho\epsilon$, i.e. $z(x, \epsilon, \delta) = \mathbb{E}\{Z(x, \epsilon, \delta)\} = \rho\epsilon + x + \delta$ and

$$T^{FCFS}(x) = \lim_{\epsilon \to 0, \delta \to 0} (\rho\epsilon + x + \delta) = x.$$
11.4.3 LCFS and P-LCFS

We start with the non-preemptive case. There are two possible orders of service, either (i) $C_1$ is served ahead of $C_2$, i.e. either the server was idle on $a_1$, or the server finished serving the job that was served on $a_1$ at some epoch in the interval $[a_1, a_2)$, and no other job arrived between $a_1$ and that epoch, or (ii) $C_2$ is served first.

(i) We can ignore all jobs served ahead of $C_1$. The service order from then onward is as follows. First $C_1$ is served for $x$ units of time. $C_2$ then waits for a busy period created by all jobs arriving while $C_1$ was served, and is served at the completion of this busy period, for $x + \delta$ units of time. Let $V(x, \epsilon, \delta)$ be a random variable denoting the time elapsing between $s_1$ and $s_2$. Thus $Z(x, \epsilon, \delta) = V(x, \epsilon, \delta) + \delta$. Clearly $V(x, \epsilon, \delta) \sim BP(x)$, thus

$$z(x, \epsilon, \delta) = \frac{x}{1 - \rho} + \delta.$$ 

For example, this can be derived from the transform of the distribution of a busy period starting with a job of size $x$, $G^*(s, x) = e^{-x[s + \lambda - \lambda G^*(s)]}$, where $G^*(s)$ is the transform of the distribution of the busy period length (see e.g. Kleinrock [5], p. 212).

Taking the limit we have

$$T^{LCFS}(x) = \lim_{\epsilon \to 0, \delta \to 0} \left( \frac{x}{1 - \rho} + \delta \right) = \frac{x}{1 - \rho}.$$ 

(ii) We can ignore all jobs served ahead of $C_2$. The service order from then onward is as follows. First $C_2$ is served for $x + \delta$ units of time. Then a busy period created by all jobs arriving while $C_2$ was served is being served. Then a busy period created by all jobs arriving between $a_1$ and $a_2$ is served, followed by $C_1$ being served for $x$ units of time. Thus $Z(x, \epsilon, \delta) = V(x, \epsilon, \delta) - \delta$, where $V(x, \epsilon, \delta) \sim BP(x + \delta + \epsilon)$. Therefore

$$z(x, \epsilon, \delta) = \frac{x + \delta + \epsilon}{1 - \rho} - \delta$$ 

$$T^{LCFS}(x) = \lim_{\epsilon \to 0, \delta \to 0} \left( \frac{x + \delta + \epsilon}{1 - \rho} - \delta \right) = \frac{x}{1 - \rho}.$$
For the preemptive case, all jobs arriving in the interval \([a_1, a_2]\), including \(C_1\), are served in total for a period of length \(\epsilon\) until they are preempted by \(C_2\). \(C_2\) and any preemption jobs are then served until \(d_2\), and can be ignored. The period \((d_2, d_1)\) is composed of \(C_1\) and jobs arriving in the interval \((a_1, a_2)\), and jobs preempting them, minus a service of \(\epsilon\) units of time already done. Therefore \(Z(x, \epsilon, \delta) \sim BP(x + \epsilon) - \epsilon\), so

\[
T^{P-LCFS}(x) = \lim_{\epsilon \to 0, \delta \to 0} \left( \frac{x + \epsilon}{1 - \rho} - \epsilon \right) = \frac{x}{1 - \rho}.
\]

To summarize,

\[
T^{P-LCFS}(x) = T^{LCFS}(x) = \frac{x}{1 - \rho}.
\]

### 11.4.4 SJF and P-SJF

The analysis for SJF is quite tedious if one uses Definition 11.1. Therefore, the analysis provided here uses Definition 11.2, i.e. we assume the twin jobs are of equal size. One can verify that this simplification does not alter the result. We will use this definition from this point onwards.

We start with the non-preemptive case.

Note that in general, SJF does not determine the order of service between equally sized jobs. Obvious choices are either FCFS or LCFS. We call these policies SJF-FCFS and SJF-LCFS respectively.

Starting with SJF-FCFS, \(C_1\) is always served first. We can ignore all jobs served ahead of \(C_1\). The service order from then onwards is as follows. First \(C_1\) is served for \(x\) units of time. This is followed by a busy period composed of jobs of sizes smaller than \(x\) arriving while \(C_1\) was served. Following this, jobs of size \(x\) arriving in the interval \((a_1, a_2)\) are served, followed by a busy period composed of jobs of size smaller than \(x\) arriving while they were served. Lastly, \(C_2\) is served for \(x\) units of time. If we let \(V(x, \epsilon)\) be a random variable denoting the time elapsing between \(s_1\) and \(s_2\) it is easy to see that \(Z(x, \epsilon) = V(x, \epsilon)\).
Chapter 11. The Twin Measure for Predictability

Considering the service order we have \( V(x, \epsilon) \sim BP_x(x) + BP_x(\epsilon \lambda x B_x) \), and therefore

\[
T^{SJF-FCFS}(x) = \lim_{\epsilon \to 0} \left( \frac{x}{1 - \rho(x^-)} + \frac{\epsilon \lambda x B_x}{1 - \rho(x^-)} \right) = \frac{x}{1 - \rho(x^-)}. \tag{11.1}
\]

For SJF-LCFS, note that \( C_1 \) can still be served first, e.g. if the server is idle on \( a_1 \). Therefore, let \( C_f \) and \( C_s \) be the first and second twins to be served, respectively. The order of service starts with \( C_f \), followed by a busy period composed of jobs of sizes not larger than \( x \) arriving while \( C_f \) was served. Following this, jobs of size \( x \) arriving in the interval \((a_1, a_2)\) are served, followed by a busy period composed of jobs of size not larger than \( x \) arriving while they were served. Lastly, \( C_s \) is served for \( x \) units of time. Using the same notation \( Z(x, \epsilon) = V(x, \epsilon) \) and \( V(x, \epsilon) \sim BP_x(x) + BP_x(\epsilon \lambda x B_x) \), leading to

\[
T^{SJF-LCFS}(x) = \lim_{\epsilon \to 0} \left( \frac{x}{1 - \rho(x^-)} + \frac{\epsilon \lambda x B_x}{1 - \rho(x^-)} \right) = \frac{x}{1 - \rho(x^-)}. \tag{11.2}
\]

Note that for continuous service distributions SJF-FCFS and SJF-LCFS have the same twin measure.

We now move on to the preemptive case. We define P-SJF-FCFS and P-SJF-LCFS.

For P-SJF-FCFS \( C_1 \) is served first. When \( C_1 \) finishes service there are no jobs of size smaller than \( x \) in the queue. The next to be served are jobs of size \( x \) arriving in \((a_1, a_2)\), followed by \( C_2 \). Each of those can be interrupted, but only by jobs of size smaller than \( x \). Therefore \( Z(x, \epsilon) \sim BP_x(\epsilon \lambda x B_x) + BP_x(x) \) and (11.1) holds, i.e.

\[
T^{P-SJF-FCFS}(x) = \frac{x}{1 - \rho(x^-)}.
\]

For P-SJF-LCFS \( C_1 \), and other jobs of size \( x \) arriving in \((a_1, a_2)\), can be served for a total service no loner than \( \epsilon \) before being preempted by \( C_2 \). Let \( \beta(\epsilon) \) be a random variable denoting this amount of service. When \( C_2 \) finishes service the next to be served are jobs of size \( x \) arriving in \((a_1, a_2)\), followed by \( C_1 \). Each of those can be interrupted by jobs of size not larger than \( x \). Therefore \( Z(x, \epsilon) \sim BP_x(\epsilon \lambda x B_x) + BP_x(x) - \beta(\epsilon) \). As \( 0 \leq \beta(\epsilon) \leq \epsilon \)}
we have \( \lim_{\epsilon \to 0} \beta(\epsilon) = 0 \) and \( (11.2) \) holds, so

\[
T^{P-SJF-LCFS}(x) = \frac{x}{1 - \rho(x)}.
\]

11.4.5 LJF and P-LJF

We define LJF-FCFS, LJF-LCFS, P-LJF-FCFS, and P-LJF-LCFS in a similar manner to the ones defined for SJF.

Using the same arguments as in Section 11.4.4, and the fact that

\[
\int_{x^+}^{\infty} tf(t)dt = \rho - \rho(x)
\]

\[
\int_{x}^{\infty} tf(t)dt = \rho - \rho(x^-)
\]

we have

\[
T^{P-LJF-FCFS}(x) = T^{LJF-FCFS}(x) = \frac{x}{1 - (\rho - \rho(x))}
\]

\[
T^{P-LJF-LCFS}(x) = T^{LJF-LCFS}(x) = \frac{x}{1 - (\rho - \rho(x^-))}.
\]

This applies to both the non-preemptive and the preemptive case.

11.4.6 LAS

Note that in the LAS scheduling policy jobs with equal attained service share the processor. Upon arrival, \( C_1 \) will be served for at most \( \epsilon \), then \( C_2 \) will be served for an equal amount, and from then on they will have equal attained service, and keep sharing the processor, until they leave the system together. Therefore

\[
T^{LAS}(x) = 0.
\]
11.4.7 SRPT

Note that since both jobs start with equal jobs sizes, from the first epoch in which one of them is served, the other will not be served until the first one leaves the system. It can easy to observe that the twin measure is the same no matter which of the jobs is first served. We will therefore assume w.l.o.g. that $C_1$ is served first. Following Schrage and Miller [139], $Z(x, \epsilon)$ can be decomposed into the sum

$$Z(x, \epsilon) = W(x)^{SRPT} + R(x)^{SRPT}$$

where $W(x)^{SRPT}$ is a random variable denoting the waiting time for $C_2$ of size $x$, i.e. the time from $d_1$ to $s_2$, and $R(x)^{SRPT}$ is a random variable denoting the residence time for $C_2$ of size $x$, i.e. the time from $s_2$ to $d_2$. Both $W(x)^{SRPT}$ and $R(x)^{SRPT}$ do not depend on $\epsilon$ and therefore

$$T^{SRPT}(x) = \lim_{\epsilon \to 0} \frac{1}{x} \{ W(x)^{SRPT} + R(x)^{SRPT} \} = \mathbb{E}\{ W(x) \}^{SRPT} + \mathbb{E}\{ R(x) \}^{SRPT}, \quad (11.3)$$

Starting with $\mathbb{E}\{ W(x) \}^{SRPT}$, note that once $C_1$ enters service, we can divide the arriving jobs into three categories: 1) jobs with service requirement over or equal to $x$. These jobs are served after $d_2$ and therefore can be ignored. 2) jobs with service requirement below the remaining service requirement of $C_1$ on the epoch of their arrival. These jobs will preempt $C_1$ and be served before $d_1$. 3) jobs with service requirement lower than $x$, but above the remaining service requirement of $C_1$ on the epoch of their arrival. These jobs are served in the interval $(d_1, s_2)$.

To carry out this analysis, observe jobs arriving in an infinitesimal interval of size $dt$ when $C_1$ has remaining service requirement $t$. Category 2) jobs preempt $C_1$ and create a sub busy period of size $dt/(1 - \rho(t^-))$ (including the initial $dt$ interval). Category 3) jobs arriving in this sub busy period are to be served after $d_1$, and the work load created by these jobs is

$$\frac{dt}{1 - \rho(t^-)} \int_{t^+}^{x^-} y b(y) dy = \frac{(\rho(x^-) - \rho(t)) dt}{1 - \rho(t^-)}, \quad 0 \leq t < x,$$
and zero for $t = x$.

Note that any job arriving while these jobs are served, with service requirement below $x$, will also be served before $s_2$. Therefore we are facing a busy period of size

$$\frac{(\rho(x) - \rho(t))}{1 - \rho(t)} \frac{dt}{1 - \rho(x)}.$$

Integrating over the service duration of $C_1$ this yields the mean size of the waiting interval, namely

$$\mathbb{E}\{W(x)\}^{SRPT} = \int_0^{x^-} \frac{\rho(x) - \rho(t)}{(1 - \rho(t^-))(1 - \rho(x^-))} dt.$$

As for $\mathbb{E}\{R(x)\}^{SRPT}$, this is simply the residence time of a job with service requirement $x$ under SRPT. This is true since like a regular job, jobs already in the system once $C_2$ begins service are guaranteed to have remaining processing time over $x$, and therefore will not affect the residence time. Thus,

$$\mathbb{E}\{R(x)\}^{SRPT} = \int_0^{x^-} \frac{dt}{1 - \rho(t^-)}.$$

Using (11.3) we get

$$T^{SRPT}(x) = \int_0^{x^-} \frac{\rho(x) - \rho(t)}{(1 - \rho(t^-))(1 - \rho(x^-))} dt + \int_0^{x^-} \frac{dt}{1 - \rho(t^-)}$$

$$= \int_0^{x^-} \frac{1 - \rho(t)}{(1 - \rho(t^-))(1 - \rho(x^-))} dt,$$

which for a continuous pdf $b(t)$ is simply

$$T^{SRPT}(x) = \frac{x}{1 - \rho(x)}.$$
11.4.8 LRPT

Under LRPT all jobs leave the system at the end of the busy period in which they arrive. Specifically, \( C_1 \) and \( C_2 \) leave the system simultaneously, leading to

\[
T^{LRPT}(x) = 0.
\]

11.4.9 RR

We analyze the RR policy with service quantum \( \Delta \) where \( \Delta \ll x \), and for simplicity we assume that service requirements are multiples of \( \Delta \). We use a model quite similar to the one described in Kleinrock [85], Sec 4.4, except that newly arriving jobs join the queue after the last arriving job.

As \( \epsilon \to 0 \), i.e. \( \Delta \gg \epsilon \) we can be certain that even if \( C_1 \) begins service before \( a_2 \), it cannot finish a single quantum before \( a_2 \). Therefore the only jobs between \( C_1 \) and \( C_2 \) in the queue are jobs arriving in the interval \((a_1, a_2)\) and \( C_1 \) will never be more than one service cycle ahead of \( C_2 \). Let \( N(\epsilon) \) be a random variable denoting the number of jobs arriving in the interval \((a_1, a_2)\). Some of these jobs will have shorter service requirement than \( x \), and therefore \( \Delta \leq Z(x, \epsilon) \leq (N(\epsilon) + 1)\Delta \). However, note that \( N(\epsilon) \) has an expected value of \( \lambda\epsilon \), and therefore \( \Delta \leq y(x, \epsilon) \leq \Delta + \lambda\epsilon \) leading to

\[
T^{RR}(x) = \lim_{\epsilon \to 0} y(x, \epsilon) = \Delta.
\]  \:(11.4)

Note that \:(11.4) holds also for other models of RR, e.g. when arriving jobs join the queue in other positions, though the analysis in some cases is somewhat more complicated.

11.5 Discussion: Twin Measure Results

In this section we discuss the twin measures obtained in the previous sections. We start with proposing a classification and comparing the policies in each class. We then compare this classification to the classification provided by Wierman and Harchol-Balter
which we call the Conditional Response Time Criterion, or CRTC.

We finalize with a discussion of the meaning of optimality under the twin measure.

For simplicity, we use the measures obtained for continuous service distributions. This is also convenient since Wierman and Harchol-Balter \cite{154}, Wierman \cite{152} deals mainly with continuous service distributions (although the results can probably be extended to the non-continuous case).

11.5.1 Classifying the Scheduling Policies

\textbf{Definition 11.3 (Twin Measure Classification).} A scheduling policy $\phi$ will be called \textit{Absolutely Twin Predictable} if $T^\phi(x) = 0$ for every $x$.

A scheduling policy $\phi$ will be called \textit{Strongly Twin Predictable} if $T^\phi(x) \leq x$ for every $x$.

A scheduling policy $\phi$ will be called \textit{Weakly Twin Predictable} if $T^\phi(x) < x/(1 - \rho)$ for every $x$.

A scheduling policy $\phi$ will be called \textit{Not Twin Predictable} if $T^\phi(x) \geq x/(1 - \rho)$ for at least one value of $x$.

The reason we use this classification will be made clear once we discuss the scheduling policies within each class.

\textit{Absolutely Twin Predictable}

This class includes all policies for which twin jobs will leave the system simultaneously, namely PS, LAS and LRPT.

This clearly demonstrates that the twin measure has very little to do with the efficiency aspects of performance, as these three service policies have very distinct service characteristics. LRPT is notoriously inefficient, in fact, it is the most inefficient policy possible in a non-idling server, while LAS is optimal for some service distributions, as first mentioned by Yashkov \cite{160}, see Nuyens and Wierman \cite{103} for a survey of results.
It also demonstrates that the twin measure is not sufficient to guarantee sojourn time predictability. For example LRPT has notoriously unpredictable sojourn times, as the sojourn time depends not only on the amount of work in the system upon arrival, but also on the amount of work arriving from that moment until the end of the busy period. So even with full state information, good prediction of the sojourn time is impossible.

**Strongly Twin Predictable**

This class includes policies for which the twin measure for a job of size $x$ is not larger than $x$. Observe that the minimum twin measure one can expect from any non-preemptive scheduling policy is $x$. One can see that FCFS achieves this optimal value for non-preemptive scheduling policies.

The other policy in this class is RR which in the limit $\Delta \to 0$ becomes PS.

**Not Twin Predictable**

LCFS has the largest measure analyzed, $x/(1-\rho)$. We conjecture that LCFS had the largest twin measures amongst non-preemptive scheduling policies. Note that the difference between the conjectured best and worst non-preemptive scheduling policies, FCFS and LCFS, can be extreme in cases where $\rho \to 1$.

**Weakly Twin Predictable**

Policies in this class include SJF, LJF and SRPT.

For small jobs, such that $\rho(x) < \rho - \rho(x)$, SJF has the lower measure, and the opposite for long jobs.

Interestingly, SJF and SRPT have the same measure. This can be explained in the following way. We call a job *intervening* if the job gets served in the interval $(d_1, d_2)$, or $(d_2, d_1)$ if $C_2$ is served first, and is not $C_1$ or $C_2$. Observe that under SJF a job of size $x$ is only intervened by jobs of size $y < x$ arriving in a period of size $x$. In the non-preemptive case this period is the period in which the first job is served, while in the preemptive case
this period is the period in which the second job is served.

Now consider SRPT. Observe a period of time of length $dt$ in which the first job to be served was already served for $t$ units of time. Jobs intervening in this period are jobs with size $t \leq y < x$. Observe a period of time of length $dt$ in which the second job to be served was already served for $t$ units of time. Jobs intervening in this period are jobs with size $y < t$. So in total, intervening jobs for every such interval $dt$ are of size $y < t$. Now consider that the total length of such intervals of length $dt$ is $x$, so the jobs intervening a job of size $x$ under SRPT are also jobs of size $y < x$ arriving in a period of size $x$.

This is an interesting observation as it shows a similarity un-observed before between SRPT and SJF.

### 11.5.2 Comparing Predictability Criteria

In this section we compare the results of the twin measure classification with that of CRTC. CRTC was proposed in Wierman and Harchol-Balter [154]. Further results are provided in Wierman [152], specifically for the case $\mathbb{E}\{X^3\} = \infty$. We start by summarizing

![Figure 11.1: Classification According to CRTC](image-url)

the Conditional Response Time Criterion in the settings and notation of our work.
**Definition 11.4** (Conditional Response Time Criterion). A job of size $x$ is treated predictably under policy $\phi$, service with pdf $b(x)$, and load $\rho$ if the conditional variance in response time seen by a job of size $x$ under policy $\phi$, $\text{Var}\{T(x)\}^\phi$, follows

$$\frac{\text{Var}\{T(x)\}^\phi}{x} \leq \frac{\lambda x^2}{(1-\rho)^3}.$$ 

A scheduling policy $\phi$ is predictable if every job size is treated predictably.

A scheduling policy $\phi$ is: (i) **Always Predictable** if $\phi$ is predictable under all loads and service distributions; (ii) **Sometimes Predictable** if $\phi$ is predictable under some loads and service distributions; and unpredictable under other loads and service distributions or (iii) **Always Unpredictable** if $\phi$ is unpredictable under all loads and service distributions.

Figure 11.1 summarizes the results of classifying common scheduling policies according to CRTC (the figure is from Wierman [152], brought with permission of the author).

<table>
<thead>
<tr>
<th>Policy</th>
<th>Twin</th>
<th>CRTC</th>
</tr>
</thead>
<tbody>
<tr>
<td>PS</td>
<td>Absolutely</td>
<td>Always</td>
</tr>
<tr>
<td>LAS</td>
<td>Absolutely</td>
<td>Sometimes</td>
</tr>
<tr>
<td>LRPT</td>
<td>Absolutely</td>
<td>Never</td>
</tr>
<tr>
<td>FCFS</td>
<td>Strongly</td>
<td>Sometimes</td>
</tr>
<tr>
<td>LCFS</td>
<td>Not</td>
<td>Never</td>
</tr>
<tr>
<td>P-LCFS</td>
<td>Not</td>
<td>Always</td>
</tr>
<tr>
<td>SJF</td>
<td>Weakly</td>
<td>Sometimes</td>
</tr>
<tr>
<td>P-SJF</td>
<td>Weakly</td>
<td>Sometimes</td>
</tr>
<tr>
<td>LJF</td>
<td>Weakly</td>
<td>Never</td>
</tr>
<tr>
<td>P-LJF</td>
<td>Weakly</td>
<td>Never</td>
</tr>
<tr>
<td>SRPT</td>
<td>Weakly</td>
<td>Sometimes</td>
</tr>
</tbody>
</table>

*Table 11.1*: Comparing the Twin and CRTC Predictability Criteria.

Table 11.1 compares the classification of policies according to the two criteria. Note
that for continuous service distributions, our analysis has shown same results for preemptive and non-preemptive policies, which is not the case with CRTC.

Only policies for which analysis was provided in both works are listed.

For some policies the two criteria agree. For example PS is both Absolutely Twin Predictable and Always Predictable. LCFS is both Not Twin Predictable and Never Predictable. However, for some policies the criteria totally disagree. LRPT is Absolutely Twin Predictable, yet Never Predictable. P-LCFS is Not Twin Predictable, yet Always Predictable. LAS is Absolutely Twin Predictable and Sometimes Predictable, but this predictability is only in the case $E\{X^3\} = \infty$ (see Wierman \[152\]), so for most service distributions the criteria disagree.

This dissimilarity suggests that for guaranteeing predictability one might want to combine the two criteria, or require both.

11.5.3 Optimality under the Twin Measure

As several policies have zero twin measure, these policies can all be considered twin measure wise optimal. One might therefore want to find a policy that is both optimal in the twin measure sense, and has low sojourn time. Consider the following variant of SRPT, called SRPT$_\alpha$. Assume the job with the shortest remaining processing time has remaining processing time $x$. Service is given in a PS manner, to all jobs with remaining processing time not larger than $x + \alpha$. It is easy to see that if $\alpha$ can be arbitrarily small, this policy’s sojourn times are arbitrarily close to SRPT. On the other hand, if $\alpha \geq \epsilon + \delta$ (which since $\epsilon \to 0, \delta \to 0$ can be done for arbitrarily small $\alpha$), twin jobs will be served in a PS manner from $a_2$ until $\min(d_1, d_2)$, at which point one of them will leave the system and the other one will have at most $\epsilon + \delta$ service left. As $\epsilon \to 0, \delta \to 0$, this job will receive full service from that epoch onwards, and thus $Z(x, \delta, \epsilon) \leq \epsilon + \delta$ and $T^{SRPT\alpha}(x) = 0$. Thus SRPT$_\alpha$ has both optimal twin measure, and optimal sojourn time.
11.6 The Twin Measure in Multi-Server Systems

In this section we analyze the twin measure in some common multi-server settings. The scheduling policy between members of the same queue is FCFS.

11.6.1 Single Queue

We start our analysis with the simple single queue system, denoted SingleQueue, where the first job in the queue is served by the first server to become idle.

For simplicity we ignore effects caused by jobs arriving in the interval \((a_1, a_2)\) in this analysis. As the scheduling policy is FCFS, it is easy to see that these effects would be negligible as \(\epsilon \to 0\).

A simple observation is that the twin measure is smaller than \(x + \epsilon\), or \(x\) when \(\epsilon \to 0\). This is so because at the worse case (i) \(C_2\) will be served right after \(C_1\). However, there is a probability that either (ii) both twins arrive when two or more servers are idle, in which case both twins are served immediately, and leave the system simultaneously, or (iii) some server other than the one serving \(C_1\) will become idle while \(C_1\) is being served, in which case \(C_2\) will be served partially in parallel to \(C_1\), and leave system before \(d_2 + x\). Letting \(\epsilon \to 0\) and taking expectations we have

\[
T_{\text{SingleQueue}}(x) = (1 - \alpha) \left( \int_0^x \beta(y)ydy + \gamma(x)x \right),
\]

where \(\alpha\) is the probability of arriving when two or more servers are idle, \(\beta(y)\) is the probability that \(C_2\) will be delayed for \(y\) units of time until another server is idle, given that no more than one server was idle on arrival, and \(\gamma(x)\) is the probability that no other server will be idle until \(C_1\) is served, given that no more than one server was idle on arrival, in which case \(C_2\) is delayed for exactly \(x\) units of time.

For \(\alpha\) we have

\[
\alpha = \sum_{k=0}^{M-2} p_k,
\]
where $p_k$ is the probability of finding $k$ jobs in the system. For example, in the case of Exponential service requirements we have (9.2). In the general distribution case one can map the distribution to a PH distribution (see Remark 7.3) and use matrix analytic methods (e.g. Latouche and Ramaswami [90]) to obtain a good approximations of $p_k$.

For evaluating $\beta(y)$ one needs to look at the remaining service requirement on each of the servers at the epoch $s_1$. For $C_2$ to be delayed exactly $y$ units of time one server needs to have a remaining service requirement of $y$, and the other servers need to have a remaining service requirement larger than $y$. If one assumes that the remaining time is distributed the same way, or can be approximated by, the remaining service requirement upon arrival, then due to the Poisson arrivals it is the residual life of the service requirement, and has a pdf $\hat{b}(x) = (1 - B(x))/\bar{x}$ and a cdf $\hat{B}(x) = \int_0^x \hat{b}(t)dt$. If one further assumes that the remaining service requirements on the servers are independent, then the probability of one specific server having a residual lifetime of $y$ and the others having a residual lifetime larger than $y$ is $\hat{b}(y)(1 - \hat{B}(y))^{M-2}$, and finally $\beta(y) = (M - 1)\hat{b}(y)(1 - \hat{B}(y))^{M-2}$. Note that both assumptions are correct if service times are Exponential. In other cases, the evaluation is more difficult.

For evaluation $\gamma(x)$, using the same assumptions and the same arguments, $\gamma(x) = (1 - \hat{B}(x))^{M-1}$. To summarize, under the above assumptions

\[
T^{Single\ Queue}(x) = \left(1 - \sum_{k=0}^{M-2} p_k\right) \times \left(\int_0^x (M - 1)\hat{b}(y)(1 - \hat{B}(y))^{M-2}ydy + (1 - \hat{B}(x))^{M-1}x\right).
\]

One can observe that the twin measure is decreasing with $M$, as both $\alpha$ and $\beta(y)$ increase with $M$.

11.6.2 Multiple Queues

In this section we make some observation about the multiple queue system, where each queue is assigned one server, and that server serves the jobs in that queue in FCFS manner.
If a job joins the system and finds an empty queue it joins that queue and is served immediately. Otherwise, the job is assigned a queue using some queue assignment policy. Once a job is assigned a queue, it cannot jockey to another queue, even if the other queue’s server is idle. We denote this setting `MultipleQueue`.

Note that analysis is dependent on the queue assignment policy. However, for all queue assignment policies there are three possible cases: (i) Two or more queues are empty upon arrival of the twins. In this case both twins are served in parallel and leave the system simultaneously. (ii) Both twins join the same queue. In this case $C_2$ departs $x$ units after $C_1$, and (iii) the twins join different queues. In this case the twins will depart $|y|$ units of time apart, where $y$ is the difference in the remaining work in the two queues, which can be negative.

Using very similar notation to (11.5)

$$\tau_{MultipleQueue}(x) = (1 - \alpha) \left( \int_0^x \beta(y)|y|dy + \gamma(x)x \right),$$

where $\alpha$ is the probability of arriving when two or more queues are empty, $\beta(y)$ is the probability that the difference in the remaining work in the two queues $C_1$ and $C_2$ joined is $y$, given that no more than one queue was empty, and $\gamma(x)$ is the probability that the twins will join the same queue, given that no more than one queue was empty.

Evaluating $\alpha$, $\beta(y)$ and $\gamma(x)$ is much more complicated in this case than it was for `SingleQueue`. We state some methods for this evaluation below. One can also resort to simulation methods, which are in fact not very complicated.

Evaluating $\alpha$, after the general distribution is mapped to a PH distribution, involves a Markovian-chain which is infinite in more than one dimension. In these cases matrix analytic methods do not work and one needs to use other methods, such as the Dimensionality Reduction method proposed in Osogami [104], and see discussion there of other methods. This provides us with full state probabilities.

Analyzing $\beta(y)$ requires knowledge of the distribution of remaining work in a queue. This is in general much more complicated than the residual life of a single job, although
it is possible, using the queue length distribution, which is obtainable from the state probabilities.

\( \gamma(x) \) might be simple or complicated, depending on the job assignment policy. For example if the jobs are assigned to queues in random, \( \gamma(x) = 1/M \). If jobs are assigned to the shortest queue, the queue length distribution can be used.

One queue assignment policy of interest is to assign jobs to the queue with the least remaining work. Although this joining policy might not always be practical, it is possible in some computing systems where the length of each job is predetermined. Note that in this case \( C_1 \) always departs first, and \( C_2 \) always departs \( x \) or less units of time after \( C_1 \). One can therefore easily observe that the twin measure is smaller than \( x \). In general, this is not guaranteed for other queue assignment policies. We conjecture that this queue assignment policy is the most predictable.

### 11.7 Extending the Twin Measure

In this section we discuss two ways to extend the twin measure: using more than two jobs, which we call *trains*, and not sending the jobs simultaneously.

#### 11.7.1 Job Trains

One way to extend the results of the twin measure is to consider *Job Trains*, i.e. situations where more than two identical jobs are sent.

*Packet Trains* were proposed as means for measuring link bandwidth and available bandwidth (e.g. Carter and Crovella [30], Hu and Steenkiste [67], Jain and Dovrolis [72]). In these, packet trains are injected into the network. The dispersion of the probe packets at the receiver side is then used in different bandwidth estimation algorithms, using, for example, dispersion mean values or dispersion variance. However, this entire body of work assumes that packets are served using the *FCFS* policy, as indeed is the case with packet routers, at least for packets of the same flow. However, job schedulers may choose different policies, with quite different results.
We provide here only the simplified definition (parallel to Definition 11.2), as the nonsimplified one is much more difficult to define rigorously and provides little benefit.

**Definition 11.5 (Job Train Measure).** Let $C_1, C_2, \ldots, C_m$ be $m$ identical jobs, with equal service requirements $x$, arriving at epochs $a_1, a_2 = a_1 + \epsilon, a_3 = a_1 + 2\epsilon, \ldots, a_m = a_1 + (m - 1)\epsilon$ where $\epsilon > 0$, and departing at epochs $d_1, d_2, \ldots, d_m$. Assume that $C_1$ arrives when ($a_1$) the system is in steady state. Define the random variable $Z(x, \epsilon, m) = \max_i d_i - \min_i d_i$, given $x, m$ and $\epsilon$. Let $z^n(x, \epsilon, m)$ be the $n$-th moment of $Z(x, \epsilon, m)$, i.e. $z^n(x, \epsilon, m) = \mathbb{E}\{Z(x, \epsilon, m)^n\}$.

For scheduling policy $\phi$ and job size $x$, the $n$-th job train measure $T_n^\phi(x, m)$ is defined as the limit, when $\epsilon$ tends to zero, of $z^n(x, \epsilon, m)$, assuming a limit exists. Namely $T_n^\phi(x, m) = \lim_{\epsilon \to 0} z^n(x, \epsilon, m)$.

The shortened term job train measure, denoted $T^\phi(x, m)$, is used to describe the first job train measure, namely $T^\phi(x, m) = T_1^\phi(x) = \lim_{\epsilon \to 0} \mathbb{E}\{Z(x, \epsilon, m)\}$.

Analysis of the job train measure for the policies analyzed in Section 11.4 is quite straightforward. In fact, for all the policies $T^\phi(x, m) = (m - 1)T^\phi(x)$. However, this isn’t always the case.

Consider for example a synchronous server setting. All jobs are of size $x$ and the server works in service cycles of length $2x$ which serve either one or two customers, depending on availability of jobs. Each job receives a service rate of $1/2$. If $p_{odd}$ is the probability of finding an odd number of customers in the system upon arrival, then the job train measure is $p_{odd}2x[(m - 1)/2] + (1 - p_{odd})2x[(m - 2)/2]$ which is definitely not linear with $m$. For example, for $m = 2$ we get $p_{odd}2x$, for $m = 3$ we get $2x$.

A second example is a round robin policy, where jobs join the queue at a random location. One can observe that all the jobs in the job train will be served in the same service cycle, so the job train measure is at most as large as the service cycle length. This will grow with $m$, but not linearity.
11.7.2 Non-Simultaneous Twins

A second way to extend the results of the twin measure is to consider twins which do not arrive at the system concurrently. For example, the twins can arrive exactly $x$ units apart, i.e. $a_2 - a_1 = x$. This might provide more insight on the predictability of the system. For example, note that only for LRPT the measure is still zero.

Another case of specific interest is the case where the second twin enters the system when the first one departs, i.e. $a_2 = d_1$. For example this setting can represent a customer refreshing a Web page just as it finished loading, expecting a similar load time. It might be more interesting to measure the difference or ratio between the sojourn times, i.e. $|d_1 - a_1 - (d_2 - a_2)|$ or $\max(d_1 - a_1, d_2 - a_2)/\min(d_1 - a_1, d_2 - a_2)$.
Chapter 12

WORK IN PROGRESS AND FUTURE RESEARCH SUBJECTS

In this chapter we mention some areas in which work is still in progress. We also list open problems and suggestions for future research.

12.1 Online Algorithms and Competitive Analysis

One area in which research is ongoing is that of competitive analysis and online algorithms for non-preemptive policies, i.e. the class $\Phi$. In this area we compare online service policies, i.e. policies which make decisions only based of knowledge of past events, to an hypothetical optimal off-line policy $\text{OPT}$ which makes decisions based on full knowledge of future arrivals and service requirements.

Definition 12.1 (Competitive Ratio). We say an online service policy is $x$-competitive, if for every sample path $F_{D^2}^\phi \leq x F_{D^2}^{\text{OPT}}$.

While $\text{PS}$ is optimal (Theorem 5.3), it is a preemptive policy, which might not be implementable in many cases. More so, we believe that in order to execute an exact $\text{PS}$ imitation a scheduler must know all the exact service requirements and arrival epochs of all the customers ahead of time (Remark 5.5).

For identical job sizes it was shown that $\text{FCFS}$ is 1-competitive (optimal) for both the single server case (Remark 5.1) and multiple server case (Theorem 9.1). However, for arbitrary job sizes $\text{FCFS}$ is optimal only in expectation, only in the single server case, and only when job sizes are i.i.d. (Theorem 5.1).
Note that job sizes must be known to both $\text{OPT}$ and $\phi$. If job sizes are only known to the $\text{OPT}$, a simple case where two jobs of sizes $\alpha$ and 1 arrive together is enough to show an unbounded competitive ratio of $\alpha$.

Also note that in order to compare algorithms competitively we must assume idling is not allowed, since RAQFM does not penalize for idling, and therefore any algorithm can idle indefinitely and act only when all information is available. Penalizing for idling does not seem a promising route as well, since every online algorithm will always have to make a decision at some point in time, or suffer infinite penalty. Thus, for every online algorithm we can look at sample paths where events happen just after the decision to stop idling was made.

**Theorem 12.1.** A lower bound on the competitive ratio of online algorithms is the following: No online policy in $\Phi$ has a multiplicative competitive ratio better than 1.1.

**Proof.** Observe the following two arrival and service patterns: $\pi_1 = \{(0, 1), (\epsilon, b), (1 - \epsilon, c)\}$ and $\pi_2 = \{(0, 1), (\epsilon, b), (1 - \epsilon, c), (1 + \epsilon, \epsilon)\}$ where $\epsilon \to 0$. Note that there is a decision point at $t = 1$, where the policy needs to choose between $C_2$ and $C_3$. As the scenarios are indistinguishable up to this point an online policy must make the same decision in both scenarios.

For $\pi_1$ choosing to serve $C_2$ first yields an unfairness of $(b^2 - b + 1)/2$ while choosing to serve $C_3$ first yields an unfairness of $(c^2 + c + 1)/2$.

For $\pi_2$ choosing to serve $C_2$ first yields an unfairness of $(4b^2 - 4b + 3)/6$ while choosing to serve $C_3$ first yields an unfairness of $(4c^2 + 2c + 3)/6$. Note that in both cases it is better to serve $C_4$ as soon as possible.

A policy serving $C_2$ first therefore has a competitive ratio of

$$\max\left(\frac{b^2 - b + 1}{c^2 + c + 1}, \frac{4b^2 - 4b + 3}{4c^2 + 2c + 3}\right),$$

while policy serving $C_3$ first has a competitive ratio of

$$\max\left(\frac{c^2 + c + 1}{b^2 - b + 1}, \frac{4c^2 + 2c + 3}{4b^2 - 4b + 3}\right).$$
Choosing $c = 1, b = 1.90573$ yields

$$\frac{c^2 + c + 1}{b^2 - b + 1} = \frac{4b^2 - 4b + 3}{4c^2 + 2c + 3} = 1.10048.$$

Note that the mechanism of this proof takes advantage of the fact that RAQFM gives preference to both size and seniority. The policy must choose at $t = 1$ between serving $C_2$ first, thus preferring seniority or serving $C_3$ first, thus preferring size. A policy preferring seniority will do better in $\pi_1$, since the seniority difference is one unit, larger than the size difference. However, in $\pi_2$ another customer arrives, and it turns out that the original decision preferring the seniority of $C_2$ over the size of $C_3$ also mandates that it is preferred over the size of $C_4$, thus making it the wrong decision.

This lower bound can probably be improved. Exact optimization of $b$ and $c$ yields values only very slightly higher, but maybe other patterns will yield significantly higher values, especially in the multi-server case. However, note that with RAQFM increasing the number of customers usually does not yield larger multipliers because of the $1/N(t)$ factor.

Since accounting for future arrivals is in many cases impossible, we conjecture that a service policy which ignores future arrivals, and serves the jobs in an order that will optimize the fairness if no jobs arrive, will have a good competitive ratio. Finding how to implement this policy in less than exponential complexity is an ongoing research subject, as well as finding the exact competitive ratio of that policy. Other subjects include finding better lower bounds and the exact competitive ratio of other policies, for both the single server and the multiple server case. Random algorithms are also considered.

12.2 Tail Analysis

As there is much recent interest in the tail behavior of the response time (see e.g. Borst et al. [24, 23]), it is only natural to study the tail behavior of the discrimination as well.
Specifically we are interested in the tail behavior of the discrimination when the service
distribution has a heavy tail. This material is part of a paper in preparation Boxma et al. [25].

Theorem 5.6 indicates that the discrimination is bounded from above by the service
requirement, while from below it is only bounded by the waiting time. But are high dis-

Consider a non-negative random variable $X$ with distribution function $F(x) = \mathbb{P}\{X \leq x\}$, and with $\bar{F}(x) := 1 - F(x)$.

**Definition 12.2 (Long Tailed).** $F$ is long-tailed ($F \in \mathcal{L}$) if, for any $y > 0$ and $x \to \infty$,

$$\mathbb{P}\{X > x + y \mid X > x\} \to 1.$$ 

The convergence is uniform in $y$ on compact subintervals.

The following lemma is well-known:

**Lemma 12.1.** If $X \in \mathcal{L}$ and $Y$ is independent of $X$ and non-negative, then

$$\frac{\mathbb{P}\{X - Y > x\}}{\mathbb{P}\{X > x\}} \to 1.$$ 

**Definition 12.3 (Regularly Varying).** $X$ is called regularly varying of index $\alpha \geq 0$ ($X \in \mathcal{R}_\alpha$) if $\mathbb{P}\{X > x\} = L(x)x^{-\alpha}$, with $L(\cdot)$ a slowly varying function, i.e., $L(cx)/L(x) \to 1$

Refer to Sigman [140] for a brief overview of these classes and their application in queue-
ing theory. For extensive expositions, refer to the books Bingham et al. [18], Embrechts

It can be shown that for every $\alpha \geq 0$, $\mathcal{R}_\alpha \subset \mathcal{L}$ (cf. Sigman [140]).

**Definition 12.4.** $\mathcal{L}^*$ is the subclass of distribution functions contained in $\mathcal{L}$ with the property that if $X \in \mathcal{L}^*$ then also $X - \ln X \in \mathcal{L}^*$. 
Remark 12.1. The following example shows that \( \mathcal{L} \neq \mathcal{L}^* \): 
\[
\bar{F}(x) = \exp\left(-\frac{x}{\sqrt{\ln \ln x}}\right).
\]

Some more notation that we need for this section: for every two real functions \( g, h \) we use the notational convention \( g(x) \sim h(x) \) to denote \( \lim_{x \to \infty} g(x)/h(x) = 1 \). \( g(x) \geq h(x) \) denotes \( \lim_{x \to \infty} g(x)/h(x) \geq 1 \).

We use \( B \) to denote the customer’s service requirement.

**Theorem 12.2** (Positive Tail for Non-Preemptive Service Policies). Assume that \( B \in \mathcal{L}^* \), and that \( \mathbb{P}\{B - \ln B > x\} \sim \mathbb{P}\{B > x\} \). For every service policy in \( \Phi \),

\[
\mathbb{P}\{D > x\} \sim \mathbb{P}\{B > x\}.
\]

**Proof.** From (4.1),

\[
\mathbb{P}\{D > x\} = \mathbb{P}\{B - \int_a^d \frac{1}{N(t)} dt > x\} = \mathbb{P}\{B - \int_a^r \frac{1}{N(t)} dt - \int_r^d \frac{1}{N(t)} dt > x\},
\]

where \( r \) is the epoch in which the customer’s service begins. So obviously

\[
\mathbb{P}\{D > x\} \leq \mathbb{P}\{B > x\}.
\]

It remains to prove that \( \mathbb{P}\{D > x\} \geq \mathbb{P}\{B > x\} \). Observe that, for any policy in \( \Phi \),

\[
\int_a^r \frac{1}{N(t)} dt \leq r - a = W,
\]

where \( W \) denotes the waiting time, and notice that this term is independent of \( B \). Further observe that, for large \( x \),

\[
\mathbb{P}\{B - \int_a^r \frac{1}{N(t)} dt - \int_r^d \frac{1}{N(t)} dt > x\} \geq \mathbb{P}\{B - W - \int_0^B \frac{1}{1 + (\lambda - \epsilon)t} dt > x\}
\]

\[
= \mathbb{P}\{B - \frac{\ln(1 + (\lambda - \epsilon)B)}{\lambda - \epsilon} - W > x\}.
\]

The inequality is due to the fact that in the service period \( N(t) \geq 1 + (\lambda - \epsilon)t \) with high probability for every \( 0 < \epsilon < \lambda \), due to the law of large numbers. The last step follows using Definition [12.4].

The result follows by combining the fact that \( W \) and \( B \) are independent with [12.2] and Lemma [12.1].
Corollary 12.1. For every service policy in $\Phi$,

$$B \in \mathcal{R}_\alpha \Rightarrow D \in \mathcal{R}_\alpha.$$  

The proof is immediate from Theorem 12.2 and the fact that $\mathbb{P}\{B \geq x\} \sim \mathbb{P}\{B \geq x - \ln x\}$ for $B \in \mathcal{R}_\alpha$.

Ongoing research focuses first on the negative tail of the discrimination. First indication show that it might only be the size of the logarithm of the service distribution. Compare to similar results in Brosh et al. [27], which we also mention in Section 8.4. Other area of research focuses on the positive and negative tail of the discrimination in other setups such as the Discriminatory Processor Sharing (DPS) policy, see e.g. Rege and Sengupta [124].

12.3 Future Research

Following is a list of open problems and future areas for research. Many of these were mentioned throughout this work as open issues and conjectures.

1. Studying fairness in networks of queues, e.g. tandem queues and other more complex constructs. A measure is proposed in Section 4.3.

2. Finding a lower bound for the unfairness under $\Phi$ and $\Phi^*$ (Remark 5.6).

3. Verifying that in order to execute PS imitation a scheduler must know all the exact service requirements and arrival epochs of the customers ahead of time (Remark 5.5).

4. Check if the class of performance metrics $\xi$ can be made more general, making the statements in Section 6.3 stronger (Section 6.3).

5. Finding an efficient way to explicitly evaluate the intra-variance of common metrics, e.g. the waiting time (Section 6.4).
Chapter 12. Work in Progress and Future Research Subjects

6. Showing that when a large number of customers in encountered upon arrival, \( P-R\)ros behaves like \( PS \) even more than \( ROS \) (Figure 8.4 observation 2).

7. Finding the reason for the decrease in unfairness at high loads for single queue multiple server systems (Figure 9.4, observation 1).

8. Proving, or disproving, that jockeying is beneficial (Conjecture 9.1).

9. Finding necessary conditions under which the properties in Chapter 9 hold for the G/GI/m model or the M/GI/m one (Remark 9.4).

10. Analyzing arbitrary jockeying, when servers aren’t idle, or from an arbitrary position (Section 9.1.2).

11. Weighted Fairness, e.g. for systems where classes have different economic values (mentioned in the introduction to Chapter 10).

12. Proving that in the M/GI/1 case with two classes, where each is assigned one server, there is dominance in discrimination to the class with lower load (Conjecture 10.1).

13. Analysis of fairness in polling systems, include exhaustive, gated, \( k \)-limited and more, see Takagi [144], Levy and Sidi [92].

14. Finding if splitting servers while keeping the same service rate is always more fair (Section 10.4.3).

15. Analyzing in detail the twin measure for job trains, and its relation to other measures done through job trains (Section 11.7.1).

16. Analyzing in detail the twin measure for non simultaneous twins. Of specific interest is the case where the second twin enters the system when the first one departs (Section 11.7.2).
BIBLIOGRAPHY


[124] K. M. Rege and B. Sengupta. A decomposition theorem and related results for the


Appendices
Appendix A

STRONGER PROOF FOR REACTION TO DIFFERENCES IN SENIORITY

Theorem A.1 (Preference of Seniority Between Customers with Stochastically Equal Service Requirements). Let $a_1, a_2, \ldots, a_L$ be an arbitrary (deterministic) arrival pattern where $a_j < a_k$, $1 \leq j, k \leq L$. Let $\{s_i\}$, $i = 1, 2, \ldots, L$, $i \neq j, k$ be an arbitrary (deterministic) set of service requirements. Let $S_j$ and $S_k$, the service requirements of $C_j$ and $C_k$, respectively, be i.i.d. random variables, with realizations $s_j$ and $s_k$ respectively. Let $\phi \in \Phi$ be a service policy that serves $C_k$ ahead of $C_j$. Then there exists a service policy $\phi' \in \Phi$ such that $\phi'$ has lower expected overall unfairness than $\phi$, where expectation is taken over the joint distribution of $S_j$ and $S_k$, and $\phi'$ serves $C_j$ ahead of $C_k$.

Proof. First we construct $\phi'$ from $\phi$: Assume that $S_j = \tau_1$, $S_k = \tau_2$. Let $\pi$ denote this sample path and $\pi'$ the same path where $S_j = \tau_2$, $S_k = \tau_1$. Let $t$ be the epoch where $\phi$ starts serving $C_k$. Then up to time $t$ $\phi'$ is identical to $\phi$. At $t$ it chooses to serve $C_j$ ($S_j = \tau_1$). From $t + \tau_1$ and on $\phi'$ imitates what $\phi$ would do under $\pi'$, with the exception that when $\phi$ serves $C_j$ $\phi'$ serves $C_k$ (see Figure A.1(b) and (d)). Since $\phi \in \Phi$ and $\phi'$ imitates it, $\phi' \in \Phi$.

Next we compare the unfairness of $\phi$ and $\phi'$. For $s_j = s_k$, $\phi'$ serves all customers in the same order as $\phi$, except that it interchanges $C_j$ and $C_k$. From Remark 5.1 it follows that the unfairness of $\phi'$ is lower in this sample path.

For $s_j \neq s_k$ we need to track two sample paths for each policy: For every pair of requirements $\tau_1, \tau_2$ (w.l.o.g. assume $\tau_1 < \tau_2$) we consider one path where $S_j = \tau_1$ and $S_k = \tau_2$ and one path where $S_j = \tau_2$ and $S_k = \tau_1$ and apply both $\phi$ and $\phi'$ to both. To clarify this
observe Figure A.1 (a) and (b) show how $\phi$ serves the customers when $S_j = \tau_1, S_k = \tau_2$ and $S_j = \tau_2, S_k = \tau_1$, respectively. (d) shows how $\phi'$ serves the customers when $S_j = \tau_1, S_k = \tau_2$ which is the order in which $\phi$ serves them for $S_j = \tau_2, S_k = \tau_1$, interchanging $S_j$ and $S_k$, and likewise (c) shows $\phi'$ for $S_j = \tau_2, S_k = \tau_1$. Let $F^{(a)}, F^{(b)}, F^{(c)}, F^{(d)}$ denote

![Diagram](image)

**Figure A.1:** Two Customers With Stochastically Identical Service Requirements

the unfairness in scenarios (a), (b), (c), (d) respectively. Note that the probability density of $S_j = \tau_1, S_k = \tau_2$ is equal to that of $S_j = \tau_2, S_k = \tau_1$, since $S_j$ and $S_k$ are i.i.d. We can therefore show that the expected overall unfairness under $\phi$ is larger than the one under $\phi'$, by showing that $F^{(a)} + F^{(b)} - F^{(c)} - F^{(d)} > 0$.

Let $F^{(a)}$ denote the total unfairness under (a) due to $C_j$ and $C_k$, and $F^{(a)}$ denote the total unfairness under (a) due to all customers other than $C_j$ and $C_k$. Then $F^{(a)} = F^{(a)} + F^{(a)}$. We define similar notations for (b), (c) and (d). Note that for any customer other than $C_j$ and $C_k$, (a) and (c) are identical, and so are (b) and (d). Therefore $F^{(a)} = F^{(c)}$ and $F^{(b)} = F^{(d)}$. Thus

$$F^{(a)} + F^{(b)} - F^{(c)} - F^{(d)} = F^{(a)} + F^{(a)} + F^{(b)} + F^{(b)} - F^{(c)} - F^{(d)} - F^{(d)}$$

$$= F^{(a)} + F^{(b)} - F^{(c)} - F^{(d)}.$$


For (a) and (c) we divide the time interval between $a_j$ and $\max(d_j, d_k)$ (namely until both $C_j$ and $C_k$ depart) into five sub-intervals $(E_i, E_{i+1})$, $i = 1, 2, 3, 4, 5$, where 1) $E_1 = a_j$, 2) $E_2 = a_k$, 3) $E_3$ is the first point in time where service, to either $C_j$ or $C_k$, starts, 4) $E_4 = E_3 + \tau_1$, 5) $E_5 = E_3 + \tau_2$, and 6) $E_6 = \max(d_j, d_k)$. Let $R_{(i)}$ be the warranted service accumulated in sub-interval $i$, under (a) and (c).

Moving on to (b) and (d), note first that $E_1, E_2$ and $E_3$ are the same as in (a) and (c) since they are all prior to the first service start (of $C_j$ or $C_k$). Also, $E_4$ is the same as in (a) and (c) since $E_4 = E_3 + \tau_1$. Note also that the number of customers in (b) and (d) in these intervals is also identical to the number of customers in (a) and (c), and thus so is the warranted service. We define $E'_5 = \max(d_j, d_k)$, for $\phi$ when $S_j = \tau_2, S_k = \tau_1$, which can be different from $E_6$. We define the sub-interval $4'$ which is $(E_4, E_5')$ and $R_{(4')}$, the warranted service in the sub-interval $4'$. We now have

$$F^{(a)} + F^{(b)} - F^{(c)} - F^{(d)} = \frac{1}{L} \left[ (\tau_1 - (R_{(1)} + R_{(2)} + R_{(3)} + R_{(4)} + R_{(5)}))^2 + (\tau_2 - (R_{(2)} + R_{(3)} + R_{(4)}))^2 + (\tau_1 - (R_{(2)} + R_{(3)}))^2 - (\tau_2 - (R_{(1)} + R_{(2)} + R_{(3)})^2 - (\tau_1 - (R_{(2)} + R_{(3)} + R_{(4)})^2) - (\tau_1 - (R_{(1)} + R_{(2)} + R_{(3)})^2 - (\tau_2 - (R_{(2)} + R_{(3)})^2 \right]$$

$$= \frac{2}{L} R_{(1)} (R_{(5)} + R_{(4')}) > 0.$$
Appendix B

PROOF OF THE LOCALITY OF REFERENCE THEOREM

In this appendix we provide the Locality of Reference theorem and its proof in details.

Definition B.1 (Resource Allocation Schedule). A Resource Allocation Schedule (RAS) is a functional way to represent a policy, by observing the service policy’s result. Specifically it is a function \( \varphi : [0, \infty) \rightarrow \mathbb{N}^0 \) that for each epoch \( t \in [0, \infty) \) returns the index \( \varphi(t) \) of the job served at that epoch or 0 if the server is idle.

Remark B.1 (Processor Sharing Policies). Note that while the definition of RAS limits our model to serving one job at each epoch, it can still be used to model any processor sharing type system, by using infinitesimally small intervals.

For a given pattern we will say that RAS \( \varphi \) belongs to \( \Phi_1 \) if it satisfies the following requirements:

1. Arrival Validity: \( \forall t, \varphi(t) = k \Rightarrow t \geq a_k \) i.e. no job is given service before its arrival.
2. Work Conserving: \( \forall k, \int_{\varphi(t)=k} dt = s_k \) i.e. no job is served more (or less) than its service requirement.
3. Non-Idling: \( \forall t, \varphi(t) = 0 \Rightarrow \forall k, a_k < t: \int_{\varphi(u)=k}^{t} du = s_k \) i.e., the server does not idle at \( t \) if there is a job in the system that still requires service.

Note that requirement 3 implies that for each \( C_k \) there exists \( \epsilon > 0 \) such that

\[
\forall t \in (d_k - \epsilon, d_k): \varphi(t) = k. \tag{B.1}
\]
Definition B.2 (Swappable). Consider RAS \( \varphi_a \) and RAS \( \varphi_b \). We say that \( \varphi_a \) and \( \varphi_b \) are 1-swappable for a pair of jobs \( \langle C_i, C_j \rangle \) if:

\[
\exists t_1, t_2, \varepsilon: (\forall t \in (t_1, t_1 + \varepsilon): \varphi_a(t) = i, \varphi_b(t) = j),
\]

\[
(\forall t \in (t_2, t_2 + \varepsilon): \varphi_a(t) = j, \varphi_b(t) = i),
\]

\[
(\forall t \notin (t_2 + \varepsilon) \cup (t_1, t_1 + \varepsilon): \varphi_a(t) = \varphi_b(t)).
\]

In other words, the RASs are identical except for a shift of resources between \( \langle C_i, C_j \rangle \) at the segments \( (t_1, t_1 + \varepsilon) \) and \( (t_2, t_2 + \varepsilon) \).

We recursively define \( \varphi_a \) and \( \varphi_b \) to be \( n \)-swappable for \( \langle C_i, C_j \rangle \) if there exists a RAS \( \varphi_c \) and a job \( C_k \) such that \( \varphi_a \) and \( \varphi_c \) are \( (n - 1) \)-swappable for \( \langle C_i, C_k \rangle \) and \( \varphi_c \) and \( \varphi_b \) are 1-swappable for \( \langle C_k, C_j \rangle \). We say that \( \varphi_a \) and \( \varphi_b \) are resource-swappable for \( \langle C_i, C_j \rangle \) if there exists a value \( n \in \mathbb{N} \) such that they are \( n \)-swappable for \( \langle C_i, C_j \rangle \).

Lemma B.1. For every job \( C_i \) that is not the first job in a busy period, there exists another job \( C_j \) such that \( a_j < a_i, d_j > a_i \).

Proof. Assume there is no such job. Then every job arriving before \( C_i \) also leaves before \( C_i \) arrives, and that makes \( C_i \) the first in a busy period, in contradiction to the assumption. \( \square \)

Theorem B.1 (Locality of Reference). Let \( \{(a_i, s_i)\}_{i=1,2,\ldots} \) be an arbitrary arrival and service pattern, let \( \varphi_a \) be an arbitrary RAS that belongs to \( \Phi_1 \) for that pattern, and let \( \langle C_i, C_j \rangle \) be an arbitrary pair of jobs. Then, \( C_i \) and \( C_j \) are in the same busy period if and only if there exists a RAS \( \varphi_b \) that belongs to \( \Phi_1 \) for that pattern, and \( \varphi_a \) and \( \varphi_b \) are resource-swappable for \( \langle C_i, C_j \rangle \).

What this theorem means is in essence the same as Theorem 6.1: resources can be shifted from \( C_i \) to \( C_j \) if and only if \( C_i \) and \( C_j \) are in the same busy period.

Proof. i) Assume that \( C_i \) and \( C_j \) are in the same busy period and prove the existence of RAS \( \varphi_b \). We denote the departure epoch of \( C_k \) under \( \varphi_a \) and \( \varphi_b \) as \( d_k^a \) and \( d_k^b \) respectively.
Appendix B. Proof of the Locality of Reference Theorem

Assume $d_i^a < d_j^a$. We will prove the claim by induction over $a_j - d_i^a$.

**Induction base:** $a_j - d_i^a < 0$. Then according to (B.1) there exists $\epsilon_1$ such that $\forall t \in (d_i^a - \epsilon_1, d_i^a): \varphi(t) = i$ and there exists $\epsilon_2$ such that $\forall t \in (d_j^a - \epsilon_2, d_j^a): \varphi(t) = j$. We define $\epsilon = \min\{\epsilon_1, \epsilon_2\}$ and construct $\varphi_b$ as follows:

$$
\varphi_b(t) = \begin{cases} 
  j & t \in (d_i^a - \epsilon, d_i^a) \\
  i & t \in (d_j^a - \epsilon, d_j^a) \\
  \varphi_a(t) & \text{otherwise}
\end{cases}
$$

It is easy to see that $\varphi_a$ and $\varphi_b$ are 1-swappable for $\langle C_i, C_j \rangle$ at the segments $(d_i^a - \epsilon, d_i^a)$ and $(d_j^a - \epsilon, d_j^a)$.

**Induction step:** we assume that if $a_j - d_i^a < l$ there exists $\varphi_b$ such that $\varphi_a$ and $\varphi_b$ are resource-swappable for $(C_i, C_j)$ and show that if $a_j - d_i^a = l$ there also exists such a RAS.

Assume $a_j - d_i^a = l$. We choose $C_m$ such that $a_m < a_j, d_m > a_j$, which must exist according to Lemma B.1. As $a_m < a_j$ we have $a_m - d_i^a < l$ and according to the induction assumption there exists a RAS, say $\varphi_c$, such that $\varphi_a$ and $\varphi_c$ are resource-swappable for the pair $(C_m, C_j)$.

We now show that there exists a RAS $\varphi_b$ such that $\varphi_c$ and $\varphi_b$ are 1-swappable for $(C_m, C_j)$, and conclude that $\varphi_a$ and $\varphi_b$ are resource-swappable.

Let $\tau$ be the first epoch in which $C_j$ is served according to $\varphi_c$ and assume it is served for a period of $\epsilon_1$. According to (B.1) there exists $\epsilon_2$ such that $\forall t \in (d_m^c - \epsilon_2, d_m^c): \varphi(t) = m$. We define $\epsilon = \min\{\epsilon_1, \epsilon_2\}$ and construct as follows:

$$
\varphi_b(t) = \begin{cases} 
  j & t \in (d_m^c - \epsilon, d_m^c) \\
  m & t \in (\tau, \tau + \epsilon) \\
  \varphi_c(t) & \text{otherwise}
\end{cases}
$$

It is easy to see that $\varphi_b$ and $\varphi_c$ are 1-swappable for $\langle C_m, C_j \rangle$ at the segments $(d_m^c - \epsilon, d_m^c)$ and $(\tau, \tau + \epsilon)$.

The proof for the case $d_i^a > d_j^a$ is similar.
Appendix B. Proof of the Locality of Reference Theorem

Remark B.2 (Validity of the Induction). While the induction is defined on a continuous space (real numbers) it does not require infinite number of steps. This can be shown to be true as the choice of \( J_m \) requires \( a_m < a_j \). This can be done at most a number of times equal to the number of customers in the system. Furthermore, let \( \epsilon_a \) be the minimum inter arrival interval length in the busy period. Then a sufficient assumption for the induction step is that if \( a_j - d_i^a \leq l - \epsilon_a \) there exists \( \varphi_b \) such that \( \varphi_a \) and \( \varphi_b \) are resource-swappable for \( \langle C_i, C_j \rangle \), thus each step reduces \( l \) by at least \( \epsilon_a \).

This concludes the proof of the first direction.

ii) To prove the other direction we assume \( C_i \) and \( C_j \) are not in the same busy period and show that there cannot exist \( \varphi_b \) such that \( \varphi_a \) and \( \varphi_b \) are resource-swappable for \( \langle C_i, C_j \rangle \). Assume that \( \varphi_b \) as described above exists, and \( \varphi_a \) and \( \varphi_b \) are \( n \)-swappable for \( \langle C_i, C_j \rangle \). Then there exists a series of \( n \) pairs (\( \langle C_i, C_1 \rangle, \langle C_1, C_2 \rangle, \ldots, \langle C_{n-2}, C_{n-1} \rangle, \langle C_{n-1}, C_j \rangle \)) such that \( \varphi_a \) and \( \varphi_b \) are 1-swappable for each pair. As \( C_i \) and \( C_j \) are not in the same busy period, there is at least one pair in this series, say \( \langle C_l, C_m \rangle \), such that \( C_l \) belongs to the same busy period of \( C_i \), and \( C_m \) does not belong to that busy period. However since \( C_i \) and \( C_m \) do not belong to the same busy period either \( d_l < a_m \) or \( d_m < a_l \). In either case, \( \varphi_a \) and \( \varphi_b \) are not 1-swappable for \( \langle C_l, C_m \rangle \). \( \square \)
Appendix C

ANALYSIS OF CONDITIONAL DISCRIMINATION AND UNFAIRNESS IN AN M/E_R/1 SYSTEM UNDER FCFS

The analysis below is a somewhat amended version of the analysis provided in Brosh et al. [27]. It is based on the method in Section 7.1.

The service distribution is Erlang with r i.i.d. stages 1, . . . , r, each exponentially distributed with parameter rµ. Note that the time slots are defined by arrivals and stage completions, which serve in the same way as departures did in Section 7.1.

(1) The state definition is as follows: a ∈ N^0 denotes the number of customers ahead of C, b ∈ N^0 the number of customers behind C, and j = 1, . . . , r the service stage of the served customer.

(2) N(a, b, j) = a + b + 1, s(a, b, j) = I(a = 0), ω(a, b, j) = 1. The rates of arrival and stage completion are constant at µ(a, b, j) = rµ and λ(a, b, j) = λ.

The momentary discrimination at state (a, b, j), denoted δ(a, b, j), is

\[ δ(a, b, j) = I(a = 0) - \frac{1}{a + b + 1}, \]

and the moments of the slot length are,

\[ t^{(1)}(a, b, j) = \frac{1}{\lambda + r\mu}, \quad t^{(2)}(a, b, j) = \frac{2}{(\lambda + r\mu)^2} = 2(t^{(1)})^2, \quad a, b \geq 0. \]

Since they are state independent we can denote them \( t^{(1)} \) and \( t^{(2)} \).

(3) Assume C is in state (a, b, j) at slot i. At the slot end the system can encounter one of the following events:
1. A customer arrives into the system. The probability of this event is $\lambda \equiv \lambda/(\lambda + r\mu)$. Afterwards $C$’s state will change to $\langle a, b + 1, j \rangle$.

2. A customer completes a service stage. The probability of this event is $\tilde{\mu} \equiv r\mu/(\lambda + r\mu)$. If $j < r$ $C$’s state will change to $\langle a, b, j + 1 \rangle$, otherwise ($j = r$), if $C$ is not being served ($a > 0$) $C$’s state will change to $\langle a - 1, b, 1 \rangle$, otherwise ($j = r, a = 0$) $C$ will leave the system.

This leads to the following recursive expression:

$$d(a, b, j) = t^{(1)}(a, b, j) + \tilde{\lambda}d(a, b + 1, j) + \begin{cases} \tilde{\mu}d(a, b, j + 1) & j < r, \\ I(a > 0)\tilde{\mu}d(a - 1, b, 1) & j = r. \end{cases}$$

(4) Similarly, the equations for $d^{(2)}(a, b, j)$ are

$$d^{(2)}(a, b, j) = t^{(2)}(a, b, j)^2 + \tilde{\lambda}d^{(2)}(a, b + 1, j) + 2t^{(1)}(a, b, j)\tilde{\lambda}d(a, b + 1, j) +$$

$$\begin{cases} \tilde{\mu}d^{(2)}(a, b, j + 1)2t^{(1)}(a, b, j)\tilde{\mu}d(a, b, j + 1) & j < r, \\ I(a > 0)(\tilde{\mu}d^{(2)}(a - 1, b, 1)2t^{(1)}(a, b, j)\tilde{\mu}d(a - 1, b, 1)) & j = r. \end{cases}$$

(5) The states possible upon arrival are $\langle 0, 0, 1 \rangle$ if an empty system is encountered and $\langle k, 0, j \rangle$, $k = 1, \ldots, j = 1, \ldots, r$ otherwise, where $k$ is the number of customers seen on arrival and $j$ is the state in which the served customer resides. Using that

$$F_{D2} = (1 - \rho)d^{(2)}(0, 0, 1) + \sum_{k=1}^{\infty} \sum_{j=1}^{r} p_{k,j}d^{(2)}(k, 0, j),$$

where $p_{k,j}$ is the steady state probability of encountering $k$ customers in the system, where the served customer is at stage $j$. $p_{k,j}$ can be derived or computed using standard techniques for solving steady state balance equations.