Integral Inventory Control in Spare Parts Networks with Capacity Restrictions

Andrei Sleptchenko

2002

Ph.D. thesis
University of Twente

Twente University **Press**

Integral Inventory Control in Spare Parts Networks with Capacity Restrictions

Beta
Research School for Operations Management and Logistics

This thesis is number D-54 of the thesis series of the Beta Research School for Operations Management and Logistics. The Beta Research School is a joint effort of the departments of Technology Management, and Mathematics and Computing Science at the Technische Universiteit Eindhoven and the Centre for Production, Logistics and Operations Management at the University of Twente. Beta is the largest research centre in the Netherlands in the field of operations management in technology-intensive environments. The mission of Beta is to carry out fundamental and applied research on the analysis, design and control of operational processes.

Dissertation committee

Chairman / secretary	Prof. dr. W. van Rossum	Universiteit Twente
Promotor	Prof. dr. A. van Harten	Universiteit Twente
Assistant promotor	Dr. M.C. van der Heijden	Universiteit Twente
Members	Prof. dr. W.H.M. Zijm	Universiteit Twente
	Prof. dr. ir. J.H.A. de Smit	Universiteit Twente
	Prof. dr. K. Inderfurth	Otto-von-Guericke University Magdeburg
	Prof. dr. A.G. de Kok	Technische Universiteit Eindhoven
	Prof. dr. O.J.Boxma	Technische Universiteit Eindhoven
	Prof. dr. J. Telgen	Universiteit Twente

Twente University Press

Publisher: Twente University Press
P.O. Box 217, 7500AE Enschede, the Netherlands
www.tup.utwente.nl

Cover design: Alexandru Dina and Maria Nicola
Print: Océ Facility Services, Enschede

ISBN 9036518172

INTEGRAL INVENTORY CONTROL IN SPARE PARTS NETWORKS WITH CAPACITY RESTRICTIONS

PROEFSCHRIFT

ter verkrijging van
graad van doctor aan de Universiteit Twente,
op gezag van de rector magnificus,
prof. dr. F.A. van Vught,
volgens besluit van het College voor Promoties
in het openbaar te verdedigen
op vrijdag 22 november 2002 te 13.15 uur

door

Andrei Vasilievich Sleptchenko

geboren op 9 januari 1974
te Barabinsk, Rusland

Dit proefschrift is goedgekeurd door de promotor

Prof. dr. A. van Harten

en de assistent-promotor

Dr. M.C. van der Heijden

To my parents

Table of contents

Summary

This thesis deals with repair capacities and job priorities in repairable spare parts supply systems. Such supply systems can be used to support many different types of technically advanced systems, such as computer systems, medical equipment and military systems. The high price of some components and modules makes repair more profitable than scrap and replace. Upon system failure, it is important that spare parts are available, so that off-line repair is possible, thereby avoiding system down time. A key issue in such systems is to determine adequate spare part stock levels. This is a complex problem, because we generally have to take into account the technical system structure (should we replace modules containing a failed component or should we first disassemble and replace just a component level?) as well as the geographical distribution structure. Hence, we should decide at which level in the product structure spares are needed and in which amount we will store them at which location. Clearly, there is a trade-off between the system availability and spare part investment.

A large range of mathematical models to support such decisions has been developed since the sixties. Most models do not explicitly take into account repair capacities. As a consequence, it is hard to find out how the system performance is influenced by extending or reducing repair capacities. Besides, it is not possible to find out to which extend efficiency gain is possible by proper priority setting. A key idea when starting our research was that we can reduce inventory investment of extremely expensive spare parts by giving them high priority during repair, at the expense of higher stock levels of cheaper items, caused by longer repair throughput times. To find out to which extend this is true, we had to develop more detailed repair shop models than have been used up to now in the literature. When selecting a suitable queueing model for the repair shop, we found out that no suitable results were available in the literature. Therefore, a significant part of the research in this thesis deals with new queueing results that we needed to analyse spare part supply systems with finite repair capacities and repair job priorities. We used these results for three purposes: (1) to examine the impact of finite repair capacities on the spare part inventory optimisation, (2) to make a trade-off between spare part investment and repair capacity investment, and (3) to find rules for repair priority setting. To test and validate our new methods, we conducted extensive numerical results in each step of our research, heavily using discrete event simulation as

benchmark. A case from the Royal Netherlands Navy inspired us when setting up our numerical experiments.

The models of spare parts supply system described in this thesis are based on the well-known VARI-METRIC model by Sherbrooke. Therefore, after presenting our research design and reviewing the relevant literature in Chapter 1, we summarise in **Chapter 2** the key issues of VARI-METRIC and we introduce the main notation that we use within this thesis. Also, we motivate our choice to model repair shops by Markovian multi-class, multi-server queues, either with first-come, first serve discipline or with preemptive priorities.

As model for repair shops with finite capacity, we take a *multi-class, multi-server* queue in which repair jobs are processed in the order of arrival. The multi-class nature of this model means that various spare parts, each having their own interarrival and service time, are sharing the same repair capacity. In **Chapter 3**, we describe an exact solution of this model. Our procedure requires the solution of quadratic eigenvalue equations of high dimension, which grows exponentially with the number of classes and servers in the system. Therefore, we have also developed an approximation method that only requires solving a system of linear equations with a dimension equal to the number of classes. Numerical experiments show good results, especially for high utilisation rates. The queueing results that we obtain are more generally applicable, e.g. to analyse production, service and telecommunication, that are processing multiple job types.

In **Chapter 4**, we examine to which extend we can improve VARI-METRIC using our finite capacity repair shop model. We demonstrate that the classical VARI-METRIC model usually overestimates the system availability, whereas our method provides an estimate for the system availability that is much closer to reality. Also, we conclude that our models gives a better distribution of spare parts, especially if the repair shops only have a few servers and if the utilisation of repair shop is high.

It is not surprising that a high repair shop utilisation leads to high spare part stock levels. Therefore, we analyse added repair server capacities as decision variables in **Chapter 5**. This gives rise to an extension of VARI-METRIC. To this end, we present a nested optimisation procedure for spare part stock levels and repair capacities. We illustrate our optimisation heuristic with some numerical experiments. A remarkable finding is that the optimal distribution of budget between repair capacities and spare parts stocks seems to be almost independent of the price of repair capacity relatively to spare parts prices. If the price of repair capacity increases, we take less capacity and more stocks, but in the end the total budget spent to attain the target availability increases and a more or less fixed percentage of this budget is spent to repair capacity.

As mentioned before, the introduction of repair priorities can reduce throughput times of expensive items and, hence, the stock levels needed. To be able to use the repair priorities in the spare parts supply systems, we introduce in **Chapter 6** a multi-class, multi-server queueing model with preemptive priorities and two priority levels (high and low). Preemptive priority means that a high priority item may push a low priority item out of service if it finds all servers busy upon arrival while some of the servers are working on low priority items. The repair process of the postponed items is resumed if there is a free server and no high priority items are in the queue anymore. We present an iterative method to solve this priority queueing model. Our approach is exact, but we encounter an approximation error because we can only use a finite number of iterations. We have shown that our method performs quite well, even for a moderate number (~20) of iterations. As a side result, we show in **Chapter 7** how we can analyse multi-class, multi-server priority queueing models with preemptive and non-preemptive priority rules if high priority items are immediately outsourced if they do not find a free server upon arrival.

Finally, we reach our main research goal in **Chapter 8**, where we embedded our priority queueing model from Chapter 6 in the VARI-METRIC framework. We present the analysis of the spare parts supply system with finite amounts of repair servers and preemptive repair priorities at the repair facilities using the queueing models presented in Chapter 6. We present a few heuristics to assign priority to repairable modules and components sharing the same repair capacity, based on mean service times and item prices. We show that items with the highest price / service time ratio should usually receive high priority. We also show that it does not harm to separate priority assignment from spare part stock optimisation. An appropriate sequential procedure yields results that are only little worse than enumeration, whereas the computation requirements are much less. Therefore, we conclude that sequential optimisation is a practical approach. Further numerical experiments revealed that it is possible to reduce the budget for spare parts by 10%-20% using repair priorities. In a few extreme cases, we even found 70% budget reduction.

Chapter 1
Introduction

In this chapter we discuss the design of our research and the positioning in the literature. We formulate and motivate the research questions that we will answer in this thesis and we explain what is new compared with existing literature. Further, we give an overview of the remainder of this thesis.

1.1 Motivation

1.1.1 Service part logistics

Technically advanced systems play an ever more important role in society. As a consequence, the availability of such systems may strongly affect daily operations. This applies to e.g. heavily automated production processes, computer systems, medical equipment, and military systems. Downtime of critical equipment may have serious consequences, e.g. in terms of loss of production, quality reduction in health care or ineffective military missions. Various measures can be taken to reduce the amount of system downtime, such as system redundancy, appropriate preventive maintenance and effective corrective maintenance. Especially with respect to the latter, fast supply of the service parts required is essential. Here we define service parts as all parts that are used to maintain systems, both spare parts to replace failed parts and diagnostic items that are used to analyse the system performance and to find failure causes.

The importance of service parts management has increased in the past decades. One reason is the fact that system availability and high quality after sales service have become important criteria when selecting suppliers of technically advanced systems. A second reason is the increasing value of service part inventory investment. A survey by Cohen et al. (1997) reports that service parts inventories equal 8.75% of the value of product sales in their sample, being over $23 mln. inventory investment on average. In this survey, the following characteristics and trends in service delivery organisations are observed:

- a large and geographically dispersed installed base;
- a large number of service parts to be stocked, varying between 2,500 and 300,000 in the sample;
- increasing costs of service parts due to increasing complexity and modularity, in the sample being $270 on average with exceptions up to several hundreds thousands of dollars;
- a high part obsolescence rate caused by short product life cycles;
- a large and increasing fraction of slow moving service parts, caused by increased system customisation, and design improvement; the average inventory turnover in the sample equals 0.87 parts per year, hence there are many parts having a demand rate less than once per year

As a consequence of the increasing costs of spare parts (third characteristic above), it is worthwhile to consider repair rather than scrap and replace. In the survey by Cohen et al. (1997), it appears that in the sample 27.2% of the parts are *repairable*, i.e. parts for which repair is technically possible and economically profitable. Although this suggests that consumables are more important, one should keep in mind that repairables are generally more expensive, so the share of repairables in total service part investment is probably considerably higher.

The spare parts are needed to maintain an *installed base* of technical systems. Examples are aircrafts, locomotives, frigates and computer systems. The product structure of failed units (assemblies, subassemblies) is an important characteristic to be taken into account when analysing service part networks. A system is only available if all critical parts function well. Therefore, the availability of all parts and modules in the (multi-layered) product structure simultaneously determine the availability. This is generally addressed in literature as *multi-indenture* problems.

Service parts are often supplied via a *multi-echelon* distribution network, i.e. a hierarchical network of stocking locations through which service parts are supplied to the installed base at customer's sites. A reason to have a multi-echelon structure is the need for both local stocks close to the customer's sites in order to achieve fast supply and the need for stock centralisation to reduce holding costs. Cohen et al. (1997) report that three-echelon networks are prevalent in their sample followed by two-echelon systems. Four-echelon networks occur in practice as well. There is a trend however to reduce the number of echelons and the number of locations per echelon in order to reduce fixed warehousing costs and service parts obsolescence costs. This striving for lean and efficient service part networks is facilitated by e.g. stocking essential parts at the customer sites and using possibilities for fast emergency transportation.

All these characteristics cause that service parts management is an increasingly important, yet complex task. A key challenge is to attain high availability of the installed base at low service costs. These service costs include costs for stock holding, warehousing, transportation, service engineers, repair shops and overhead. Effective and efficient spare part management means that several design choices have to be made (e.g. network structure) and a suitable logistical control structure has to be developed. In this thesis, we focus on some logistics control issues, given strategic design choices such as product design and distribution and return network design.

1.1.2 Logistic control issues

Logistic control in spare part networks consists of a variety of decisions. Verrijdt (1997) discusses these decisions and structures them in a framework for control that he calls the Service Part Supply System (SPSS) for repairable service parts. We will summarise this framework and select the issues on which we will focus in this thesis.

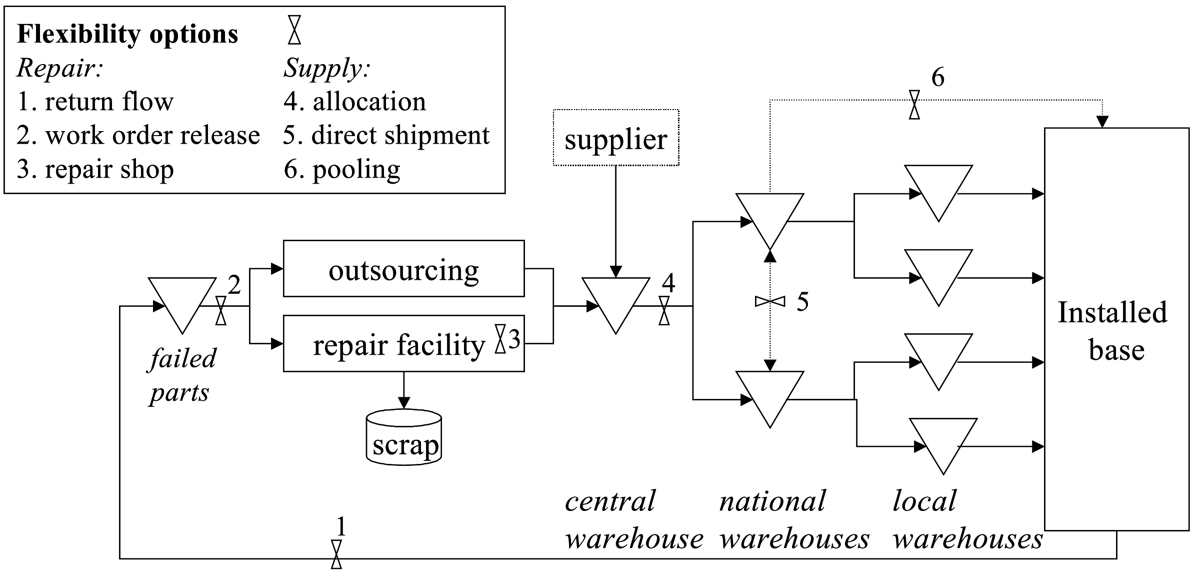

Figure 1.1. A Service Part Supply System with central repair facility, cf. Verrijdt (1997)

According to Verrijdt (1997), the Service Part Supply System "describes the operational processes that service parts go through in a repair and distribution network", see Figure 1.1. The network consists of a supply subsystem and a repair subsystem. The parts needed to maintain the installed base are supplied via a multi-echelon distribution network. Figure 1.1 shows a three-echelon network, consisting of a central warehouse, national warehouses and local warehouses. Upon failure, a part is dispatched to a location where it can be repaired, either in an own repair shop or by a third party (outsourcing). After repair, the part is forwarded to the distribution network for re-use. If it is technically not feasible or economically not profitable to repair the part, it is scrapped and replaced by a new part that is ordered from a supplier.

Although Figure 1.1 shows a single repair facility only, the repair subsystem can have a similar network structure as the distribution subsystem, see Figure 1.2. Repair facilities may exist at a local and/or a central level in the network. A reasonable option is to have repair shops close to the customer sites for relatively simple repairs. Defects that require specific expertise and/or tools can be handled at a central level, because of the relatively high costs of such special resources. When an item fails, it has to be diagnosed to find out which repair facilities are able to solve the problem. Next, the item is dispatched to a suitable repair shop.

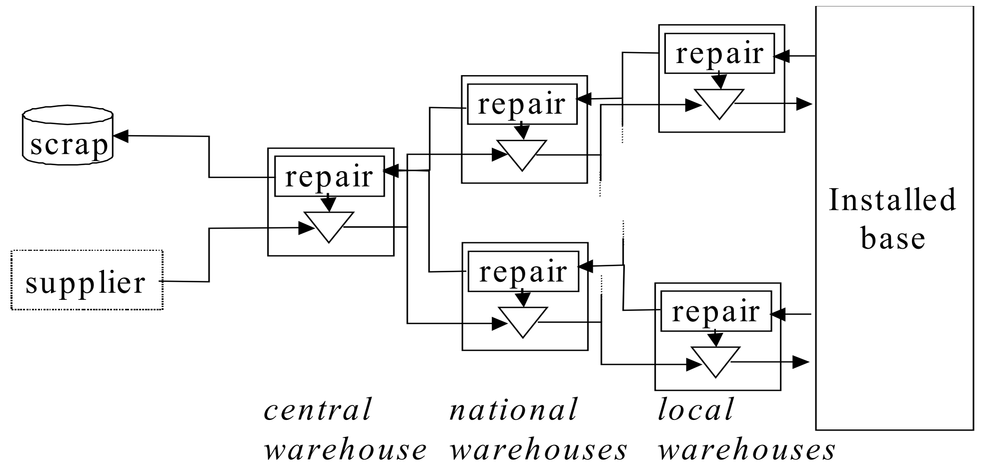

Figure 1.2. The Service Part Supply System with local repair facilities

Verrijdt (1997) distinguished six so-called flexibility options that can be applied to increase the performance of the system:

Repair subsystem

1. *Return flow flexibility* covers decisions to control the return flow of failed items from the installed base to the repair facilities. Examples are:
   a. should we ship the parts immediately after failure to the repair shops (one-by-one) or should we wait until we have a certain minimum batch size? This is a trade-off between holding costs (immediate shipment reduces throughput times and hence the number of items in repair) and transportation costs (batching reduces the number of shipments).
   b. should we return failed parts via the distribution network or should we use a separate return network? In the first case, we return items from the customer in the reversed direction order through the distribution network until it arrives at the appropriate repair shop. In the second case, we reduce return times by direct shipment to the repair shops.
2. *Work order release flexibility* covers the release of repair orders to the repair shops. Examples are:
   a. should parts be repaired one-by-one or in batches?

   b. which repair shop at which location should repair a failed part?

3. *Repair shop flexibility* covers the capacity of repair shops as well as planning and scheduling of repair shops. Examples are:
   a. which capacity should be assigned to a repair shop?
   b. which repair jobs should have high priority and which low priority?
   c. to which extent should we use flexible manpower planning, such as variable working times and outsourcing of activities?

<u>Supply subsystem</u>

4. *Allocation flexibility* covers the allocation of part inventories in the distribution network. Examples are:
   a. at which locations should we keep a part on stock (central / local)?
   b. which norm stocks do we need for each part at each location?
5. *Pooling flexibility* covers possible joint use of inventories at the *same* level in the supply network, which is also known as lateral supply (e.g., sharing stocks between national warehouses). Examples are:
   a. under which circumstances should we use regular supply of parts and when should we use lateral supply?
   b. which locations should we check in which order for lateral supply of parts in case of an-out-of stock situation?
6. *Direct shipment flexibility* covers emergency shipments from an inventory location at a higher level in the supply chain. Similarly to pooling flexibility, decisions should be taken on item routing and criteria for use of (expensive) direct shipment instead of regular supply.

The tactical and operational decisions in the Service Part Supply System framework directly influence inventory holding costs, (normal and emergency) transportation costs and out-of-stock costs. Besides, costs for information technology and organisational complexity may play its role.

In the last decades, many models have been developed to support logistic decisions as stated above. Most research focuses on one flexibility option, sometimes on the combination of two flexibility options. Especially, much attention has been paid to allocation flexibility. In this thesis, we will focus on the *interaction between allocation flexibility and repair shop flexibility*.

As mentioned above, stock allocation deals with decisions about which parts to stock at which locations in the network in which amounts. In service networks, the stock allocation is different from traditional inventory models, because the relevant performance measure is the availability of the installed base rather than part fill rates. A specific availability level can be attained by several combinations of part fill rates. One option is to use an equal fill rate for each part such that the target availability is attained. A more clever option is to focus on high fill rates for cheap, slow movers, so that relatively low fill rates (and hence low stock levels) of expensive, fast movers are sufficient. In this way, the same system availability can be obtained at lower costs.

Many models for these kinds of stock allocation problem have been developed in the past decades. Already in the 60's, Sherbrooke (1968) developed the famous *Multi-Echelon Technique for Recoverable Item Control* (METRIC). This model aims to allocate a fixed budget for spare parts, such that the availability of the installed base is maximised. The focus is the initial investment in spare parts at the start of a product life cycle. METRIC has been the basis for a lot of extensions and applications later on. One of these extensions, the *VARI-METRIC* model proposed by Slay (1984) and improved later by Graves (1985), fits well as a framework for the spare parts models with finite repair capacity that we will develop. An overview of the most important models based on the METRIC approach is given in Sherbrooke (1992). A key assumption in the METRIC models is that the repair shops have *infinite* capacity and that they can be modelled as $M/G/\infty$ queues. Therefore, we can exploit Palm's theorem that states that the number of items in an $M/G/\infty$ queue is Poisson distributed (Palm, 1938). This simplifies stock analysis of spare part networks. In the next section, we will discuss the validity of the infinite capacity assumption from a practical point of view.

1.1.3 The role of repair shops with finite capacity

Although it is generally not true that all repair shops in the network have infinite capacity, this key assumption of the (VARI-)METRIC class of models may be justified by some arguments. Firstly, in some situations the repair of service parts is just one of the activities of the repairmen, e.g. next to preventive maintenance. If service part repair gets highest priority, the effect of finite capacity can be negligible. However, this is only true if service part repair is not a major task for each repairman. Besides, preventive maintenance can have high priority as well, e.g. when preventive maintenance of a production system has to be carried out during some limited time period outside regular production hours. Secondly, in some situations the repair shop capacity is flexible, e.g. because the repairmen work overtime if the workload is high or because it is possible to outsource service part repair if the workload is high. Using this flexible capacity, the repair shop has virtually infinite resources. However, working overtime may cause high overtime costs, while outsourcing is not always possible, in particular if the parts or modules are technically complicated and repair requires specific skills. Thirdly, the finite capacity could be taken into account by measuring the actual throughput time in the repair shop and plugging these values into the model as gross repair times (= net repair time plus waiting time). Although this seems to be a simple and straightforward approach, the throughput time measured is only valid under the current circumstances. Obviously the throughput times depend on service part demand, return procedures and repair shop capacity. Therefore this procedure is not suitable for what-if analyses, which is a serious drawback.

Another reason to include finite repair capacity in the model is the following. Under finite capacity, the item throughput times can be influenced using appropriate priority setting. For example, expensive items can be given high priority, so that repair shop throughput times are short and hence the stock levels required can remain low. This is only possible at the expense of lower priority for the other (cheaper) items, so that these items will face longer throughput times and hence require higher stock levels. If the item values are very different, such priority setting might be worthwhile to consider in order to reduce the investment needed to obtain a target availability of the installed base or, equivalently, to attain a higher availability for the same budget. We expect that, in particular, this can be attractive if the repair shop utilisation is high, because then waiting times are significant and priorities influence waiting times only. To judge the benefits of repair priorities, we need models to quantify the relation between spare part inventories, repair capacities and repair job priorities. We have not found such integral models in the literature.

Of course, the capacity effects have been recognised both in practice and in literature. Pyke (1990) discusses the impact of finite capacity and repair priority setting using discrete event simulation. He finds that applying appropriate priority rules can have significant impact on the system performance if the utilisation is high. An analytical method is not developed, however. De Haas and Verrijdt (1997) examine capacity effects and encounter heavily fluctuating work loads in the repair shops of the aircraft maintenance system they study, causing varying throughput times and frequent overtime and stress for the repair men. They suggest that the throughput times in the repair shop should be corrected for utilisation of repairmen. Although they consider various options for repair shop throughput times in their numerical illustration, they do not describe a method to quantify the relation between utilisation and throughput time. Hence the issue how to deal exactly with finite capacity remains unsolved.

Our choice to examine finite repair capacity is further motivated by Rustenburg et al. (2001), who state that "one of the most criticised assumptions on (VARI-)METRIC-models and their extensions is the assumption of unlimited repair capacity". They put capacity restrictions on their agenda for research, but note that "the combination of capacitated multi-indenture and multi-echelon structures however will require a substantial research effort". We aim to contribute to this research effort with this thesis.

1.2 Research design

1.2.1 Research objective

In section 1.1, we discussed the importance of inventory control in spare part networks, and we observed that there is a lack of appropriate models that facilitate a trade-off between spare part inventories, repair capacities and repair priorities. In line with the (VARI-)METRIC-class of models, we focus on the *initial supply* of spare part inventories and the relation with repair capacity and job priorities. That is, we deal with logistic control at a tactical level. Based on the discussions above, we formulate our research objective as follows:

to develop a set of tools for optimising repairable spare part inventory levels in multi-echelon, multi-indenture networks with finite capacity repair shops, and to gain insight in the relation between inventories, repair capacities and repair priorities using these tools.

Below, we discuss the important elements of our research objective.

1. *Tool*: We aim to develop simple software prototypes, programmed in Delphi. The main purpose of these tools is to test our algorithms and to gain insight in the relation between inventories, capacities and priorities by conducting numerical experiments. Use by third parties is not directly facilitated, in the sense that we do not develop advanced user-interfaces or database connections. We will verify and validate our algorithms by comparison to discrete simulation results. To this end, a simulation model will be developed in the object oriented simulation software eM-Plant™ (Tecnomatix, 2000).
2. *Optimising*: As goal function, we will minimise the relevant costs needed to attain a target availability of the installed base. A key cost factor in our models is the initial investment

of spare parts and its relation to repair shop utilisation and priority assignment. In one model, we will explicitly include the costs of repair capacity to model an inventory-capacity trade-off.

3. *Insight*: We strive to gain insight for the following topics:
    - the impact of assuming infinite repair capacity on the stock levels and system availability if the actual repair capacities are finite;
    - the impact of repair shop utilisation on stock levels and stock distribution;
    - the relation between the preferable repair shop utilisation rate on one hand and the costs of spare parts and repair capacity on the other hand;
    - the reduction in the costs of spare parts provisioning that we can achieve using appropriate repair priorities, given fixed repair capacities.

1.2.2 Research questions and approach

To reach our objective, we define a number of *research questions* that we have to answer. These questions also define a logical sequence of research activities. Below each question, we discuss by what approach we are going to answer it. Between parentheses, the chapter is given that deals with the specific research question.

1. *Which model for finite capacity repair shops is appropriate to be embedded in the (VARI-)METRIC framework (Chapter 2)?*

    When dealing with this question, we first discuss the (VARI-)METRIC framework for the initial allocation of repairable spare part inventories in service networks. The model assumptions, definitions and notation will be clarified. Also, we explain the optimisation algorithm for infinite capacity repair shops, based on Sherbrooke (1992) and Rustenburg (2000). Finally, we discuss the requirements for a finite capacity repair shop model that can be embedded in the (VARI-)METRIC framework and we choose an appropriate repair shop model: multi-class, multi-server queues with Poisson arrivals and class-dependent exponential service time distributions.

2. *How do we analyse the selected repair shop model and how do we derive the performance characteristics needed for (VARI-)METRIC (Chapter 3)?*

    It will appear that we need to analyse a queueing model for which the required performance characteristics are not yet available in the literature. Therefore, we develop our own method. We validate our method by comparison to discrete event simulation results. We also evaluate its computational efficiency, because we need to calculate performance characteristics for several parameter settings during a (VARI-)METRIC-like optimisation routine.

3. *To which extent can we improve (VARI-)METRIC by introducing our queueing model for the finite capacity repair shops (Chapter 4)?*
   We extend (VARI-)METRIC with finite capacity repair shops using our model from Chapter 3. We compare the system availability that is given by the traditional and the extended optimisation method to the actual availability. We estimate the actual performance characteristics using a discrete event simulation. To examine the impact of the infinite capacity assumption that is usually made, we conduct a numerical experiment comparing the traditional method with our extended variant. We measure the impact by (a) comparing the estimated availability to the actual (simulated) availability, and by (b) comparing the stock allocation under infinite and finite repair capacities. At this stage of our research, we do not deal with repair priorities yet.

4. *How can we choose an appropriate combination of spare part inventories and repair capacities (Chapter 5)?*
   Repair shop capacity is not a given parameter in many cases. Therefore, we extend our algorithm from Chapter 4 by adding the capacities of the repair shops as decision variables and by adding the related costs to the optimisation criterion. We show that we can extend (VARI-)METRIC to a simultaneous capacity-inventory optimisation method. Using the tool that we develop, we can gain insight in the optimal utilisation of repair shops and the optimal allocation of budget between inventories and repair capacities. To this end, we conduct another numerical experiment.

5. *How do we extend and analyse our repair shop model to deal with repair priorities (Chapter 6 and 7)?*
   To find out what we can gain by proper priority setting, we first need to extend our queueing model to account for repair job priorities. In Chapter 6, we present a method to analyse repair shops with two priority classes and preemptive priority of the high priority class over low priority class. The analysis is done along the same lines as in Chapter 3. We validate our method by comparison with results from discrete event simulation. We also discuss a variant of our model in which high priority jobs are outsourced if they

cannot be served immediately upon arrival (Chapter 7). In this chapter we discuss both preemptive and non-preemptive priority of the high priority class over low priority class. The outsourcing of the high priority items leads to a simplified analysis of the queueing model. We compare both models and we show the impact of outsourcing on the number of low priority items in the system. For the outsourcing case, we also compare the preemptive and the non-preemptive priority rule.

6. *To which extent can we reduce inventory investment by introducing repair priorities, and how should we allocate items to priority classes then (Chapter 8)?*
   We extend our finite capacity variant of (VARI-)METRIC further to include repair shops with job priorities. Next, we discuss heuristics for simultaneous inventory optimisation and priority assignment. We use our method to examine the impact of priority assignment in a numerical experiment.

By answering the last research question, we bring together the factors finite repair capacity, repair priorities and spare part inventories, thereby satisfying our objective as stated in section 1.2.1. As a summary of our research design, we give the relation between the research questions in Figure 1.3. The arrows in this figure show the order of the research topics.

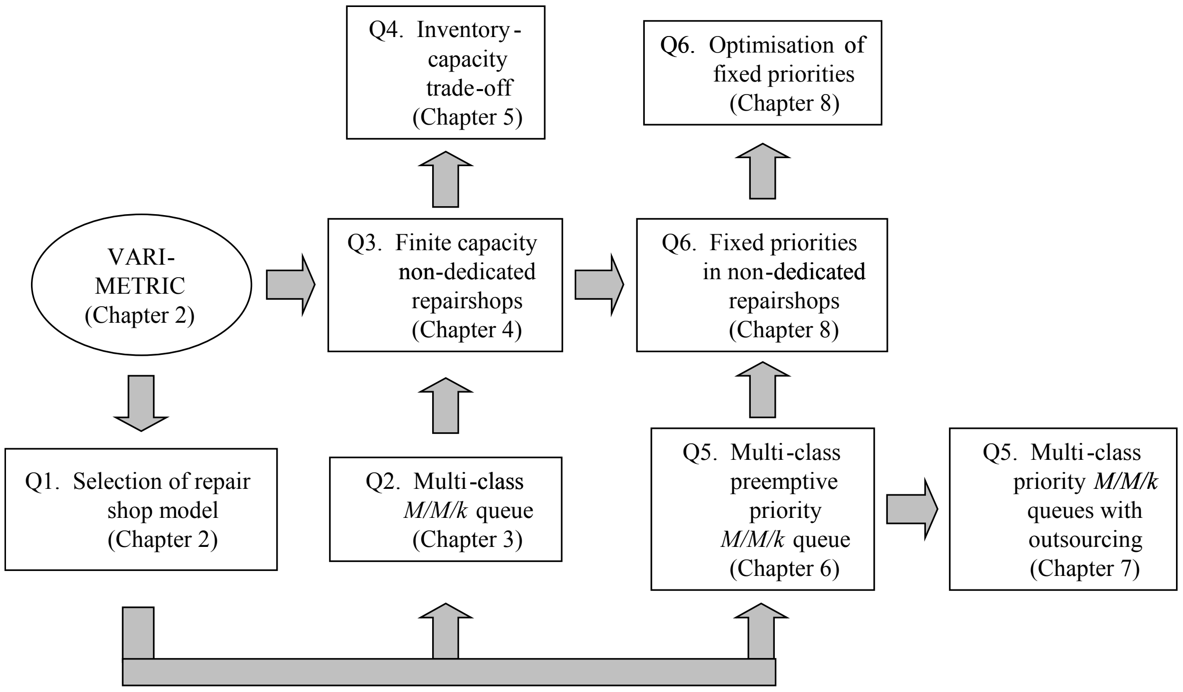
Q4. Inventory-capacity trade-off (Chapter 5)
Q6. Optimisation of fixed priorities (Chapter 8)
VARI-METRIC (Chapter 2)
Q3. Finite capacity non-dedicated repairshops (Chapter 4)
Q6. Fixed priorities in non-dedicated repairshops (Chapter 8)
Q1. Selection of repair shop model (Chapter 2)
Q2. Multi-class M/M/k queue (Chapter 3)
Q5. Multi-class preemptive priority M/M/k queue (Chapter 6)
Q5. Multi-class priority M/M/k queues with outsourcing (Chapter 7)

Figure 1.3. Overview of research topics and their relations

1.2.3 Demarcation

To control the project size, we need to make some choices on the boundaries of our research. This is done in the following way:

1. We will focus on developing planning algorithms and to gain theoretical insight using these algorithms. This already requires a lot of effort, because only few results were available in the literature at the start of this research. Because a lot of methodology still has to be developed, this thesis has a strong operations research character. Although this leaves little time for case studies, we will use the problem setting from the US Navy (Sherbrooke, 1992) and the Royal Netherlands Navy (cf. Rustenburg, 2000) as test cases.
2. We focus on spare parts, i.e. parts that are used to replace a failed item, and not on other service parts, e.g. tools and equipment that is needed for failure cause analysis and repair.
3. We assume that the multi-indenture product structure (bill of material) and the multi-echelon network structure are given.
4. We focus on *repairable* items. Of course, it may occur that it is technically not possible or economically not profitable to repair. In that case, the failed item will be scrapped and replaced. Because our focus is to examine finite capacity repair shops and repair priorities, it does not make sense to take into account consumable items, i.e. items that are always scrapped after failure.
5. As mentioned before, we focus on the *initial stock allocation*. In fact, we assume a steady state situation in which failed items are either repaired or replaced if necessary, so that the number of items circulating through the network remains constant.

We refer to Chapter 2 for a detailed overview of all assumptions that are commonly used in the (VARI-)METRIC-class of models and that we will use as well throughout this thesis.

1.3 Positioning in the literature and our contributions

In this section, we discuss the literature that is related to our research objective and questions. First, we describe some general literature on spare parts management and its applications (section 1.3.1). Section 1.3.2 deals with the (VARI-)METRIC class of models for repairable item inventory control. In section 1.3.3, we discuss the results on finite repair capacities in these models. Next, we address the issue of job priorities in finite capacity repair shops (section 1.3.4). As repair shops are modelled as queueing systems in such models, we discuss relevant queueing systems in section 1.3.5. Finally, we state our contribution to literature in section 1.3.6.

1.3.1 Spare part management

Optimisation of repairable spare part supply networks has already been the subject of research for many years. Most effort has been spent to optimisation of stock levels. Results have been achieved for both consumables and repairables. Consumables are spare parts that are scrapped once they have failed. A line of research has been dedicated to this area, see Cohen et al. (1986, 1988, 1989, 1992). They focus on periodic review systems with batching. Their models have successfully been applied at IBM (Cohen et al, 1990). In the area of consumables, the decomposition approach of Hopp et al. (1999) is worthwhile to mention. They minimise inventory costs of consumable spare parts under service constraints, representing the total delay per facility because of lack of spare parts. Another approach is discrete event simulation, see e.g. Seyboth (1997) for an application to spare part supply of trucks.

Another line of research has been dedicated to repairable items. The basic reference in this area is the METRIC-method of Sherbrooke (1968). Models of the METRIC-class have initially been applied to various military systems in the air force, the navy and the army (Sherbrooke, 1992, Rustenburg, 2000). This model class is generally recognised to be useful for commercial applications as well, insofar repairable items are involved. Guide and Srivastava (1997) mention “the leasing of office equipment (e.g. reproducing equipment and computers), heavy equipment (e.g. tractors, earth moving equipment, industrial presses), and transportation equipment such as railroad, subway cars and buses.” Practical applications are described in various commercial settings, such as aircrafts (Tedone, 1989; de Haas and Verrijdt, 1997), a metro system (Diaz and Fu, 1997) and electronic testing equipment (Cohen et al., 1999). Further, combinations of repairable item inventory theory and simulation have been applied to the personal computer business (Ashayeri et al., 1996) and diesel locomotives (Kumar et al., 1994). Because the classical applications in the area are military systems, we will use cases derived from Rustenburg (2000) and Sherbrooke (1992) in this thesis.

Most research effort has been spent to the optimisation of spare part inventories as discussed above, i.e. the *allocation flexibility* of the framework of Verrijdt (1997), see also section 1.1.2. As mentioned before, this flexibility option also includes the allocation of spare parts to stockpoints after repair, taking into account local shortages, see Miller (1974), Pyke (1990) and Büyükkurt and Parlar (1993). A related problem is to allocate spare parts to the installed based, taking into account differences in equipment criticality (Dekker et al, 1998). Also, we find a similar problem setting in reverse logistics, see Fleischmann et al. (1997) and Inderfurth and Teunter (2001) for an overview. Inventory control is one of the main research areas within this field. Fleishmann et al. (1997) state that the key differences between reverse logistics and spare part management are (i) returned items are immediately replaced by new ones insofar possible, and (ii) a spare part system is essentially a closed-loop system. Similarities include the need to organise return flows, the need to disassemble equipment and the uncertainty in both failure cause and routing through the networks. As a consequence, logistic decisions are related, not only regarding inventory control.

Next to allocation flexibility, the other flexibility options have received less attention in the literature. *Pooling* flexibility (lateral supply) has been studied amongst other by Lee (1987), Axsäter (1990) and Dada (1992) and more recently by Alfredsson and Verrijdt (1999) and Grahovac and Chakravarty (2001). A special type of pooling flexibility is *cannibalisation* (Fisher and Brennan, 1986; Sherbrooke, 1992). That is, a working part from a failed system in the installed base is used to replace a failed item of another system, so that at least one of the two systems can be available. *Direct shipment flexibility* focusing on the trade-off between normal supply and emergency supply has been examined by Shibuya et al. (1998) and Alfredsson and Verrijdt (1999). All these issues relate to the supply subsystem. Regarding the repair subsystem, the literature dedicated to spare parts is limited. Partly, contributions can be found in reverse logistics literature as mentioned above, for example, the relation between the distribution and return network (*return flow flexibility*). Dedicated literature on *work order release flexibility* in spare part networks deals with using flexible repair shop capacity (de Haas, 1995; de Haas and Verrijdt, 1997) and batching policies (Chua et al, 1993). Within *repair shop flexibility*, we mention the option of emergency repair in case of shortages (Verrijdt et al,1998). Of course, standard operations scheduling literature applies to planning and scheduling of the repair facility as well, see e.g. Pinedo and Chao (1999).

Most of the literature focuses on a steady state situation. Another approach is to plan spare parts over the life cycle of equipment. Examples of dynamic inventory in such a setting can be found in Cho et al. (1996), Klein Haneveld and Teunter (1997), Walker (1997) and Cho (2001). All this work relates to single-item problems, and the extension to multi-item (let alone multi-indenture problems) is still a challenge. Related to dynamic inventory problems is the yearly resupply of spare parts under a finite budget, see Rustenburg et al, 2000.

Further, there is literature on the relation between maintenance optimisation and spare parts inventory control, see Kabir and Al-Olayan (1996a, 1996b), Sarker and Haque (2000).

For more literature on spare part management and related research issues, we refer to the reviews by Guide and Srivastava (1997), Verrijdt (1997), Rustenburg et al. (2001), and Kennedy et al. (2002).

1.3.2 The METRIC class of models

The METRIC approach (Sherbrooke, 1968, 1992) is concerned with setting the initial levels of repairable spare part inventories and their distribution among various locations in a distribution network. The goal is to maximise the average availability of a given installed base of technical systems that use the spare parts. Sherbrooke (1992) shows that maximising the average availability of the installed base is approximately equivalent to minimising the sum of the expected backorders of all highest indenture items at all downstream stock locations (i.e., the direct suppliers of spare parts to the installed base). METRIC is a greedy, iterative heuristic that increases the inventory level of a certain item at a certain location in each iteration. The selection criterion for item and location is the maximum backorder reduction per invested dollar. The original model by Sherbrooke (1968) is a multi-echelon, single-indenture model. This model is extended by Muckstadt (1973) to the multi-indenture case, which is known under the name MOD-METRIC. A key assumption in this model is that the so-called *pipeline* (i.e., the number of items in repair or in resupply) is Poisson distributed. This is not true for general multi-echelon, multi-indenture models. Therefore, Slay (1984) proposed to calculate the first two moments of the pipeline and to assume a negative binomial distribution with the same first two moments for the pipeline distribution. Because the variance is included, this variant is known as VARI-METRIC. Later, Sherbrooke combined MOD-METRIC and VARI-METRIC to a multi-echelon, multi-indenture spare parts supply model, which produces accurate results in comparison to simulation results (cf. Sherbrooke, 1986).

One of the assumptions of the VARI-METRIC models is commonly known as the ample service assumption. It means that the repair capacity is infinite, i.e. there is no queue of items waiting for a repair channel. This has the effect that the replenishment lead times can be considered as statistically independent, and the mean and variance of the number of items under repair service are equal. As discussed in section 1.1.3, the infinite capacity assumption may not be applicable in practice. As a consequence, the independence of different lead times does not apply anymore. This influences performance calculations (backorders, availability) as well the optimisation procedure.

1.3.3 Spare part inventories and finite capacities

In the literature, various ways to deal with finite capacity in service part networks have been discussed. One of these methods is to model the network as a closed queueing network (Jackson network, cf. Gross et al., 1978, 1983). This method provides very good estimations of the steady state probabilities in a closed network with fixed parameters, but the numerical algorithms involved make it difficult to find optimal stock levels for each location and each part type. Another approach is based on Markov processes; see Albright and Soni (1988), Gupta and Albright (1992) and Albright and Gupta (1993). A drawback of this approach is the fact that the number of states may become very large and that existing methods to reduce the model size to acceptable dimensions are rather rough.

A similar approach is developed by Avsar and Zijm (2000). They construct an excellent approximation for a two-echelon inventory model, where repair shops can be modelled as open Jackson queueing networks. However, their model considers only item-dedicated repair shops and is difficult to extend to multi-echelon model or model with different types of repair shops, as we consider.

Another possibility is to extend the VARI-METRIC method to deal with finite capacity by replacing the *M/G/∞* queueing model for the repair shop by some finite capacity system, cf. Aboud (1996), Diaz and Fu (1997), Kim et al. (2000) and Perlman et al. (2001). They use their method to analyse the impact of finite capacity. Diaz and Fu (1997) show that finite capacity has a serious impact on system performance for a single indenture, two-echelon system with only one central repair shop. They model the repair shop as a *GI/G/k* multi-class queueing system, where the part flow of one item type is modelled as one class in the queueing system. Although they discuss formulas for multi-server queues, their numerical results refer to single server queues only. In addition, they discuss an alternative method to plug in throughput times as observed in practice in the *M/G/∞* model, so that waiting times are included. This approach is also used in the case study for the Caracas Metro subway system that Diaz and Fu (1997) present. Then the impact of finite capacity is less, but still significant, and as we mentioned already this procedure is not suitable for what-if analyses.

Although the approach by Diaz and Fu (1997) is simple and attractive, they restrict themselves to a very simple situation, namely a single repair shop consisting of one or more single server queues for a single-indenture, two-echelon system. It can be expected that the model complexity increases if the service part supply system contains multiple repair shops at various locations in the network and for various levels in a multi-indenture system, because then the various queueing models interact which may cause a serious deterioration of numerical accuracy. Also, the model performance for multi-server repair shops has not been analysed.

1.3.4 Spare part inventories and repair priorities

When repair capacities are finite, an interesting question is how to use this capacity in an efficient way, particularly by sequencing or scheduling repair jobs. It has been noted in several publications that efficiency gain is possible using repair priorities, see e.g. Pyke (1990) and Verrijdt et al. (1998). Still, repair priority setting in spare part distribution networks has received only little attention in the literature. A possible cause is the lack of appropriate queueing models to analyse such repair shops until recently. Even, the analysis of spare part networks with finite capacity repair shops using simple First-Come-First-Serve (FCFS) rule is not straightforward, as is shown in Diaz and Fu (1997), Avsar and Zijm (2000) and Harten et al. (2000).

Some insight has been gained using two approaches. In the first approach, repair priorities are modelled as a dual mode repair shop providing normal and emergency service, see Verrijdt et al. (1998) and their references. In their model, a spare part might receive a fast emergency repair if net inventory of these parts is lower then or equal to some level, but at higher costs. The model is restricted to a single-indenture, single-echelon case with ample repair capacity. A related model is discussed in Perlman et al. (2001), who assume that items arriving at repair shop i are sent to emergency repair with probability p_i and to normal repair with probability 1 - p_i. They consider a single-item, two-echelon model with finite capacity repair shops. They optimise the system with respect to p_i. Because it is a single-item model, the impact of priority assignment to various items sharing the same repair capacity is not examined. Both Verrijdt et al. (1998) and Perlman et al. (2001) assume independent processing of normal and emergency items, which is not valid if both processes share the same repair capacity.

A second approach is to use simulation modelling. Hausman and Scudder (1982) analyse the impact of static and dynamic priority rules for infinite capacity repair shops using simulation. Pyke (1990) uses simulation to examine the impact of finite capacities and repair priorities along with dispatch priorities. He studies a single repair facility in a three-echelon network with single-indenture items only. For repair priorities, he considers for each spare part the number of systems that are down because of the failure of that specific item. The item that is contributing to most systems is repaired first, so this is a dynamic (state-dependent) priority rule. Pyke (1990) shows that his priority rule leads to a drastic decrease in the number of non-available systems if repair shops are heavy loaded (utilisation = 0.96). The impact of the priority rule is only small if the utilisation is 0.6. Also, he finds that the average number of failed systems depends strongly on the repair shop utilisation under FCFS repair, whereas the impact is much smaller if proper priority rules are used.

Next to these approaches to gain insight in the impact of repair priorities, other related research deals with the choice of dynamic priority rules in repair shops that are characterised by uncertain processing times, see e.g. Guide et al. (2000). Their main question is which priority scheduling policies minimise the average repair shop throughput time if no spares are allowed. Based on a simulation model, they find that the Shortest Processing Time (SPT) rule is usually dominant. However, they do not take into account differences in item costs. If item costs are very different, it is intuitively clear that this characteristic should be taken into account when setting priorities.

1.3.5 Queueing models for repair shops

From section 1.3.3, it is clear that we need queueing models for finite capacity repair shops. There is an extensive literature on queueing analysis, see e.g. Kleinrock (1975) and Gross et al. (1998). Applications are particularly in computer and telecommunication system analysis and the analysis of manufacturing systems, cf. Hall (1991), Papadopolous (1993) and Buzacott and Shanthikumar (1993). Before discussing the relevant literature, we have to discuss which kind of queueing models are relevant and which performance characteristics we need.

Starting with the latter issue, we have seen in section 1.3.2 that VARI-METRIC requires for each item type the first two moments of the pipeline, being the number of items in repair and in resupply. Therefore, we need the first two moments of the number of items in the system for each item type. If multiple item types share the same repair capacity and their repair time distribution is different (as is plausible), this implies that we need a *multi-class* queueing model. Here each class represents one item type from the multi-indenture product structure with its own arrival rate and repair process. Because a repair shop can have one or more parallel repair men or work stations for a well-defined set of repair jobs, we conclude that we need *multi-class, multi-server (MCMS) queues* for the repair shops. When we wish to examine the impact of repair priorities, it even means that we have to take into account priority classes. Therefore, we focus in our literature review on (i) multi-class, multi-server queues and (ii) priority queues. We will discuss the available literature on these two topics next. Thereby, we keep in mind that we need to evaluate *the first two moments of the number of items in the system for each class* as performance measures. Hence, results on only mean waiting time or mean queue length in MCMS (priority) queues are not sufficient for our purpose.

Multi-class, multi-server queues

The analysis of multi-class multi-server problems starts as a generalisation of the well-known *M/M/k* queue. In the more general MCMS context, the *M/M/k* queue can be considered as a special case where the classes of clients are indistinguishable with respect to their service characteristics. That is, the service rate of each class is equal to the average service rate. As a matter of fact, all classes can be considered to merge into one class only. The theory of the *M/M/k* queue is dealt with in any good textbook on operations research (e.g. Winston, 1997), or in many queueing book (e.g. Kleinrock, 1975 and Gross et al., 1998).

One line of research on the MCMS queue is the application of approximation techniques or asymptotic methods. Bertsimas and Mourtzinou (1997) consider an approach to analyse the multi-class queueing systems with general arrival and service distributions in heavy traffic based on distributional and conservation laws. However, their results are restricted to single server systems. Diaz and Fu (1997), give some approximations mainly for the single server case. Basic to the reasoning underlying their approximations is that the expected waiting time is class-independent, see also Whitt (1993). Moreover, they assume the correlation between items in queue and in service is related in a simple way to the class utilisation fractions. As we will see in the remainder of this thesis, the exact solution of MCMS problems has a more complex structure. Other interesting work in this respect is Adan and van der Wal (1998). They relate mean lead time of a two-class system to those of a single class system with the Coxian-2 service process and show that systems where different items are processed by the same servers might perform better than the system with dedicated servers if the difference between service rates (μ_1/μ_2) is not too big (> 7).

There are only few papers that are directly devoted to MCMS models, as far as we know (de Smit, 1983a, 1983b). De Smit (1983a) analyses the *GI/M/k* multi-class queue where different types of customers have different service rates and which is equivalent to the *GI/H_m /k* queue. Here *GI* refers to general arrival processes and H_m denotes a mixture of different exponential distributions, as one obtains in a multi-class model, if one abstracts from the identity of the different jobs and introduces one overall jobtype. His approach uses phase vectors and a solution is derived by Laplace transform and Wiener-Hopf decomposition techniques. He obtains explicit results for the stationary distributions of waiting times and the queue length. It is shown that the stationary waiting time distribution is a mixture of exponentials, generalizing a previous result for $M/H_m/2$ of Cohen (1982). This result is theoretical, but de Smit (1983b) develops a numerical solution based on these results. However, the numerical solution procedure is limited to H_2 service time distribution or equivalently two-class system with exponential service time distributions. For our purposes, we need systems with more item classes, particularly because we may need to evaluate multiple queueing models and variants when optimising spare part inventory levels under finite repair capacities.

Multi-server queues with priorities

In this thesis we consider only multi-class priority queue with fixed assignments of customers. A significant part of the literature deals with two priority classes. In our application, we may have more than two item types that share the same repair capacity. As a consequence, we either need a queueing model that can deal with an arbitrary number of priority classes (one for each item type), or a model with two priority classes that allows for subclasses within both high and low priority customers (items). We keep this in mind while discussing the related literature.

There is quite some literature on single server priority queueing systems. However, multi-server priority queueing systems have received much less attention. Recently, such models have been studied by Mitrani et al. (1981), Gail et al. (1988, 1992), Kao et al. (1990, 1999), Kella and Yechiali (1985), Wagner (1997a, 1997b, 1998). Results are available both if preemption is allowed, so high priority items may interrupt the service of low priority items, or not.

The non-preemptive queues are analysed most extensively by Wagner (1997a, 1997b, 1998), who considers multi-server, non-preemptive priority systems with Markovian arrival process, service times having phase type distributions and both finite or infinite queueing space. Another interesting approach to non-preemptive priority queues is proposed by Kao and Wilson (1999). They apply a power-series approach to solve the equilibrium equations and to estimate the main performance characteristics (mean, variance, etc.). The power-series approach has been introduced by Hooghiemstra et al. (1988) and has been applied before to solve a variety of queueing problems – particularly those with multidimensional state space. Kao and Wilson (1999) apply this approach to a multi-server queue with two priority classes and no preemption. They compare the performance of their method to a matrix analytic approach as described in Kao and Narayanan (1998). The power series approach is interesting, because it can easily be implemented and it can be extended to include more than two priority classes and to preemptive priorities in theory. However these extensions cause an enormous growth of state space, and hence memory requirements and computation time.

Among the papers on preemptive priorities, we mention the approximation approach of Buzen and Bondi (1983) and the generating function approach as proposed by Mitrani et al. (1981) and by Gail et al. (1982, 1992). The basic idea of the Buzen's approximation approach is to replace k servers by a single server that works k times as fast and to use a correction factor, being the ratio of the waiting times when the same trick would be applied to the non-priority multi-server queue. Although this approach is attractive because of its simplicity and its extendability to general service times, it was done for the mean number of items in the system only. In contrast to Buzen's idea, the generating function approach gives exact results for the first two moments of the number of items in the system. However, these approaches (Gail et al., 1992, Mitrani et al., 1981) can only be applied to cases with two classes, each with one item-type.

1.3.6 Our contributions

Our contribution to the literature consists of

1. The analysis of *new queueing models*, namely multi-class, multi-server queues. Because these models are complicated, we restrict ourselves to models with Poisson arrivals and exponential service times (the multi-class *M/M/k* model). We will both address non-priority model (Chapter 3) and a model with preemptive priorities (Chapter 6). As a special case, we will discuss a new the multi-class *M/M/k* queue with outsourcing of high priority items that cannot be served immediately upon arrival. For this new model, we will discuss both preemptive and non-preemptive priorities (Chapter 7).
2. An *extension of VARI-METRIC with finite capacity repair shops*. This model is new in the sense that such a general model with multiple item classes sharing the same repair capacity has not been analysed before in such detail (Chapter 4). It is also new in the sense that trade-off between extra stocks and extra capacities is considered (Chapter 5).
3. An *extension of VARI-METRIC with repair priorities*. We do not just simply replace the multi-class *M/M/k* queue by the multi-class *M/M/k* priority queue, but we also address *simultaneous priority assignment and inventory optimisation* (Chapter 8). The latter is completely new, as no related models are available in the literature as far as we know.

In this thesis, we will show that significant gain in inventory investment is possible by proper priority assignment to items and/or by adequate investments in extra capacity.

1.4 Outline of the thesis

This thesis is structured as a set of self-contained papers that subsequently answer the research questions as posed in section 1.2.1. In each chapter, we will answer one research question. In Chapter 2, we discuss the (VARI-)METRIC model and we choose a queueing model for the repair shops. We analyse this queueing model in Chapter 3. Next, we extend METRIC to include finite capacity repair shops using our queueing analysis in Chapter 4. In the same chapter, we examine the impact of assuming infinite capacities whereas the actual repair capacities are finite. Also, we address the impact of repair shop utilisation on stock levels and distribution. Based on our finite capacity model, we can introduce a trade-off between inventories and repair capacities in METRIC. This is the subject of Chapter 5, in which we will also use our model to gain insight in the relation between the preferable repair shop utilisation rate on one hand and the costs of spare parts and repair capacity on the other hand. In Chapter 6, we extend our queueing model for the repair shop by introducing fixed priority classes. A variant of this model with outsourcing of high priority items if they cannot be served immediately upon arrival is discussed in Chapter 7. We extend METRIC with priority repair shops in Chapter 8 and we discuss simultaneous inventory optimisation and priority assignment. We use our extended model to gain insight in the extent to which repair priorities can be used to reduce the costs of spare parts provisioning. Finally, we summarise our conclusions and we discuss directions for follow-up research in Chapter 9.

Chapter 2
Inventory optimisation with infinite repair capacity: VARI-METRIC

Abstract

In this chapter we formulate the model of supply systems for repairable service parts with infinite repair capacity. We introduce the basic definitions and assumptions of our multi-echelon, multi-indenture spare part models. We give an outline of VARI-METRIC, the standard method to optimise inventory levels of repairable items in such networks. This method will be the basis for our model extensions to finite capacity repair shops in the remainder of this thesis. Therefore, we discuss the requirements for a finite capacity repair shop model that can be embedded in the VARI-METRIC framework. Based on these requirements, we choose a repair shop model, thereby answering research question 1 as stated in Chapter 1. This model is a multi-class M/M/k queue, possibly with two fixed priority classes.

2.1 Introduction

In this chapter, we describe Sherbrooke's (1992) model for general multi-item, multi-indenture, multi-echelon inventory systems and the corresponding optimisation heuristic, VARI-METRIC. First, we introduce the key concepts and definitions (section 2.2). Next, we formulate the VARI-METRIC optimisation model and we discuss the assumptions and notation that we will use in the remainder of this thesis (section 2.3). In section 2.4, we describe the VARI-METRIC method for optimising repairable spare part inventory levels. Finally, in section 2.5 we discuss which repair shop model is appropriate to extend VARI-METRIC with finite repair capacities, and which new queueing theoretic results we need.

2.2 Concepts and definitions

For better understanding, we first define the multi-echelon, multi-indenture systems and the flows of spare parts through such systems. Also, we address the definition of system availability.

2.2.1 The multi-indenture hierarchy

For maintenance and spare parts supply, the internal structure of a technical system consisting of modules and parts is relevant. In MRP-systems, this structure is described by a Bill of Material (BOM), see Figure 2.1 for an example. Such structures are used in spare part management as well. A technical system has one or more products, each consisting of modules and parts. The levels are usually referred to as *indentures*. Therefore, we deal with *multi-product, multi-indenture systems*. We number the indentures from the highest to the lowest level in the sub-assembly hierarchy, where level 1 represents the product level. With *single-indenture* systems, we refer to systems having several products without substructures. If at least one of the products consists of subassemblies, we have a multi-indenture system.

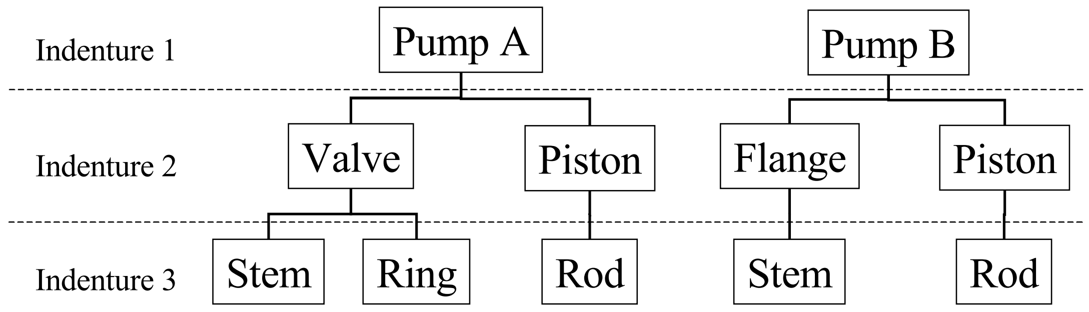

Figure 2.1. The multi-indenture item hierarchy in Sherbrooke's example

Further, we will use the following formal definitions:

*an **assembly** is an item that consists of lower level items in the system structure;*
*a **subassembly** is an item that is a part of a higher level item in the system structure;*
*a **product** is an assembly that is not a subassembly of any other item at the same time;*
*an **item** or a **part** is general terminology for assemblies, subassemblies and products;*
*the **multiplicity** of a subassembly is the number of identical items in a single assembly.*

In our example, the systems constituting the installed base are submarines. Each submarine has two products: pump A and pump B. Pump A consists of a valve and a piston. Similarly, pump B consists of a flange and a piston. The valve, flange and piston are assembled from a stem, ring and a rod as shown in Figure 2.1. In all cases, the multiplicity equals 1. Pump A, pump B, the valve, the piston and the flange are assemblies, whereas the stem, the ring, the rod, the valve and the piston are subassemblies.

2.2.2 The multi-echelon hierarchy

Because the installed base of technical systems is usually geographically dispersed, spare parts are stocked on various locations. Together, these locations form a network through which spare parts are distributed to the installed base. Items can be stocked at a central or at local locations (or a combination). This gives us a hierarchic, multi-echelon distribution structure. We introduce the following definitions:

*A **location** is a place that has either spare part stocks, or repair facilities for failed items, or both;*
*An **echelon** is a set of locations at the same hierarchical level, where each location will have at most one supplier (location from higher echelon) and at least one customer (location from lower echelon, or the installed base for the lowest echelon);*
*A **downstream** location is a location closer to the installed base than the current location in the sense of distribution flow; locations farther away are called **upstream**.*

We number the echelon levels from upstream to downstream in the geographical hierarchy, where level 1 represents the location most upstream. With *single-echelon* systems, we refer to systems having one or more locations that directly supply spare parts to the installed base without mutual deliveries. If at least one of the locations receives regular resupply of spare parts from another location, we have a multi-echelon system.

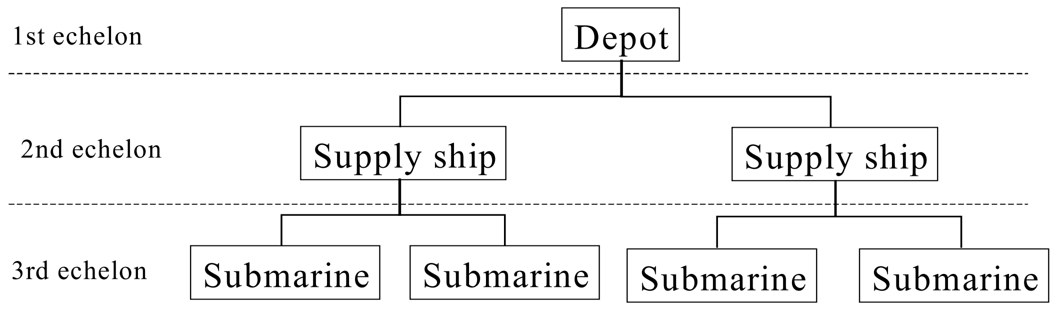

Figure 2.2. The example of the supply system hierarchy

In our example, we have three echelons: submarines, supply ships and the depot. The submarines are downstream locations for the supply ships and the depot is an upstream location for the supply ships.

2.2.3 Spare part flows

The item flows in multi-echelon, multi-indenture systems consist of the distribution of spare parts that are ready for use between stock locations and the installed base, and the return flows of failed items that should be repaired from the installed base to the repair shops. We will first describe how we model these item flows for simple single-site, single-indenture systems. Next, we extend the description stepwise to general multi-echelon, multi-indenture systems. Note that the description below coincides with the standard VARI-METRIC item flows, cf. Sherbrooke (1992).

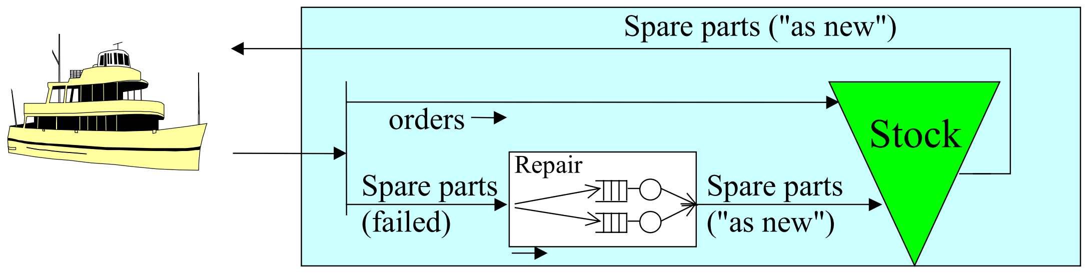

Figure 2.3. Single-site, single-indenture system (one location consisting of a stockpoint and a repairshop).

Figure 2.3 presents an example of a *single-site, single-indenture* system. The single site has a repair facility as well as a stockpoint. A sequence of events is triggered by the failure of a system in the installed base that is due to the failure of a specific product. That item arrives at the location and is directed to a repair facility. At the same time, the inventories are checked for a new item to replace the failed one. If such an item is available, it is issued immediately to replace the failed item. Assuming that the replacement time is negligible, there is no system down time. However, if the location is out of stock, the item is backordered and the system is temporarily down until a new item becomes available.

The failed item enters the repair facility, where it possibly has to wait for repair capacity. After repair, the item is assumed to be as good as new. If repair appears to be not possible, the item is scrapped and a new item is ordered at an external supplier. As a consequence, the total number of spare parts circulating through the system is constant.

When the item arrives at the stockpoint (either from the repair shop or from the external supplier), it is checked whether there are still backorders for that item. If so, the backorder waiting longest is filled (First-Come, First-Serve) and the corresponding system is switched into operation again. If there are no backorders, the item is added to stock.

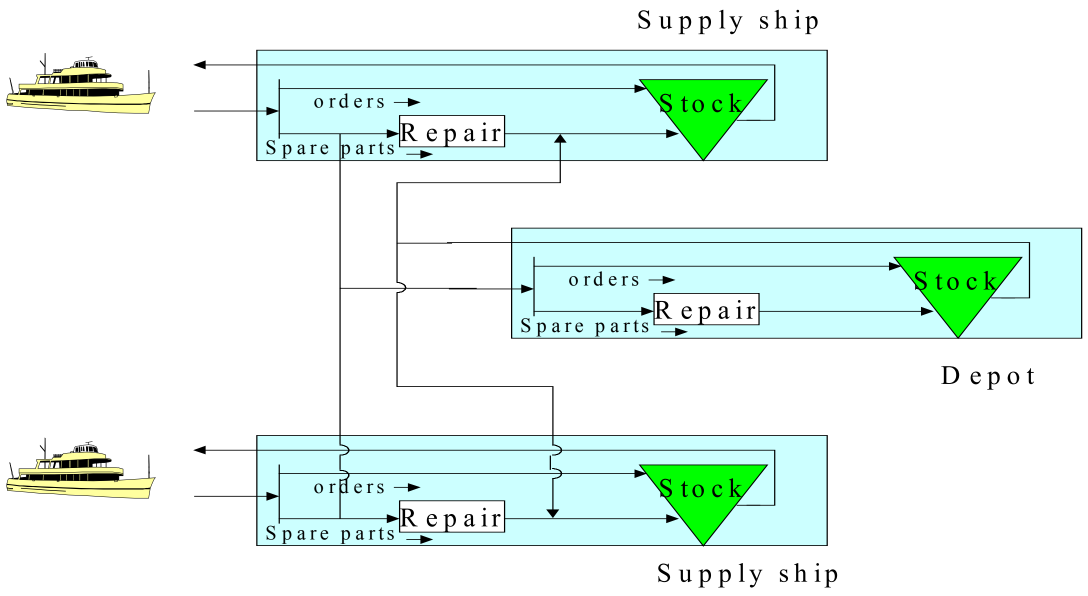

Figure 2.4. Two-echelon, one-indenture system

Next, we extend this simple system to a *multi-echelon* structure, e.g. with supply ships as downstream echelon and a depot in the harbour as the upstream echelon. Upon diagnosis, the failed item can follow two routes through the system, see Figure 2.4: either it enters the local repair facility, or it is forwarded to the next echelon upstream to be repaired there. Usually, items are sent to the higher echelon if local repair is technically impossible, i.e. if the local repair shop doesn't have appropriate equipment or skills. If an item cannot be repaired at the most upstream location, it is sent to an external supplier for repair or replacement; we model the internal structure of the external supplier as a "black box" by throughput times only.

If a new item is available at the local stockpoint, it is used to replace the failed item, otherwise the item is backordered. If the failed item has been forwarded upstream for repair, an order for a new item is issued to replenish the local stock. In this way, all local inventory positions (physical stock + in repair + on order - backorders) remain constant for all locations. If the upstream location does not have sufficient stock, the item is backordered on its turn until repair has been completed. When an item arrives at any location, either from the repair shop or from an upstream location as replenishment order, it can be used to fill a backorder.

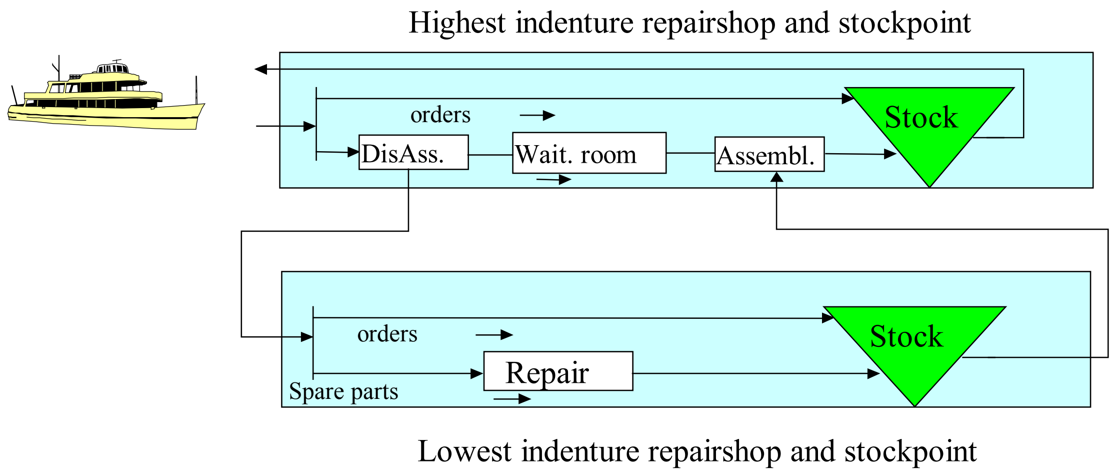
Highest indenture repairshop and stockpoint
orders
Stock
DisAss.
Wait. room
Assembl.
orders
Stock
Repair
Spare parts
Lowest indenture repairshop and stockpoint

Figure 2.5. One-echelon, two-indenture system

If we extend the single-site, single-indenture system to a *multi-indenture* structure, we have product flows as shown in Figure 2.5. A failure of a product in the system is either caused by one of its subassemblies (so the replacement of this subassembly is sufficient for repair), or there is no specific subassembly causing the failure (hence the product has to be repaired). In the second case, the procedure is identical to the single-indenture model. In the first case, the repair procedure is as follows (see Figure 2.5): the item is disassembled and the failed subassembly is sent to the repair facility. At the same time, the product is replaced by a new one if it is available on stock. If not, it is backordered. The disassembled product on its turn generates demand for a subassembly. If a new subassembly is available, the item is assembled and it is available to fill a backorder at product level.

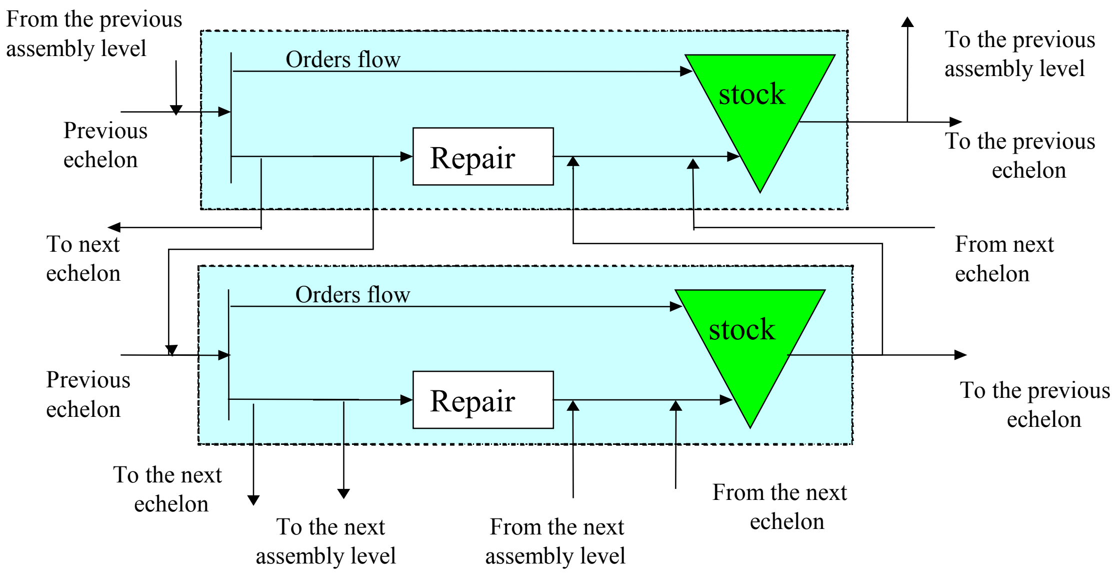

Figure 2.6. The flow between repair shops and stockpoints

If we combine both extensions to a general *multi-echelon, multi-indenture* system, we have flows between repair shops and stockpoints as shown in Figure 2.6. In general, failed parts arrive at a repair shop from the operational field, possibly via a downstream location where the item cannot be repaired. The failed product is replaced from the corresponding downstream stock location if available. At the location where the item is repaired, an item is requested to replenish the inventory at the downstream location. This is according to the multi-echelon procedure as described above. Next, we follow the multi-indenture procedure. That is, the cause of the failure is determined and the failed subassembly is sent to the repair facility or the item itself enters the repair process. The rest of the procedure is as described above.

An important notion within VARI-METRIC is the so-called pipeline, being the number of items in process between the arrival of the failed item at a repair shop and the repair completion. To be precise, we define:

the ***pipeline*** *of an item at a location m consists of all items in repair at location m, all items on order at the supplier of location m and all items waiting at location m for subassembly replacement*

2.2.4 Supply availability

A key performance measure for spare part networks is the average availability of the installed base. Sherbrooke (1992) mentions several relevant definitions of availability, depending on the extent to which preventive maintenance is included. Because we are interested in the impact of spare part provisioning (via stock levels, repair capacities and repair priorities), the *supply availability* is a relevant performance measure. Similar to Sherbrooke (1992), this availability measure is defined as

$$\textit{Supply availability} = \frac{MTBM}{MTBM + MSD} \times 100\%$$

where MTBM denotes the mean time between maintenance and MSD the mean supply delay, i.e. the system downtime caused by unavailability of spare parts. We will simply refer to this definition as "availability" in the remainder of this thesis. For a further discussion on the relation between availability and the spare part stock levels, we refer to section 2.3.4.

2.3 Model and assumptions

In this section, we first address the optimisation problem qualitatively (2.3.1) and we state the key model assumptions (2.3.2). Also, we give an overview of the general notation that we will use throughout this thesis (2.3.3). Finally, we give a formal mathematical formulation of the optimisation problem (2.3.4).

2.3.1 The optimisation problem

Two key performance indicators involved in our problem are the average availability of the installed base and the costs for the initial purchase of spare parts. We aim to make a trade-off between these two factors. Therefore, we can formulate our problem using two combinations of *goal function* and *restrictions*, namely (i) maximise the average availability of the installed base under a budget restriction for the initial purchase of spare parts, and (ii) minimise the budget spent for the initial purchase of spare parts under a constraint of a target value for the average availability. The VARI-METRIC method can deal with both variants, as the only difference is the stopping criterion of the algorithm, see section 2.4 for details. In practice, it is nice to show the (near-)optimal relation between average availability and budget in a graph as decision support for management.

The *decision variables* in VARI-METRIC are the spare part inventory positions for each item at each location, i.e. the number of items on order plus the number of items in repair or resupply minus the number of backorders. Note that we will extend the set of decision variables in Chapter 5 when we make a trade-off between spare part inventories and repair capacities. In that case, the optimisation model should include the costs of repair capacity.

Sherbrooke (1992) shows that, for *infinite* capacity repair shops, maximising the availability, the goal function of variant (i), is more or less equivalent to minimising the sum of the expected backorders of all highest indenture items at all most downstream locations. For the special case that every most downstream location serves exactly one system of the installed base, we can express the average system availability in the backorder probabilities rather than the expected backorders, see Rustenburg (2000). In any case, we must know the distribution of backorders at an operational level, or in other words the distribution of the highest indenture backorders at the most downstream echelon, to estimate the availability of the system.

We point out that the relation between availability and backorders is more complicated in the case of *finite* capacity repair shops. We will return on this issue in Chapter 4.

2.3.2 Assumptions

VARI-METRIC uses the following assumptions (cf. Sherbrooke, 1992, Rustenburg, 2000):

1. System failures occur according to a stationary Poisson process, so with constant failure rate.
2. Each assembly failure is caused by at most one subassembly failure.
3. Request for items are handled according to the First Come, First Serve (FCFS) rule at each stockpoint.
4. Each stockpoint has exactly one supplier; there is no lateral supply from other stockpoints.
5. Each stockpoint uses an (S-1, S) one-for-one inventory replenishment rule for each item.

6. The assignment of jobs to repair shops is only based on technical issues, such as the failure cause, the skills of the repair men and the availability of equipment; this is represented by a discrete probability distribution.
7. The item repair times are independent, identically distributed random variables.
8. The repair shops handle their jobs according to the FCFS discipline (i.e. no repair priorities).
9. After repair, the items are as good as new.
10. All item failures cause system failures, so all items are considered equally critical.

Although one might question any of these assumptions, they are commonly accepted to make the analysis tractable. For a discussion on the validity of the assumptions, we refer to Rustenburg (2000), who makes a comparison with the situation at the Royal Netherlands Navy. On going research focuses on relaxing one or more of the basic assumptions, see Chapter 1. In this thesis, we will relax the assumptions 7 and 8.

Assumption 7 refers to the independence of repair times if the repair capacities are infinite, so if the repair shops are modelled by $M/G/\infty$ queues. In practise, one uses repair shop *throughput* times (including possible delay) as repair times. In this thesis, we explicitly model the repair capacity, so that we can distinguish net repair times and waiting time. We still assume that the net repair times are independent, but the repair shop throughput times of various items will be correlated because waiting times are interrelated. Additionally, we will assume that repair times are exponentially distributed to keep the analysis tractable.

First, we shall also use Assumption 8 when developing models with finite repair capacity. In Chapter 8, when we examine the impact of repair priorities, we relax this assumption.

2.3.3 Notation

Below, we discuss the basic notation that we will use in the remainder of this thesis. Additional notation will be introduced when we use it. As reference, an overview is given in Appendix A.

Geographical structure

m	=	location index, $n = 1, \ldots, N^{loc}$, where N^{loc} denotes the total number of locations in the system;
n	=	echelon index, $n = 1, \ldots, N^{ech}$, where N^{ech} denotes the total number of echelons in the system; here *n* = 1 (*n* = N^{ech}) denotes the most upstream (downstream) level;
ECH(n)	=	set of locations corresponding to echelon *n*;
CUS(m)	=	set of customers (locations from which replenishment orders are accepted) of location *m*; as the most downstream echelon supports the operational field, we say that $CUS(m) = \varnothing$ if $m \in ECH\left(N^{ech}\right)$;
SUP(m)	=	the location that supplies items to location *m* (i.e. the direct supplier of location *m*);

Product structure

j	=	item index, *j* = 1,..,N^{item}, where N^{item} denotes the total number of items in the system;
i	=	indenture index, *i* = 0,…,N^{ind} ; here *i*=0 denotes the system level and *i* = N^{ind} denotes the lowest subassembly level, i.e. the subassemblies that are not decomposed in smaller units;
IND(i)	=	set of items belonging to indenture *i*.
SA(j)	=	set of subassemblies of item *j*, where $SA(j) = \varnothing$ for all $j \in IND\left(N^{ind}\right)$.
AS(j)	=	set of assemblies that have item *j* as a subassembly, where $AS(j) = \varnothing$ for all $j \in IND(1)$.
Z_j	=	the multiplicity of product *j* (i.e. the number of class *j* items in a single system in the installed base)

Routing characteristics

p_{mj}^{rep}	=	probability that item *j* can be repaired at location *m*; the item is dispatched to the next higher level in the multi-echelon structure with probability (1 - p_{mj}^{rep});

p_{mjk}^{cause} = probability that a failure of item *j* at location *m* is caused by a failure of subassembly *k* (*k*∈*SA(j)*); item *j* should be repaired itself at location *m* with probability $1-\sum_{k\in SA(j)} p_{mjk}^{cause}$;

Item data

B_m = the size of the installed base at location *m*

st_{mj} = stock level of item *j* at location *m*, i.e. the number of items on hand plus the number in the pipeline minus the backorders; $\overline{st}$ denotes the matrix of all stock levels in the system.

c_j = price of item *j*.

Repair and transport characteristics

λ_{mj} = demand rate for item *j* at location *m*

STm_{mj} = repair time, a random variable with mean E[STm_{mj}] and coefficient of variation $C_{S_{mj}}$, where the coefficient of variation is defined as the ratio between the standard deviation and the mean

O_{mj} = order-and-ship time of item *j* to location *m* from its supplier, a random variable with mean E[O_{mj}] and coefficient of variation $C_{O_{mj}}$; for *m*∈*ECH*(*1*), these variables represent the external procurement / repair time.

System state

Q_{mj} = number of type *j* items in the repair queue at location *m*.

R_{mj} = number of type *j* under repair (in queue and in service) at location *m*.

PL_{mj} = number of type *j* items in the pipeline at location *m*, i.e. all items *j* under repair at location *m*, all items *j* on order at the supplier of location *m* and all items *j* waiting at location *m* for subassembly replacement.

$BO_{mj}\left(\overline{st}\right)$ = number of backorders for item *j* at location *m* as a function of the stock levels $\overline{st}$;

$$BO_{mj}\left(\overline{st}\right)=\max\left(PL_{mj}-st_{mj},0\right)$$

2.3.4 Mathematical formulation

To formulate the optimisation problem, we have to express the average availability in the decision variables, being the inventory positions $\overline{st}$. For the supply availability, we have that a system is available if all products (highest indenture items) are available. Sherbrooke (1992) shows that maximising this availability function is approximately equivalent to minimising the sum of the expected backorders for single site systems and if the repair shops have *infinite* capacity. We will now give an outline of the derivation of availability in terms of expected backorders for general multi-echelon systems.

Suppose that at a certain point in time the number of backorders for product $j \in IND(1)$ at a most downstream location $m \in ECH(N^{ech})$ equals $BO_{mj}\left(\overline{st}\right)$. Noting that $B_m Z_j$ products need to be operational for the installed base at location *m* to be fully available, we find that a single system in the installed base at location *m* is waiting for product *j* with approximate probability $BO_{mj}\left(\overline{st}\right)\Big/\left(B_m Z_j\right)$, provided that the numerator is much smaller than the denominator. Now let us assume that a system is only available if all products are available, so it is a serial system in reliability terminology. Then, given the specific state as described by the number of backorders, we find that the system is available (i.e., not waiting for any product) with probability

$$A_m\left(\overline{s}\right) = \prod_{j \in IND(1)} \left\{1 - BO_{mj}\left(\overline{st}\right)\Big/\left(B_m Z_j\right)\right\}^{Z_j} \tag{2.1}$$

By taking the expectation of (2.1), we get the average availability of all systems at downstream location *m*. We find the average availability of the entire installed base by averaging over all most downstream locations and by taking the expectation:

$$A\left(\overline{st}\right) = E\left[\frac{\sum\limits_{m \in ECH(N^{Ech})} B_m A_m\left(\overline{st}\right)}{\sum\limits_{m \in ECH(N^{ech})} B_m}\right] \tag{2.2}$$

To arrive at the same result as Sherbrooke (1992), we use a modification of Rustenburg's (2000) derivation[1]. Using the shorthand notation $B = \sum_{m \in ECH(N^{ech})} B_m$, we can write (2.2) as

$$A = \frac{1}{B} E\left[B - \sum_{m \in ECH(N^{ech})} B_m (1 - A_m) \right]$$

Noting that $\ln(1-x) \approx -x$ for x small and that the target value for 1 - A_m is generally small, we can rewrite this equation as

$$A\left(\overline{st}\right) \approx 1 + \frac{1}{B} \sum_{m \in ECH(N^{ech})} B_m E \ln\left[A_m \left(\overline{st}\right) \right] \tag{2.3}$$

From equation (2.1), we find that

$$\ln(A_m) = \sum_{j \in IND(1)} Z_j \ln\left[1 - BO_{mj}\left(\overline{st}\right) \Big/ \left(B_m Z_j\right) \right]$$

Applying again that $\ln(1-x) \approx -x$ for x small, we have

$$\ln\left[A_m \left(\overline{st}\right) \right] \approx -\sum_{j \in IND(1)} BO_{mj}\left(\overline{st}\right) \Big/ B_m \tag{2.4}$$

Combining (2.3) and (2.4) gives

$$A\left(\overline{st}\right) \approx 1 - \frac{1}{B} \sum_{m \in ECH(N^{ech})} \sum_{j \in IND(1)} E\left[BO_{mj}\left(\overline{st}\right) \right] \tag{2.5}$$

So, we find that maximising the average system availability is approximately equivalent to minimising the sum of the expected backorders of all highest indenture assemblies at all most downstream locations. Therefore, we can take the following goal function:

[1] See Section 3.4 of his thesis

$$\min_{\underline{st}} \sum_{m \in ECH(N^{ech})} \sum_{j \in IND(1)} E\left[BO_{mj}\left(\overline{st}\right)\right] \tag{2.6}$$

Another availability function was introduced by Rustenburg (2000) in a NAVY case. In this application, the most downstream locations are frigates and the failure of any unit implies that the frigate is not available. In fact, the size of the installed base at each most downstream location is exactly one ($B_m = 1$). For this case the availability function can be defined as:

$$A = \frac{1}{|ECH(N^{ech})|} \sum_{m \in ECH(N^{ech})} \prod_{j \in IND(1)} \left\{1 - \mathrm{P}\left[BO_{mj}\left(\overline{st}\right)\right]\right\} \tag{2.7}$$

where $|ECH(N)|$ denotes the number of elements in the set $ECH(N)$ and where we use the shorthand notation $PBO_{mj}\left(\overline{st}\right) = P\left(BO_{mj}\left(\overline{st}\right) > 0\right)$ for convenience. Rustenburg (2000) shows that maximising this availability function is approximately equivalent with minimising the sum of the backorder probabilities. So an alternative goal function is

$$\min_{\underline{st}} \sum_{m \in ECH(N^{ech})} \sum_{j \in IND(1)} P\left[BO_{mj}\left(\overline{st}\right)\right] \tag{2.8}$$

Now we can state the formal optimisation problems:

<u>Variant (i), maximise the average availability subject to a budget constraint:</u>

$$\begin{aligned} &\min_{\underline{st}} \sum_{m \in ECH(N^{ech})} \sum_{j \in IND(1)} E\left[BO_{mj}\left(\overline{st}\right)\right] \\ &\text{subject to} \\ &\sum_{j \in IND(1)} c_j \sum_{m \in ECH(N^{ech})} st_{mj} \leq BUDGET \\ &st_{mj} \in \mathbb{N} \end{aligned} \tag{2.9}$$

where as a sub-variant (i)', we can replace the goal function by (2.8) for Rustenburg's (2000) model.

Variant (ii), minimise inventory investment subject to a constraint on the average availability:

$$\min_{\underline{st}} \sum_{j \in IND(1)} c_j \sum_{m \in ECH(N^{ech})} st_{mj}$$

subject to

$$\sum_{m \in ECH(N^{ech})} \sum_{j \in IND(1)} E\left[BO_{mj}\left(\overline{st}\right)\right] \leq B(1 - A_{Target}) \tag{2.10}$$

$$st_{mj} \in \mathbb{N}$$

where as a sub-variant (ii)', we can modify the constraint using (2.7) for Rustenburg's (2000) model.

Of course, the expected backorders is a non-linear function of the stock levels $\overline{st}$. Therefore, we have a large non-linear integer optimisation problem for both variants. An exact solution procedure that is practical for large instances is not known. VARI-METRIC is a greedy optimisation heuristic to both variant (i) and variant (ii) and the corresponding sub-variants. The methodology that we will develop in this thesis is also suitable for both model variants and for both Sherbrooke's and Rustenburg's formulation. In the numerical examples that we will discuss, we will choose variant (ii)'.

2.4 The VARI-METRIC optimisation heuristic

The (VARI-)METRIC method analyses the relevant backorder distributions by subsequently deriving the pipeline and backorder distributions of various items and various locations, starting with the lowest indenture items at the most upstream location in the echelon structure. To facilitate this, the arrival rates of all items (assemblies and subassemblies) at all repair shops in the network have to be calculated. Below, we give an outline of the basic mathematics of VARI-METRIC.

2.4.1 Calculating the item arrival rates

The arrival rate λ_{mj} of item j at location m consists of two parts:

- the arrival rates of this item at downstream locations l (λ_l) multiplied by the probability that these items cannot be repaired at these down stream locations ($1 - p_{mk}^{rep}$)

- the arrival rates of higher indenture items (i.e. assembly *k* has item *j* as subassembly) at location *m* (λ_{mk}) multiplied by the probability that item *k* can be repaired at location *m* (p_{mk}^{rep}) and multiplied by the probability that item *j* is the cause of the failure (p_{mkj}^{cause})

Hence, we arrive at the expression

$$\lambda_{mj} = \sum_{l \in CUS(m)} \lambda_{lj} \left(1 - p_{lj}^{rep}\right) + \sum_{k \in AS(j)} \lambda_{mk} p_{mk}^{rep} p_{mkj}^{cause} \tag{2.11}$$

All arrival rates can be computed recursively, starting with the failure rates of the installed base, i.e. the highest indenture items (*j* ∈*IND(1)*) at the most downstream locations (*m* ∈*ECH(N^{ech})*) and next moving to lower indenture and more upstream levels, see Figure 2.7.

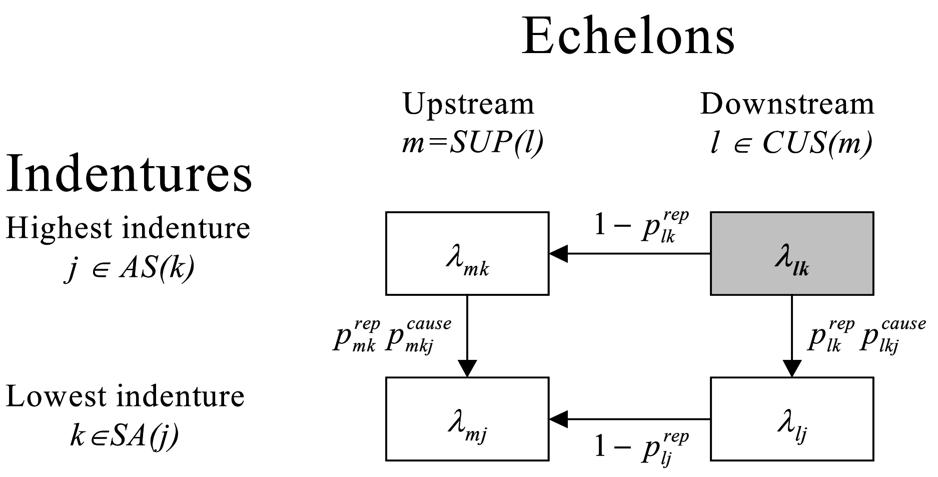

Figure 2.7. Calculation sequence of the item arrival rates; the grey box is the starting point

2.4.2 Calculating pipeline and backorder characteristics

Consider a subsystem with one repair facility and an inventory function at location *m*. The number of backorders of item *j* in such a subsystem is the non-negative difference between number of items in pipeline PL_{mj} and the number of items in stock st_{mj} :

$$BO_{mj} = \max\left\{PL_{mj} - st_{mj}, 0\right\} \tag{2.12}$$

Recall that the pipeline refers to all items in repair at location *m*, all items on order at the supplier of location *m* and all items waiting at location *m* for subassembly replacement. Hence, the pipeline distribution depends on the backorders upstream in the echelon structure and downwards in the indenture structure. We can derive an expression for the pipeline PL_{mj} by considering the three components separately:

- <u>All items *j* in repair at location *m*.</u> If the repair shop has infinite capacity, the mean number of items under repair is given $p_{mj}^{rep} E[R_{mj}] = \lambda_{mj} p_{mj}^{rep} E[STm_{mj}]$.
- <u>All items *j* waiting at location *m* for subassembly replacement</u>. The total number of items waiting for replacement of subassembly $k \in SA(j)$ equals the number of backorders BO_{mk}. Only a fraction h_{mjk} of the backorders for item *k* at location *m* is due to a request from item *j*. A reasonable approximation for this fraction is the effective demand rate for subassembly *k* arising from item *j* as fraction of the total demand rate for item *k* at location *m*: $h_{mjk} = p_{mk}^{rep} \lambda_{mj} p_{mkj}^{cause} / \lambda_{mk}$. So for this part of the pipeline, we arrive at the mean value $\sum_{k \in SA(j)} h_{mjk} E[BO_{mk}]$.
- <u>All items *j* on order at the supplier of location *m*</u>. These items include the items being transported to location *j* from its supplier ($\lambda_{mj}(1 - p_{mj}^{rep})E[O_{mj}]$) and the items waiting at the supplier $n = SUP(m)$ for replacement. The items waiting for replacement can be derived from the number of backorders for item *j* at the supplier *n*, BO_{nj}. Only a fraction f_{mj} of these backorders are destined for location *m*. This fraction can be calculated as the ratio between the demand for item *j* by location *m* and the total demand for item *j* at the supplier *n*: $f_{mj} = (1 - p_{mj}^{rep})\lambda_{mj} / \lambda_{SUP(m),j}$. So for this part of the pipeline, we arrive at the mean value $\lambda_{mj}(1 - p_{mj}^{rep})E[O_{mj}] + f_{mj} E[BO_{SUP(m),j}]$.

Putting it all together, we find the following expression:

$$E[P_{mj}] = \lambda_{mj} p_{mj}^{rep} E[STm_{mj}] + \sum_{k \in SA(j)} h_{mjk} E[BO_{mk}] + \lambda_{mj}(1 - p_{mj}^{rep})E[O_{mj}] + f_{mj} E[BO_{SUP(m),j}] \quad (2.13)$$

Similarly, an expression for the variance of the backorders can be derived as well. Then we arrive at (cf. Sherbrooke, 1992, Rustenburg, 2000):

$$\begin{aligned}Var[P_{mj}] = \lambda_{mj} p_{mk}^{rep} E[STm_{mj}] + \sum_{k \in SA(j)} \left\{ h_{mjk}(1-h_{mjk})E[BO_{mk}] + h_{mjk}^2 Var[BO_{mk}] \right\} \\ + \lambda_{mj}(1-p_{mk}^{rep})E[O_{mj}] + f_{mj}(1-f_{mj})E[BO_{SUP(m),j}] + f_{mj}^2 Var[BO_{SUP(m),j}]\end{aligned} \tag{2.14}$$

Equation (2.12) shows that the backorder characteristics can be calculated from the pipeline characteristics, whereas equation (2.13) and (2.14) show that the pipeline characteristics can be calculated from the backorder characteristics. This provides a way to obtain all backorder characteristics recursively, starting with the pipeline of the lowest indenture items (*k∈IND(N^{ind})*) at the most upstream location (*l∈ECH(1)*) and next moving to higher indenture and more down levels, see Figure 2.8. Hence the direction of pipeline and backorder calculations is reversed compared to the calculation of the item rates.

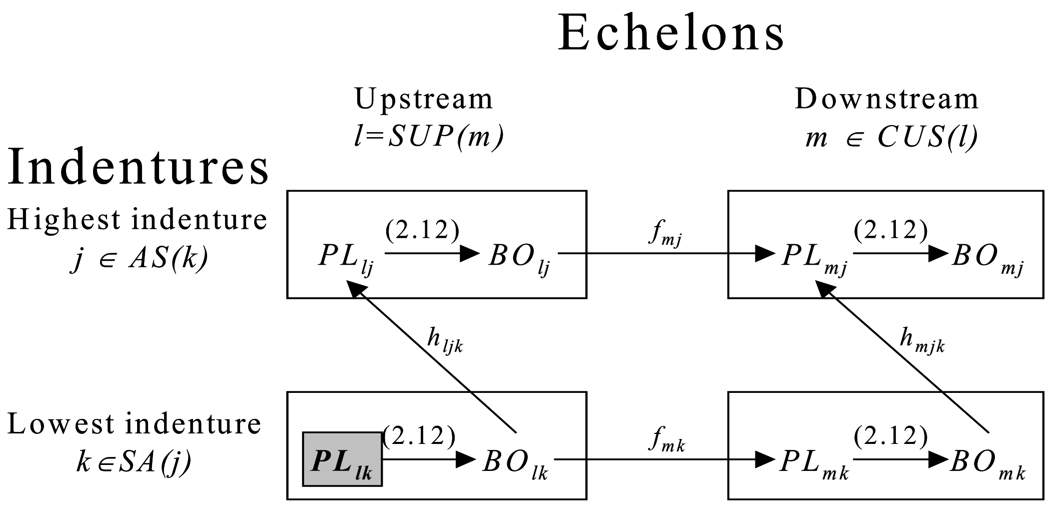

Figure 2.8. Calculation sequence of the backorders; the grey box is the starting point

As a simple approximation to obtain the backorder characteristics from (2.12), we only use the first two moments of the numbers of items in the pipeline and we fit a discrete distribution on the first two moments. The most general way to achieve this is given by Adan et al. (1996). Based on this approximate distribution for the pipeline, we can find the first two moments of the backorders. So, we use the following equations

$$E[BO_{mj}] = F_1\left(E[PL_{mj}], Var[PL_{mj}]\right)$$
$$Var[BO_{mj}] = F_2\left(E[PL_{mj}], Var[PL_{mj}]\right)$$

where the functions F_1 and F_2 are some specific functions whose structure depend on the mean and variance of the pipeline, see Adan et al. (1996) for details. When we have the mean and variance of the backorders, we can use the values for the mean backorders to calculate the mean pipeline for higher indenture items and for more downstream locations from (2.13) and (2.14) as shown in Figure 2.8. For the availability calculations, we need the backorders of all highest indenture items $j \in IND(1)$ at all most downstream locations $m \in ECH(N^{ech})$, see (2.5).

2.4.3 The VARI-METRIC algorithm

Based on the equations for the item arrival rates and the backorder characteristics, the VARI-METRIC algorithm uses a greedy rule to optimise item stock levels within a budget constraint. The idea is very simple: the system availability is related to the sum of the mean backorders of all highest indenture items ($i \in IND(1)$) at all most downstream locations ($m \in ECH\left(N^{ech}\right)$), see equation (2.5). Hence, a greedy rule adds an item i^* to stock at location n^* in each step ($\overline{st} + e_{i^* n^*}$, where e_{in} is a matrix with all elements equal to zero, except for element i, n which is equal to 1), such that the reduction in the total expected backorders per invested dollar is maximised. For Rustenburg's formulation, the equivalent rule is the maximisation of the reduction in the sum of the backorder probabilities. Stating the rule for Rustenburg's variant, the rule is to select (j^*, m^*) such, that the expression:

$$\Delta_{i^* m^*}^{stock} = \frac{\sum\limits_{j \in IDN(1)} \sum\limits_{m \in ECH(N^{ech})} PBO_{mj}\left(\overline{st}\right) - \sum\limits_{j \in IDN(1)} \sum\limits_{m \in ECH(N^{ech})} PBO_{mj}\left(\overline{st} + e_{j^* m^*}\right)}{c_{j^*}^{item}} \tag{2.15}$$

is maximised. Here we use as before the shorthand notation $PBO_{mj}\left(\overline{st}\right) = P\left(BO_{mj}\left(\overline{st}\right) > 0\right)$.

For Sherbrooke's formulation, we should simply replace the backorder probabilities by the expected backorders. Rustenburg (2000) shows that this greedy rule yields good results for two-echelon, two-indenture models, provided that the following nonnegative starting values are chosen for the stock levels:

$$\begin{aligned} st_{jm} &:= round\left[\tfrac{1}{2}\lambda_{jm}\left(p_{mj}^{rep} E[STm_{jm}] + (1 - p_{mj}^{rep}) E[O_{jm}]\right)\right] \quad \forall j, m \in ECH(1) \\ st_{jm} &:= round\left[\lambda_{jm}\left(p_{mj}^{rep} E[STm_{jm}] + (1 - p_{mj}^{rep}) E[O_{jm}]\right)\right] \quad \text{otherwise} \end{aligned} \tag{2.16}$$

Summarised, the VARI-METRIC method works as follows:

VARI-METRIC: inventory optimisation for variant (i), formulation Rustenburg

Step 1. Initialise the stocks levels according to (2.16). Set the budget spent $\hat{C}$ equal to total costs of the initial stock levels: $\hat{C} = \sum_j \sum_m c_{mj} st_{mj}$;

Step 2. Calculate the demand rates of all items at all locations according to (2.11);

Step 3. Recursively calculate the first two moments of the pipeline and the backorders from (2.12) - (2.14);

Step 4. Calculate Δ_{mj}^{stock} for all combinations of item and location according to (2.15) and determine the maximum of all the increments Δ_{mj}^{stock}; say that this maximum is obtained for item j^* at location m^*;

Step 5. $st_{j^*m^*} := st_{j^*m^*} + 1;\ \ \hat{C} := \hat{C} + c_{j^*}^{stock}$

Step 6. If $\hat{C} < \text{budget}$, then go to Step 3, else STOP

In a similar way, we can define an algorithm for variant (ii): minimise the budget for parts and repair capacity given a target availability. Then, we should only modify the stop criterion in step 6. Instead of the budget restriction, we use a restriction on the average system availability:

$$A\left(\overline{st}\right) = \frac{1}{\left|ECH(N)\right|} \sum_{m \in ECH(N^{ech})} \prod_{j \in IND(1)} P\left(BO_{mj}\left(\overline{st}\right) > 0\right) \geq A_{Target}$$

2.5 The repair shop model

2.5.1 The queueing model

It is natural to model repair facilities as queues, where we analyse each repair shop independently. However, even modelling a single repair facility is not an easy task, because their operations are complex in practice. A variety of issues is relevant, such as the availability of multiple resources (machines, personnel, tools), flexibility of capacity (outsourcing, overtime), planning (preventive maintenance), job priorities, job routing, etc. A detailed model leads to an optimisation problem that is very hard to solve. Besides, we are interested in the impact of finite capacities and repair priorities on *tactical* decisions as initial inventory investment and repair capacities. Therefore, we choose for a less detailed model capturing the main issues from finite repair capacity. From the preceding, we derive the following minimum requirements for our model:

1. the repair facility at a certain location can either be a generic facility that can handle all job types, or it consists of a set of dedicated repair shops; a dedicated repair shop is able to handle a subset of all repair jobs that arrive at a certain location;
2. each repair shop can have a single or several parallel channels to handle the jobs;
3. it should be possible to include repair priorities;
4. it should be possible to derive (good approximations for) the first two moments of the number of items in the repair shop; we need these characteristics to apply VARI-METRIC.

Above, we distinguished between repair shops and repair facilities. For clarity, we give the following definitions:

> *a* ***repair shop*** *is a resource to handle a subset of repair jobs with either one channel or multiple identical parallel channels;*
> *a* ***repair facility*** *is the set of all repair shops at a certain location; the repair shops belonging to a single repair facility may be different;*

The first three requirements lead us to multi-class, multi-server models with priorities. As we have seen in section 1.3.5, there are no analytic results for these models readily available in the literature. Moreover, it is not sufficient to derive (approximations for) standard performance characteristics as the mean waiting time and the mean queue length. Therefore, we are confronted with a complex queueing analysis task. To be able to derive the performance estimators needed, we choose to model repair times by exponential distributions. Although this is not a general model (we cannot choose the repair time variance), it can provide us with a good indication of the impact of finite repair capacities and repair priorities. Therefore, we choose to model repair shops by *multi-class M/M/k queues*, in a later stage to be extended with priorities.

Although our repair shop model provides us with quite some flexibility, we are aware that we ignore some relevant aspects for the tactical decisions. Particularly, we mention preventive maintenance done by the repair men from the repair shops (these jobs can be planned to some extent) and flexibility of repair capacity (overtime, outsourcing). Although these aspects can have a significant impact on the results, we choose to leave many of these issues for further research. Only outsourcing will be considered up to a certain degree in Chapter 7. We simply have to be realistic that not all issues can be solved at once.

2.5.2 Job classes

The former discussion leaves us with the question how to define job classes. For simplicity, we choose *item types* for the class structure. Of course, it is possible that repair shops are dedicated to specific job types that do not coincide with the item classification. This has impact on the item arrival rates at the repair shops and on the pipeline calculation. In principle, we can model this by introducing a (discrete) probability distribution for the assignment of a specific item type to a specific repair shop at a fixed location. To this end, we introduce the following definitions:

h	=	repair shop index, $h = 1,\ldots,N^{shop}$, where N^{shop} denotes the total number of repair shops in the system;
SH(m)	=	set of repair shops at location *m*;
p_{mjh}^{shop}	=	probability that item *j* will be repaired by repair shop *h*, given that it will be repaired at location *m*;
λ_{mjh}	=	arrival rate of item *j* at repair shop *h* at location *m*.

Now we can modify the arrival rate calculation (2.11) simply using the discrete probability distribution over all repair shops at location *m*:

$$\lambda_{mjh} = p_{mjh}^{shop} \sum_{l \in CUS(m)} \lambda_{lj} \left(1 - p_{lj}^{rep}\right) + p_{mjh}^{shop} \sum_{k \in AS(j)} \lambda_{mk} \, p_{mk}^{rep} \, p_{mkj}^{cause}$$

Further, we can modify the pipeline calculation as follows (compare section 2.4.2). Distinguishing between various repair shops $h \in SH(m)$ at location *m*, we redefine R_{hj} as the number of items of type *j* in repair shop *h* (in repair or waiting for repair). Then, the mean number of items of type *j* in repair at location *m* equals $p_{mj}^{rep} \sum_{h \in SH(m)} p_{mjh}^{shop} E[R_{hj}]$. Replacing this first component in equation (2.13) is sufficient to model a more generic job class structure. A similar modification can be done in a straightforward way for the variance of the number of items in the pipeline (2.14). Although it is no problem to define job classes in a more general way, we will use item types as job classes in the remainder of this thesis for the sake of simplicity of presentation.

2.5.3 Job priorities

Regarding job priorities, it is logical to assign priorities to item types. The main reason is our focus on the tactical decision on initial inventory levels. If we wish to influence these inventory levels using priorities, it is reasonable to assign these priorities to items based on item characteristics only, such as the price, the failure rate and the repair time. In principle, we can assign a separate priority to each item then. For simplicity of the analysis, we choose for *two priority classes* only in this thesis. This means that items should be given either high or low priority.

The priority assignment may vary between repair shops, so an item may have high priority in one repair shop and low priority in the other one. Although this solution does not seem to be logical at first sight, we leave this open. After all, the situation at central and local level may be entirely different in terms of assignment of items to repair shops (dedicated or generic repair shops), repair times, item arrival rates and repair capacities.

Given job priorities, we still need to decide upon the queueing discipline. That is, how is a high priority job handled that finds all servers occupied upon arrival, some of them with low priority jobs? The common choices are:

1. the high priority job waits until the next server comes available; this is the non-preemptive discipline;
2. if any low priority job is in service, the high priority job is taken into service immediately, thereby postponing the low priority job; this is the preemptive discipline; for this option, we have to make an assumption on the continuation of the low priority job:

a. preemptive resume: when capacity is available for the postponed low priority job, it is taken into service, and the new service time equals the remaining service time when the job was interrupted;
b. preemptive restart: the new service time equals the complete repair time, so we assume that the repair man has to start the job all over again.

Because repair jobs may have long throughput times and to strengthen the difference between low and high priority jobs, we choose for *preemptive priorities*. A more practical reason is that this queueing model variant is easier to analyse, as we will show in Chapter 6. Because we assume exponential repair times, the preemptive resume and preemptive restart variants are identical from a mathematical point of view.

2.5.4 Overview of assumptions

Summarising, we will use the following assumptions for our finite capacity repair shops, additionally to the VARI-METRIC assumptions (section 2.3.2),

1. at every location, there is at most one repair facility, consisting of one or more repair shops; each repair shop is dedicated to a subset of items;
2. at each location, there is at most one repair shop where a certain item type can be repaired;
3. the repair capacity is constant;
4. repair times are independent and exponentially distributed random variables;
5. priorities, if any, are assigned as either high or low priority, so we have two priority classes only; priorities are assigned to items for each repair shop separately;
6. in case of priorities, the queueing discipline is preemptive resume.

With respect to the last assumption, note that we will consider non-preemptive priorities as well in Chapter 7.

2.6 Summary

In this chapter we have introduced the spare parts supply model used in this thesis along with the main definitions and notations. We gave an outline of VARI-METRIC and discussed the repair shop model to be used in the remainder of this thesis. We choose to model the spare part network with finite repair capacities as a network with one or more repair shops per location. We model every repair shop as a multi-class *M/M/k* queue. In the case of repair priorities, we use two priority classes with a fixed assignment of items to priority classes per repair shop and a preemptive resume queueing discipline. Herewith, we answered research question 1 from the introduction.

Chapter 3
Multi-Class *M/M/k* queue

Abstract

Multi-class multi-server queueing problems are a generalisation of the well-known M/M/k situation to arrival processes with clients of N types that require exponentially distributed service with different average service times. Problems of this sort arise naturally in various applications, such as spare parts management, for example. In this chapter, we give a procedure to construct exact solutions of the stationary state equations. Essential in this procedure is the reduction of the problem for n (= the number of clients in the system) > k to a backwards second order difference equation with constant coefficients for a vector in a linear space with dimension depending on N and k, denoted by d(N, k). Precisely d(N, k) of its solutions have exponential decay for n $\rightarrow \infty$. Next, using this as input, the equations for n $\leq$ k can be solved by backwards recursion. It follows that the exact solution does not have a simple product structure as one might expect intuitively. Further, using the exact solution, several interesting performance measures related to spare parts management can be computed and compared with heuristic approximations. This is illustrated with numerical results. The algorithm as developed in this chapter is the answer on our second research question as stated in Chapter 1.

This chapter is based on the paper:

“On multi-class multi-server queueing and spare parts management.” A. van Harten, A. Sleptchenko, BETA publication (WP-49), 2000. Submitted for publication.

3.1 Introduction

Multi-class, multi-server (MCMS) models arise for stochastic processes where clients with different service characteristics ask for service capacity from a non-differentiated resource with finite capacity. Such a resource can be a spare parts repair shop as described in the previous chapter. The main questions that we will address here are:

1. How do we construct exact/approximate solutions for such models?
2. How do we use these solutions for improved design and management of spare part systems?

The analysis of multi-class multi-server problems starts as a generalisation of the well-known *M/M/k* queue. The theory of the *M/M/k* queue is dealt with in any good textbook on operations research, such as Winston (1997). In the more general MCMS context, the *M/M/k* queue can be considered as a special case where the classes of clients are indistinguishable with respect to their service characteristics. Then, the service rate μ_i of class i is equal to the average service rate μ. As a matter of fact, all classes can be considered to merge into one class only. In the general case with $N > 1$ classes, μ_i has a different value for each class and $\mu_i = \mu(1+\delta_i)$ where δ_i measures the strength of the perturbation relative to the average service rate μ.

Compared with previous work on multi-class multi-server queues (Cohen, 1982, de Smit, 1983a, 1983b), our queueing discipline is different and our approach is more straightforward, since the stationary state equations in our multi-class problem are solved explicitly. This provides us with more detailed information on class-dependent performance measures. Such information is necessary in spare parts management applications. In our work, there is a first-come-first-serve (FCFS) service discipline for jobs from different classes of clients. In spare parts management, this is consistent with the usual way of thinking.

The remainder of this chapter is structured as follows. First, we give an outline of our approach in section 3.2. Next, we discuss the steady state equations (section 3.3). Then, we solve these equations explicitly for the special cases of equal service rates (section 3.4). We discuss the solution of the general model in section 3.5. There, we provide both an exact and an approximate solution. In section 3.6, we derive relevant performance measures based on the steady state probabilities. Numerical results are discussed in section 3.7. Finally, we give our conclusions in section 3.8.

3.2 Approach

Let us now briefly sketch our approach. Here we focus on the stationary probability distribution over the states of a general MCMS system. The state definition has to take into account the types of jobs in execution and in the queue. The state equations can be given explicitly in the general case, see section 3.3. For unperturbed case, when service times of jobs are equal, an analytical solution can be found, see section 3.4. Only a few structural aspects of this solution survive the perturbation of service times. It is crucial that in the general case, the distribution over the ordering in the queue keeps a product structure based on the arrival fractions of the client classes. Using this structure for *n* (= the number of clients in the system) > *k*, we obtain a much more transparent structure for the state equations. For $n > k$, they reduce to a backwards second order difference equation for a vector in a linear space of dimension $d(N,k) = \frac{(N+k-1)!}{k!(N-1)!}$. For example, for $N = 3$ and $k = 4$ we find $d(N,k) = 15$. Of course, this dimension increases rapidly with *N* and *k*. Now the difference equations can be solved in terms of eigenvalues and (generalised) eigenvectors of the iteration matrix. All eigenvalues and eigenvectors can be determined explicitly if the service times of jobs are equal. Precisely $d(N,k)$ of the eigenvalues are real and > 1 in the unperturbed case, which corresponds with decay for $n \to \infty$. This property survives the perturbation of the service times. It is the clue to solve the state equations, first for $n > k$ with the start vector for $n = k$ as an unknown and next, again with backwards recursion, also for $n < k$, until only one scaling constant, the probability of the empty state, remains. That scaling constant follows from the fact that the state probabilities sum to 1.

Using this procedure, the exact solution follows, see section 3.5, and its structure can be analysed. Next, coming back to the second question in the beginning, all sorts of interesting performance criteria for MCMS systems can be determined exactly, see section 3.6. Examples of such performance indicators are the expected number of items of type *i* in the system or in queue, its variance, the expected number of backorders of type *i*, its variance, the expected waiting time for clients in a certain class. Their behaviour is discussed in section 3.7.

To conclude this introduction, we summarise the main relevance of these results for MCMS queues for process design and - management questions:

– More accurate insight in the performance differences between different classes of clients

– Better insight in the relation between capacity and performance when several classes of clients compete for capacity, hence correlation between classes occurs.

In spare parts management, both issues play an important role. Potential applications are also found in other areas, such as manufacturing and factory layout.

3.3 Stationary state equations for MCMS problems

In Figure 3.1, we visualise the *N* different job types originating from multiple sources by arrows with different line types, and their flow through the *k* identical servers.

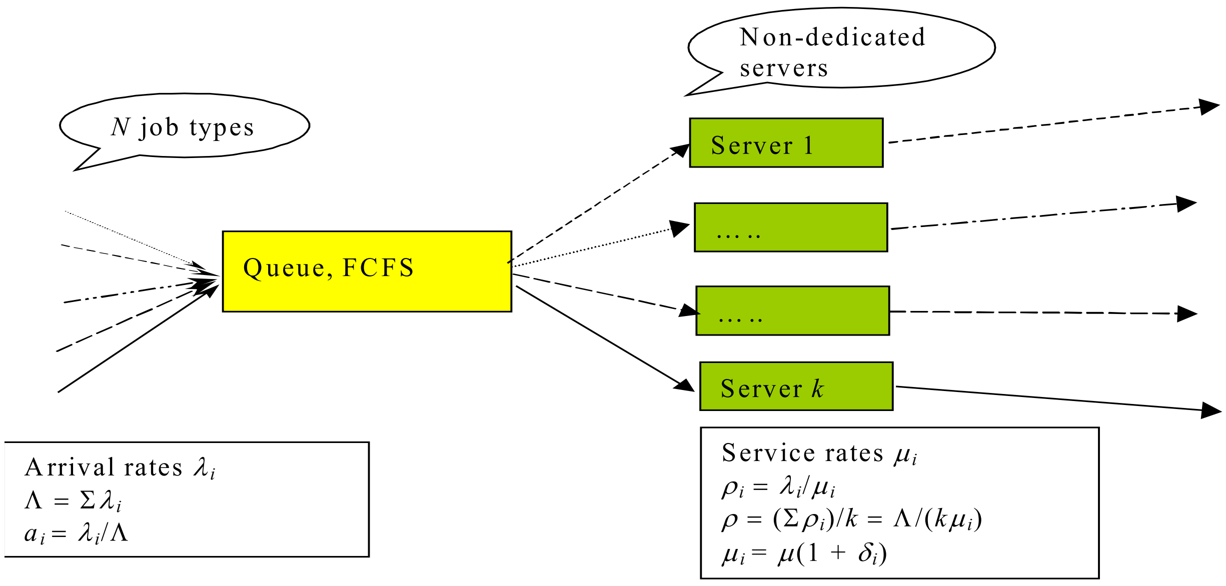
Non-dedicated servers
N job types
Server 1
.....
.....
Server k
Queue, FCFS
Arrival rates λ_i
$\Lambda = \Sigma\lambda_i$
$a_i = \lambda_i/\Lambda$
Service rates μ_i
$\rho_i = \lambda_i/\mu_i$
$\rho = (\Sigma\rho_i)/k = \Lambda/(k\mu_i)$
$\mu_i = \mu(1 + \delta_i)$

Figure 3.1. Flows of jobs through a MCMS queueing system

Jobs with type *i* arrive according to a Poisson process with arrival rate λ_i. Due to the non-dedicatedness of the servers, the flow through a server consists of all job types. The service distribution of job type *i* is exponential with rate μ_i and it is the same for all servers. The service discipline by which jobs are assigned is first-come-first-serve (FCFS) and if more than one server is available, say *k* of them are available, then each of them has an equal probability *1/k* to get the next job. Note that the latter assumption is only used to derive the state equations. Other server selection criteria (e.g. lexicographically) should lead to the same expressions for the performance measures.

The total arrival rate is given by $\Lambda = \sum \lambda_i$. We denote the arrival fraction of class *i* by $a_i = \lambda_i / \Lambda$. The utilisation is represented by ρ, hence the stability condition for this queueing system is given by $\rho < 1$. The average service rate μ is defined by $1/\mu = \sum a_i/\mu_i$. After some simple calculations we obtain the equivalent expression $\mu = \Lambda/(k\rho)$ as given in the figure. The notation for the relative perturbation from the average service rate is δ_i as already introduced in the introduction.

To describe state of the queueing system, we will use two vectors *w* and *s* of dimension *N*, i.e. each component *i* contains information about the amount of items of each class *i* in queue or in service, correspondently. Because of the PASTA property (Poisson Arrivals See Time Averages), these vectors will provide us all the information about the system that we need. Let us now have a closer look at the stationary state equations of the MCMS system.

The state equations follow from a micro-balance reasoning illustrated in Figure 3.2. The net exchange of probability in an infinitesimal time interval from a given state with its neighbours has to be zero in an equilibrium situation. Neighbours of a state $(\overline{w}, \overline{s})$ with *n* clients are states to or from which a one step transition is possible, either by an arrival event (A) or a service completion event (C).

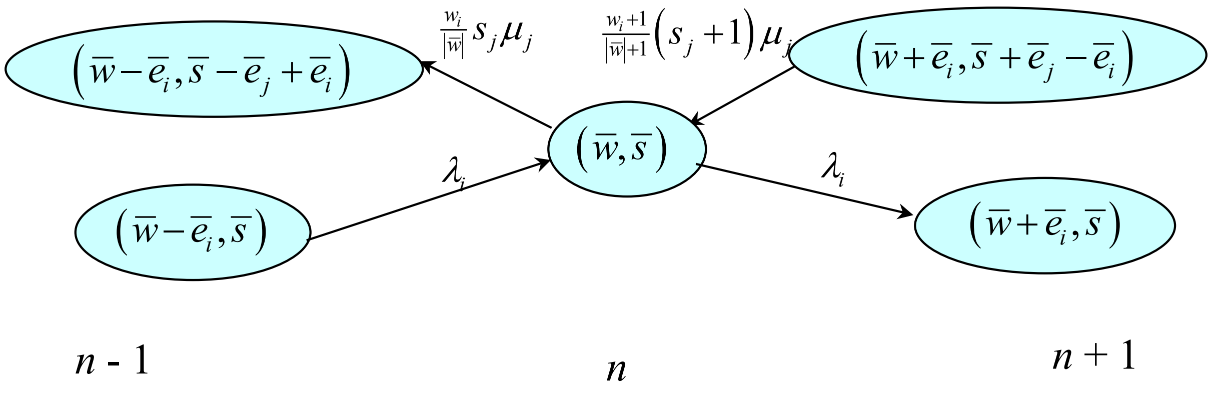
$\frac{w_i}{|\overline{w}|} s_j \mu_j$
$\frac{w_i+1}{|\overline{w}|+1}(s_j+1)\mu_j$
$(\overline{w}-\overline{e}_i, \overline{s}-\overline{e}_j+\overline{e}_i)$
$(\overline{w}+\overline{e}_i, \overline{s}+\overline{e}_j-\overline{e}_i)$
$(\overline{w}, \overline{s})$
λ_i
λ_i
$(\overline{w}-\overline{e}_i, \overline{s})$
$(\overline{w}+\overline{e}_i, \overline{s})$
$n - 1$
n
$n + 1$

Figure 3.2. The micro-balance between a state and its neighbours leads to the stationary state equations.

Neighbour states have $n-1$ or $n+1$ clients. If $n > k$, transition from/to neighbours $(\overline{w}', \overline{s}')$ with $n+1$ clients are specified by one of the following events:

(a) transitions from $(\overline{w}, \overline{s})$ to $(\overline{w}', \overline{s}')$: a job arrives and the queue vector is changed as $\overline{w}' = \overline{w} + \overline{e}_i$ if the job is of type *i*, where $\overline{e}_i$ is a vector with all zero components except, for component *i*, which is 1.

(b) transitions from $(\overline{w}',\overline{s}')$ to $(\overline{w},\overline{s})$: a service has been completed; if the service of job *j* was completed and job *i* was in front of the queue before completion, then the state $(\overline{w}',\overline{s}')$ is specified by $\overline{w}'=\overline{w}+\overline{e}_i$ and $\overline{s}'=\overline{s}+\overline{e}_i-\overline{e}_j$. The probability to have job *i* in front of the queue is equal to the number of jobs of type *i* in the queue before transition $w_i + 1$ divided by the total number of jobs in queue before transition $|\overline{w}|+1$: $\frac{w_i+1}{|\overline{w}|+1}$.

Analogously, if $n > k$, then transitions from/to neighbours with $n-1$ clients are specified by:

(a) transitions from $(\overline{w},\overline{s})$ to $(\overline{w}',\overline{s}')$: a job *j* has been completed by one of the servers and a job in front of the queue moves to server; if the job in front of the queue was a job of type *i* (with probability $\frac{w_i}{|\overline{w}|}$), then we have $\overline{s}'=\overline{s}-\overline{e}_j+\overline{e}_i$; since job *i* was in the queue before completing of job *j*, the vector $\overline{w}'$ is $\overline{w}'=\overline{w}-\overline{e}_i$.

(b) transitions from $(\overline{w}',\overline{s}')$ to $(\overline{w},\overline{s})$: an arrival of a job with type *i,* then $\overline{w}'=\overline{w}-\overline{e}_i$ and $\overline{s}'=\overline{s}$;

The characterisation of the neighbours for states with $n \le k$ can be done analogously, the details on the neighbours are self-evident from the terms in the stationary state equations given below.

Before stating the stationary state equations, we first introduce the following notation. We used already notation $|\overline{w}|$ as the total number of items in the queue, analogously we use $|\overline{s}|$ as total number of items in service. The next expressions are obvious:

$$|\overline{w}|+|\overline{s}|=n;\quad |\overline{w}|=0,\ \text{if}\, n\le k;\quad |\overline{s}|=k,\ \text{if}\, n\ge k.$$

The vector $\overline{e}_i$ with all zero components except for the component *i*, which is 1, has also been used before. In addition to this vector, we define e_{ij} as 1 if $i = j$ and 0 otherwise. Also, we define:

$$\delta(\overline{s}) \overset{def}{=} \frac{1}{k}\sum_{i=1}^{N} s_i\delta_i$$

and now we can formulate the stationary state equations, dividing the transition rates by the average service rate μ and the number of servers *k*:

- $n > k$

$$\left(1+\rho+\delta(\overline{s})\right)P(\overline{w},\overline{s}) = \rho\sum_{i=1}^{N} a_i P(\overline{w}-\overline{e}_i,\overline{s}) + \tfrac{1}{k}\sum_{i=1}^{N}\sum_{j=1}^{N}\frac{w_i+1}{|\overline{w}|+1}\left(1+\delta_j\right)\left(s_j+1-e_{ij}\right)P\left(\overline{w}+\overline{e}_i,\overline{s}-\overline{e}_i+\overline{e}_j\right) \tag{3.1}$$

- $n < k$

$$\left(\tfrac{n}{k}+\rho+\delta(\overline{s})\right)P(0,\overline{s}) = \rho\sum_{i=1}^{N} a_i P(0,\overline{s}-\overline{e}_i) + \frac{1}{k}\sum_{j=1}^{N}\left(1+\delta_j\right)\left(s_j+1\right)P\left(0,\overline{s}+\overline{e}_j\right) \tag{3.2}$$

- $n = k$

$$\left(1+\rho+\delta(\overline{s})\right)P(0,\overline{s}) = \rho\sum_{i=1}^{N} a_i P(0,\overline{s}-\overline{e}_i) + \frac{1}{k}\sum_{i=1}^{N}\sum_{j=1}^{N}\left(1+\delta_j\right)\left(s_j+1-e_{ij}\right)P\left(e_i,\overline{s}-\overline{e}_i+\overline{e}_j\right) \tag{3.3}$$

- $n = 0$

$$\rho P(0,0) = \frac{1}{k}\sum_{j=1}^{N}\left(1+\delta_j\right)P\left(0,\overline{e}_j\right) \tag{3.4}$$

Using the structure of the arrival process, the equation (3.1) for $n > k$ can be further reduced. For $n-k$ consecutive arrivals, the probability distribution over the possible vectors of the queue state $\overline{w}$ is given by the product of the arrival fractions and by the number of possible arrival sequences within the vector $\overline{w}$. As a consequence, we may write the steady state distribution $P(\overline{w},\overline{s})$ as:

$$P(\overline{w},\overline{s}) = P_n(\overline{s})|\overline{w}|!\prod_{i=1}^{N}\frac{a_i^{w_i}}{w_i!} \tag{3.5}$$

where the unknown vector $P_n(\overline{s})$ represents the stationary probability distribution over the server states $\overline{s}$, given that the system contains *n* jobs (in service plus in queue). A key issue in the remainder of this chapter will be how to derive an expression for the $d(N,k)$ - vector $\mathbf{P}_n$ with components $P_n(\overline{s})$.

Firstly, we note that substitution of (3.5) in (3.1) gives us the following equation for $n > k$:

$$\begin{aligned}\left(1+\rho+\delta(\bar{s})\right)P_n(\bar{s})|\bar{w}|!\prod_{i=1}^{N}\frac{a_i^{w_i}}{w_i!} &= \rho\sum_{i=1}^{N}a_i\left(|\bar{w}|-1\right)!\frac{w_i}{a_i}\prod_{q=1}^{N}\frac{a_q^{w_q}}{w_q!}P_n(\bar{s}) \\ &+\frac{1}{k}\sum_{i=1}^{N}\sum_{j=1}^{N}\frac{w_i+1}{|\bar{w}|+1}\left(1+\delta_j\right)\left(s_j+1-e_{ij}\right)\left(|\bar{w}|+1\right)!\frac{a_i}{w_i+1}\prod_{q=1}^{N}\frac{a_q^{w_q}}{w_q!}P_n\left(\bar{s}-\bar{e}_i+\bar{e}_j\right)\end{aligned}.$$

Then, summing all equations for the same $n > k$ and taking into account that $\sum_{i=1}^{N}a_i = 1$, we obtain:

$$\left(1+\rho+\delta(\bar{s})\right)P_n(\bar{s}) = \rho P_n(\bar{s})+\frac{1}{k}\sum_{i=1}^{N}\sum_{j=1}^{N}a_i\left(1+\delta_j\right)\left(s_j+1-e_{ij}\right)P_n\left(\bar{s}-\bar{e}_i+\bar{e}_j\right),$$

Hence, in a shorthand vector-notation:

$$\left((1+\rho)\mathbf{I}+\bar{\boldsymbol{\delta}}\right)\mathbf{P}_n = \rho\mathbf{P}_{n-1}+\mathbf{A}\mathbf{P}_{n+1} \tag{3.6}$$

Here **I** denotes the identity matrix and $\bar{\boldsymbol{\delta}}$ represents a diagonal matrix with elements $\delta(\bar{s})$ on the diagonal, i.e. $\bar{\boldsymbol{\delta}}\xi[\bar{s}] = \delta(\bar{s})\xi[\bar{s}]$, and the matrix **A** is defined by its generation on an arbitrary $d(N,k)$ - vector ξ:

$$\mathbf{A}\xi[\bar{s}] \overset{def}{=} \frac{1}{k}\sum_{i=1}^{N}\sum_{j=1}^{N}a_i\left(1+\delta_j\right)\left(s_j+1-e_{ij}\right)\xi\left[\bar{s}-\bar{e}_i+\bar{e}_j\right]$$

The equation (3.6) is a second order difference equation for $\mathbf{P}_n$ in a $d(N,k)$ dimensional linear space, which plays a central role in our analysis.

Note that $\mathbf{P}_n$ can also be defined for $n \le k$. It is a vector of dimension $d(N,n)$, since only n of the servers are occupied. Let $\mathcal{L}_n$ denote the corresponding linear space of dimension $d(N,n)$ accommodating that vector. The equations (3.2) - (3.4) are equivalent to:

$$\mathbf{D}_n\mathbf{P}_n = \rho\mathbf{F}_n\mathbf{P}_{n-1} + \mathbf{B}_n\mathbf{P}_{n+1} \tag{3.7}$$

where the operators **D**$_n$, **F**$_n$ and **B**$_n$ on the $d(N,n)$- vector ξ are respectively defined as:

$$\mathbf{D}_n : \mathcal{L}_n \to \mathcal{L}_n, \qquad \mathbf{D}_n\xi[\bar{s}] \overset{def}{=} \left(\tfrac{n}{k} + \rho + \delta(\bar{s})\right)\xi[\bar{s}]$$

$$\mathbf{F}_n : \mathcal{L}_{n-1} \to \mathcal{L}_n, \qquad \mathbf{F}_n\xi[\bar{s}] \overset{def}{=} \sum_{i=1}^{N} a_i\xi[\bar{s} - \bar{e}_i]$$

$$\mathbf{B}_n : \mathcal{L}_n \to \mathcal{L}_{n+1}, \qquad \mathbf{B}_n\xi[\bar{s}] \overset{def}{=} \frac{1}{k}\sum_{i=1}^{N}(s_i+1)(1+\delta_i)\xi[\bar{s}+\bar{e}_i], \quad \text{for } n<k$$

For $n = k$, we get **B**$_n$ = **A** operating between $d(N,k)$ dimensional linear spaces. We come back to solving these equations in section 3.5.

3.4 The unperturbed case of an MCMS queueing system with class independent mean service time

In the unperturbed case, it is possible to analyse the system using states *n*, which show the number of clients in the system without type differentiation. The transition diagram is quite simple (Figure 3.3).

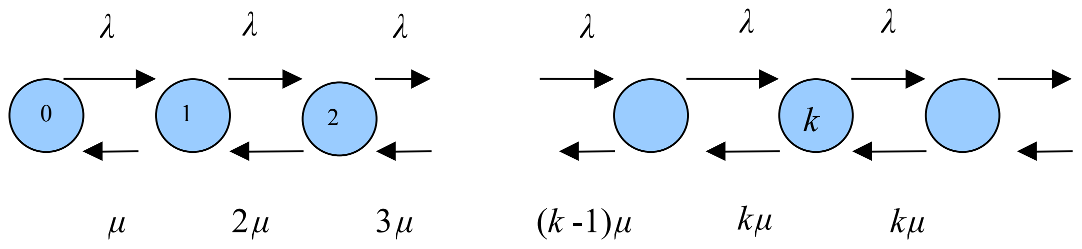

Figure 3.3. Transitions between states in the unperturbed case

The state equations for the probability distribution over *n* are given by:

$$P(n) = \max\left(1, \tfrac{k}{n}\right)\rho P(n-1)$$

and the solution is found in a straightforward way:

for $n \geq k$:

$$P(n) = \rho^{n-k} P(k)$$

for $n < k$:

$$P(n) = \frac{1}{n!}(k\rho)^n P(0)$$

with $P(0)$ such that the probabilities sum up to 1.

Now in case when all service times are equal ($\delta_i = 0$), the multi-class probabilities $P(\overline{w}, \overline{s})$ are just a multinomial modification of the previously found $P(n)$:

for $n \leq k$:

$$P(0, \overline{s}) = P(n)|\overline{s}|! \prod_{i=1}^{N} \frac{a_i^{s_i}}{s_i!}$$

for $n > k$:

$$P(\overline{w}, \overline{s}) = P(n)|\overline{w}|! \prod_{i=1}^{N} \frac{a_i^{w_i}}{w_i!} k! \prod_{i=1}^{N} \frac{a_i^{s_i}}{s_i!}$$

This can be checked by substitution in the state equations, and with some computational effort it can be seen that they are satisfied.

But, a lot more information about other solutions of the state equations can be derived. This is useful for the exploration of the general case in the next section. First, we observe that the eigenvalues and eigenvectors of the matrix **A** can be explicitly found if all $\delta_i = 0$.

In this case ($\delta_i = 0$), the matrix $\overline{\boldsymbol{\delta}}$ in equation (3.6) vanishes and the solutions of this equation are immediately found using the eigenvalues and eigenvectors of the matrix **A.** That is, if we have that $\mathbf{P}_n = z^{-(n-k)}V$ for some z and V equal to an eigenvector of **A** for the eigenvalue v, then by substituting of this $\mathbf{P}_n$ into the equation

$$(1+\rho)\mathbf{P}_n = \rho\mathbf{P}_{n-1} + \mathbf{A}\mathbf{P}_{n+1}$$

we obtain the following equation for *z:*

$$\begin{aligned}
&(1+\rho)z^{-(n-k)}V = \rho z^{-(n-1-k)}V + z^{-(n-k)}\mathbf{A}V \quad \Rightarrow \\
&(1+\rho)zV = \rho z^2 V + vV \quad \Rightarrow \\
&(1+\rho)z = \rho z^2 + v
\end{aligned}$$

The eigenvectors and eigenvalues of the matrix **A** can be constructed as follows. First, we observe that

$$V_{(k,0,\ldots,0)}[\bar{s}] = k!\prod_{i=1}^{N}\frac{(a_i)^{s_i}}{s_i!}$$

defines an eigenvector of A for the eigenvalue 1. Let us interpret it as an eigenvector where all *k* servers are occupied with "real" jobs. In an analogous way, we can find eigenvectors $V_{(m,h_2,\ldots,h_N)}$ for eigenvalues m/k where only *m* servers are occupied with "real" jobs and *k* – *m* are filled "artificially". Index h_j refers to the number of servers occupied with "virtual jobs" in "virtual mode *j*" with *j* = 2, …, *N*.

Mathematically, this means that we decompose each state $\bar{s} = (s_1,\ldots,s_N)$ with $|\bar{s}| = k$ as

$$\begin{aligned}
&s_i = \sum_{j=1}^{N}\theta_i^j, \qquad \theta_i^j \geq 0 \\
&\sum_{i=1}^{N}\theta_i^j = h_j, \qquad h_1 = m \\
&\sum_{j=1}^{N}h_j = k
\end{aligned}$$

This allows for many possible decompositions *θ*. We define

$$V_{\bar{h}}[\bar{s}] = k!\sum_{\theta}\prod_{j=1}^{N}\prod_{i=1}^{N}\frac{\left(t_i^j\right)^{\theta_i^j}}{\theta_i^j!} \tag{3.8}$$

Here we define the real mode $t^1 = (a_1, \ldots, a_N)$ and the virtual mode of type *j* as t^j with $t^j_j = -1$ and $t^j_i = 1/(N-1)$ for $i \neq j$. Note that for *j* = 2, …, *N* the vector t^j is perpendicular to $(1, \ldots, 1)$.

Again, one can check these eigenvectors by substitution using the definition of **A**. The eigenvalues corresponding to these eigenvectors are completely defined by the first component of vector $\bar{h}$, which we denote as $m = h_1$. Then, there are $k + 1$ eigenvalues $v = m/k$ for $m = 0 \ldots k$. The eigenspaces corresponding to each eigenvalue of **A** have dimension $d_{m/k} = \binom{N-2+k-m}{k-m}$ due to the amount of possible combinations of the last $N - 1$ components of the vectors $\bar{h}$, given that the first component is equal to *m*.

Hence, in the unperturbed situation, each of these eigenvalues and a corresponding eigenvector of **A** gives us the solutions of the second order difference equation (3.6) for $n > k$. Then, by substituting the obtained eigenvalues $v = m/k$ for $m = 0 \ldots k$ into the equation $(1+\rho) z_m = \rho z_m{}^2 + v$ we find the values of z_m.

$$z_m = \frac{1}{2\rho}\left\{(1+\rho) \pm \sqrt{(1+\rho)^2 - 4\frac{m}{k}\rho}\right\}, \quad m = 0, \ldots, k$$

Note that the +sign leads to $z_m > 1$ and the -sign to $z_m \leq 1$. Only the solutions with $z_m > 1$ decay for $n \to \infty$ and are acceptable in constructing probability distributions. There are $d(N,k)$ of such solutions. Only one of them, namely with $m = k$, plays a role in the exact solution for the unperturbed case. In this respect, the unperturbed case turns out to be special.

Of course, one should expect that the situation will be more complex in a general case with $\delta_i \neq 0$.

3.5 Solving general MCMS problems

Let us first construct the solutions of the state equations for $n > k$ in general (section 3.5.1). Then we solve the remaining equations for $n \leq k$ and, thereby, construct the full solution of the state equations (section 3.5.2). Next, in section 3.5.3, we consider some special cases. Finally, we give some fast approximations of the exact solution in section 3.5.4.

3.5.1 The exact solution for *n* > *k*

Let us first reformulate the second order difference equation for the $d(N,k)$ vector $\mathbf{P}_n$ given in (3.6) as a first order difference equation for a $2d(N,k)$ dimensional vector. Now let us consider the vector $\mathbb{P}_n = (\mathbf{P}_{n-1}, \mathbf{P}_n)^t$. It has to satisfy $\mathbb{P}_n = \mathbb{H}\mathbb{P}_{n+1}$ with the matrix $\mathbb{H}$ given by

$$\mathbb{H} = \begin{pmatrix} \frac{1}{\rho}\left((1+\rho)\mathbf{I}+\boldsymbol{\delta}\right) & -\frac{1}{\rho}\mathbf{A} \\ \mathbf{I} & \mathbf{0} \end{pmatrix}$$

Note that solving this backward recursion boils down to determining the eigenvalues and eigenvectors of the matrix $\mathbb{H}$. Compared with the previous section, the main difference is that all solutions of (3.6) that decay for $n \to \infty$ will play their part in the solution of the state equations. The next lemma is about finding the decay solutions for $n > k$. The following information is crucial.

Lemma 3.1

1. *Under the stability condition with all δ_i sufficiently small, $\mathbb{H}$ has:*
   - *eigenvalues z satisfying $|z| \le 1$ with total multiplicity $d(N,k)$*
   - *eigenvalues z satisfying $|z| > 1|$ with total multiplicity $d(N,k)$*

2. *If $\delta_i \neq 0$, $\mathbf{A}$ has eigenvalues $\alpha\frac{m}{k}$, with $\alpha = \sum_{i=1}^{N} a_i(1+\delta_i)$, m = 0, ..., k with the same multiplicities and similar eigenspaces as in the unperturbed case. As a consequence, 0 is an eigenvalue of $\mathbb{H}$ with eigenspace $(0, \ker(\mathbf{A}))^t$ and the same multiplicity $d(N-1,k)$ as before.*
3. *N + 1 special eigenvalues of $\mathbb{H}$ corresponding with eigenvectors possessing a product structure can be found:*
   - *there is an eigenvalue 1 with eigenvector $(\mathbf{B},\mathbf{B})^t$ with:*

$$\mathbf{B}(\bar{s}) = |\bar{s}|!\prod_{i=1}^{N}\frac{a_i^{s_i}}{s_i!(1+\delta_i)^{s_i}} \tag{3.9}$$

- *under the stability condition, there are N eigenvalues*

$$z(\eta) = (1+\rho-\eta)/\rho \qquad >1 \tag{3.10}$$

where η satisfies a polynomial equation of degree N + 1:

$$(1+\rho-\eta)/\rho = \sum_{i=1}^{N} a_i \frac{1+\delta_i}{\eta+\delta_i} \tag{3.11}$$

*which also has the solution η = 1, i.e. z(1) = 1 as above, see Figure 3.4. The eigenvector for z(η) is (z(η) **C**,**C**)^t with:*

$$\mathbf{C}(\overline{s}) = |\overline{s}|! \prod_{i=1}^{N} \frac{a_i^{s_i}}{s_i!(1+\delta_i/\eta)^{s_i}} \tag{3.12}$$

One of these eigenvalues z(η_0) crosses z = 1 into z < 1 if ρ crosses 1 into the region ρ > 1 where the stability condition is violated.

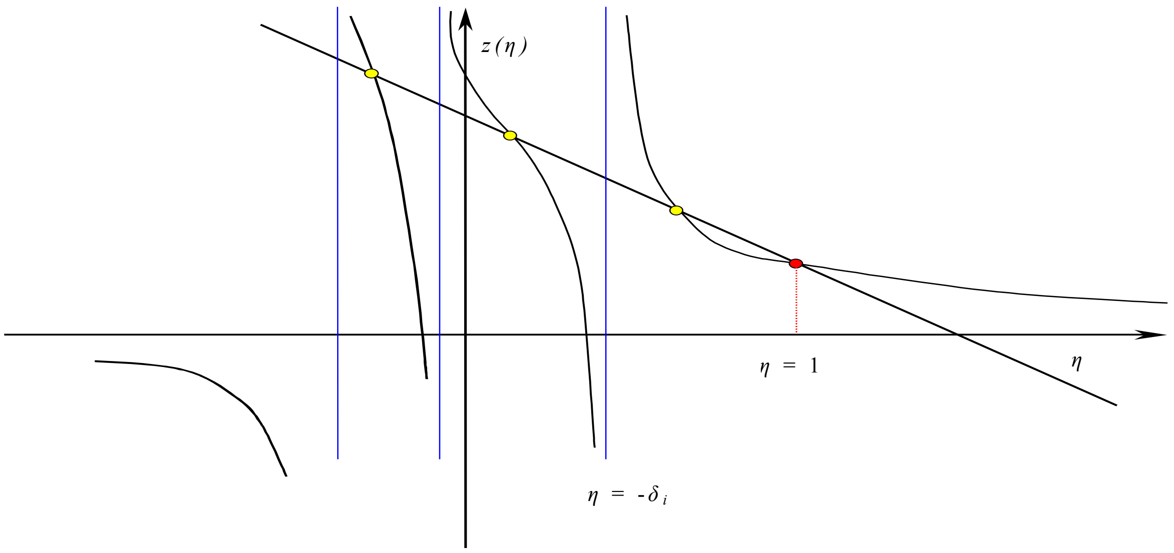

Figure 3.4. Graphical solution of the polynomial equation for some special eigenvalues of ℍ

Proof. To start with we note that $z = 1$ is an eigenvalue of $\mathbb{H}$ corresponding with the given eigenvector. This follows simply by substitution. Note that in the eigenvector the probabilities over the states are proportional to the products of the required service fractions. Now the first part of the lemma is simply a consequence of the well-known perturbation theory for eigenvalues of matrices, cf. Golub (1996). Next, we observe that also in the perturbed case the eigenvalues of **A** are $\alpha \frac{m}{k}$, with $m = 0, \ldots, k$, but now the eigenspaces are slightly different. They are found using a N-vector perpendicular to $\mathbf{1} + \boldsymbol{\delta} = (1 + \delta_1,\ldots,1 + \delta_N)^t$ instead of the expressions in section 3.4. Hence, the dimension of the eigenspaces corresponding with m/k is the same as before.

The remainder of the lemma is a matter of substituting the specified eigenvectors in the eigenvalue equation and checking that it is satisfied. Due to the special structure of the eigenvectors, this is straightforward, therefore the details are left to the reader. The behaviour of the crucial eigenvalue follows immediately from Figure 3.5 by noticing that the derivative of the right hand side at $\eta = 1$ equals $-\rho$. ■

Let us now discuss the consequences of the Lemma 3.1. First, it should be noted that in the case $k = 1$ (a) there are $N - 1$ eigenvalues 0 for any N, (b) there is an eigenvalue 1, and (c) there are N real eigenvalues >1 given by the special polynomial equation. This provides complete information on the eigenvalues. In other cases, other eigenvalues besides the special ones play a role. In Figure 3.5 below, it is shown how the eigenvalues evolve from their unperturbed values if the perturbation is "turned on".

To plot the changes of eigenvalues, we use a case with $N = 4$, $k = 3$, $\rho = 0.9$. To change δ_i, we use function $\delta_i = a_i t$ with $a_2 = 0.1$, $a_3 = 0.15$, $a_4 = 0.2$, when δ_1 is chosen such, that $\sum_{i=1}^{N} \frac{a_i}{(1+\delta_i)} = 1$. In the Table 3.1, we present the perturbed δ_i for $t = 0,\ldots,9$, and next we present how the eigenvalues change with these perturbations.

Table 3.1. Perturbation of delta

t	δ_1	δ_2	δ_3	δ_4
0	0	0	0	0
1	-0.529	0.1	0.15	0.2
2	-0.665	0.2	0.3	0.4
3	-0.727	0.3	0.45	0.6
4	-0.763	0.4	0.6	0.8
5	-0.786	0.5	0.75	1
6	-0.803	0.6	0.9	1.2
7	-0.815	0.7	1.05	1.4
8	-0.825	0.8	1.2	1.6
9	-0.832	0.9	1.35	1.8

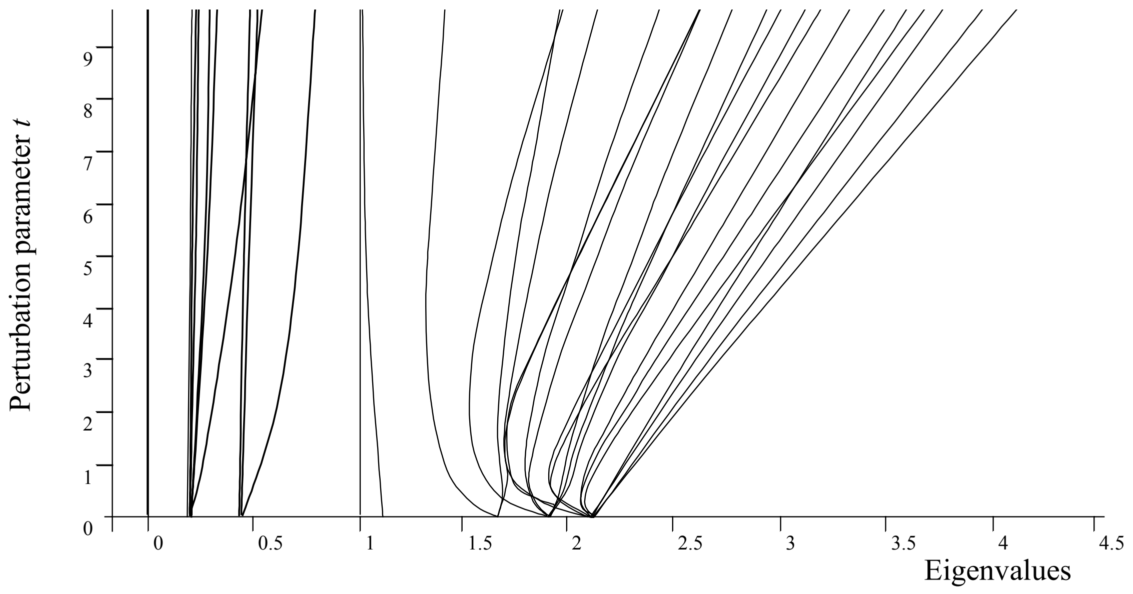

Figure 3.5. A sketch of the behaviour of the eigenvalues of H with the strength of the perturbation

Let us now use the information on the eigenvalues and eigenspaces of $\mathbb{H}$ to solve the state equations for $n \geq k$ exactly in terms of the state probabilities for $n = k$. This can be done using the following recipe. Let us use the standard notation $d = d(N,k)$.

Introduce:

- ***Ω*** *as the d × d diagonal (or Jordan) matrix corresponding with the eigenvalues of* $\mathbb{H}$ *with |z| > 1*
- ***Ξ*** *as the d × d matrix of upper parts of the corresponding (generalised) eigenvectors*

- *Note that* $\mathbf{Z} = \mathbf{\Xi\Omega\Xi}^{-1}$ *satisfies*

$$\left(1+\rho\right)\mathbf{I}+\overline{\boldsymbol{\delta}} = \mathbf{A}\mathbf{Z}^{-1}+\rho\mathbf{Z}$$

- *Now*

$$\mathbf{P}_n = \left(\mathbf{Z}^{-1}\right)^{n-k}\mathbf{P}_k = \mathbf{\Xi\Omega}^{n-k}\mathbf{\Xi}^{-1}\mathbf{P}_k$$

is an exact solution of(3.6) *for* $n \geq k$ *starting at* $\mathbf{P}$ *for* $n = k$.

3.5.2 The exact solution for $n \leq k$

Consider $\mathbf{P}_n\left(\overline{s}\right)$ for $n < k$ with $\mathbf{P}_n$ in $\mathcal{L}_n$ as in (3.7). Again, the state equations for $n < k$ have a backward recursion structure.

The equation for $n = k$:

$$\left(\left(1+\rho\right)\mathbf{I}+\overline{\boldsymbol{\delta}}\right)\mathbf{P}_n = \rho\mathbf{P}_{n-1}+\mathbf{A}\mathbf{P}_{n+1}$$

reduces to:

$$\mathbf{P}_k = \mathbf{Z}^{-1}\mathbf{F}_k\mathbf{P}_{k-1}$$

by using the equation for $\mathbf{Z}$.

For $n < k,$ we obtain:

$$\mathbf{P}_n = \rho\mathbf{D}_n^{-1}\mathbf{F}_n\mathbf{P}_{n-1}+\mathbf{D}_n^{-1}\mathbf{B}_n\mathbf{P}_{n+1}$$

Hence, the complete solution can now be represented as:

$$\mathbf{P}_n = \mathbf{Q}_n\mathbf{P}_{n-1} = \mathbf{Q}_n\mathbf{Q}_{n-1}\cdots\mathbf{Q}_1\mathbf{P}_0$$

where $\mathbf{Q}_n$ follows recursively from

$$\mathbf{Q}_k = \mathbf{Z}^{-1}\mathbf{F}_k$$

and

$$\mathbf{Q}_n = \rho\left(\mathbf{D}_n - \mathbf{B}_n\mathbf{Q}_{n+1}\right)^{-1}\mathbf{F}_n$$

Of course, $\mathbf{P}_0$ is a free constant determined by

$$\sum P\left(\overline{w},\overline{s}\right) = 1$$

A simple computation shows that

$$\mathbf{P}_0 = \left\{1 + \left\langle \mathbf{1}_1, \mathbf{Q}_1 \right\rangle_1 + \cdots + \left\langle \mathbf{1}_{k-1}, \mathbf{Q}_{k-1}\cdots\mathbf{Q}_1 \right\rangle_{k-1} + \left\langle \mathbf{1}_k, \left(\mathbf{I}-\mathbf{Z}^{-1}\right)^{-1}\mathbf{Q}_k\cdots\mathbf{Q}_1 \right\rangle_k \right\}^{-1}$$

Here we denote with $\left\langle \mathbf{1}_n, \mathbf{X} \right\rangle_n$ the innerproduct of $\mathbf{X}$ in a $d\left(N,n\right)$ - dimensional space with the $d\left(N,n\right)$ - vector $\mathbf{1}_n$ with all components equal to 1.

Thus, the exact solution for all n has been derived.

3.5.3 Some special cases

To illustrate the theory, we consider two cases where the algorithm can be executed more explicitly: (I) the case $k = 1$, (II) the case $k = 2$, $N = 2$.

In the case $k = 1$, we have states $s = e_1,\ldots,e_N$ if the server is occupied with s indicating the type of job in execution. The eigenvalues are given by $z(\eta_j)$, $j = 1,\ldots,N$ as defined by equations (3.10) - (3.11); the matrix of eigenvectors Ξ has elements $\frac{a_i}{\left(1+\delta_i/\eta_j\right)}$. The solution of (3.6) is given by:

$$\mathbf{P}_n = \mathbf{P}_0\mathbf{Z}^{-n}\mathbf{a}$$

Here **a** denotes the N-vector with components a_i. As an example we consider the case of 2 classes. Then the eigenvalues are $z_1 = 0$, $z_2 = 1$ and

$$z_{3,4} = \frac{1}{2\rho}\left[\left(\rho + (1+\delta_1) + (1+\delta_2)\right) \pm \sqrt{\rho^2 - 4\rho\left(\sum_{i=1}^{N} a_i (1+\delta_i)\right) + \rho\left(2 + \sum_{i=1}^{N} (1+\delta_i)\right) + (\delta_1 - \delta_2)^2}\right],$$

Also, we obtain the following explicit expressions for $\mathbf{Z}$ and $\mathbf{P}_0$:

$$\mathbf{Z} = \begin{pmatrix} \frac{1}{\rho}\left((1+\delta_1) + a_2\rho\right) & -a_1 \\ -a_2 & \frac{1}{\rho}\left((1+\delta_2) + a_1\rho\right) \end{pmatrix}, \quad \mathbf{P}_0 = 1-\rho$$

In case $k = 2$ and $N = 2$, we start with the observation that the special eigenvalues in equations (3.10) - (3.11) can be explicitly computed, because the degree of the equation is 3 and the solution $\eta = 1$ is already known. This leads us to the following quadratic equation for z:

$$0 = eq_1(z) \overset{def}{=} \rho^2 z^2 - \rho\{2 + \rho + \delta_1 + \delta_2\} z + \{1 + \rho + \delta_1 + \delta_2 + (1-\rho)\delta_1\delta_2\}$$

Now, only 2 of the 6 eigenvalues of $\mathbb{H}$ are still unknown; they can be found from the characteristic polynomial $0 = \mathbf{eq}(z) \overset{def}{=} \det\left(-(1+\rho+\mathbf{D})z + \rho z^2\mathbf{I} + \mathbf{A}\right)$ by dividing out the known eigenvalues. Here we denote by $\mathbf{D}$ the diagonal matrix with the average perturbation δ for the respective states on the diagonal. Hence, we decompose $\mathbf{eq}(z) = z(z-1)eq_1(z)eq_2(z)$. Therefore, the remaining 2 eigenvalues have to satisfy a quadratic equation with coefficients that can be explicitly given in terms of the original matrix coefficients as:

$$0 = eq_2(z) \overset{def}{=} \rho z^2 - \rho\left\{1 + \rho + \frac{1}{2}(\delta_1 + \delta_2)\right\} z + \frac{1}{2}\{1 + a_1\delta_1 + a_2\delta_2\}$$

Note that the latter equation gives rise to one eigenvalue $z > 1$.

Now $\mathbf{Z}$ is a 3×3 matrix operating on the probabilities of the states (2, 0), (1, 1), (0, 2) for the jobs in execution when the servers are occupied. The column of Ξ corresponding with an eigenvalue $z = (1+\rho-\eta)/\rho$ becomes $(v, 1, 1/v)^t$ with $v = a_1(\eta+\delta_2)/\{a_2(\eta+\delta_1)\}$. For the other eigenvector corresponding with $z > 1$ satisfying $eq_2(z) = 0$, we find

$\left(-a_1\left(1+\delta_2\right)/q(z), 1, a_2\left(1+\delta_1\right)/q(z)\right)^t$ with $q(z)=\left\{\left(\delta_2-\delta_1\right)/2\right\}z+\left\{a_1\left(1+\delta_1\right)-a_2\left(1+\delta_2\right)\right\}/2$. Note that one of the components of this eigenvector is negative. Apparently, this can happen even though the probabilities will eventually be positive. It is clear that also in this case the solution can be given explicitly in terms of the parameters of the problem.

3.5.4 Approximate solution

To conclude this section, we shall introduce some approximations of the state probabilities $P_n\left(\bar{s}\right)$. These approximations will be constructed using partial solutions of the state equation for $n > k$ and $n < k$, which have a product structure. However, the equation for $n = k$ can only be satisfied if $k = 1$. In the general case with $k > 1$, the equation for $n = k$ fails to be satisfied. The advantage of the approximations is that (I) the computation of the approximation requires algebra in spaces of considerably lower dimension and (II) all sums over states in the performance measures can be determined explicitly due to the product structure. The approximation is given by:

for $n < k$:

$$P_n\left(\bar{s}\right)=C\left(k\rho\right)^n\prod_{i=1}^{N}\frac{a_i^{s_i}}{s_i!\left(1+\delta_i\right)^{s_i}} \tag{3.13}$$

for $n \geq k$:

$$P_n\left(\bar{s}\right)=\sum_{i=1}^{N}\gamma_i z_i^{-(n-k)}D\left(\eta_i\right)^{-k}\left|\bar{s}\right|!\prod_{j=1}^{N}\frac{a_j^{s_j}}{s_j!\left(1+\delta_j/\eta_i\right)^{s_j}} \tag{3.14}$$

with η_i, z_i as defined in (3.10) - (3.11) and

$$D\left(\eta_i\right)=\sum_{j=1}^{N}\frac{a_j}{\left(1+\delta_j/\eta_i\right)}$$

In order to determine the constants C and γ_i we require that the total probability sums to 1 and that the expected number of items of type i in service is correct for each i. This provides us $N + 1$ equations for $N + 1$ unknowns:

$$\sum_{i=1}^{N}\left\{D(\eta_i)^{-1}\frac{a_m}{\left(1+\delta_m/\eta_i\right)}\right\}\varphi_i + Cb_m\sum_{n=1}^{k-1}\frac{n}{k}\frac{(k\rho)^n}{n!} = \rho_m$$

$$\sum_{i=1}^{N}\varphi_i + C\sum_{n=0}^{k-1}\frac{(k\rho)^n}{n!} = 1$$

with $\varphi_i \overset{def}{=} \gamma_i\left(1-z_i^{-1}\right)^{-1}$ and hence:

$$C = \left\{\sum_{n=0}^{k-1}\frac{(k\rho)^n}{n!}\left(1-\frac{n}{k}\right)\right\}^{-1}(1-\rho)$$

An even simpler approximation is found, if we use for $n > k-1$ only the critical mode corresponding with the least decay for increasing *n*. In this approximation, we use a simple 2 class approximation to determine $\eta_{cr} < 1$ from a quadratic equation. The first class is the critical class, all the other classes are joined into a non-critical class with $a_{ncr} = 1 - a_{cr}$, and δ_{ncr} follows from the requirement $a_{cr}/(1+\delta_{cr}) + a_{ncr}/(1+\delta_{ncr}) = 1$. Further, the reasoning to get a quadratic equation exploits that η satisfies the special two-class equation (3.10) - (3.11) of degree 3. This is similar to the example with $N = 2$ at the end of the previous subsection. In this case, we require the expected number in service disregarding type to be correct and we obtain the same value for *C* as given here above, but now:

$$\gamma_{cr} = \left(1 - z_{cr}^{-1}\right)\left\{1 - C\sum_{n=1}^{k-1}\frac{(k\rho)^n}{n!}\right\}$$

and the other coefficients vanish.

3.6 Accurate information on performance measures

Here we shall provide some further insight in the behaviour of some interesting performance measures related to stochastic variables, such as the waiting time *w*, the queue length *q* and the number of items of type *i* in queue q_i. Performance measures that we consider are expectations, variances and probability for positivity of these stochastic variables. Moreover, since it is clear that there is correlation between these stochastic variables, we also shall provide formulas for some conditional expectations and correlation coefficients. Of course, the clue that such exact formulas can easily be derived lies in the fact that series of geometric type can simply be summed. To start, we notice that:

$$P(q>0)=\left\langle \mathbf{1}_k,(\mathbf{Z}-\mathbf{I})^{-1}\mathbf{P}_k\right\rangle$$
$$P(w>0)=\left\langle \mathbf{1}_k,(\mathbf{Z}-\mathbf{I})^{-1}\mathbf{ZP}_k\right\rangle$$
$$P(q_i>0)=\left\langle \mathbf{I}_k^{\,i},(\mathbf{Z}-\mathbf{I})^{-1}\mathbf{P}_k\right\rangle$$

and the performance estimators for the total number of items in queue are:

$$E[q]=\left\langle \mathbf{1}_k,(\mathbf{Z}-\mathbf{I})^{-2}\mathbf{ZP}_k\right\rangle$$
$$E\left[q^2\right]=E\left[q(q-1)\right]+E[q]=2\left\langle \mathbf{1}_k,(\mathbf{Z}-\mathbf{I})^{-3}\mathbf{ZP}_k\right\rangle$$
$$Var[q]=E\left[q^2\right]-E[q]^2$$

Then we can write down the performance estimators for the number of items in queue for each type of items:

$$E\left[q_i\mid q=n-k\right]=a_i(n-k)$$
$$E[q_i]=a_iE[q]$$
$$E\left[q_i(q_i-1)\mid q=n-k\right]=a_i^2(n-k)(n-k-1)$$
$$E\left[q_i(q_i-1)\right]=a_i^2E\left[q(q-1)\right]$$
$$Var[q_i]=E\left[q_i(q_i-1)\right]+E[q_i]-E[q_i]^2$$

We can also estimate the covariance between the numbers of items of different classes in queue:

$$E\left[q_iq_j\mid q=n-k\right]=a_ia_j(n-k)(n-k-1)\quad \text{for } i\neq j$$
$$E\left[q_iq_j\right]=a_ia_jE\left[q(q-1)\right]\quad \text{for } i\neq j$$
$$corr\left(q_i,q_j\right)=E\left[q_iq_j\right]-E[q_i]E\left[q_j\right]$$

Here we use the notation $\mathbf{1}_n$ as before in determining $\mathbf{P}_0$. The matrix $\mathbf{I}_n^{\,i}$ is a diagonal matrix of dimension $d(N,n)$ with 1 for each state $\overline{s}$ that contains type *i* and 0 elsewhere. Note that Little's law is satisfied and the expected waiting time does not depend on the type of item.

In an analogous way, we can analyse the stochastic variables R = the number of items in the system, R_i = the number of items of type i in the system and BO_i = the number of backorders for type i, $BO_i = \max(0, R_i - st_i)$ with st_i a given positive integer representing the stock level in spare part networks. Now we obtain:

$$P(R_i = m \mid R < k) = \sum_{n=m}^{k-1} P(R_i = m \mid R = n) = \sum_{n=m}^{k-1} \left\langle \mathbf{I}_n^{\,i,m}, \mathbf{P}_n \right\rangle$$

$$P(R_i = m \mid R \geq k) = \sum_{r=\max(m-k,0)}^{m} \sum_{n=k+r}^{\infty} P(q_i = r \mid R = n) P(R_i - q_i = m - r \mid R = n)$$

$$= \sum_{r=\max(m-k,0)}^{m} \sum_{n=k+r}^{\infty} \binom{n-k}{r} a_i^{\,r} (1-a_i)^{n-k-r} \left\langle \mathbf{I}_k^{\,i,m-r}, \mathbf{Z}^{-(n-k)} \mathbf{P}_k \right\rangle$$

$$= \sum_{r=\max(m-k,0)}^{m} a_i^{\,r} \sum_{l=r}^{\infty} \binom{l}{r} \left\langle \mathbf{I}_k^{\,i,m-r}, \left((1-a_i)\mathbf{Z}^{-1}\right)^{(l-r)} \mathbf{Z}^{-r} \mathbf{P}_k \right\rangle$$

$$= \sum_{r=\max(m-k,0)}^{m} a_i^{\,r} \sum_{l=r}^{\infty} \binom{l}{r} \left\langle \mathbf{I}_k^{\,i,m-r}, \left(\mathbf{I} - (1-a_i)\mathbf{Z}^{-1}\right)^{-(r+1)} \mathbf{Z}^{-r} \mathbf{P}_k \right\rangle$$

$$P(R_i = m) = P(R_i = m \mid R < k) + P(R_i = m \mid R \geq k)$$

$$P(R_i = m \wedge R_j = l) = \sum_{n=m+l}^{k-1} \left\langle \mathbf{I}_n^{\,i,m} \mathbf{I}_n^{\,i,l}, \mathbf{P}_n \right\rangle$$
$$+ \sum_{r,t \in V(m,l)} \binom{r+t}{t} a_i^{\,r} a_j^{\,t} \left\langle \mathbf{I}_n^{\,i,m-r} \mathbf{I}_n^{\,i,l-t}, \left(\mathbf{I} - (1-a_i-a_j)\mathbf{Z}^{-1}\right)^{-r-t-1} \mathbf{Z}^{-r-t} \mathbf{P}_k \right\rangle$$

Here we use the notation $\mathbf{I}_n^{\,i,m}$ for the diagonal matrix of dimension $d(N,n)$ indicating the states with *m* items of type *i* with 1 and 0 elsewhere. The matrix **I** is the usual identity matrix. In this derivation, we use *r* to count the number of type *i* items in queue. In the joint distribution, *t* counts the number of type *j* items in queue. In that case, the summation runs over a domain *V*(*m*, *l*), where *r* runs from max(0, *m* - *k*) to m and *t* runs from max(0, l - *k*) to l, while (*m* - *r*) + (*l* - *t*) ≤ *k*. The precise information on the distribution of the numbers of type *i* and *j* in the system will play a role in the backorder computations.

Some other quantities are simpler to derive. Using Little's theorem, we find the first moments:

$$E[R_i] = E[q_i] + \lambda_i / \mu_i$$
$$E[S_i] = \lambda_i / \mu_i$$

Moreover, we can now easily derive the following second moments and correlation coefficients:

$$E\left[R_i^2\right] = E\left[q_i^2\right] + E\left[S_i^2\right] + 2E[S_i q_i]$$
$$E\left[S_i^2\right] = \sum_{n=1}^{k-1} \left\langle \chi_n^{s_i^2}, \mathbf{P}_n \right\rangle + \left\langle \chi_k^{s_i^2}, \left(\mathbf{I} - \mathbf{Z}^{-1}\right)^{-1} \mathbf{P}_k \right\rangle$$
$$E\left[S_i q_j\right] = a_j \left\langle \chi_k^{s_i}, \left(\mathbf{I} - \mathbf{Z}^{-1}\right)^{-2} \mathbf{Z}^{-1} \mathbf{P}_k \right\rangle$$
$$E\left[S_i S_j\right] = \sum_{n=1}^{k-1} \left\langle \chi_n^{s_i \cdot s_j}, \mathbf{P}_n \right\rangle + \left\langle \chi_k^{s_i \cdot s_j}, \left(\mathbf{I} - \mathbf{Z}^{-1}\right)^{-1} \mathbf{P}_k \right\rangle$$
$$E\left[R_i R_j\right] = E\left[S_i S_j\right] + E\left[S_i q_j\right] + E\left[S_j q_i\right] + E\left[q_i q_j\right]$$

In these formulas, $\chi_n^{s_i^2}$ and $\chi_n^{s_i \cdot s_j}$ are the diagonal matrices of dimension $d(N,n)$ with the square number of items of type *i* in $\bar{s}$ (i.e. s_i^2) or, accordingly, the product of the numbers of types *i* and *j* items in $\bar{s}$ (i.e. $s_i \cdot s_j$) as the diagonal element corresponding to $\bar{s}$.

The first and second moments and the correlation coefficients of the backorder variables with stock levels $st_i = 0$ are also known, since the backorders and the numbers of items in the system coincide then. Next, this sort of information for backorders with safety levels $st_i > 0$ can be derived using recursion with respect to st_i. This basic idea can already be found in Sherbrooke (1992). Here we extend it to include correlations between backorders of different types. The recursion relations are as follows:

$$\begin{aligned} & E\left[BO_i(st_i+1)\right] = E\left[BO_i(st_i)\right] - P(R_i \geq st_i + 1) \\ & \quad \text{with } P(R_i \geq st_i + 1) = P(R_i \geq st_i) - P(R_i = st_i) \\ & \quad \text{and } E\left[BO_i(0)\right] = E[R_i], \quad P(R_i \geq 0) = 1 \end{aligned} \tag{3.15}$$

$$E\left[BO_i(st_i+1)^2\right] = E\left[BO_i(st_i)^2\right] - E\left[BO_i(st_i)\right] - E\left[BO_i(st_i+1)\right]$$
$$\text{and } E\left[BO_i(0)^2\right] = E\left[R_i^2\right]$$

$$E\left[BO_i\left(st_i+1\right)BO_j\left(st_j+1\right)\right]=E\left[BO_i\left(st_i\right)BO_j\left(st_j\right)\right]-E\left[BO_j\left(st_j\right)H\left(R_i-\left(st_i+1\right)\right)\right]$$

$$\text{with}\quad E\left[BO_j\left(st_j\right)H\left(R_i-\left(st_i+1\right)\right)\right]=E\left[BO_j\left(st_j\right)H\left(R_i-st_i\right)\right]-E\left[BO_j\left(st_j\right)\Delta_0\left(R_i-st_i\right)\right]$$

$$E\left[BO_j\left(st_j+1\right)\Delta_0\left(R_i-st_i\right)\right]=E\left[BO_j\left(st_j\right)\Delta_0\left(R_i-st_i\right)\right]-P\left(R_j\geq st_j+1,R_i=st_i\right)$$

$$P\left(R_j\geq st_j+1,R_i=st_i\right)=P\left(R_j\geq st_j,R_i=st_i\right)-P\left(R_j=st_j,R_i=st_i\right)$$

$$\text{and}\quad E\left[BO_i\left(0\right)BO_j\left(0\right)\right]=E\left[R_iR_j\right],\quad P\left(R_i\geq 0,R_i=st_i\right)=P\left(R_i=st_i\right)$$

Here we used the notation *H* for the Heaviside function *H*(*x*) = 1 for $x > 0$ and = 0 else. Further, Δ_0 denotes the Dirac function with $\Delta_0\left(x\right)$ = 1 if $x = 0$ and $\Delta_0\left(x\right)$ =0 else. If necessary, the higher order moments and the correlations between backorders of various item types can be derived in an analogous way.

3.7 Some numerical results

In order to derive numerical results, we implemented the exact solutions and the approximations as described above in MATLAB. A wealth of interesting phenomena can now be quantitatively analysed. Let us just show a few of the results.

3.7.1 The average waiting time in MCMS systems

The effects of different service times for various classes of items can easily be illustrated. Consider a system with 2 classes and *k* servers. The parameters are chosen as $a_1 = 1/3$, $a_2 = 2/3$. The first type of items has an average service rate smaller than the second type of items. In terms of δ_1 and δ_2 this can be represented as:

$$\delta_1=-\delta,\quad \delta_2=\tfrac{1}{2}\delta\big/\left(1-\tfrac{3}{2}\delta\right),\quad \text{with}\quad \delta\in\left[0,\tfrac{2}{3}\right)\quad \text{i.e.}\quad \sum_{i=1}^{2}\tfrac{\alpha_i}{1+\delta_i}=1$$

Let us now compare for various choices of ρ and *k* the expected waiting time in the system with $\delta > 0$ with those for *M/M/k* where $\delta = 0$ (Figure 3.6). The conclusion that the ratio increases with δ is not surprising, but it is nice that we can explicitly determine how it increases.

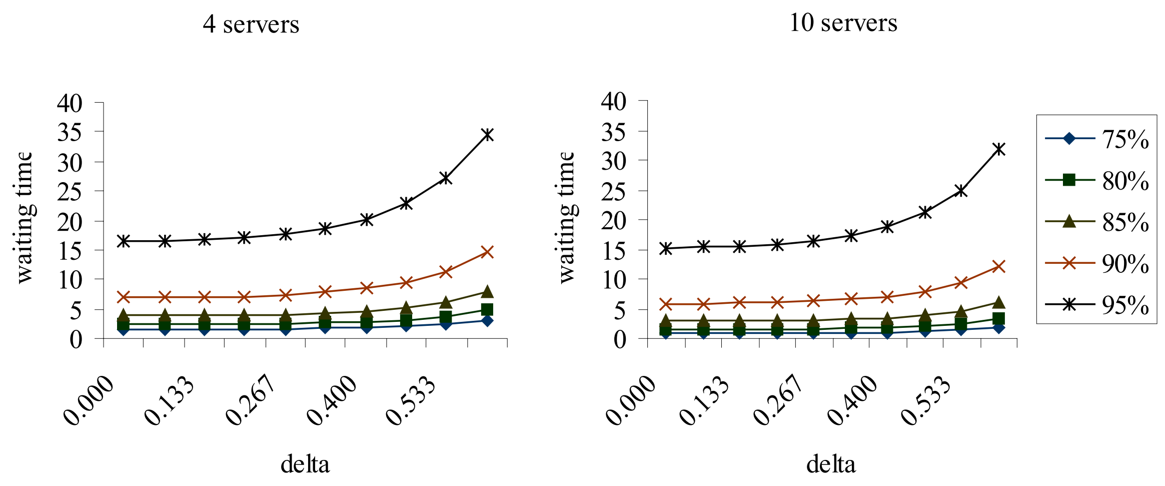

Figure 3.6 Increasing differences between service characteristics of classes lead to increasing waiting times

3.7.2 Variance per item and correlation between different types of items.

Of course, a similar behaviour as in Figure 3.6 is found for the average number of items in the system. It is also interesting to notice that the variance-to-mean ratio of the number of items in the system can easily be computed. It behaves as shown on Figure 3.7. As a comparison one should keep in mind that the variance-to-mean ratio equals 1 in the classic VARI-METRIC model, because the repair shops are modelled as $M/G/\infty$ queues.

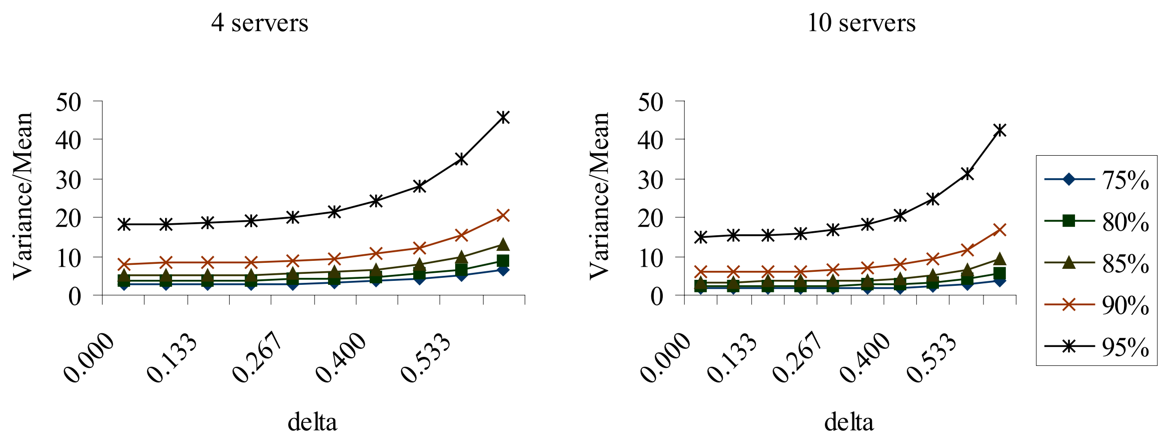

Figure 3.7 Increasing differences between service characteristics of classes lead to increasing variance to mean ratio

Let us again consider the same example, but now with our focus on the interdependency between stochastic variables for different types of items. In Figure 3.8, the behaviour of the correlation coefficient is shown.

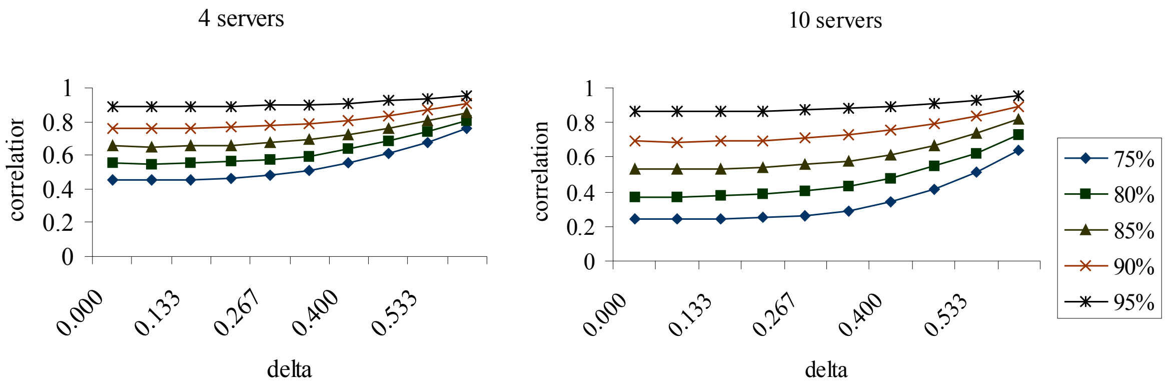

Figure 3.8 Increasing differences between service characteristics of classes lead to increase of the correlation coefficient

From the figure above, we see that the correlation between backorder levels can be significant. Even at a moderate utilisation (75%), the correlation coefficient may exceed 0.5. For high utilisation (ρ = 0.95), it is clear that the correlation is very high (around 0.9), suggesting that it is relevant to take the correlations into account when estimating the system availability. In the next subsection, we deal with this issue.

3.7.3 Comparing approximations for the system availability

Let us first have a closer look at the asymptotic expansion of the nonlinear system availability function (2.1). The VARI-METRIC models use normally a linear approximation (2.5) of this function (cf. Chapter 2):

$$A_1 = 1 - \frac{1}{B} \sum_{m \in ECH(N^{ech})} \sum_{j \in IND(1)} E\left[BO_{mj}\left(\overline{st}\right)\right]$$

where B is the size of the installed base at location m. Although this approximation seems adequate for infinite repair shop capacities, it is questionable whether this is also true in the case of *finite* capacities. The latter means that the number of backorders of different components at the same repair shop are mutually *correlated*. This is a severe complication, as the expectation of the product of backorders cannot be taken term-wise anymore. Still, equation (2.5) can be seen as a linear approximation of (2.1), and then the correlation does not play a role anymore. The quality of this approximation is still unknown, however. The analysis of MCMS models provides insight in the *joint* probability distribution of the number in repair for all items. This gives us the expectations for backorders per item as well as the correlations between items. Therefore, it is possible to have an approximation where also the quadratic terms are taken into account:

$$A_2 = 1 - \sum_{m \in ECH(N^{ech})} \sum_{i \in IND(1)} E\left[BO_{mi}\left(\overline{st}\right)\right] \Big/ B_m Z_i$$

$$+ \sum_{m \in ECH(N^{ech})} \sum_{\substack{i,j \in IND(1) \\ i<j}} E\left[BO_i\left(st_i\right) BO_j\left(st_j\right)\right] \Big/ \left(B_m Z_i \cdot B_m Z_i\right)$$

Note that the correction to the linear approximation is positive. Here we use exact expressions for the moments of the backorders. In the next figure, we show the difference between the two approximations. Let us take an example of a single-location, single indenture system with k = 4 servers and N = 5 types of items processed in the same repair shop. The arrival (failure) rates fractions are a_i = 0.2 for all i. For the service rate deviation δ_i, we put $0.5(i-1)$ for i = 2,…,5 and δ_1 is negative = -21/31 so that $\sum a_i/(1+\delta_i)=1$. All spare part levels are equal to st_i = 7 items in stock. The size of the installed base B is 15. The utilisation rate is varied between ρ = 0.6 and ρ = 0.84.

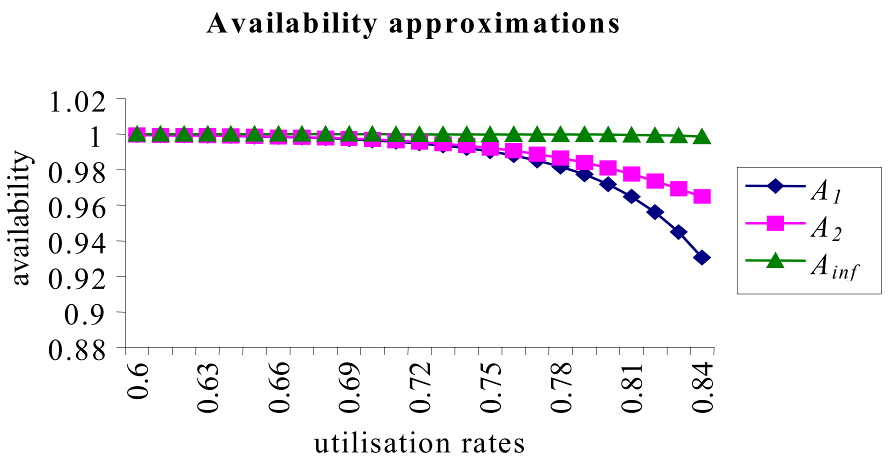

Figure 3.9 Approximations of the system availability

In the Figure 3.9, we have also shown another approximation for the system availability, denoted by A_{inf}. This estimate is obtained under the assumption of infinite repair capacity and based on a general service time distribution (waiting and repair together!) for items of type i, having a mean value R_i as derived in section 3.6. Using Palm's theorem, cf. Palm (1938), then the number of items of a type i in the repair shop has a Poison distribution with R_i as parameter. This infinite capacity approximation is often used in practice, cf. Rustenburg (2000). Note from the figure above that considerable differences between the approximations of the system availability arise for high utilisation rates. Moreover, the approximation A_2, which is closer to reality, lies between the approximation A_1 and A_{inf}. This means that the approximation errors of the new VARI-METRIC method are caused by ignoring backorders correlations.

3.7.4 Computational efforts

To estimate the computational efforts needed to solve the problem exactly, we estimated the CPU time usage for the most demanding computations, namely the calculation of the eigenvectors of the matrix H. Below, we show both the dimension d of the vector **P** and the matrix $\mathbb{H}$, and the CPU-time required to compute the eigenvectors. The values are given for various values for the number of item classes N and the number of servers k. The computing time was estimated in seconds using MATLAB 5.3 with NAG toolbox, using a Pentium II-350 PC with 128Mb RAM and under Windows NT.

The matrix $\mathbb{H}$ is first balanced and then reduced to upper Hessenberg form using real stabilised elementary similarity transformations. The eigenvalues and eigenvectors of the Hessenberg matrix are calculated using the QR algorithm. Next, the eigenvectors of the Hessenberg matrix are transformed back to the eigenvectors of the original matrix $\mathbb{H}$ (cf. Golub and Loan, 1996, Saad, 1992). The total computation time of both these algorithms is polynomial in dimension of the matrix $\mathbb{H}$ (Figure 3.10, Table 3.2 and Table 3.3).

The highest dimension of the cases solved here (d =715, for 5 items and 9 servers) seems already quite high for a practical use, and shows that this exact algorithm can be easily used in practice, certainly if a faster computer is used. Otherwise, the approximation scheme (section 3.5.4) can be applied.

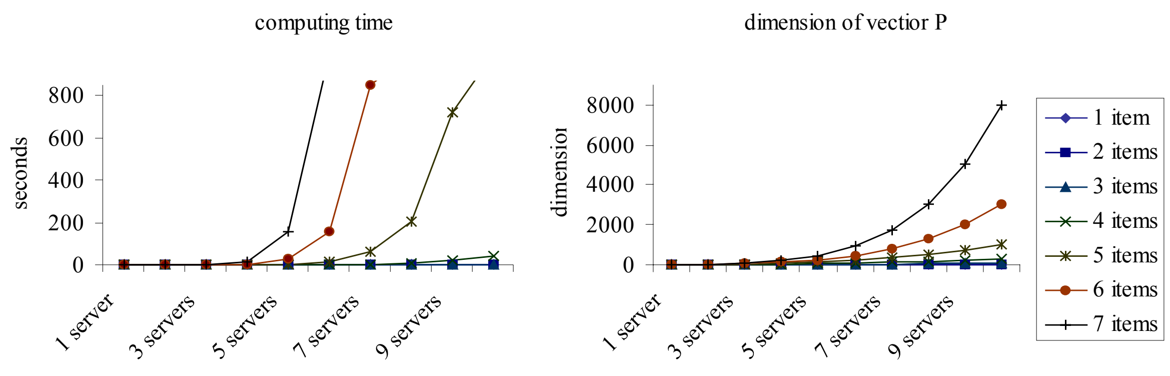

Figure 3.10 Computing times and dimensions of the vector P for different number of item classes in the system and different amounts of servers

Table 3.2 Computing time (sec.) required to find eigenvalues for different number of item classes in the system and different amounts of servers

	1 ser.	2 ser.	3 ser.	4 ser.	5 ser.	6 ser.	7 ser.	8 ser	9 ser.	10 ser.
2 classes	0	0	0	0	0	0	0	0	0	0.01
3 classes	0	0	0	0.01	0.02	0.05	0.09	0.16	0.271	0.41
4 classes	0	0	0.01	0.07	0.26	0.771	2.583	8.382	19.618	43.853
5 classes	0	0.01	0.07	0.43	2.484	15.752	60.387	201.38	720.55	-
6 classes	0	0.07	0.24	2.474	26.448	155.30	-	-	-	-

Table 3.3 Dimensions of the vector P for different number of item classes in the system and different amounts of servers

	1 ser.	2 ser.	3 ser.	4 ser.	5 ser.	6 ser.	7 ser.	8 ser	9 ser.	10 ser.
2 classes	2	3	4	5	6	7	8	9	10	11
3 classes	3	6	10	15	21	28	36	45	55	66
4 classes	4	10	20	35	56	84	120	165	220	286
5 classes	5	15	35	70	126	210	330	495	715	1001
6 classes	6	21	56	126	252	462	792	1287	2002	3003

3.7.5 On the quality of the approximate probability distribution.

Finally, we shall provide some insight in the errors by using the simpler N mode or 1-critical mode approximations introduced in section 3.5.4. Let us take an example with $N = 5$ classes of items with arrival fractions $a_i = 0.2$ for all i. For the service rate deviation δ_i, we put

$0.5(i-1)$ for $i = 2,\ldots,5$, $\delta_1 = -21/31$, so that $\sum a_i/(1+\delta_i) = 1$, where $a_i/(1+\delta_i)$ represent the service fractions. The utilisation rate is varied between $\rho = 0.7$ and $\rho = 0.95$. The number of servers is varied between $k = 1$ and $k = 4$. In the next figure (Figure 3.11), the relative errors for the first and second moments are compared to the exact solution.

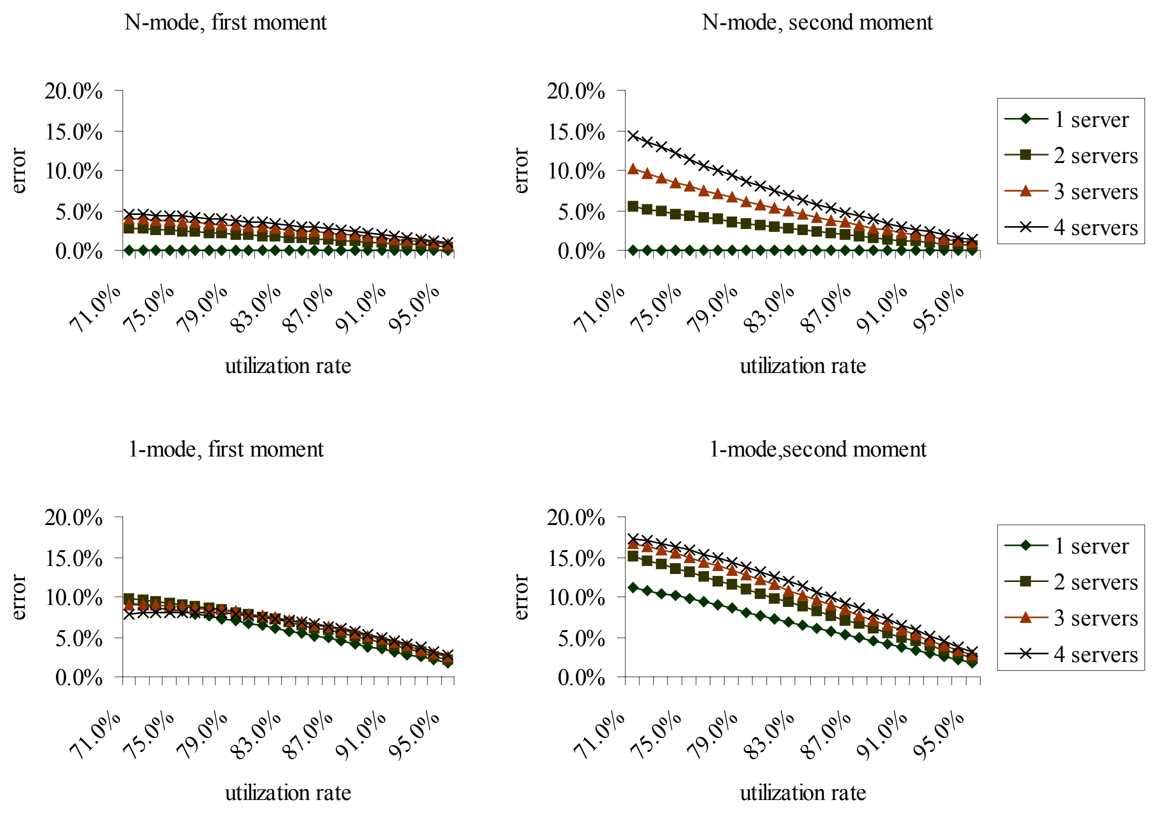

Figure 3.11 The approximate distributions are usually rather accurate for high utilisation rates

In these examples, with moderate ratios of service fractions for different classes and high utilisation rates, the errors are around 5%, which is quite reasonable. Of course, one should be careful with these approximations if the number of servers is large.

The approximations are better for high utilisation rates, since the smallest (critical) eigenvalue (>1) becomes closer to one and the influence of the others eigenvalues becomes negligible. This is known as state space collapse in heavy traffic approximations. For further discussions of this phenomenon, we refer to R.J. Williams (1998) and M. Bramson (1998).

3.8 Conclusions and generalisations

In this chapter, we derived a method for the exact analysis of multi-class, multi-server queues, based on a classical method using the stationary state equations. Thereby, we have answered research question 2 as stated in Chapter 1. Though conceptually we only deal with a perturbation of the well-known single class *M/M/k* system, the structure of the solution becomes a lot more complex. Using the exact solution, several performance measures of the MCMS system can be studied in terms of formulas with a finite number of terms. The computational effort to find the exact solution depends on the number of classes *N* and the number of servers *k*. Representative for the computational effort is the dimension of the linear space used in section 3.5, i.e. $d(N,k)=(N+k-1)!/\{k!(N-1)!\}$. For large instances of the problem, some well-founded approximation can be given which only relies on matrix inversion in a *N*-dimensional space, or even simpler, using only a 2 class reduction, have a 2-dimensional character.

The exact method introduced in this chapter can in principle be generalised to problems with non-identical servers, but the computational effort will increase since we will have to keep track of states of each server and then we get $d(N,k)=N^k$. Also, certain situations with priority classes can be tackled in the spirit of this chapter (cf. Chapter 6).

Chapter 4
Effects of finite repair capacity in multi-echelon, multi-indenture service part supply systems

Abstract

In this chapter, we consider multi-echelon, multi-indenture supply systems for repairable service parts with finite repair capacity. We show that the commonly used assumption of infinite capacity may seriously affect system performance and stock allocation decisions if the repair shop utilisation is relatively high. Both for the case of item-dedicated and shared repair shops, we modify the well-known VARI-METRIC method to allocate service part stocks in the network. The repair shops are modelled by (single or multi-class) multi-server queueing systems. We validate our procedure by comparison to results from discrete event simulation. This comparison shows that the accuracy of the technique presented in this chapter is on average more than five times as close to simulated values as the classical VARI-METRIC technique. This is also the answer to our third research question as stated in Chapter 1.

This chapter is based on the paper:

“Effects of finite repair capacity in multi-echelon, multi-indenture service part supply systems.” A. Sleptchenko, M.C. van der Heijden, A. van Harten, *International Journal of Production Economics,* 73(3), 209-230

4.1 Introduction

In this chapter, we extend the VARI-METRIC model to a model with finite repair capacities. We model the repair shops using multi-class, multi-server queueing systems as presented in the previous chapter. Our aim is to get insight in the impact of capacity restrictions on the system performance in terms of the expected number of backorders, and system availability. Also, we will examine how the service part allocation in the network is affected by the finite capacity. In other words, we will examine to which extent the use of the infinite capacity assumption will lead to sub-optimal decisions (lower availability for the same budget).

Similar to Chapter 2, we define a repair shop as a part of a location that has its own repair facilities. We consider two variants of repair facilities:

- an item-dedicated repair facility that is able to repair only one kind of items. Then every location has as many repair facilities as the number of items.
- a cluster-dedicated repair facility, where multiple kinds of items can share the resources. Here we define a *cluster* as a set of items, which can be repaired at the same facility using the same shared repair resources (personnel, equipment). There is no restriction in the composition of the clusters.

In the next section, we will first give a preliminary analysis of the impact of finite capacity. We will show that simply “plugging in” average throughput times in the VARI-METRIC model may lead to inaccurate results. Next, we will develop a method to optimise spare part stock levels under finite repair shop capacity, which is an extension of Slay’s VARI-METRIC method (section 4.3). In section 4.4, we compare the system availability obtained from a numerical simulation for different input sets to the system availability estimated by presented approximation. Finally, we present our conclusions in the last section.

4.2 A preliminary analysis of the impact of finite capacity

As mentioned in the introduction, a simple approach to deal with finite repair capacity is to measure throughput times and plug these values as repair times in the (VARI-)METRIC model. However, this may lead to inaccurate intermediate results, as we will show in this section. The issue is that the aforementioned approach does not guarantee that the *variance* of the number of items under repair is correctly estimated.

To this end, we compare two models. In the first model, we have a finite capacity repair shop where the part throughput time consists of waiting time and repair time. A single repair shop handles all items. Assuming for this simple example that the repair times are the same for all items, we can model the repair shop as an *M/M/k* queue. In the second model, we assume an infinite capacity repair shop, modelled as an *M/G/∞* queue, where the service time equals the sum of the waiting time and repair time resulting from the finite capacity system as found from the first model. Then, the number of parts under repair in the second model is Poisson distributed, so that the variance of the number of parts in the repair shop equals the mean. Obviously, the latter need not to be true in the first model. Hence the error made by using a simple *M/G/∞* queueing system is caused by misspecification of the *variance* of the number of items in the queueing system.

To compare these models for repair shops, we analyse simple models for both single-echelon and two-echelon models. The two-echelon model consists of one central depot and one local base. We assume that *all* repair shops have finite capacity and that the utilisation rates and number of servers are equal for all repair shops. For these models, we first find the mean number of items in service based on utilisation rate and number of servers. Then, we optimise the stock levels according to the VARI-METRIC method, where the variance of number of items in repair is equal to the mean of this number for different availability levels, from 0% to 100%. Thereby, we estimate the system availability using the infinite capacity assumption. Also, we estimate the system availability based on a *finite* capacity repair shop for each combination of stock levels, as follows. We can easily find the variance of the number of items in the repair shop, proceeding from a *M/M/k* queue. This variance, which is generally different from the mean, is used to improve the estimate of the system availability using the VARI-METRIC method.

Using the results for two simple systems (one echelon and two echelon), we can find the maximum availability error (absolute difference in availability between an *M/G/k* and an *M/G/∞* repair shop) corresponding to every combination of number of servers and utilisation rate. Then, we can make plots (Figure 4.1), which show us for every number of servers and for every utilisation rate the maximum error in the system availability if we ignore finite repair capacity.

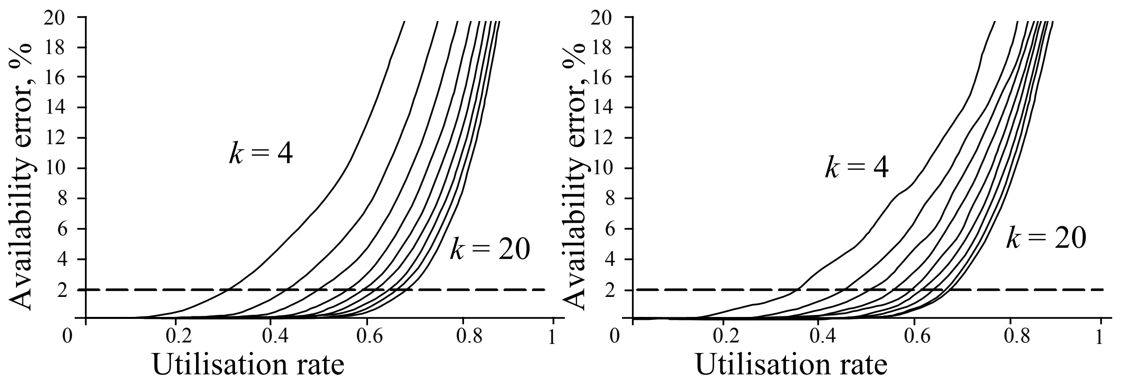

Figure 4.1. The availability errors for different numbers of servers and different utilisation rates (one- and two-echelons systems).

These plots show us the critical utilisation rates in simple systems. For example, if one tolerates an error of 2 percent points in availability for a two-echelon system with $k = 4$ servers, the maximum acceptable utilisation rate to use VARI-METRIC is approximately 0.35 or less (all values for which left function on every plot is below the dashed line).

4.3 Stock allocation under finite capacity

It has been shown in the previous section that ignoring finite capacity can have a significant influence on the estimate of the availability. Therefore, we will develop a variant of the VARI-METRIC method, taking into account finite capacity of repair shops.

Sherbrooke (1992) shows that the expected number of backorders depends on the mean and variance of the number of items in the pipeline. To calculate these pipeline characteristics, we need the mean and variance of the number of items in the repair shop. If the repair shop is modelled as *M/G/∞* queue, the number of items in the repair shop is Poisson distributed. However, such a simple expression is not available for single-class, multi-server queues in case of a general service time distribution (if repair shops are item-dedicated) or multi-class multi-server queues (if repair shops are cluster-dedicated). There are only few useful results from literature, because most queueing theory focuses on the mean and (sometimes) variance of the number of items *in queue* rather than *in the system* (in queue plus in service). It is hard to derive the variance of the total number of items in the system from these results, because (a) the variance of the number of items in repair is not known and (b) the numbers of items in queue and in repair are *not* independent. For the multi-class, multi-server *M/M/k* queue, we can use the results from Chapter 3. This will be discussed in section 4.3.1. Obviously, we can use these results for the *single*-class, multi-server *M/M/k* queue as well. Then, we discuss the impact of finite capacity on the optimisation procedure in section 4.3.2.

4.3.1 Multi-class *M/M/k* queue

The repair shops at the lowest echelon are usually run by a small crew of multi-skilled specialists that are able to handle a certain set of repair jobs. This gives rise to multi-class multi-server models, where several classes of items with their own arrival and service processes share the same queue and the same servers. A complication of this model is that the numbers of items in the system are *correlated* amongst classes. If the number of class *j* items is high at a particular point in time, it is likely that the servers had a high utilisation recently, and so the number of class *j'≠j* items in the system can expected to be high as well. The consequences of this phenomenon on the optimisation procedure will be discussed in section 4.3.2.

In this chapter, we will use the approximation for the *M/M/k* multi-class queue as presented in Chapter 3. This is quite easy from a computational point of view and it gives accurate results for high utilisation rates. As can be seen from section 4.2, this is the most interesting situation. For low utilisation rates, we might as well use the *M/G/∞* queue. The main feature of the multi-class queue is that all items of one cluster C are sharing the same repair facility, which has k_C servers and an utilisation rate ρ_C. Similar to Chapter 3, we define the relative arrival rate a_{jm} of item j at location m as

$$a_{mj} = \frac{\lambda_{mj}}{\Lambda_C}, \text{ where } \Lambda_C = \sum\nolimits_{(j')\in C} \lambda_{mj'}$$

The relative deviation of the service rate of item j at location m from its cluster average is denoted by the variable δ_{mj}, i.e.

$$\delta_{mj} = \frac{E\left[STm_C\right]}{E\left[STm_{mj}\right]} - 1, \text{ where } E\left[STm_C\right] = \frac{1}{\Lambda_C}\sum\nolimits_{(m',j')\in C} \lambda_{m'j'} E\left[STm_{m'j'}\right]$$

As the analysis focuses on a repair shop for a single cluster of items at a single location, we simplify the notation by omitting the cluster index C and the location index m. So we will work with an abstract queue with k servers and utilisation rate ρ. The queue is shared by $|C|$ (number of items in cluster C) items having arrival and service time characteristics a_j and δ_j, respectively ($j = 1,\ldots |C|$).

To analyse the system as a Markov chain, the system state is described by two vectors $\overline{w}$ and $\overline{s}$, where the i^{th} element w_i of the vector $\overline{w}$ gives the class of the i^{th} item in queue and the i^{th} element s_i of the vector $\overline{s}$ gives the class of the item in service at server i ($i = 1, 2, \ldots, k$). This abundant state representation facilitates the solution of the equilibrium equations. The approximate solution is based on constructing a product form solution of the equilibrium equations. It first gives an approximation for the system state probabilities, and next these probabilities are used to approximate the necessary performance measures, such as the mean and the variance of the number of items in the system for each class j ($E[R_j]$ and $Var[R_j]$). We can even approximate correlations between R_i and R_j ($i \neq j$) from the state probabilities.

The product form of the probability on system state $(\overline{w},\overline{s})$ is:

$$P(\overline{w},\overline{s}) = \left(\sum_{j=1}^{|C|} w_j\right)! \prod_{j=1}^{|C|} \frac{a_j^{\,w_j}}{w_j!} P_n(\overline{s}),$$

where $P_n(\overline{s})$ denotes the probability on server state $\overline{s}$ given that the total number of items in the system equals n. These probabilities can be calculated as shown in Chapter 3, section 3.5 using either exact or approximate methods. Here we present the performance measures based on the approximate system state probabilities.

To calculate the mean value, we have to find the sum in the following form:

$$E[R_i] = \sum_{r=1}^{\infty} r \sum_{\overline{w},\overline{s}} P(\overline{w},\overline{s} \mid w_i + s_i = r),$$

which can be divided into two terms ($R < k$ and $R \geq k$). The first term is a finite sum of probability states that can be calculated directly from formula (3.13). The second term is in fact an infinite series of state probabilities, to be approximated by (3.14). After some algebra, we find the following approximation for the mean number of class i items in the system:

$$E[R_i] \approx p_0 \sum_{r=1}^{k-1} r \frac{(k\rho)^r}{r!} \sum_{\bar{s}:\, s_i = r} \prod_{j=1}^{N} \frac{a_j^{s_j}}{s_j!(1+\delta_j)^{s_j}}$$
$$+ \sum_{\bar{s}} \sum_{j=1}^{N} \left(a_i (z_j - 1)^{-1} + s_i\right) \gamma_j (z_j - 1)^{-1} z_j D(\eta_j)^{-k} \prod_{l=1}^{N} \frac{a_l^{s_l}}{s_l!(1+\delta_l/\eta_j)^{s_l}}$$

Analogously, we can approximate the second moment of the squared number of items in the system:

$$E\left[R_i^2\right] \approx p_0 \sum_{r=1}^{k-1} r^2 \frac{(k\rho)^r}{r!} \sum_{\bar{s},\, \text{s.t}\, s_i = r} s_i \prod_{s=1}^{N} \frac{a_j^{s_j}}{s_j!(1+\delta_j)^{s_j}}$$
$$+ \sum_{\bar{s}} \sum_{j=1}^{N} \gamma_j \left(2a_i^2 (z_j - 1)^{-2} + a_i (2s_i + 1)(z_j - 1)^{-1} + s_i\right)(z_j - 1)^{-1} z_j D(\eta_j)^{-k} \prod_{l=1}^{N} \frac{a_l^{s_q}}{s_l!(1+\delta_l/\eta_j)^{s_l}}$$

From the two equations above, we can find the variance using $Var[R_i] = E\left[R_i^2\right] - E[R_i]^2$.

In Chapter 3 we have tested these approximations extensively and compared the approximate results to the exact results that we derived. It appeared that the approximation error remains within reasonable bounds (less than 10%). Besides, we observed that the approximation error decreases with the repair shop utilisation. The latter is relevant, because our method is especially meant for instances with high utilisation.

Remark

An interesting generalisation is to combine these results for the multi-class *M/M/k* queue with the approximate equations that Whitt (1993) developed from the relation between single-class *M/M/k* and *GI/G/k* systems. This approximation does not make a deep analysis of arrival and repair process and the only characteristics of this processes used in this approximation are squared coefficients of service and interarrival time variations ($C_{A_{mi}}^2$ and $C_{S_{mi}}^2$) of item *i* at location *m*. In this way, we arrive at the following new approximations for the multi-class *GI*/*G*/*k* queue:

$$E[R_{mi}]_{GI/G/k} \approx \left(\frac{C_{A_{mi}}^2 + C_{S_{mi}}^2}{2} \right) E[Q_{mi}]_{M/M/k} + k_{mj}\rho_{mj},$$

$$Var[R_{mi}]_{GI/G/k} = E\left[R_{mi}^2\right]_{GI/G/k} - \left(E[R_{mi}]_{GI/G/k}\right)^2 \tag{4.1}$$

and $$E\left[R_{mi}^2\right]_{GI/G/k} \approx \frac{E\left[R_{mi}^2\right]_{M/M/k}}{\left(E[R_{mi}]_{M/M/k}\right)^2} \cdot \left(E[R_{mi}]_{GI/G/k}\right)^2$$

where the characteristics of multi-class *M/M/k* queue as described in this section are substituted. This approximation still has to be tested, however.

4.3.2 Optimisation procedure

We will base our optimisation procedure on the VARI-METRIC method. We proceed from the definition of the system availability as introduced by Sherbrooke (1992):

$$A_m \approx E\left[1 - \sum_{j \in IND(1)} \frac{BO_{mj}\left(\overline{st}\right)}{B_m} \right] = 1 - \frac{1}{B_m} \sum_{j \in IND(1)} E\left[BO_{mj}\left(\overline{st}\right) \right] \tag{4.2}$$

and the average availability *A* over all systems at all downstream locations equals

$$A = 1 - \frac{1}{|ECH(N)|} \sum_{m \in ECH(N)} A_m \approx 1 - \frac{1}{|ECH(N)|} \sum_{m \in ECH(N)} \frac{1}{B_m} \sum_{j \in IND(1)} E\left[BO_{mj}\left(\overline{st}\right) \right] \tag{4.3}$$

As a consequence, maximising average availability is roughly equal to minimising the weighted average of expected backorders at each location.

As noted in Chapter 2, the approximations (4.2) and (4.3) can be refined by using a second order approximation rather than a first order approximation. If we use that $\left(1-\varepsilon\right)^k \approx 1 - k\varepsilon + \frac{k(k-1)}{2}\varepsilon^2$, we can modify equation (4.3) as

$$A_m \approx 1 - \sum_{j \in IND(1)} \left\{ \frac{E\left[BO_{jm}\left(\overline{st}\right)\right]}{B_m} - \frac{Z_j(Z_j-1)E\left[BO_{jm}^2\left(\overline{st}\right)\right]}{2B_m^2 Z_j^2} \right\} + \frac{1}{B_m^2} \sum_{\substack{i,j \in IND(1) \\ i<j}} E\left[BO_{im}\left(\overline{st}\right) BO_{jm}\left(\overline{st}\right)\right] \tag{4.4}$$

If the repair shops are modelled by *M/G/∞* queues, the backorders of various items are mutually independent. If the multiplicity of each item Z_i equals one, which is not uncommon in practical situations (cf. Rustenburg, 2000), the second term in the first summation drops out. Then we only need the mean number of backorders for all assemblies at all downstream locations to optimise availability.

However, the backorder distributions are *not* independent if they share the same repair facility having finite capacity. Then it is clear from formula (4.4) that we also need the correlations between backorders. In a multi-echelon, multi-indenture setting, these correlations may propagate throughout the network. For example, suppose that a central repair shop (say location 1) has finite capacity and repairs two different subassembly types (two classes), *A* and *B*. If subassembly *A* is part of assembly *i* and subassembly *B* is part of assembly *j*, there is theoretically a correlation between the backorders of assembly *i* at location $m \in CUS(1)$ and the backorders of assembly *j* at another location $m \in CUS(1)$. So the system availability at various locations (say A_n and A_m) may be dependent. However, this is not an issue when calculating the average availability *A* as in (4.3).

The optimisation procedure depends on the availability function and on its approximation. In principle, we can use the second order approximation, because the approximation for the multi-class *M/M/k* queue facilitates the computation of correlations between the pipeline distributions and hence also the computation of correlations between backorders. In Chapter 2 is shown the impact of using a second order approximation in a single-indenture, single-site model. Extension to general multi-indenture, multi-echelon models requires additional research.

Because we will base our numerical experiments on the case of the Royal Netherlands Navy as introduced by Rustenburg (2000), we will use the alternative availability function (2.7). Therefore, our goal function for our computations in section 4.4 is defined by (2.8). However, we emphasise that our approach also works with other definitions of availability.

4.4 Computational results

To test our model, we proceed as follows. We use the modified greedy optimisation heuristic as described in section 4.3 to calculate stock levels for a range of experiments. We focus on cluster-dedicated repair shops (modelled by multi-class, multi-server queueing systems), because this is a new aspect in the spare parts management literature that has not been examined before. Each time, we choose the budget such, that 95% system availability can be reached. Then, we simulate the system to estimate the true availability. Also, we approximate the system availability using the traditional VARI-METRIC method, assuming that the repair shop has infinite capacity and by plugging in the observed repair shop throughput time in the *M/G/∞* model for the repair shop. Next, we examine to which extent the inclusion of finite repair capacity in the VARI-METRIC model improves the availability estimates. In this approach, we use the theory as sketched in section 4.3.1 for deriving first and second moments of backorders at the operational level. Using these two moments, we fit a discrete distribution and find the probability of backorders at the operational level given a stock distribution. These backorder probabilities are the input of the greedy algorithm of Sherbrooke (1992) that optimises the stock levels such, that the inventory investment to attain a given fixed availability (say 95%) is minimised. Here the system availability is calculated by (2.7).

In the experimental design, we focus on cases where we expect that finite capacity has significant impact. In section 4.2, we already showed that the impact of finite capacity increases with repair shop utilisation and decreases with the number of servers. Therefore, we choose relatively high repair shop utilisation in our experimental design (≥0.8). We vary the following system characteristics: indenture structure, echelon structure, system failure rate, number of servers and repair shop utilisation rate. Altogether, we have 5 experimental factors. We will define two values for each factor and we will include all parameter combinations in our experimental design, resulting in $2^5 = 32$ experiments.

We use the system structure and the indenture structure as shown in Figure 4.2, where numbers display probabilities to be sent to higher echelon repair p_{mj}^{rep} and the cause probabilities p_{mjk}^{cause}. For simplicity, these probabilities are the same throughout the whole system.

We include both a 2-echelon and a 3-echelon system in our numerical experiments. Figure 4.2 shows that the 3-echelon system consists of 1 central depot, 4 frigates and 16 submarines. The 2-echelon system does not have an intermediate frigate-level and simply consists of the depot and 16 submarines. Similarly, we analyse a 3-indenture system as shown in Figure 4.2 and the corresponding 2-indenture system that arises when the lowest indenture level is omitted.

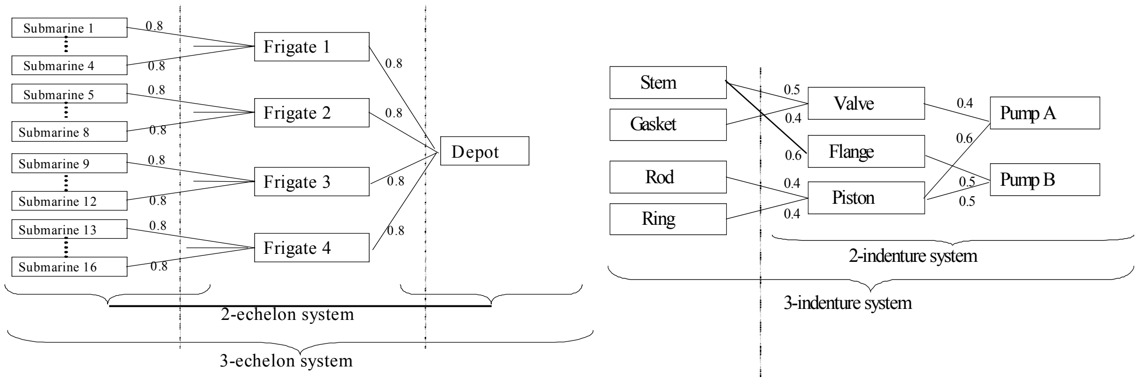

Figure 4.2. An example of a supply system and an item hierarchy

The installed base consists of one pump A and one pump B per submarine. We fix the failure rate of pump B in all experiments and vary the failure rate of pump A as shown in Table 4.1.

Table 4.1. Values for the failure rates in the experiments (mean number of failures per pump per year)

Notation	Pump A	Pump B
Low	10	15
High	19	15

All locations have finite capacity cluster-dedicated repair shops. As clusters, we take all items at the same indenture level. That is, each location has two respectively three (multi-class, multi server) repair shops in the two- respectively three-indenture model. One repair shop is dedicated to pump A and pump B, one repair shop is dedicated to the valve, the piston and the flange and the third repair shop (only in the 3-indenture model) is dedicated to the other items.

Next, we have to specify the repair shop parameters, namely the mean repair times and the number of servers per repair shop. We choose these parameters such, that the utilisation rates of all repair shops equal 80% and 95%, respectively. For the numbers of servers we define two different sets with 10 servers at each repair shop and with 3 servers, and repair times are defined completely by these parameters (arrival rates, utilisation rates and number of servers).

Table 4.2. Varying parameters of experiments

Echelons	Indentures	Failure rates	Number of servers per repair shop	Utilisation rates
Three	Three	Low	3	80%
Two	Two	High	10	95%

All parameter combinations (Table 4.2) lead to $2^5 = 32$ experiments. As mentioned above, we used our optimisation program to calculate the 32 different stock levels sets, which guarantee 95% system availability. Next, we estimated the true availability using discrete event simulation. The results of simulation runs (availability estimates with half lengths of the 95% confidence intervals) are given below.

Table 4.3. Results of the approximation program and the simulation runs

Echelons	Indentures	# servers	Arrival rates	Utilisation Rates	Finite capacity.	Simulation	∞ — capacity approximation
Three echelons	Three indentures	3	Low	80%	95.03%	93.9%±0.39	99.89%
				95%	95.01%	97.0%±0.23	100.00%
			High	80%	95.06%	94.8%±0.29	99.92%
				95%	95.01%	97.1%±0.64	100.00%
		10	Low	80%	95.03%	95.4%±0.37	99.10%
				95%	95.01%	96.9%±0.53	100.00%
			High	80%	95.00%	94.2%±0.43	99.14%
				95%	95.00%	96.9%±0.45	100.00%
	Two indentures	3	Low	80%	95.02%	94.6%±0.33	99.91%
				95%	95.01%	95.9%±0.42	100.00%
			High	80%	95.01%	95.9%±0.71	99.86%
				95%	95.01%	96.5%±0.47	100.00%
		10	Low	80%	95.00%	94.9%±0.52	99.33%
				95%	95.01%	95.7%±0.48	100.00%
			High	80%	95.00%	95.2%±0.51	99.24%
				95%	95.01%	96.1%±0.54	100.00%
Two Echelons	Three indentures	3	Low	80%	95.05%	94.1%±0.37	99.91%
				95%	95.01%	95.3%±0.52	100.00%
			High	80%	95.05%	95.0%±0.51	99.92%
				95%	95.01%	96.9%±0.41	100.00%
		10	Low	80%	95.01%	94.0%±0.50	99.92%
				95%	95.01%	94.8%±0.51	100.00%
			High	80%	95.05%	95.3%±0.45	99.91%
				95%	95.02%	96.2%±0.43	100.00%
	Two indentures	3	Low	80%	95.03%	95.3%±0.35	99.90%
				95%	95.01%	96.2%±0.56	100.00%
			High	80%	95.03%	94.3%±0.28	99.88%
				95%	95.01%	96.0%±0.51	100.00%
		10	Low	80%	95.02%	94.7%±0.42	99.37%
				95%	95.01%	93.5%±0.51	100.00%
			High	80%	95.00%	95.3%±0.45	99.24%
				95%	95.01%	96.2%±0.63	100.00%

It is clear that including the finite capacity of the repair shops leads to better results. The average deviation between simulation and the finite capacity approximation is 0.9%, whereas the average error is 4.8% if the infinite capacity assumption is used. Hence, the inclusion of finite repair capacity improves the approximation accuracy by a factor 5 on average.

To analyse these results in more detail, we examine how this approximation error depends on the five factors in our experimental design, see Table 4.4. The ordering of the factor levels in this table corresponds to Table 4.2.

Table 4.4. Average errors for all experiments and for different subsets of experiments.

Average error	Echelons	Indentures	Failure rates	Number of servers	Utilisation rates
0.89%	“3”- 1.01%	“3”-1.02%	“Low”- 0.83%	“3 set”- 0.97%	“80%” - 0.50%
	“2”- 0.77%	“2”-0.77%	“High”- 0.96%	“10 set”- 0.81%	“95%” - 1.28%

This table shows us that the utilisation rates have the strongest effect on the approximation error. The experiments with 80% percent utilisation rate have a smaller approximation error than the experiments with 95% utilisation rate.

Repeating the same analysis for the VARI-METRIC method (see Table 4.5), we see that the average approximation error is considerably higher for all subsets of the experiments.

Table 4.5. Average errors for the same set of experiments when the availability is estimated with the assumption of infinite repair capacity[2].

Average error	Echelons	Indentures	Failure rates	Number of servers	Utilisation rates
4.81%	“3”- 4.76%	“3”- 4.83%	“Low”- 4.82%	“3 set”- 4.93%	“80%” - 4.63%
	“2”- 4.86%	“2”- 4.78%	“High”- 4.80%	“10 set”- 4.69%	“95%” - 4.99%

[2] In fact, we calculate the mean number of items in repair using formulas of the multi-server queue, but we set the variance of this number equal to the mean.

Table 4 shows that the utilisation of the repair facilities has the highest impact on the approximation error.

To examine it in detail, we have run an additional set of experiments for two-echelon, two indenture systems. Here the failure rates are fixed to 15 (19) for Pump A (Pump B) and the numbers of servers to 3. The other parameters are defined by the repair and cause probabilities shown in Figure 4.2 and by the utilisation rates of the repair shops, which we vary from 0.4 to 0.95. The stock levels are optimised to a 95% availability level using our modification of the VARI-METRIC method.

In the next figure (Figure 4.3), we show the values of the system availability estimated by the classical VARI-METRIC method, by the modified VARI-METRIC method and by simulation.

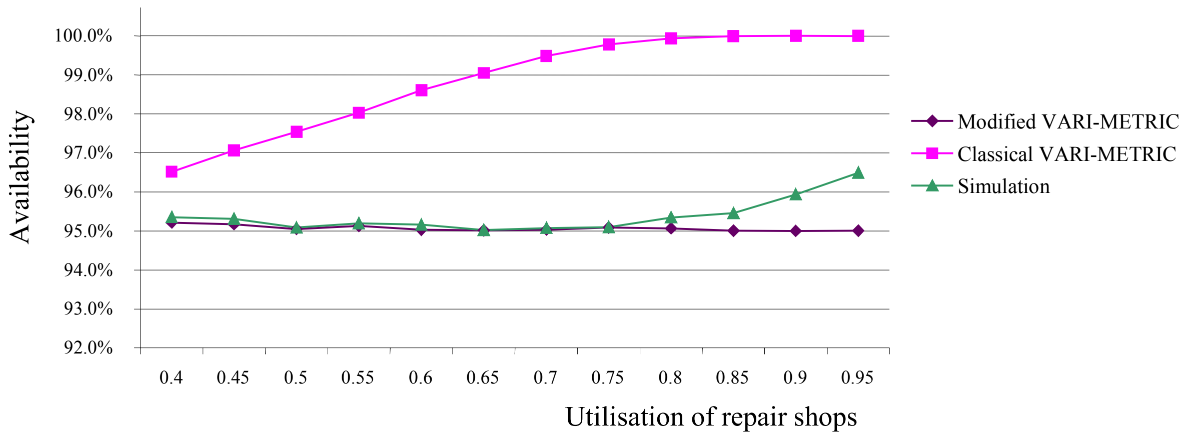

Figure 4.3 Comparison of system availabilities obtained by different approximations.

It is clear from this picture that the classical VARI-METRIC method significantly overestimates the system availability and, therefore, it underestimates the required stock levels. The modified VARI-METRIC method produces much better results if the utilisation of repair shops is not too high, say less than 0.85. The deviation for high utilisation rates is probably caused by backorder correlations that we ignored (see section 3.7.3). We can also see that the modified VARI-METRIC method underestimates the system availability for high utilisations, thereby guaranteeing that the target availability will be reached.

We stress that our results are valid only within our experimental range. As we already showed in section 4.2, the impact of the infinite capacity assumption is high if the repair shop utilisation is high, especially if the number of repair men in the shop is low (Figure 4.1).

Some side experiments showed that the infinite capacity model is appropriate indeed if the utilisation is low, especially if the number of servers is not too small. For example, the average error when using the infinite capacity assumption reduces to 1.5% if the utilisation equals 60% and the repair shop has 10 servers.

Heavy utilisation also leads to long queues and hence the simulation model requires more time to stabilise. In our experiments we have found that a simulation model of a single queue needs around 2000 subruns with 1000 arrivals each to arrive at 10%-errors in estimation of mean number of items in queue. The fraction of failed items that are repaired at the lowest indenture repair shops is determined by the parameters p_{mj}^{rep}. In our three-echelon model, we find that this fraction equals 0.2*0.2*0.2 = 0.008 (or 0.04 in case of two-echelon models). A rough estimate of the simulation run length required for three-echelon models yields that we need 2000 (subruns) * 1000 (item failures per subrun)/0.008 = 250,000,000 failures at each submarine for each model (out of 32 models). We could not make such long runs and we stopped the simulations after 50,000 - 60,000 failures at each submarine. However, the errors *in the system availability* are within reasonable bounds. It shows that the errors in estimation of lowest indenture repair shops performance have only a limited impact on the estimation of the system performance. On the other hand, we see that the high utilisation of repair facilities can destabilise the system. Despite the limit on the number of submarine failures, we still needed up to 200 hours for a single simulation run in some cases (3-echelon models). Although this is long, it is still manageable if the number of experiments is not too high as we did (32 experiments).

These experiments also show us that the stock levels at lowest echelon must be high to attain the target availability level (cf. Table 4.6). As a result, it can be cheaper to increase the capacity of the repair facilities then to keep many spare parts in stock. Hence, the optimisation of the availability with finite capacity repair shops can include a trade-off between stocks and servers capacities. A formal method for this trade-off still has to be developed. We will come back to this issue in Chapter 5.

Table 4.6. Optimal stock levels at *each* location of a three-echelon three-indenture model with 10 servers and "high" failures rate using the finite capacity model.

Utilisation	Echelons	Pump A	Pump B	Valve	Flange	Piston	Stem	Gasket	Rod	Ring
80%	1st echelon	2	3	7	13	16	4	4	5	7
	2nd echelon	3	5	6	12	16	4	3	4	6
	3rd echelon	12	13	8	15	19	5	4	5	7
95%	1st echelon	5	5	21	35	47	11	7	19	18
	2nd echelon	7	7	20	33	45	8	6	16	16
	3rd echelon	49	42	27	43	63	14	8	21	20

Another interesting question is whether the use of our model with finite capacities leads to significantly different decisions than the traditional infinite capacity model. Therefore, we compared the stock allocations using both models. As an example, we show in Table 4.7 the stock allocation based on the traditional infinite capacity model for the same cases as in Table 4.6. As throughput times in the infinite capacity repair shops, we used the throughput times corresponding to repair shops with 10 servers and an utilisation of 80% and 95% in the finite capacity model, respectively.

Table 4.7. Optimal stock levels at *each* location of three-echelon three-indenture model with "high" failure rates using the infinite capacity model; the mean repair throughput times are equal to finite capacity repair shops with 10 servers and a utilisation of 80% and 95%, respectively.

Mean repair throughput time	Echelons	Pump A	Pump B	Valve	Flange	Piston	Stem	Gasket	Rod	Ring
Equal to repair shop with 10 servers and 80% utilisation	1st echelon	3	4	6	12	15	4	4	4	6
	2nd echelon	4	6	6	11	14	3	3	3	6
	3rd echelon	9	11	7	13	16	4	4	4	7
Equal to repair shop with 10 servers and 95% utilisation	1st echelon	10	10	14	23	35	10	6	11	13
	2nd echelon	13	13	14	23	34	9	6	10	12
	3rd echelon	22	22	15	25	37	11	7	11	13

The shifts in stock allocation as shown in both tables are representative for our experiments. We see that decisions are modified in the following direction:

1. The total number of items on stock is considerable higher using the finite capacity model. As we also see from Table 4.3, the infinite capacity model overestimates the real system availability. Therefore, the model suggests to put considerably less items on stock than actually required to reach the target availability.
2. Using the finite capacity model, a somewhat larger fraction of the stocks is allocated to the downstream locations.
3. There is no clear shift in stock distribution over indenture levels. That is, the distribution of stocks over first, second and third indenture is approximately similar for both the finite and infinite capacity model in our experiments.

4.5 Conclusions

In this chapter, we studied multi-echelon, multi-indenture service parts supply systems with finite repair capacity. We considered both item-dedicated repair shops, modelled by *M/M/k* queueing systems, and cluster-dedicated repair shops, modelled by multi-class *M/M/k* queueing systems. Our finite capacity VARI-METRIC approach yields more accurate results than the traditional approach of modelling repair shop throughput times by *M/G/∞* queues if the utilisation is high (say > 0.7), and especially if the number of servers is low (say < 4). In our numerical experiments, we found that the average absolute deviation between estimated and simulated availability is only 0.9% using our method whereas this average deviation is 4.8 % if we use the traditional infinite capacity VARI-METRIC approach. The VARI-METRIC approach generally overestimates the system availability. Herewith, we answered research question 3 from Chapter 1

Regarding the spare part stocks distribution within the system, we see that our finite capacity model leads to some shift in stock distribution to downstream locations. The distribution over the indenture levels is hardly affected, however.

We note that our conclusions are only valid within our experimental range, i.e. if the repair shop utilisation is relatively high. Naturally, our finite capacity model converges to the traditional infinite capacity approach if the utilisation decreases. Therefore, our approach is especially useful for practical situations in which (some of) the repair shops have a high work load of urgent repair jobs. Obviously, there are also cases where the traditional infinite capacity model is sufficient. This is true if outsourcing of repair jobs against little additional costs and lead time is possible or if the repair shop has low priority jobs (e.g. preventive maintenance or special projects) causing that the utilisation for high priority repair jobs is relatively low.

Furthermore, our method facilitates what-if analysis with important design parameters as the number of servers in the repair shops and the assignment of items to repair shop clusters. Another topic is refining the finite capacity model for the repair shop to multi-class, multi-server priority systems. Such a model enables us to improve further on the system efficiency. We will elaborate on the latter issues in Chapter 8. Such analysis is not possible for sure using the traditional infinite capacity VARI-METRIC approach.

Chapter 5
Trade-off between inventory and repair capacity in spare part networks

Abstract

The availability of repairable technical systems depends on the availability of (repairable) spare parts, to be influenced by (1) inventory levels and (2) repair capacity. In this chapter, we present a procedure for simultaneous optimisation of these two factors. Our method is based on a modification of the well-known VARI-METRIC procedure for determining near-optimal spare part inventory levels and results for multi-class, multi-server queueing systems, representing repair shops. The modification is required to avoid non-convexity problems in the optimisation procedure. To include part time and overtime working, we allow for a non-integer repair capacity. To this end, we develop a simple approximation for queueing systems with a non-integer number of servers. Our computational experiments show that the optimal utilisation rate of the repair servers is usually high (0.80-0.98) and depends mainly on the relative price of the servers compared with inventory items. Further, the size of the repair shop (the minimal number of servers required for a stable system) plays its part. We also show that our optimisation procedure is robust for the choice of the step size for the server capacity. The obtained results answer research question 4 (Chapter 1).

This chapter is based on the paper:

"A trade-off between inventory and repair capacity in spare part networks." A. Sleptchenko, M.C. van der Heijden, A. van Harten, accepted for publication in *Journal of Operational Research Society*.

5.1 Introduction

Timely supply of spare parts is essential to attain a high system availability. This can be achieved by holding sufficient stock and/or minimising the repair shop throughput times. This provides us with a trade-off between spare part inventories and repair capacity: we can improve the system availability by (1) increasing spare part inventories to buffer against repair shop throughput times, and by (2) investing in additional repair capacity, thereby reducing the number of items waiting for repair and thus inventory requirements. So, an optimal investment allocation to both spare part inventories and repair capacity is needed for an efficient system operation.

The number of decision variables in such an optimisation problem may be very large. Stock levels have to be determined for each item in the product hierarchy and for each stocking location. Also, the number of servers per repair shop has to be chosen. In principle, each location in the multi-echelon structure may have one or more repair shops, each having the ability to carry out some specific repair jobs. A repair shop may be *generic* (able to carry out all repair jobs for all items) or *dedicated* to a specific set of items and repair jobs. In this chapter, we assume that the number and type of repair shops are given. A location may have one or more repair shops, each being dedicated to a fixed but different set of items. Such repair shops are modelled by multi-class, multi-server queueing systems (cf. Chapter 3). We aim to find the optimal values of the decision variables spare part stock levels (for each item and each location) and repair capacities (for each repair shop ant each location). We consider two variants of our optimisation problem:

(i) maximise the average availability of all field systems given a fixed investment budget for spare part inventories and repair capacities;

(ii) minimise the investment budget for parts and repair capacity given a target value for the system availability;

To our knowledge, *analytical* models for the trade-off between spare part inventories and repair capacity are not available in literature. Our contribution is two-fold. Firstly, we will present an extension of VARI-METRIC facilitating the simultaneous optimisation of inventories and repair capacity. This method is useful for companies to judge whether they can better spend their budget to additional repair capacity (thereby reducing repair throughput times) or to additional spare parts (thereby buffering against long repair throughput times). Second, our method provides us the opportunity to gain insight in the appropriate repair shop utilisation and the allocation of budget to spare parts and repair capacity (see section 5.4). Note that our algorithm does not lead to an optimal solution, although the analysis of the *M/M/k* multi-class queue that we use is exact. Even the case infinite repair shops is already that complex, that optimal stock levels cannot be guaranteed using VARI-METRIC, except for some special cases.

In the next section, we give details on our model. In section 5.3, we present our optimisation procedure. Numerical experiments and their results are the subject of section 5.4. Our conclusions follow in section 5.5.

5.2 The model

In this section, we describe the basics of VARI-METRIC (5.2.1) and the issues when optimising repair capacity and spare part inventories simultaneously (5.2.2).

5.2.1 The VARI-METRIC-approach for inventory optimisation

Additionally to the general notation as introduced in Chapter 2 we will use the following notation:

$\overline{k}$	= the matrix of capacities with elements k_{lm}, denoting the number of servers in repair shop *l* at location *m*.
c_j^{stock}	= the price of item *j*.
c_{lm}^{cap}	= the price of one server in repair shop *l* at location *m*.
$PL_{mj}\left(\overline{st},\overline{k}\right)$	= the number of type *j* items in the pipeline at the location *m*, i.e. all items *j* under repair at location *m*, waiting for subassembly replacement or on order at the supplier of location *m*.
$BO_{mj}\left(\overline{st},\overline{k}\right)$	= the number of backorders (unfilled demand) for item *j* at location *m*.

VARI-METRIC focuses on maximising the average availability of the installed base using a fixed budget by allocating spare part stocks, assuming infinite capacity repair shops. Sherbrooke (1992) shows that this objective is approximately equivalent to minimising the sum of the expected backorders of all highest indenture items at all downstream locations. Rustenburg et al. (2001) show that under certain conditions the availability function can be expressed in the backorder *probabilities* rather than the expected backorders. This is the case if each base serves one system and if each item type occurs only once in each system. Then, maximising the average system availability is exactly equivalent to minimising the sum of backorder probabilities and the objective function is:

$$F\left(\overline{st},\overline{k}\right)=\sum_{m\in ECH(N^{ech})}\sum_{j\in IND(1)}P\left[BO_{mj}\left(\overline{st},\overline{k}\right)>0\right] \tag{5.1}$$

where the repair shop capacities $\overline{k}$ are traditionally infinite. In general, the optimisation algorithms of the VARI-METRIC class can be used for both types of objective functions (sum of backorder probabilities and sum of expected backorders). We will refer to model variants (i) and (ii), as stated in Chapter 2, for Sherbrooke's objective function and to model variants (i)' and (ii)' for Rustenburg's object function. Note that the number of backorders is related to the number of items in the pipeline by the following equation:

$$BO_{mj}\left(\overline{st},\overline{k}\right)=\max\left\{PL_{mj}\left(\overline{st},\overline{k}\right)-st_{mj},0\right\} \tag{5.2}$$

Sherbrooke (1992) shows how for infinite capacity repair shops the first two moments of the number of items in the pipeline PL_{mj} can be calculated from demand, repair and order time characteristics. We refer to Chapter 2 for mathematical details.

Based on the equations for the pipeline and backorder characteristics, VARI-METRIC uses a greedy algorithm to optimise item stock levels within a budget constraint. The idea is very simple: in each iteration we add an item j^* to stock at location m^*, such that we get a maximum decrease in backorder probabilities (and thus a maximum increase of availability) per unit of money spent. Let us denote:

$$\Delta_{jm}^{stock}=\frac{F\left(\overline{st},\overline{k}\right)-F\left(\overline{st}+e_{jm},\overline{k}\right)}{c_j^{stock}}, \tag{5.3}$$

where e_{jm} is a zero-matrix having only element (j, m) equal to 1. Then we choose for (j^*, m^*) maximising Δ_{jm}. Rustenburg et al. (2001) show that nonnegative starting values for the stock levels are required in order to avoid non-convexity problems in the greedy algorithm. They recommend certain starting values for two-echelon models and *infinite* repair capacity. We have to reconsider these starting values since we also have to set the repair shop capacities initially. We will return on this issue in section 5.3.2.

5.2.2 Issues for the inventory-capacity trade-off

To be able to optimise stock levels and repair shop capacities simultaneously, we need prices for repair capacity to compare to spare part prices. As spare parts are purchased and can circulate through the system during many years, we should calculate such a kind of "purchasing price" for repair men keeping this in mind. In other words, we should use the net present value of all the expenditures for a repair man in the next, say, 5 years. These expenditures include wages, taxes and social premiums, housing, education, tools. etc. As a consequence, the "price" of a full-time server in a repair shop may be very high, say around 500,000 Euro. Because of the high price of repair capacity, one might consider repair shops with a non-integer number of servers. Such a number, say 4.5, may represent various practical intermediate solutions, such as:

- using overtime or flexible labour contracts; for example, 4 servers each working 9 hours per day instead of 8 lead effectively to a repair shop with 4.5 servers (4x9 = 4.5x8); as repair jobs usually take multiple days, one might also model this case as a repair shop with 4 servers working 12.5% faster.
- contracting part-time personnel; then, 4.5 represents an alternating process between 4 and 5 servers (e.g. only 5 servers on Tuesday, Wednesday morning and Thursday); part time working can also imply that 5 servers work 7.2 hours per day each (10% less than 8 hours), which can be modelled as a repair shop with 5 full-time servers each working 10% slower.
- outsourcing some repair jobs or hiring external repair men; then the repair capacity is an alternating process between 4 and 5 (or even 6) servers, but the actual capacity depends on work load; also, the price of outsourcing or hiring may be different from own personnel;

We see that repair shops with a non-integer number of servers may have various practical interpretations. In fact, each interpretation requires its own model. Of course, this is hard to include in an integral optimisation framework. Therefore, we prefer to use a relatively simple model. We will go into more detail on the exact modelling of repair shops having a non-integer number of servers in the next section. Thereby, we assume that the price per server is fixed for each repair shop (but the price may vary between repair shops).

5.3 Simultaneous inventory and repair capacity optimisation

5.3.1 Repair shops with a non-integer number of servers

For a VARI-METRIC-like optimisation procedure, we need to calculate the first two moments of the distribution of the number of items in each repair shop. For an integer number of servers, we can use the results for the *M/M/k* multi-class queue as described in Chapter 3. When the number of servers is not an integer, various modelling options can be considered. Two options have already been mentioned in the previous section: [1] round the number down to the nearest integer and assume that these servers work somewhat faster (modelling overtime), and [2] round the number up to the nearest integer and assume that these servers work somewhat slower (modelling part time work). Both options are relatively easy to analyse, but yield different results for the distribution of the number of items in the system (and hence for the mean number of backorders).

A variant on option [2] is to use server dependent repair times, where one or more servers work at a slower speed than others. Although such a model is not difficult to analyse and can be found as an exercise in a queueing theory course (Adan, 2000), it leads to significant computational effort. It is necessary to compute repair shop characteristics (and consequently the availability of the whole system) each time when we make a slight modification of the repair shop capacities.

Finally, we can use some interpolation procedure proceeding from the performance characteristics of queueing systems with an integer number of servers. However, we found that the interpolation function has to be selected carefully. If not, the goal function is not convex anymore and a greedy optimisation procedure may end up in a local optimum that is considerably worse than the global optimum. Another issue is that interpolation only is not sufficient, but extrapolation is required as well. For example, suppose that the smallest integer number of servers possible in a repair shop equals 2 and that the utilisation equals 0.6. Then it is possible that the optimum number of servers in this repair shop equals some smaller number, for example 1.5 servers having utilisation 0.8. It is well-known that extrapolation may lead to serious approximation errors.

Therefore, we chose to combine option [1] and [2]: a non-integer number of servers *k* is represented by a weighed average of

1. a repair shop with $\lfloor k \rfloor$ servers, each working at a *faster* rate such that utilisation is the same, and
2. a repair shop with $\lceil k \rceil$ servers, each working at a *slower* rate such that utilisation is the same.

For the backorder calculations, we need first two moments of the number of items in the queueing system *R*. So, we use the following approximations:

$$\begin{aligned} E[R]_{frac} &\approx \left(1-\left(k-\lfloor k \rfloor\right)\right)E[R]_{\text{"}fast\text{"}}+\left(k-\lfloor k \rfloor\right)E[R]_{\text{"}slow\text{"}} \\ Var[R]_{frac} &\approx \left(1-\left(k-\lfloor k \rfloor\right)\right)Var[R]_{\text{"}fast\text{"}}+\left(k-\lfloor k \rfloor\right)Var[R]_{\text{"}slow\text{"}}, \end{aligned} \tag{5.4}$$

where “fast” and “slow” denote the system characteristics of the approximations 1 and 2 respectively.

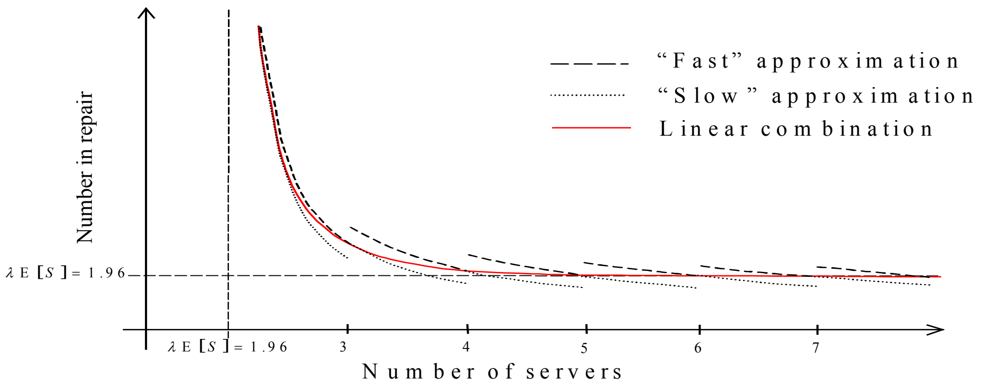

Figure 5.1. Approximation for the mean number of items in a system with a fractional number of servers

In Figure 5.1, we present the behaviour of the approximation. We take $\lambda E[S] = 1.96$ and vary the number of servers from 2 to 8 (equivalent to a utilisation rate between 0.98 and 0.245). The figure shows that the mean number of items in the system is higher for a repair shop with “fast” servers than for a repair shop with “slow” servers (having the same utilisation). A similar phenomenon applies to the variance. As approximation (5.4) provides a smooth curve between two reasonable bounds, we chose to use this approximation for our optimisation.

5.3.2 Optimisation procedure

As mentioned in the introduction, we consider three variants of the optimisation problem. We will use variant A, maximise the availability given a fixed budget, to explain how the procedure works in general. Next, we will discuss how we can use similar procedures to deal with the other two variants.

In principle, we can extend the marginal (local search) VARI-METRIC algorithm in a straightforward way. Now additional system availability can not only be obtained by increasing the spare part inventory levels at a certain location, but also by increasing repair shop capacity. Therefore, we need a second set of marginal values for the repair shop capacity, next to the marginal values for the stock level $\Delta_{j^* m^*}^{stock}$ as defined by (5.3):

$$\Delta_{lm}^{cap} = \frac{F\left(\overline{st}, \overline{k}\right) - F\left(\overline{st}, \overline{k} + d_{lm}\right)}{\varepsilon c_{lm}^{cap}} \tag{5.5}$$

where d_{lm} is a zero-matrix having only element (l, m) equal to ε ($0 < \varepsilon \leq 1$). Here ε denotes a certain (non-integer) step size for the server capacity. In each step of the greedy algorithm, we search the maximum value of all $\Delta_{j^* m^*}^{stock}$ and $\Delta_{l^* m^*}^{cap}$ over all locations m, repair shops l and items j. Depending on the maximum, either the corresponding stock level or the corresponding repair capacity is increased. We continue the iterations until the given budget has been used, or until the target system availability has been reached.

Still, the step size for the (non-integer) number of servers ε and the starting values $\left\{\overline{st}, \overline{k}\right\}$ of the algorithm have to be defined. It is reasonable to define the step size such, that the price of a fractional server has the same magnitude as an item (i.e. either the most expensive or the cheapest spare part). As this still allows for a wide range of values for ε, we will examine the sensitivity of our algorithms' solution to the step size ε in section 5.4.

As starting values for the matrix $\overline{st}$, the equations of Rustenburg et al. (2001) cannot be used, because these equations depend on the repair shop throughput times. If the capacity is finite but unknown (because it is a decision variable), these throughput times are not known in advance. Therefore, we modified Rustenburg's equation by assuming that the number of servers coincides with a low utilisation of 0.4. However, we found that this may still leave us with serious non-convexity problems, especially if the system availability is low. In Figure

5.2, we show the average availability as function of the budget used during the optimisation procedure for a case in which we encountered problems ("Classical VARI-METRIC"). The availability function shows many 'jumps' at several values of the budget. This is caused by the fact that stock levels and server capacities are not balanced over various locations during the algorithm. In contrary, first one location (e.g. submarine) receives a lot of stock and server capacity, then the algorithm moves to the next (identical) location.

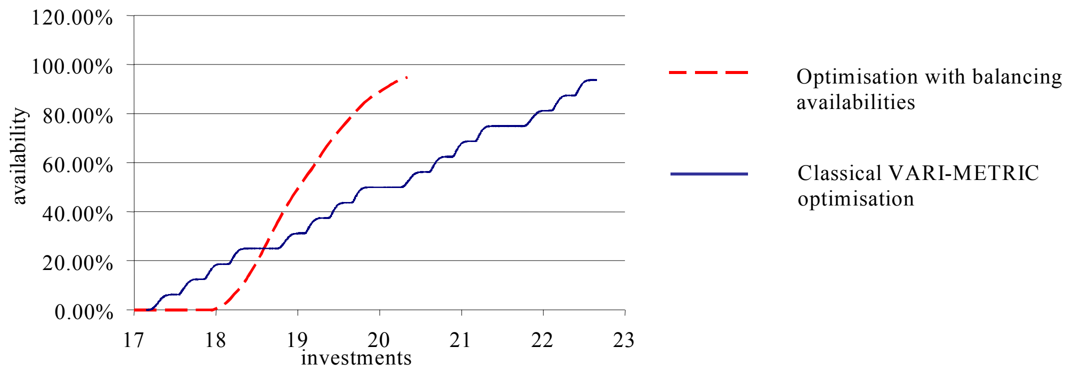

Figure 5.2. Misbehaviour of the greedy optimisation procedure and the impact of availability balancing

To solve these problems, we may consider the following solutions:

1. Use higher starting values for the number of servers and for the stock levels. However, we need less stock if we have more servers and the other way round. As a consequence, we have to deal with a trade-off between repair shop capacity and stock levels during the initialisation. It shifts the problem to the initialisation and it is not straightforward how to solve this.

2. Use a look-ahead heuristic by calculating the reduction in backorder probabilities for two or more sequential steps and select the best option. A drawback is that the number of function evaluations increases drastically, whereas it is not sure whether non-convexity will be avoided.

3. Balance the availability of all systems in the installed base during the optimisation procedure.

Because of the drawbacks of the first two options, we chose to select option 3. Balancing availabilities can be achieved in the following way. If it is best to add an item to stock on one location, we add the same item to stock on all equivalent locations in the other branches of the network. For example, if we add one item to Submarine 1, only the availability of the installed base (Pump A and/or Pump B) at submarine 1 will be improved to a certain level. Then we add the same item to stock at the other submarines, insofar the availability of the pumps at those submarines does not surpass the availability at Submarine 1. In a perfectly balanced case (same demand and repair characteristics for all branches in the multi-echelon network), we always add one item to stock at all equivalent locations (submarines or supply ships). Similarly, we balance the repair shop capacities in equivalent repair shops throughout the network. This will avoid the problems as shown in Figure 5.2, but it has the drawback that we use more budget in each iteration. This is a problem if we approach our budget limit, but it can easily be resolved by implementing the final decision partially or by moving to the traditional one-by-one heuristic. The latter is possible, because most serious problems with the function behaviour occur if the average system availability is very low.

The dashed line in Figure 5.2 shows that our modification leads to a better optimisation procedure and solution indeed. For this specific case, we found a cost reduction of 10% while attaining the same availability of 95%. Besides, our modified greedy algorithm is considerably faster, because multiple stock levels or repair shop capacities may be increased in a single iteration. Still it is true that the problems as shown in Figure 5.2 only occur in a part of the cases. Also, we found that such problems mainly occurred with the sum of backorder probabilities as objective function as suggested by Rustenburg (2000). We did not find problems with the sum of expected backorders as objective function as suggested by Sherbrooke (1992).

Because we chose to use availability balancing as described above, we can simply set zero stock levels and a minimal number of servers as initialisation. To prevent numerical problems arising from extremely high utilisation rates, we set the numbers of servers $\overline{k}$ initially such that the utilisation equals 0.98. Now we can summarise our optimisation heuristic as follows:

Algorithm: Stock/capacity trade-off optimisation

```
Step 1. Set the initial stocks levels to zero and the initial numbers of servers as small as
    possible guaranteeing a utilisation of 0.98. Set the budget spent $\hat{C}$ equal to total costs of
    the initial repair shop capacities.
Step 2. Calculate $\Delta^{stock}_{j^*m^*}$ and $\Delta^{cap}_{l^*m^*}$ for all stocks and servers according to (5.3) and (5.5), using
        (5.4).
Step 3. Determine the maximum of the increments $\Delta^{stock}_{j^*m^*}$ and $\Delta^{cap}_{l^*m^*}$ over all l, j and m.
        If $\Delta^{stock}_{j^*m^*} > \Delta^{cap}_{l^*m^*}$, then go to Step 4, else go to Step 5.
Step 4. Set the marginal costs of the best improvement equal to $c^{stock}_{j^*}$.
        If $\hat{C} + c^{stock}_{j^*} \le \text{budget}$, then
            $st_{j^*m^*} := st_{j^*m^*} + 1;\ \hat{C} := \hat{C} + c^{stock}_{j^*}$
          Set $PBO(m^*)$ := sum of backorder probabilities for the installed base in the subtree
              of location $m^*$
          Let n be the echelon of location $m^*$;
          For all other locations $m \in ECH(n)$ and while $\hat{C} + c^{stock}_{j^*} \le \text{budget}$ do
              while $PBO(m) \le PBO(m^*)$ do $st_{j^*m} := st_{j^*m} + 1;\ \hat{C} := \hat{C} + c^{stock}_{j^*}$
        Go to Step 6.
Step 5. Set the marginal costs of the best improvement equal to $\varepsilon c^{cap}_{l^*m^*}$.
        If $\hat{C} + \varepsilon c^{cap}_{l^*m^*} \le \text{budget}$, then
            $k_{l^*m^*} := k_{l^*m^*} + \varepsilon;\ \hat{C} := \hat{C} + \varepsilon c^{cap}_{l^*m^*}$ Calculate $PBO(m^*)$; Let n be the echelon of location
        $m^*$.
          For all other locations $m \in ECH(n)$ and equivalent repair shops l and while
        $\hat{C} + \varepsilon c^{cap}_{lm} \le \text{budget}$ do
              while $PBO(m) \le PBO(m^*)$ do $k_{lm} := k_{lm} + \varepsilon;\ \hat{C} := \hat{C} + \varepsilon c^{cap}_{lm}$
              Recalculate the repair shop characteristics using (5.4)
Step 6. If $\hat{C} < \text{budget}$, then go to Step 2, else STOP
```

Note that the balancing of availability as discussed before takes place in the inner loops at the end of step 4 and step 5. Recall that this algorithm solves problem variant (i)’ from the Chapter 2. In a similar way, we can define an algorithm for variant (ii)’: minimise the budget for parts and repair capacity given a target availability. Then, we should only modify the stop criterion in step 4, 5 and 6. Instead of the budget restriction, we use a restriction on the average system availability $A\left(\overline{st},\overline{k}\right)=\frac{1}{\left|ECH(N^{ech})\right|}\sum_{m\in ECH(N^{ech})} A_m\left(\overline{st},\overline{k}\right)$. Here we denote the availability at location *m* by $A_m\left(\overline{st},\overline{k}\right)=\prod_{j\in IND(1)} P\left[BO_{mj}\left(\overline{st},\overline{k}\right)=0\right]$. Now the stop criterion in step 6 (and similarly in step 4 and 5) becomes:

$$A\left(\overline{st},\overline{k}\right)=\frac{1}{\left|ECH(N^{ech})\right|}\sum_{m\in ECH(N^{ech})}\prod_{j\in IND(1)} P\left[BO_{mj}\left(\overline{st},\overline{k}\right)=0\right]\geq A_{Target}$$

5.4 Computational results

In this section we discuss some numerical experiments using our algorithm with the following goals:

1. obtaining indications for optimal repair shop utilisation rates,
2. examining the sensitivity of the results to the pricing of servers,
3. examining the impact of the step size for the (non-integer) repair shop capacity on the optimisation.

5.4.1 Experimental design

We choose Variant (i)’ of our algorithm in all our experiments. Our experimental design is derived from the one presented in Chapter 4. We consider six experimental factors: indenture structure, echelon structure, system failure rate, minimal number of servers, price of servers, and size of server step. The network and indenture structure are shown in Figure 5.3.

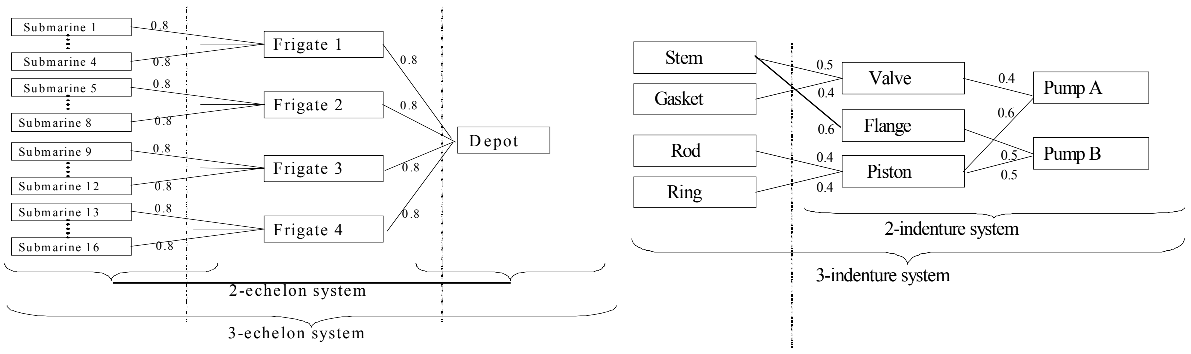

Figure 5.3. An example of a supply system and an item hierarchy

The *echelon structure* as experimental factor means that we consider both a 2-echelon and a 3-echelon system as shown in Figure 5.3. The 2-echelon system does not have an intermediate supply ship level, only the depot and 16 submarines. Similarly, for the *indenture structure* as an experimental factor we consider a 3-indenture system as shown in Figure 5.3 and the corresponding 2-indenture system that arises when the lowest indenture level is omitted. The installed base consists of one pump A and one pump B per submarine. The *failure rate* scenarios are given in Table 5.1: We fix the failure rate of pump B in all experiments and vary the failure rate of pump A.

Table 5.1. Values for the failure rates in the experiments (mean number of failures per pump per year)

Notation	Pump A	Pump B
Low	10	15
High	19	15

A repair shop is able to handle a given set of items. In our experiments, we assume that all items having the same indenture level are assigned to the same repair shop. That is, each location has two or three repair shops, depending on the scenario for the indenture structure. Referring to Figure 5.3, we see that the number of product classes in a repair shop equals 2 (system level), 3 (assembly level) and 4 (subassembly level). Further, the repair shop parameters are defined by the minimal number of servers, the arrival rate of failed parts and the repair times. For the *minimum number of servers* we consider two cases with either 2 servers or 10 servers per repair shop. We chose the mean repair times such, that the utilisation rate for the minimal number of servers equals 0.98. Although the number of servers is one of the optimisation parameters, the minimal possible number of servers defines in this way the sensitivity of the server utilisation to the number of servers, therefore it plays an important role in the optimisation procedure.

The *price of a server* is important if considered as a relative value compared to the item prices as shown in Table 5.2. In our experiments, the price of a server equals either 10 times or 100 times the price of the most expensive item. Referring to Table 5.2, we see that the server price varies as 71,000 and 710,000 euro.

Table 5.2. Item prices

Pump A	Pump B	Valve	Flange	Piston	Stem	Gasket	Rod	Ring
5000	7100	200	300	500	120	60	50	80

We vary the *server step size* ε as well, as a choice has to be made between adding one item to stock and increasing server capacity of one repair shop by ε in each iteration of the algorithm. The required computation time of our algorithm depends on the step size. As an indication, we found in our experiments that the computation time varied between 2 minutes and 5 hours on a Pentium III 800 MHz PC for step size $\varepsilon = 0.1$, whereas for step size $\varepsilon = 0.003$ our algorithm took three times more time on average. The three - echelon, three - indenture systems take the most time. Of course, we deal with a tactical decision that is taken only a few times per year, so that computation times are less relevant. Still, a step size that is too small may cause that practical problems with thousands of parts cannot be solved anymore.

To examine the impact of the step size, we set the step size such, that the value of ε additional servers equals the price of the cheapest item and the price of the most expensive item, respectively. Depending on the scenario for the server price, this implies that the value of ε varies between 0.00007 and 0.1. To avoid excessive computation times because of extremely small and practically useless step sizes, we take $\varepsilon = 0.003$ as a lower bound.

In the next table, we summarise all parameter settings.

Table 5.3. Varying parameters of experiments

Echelons	Indentures	Failure rates	Minimal number of servers per repair shop	$\frac{\text{Price of a server}}{\text{Price of an item}}$	Server steps
Two	Two	Low	2	10	~ cheapest item
Three	Three	High	10	100	~ most expensive item

5.4.2 Results

The large amount of experiments and their extensive results (tables of the optimal stock and server allocation) does not allow us to present all details here. Therefore, we will only show the key performance characteristics.

Table 5.4. Average utilisation rates of repair servers in the system for different parameters

Experimental factor	**Echelons**	**Indentures**	**Failure rate Pump A**	**Minimal number of servers per repair shop**	**Quotient of server price and item price**	**Server steps**
Low/high value	2, 3	2, 3	10, 19	2, 10	10, 100	0.1, 0.003
Optimal utilisation rate for low/high value	0.92	0.91	0.92	0.90	0.89	0.91
	0.92	0.93	0.92	0.95	0.96	0.92

First, we summarise the average repair shop utilisation rates in Table 5.4. Each cell in the table gives the average optimal utilisation rate over all repair shops in the 32 experiments corresponding to a low (high) value for a certain experimental factor. We see that the average utilisation varies around 0.9. We found that the utilisation rates of *individual* repair shops vary between 0.74 and 0.98, and in the most cases, the optimal utilisation is between 0.80 and 0.95. Table 5.4 also shows the impact of our six experimental factors on the optimal utilisation rate. We see that the optimal utilisation rate is only significantly influenced by two factors, namely the relative price of repair capacity compared to item costs and the minimum number of servers per repair shop. Obviously, a high price of repair capacity yields a high utilisation rate. Also, a higher utilisation rate is recommended if repair shops are relatively large (10 servers or more). This is not surprising, since we know from queueing theory that a queueing system with a small number of servers has more items in the queue than a queueing system with more servers and the same utilisation rate.

Also, we found that the optimal utilisation rates in central repair shops (on average 0.95 and varying between 0.80 and 0.98) are usually somewhat higher than in local repair shops (on average 0.92 and varying between 0.74 and 0.98). This can be explained by the fact that long throughput times in central repair shops are usually less critical and by scale effects (higher demand rates). Comparing repair shops at the same location, we see that those shops receiving the lowest percentage of items entering the location usually have the highest utilisation.

In Figure 5.4, we examine the relation between the relative price of repair capacity and the optimal utilisation rate in more detail. We show the minimum, average and maximum utilisation rate for all repair shops in experiments two-echelon, two-indenture systems with low failure rate, both for small and large repair shops (minimum 2 and 10 servers, respectively). The relative price of servers is varied between 10 and 100 times the price of the most expensive item. As could be expected, the repair shop utilisation is much more sensitive to the relative server price if the repair shop is small. In all cases, we find relatively high utilisation rates (≥ 0.74).

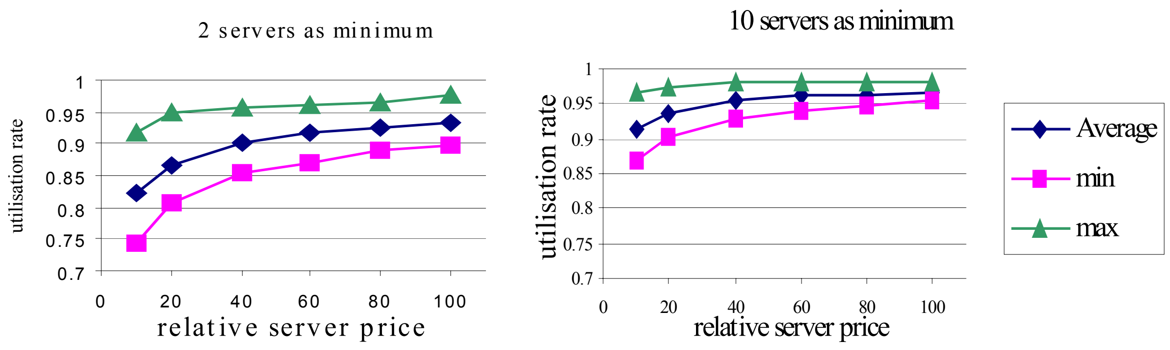

Figure 5.4. Relation between relative server price and utilisation in 2-echelon, 2-indenture systems

The trade-off as mentioned in the title of the chapter is illustrated in Figure 5.5, where we depict the percentage of the budget allocated to repair capacity as function of the relative price of servers (for same cases as in Figure 5.4). The budget as shown in the figure only refers to the investment influenced by our decision model, i.e. the costs for the minimum number of servers corresponding to 100% utilisation are not included. Because we optimise until the availability equals 0.95, the total budget spent increases with the relative server price. The figure shows that the percentage spent to repair shop overcapacity is rather stable for most reasonable relative prices (say ≥ 10). Although repair capacity decreases with the price, the costs of the total capacity increases. Besides, a lot of stock is required to reach 95% availability if the repair shop utilisation becomes high (if server capacity is expensive).

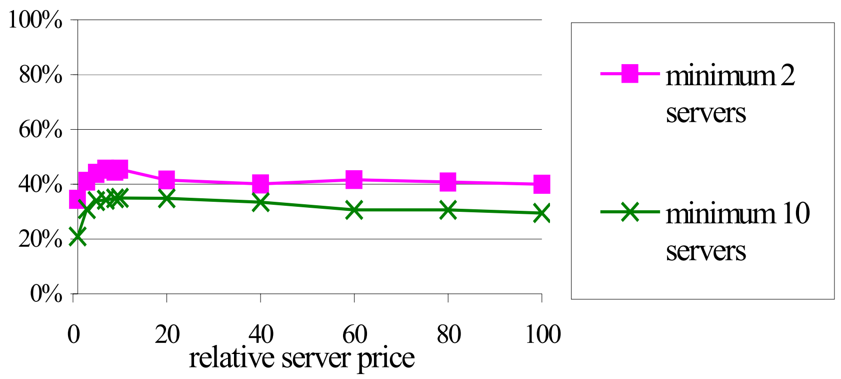

Figure 5.5. Percentage of budget spent to repair capacity as function of the relative price of servers.

Regarding the step size for the number of servers ε, we found that the solution is insensitive to the value of ε. For example, Table 5.4 shows us that the optimal utilisation rate does not depend on size of server steps. Also, we found from detailed results that the difference between the number of servers optimised with a small or a large step size is 0.03 at most. When a small step size is used, the same system availability can be achieved at 0.4% less investment on average. However, it requires much more computational time, as mentioned in section 5.4.1. Also, one might wonder what the practical use is of a detailed figure like 2.34 servers in a repair shop as optimal value, since the exact repair shop capacity will not be determined in such detail. Therefore, we suggest not to use too small server steps during optimisation. From a mathematical point of view, a step size $\varepsilon = 0.01$ is sufficient. However, one may consider larger step sizes as $\varepsilon = 0.1$, because intermediate solution will not be chosen for implementation anyway.

5.5 Conclusions

In this chapter, we analysed the trade-off between inventory levels and number of servers at repair facilities in multi-echelon multi-indenture service part supply systems. We provided a greedy optimisation procedure to find the optimal combination of item stocks and repair capacities. For a valid optimisation procedure, a modification of the VARI-METRIC heuristic appeared to be necessary. Balancing availabilities of the various systems of the installed base during the optimisation proved to be worthwhile. Also, we provided a simple procedure to estimate the characteristics of repair facilities, and therefore system availability, for non-integer number of servers. In this way, we answered research question 4 of Chapter 1.

We found that the optimal utilisation of repair shops usually varies between 0.8 and 0.95. This utilisation rate is mainly influenced by the number of servers and the price of servers relatively to the price of items. Also, we have shown that our algorithm is not very sensitive to the step size in the non-integer servers optimisation: a step size of $\varepsilon = 0.1$ reduces computation time and seems sufficiently accurate for practical purposes.

Chapter 6
Multi-class, multi-server queue with preemptive priorities

Abstract

A multi-class, multi-server queueing system with preemptive priorities is considered. In this system we distinguish two priority classes and each of them consist of multiple item types (subclasses). Each item type has its own Poisson arrival and exponentially distributed service rate. An approximate method to estimate the steady state probabilities is derived with an approximation error that can be made as small as desired at the expense of some more numerical matrix iterations. Based on these probabilities, approximations for a wide range of relevant performance characteristics, such as the expected postponement time for each item class and the first and second moment of the number of items of a certain type in the system, can be derived. The method is illustrated with some numerical examples. Comparison with simulation results shows that with a moderate number of matrix iterations (~20), key performance measures, such as the mean and variance of the number of items in the system, with an error less than 1% in most cases, can be estimated. In this way, we answer research question 5 (Chapter 1), insofar high priority jobs are not outsourced.

This chapter is based on the paper:

“On multi-class multi-server queue with preemptive priority rule.” A. Sleptchenko, A. van Harten, M.C. van der Heijden, submitted for publication.

6.1 Introduction

A repair shop in a spare part network is generally able to handle multiple items. Here, we assume that some items have high priority and the others have low priority. Each item has its own arrival rate and service time distribution. As a consequence, we need to model a repair shop by a (multi-server) priority queueing system with two priority classes, where each class consists of multiple subclasses (item types). An algorithm to determine performance characteristics of such multi-server, multi-class priority queueing systems is not available in the literature as far as we know. Therefore, we develop our own algorithm in this chapter, assuming Poisson arrivals and exponential service times. We expect that such an algorithm can be used for other applications as well, such as computer, communication and production systems. Although we proceed from a model with exponential service times and two priority groups, it is clear that we can in principle extend our method to analyse systems with more priority classes and with hyperexponential service times.

To analyse multi-class, multi-server queues with two priority groups each containing several item classes, we proceed as follows. First, we construct the equilibrium state equations (section 6.2). To solve this set of equations, we develop an approximate approach, which gives estimates of the system states probabilities. From a computational point of view, an exact solution is difficult to achieve, but it can be approximated as close as desired at the expense of moderate numerical work. To demonstrate this, we shall heavily rely upon in-depth insight in the structure of the exact solution. We distinguish three regions when solving the equilibrium equations, namely (1) states with at least one high priority item in the queue, (2) states with only low priority items in the queue, and (3) states in which the queue is empty. We will deal with each region separately. In section 6.3, we will solve the equilibrium equations in region 1 (high priority items in the queue). Next, we will show how to deal with the remaining equations (no high priority items in the queue) in section 6.4. We can solve the latter equations iteratively only, thereby obtaining a cut-off error. Using the (approximate) state probabilities, we can derive various system performance characteristics (expected waiting times per type, expected queue length per type and even correlations between types, expected postponement time per type). In section 6.5, we show as examples how to derive the first two moments of the number of items in the system for each item type and the expected postponement time per item type. We compare our approximations to simulation results in section 6.6. Finally, we present our conclusions and we discuss some model extensions in section 6.7.

6.2 The Model

6.2.1 Definitions and notation

As mentioned before, customers are processed according to a preemptive priority rule, That is, when a high priority item arrives and no server is available, one of the low priority items in service (if there is one) is taken out of service (postponed) to allow the high priority item to be served. We assume a random preemption discipline, i.e. each low priority item in service is chosen with equal probability. When a server comes available again, one of the postponed items is taken back into the service (resumed). We also assume a random resume discipline, i.e. the postponed items are selected with equal probability, independent of their postponement time. Note that other preemption and resume disciplines require larger state spaces and, therefore, more computational efforts. We denote the number of item classes with high (low) priority by N^h (N^l). High priority jobs from subclass i arrive according to a Poisson process with rate λ_i^h and low priority jobs from subclass j arrive with rate λ_j^l. The service times of the subclasses are exponentially distributed with rates μ_i^h and μ_j^l for high and low priority item classes, respectively. Because of the memoryless property of the exponential distribution, it does not make a difference whether postponed jobs are resumed from the moment of interruption on or whether they are restarted completely. All servers are equal, and if multiple servers are available to process a job, each available server has an equal chance to get this job. We use ρ_i^h, ρ_j^l to denote the utilisation rates of high and low priority item classes in the system. We denote the total number of high priority items by n and the number of high priority items of type i by n_i. For low priority items, we will use the notation m and m_i respectively.

We characterise the system state by five vectors of dimensions N^h and N^l, where the components of each vector refer to the (high and low priority) subclasses. The first four vectors are obvious, i.e. vectors containing information about the items in queue and in service:

$\overline{s}^h$ and $\overline{s}^l$ – vectors containing the number of high and low priority items in service per item class.

$\overline{w}^h$ and $\overline{w}^l$ – vectors containing the number of high and low priority items in the queue waiting for first service per item class (these vectors *exclude* postponed items).

Next, we need one more vector to keep track of low priority items that have been withdrawn from service when a high priority item arrived. This vector is necessary, since items with longer processing time will be withdrawn (postponed) more often and then the probability to have in front of the queue one of these items is higher. Thus, the fifth vector is:

$\overline{r}^l$ – vector containing the number of postponed low priority items per item class.

Then the systems state probabilities are denoted by $P_{n,m}\left(\overline{w}^h,\overline{s}^h,\overline{w}^l,\overline{s}^l,\overline{r}^l\right)$.

Other general notations used throughout the chapter are:

$\Lambda^h,\Lambda^l,\mu^h,\mu^l$ – overall arrival and service rates per each priority class, i.e. $\Lambda^h=\sum_{i=1}^{N^h}\lambda_i^h$, $\Lambda^l=\sum_{i=1}^{N^l}\lambda_i^l$ and $\mu^h=\Lambda^h\Big/\sum_{i=1}^{N^h}\frac{\lambda_i^h}{\mu_i^h}=\Lambda^h/k\rho^h$, $\mu^l=\Lambda^l\Big/\sum_{i=1}^{N^l}\frac{\lambda_i^l}{\mu_i^l}=\Lambda^l/k\rho^l$, where the overall utilisation rates for each priority class are $\rho^h=\Lambda^h/k\mu^h$, $\rho^l=\Lambda^l/k\mu^l$ and the total utilisation rate is $\rho=\rho^h+\rho^l$.

a_i^h, a_j^l – fractions of arrival rates, i.e. $a_i^h=\lambda_i^h/\Lambda^h$, $a_j^l=\lambda_j^l/\Lambda^l$.

δ_i^h, δ_j^l – perturbations of service rates, i.e. $\left(1+\delta_i^h\right)=\mu_i^h/\mu^h$ and $\left(1+\delta_j^l\right)=\mu_j^l/\mu^l$.

γ – ratio of the service rates for high and low priority items, i.e. $\gamma=\mu^l/\mu^h$.

$\mu\left(\overline{s}^h,\overline{s}^l\right)$ – sum of service rates of all items in service, i.e. $\mu\left(\overline{s}^h,\overline{s}^l\right)=\sum_{i=1}^{N^h}s_i^h\mu_i^h+\sum_{i=1}^{N^l}s_i^l\mu_i^l$.

$\overline{\delta}\left(\overline{s}^h,\overline{s}^l\right)$ – average perturbation of service rates of all items in service, i.e. $\overline{\delta}\left(\overline{s}^h,\overline{s}^l\right)=\frac{1}{k}\left(\sum_{i=1}^{N^h}s_i^h\delta_i^h+\sum_{i=1}^{N^l}s_i^l\delta_i^l\right)$.

x_i – the ith component of any vector $\overline{x}$.

$\overline{e}_i^h$ ($\overline{e}_i^l$) – a vector of dimension N^h (N^l) with component i equal to 1 and all other components equal to 0; this vector is used to indicate the changes in vectors $\overline{w}^h$ and $\overline{s}^h$ ($\overline{w}^l$, $\overline{s}^l$ and $\overline{r}^l$) during transitions from state to state.

e_{ij}^{h} (e_{ij}^{l}) – denotes the j^{th} component of the vector $\overline{e}_i^{h}$ ($\overline{e}_i^{l}$), so e_{ij}^{h} (e_{ij}^{l}) = 1 if $i = j$ and 0 otherwise.

$|\overline{x}|$ – denotes the sum of all components of any vector $\overline{x}$.

We will introduce the remaining notation later on. The full list of notation can be found in the appendix.

6.2.2 Stationary state equations.

In this section, we will write down the equilibrium state equations for the continuous time Markov chain. That is, the net exchange of probability in an infinitesimal interval from a given state with its neighbours has to be zero in an equilibrium situation. Neighbours of a state $\left(\overline{w}^h, \overline{s}^h, \overline{w}^l, \overline{s}^l, \overline{r}^l\right)$ with n clients of high priority and m clients of low priority are states to/from which a one-step transition is possible, either by an item arrival or by a service completion. According to the numbers n and m of items in the system, the system states (so the equilibrium equations) can be divided into three areas (see Figure 6.1):

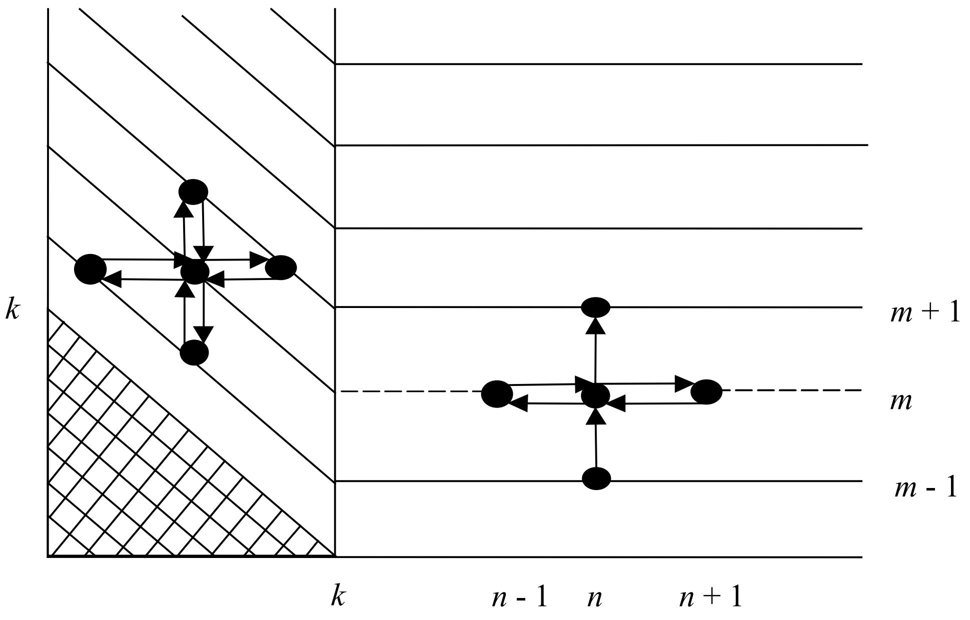

Figure 6.1. Three sets of probability states

I. there is at least one high priority item in the queue ($n > k$, $m \geq 0$),

II. all servers are busy, but there is no high priority item in the queue ($n \leq k$, $n + m \geq k$),

III. there is at least one server available ($n + m < k$).

All these subspaces have different equilibrium equations. Besides, we have to consider the equations for the two boundaries between the regions separately.

In **area I** ($n > k$, $m \geq 0$), we have states with all servers occupied and high priority items in the queue, so no low priority items are in service ($\overline{s}^l = 0$). Therefore, the transitions from the neighbours of state $\left(\overline{w}^h, \overline{s}^h, \overline{w}^l, 0, \overline{r}^l\right)$ are due to:

(1) the arrival of high and low priority jobs that enter the queue;

(2) the service completion of a high priority job; in this case, we consider all combinations (i, j) where i represents the subclass of the item for which service is completed and j is the subclass of the high priority item that enters service.

Hence the equilibrium equations are:

$$\begin{aligned}
&\left(\Lambda^h + \Lambda^l + \overline{\mu}\left(\overline{s}^h, 0\right)\right) P_{n,m}\left(\overline{w}^h, \overline{s}^h, \overline{w}^l, 0, \overline{r}^l\right) \\
&= \sum_{i=1}^{N^h} \lambda_i^h P_{n-1,m}\left(\overline{w}^h - \overline{e}_i^h, \overline{s}^h, \overline{w}^l, 0, \overline{r}^l\right) + \sum_{i=1}^{N^l} \lambda_i^l P_{n,m-1}\left(\overline{w}^h, \overline{s}^h, \overline{w}^l - \overline{e}_i^l, 0, \overline{r}^l\right) \\
&+ \sum_{i=1}^{N^h}\sum_{j=1}^{N^h} \frac{w_j^h + 1}{\left|\overline{w}^h\right| + 1}\left(s_i^h + 1 - e_{ij}^h\right)\mu_i^h P_{n+1,m}\left(\overline{w}^h + \overline{e}_j^h, \overline{s}^h + \overline{e}_i^h - \overline{e}_j^h, \overline{w}^l, 0, \overline{r}^l\right)
\end{aligned} \tag{6.1}$$

where $\frac{w_j^h + 1}{\left|\overline{w}^h\right| + 1}$ characterises the probability that the high priority item with class j is in front of the queue.

In **area II** ($n \leq k$, $n + m \geq k$), the equilibrium equations are different for the internal states (i.e. $n < k$, $n + m > k$) and for the boundary states (i.e. $n = k$, $n + m > k$ for **II-III** and $n + m = k$ for **I-II**).

In the **internal states of area II** ($n < k$, $n + m > k$), all servers are busy, low priority items are in service and no high priority items are in the queue (i.e. $\overline{s}^l \neq 0$, $\overline{w}^h = 0$). So, the transitions from the neighbours of state $\left(0, \overline{s}^h, \overline{w}^l, \overline{s}^l, \overline{r}^l\right)$ occur due to:

(1) the arrival of a low priority job that enters the queue,

(2) the arrival of a high priority job that is served directly, thereby preempting a low priority job and changing the vector $\overline{r}^l$; as we assume that low priority items to be withdrawn from service are selected randomly, the probability that an item of subclass j is selected equals $\frac{s_j^l+1}{\left|\overline{s}^l\right|+1}$,

(3) service completion of a (high or low priority) item type i without postponed items in the queue ($\overline{r}^l = 0$), so a new service of a low priority item type j is started; note that subclass j is at the front of the queue with probability $\frac{w_j^h+1}{\left|\overline{w}^h\right|+1}$,

(4) service completion of a (high or low priority) item type i with postponed items in the queue ($\overline{r}^l \neq 0$), so the service of a postponed low priority item type j is continued; note that subclass j is at the front of the queue with probability $\frac{r_j^l+1}{\left|\overline{r}^l\right|+1}$.

When writing down the equilibrium equations, we use the fact that the service rates of the postponed items and the items in the queue are the same due to the memoryless property of exponential distribution of the service times. Then, the equilibrium equations are:

$$
\begin{aligned}
&\left(\Lambda^h + \Lambda^l + \overline{\mu}\left(\overline{s}^h, \overline{s}^l\right)\right) P_{n,m}\left(0, \overline{s}^h, \overline{w}^l, \overline{s}^l, \overline{r}^l\right) = \sum_{i=1}^{N^l} \lambda_i^l P_{n,m-1}\left(0, \overline{s}^h, \overline{w}^l - \overline{e}_i^l, \overline{s}^l, \overline{r}^l\right) \\
&+\sum_{i=1}^{N^h}\sum_{j=1}^{N^l} \lambda_i^h \frac{s_j^l+1}{\left|\overline{s}^l\right|+1} P_{n-1,m}\left(0, \overline{s}^h - \overline{e}_i^h, \overline{w}^l, \overline{s}^l + \overline{e}_j^l, \overline{r}^l - \overline{e}_j^l\right) \\
&+I\left(\left|\overline{r}^l\right| = 0\right)\sum_{i=1}^{N^h}\sum_{j=1}^{N^l} \frac{w_j^l+1}{\left|\overline{w}^l\right|+1}\left(s_i^h+1\right)\mu_i^h P_{n+1,m}\left(0, \overline{s}^h + \overline{e}_i^h, \overline{w}^l + \overline{e}_j^l, \overline{s}^l - \overline{e}_j^l, \overline{r}^l\right) \\
&+I\left(\left|\overline{r}^l\right| = 0\right)\sum_{i=1}^{N^l}\sum_{j=1}^{N^l} \frac{w_j^l+1}{\left|\overline{w}^l\right|+1}\left(s_i^l+1-e_{ij}^l\right)\mu_i^l P_{n,m+1}\left(0, \overline{s}^h, \overline{w}^l + \overline{e}_j^l, \overline{s}^l + \overline{e}_i^l - \overline{e}_j^l, \overline{r}^l\right) \\
&+\sum_{i=1}^{N^h}\sum_{j=1}^{N^l} \frac{r_j^l+1}{\left|\overline{r}^l+1\right|}\left(s_i^h+1\right)\mu_i^h P_{n+1,m}\left(0, \overline{s}^h + \overline{e}_i^h, \overline{w}^l, \overline{s}^l - \overline{e}_j^l, \overline{r}^l + \overline{e}_j^l\right) \\
&+\sum_{i=1}^{N^l}\sum_{j=1}^{N^l} \frac{r_j^l+1}{\left|\overline{r}^l+1\right|}\left(s_i^l+1-e_{ij}^l\right)\mu_i^l P_{n,m+1}\left(0, \overline{s}^h, \overline{w}^l, \overline{s}^l + \overline{e}_i^l - \overline{e}_j^l, \overline{r}^l + \overline{e}_j^l\right)
\end{aligned}
\tag{6.2}
$$

where $I(\cdot)$ denotes the indicator function, so the value of the function is 1 if the statement between parentheses is true and 0 otherwise.

Then, we consider the **border between the areas I and II** (*n* = *k*). That is, the number of high priority jobs in the system equals the number of servers. So, no high priority jobs are waiting in the queue and no low priority jobs are in service. Then, the transitions to the state $\left(0,\overline{s}^h,\overline{w}^l,0,\overline{r}^l\right)$ are:

(1) the arrival of a high priority job *i* that enters service immediately, thereby preempting the single low priority job *j* that was in service,

(2) the arrival of a low priority job that enters the queue,

(3) the service completion of a high priority job *i*, causing that the single high priority job in the queue (type *j*) is being served; note that a low priority job cannot be completed, since all servers are busy with high priority jobs.

So the equilibrium equations are:

$$\begin{aligned}&\left(\Lambda^h+\Lambda^l+\overline{\mu}\left(\overline{s}^h,0\right)\right)P_{n,m}\left(0,\overline{s}^h,\overline{w}^l,0,\overline{r}^l\right)\\&=\sum_{i=1}^{N^h}\sum_{j=1}^{N^l}\lambda_i^hP_{n-1,m}\left(0,\overline{s}^h-\overline{e}_i^h,\overline{w}^l,\overline{e}_j^l,\overline{r}^l-\overline{e}_j^l\right)+\sum_{i=1}^{N^l}\lambda_i^lP_{n,m-1}\left(0,\overline{s}^h,\overline{w}^l-\overline{e}_i^l,0,\overline{r}^l\right)\\&+\sum_{i=1}^{N^h}\sum_{j=1}^{N^h}\left(s_i^h+1-e_{ij}^h\right)\mu_i^hP_{n+1,m}\left(\overline{e}_j^h,\overline{s}^h+\overline{e}_i^h-\overline{e}_j^h,\overline{w}^l,0,\overline{r}^l\right)\end{aligned}\tag{6.3}$$

Next, we consider the **border between the areas II and III** (*n* + *m* = *k*, *n* < *k*). That is, all servers are occupied (with high and/or low priority jobs), the queue is empty and there is at least one low priority item in service. Then, the transitions to the state $\left(0,\overline{s}^h,0,\overline{s}^l,0\right)$ are:

(1) the arrival of a (high or low priority) job *i* that enters service immediately,

(2) the service completion of a high priority job *i*, causing that a single low priority job in the queue (type *j*) is being served; note that a low priority job that is taken into service may be either a new service or a postponed service,

(3) the service completion of a low priority job *i*, causing that the single low priority job in the queue (type *j*) is being served; again, a low priority job that is taken into service may be either a new or a postponed service; note that a high priority job cannot be in the queue if a low priority job is being served.

$$\begin{aligned}
&\left(\Lambda^h + \Lambda^l + \bar{\mu}\left(\bar{s}^h, \bar{s}^l\right)\right) P_{n,m}\left(0, \bar{s}^h, 0, \bar{s}^l, 0\right) = \\
&\sum_{i=1}^{N^h} \lambda_i^h P_{n-1,m}\left(0, \bar{s}^h - \bar{e}_i^h, 0, \bar{s}^l, 0\right) + \sum_{i=1}^{N^l} \lambda_i^l P_{n,m-1}\left(0, \bar{s}^h, 0, \bar{s}^l - \bar{e}_i^l, 0\right) \\
&+ I\left(\left|\bar{r}^l\right| = 0\right) \sum_{i=1}^{N^h} \sum_{j=1}^{N^l} \left(s_i^h + 1\right) \mu_i^h P_{n+1,m}\left(0, \bar{s}^h + \bar{e}_i^h, \bar{e}_j^l, \bar{s}^l - \bar{e}_j^l, 0\right) \\
&+ I\left(\left|\bar{r}^l\right| = 0\right) \sum_{i=1}^{N^l} \sum_{j=1}^{N^l} \left(s_i^l + 1 - e_{ij}^l\right) \mu_i^l P_{n,m+1}\left(0, \bar{s}^h, \bar{e}_j^l, \bar{s}^l + \bar{e}_i^l - \bar{e}_j^l, 0\right) \\
&+ \sum_{i=1}^{N^h} \sum_{j=1}^{N^l} \left(s_i^h + 1\right) \mu_i^h P_{n+1,m}\left(0, \bar{s}^h + \bar{e}_i^h, 0, \bar{s}^l - \bar{e}_j^l, \bar{e}_j^l\right) \\
&+ \sum_{i=1}^{N^l} \sum_{j=1}^{N^l} \left(s_i^l + 1 - e_{ij}^l\right) \mu_i^l P_{n,m+1}\left(0, \bar{s}^h, 0, \bar{s}^l + \bar{e}_i^l - \bar{e}_j^l, \bar{e}_j^l\right)
\end{aligned} \qquad (6.4)$$

Finally, we consider probability states in **area III** (*n* + *m* < *k*). These states have an empty queue and hence no items are postponed ($\bar{w}^h = 0$, $\bar{w}^l = 0$ and $\bar{r}^l = 0$). Now the transitions from the neighbours of the state $\left(0, \bar{s}^h, 0, \bar{s}^l, 0\right)$ become much simpler. They consist of:

(1) the arrival of a high priority job that enters service immediately,
(2) the arrival of a low priority job that enters service immediately,
(3) the completion of a (high or low) priority job without starting a new job because the queue is empty.

Hence we find:

$$\begin{aligned}
&\left(\Lambda^h + \Lambda^l + \bar{\mu}\left(\bar{s}^h, \bar{s}^l\right)\right) P_{n,m}\left(0, \bar{s}^h, 0, \bar{s}^l, 0\right) \\
&= \sum_{i=1}^{N^h} \lambda_i^h P_{n-1,m}\left(0, \bar{s}^h - \bar{e}_i^h, 0, \bar{s}^l, 0\right) + \sum_{i=1}^{N^l} \lambda_i^l P_{n,m-1}\left(0, \bar{s}^h, 0, \bar{s}^l - \bar{e}_i^l, 0\right) \\
&+ \sum_{i=1}^{N^h} \left(s_i^h + 1\right) \mu_i^h P_{n+1,m}\left(0, \bar{s}^h + \bar{e}_i^h, 0, \bar{s}^l, 0\right) + \sum_{i=1}^{N^l} \left(s_i^l + 1\right) \mu_i^l P_{n,m+1}\left(0, \bar{s}^h, 0, \bar{s}^l + \bar{e}_i^l, 0\right)
\end{aligned} \qquad (6.5)$$

In the next sections, we will show how we can solve these equilibrium equations, thereby obtaining the exact system state probabilities. We will address area I in section 6.3 and the areas II and III in section 6.4.

6.3 System states with high priority items in queue

In this section, we focus on area I, so there is at least one high priority item in the queue.

6.3.1 Reducing the set of equations

First, we note that we can rewrite these state probabilities by conditioning on the total number of high priority items in the queue. So, $P_{n,m}\left(\overline{w}^h,\overline{s}^h,\overline{w}^l,0,\overline{r}^l\right) = P'_{n,m}\left(\left|\overline{w}^h\right|,\overline{s}^h,\overline{w}^l,\overline{s}^l,\overline{r}^l\right) \times$ Pr{distribution of high priority items in queue over subclasses = $\overline{w}^h$ | number of high priority items in queue = $\left|\overline{w}^h\right|$}. Given the total number of high priority items in the queue, the items are distributed over the item subclasses according to a multinomial distribution, $\left|\overline{w}^h\right|!\prod_{i=1}^{N^h}\frac{\left(a_i^h\right)^{w_i^h}}{w_i^h!}$. The expression $P'_{n,m}\left(\left|\overline{w}^h\right|,\overline{s}^h,\overline{w}^l,\overline{s}^l,\overline{r}^l\right)$ describes the probability that (1) the distribution of high priority items in service is given by the vector $\overline{s}^h$, (2) the low priority items in queue are given by the vectors $\overline{w}^l$ and $\overline{r}^l$, (3) no low priority items are in service, and (4) the total number of high priority items in the queue equals $\left|\overline{w}^h\right|$. Because it holds that $\left|\overline{w}^h\right| = n-k$, we see that this expression is in fact independent of $\left|\overline{w}^h\right|$. Therefore, we can omit the parameter $\left|\overline{w}^h\right|$ and rewrite the equilibrium equations for the states with $n > k$ using the product form:

$$P_{n,m}\left(\overline{w}^h,\overline{s}^h,\overline{w}^l,0,\overline{r}^l\right) = \left|\overline{w}^h\right|!\prod_{i=1}^{N^h}\frac{\left(a_i^h\right)^{w_i^h}}{w_i^h!}P'_{n,m}\left(\overline{s}^h,\overline{w}^l,0,\overline{r}^l\right). \tag{6.6}$$

In this way, we reduce the number of state probabilities to be solved.

We substitute the relation (6.6) in equation (6.1) and we divide both sides by $k\mu^h$ to obtain equations that can be transformed in matrix form. This yields:

$$\left(1+\rho^h+\gamma\rho^l+\overline{\delta}\left(\overline{s}^h,0\right)\right)\left|\overline{w}^h\right|!\prod_{i=1}^{N^h}\frac{\left(a_i^{\,h}\right)^{w_i^{\,h}}}{w_i^{\,h}!}P'_{n,m}\left(\overline{s}^h,\overline{w}^l,0,\overline{r}^l\right)$$
$$=\rho^h\sum_{i=1}^{N^h}a_i^{\,h}\left(\left|\overline{w}^h\right|-1\right)!\frac{w_i^{\,h}}{a_i^{\,h}}\prod_{j=1}^{N^h}\frac{\left(a_j^{\,h}\right)^{w_j^{\,h}}}{w_j^{\,h}!}P'_{n-1,m}\left(\overline{s}^h,\overline{w}^l,0,\overline{r}^l\right)$$
$$+\gamma\rho^l\sum_{i=1}^{N^l}a_i^{\,l}\left|\overline{w}^h\right|!\prod_{i=1}^{N^h}\frac{\left(a_i^{\,h}\right)^{w_i^{\,h}}}{w_i^{\,h}!}P'_{n,m-1}\left(\overline{s}^h,\overline{w}^l-\overline{e}_i^{\,l},0,\overline{r}^l\right)$$
$$+\frac{1}{k}\sum_{i=1}^{N^h}\sum_{j=1}^{N^h}\left(s_i^{\,h}+1-e_{ij}^{\,h}\right)\mu_i^{\,h}\frac{w_j^{\,h}+1}{\left|\overline{w}^h\right|+1}\frac{a_j^{\,h}\left(\left|\overline{w}^h\right|+1\right)}{w_j^{\,h}+1}\prod_{q=1}^{N^h}\frac{\left(a_q^{\,h}\right)^{w_q^{\,h}}}{w_q^{\,h}!}P'_{n+1,m}\left(\overline{s}^h+\overline{e}_i^{\,h}-\overline{e}_j^{\,h},\overline{w}^l,0,\overline{r}^l\right)$$

where $\gamma=\frac{\mu^l}{\mu^h}$.

Because we use the product form (6.6), we have to sum up all the equations for the state probabilities satisfying $\sum w_i^{\,h}=\left|\overline{w}^h\right|$. Taking into account that $\sum a_i^{\,h}=1$, we can write all equilibrium equations with the same *n* and *m* in a matrix form:

$$\left(\left(1+\rho^h+\gamma\rho^l\right)\mathbf{I}+\overline{\boldsymbol{\delta}}^h\right)\mathbf{P}_{n,m}\left(\overline{w}^l,0\right)=\rho^h\mathbf{P}_{n-1,m}\left(\overline{w}^l,0\right)$$
$$+\gamma\rho^l\sum_{i=1}^{N^l}a_i^{\,l}\mathbf{P}_{n,m-1}\left(\overline{w}^l-\overline{e}_i^{\,l},0\right)+\mathbf{A}\mathbf{P}_{n+1,m}\left(\overline{w}^l,0\right) \tag{6.7}$$

where $\mathbf{P}_{n,m}\left(\overline{w}^l,0\right)$ are vectors containing probabilities $P'_{n,m}\left(\overline{s}^h,\overline{w}^l,0,\overline{r}^l\right)$ as components. The dimension of vectors $\mathbf{P}_{n,m}\left(\overline{w}^l,0\right)$ is equal to amount of different server states, given that all servers are occupied with high priority items. That is, this dimension is equal to $d\left(N^h,k\right)\left(\sum_{i=1}^k d\left(N^l,i\right)\right)$, with $d\left(x,y\right)=\binom{x+y+1}{y}$. $\overline{\boldsymbol{\delta}}^h$ and $\mathbf{A}$ are linear operators on a $d\left(N^h,k\right)\left(\sum_{i=1}^k d\left(N^l,i\right)\right)$ – dimensional linear space:

$$\overline{\boldsymbol{\delta}}^h\mathbf{P}_{n,m}\left(\overline{w}^l,0\right)\left[\overline{s}^h,\overline{r}^l\right]=\left(1+\rho^h+\gamma\rho^l+\overline{\delta}\left(\overline{s}^h,0\right)\right)\mathbf{P}_{n,m}\left(\overline{w}^l,0\right)\left[\overline{s}^h,\overline{r}^l\right]$$
$$\mathbf{A}\mathbf{P}_{n+1,m}\left(\overline{w}^l,0\right)\left[\overline{s}^h,\overline{r}^l\right]=\frac{1}{k}\sum_{i=1}^{N^h}\sum_{j=1}^{N^h}\left(s_i^{\,h}+1-e_{ij}^{\,h}\right)\left(1+\delta_i^{\,h}\right)a_j^{\,h}\mathbf{P}_{n+1,m}\left(\overline{w}^l,0\right)\left[\overline{s}^h+e_i^{\,h}-e_j^{\,h},\overline{r}^l\right]$$

Solving this matrix equation, we can find all state probabilities with ($n > k$). To solve it, it is worth noticing that $\overline{r}^l$ only serves as an index, where equations with different indices $\overline{r}^l$ are decoupled. In the next lemma, we explain the structure of the solution of this equation.

Lemma 6.1

Define the matrix-function $\mathbf{Z}\left(\rho^h,\rho^l,\gamma;\xi\right)$ *as the solution of*

$$\left(\left(1+\rho^h+\gamma\rho^l\right)\mathbf{I}+\overline{\boldsymbol{\delta}}^h\right)\rho^h\mathbf{Z}+\gamma\rho^l\xi+\mathbf{A}\mathbf{Z}^{-1}, \qquad \left|\sigma\left(\mathbf{Z}\right)\right|>1 \tag{6.8}$$

Then

$$\begin{aligned} &P_{n,m}\left(\overline{w}^h,\overline{s}^h,\overline{w}^l,0,\overline{r}^l\right)\\ &=\left|\overline{w}^h\right|!\left(\prod_{i=1}^{N^h}\frac{\left(a_i^h\right)^{w_i^h}}{w_i^h!}\right)\left(\prod_{i=1}^{N^l}\frac{\left(a_i^l\right)^{w_i^l}}{w_i^l!}\right)\left(\frac{d}{d\xi}\right)^{\left|\overline{w}^l\right|}\left[\left(\mathbf{Z}^{-1}(\xi)\right)^{n-k}\mathbf{C}(\xi)\right]_{\xi=0}\left[\overline{s}^h,\overline{r}^l\right] \end{aligned} \tag{6.9}$$

satisfies all equations for $m \geq 0$, $n > k$.

Note that $\sum_{\overline{w}^h,\overline{w}^l}P\left(\overline{w}^h,\overline{s}^h,\overline{w}^l,0,\overline{r}^l\right)=\left[\left(\mathbf{Z}^{-1}(\xi)\right)^{n-k}\mathbf{C}(\xi)\right]_{\xi=1}\left[\overline{s}^h,\overline{r}^l\right]$. *The notation* $\left[\overline{s}^h,\overline{r}^l\right]$ *in the right hand side refers to the indicated vector component.*

Proof.

We will prove this lemma by induction using the matrix form (6.7) of the equilibrium equations (6.1).

For $m = 0$, equation (6.7) can be written as

$$\left(\left(1+\rho^h+\gamma\rho^l\right)\mathbf{I}+\overline{\boldsymbol{\delta}}^h\right)\mathbf{P}_{n,0}\left(0,0\right)=\rho^h\mathbf{P}_{n-1,0}\left(0,0\right)+\mathbf{A}\mathbf{P}_{n+1,0}\left(0,0\right)$$

which is similar to the multi-class multi-server equilibrium equation having a solution of the form $\mathbf{P}_{n,0}\left(0,0\right)=\left(\mathbf{Z}^{-1}\right)^{n-k}\mathbf{C}$ (see Chapter 3), where $\mathbf{Z}$ should satisfy the equation

$$\left(\left(1+\rho^h+\gamma\rho^l\right)\mathbf{I}+\overline{\boldsymbol{\delta}}^h\right)\rho^h\mathbf{Z}+\mathbf{A}\mathbf{Z}^{-1}, \qquad \left|\sigma\left(\mathbf{Z}\right)\right|>1$$

similar to the equation (6.8) with $\xi = 0$. So, we have that the solution in the form (6.9) is the solution of the equation (6.1) for $m = 0$.

For $m > 0$, we first define $\mathbf{P}'_n\left(\overline{x}\right)$ as the solution of:

$$\left(\left(1+\rho^h+\gamma\rho^l\right)\mathbf{I}+\overline{\boldsymbol{\delta}}^h\right)\mathbf{P}'_n\left(\overline{x}\right)=\rho^h\mathbf{P}'_{n-1}\left(\overline{x}\right)+\gamma\rho^l\sum_{j=1}^{N^l}a_j{}^l x_j\mathbf{P}'_n\left(\overline{x}\right)+\mathbf{A}\mathbf{P}'_{n+1}\left(\overline{x}\right) \qquad (6.10)$$

with $x_j\in\left[0,1\right]$, being just a parameter.

It follows that $\mathbf{P}'_n\left(\overline{x}\right)=\left(\mathbf{Z}\left(\rho^h,\rho^l,\gamma;\overline{x}\right)^{-1}\right)^{n-k}\mathbf{C}'\left(\overline{x}\right)$. By differentiation of (6.10) with respect to x_j, we find

$$\left(\left(1+\rho^h+\gamma\rho^l\right)\mathbf{I}+\overline{\boldsymbol{\delta}}^h\right)\frac{d\mathbf{P}'_n\left(\overline{x}\right)}{dx_j}=\rho^h\frac{d\mathbf{P}'_{n-1}\left(\overline{x}\right)}{dx_j}+\gamma\rho^l\sum_{i=1}^{N^l}a_i{}^l x_i\frac{d\mathbf{P}'_n\left(\overline{x}\right)}{dx_j}+\mathbf{A}\frac{d\mathbf{P}'_{n+1}\left(\overline{x}\right)}{dx_j}$$

Hence $\left[\frac{d\mathbf{P}'_n\left(\overline{x}\right)}{dx_j}\right]_{\overline{x}=0}$ satisfies the equation (6.1) for $m = 1$.

Using the general property $\left(\frac{d}{dx}\right)^m\left(xf\left(x\right)\right)=m\left(\frac{d}{dx}\right)^{m-1}f\left(x\right)+x\left(\frac{d}{dx}\right)^m f\left(x\right)$, we find by differentiation of (6.10) m_j times with respect to each parameter x_j that:

$$\begin{aligned}&\left(\left(1+\rho^h+\gamma\rho^l\right)\mathbf{I}+\overline{\boldsymbol{\delta}}^h\right)\prod_{i=1}^{N^l}\left(\frac{d}{dx_i}\right)^{m_i}\mathbf{P}'_n\left(\overline{x}\right)=\rho^h\prod_{i=1}^{N^l}\left(\frac{d}{dx_i}\right)^{m_i}\mathbf{P}'_{n-1}\left(\overline{x}\right)\\&+\gamma\rho^l\sum_{j=1}^{N^l}a_j{}^l x_j\prod_{i=1}^{N^l}\left(\frac{d}{dx_i}\right)^{m_i}\mathbf{P}'_n\left(\overline{x}\right)+\mathbf{A}\prod_{i=1}^{N^l}\left(\frac{d}{dx_i}\right)^{m_i}\mathbf{P}'_{n+1}\left(\overline{x}\right)\\&+\gamma\rho^l\sum_{j=1}^{N^l}m_j a_j{}^l\left(\frac{d}{dx_j}\right)^{m_j-1}\prod_{\substack{i=1\\j\neq i}}^{N^l}\left(\frac{d}{dx_i}\right)^{m_i}\mathbf{P}'_n\left(\overline{x}\right)\end{aligned}$$

Transforming this expression back to the $\mathbf{P}_{n,m}\left(\overline{w},0\right)$:

$$\mathbf{P}_{n,m}\left(\overline{w}^l,0\right)=\prod_{i=1}^{N^l}\frac{1}{m_i!}\left(\frac{d}{dx_i}\right)^{m_i}\left[\left(\mathbf{Z}^{-1}\left(\overline{x}\right)\right)^{n-k}\mathbf{C}\left(\overline{x}\right)\right]_{\overline{x}=0}$$

Now using the fact that $\frac{d}{dx_j}=\frac{d\xi(\overline{x})}{dx_j}\frac{d}{d\xi}=a_j{}^l\frac{d}{d\xi}$, we can change the term $\sum_{j=1}^{N^l}a_j{}^l x_j$ in the equation (6.10) to the function $\xi\left(\overline{x}\right)=\sum_{j=1}^{N^l}a_j{}^l x_j$ and we can rewrite the probability states as:

$$\mathbf{P}_{n,m}\left(\overline{w}^l,0\right)=\prod_{i=1}^{N^l}\frac{\left(a_i{}^l\right)^{m_i}}{m_i!}\left(\frac{d}{d\xi}\right)^{m}\left[\left(\mathbf{Z}^{-1}\left(\xi\right)\right)^{n-k}\mathbf{C}\left(\xi\right)\right]_{\xi=0}$$

Finally, we recall that $\sum_{m_j,\text{ s.t. }\sum m_j=m}m!\prod\frac{\left(a_j{}^l\right)^{m_j}}{m_j!}=\left(\sum a_j\right)^m$ and we obtain

$$\sum_m\frac{m!}{m!}\prod_{i=1}^{N^l}\frac{\left(a_i{}^l\right)^{m_i}}{m_i!}\left(\frac{d}{d\xi}\right)^{m}\left[\left(\mathbf{Z}^{-1}\left(\xi\right)\right)^{n-k}\mathbf{C}\left(\xi\right)\right]_{\xi=0}$$
$$=\sum_m\frac{1}{m!}\underbrace{\left(\sum a_j\right)^m}_{=1}\left(\frac{d}{d\xi}\right)^{m}\left[\left(\mathbf{Z}^{-1}\left(\xi\right)\right)^{n-k}\mathbf{C}\left(\xi\right)\right]_{\xi=0}=\left[\left(\mathbf{Z}^{-1}\left(\xi\right)\right)^{n-k}\mathbf{C}\left(\xi\right)\right]_{\xi=1}$$

as a well-known Taylor series expansion of $\left(\mathbf{Z}\left(\rho^h,\rho^l,\gamma;\xi\right)^{-1}\right)^{n-k}$ around ξ = 1, where the value of $\mathbf{Z}\left(\rho^h,\rho^l,\gamma;\xi\right)$ is found from

$$\left(\left(1+\rho^h+\gamma\rho^l\right)\mathbf{I}+\boldsymbol{\overline{\delta}}^h\right)\rho^h\mathbf{Z}+\gamma\rho^l\xi+\mathbf{A}\mathbf{Z}^{-1},\qquad\left|\sigma\left(\mathbf{Z}\right)\right|>1$$

This equation can be solved as in the case of the non-priority multi-class queue (Chapter 3) using decoupling by $\overline{r}^l$.

So the solution has the form:

$$P_{n,m}\left(\overline{w}^h,\overline{s}^h,\overline{w}^l,0,\overline{r}^l\right)=\left|\overline{w}^h\right|!\left(\prod_{i=1}^{N^h}\frac{\left(a_i^h\right)^{w_i^h}}{w_i^h!}\right)\left(\prod_{i=1}^{N^l}\frac{\left(a_i^l\right)^{w_i^l}}{w_i^l!}\right)\left(\frac{d}{d\xi}\right)^{\left|\overline{w}^l\right|}\left[\left(\mathbf{Z}^{-1}(\xi)\right)^{n-k}\mathbf{C}(\xi)\right]_{\xi=0}\left[\overline{s}^h,\overline{r}^l\right]$$

■

The probabilities of the system states constructed in this section have a differential form, therefore we will need derivatives of the matrix **Z**. To find these derivatives is not an easy task, since we can not derive an analytical form of the matrix **Z**, but we can use the equation (6.8) to find such derivatives iteratively.

6.3.2 On the derivatives of the matrix Z

Unlike a scalar function, finding the derivative $\frac{d}{d\xi}\left[\mathbf{Z}(\xi)^{-1}\right]$ has to take into account the non-commutativity of $\frac{d}{d\xi}\mathbf{Z}(\xi)$ and $\mathbf{Z}(\xi)$. This can be in the following way:

$$\frac{d}{d\xi}:\mathbf{I}=\mathbf{Z}(\xi)\mathbf{Z}(\xi)^{-1}\quad\Rightarrow\quad 0=\frac{d}{d\xi}\mathbf{Z}(\xi)\mathbf{Z}(\xi)^{-1}+\mathbf{Z}(\xi)\frac{d}{d\xi}\left[\mathbf{Z}(\xi)^{-1}\right].$$

This provides a first relation between $\frac{d}{d\xi}\left[\mathbf{Z}(\xi)^{-1}\right]$ and $\frac{d}{d\xi}\mathbf{Z}(\xi)$. Next, we can use the equation defining the matrix $\mathbf{Z}(\xi)$, i.e. if we differentiate (6.8) with respect to ξ, we get

$$0=\rho^h\frac{d}{d\xi}\mathbf{Z}(\xi)+\gamma\rho^l+\mathbf{A}\frac{d}{d\xi}\left[\mathbf{Z}(\xi)^{-1}\right]$$

So, we have a $2\binom{N^h+k-1}{k}\left(\sum_{i=0}^{k}\binom{N^l+i-1}{i}\right)$ – dimensional non-singular system of linear equations for $2\binom{N^h+k-1}{k}\left(\sum_{i=0}^{k}\binom{N^l+i-1}{i}\right)$ variables and, therefore, the derivatives $\frac{d}{d\xi}\left[\mathbf{Z}(\xi)^{-1}\right]$ and $\frac{d}{d\xi}\mathbf{Z}(\xi)$ for each value ξ are defined.

We can find the higher order derivatives in a similar way. We have a $2\binom{N^h+k-1}{k}\left(\sum_{i=0}^{k}\binom{N^l+i-1}{i}\right)$ – dimensional non-singular system of linear equations for $2\binom{N^h+k-1}{k}\left(\sum_{i=0}^{k}\binom{N^l+i-1}{i}\right)$ variables:

$$\frac{d^m}{d\xi^m}\mathbf{Z}(\xi)\mathbf{Z}(\xi)^{-1}+\mathbf{Z}(\xi)\frac{d^m}{d\xi^m}\left[\mathbf{Z}(\xi)^{-1}\right]=-\sum_{i=1}^{m-1}\frac{m!}{i!(m-i)!}\frac{d^i}{d\xi^i}\mathbf{Z}(\xi)\frac{d^{m-i}}{d\xi^{m-i}}\left[\mathbf{Z}(\xi)^{-1}\right]$$
$$\rho^h\frac{d^m}{d\xi^m}\mathbf{Z}(\xi)+\mathbf{A}\frac{d^m}{d\xi^m}\left[\mathbf{Z}(\xi)^{-1}\right]=0$$

Therefore, the derivatives $\frac{d^m}{d\xi^m}\left[\mathbf{Z}(\xi)^{-1}\right]$ and $\frac{d^m}{d\xi^m}\mathbf{Z}(\xi)$ for each value ξ can be obtained from the lower order derivatives.

Note that the equations for derivatives $\frac{d^m}{d\xi^m}\left[\mathbf{Z}(\xi)^{-1}\right]$ and $\frac{d^m}{d\xi^m}\mathbf{Z}(\xi)$ have a block-diagonal structure, i.e. they consist of $\sum_{i=0}^{k}\binom{N^l+i-1}{i}$ blocks for different $\overline{r}^l$ of dimension $2\binom{N^h+k-1}{k}\times 2\binom{N^h+k-1}{k}$ each. Moreover, these blocks have the same structure for different $\overline{r}^l$. Hence, to construct the derivatives $\frac{d^m}{d\xi^m}\left[\mathbf{Z}(\xi)^{-1}\right]$ and $\frac{d^m}{d\xi^m}\mathbf{Z}(\xi)$ for each m (given that derivatives of lower order are already known), it is enough to solve once a system of linear equation of dimension $2\binom{N^h+k-1}{k}\times 2\binom{N^h+k-1}{k}$ for one of the components of $\overline{r}^l$.

6.4 System states with no high priority items in queue ($n \le k$)

In this section, we describe the solution of the equilibrium equations for the states with only low priority items in the queue ($n \le k$).

6.4.1 Reducing the set of equations

As in the previous section, we can reduce the set of equations by writing the state probabilities in product form, conditioning on the total number of low priority items in the queue $\left|\overline{w}^l\right|$. Given this total number, the number of jobs in the queue per low priority subclass has a multinomial distribution with parameters $\left|\overline{w}^l\right|$ and $a_i^l=\lambda_i^l/\Lambda^l$, $i=1,\ldots,N^l$. So, we can write:

$$P_{n,m}\left(0,\overline{s}^h,\overline{w}^l,\overline{s}^l,\overline{r}^l\right)=\left|\overline{w}^l\right|!\prod_{i=1}^{N^l}\frac{\left(a_i^l\right)^{w_i^l}}{w_i^l!}P''_{n,m}\left(\overline{s}^h,\overline{s}^l,\overline{r}^l\right).$$

Note that we can omit two parameters, namely the number of high priority items in the queue per subclass (these are always zero) and the number of low priority items in the queue $\left|\overline{w}^l\right|$. For the latter, it holds that $\left|\overline{w}^l\right| = m - \left|\overline{s}^l\right| - \left|\overline{r}^l\right|$.

As before, we substitute this product form into the equilibrium equations (6.2) for $n < k$, $m > k - n$, and next we sum up the equations for all system states having the total number of (non-postponed) low priority items in the queue equal to $\left|\overline{w}^l\right|$. After dividing the resulting equations by $k\mu^h$, we obtain the following equations for $P''_{n,m}\left(\cdot,\cdot,\cdot\right)$:

$$
\begin{aligned}
&\left(1+\rho^h+\gamma\rho^l+\overline{\delta}\left(\overline{s}^h,\overline{s}^l\right)\right)\left(\sum_{i=1}^{N^l} a_i^l\right)^{\left|\overline{w}^l\right|} P''_{n,m}\left(\overline{s}^h,\overline{s}^l,\overline{r}^l\right) = \left(\sum_{i=1}^{N^l} a_i^l\right)^{\left|\overline{w}^l\right|} \gamma\rho^l P''_{n,m}\left(\overline{s}^h,\overline{s}^l,\overline{r}^l\right) \\
&+\rho^h\left(\sum_{i=1}^{N^l} a_i^l\right)^{\left|\overline{w}^l\right|} \sum_{i=1}^{N^h} a_i^h \sum_{j=1}^{N^l} \frac{s_j^l+1}{\left|\overline{s}^l\right|+1} P''_{n-1,m}\left(\overline{s}^h-\overline{e}_i^h,\overline{s}^l+\overline{e}_j^l,\overline{r}^l-\overline{e}_j^l\right) \\
&+I\left(\left|\overline{r}^l\right|=0\right)\left(\sum_{i=1}^{N^l} a_i^l\right)^{\left|\overline{w}^l\right|} \sum_{i=1}^{N^h}\sum_{j=1}^{N^l}\left(s_i^h+1\right)\left(1+\delta_i^h\right) a_j^l P''_{n+1,m}\left(\overline{s}^h+\overline{e}_i^h,\overline{s}^l-\overline{e}_j^l,\overline{r}^l\right) \\
&+I\left(\left|\overline{r}^l\right|=0\right)\left(\sum_{i=1}^{N^l} a_i^l\right)^{\left|\overline{w}^l\right|} \sum_{i=1}^{N^l}\sum_{j=1}^{N^l}\left(s_i^l+1-e_{ij}\right)\left(1+\delta_i^l\right) a_j^l P''_{n,m+1}\left(\overline{s}^h,\overline{s}^l+\overline{e}_i^l-\overline{e}_j^l,\overline{r}^l\right) \\
&+\frac{1}{k}\left(\sum_{i=1}^{N^l} a_i^l\right)^{\left|\overline{w}^l\right|} \sum_{i=1}^{N^h}\sum_{j=1}^{N^l}\frac{r_j^l+1}{\left|\overline{r}^l+1\right|}\left(s_i^h+1\right)\left(1+\delta_i^h\right) P''_{n+1,m}\left(\overline{s}^h+\overline{e}_i^h,\overline{s}^l-\overline{e}_j^l,\overline{r}^l+\overline{e}_j^l\right) \\
&+\frac{1}{k}\left(\sum_{i=1}^{N^l} a_i^l\right)^{\left|\overline{w}^l\right|} \sum_{i=1}^{N^l}\sum_{j=1}^{N^l}\frac{r_j^l+1}{\left|\overline{r}^l+1\right|}\left(s_i^l+1-e_{ij}\right)\left(1+\delta_i^l\right) P''_{n,m+1}\left(\overline{s}^h,\overline{s}^l+\overline{e}_i^l-\overline{e}_j^l,\overline{r}^l+\overline{e}_j^l\right)
\end{aligned}
\tag{6.11}
$$

where $\sum_{i=1}^{N^l} a_i^l = 1$, and $\gamma = \frac{\mu^l}{\mu^h}$.

Again, this results in a matrix equation for $n < k$, $m \le k - n$, namely:

$$
\mathbf{D}_{n,m}\mathbf{P}_{n,m} = \mathbf{F}_{n,m}\mathbf{P}_{n-1,m} + \mathbf{E}_{n,m}\mathbf{P}_{n,m-1} + \mathbf{B}_{n,m}\mathbf{P}_{n+1,m} + \mathbf{G}_{n,m}\mathbf{P}_{n,m+1}
$$

where the operators $\mathbf{D}_{n,m}$, $\mathbf{F}_{n,m}$, $\mathbf{E}_{n,m}$, $\mathbf{B}_{n,m}$ and $\mathbf{G}_{n,m}$ on the vectors $\zeta\left[\overline{s}^h,\overline{s}^l,\overline{r}^l\right]$ are respectively defined as:

$$\mathbf{D}_{n,m}\zeta\left[\overline{s}^h,\overline{s}^l,\overline{r}^l\right]\overset{def}{=}\begin{cases}\left(1+\rho^h+\gamma\rho^l+\overline{\delta}\left(\overline{s}^h,\overline{s}^l\right)\right)\zeta\left[\overline{s}^h,\overline{s}^l,\overline{r}^l\right], & n+m\geq k\\ \left(\frac{n+m}{k}+\rho^h+\gamma\rho^l+\overline{\delta}\left(\overline{s}^h,\overline{s}^l\right)\right)\zeta\left[\overline{s}^h,\overline{s}^l,0\right], & n+m<k\end{cases}$$

$$\mathbf{F}_{n,m}\zeta\left[\overline{s}^h,\overline{s}^l,\overline{r}^l\right]\overset{def}{=}\begin{cases}\rho^h\sum_{i=1}^{N^h}\sum_{j=1}^{N^l}\frac{s_j^{\,l}+1}{\left|\overline{s}^l\right|+1}a_i^{\,h}\zeta\left[\overline{s}^h-\overline{e}_i^h,\overline{s}^l+\overline{e}_j^{\,l},\overline{r}^l-\overline{e}_j^{\,l}\right], & n+m>k\\ \rho^h\sum_{i=1}^{N^h}a_i^{\,h}\zeta\left[\overline{s}^h-\overline{e}_i^h,\overline{s}^l,0\right], & n+m\leq k\end{cases}$$

$$\mathbf{E}_{n,m}\zeta\left[\overline{s}^h,\overline{s}^l,\overline{r}^l\right]\overset{def}{=}\begin{cases}\gamma\rho^l\zeta\left[\overline{s}^h,\overline{s}^l,\overline{r}^l\right], & n+m>k\\ \gamma\rho^l\sum_{i=1}^{N^l}a_i^{\,l}\zeta\left[\overline{s}^h,\overline{s}^l-\overline{e}_i^{\,l},0\right], & n+m\leq k\end{cases}$$

$$\mathbf{B}_{n,m}\zeta\left[\overline{s}^h,\overline{s}^l,\overline{r}^l\right]\overset{def}{=}\begin{cases}\frac{1}{k}\sum_{i=1}^{N^h}\sum_{j=1}^{N^l}\left(s_i^{\,h}+1\right)\left(1+\delta_i^{\,h}\right)\frac{r_j^{\,l}+1}{\left|\overline{r}^l\right|+1}\zeta\left[\overline{s}^h+\overline{e}_i^h,\overline{s}^l-\overline{e}_j^{\,l},\overline{r}^l+\overline{e}_j^{\,l}\right], & n+m\geq k\\ \frac{1}{k}\sum_{i=1}^{N^h}\sum_{j=1}^{N^l}a_j^{\,l}\left(s_i^{\,h}+1\right)\left(1+\delta_i^{\,h}\right)\zeta\left[\overline{s}^h+\overline{e}_i^h,\overline{s}^l-\overline{e}_j^{\,l},0\right], & n+m\geq k,\quad \overline{r}^l=0\\ \frac{1}{k}\sum_{i=1}^{N^h}\left(s_i^{\,h}+1\right)\left(1+\delta_i^{\,h}\right)\zeta\left[\overline{s}^h+\overline{e}_i^h,\overline{s}^l,0\right], & n+m<k\end{cases}$$

$$\mathbf{G}_{n,m}\zeta\left[\overline{s}^h,\overline{s}^l,\overline{r}^l\right]\overset{def}{=}\begin{cases}\frac{1}{k}\sum_{i=1}^{N^l}\sum_{j=1}^{N^l}\left(s_i^{\,l}+1-e_{ij}^{\,l}\right)\left(1+\delta_i^{\,l}\right)\frac{r_j^{\,l}+1}{\left|\overline{r}^l\right|+1}\zeta\left[\overline{s}^h,\overline{s}^l+\overline{e}_i^l-\overline{e}_j^{\,l},\overline{r}^l+\overline{e}_j^{\,l}\right], & n+m\geq k\\ \frac{1}{k}\sum_{i=1}^{N^l}\sum_{j=1}^{N^l}a_j^{\,l}\left(s_i^{\,l}+1-e_{ij}^{\,l}\right)\left(1+\delta_i^{\,l}\right)\zeta\left[\overline{s}^h,\overline{s}^l+\overline{e}_i^l-\overline{e}_j^{\,l},0\right], & n+m\geq k,\quad \overline{r}^l=0\\ \frac{1}{k}\sum_{i=1}^{N^l}\left(s_i^{\,l}+1\right)\left(1+\delta_i^{\,l}\right)\zeta\left[\overline{s}^h,\overline{s}^l+\overline{e}_i^l,0\right], & n+m<k\end{cases}$$

The equilibrium equations (6.3) for *n* = *k* have a similar form, with the only difference that these equations include the probabilities of the system states with one high priority item in the queue (*n* = *k* + 1), which are equal to $\frac{1}{\left|\overline{w}^l\right|!}\left(\frac{d}{d\xi}\right)^{\left|\overline{w}^l\right|}\left[\mathbf{Z}^{-1}\left(\xi\right)\mathbf{C}\left(\xi\right)\right]$. So, we can aggregate all equations for states with no high priority items in queue (*n* ≤ *k*) and write them in matrix form. To this end, we write all vectors $\mathbf{P}_{n,m}$ for the states with the same number of items in the system (i.e. *n* + *m*) into one vector $\mathbb{P}_{\hat{t}}$, with $\hat{t}=n+m$. For example, then the equations for the states with more items in the system than the number of servers (*t* > *k*) can be written as:

$$\begin{pmatrix} \mathbf{D}_{0,m} & & & \mathbf{0} \\ & \mathbf{D}_{1,m-1} & & \\ & & \ddots & \\ \mathbf{0} & & & \mathbf{D}_{k,m-k} \end{pmatrix} \begin{pmatrix} \mathbf{P}_{0,m} \\ \mathbf{P}_{1,m-1} \\ \vdots \\ \mathbf{P}_{k,m-k} \end{pmatrix} = \begin{pmatrix} \mathbf{E}_{0,m} & & & & \mathbf{0} \\ \mathbf{F}_{1,m-1} & \mathbf{E}_{1,m-1} & & & \\ & \ddots & \ddots & & \\ \mathbf{0} & & & \mathbf{F}_{k,m-k} & \mathbf{E}_{k,m-k} \end{pmatrix} \begin{pmatrix} \mathbf{P}_{0,m-1} \\ \mathbf{P}_{1,m-2} \\ \vdots \\ \mathbf{P}_{k,m-k-1} \end{pmatrix}$$

$$+ \begin{pmatrix} \mathbf{G}_{0,m} & \mathbf{B}_{0,m} & & \mathbf{0} \\ & \mathbf{G}_{1,m-1} & \ddots & \\ & & \ddots & \mathbf{B}_{k-1,m-k+1} \\ \mathbf{0} & & & \mathbf{G}_{k,m-k} \end{pmatrix} \begin{pmatrix} \mathbf{P}_{0,m+1} \\ \mathbf{P}_{1,m} \\ \vdots \\ \mathbf{P}_{k,m-k+1} \end{pmatrix} + \begin{pmatrix} \mathbf{0} \\ \mathbf{0} \\ \vdots \\ \mathbf{B}_{k,m-k} \end{pmatrix} m! \left(\frac{d}{d\xi} \right)^m \left[\mathbf{Z}^{-1}(\xi) \mathbf{C}(\xi) \right]_{\xi=0}$$

The equations for $n + m \le k$ have the same form, but they do not include the inhomogeneous term.

Rewriting these equations in a matrix form, where the vector $\mathbb{P}_{\hat{t}}$ is composed of the vectors $\mathbf{P}_{n,m}$, we obtain:

$$\mathbb{D}_{\hat{t}} \mathbb{P}_{\hat{t}} = \mathbb{F}_{\hat{t}} \mathbb{P}_{\hat{t}-1} + \mathbb{G}_{\hat{t}} \mathbb{P}_{\hat{t}+1} + \mathbb{B}_{\hat{t}} \frac{1}{(\hat{t}-k)!} \left(\frac{d}{d\xi} \right)^{\hat{t}-k} \left[\mathbf{Z}^{-1}(\xi) \mathbf{C}(\xi) \right]_{\xi=0}, \quad \hat{t} \ge k \tag{6.12}$$

$$\mathbb{D}_{\hat{t}} \mathbb{P}_{\hat{t}} = \mathbb{F}_{\hat{t}} \mathbb{P}_{\hat{t}-1} + \mathbb{G}_{\hat{t}} \mathbb{P}_{\hat{t}+1}, \quad \hat{t} < k \tag{6.13}$$

where the matrices $\mathbb{D}_{\hat{t}}$, $\mathbb{F}_{\hat{t}}$ and $\mathbb{G}_{\hat{t}}$ are fixed for $\hat{t} \ge k$ and depend on $\hat{t}$ for $\hat{t} < k$.

The dimension of the equations in (6.12) does not depend on $\hat{t}$. It can be derived from the number of all combinations of $N^h + N^l$ items on k servers including all combinations of j postponed items $j \le n \le k$. We find:

$$\dim\left(\mathcal{L}_{"\hat{t} \ge k, n \le k"}\right) = \sum_{i=0}^{k} \left[\sum_{n=i}^{k} \binom{N^h + n - 1}{n} \binom{N^l + k - n - 1}{k - n} \right] \binom{N^l + i - 1}{i} \tag{6.14}$$

The dimension of the equations (6.13) depends on $\hat{t}$ and should be equal to the number of all combinations of $N^h + N^l$ items on $\hat{t}$ servers if $\hat{t} < k$, i.e.

$$\dim\left(\mathcal{L}_{"\hat{t} < k"}\right) = \binom{N^h + N^l + \hat{t} - 1}{\hat{t}} \tag{6.15}$$

So, we now have two systems of second order linear difference equations. The first system (for $\hat{t} \ge k$) is an inhomogeneous system with a fixed dimension and with fixed coefficients. The second system (for $\hat{t} < k$) has coefficients depending on t. Also, the system dimension depends on $\hat{t}$. In the next two sections, we describe how we can construct the solution of these two systems.

6.4.2 States with only low priority items in queue (*t* ≥ *k*, *n* ≤ *k*).

The probabilities of the system states having only low priority items in queue satisfy the system of linear inhomogeneous difference equations of second order with fixed coefficients (6.12). However, the inhomogeneous term has a differential form, so the standard procedure of solving the inhomogeneous equations (solution of homogeneous part + partial solution of inhomogeneous part) is difficult to apply. Therefore, we will look for the solution in a differential form $\mathbb{P}_{t+k} = \frac{1}{t!}\left(\frac{d}{d\xi}\right)^t v(\xi)_{\xi=0}$ where $t = \hat{t} - k$. The substitution of this solution into the equation (6.12) gives:

$$\mathbb{D}\frac{1}{t!}\left(\frac{d}{d\xi}\right)^t v(\xi)_{\xi=0} = \mathbb{F}\frac{1}{(t-1)!}\left(\frac{d}{d\xi}\right)^{t-1} v(\xi)_{\xi=0} + \mathbb{G}\frac{1}{(t+1)!}\left(\frac{d}{d\xi}\right)^{t+1} v(\xi)_{\xi=0}$$
$$+\mathbb{B}\frac{1}{t!}\left(\frac{d}{d\xi}\right)^t \left[\mathbf{Z}^{-1}(\xi)\mathbf{C}(\xi)\right]_{\xi=0}, \quad t > k$$

Here we can apply the following equations: $\left(\frac{d}{dx}\right)^t \left(xf(x)\right)_{x=0} = t\left[\left(\frac{d}{dx}\right)^{t-1} f(x)\right]_{x=0}$ and $\left(\frac{d}{dx}\right)^t \left(x^2 f(x)\right)_{x=0} = t(t-1)\left[\left(\frac{d}{dx}\right)^{t-2} f(x)\right]_{x=0}$. These equations can easily be proved as shown in Lemma 6.1. These two equations allow us to remove the derivatives from equation (6.12) and to obtain a new expression for the function *v*(*ξ*) for any *t* > 0:

$$\frac{1}{(t+1)!}\left(\frac{d}{d\xi}\right)^{t+1} \left[\xi\mathbb{D}v(\xi) - \xi^2\mathbb{F}v(\xi) - \mathbb{G}v(\xi) - \xi\mathbb{B}\mathbf{Z}^{-1}(\xi)\mathbf{C}(\xi)\right]_{\xi=0} = 0, \quad t > k \qquad (6.16)$$

The function **C**(*ξ*) can be expressed as a part of the vector-function *v*(*ξ*), which corresponds to the states with *k* high priority items in the system $v_k(\xi)$, i.e. $\mathbf{C}(\xi) = v_k(\xi)$.

The right part of equation (6.16) should be a function which becomes zero for any *t* > 0, i.e. a linear function. Hence, we obtain another expression for the vector-function *v*(*ξ*), that does not contain derivatives, but that contains unknown vectors C_1 and C_2:

$$\xi\mathbb{D}v(\xi) - \xi^2\mathbb{F}v(\xi) - \mathbb{G}v(\xi) - \xi\mathbb{B}\mathbf{Z}^{-1}(\xi)v_k(\xi) = C_1\xi + C_2 \qquad (6.17)$$

or

$$\mathbb{H}(\xi)v(\xi) = C_1\xi + C_2 \tag{6.18}$$

The constants C_1 and C_2 can be easily expressed via the probability states $\mathbb{P}_k$ and $\mathbb{P}_{k-1}$, i.e. we have from equation (6.17) that for $\xi = 0$,

$$-\mathbb{G}v(0) = C_2$$

and recalling that $v(0) = \mathbb{P}_k$, we obtain an equation for C_2:

$$C_2 = -\mathbb{G}\mathbb{P}_k$$

Next, we can take the derivative of equation (6.17) in the point $\xi = 0$ and we obtain

$$\mathbb{D}v(0) - \mathbb{G}v'(0) - \mathbb{B}\mathbf{Z}^{-1}(0)v_k(0) = C_1.$$

The left hand side terms of the last equation are also encountered in equation (6.12) for $t = k$, if we take into account that $\mathbb{P}_k = v(0)$, $\mathbb{P}_{k+1} = v'(0)$ and that the matrices $\mathbb{D}_k$, $\mathbb{G}_k$ and $\mathbb{B}_k$ are equivalent to $\mathbb{D}$, $\mathbb{G}$ and $\mathbb{B}$, respectively. Then we find:

$$C_1 = \mathbb{D}_k\mathbb{P}_k - \mathbb{G}_k\mathbb{P}_{k+1} - \mathbb{B}_k\mathbf{Z}^{-1}(0)v_k(0) = \mathbb{F}_k\mathbb{P}_{k-1}.$$

So, we have

$$C_1 = \mathbb{F}_k\mathbb{P}_{k-1}$$

In this way, we have defined a function $v(\xi)$, given the probability vectors $\mathbb{P}_k$ and $\mathbb{P}_{k-1}$. Next, all probability vectors $\mathbb{P}_t$ for $t = k + 1 \ldots \infty$ follow from $\mathbb{P}_k$ and $\mathbb{P}_{k-1}$. However, an essential piece of information has not been used up to now. It is clear that we are looking for decaying solutions $\mathbb{P}_{\hat{t}}$ for $\hat{t} \to \infty$. As a consequence, *v*(*ξ*) should be analytic on a circle with radius 1 + *ε* for some *ε* > 0. Due to (6.17), extra conditions have to be satisfied at points *ξ* inside this circle where $\mathbb{H}(\xi)$ is singular. It turns out that there are several such points in general. For example, *ξ* = 0 and *ξ* = 1 are points of this type. It is easy to check that in case *ξ* = 0 any vector with 0 entries whenever $\left|\overline{r}^l\right| > 0$ is in the null space of $\mathbb{G}$. Using the equilibrium property for subsystems with $\overline{s}^h, \overline{s}^l, \overline{r}^l$ fixed and with *m* arbitrary, it is not difficult to check that $\mathbf{1} = (1,\ldots,1)^\perp$ is a left eigenvector of $\mathbb{H}(1)$ for the eigenvalue 0. In the next section, we shall show that the decay requirement boils down to a relation between the initial condition $\mathbb{P}_{k+1}$ and $\mathbb{P}_k$ of the following type:

$$\mathbb{P}_{k+1} = \mathbb{Q}_k \mathbb{P}_k$$

In the next section, it will be shown that this can be done using a direct method without reference to singular points of $\mathbb{H}(\xi)$. This result is crucial in the case without queue.

6.4.3 Decay of $\mathbb{P}_{\hat{t}}$ for $\hat{t} \to \infty$

From (6.17), it follows by differentiating $t \geq 2$ times with respect to *ξ* that:

$$\binom{t}{0}\mathbb{H}(0)\frac{d^t}{d\xi^t}v(0) + \binom{t}{1}\mathbb{H}'(0)\frac{d^{t-1}}{d\xi^{t-1}}v(0) + \cdots + \binom{t}{t}\frac{d^t}{d\xi^t}\mathbb{H}(0)v(0) = 0.$$

We can also write this in the form

$$\sum_{i=0}^{t} h_i \mathbb{P}_{k+m-i} = 0$$

where $\mathbb{P}_{k+i} = \frac{1}{i!}\frac{d^i}{d\xi^i}v(0)$ and $h_i = \frac{1}{i!}\frac{d^i}{d\xi^i}\mathbb{H}(0)$.

An important consequence of this equation is that

$$\mathbb{P}_{k+1} = \mathbf{Q}_k^t \mathbb{P}_k + \mathbf{\Omega}_k^t \mathbb{P}_{k+t} \tag{6.19}$$

We will derive this result shortly. First, this implies that the solutions $\mathbb{P}_{k+t}$ decay for $t \to \infty$ if $\mathbf{\Omega}_k^t$ remains bounded:

$$\mathbb{P}_{k+1} = \mathbb{Q}_{k+1}\mathbb{P}_k \text{ with } \mathbb{Q}_{k+1} = \lim_{t\to\infty} \mathbf{Q}_k^t$$

Let us now show how the matrices $\mathbf{Q}_k^t$ and $\mathbf{\Omega}_k^t$ can be determined.

Using backward recursion from t to $k + 1$, we can show that for any t and t^* ($t > t^*$) a relation

$$\mathbf{\Theta}_0^{t^*}\mathbb{P}_{t+k} + \sum_{i=1}^{t^*} \mathbf{\Theta}_i^{t^*}\mathbb{P}_{t^*+k-i} = 0 \tag{6.20}$$

is valid. That is:

<u>For $t^* = t - 1$</u>, it is clear from the equation for $\mathbb{P}_{t+k}$ that we can take $\mathbf{\Theta}_i^t = h_i$.

<u>For $t^* < t - 1$</u>, we have the equation derived in the previous induction step and also the original equation for $\mathbb{P}_{t^*+k}$:

$$\begin{aligned} \mathbf{\Theta}_0^{t^*+1}\mathbb{P}_{t+k} + \mathbf{\Theta}_1^{t^*+1}\mathbb{P}_{t^*+k} + \cdots + \mathbf{\Theta}_{t^*+1}^{t^*+1}\mathbb{P}_k &= 0 \\ h_0\mathbb{P}_{t^*+k} + \cdots + h_{t^*}\mathbb{P}_k &= 0 \end{aligned}$$

Then, multiplying the first equation by $h_0\left(\mathbf{\Theta}_1^{t^*+1}\right)^{-1}$ and taking the difference, we obtain

$$\begin{aligned} h_0\left(\mathbf{\Theta}_1^{t^*+1}\right)^{-1}\mathbf{\Theta}_0^{t^*+1}\mathbb{P}_{t^*+k} &+ \left(h_0\left(\mathbf{\Theta}_1^{t^*+1}\right)^{-1}\mathbf{\Theta}_2^{t^*+1} - h_1\right)\mathbb{P}_{t^*+k-1} \\ &+ \cdots \\ &+ \left(h_0\left(\mathbf{\Theta}_1^{t^*+1}\right)^{-1}\mathbf{\Theta}_{t^*+1}^{t^*+1} - h_{t^*}\right)\mathbb{P}_k = 0 \end{aligned}$$

or in other terms

$$\mathbf{\Theta}_0^{t^*}\mathbb{P}_{t+k} + \mathbf{\Theta}_1^{t^*}\mathbb{P}_{t^*+k-1} + \cdots + \mathbf{\Theta}_{t^*}^{t^*}\mathbb{P}_k = 0$$

where the matrices $\mathbf{\Theta}_i^{t^*}$ are equal to

$$\mathbf{\Theta}_0^{t^*} = h_0\left(\mathbf{\Theta}_1^{t^*+1}\right)^{-1}\mathbf{\Theta}_0^{t^*+1}$$
$$\mathbf{\Theta}_i^{t^*} = \left(h_0\left(\mathbf{\Theta}_1^{t^*+1}\right)^{-1}\mathbf{\Theta}_{i+1}^{t^*+1} - h_i\right), \quad i = 1,\ldots t^*$$

So, we have shown that for any t and t^* ($t > t^*$) the relation (6.20) is valid. Next, taking $t^* = 2$, we obtain

$$\mathbf{\Theta}_0^2\mathbb{P}_{t+k} + \mathbf{\Theta}_1^2\mathbb{P}_{k+1} + \mathbf{\Theta}_2^2\mathbb{P}_k = 0$$

This is the desired result if we identify $\mathbf{\Omega}_k^t = -\left(\mathbf{\Theta}_1^2\right)^{-1}\mathbf{\Theta}_0^2$ and $\mathbf{Q}_k^t = -\left(\mathbf{\Theta}_1^2\right)^{-1}\mathbf{\Theta}_2^2$. Herewith the relation (6.19) is shown.

Note that the matrices $\mathbf{\Omega}_k^t$ and $\mathbf{Q}_k^t$ can be computed using a straightforward iteration procedure, which requires O(t^2) matrix operations. In this iteration, we check the boundedness of $\mathbf{\Omega}_k^t$. Numerical evidence shows that this is the case in all cases we considered. Moreover we found that

$$\lim_{t\to\infty}\left\|\mathbf{\Omega}_k^t\right\| \approx \omega(1-\rho)$$

where ω is some constant depending on other system parameters. Hence, by taking t sufficiently large, we have a good approximation of the matrix $\mathbb{Q}_k$ that will play a role next. Furthermore, $\rho \to 1$ is not a problem for the convergence of the iterations.

In the computation of performance measures, we shall also need information about the function $v(\xi)$ at $\xi = 1$. As an intermezzo, we describe in the next subsection how such information can now easily be obtained.

6.4.4 On the vectors *v*(1), *v*′(1) and *v*″(1).

Let us discuss now the procedure to find $v(1)$, $v'(1)$ and $v''(1)$, which we need for the performance characteristics. This procedure is non-trivial due to singularity of the matrix $\mathbb{H}(1)$. Therefore, we cannot find the vector $v(1)$ just by inverting the equation (6.17) at $\xi = 1$

$$\mathbb{H}(1)v(1) = \mathbb{F}_k\mathbb{P}_{k-1} - \mathbb{G}\mathbb{P}_k\,. \tag{6.21}$$

We use the Taylor expansion and we derive from (6.17) that

$$\mathbb{H}(1)v'(1) + \mathbb{H}'(1)v(1) = \mathbb{F}_k\mathbb{P}_{k-1} \tag{6.22}$$

In this equation, we have to get rid of the term with $v'(1)$. It can be done if we project in (6.22) both sides onto the null-space of $\mathbb{H}(1)$. Hence, we multiply this equation by the matrix Pr_1 constructed as

$$\mathrm{Pr}_1 = \varepsilon S\begin{pmatrix}\mathbf{1} & \\ & \mathbf{0}\end{pmatrix}S^{-1}$$

where S is the matrix of eigenvectors of $\mathbb{H}(1)$. The block with 1 on the diagonal corresponds to the zero eigenvalue and the block with 0 on the diagonal to non-zero eigenvalues; ε is some number > 0 that can be used for scaling purposes. It is easy to show that multiplication of the matrices Pr_1 and $\mathbb{H}(1)$ gives zero:

$$\mathrm{Pr}_1\,\mathbb{H}(1) = \varepsilon S\begin{pmatrix}\mathbf{1} & \\ & \mathbf{0}\end{pmatrix}S^{-1}\mathbb{H}(1) = \varepsilon S\begin{pmatrix}\mathbf{1} & \\ & \mathbf{0}\end{pmatrix}\begin{pmatrix}\mathbf{0} & \\ & *\end{pmatrix}S^{-1} = 0$$

So, we have now a new system of linear equations

$$\left[\mathbb{H}(1) + \mathrm{Pr}_1\,\mathbb{H}'(1)\right]v(1) = \left(\mathbf{I} + \mathrm{Pr}_1\right)\mathbb{F}_k\mathbb{P}_{k-1} - \mathbb{G}\mathbb{P}_k$$

where the matrix $\left[\mathbb{H}(1)+\mathrm{Pr}_1\,\mathbb{H}'(1)\right]$ turns out to be non-singular. In all our experiments, it turned out that 0 is a single eigenvalue of $\mathbb{H}(1)$. As we saw before, the corresponding left eigenvector is $\mathbf{1}=(1,\ldots,1)^{\perp}$. As a consequence, the matrix Pr1 can be constructed as a matrix with elements of one row (any row) equal to 1.

$$\mathrm{Pr}_1 = \begin{pmatrix} 1 & \cdots & 1 \\ & \ddots & \\ 0 & \cdots & 0 \end{pmatrix}$$

In the same way, we can find the derivatives v′(1), v″(1), etc. That is

$$\begin{aligned}
&\left[\mathbb{H}(1)+2\,\mathrm{Pr}_1\,\mathbb{H}'(1)\right]v'(1) = -\left[\mathbb{H}'(1)+\mathrm{Pr}_1\,\mathbb{H}''(1)\right]v(1)+\mathbb{F}_k\mathbb{P}_{k-1} \\
&\left[\mathbb{H}(1)+3\,\mathrm{Pr}_1\,\mathbb{H}'(1)\right]v''(1) = -\left[2\mathbb{H}'(1)+3\,\mathrm{Pr}_1\,\mathbb{H}''(1)\right]v'(1)-\left[\mathbb{H}''(1)+\mathrm{Pr}_1\,\mathbb{H}'''(1)\right]v(1)
\end{aligned}$$

6.4.5 States with empty queue ($\hat{t} < k$)

The equilibrium equations for $\hat{t} < k$ do not have an inhomogeneous term:

$$\mathbb{D}_{\hat{t}}\mathbb{P}_{\hat{t}} = \mathbb{F}_{\hat{t}}\mathbb{P}_{\hat{t}-1} + \mathbb{G}_{\hat{t}}\mathbb{P}_{\hat{t}+1}$$

However, now the matrices $\mathbb{D}_{\hat{t}}, \mathbb{F}_{\hat{t}}, \mathbb{G}_{\hat{t}}$ depend on the value of $\hat{t}$. We can find $\mathbb{P}_{\hat{t}}$ for $\hat{t} < k$ using a recurrent expression. The complete solution can be represented as:

$$\mathbb{P}_{\hat{t}} = \mathbb{Q}_{\hat{t}}\mathbb{P}_{\hat{t}-1} = \mathbb{Q}_{\hat{t}}\mathbb{Q}_{\hat{t}-1}\cdots\mathbb{Q}_1\mathbb{P}_0$$

where $\mathbb{Q}_{\hat{t}}$ follows recursively from

$$\mathbb{Q}_{\hat{t}} = \left(\mathbb{D}_{\hat{t}} - \mathbb{G}_{\hat{t}}\mathbb{Q}_{\hat{t}+1}\right)^{-1}\mathbb{F}_{\hat{t}}, \quad \hat{t} = k,\ldots,1$$

In this recurrent relation, the matrix $\mathbb{Q}_k$ was found in the previous section. The free constant $\mathbb{P}_0$ is determined by

$$\sum_{\overline{w}^h,\overline{s}^h,\overline{w}^l,\overline{s}^l,\overline{r}^l} P_{n,m}\left(\overline{w}^h,\overline{s}^h,\overline{w}^l,\overline{s}^l,\overline{r}^l\right)=1$$

or

$$\sum_{\hat{t}\le k} P_{n,m}\left(0,\overline{s}^h,0,\overline{s}^l,0\right)+\sum_{\left|\overline{r}^l\right|\le n\le k,\,\hat{t}>k} P_{n,m}\left(0,\overline{s}^h,\overline{w}^l,\overline{s}^l,\overline{r}^l\right)$$
$$+\sum_{n>k,\left|\overline{r}^l\right|\le k} P_{n,m}\left(\overline{w}^h,\overline{s}^h,\overline{w}^l,0,\overline{r}^l\right)=1$$

It is easy to show that the sum $\sum_{\left|\overline{r}^l\right|\le n\le k,\,\hat{t}>k} P_{n,m}\left(0,\overline{s}^h,\overline{w}^l,\overline{s}^l,\overline{r}^l\right)$ is equal to $v(1)$. From the previous section, we know that $v(1)$ can be expressed in $\mathbb{P}_k$ and $\mathbb{P}_{k-1}$. Hence $v(1)=\mathbb{P}_0 v^0(1)$, where $v^0(1)$ can simply be expressed in $\mathbb{F}_k$, $\mathbb{G}$ and in products of the sort $\mathbb{Q}_{\hat{t}}\cdots\mathbb{Q}_1$. Now $\mathbb{P}_0$ follows from:

$$\mathbb{P}_0\left\{1+\sum_{i=1}^{k-1}\left\langle \mathbf{1}_{\hat{t}},\mathbb{Q}_{\hat{t}}\cdots\mathbb{Q}_1\right\rangle+\left\langle \mathbf{1}_{n\le k,\hat{t}\ge k},v^0(1)\right\rangle+\left\langle \mathbf{1}_{n>k},\left(\mathbf{I}-\mathbf{Z}^{-1}(1)\right)^{-1}v_k^0(1)\right\rangle\right\}=1,$$

where $v_k^0(1)$ is the part of the vector $v^0(1)$ with k high priority items in the system and 1_* is a vector with all elements equal to 1 and with a dimension corresponding to dimension of the state spaces (cf. expressions (6.14)-(6.15)), i.e. $\dim\left(\mathbf{1}_{\hat{t}}\right)=\binom{N^h+N^l+\hat{t}-1}{\hat{t}}$, $\dim\left(\mathbf{1}_{n\le k,\hat{t}>k}\right)$ = $\sum_{i=0}^{k}\left[\sum_{n=i}^{k}\binom{N^h+n-1}{n}\binom{N^l+k-n-1}{k-n}\right]\binom{N^l+i-1}{i}$ and $\dim\left(\mathbf{1}_{n>k}\right)=\binom{N^h+k-1}{k}$

Hence, we found the probability states for $\hat{t}=0,\ldots,k+1$ and the values of the first derivative of the function $v(1)$ that we need to calculate the performance measures for the queueing system.

6.5 Performance measures

In this section, we will concentrate on the performance criteria for the low priority items in the system, since performance indicators for the high priority items can be calculated using the non-priority multi-class, multi-serve queue analysis presented in Chapter 3. The latter is possible due to the preemptive priority rule in our system, i.e. low priority items don't influence processing of high priority items, and therefore they can be ignored.

We show here how to calculate the mean number of low priority items of type *i* in the queue, the mean number of the low priority items of type *i* in the postponed state and the first two moments of the total number of the low priority items of type *i* in the system. We denote by q_i^l, PS_i^l and R_i^l the number of low priority items of subclass *i* in the queue, in the postponed state and in the system, respectively. Such performance indicators play a role in spare part service networks as discussed in the previous chapters. Other interesting performance measures are the expected number of postponements per item of type *i*, the expected residential time of items of type *i* in service, in queue and postponed, respectively.

The mean number of the low priority item *i* in the queue can be found as the sum over all states with low priority items in the queue (i.e. zones I and II) of the state probability multiplied by the number of low priority items of type *i* in the queue in the respective states:

$$E\left[q_i^l\right]=\sum_{n=0}^{\infty}\sum_{m=0}^{\infty}\sum_{\substack{\overline{w}^h,\overline{s}^h\\ \left|\overline{w}^h\right|+\left|\overline{s}^h\right|=n}}\;\sum_{\substack{\overline{w}^l,\overline{s}^l,\overline{r}^l\\ \left|\overline{w}^l\right|+\left|\overline{s}^l\right|+\left|\overline{r}^l\right|=m}} w_i^l P_{n,m}\left(\overline{w}^h,\overline{s}^h,\overline{w}^l,\overline{s}^l,\overline{r}^l\right).$$

In this expression, various terms can be simplified via the function $v(\xi)$ and via the matrix $\mathbf{Z}(\xi)$ using the Taylor expansion:

$$\begin{aligned}E\left[q_i^l\right]=&\sum_{n=k+1}^{\infty}\sum_{m=0}^{\infty}\sum_{\substack{\overline{w}^h,\overline{s}^h\\ \left|\overline{w}^h\right|+\left|\overline{s}^h\right|=n}}\;\sum_{\substack{\overline{w}^l,\overline{r}^l\\ \left|\overline{w}^l\right|+\left|\overline{r}^l\right|=m,\left|\overline{r}^l\right|\le k}} w_i^l P_{n,m}\left(\overline{w}^h,\overline{s}^h,\overline{w}^l,0,\overline{r}^l\right)\\ &+\sum_{n=0}^{k}\sum_{m=0}^{\infty}\sum_{\substack{\overline{s}^h\\ \left|\overline{s}^h\right|=n}}\;\sum_{\substack{\overline{w}^l,\overline{s}^l,\overline{r}^l\\ \left|\overline{w}^l\right|+\left|\overline{s}^l\right|+\left|\overline{r}^l\right|=m,\left|\overline{r}^l\right|\le k}} w_i^l P_{n,m}\left(0,\overline{s}^h,\overline{w}^l,\overline{s}^l,\overline{r}^l\right)\\ =&\sum_{m=1}^{\infty}\sum_{n=k+1}^{\infty}\sum_{\substack{\overline{s}^h\\ \left|\overline{s}^h\right|=k}}\sum_{\substack{\overline{r}^l\\ \left|\overline{r}^l\right|\le k}} a_i^l\frac{1}{\left(\left|\overline{w}^l\right|-1\right)!}\left(\frac{d}{d\xi}\right)^{\left|\overline{w}^l\right|}\left[\left(\mathbf{Z}(\xi)^{-1}\right)^{n-k}v_k(\xi)\right]_{\xi=0}\\ &+\sum_{n=0}^{k}\sum_{m'=k-n}^{\infty}\sum_{w_i^l,s_i^l r_i^l} w_i^l m'!\prod_{j=1}^{N^l}\frac{\left(a_j^l\right)}{w_j^l\,!}\left[\frac{1}{m'!}\frac{d^{m'}}{d\xi^{m'}}v(\xi)\right]_{\xi=0}\end{aligned}$$

This finally gives us

$$E\left[q_i^l\right] = -a_i^l \left\langle \mathbf{1}_{n>k}, \left(\mathbf{Z}(1)-\mathbf{I}\right)^{-1} \mathbf{Z}'(1) \left(\mathbf{Z}(1)-\mathbf{I}\right)^{-1} v_k(1) \right\rangle$$
$$+ a_i^l \left\langle \mathbf{1}_{n>k}, \left(\mathbf{Z}(1)-\mathbf{I}\right)^{-1} \mathbf{Z}(1) v_k'(1) \right\rangle + a_i^l \left\langle \mathbf{1}_{n\le k, \hat{t}\ge k}, v'(1) \right\rangle$$

Here v_k refers to the vector components of v with $\left|\overline{s}^l\right| = k$. The notation **1**$_*$ is used for a vector with components 1 of the dimension indicated by the subscript as introduced before.

In an analogous way, we obtain

$$E\left[PS_i^l\right] = \left\langle \chi_{n>k}^{r_i^l}, v_k(1) \right\rangle + \left\langle \chi_{n\le k, \hat{t}\ge k}^{r_i^l}, v(1) \right\rangle$$

where $\chi_{n>k}^{r_i^l}$ and $\chi_{n\le k,\hat{t}\ge k}^{r_i^l}$ have the number of postponed low priority items of type i corresponding to the vector component $\left(\overline{s}^h, \overline{s}^l, \overline{r}^l\right)$ as components.

Using similar vectors $\chi_{n\le k,\hat{t}\ge k}^{s_i^l}$ and $\chi_{\hat{t}}^{s_i^l}$, which keep track of amounts of the low priority items in service, we can write down an expression for the mean number of items of type i in service:

$$E\left[SR_i^l\right] = \sum_{\hat{t}=1}^{k-1} \left\langle \chi_{\hat{t}}^{s_i^l}, \mathbb{Q}_{\hat{t}} \cdots \mathbb{Q}_1 \mathbb{P}_0 \right\rangle + \left\langle \chi_{n\le k,\hat{t}\ge k}^{s_i^l}, v(1) \right\rangle$$

Note that the mean number of items of type i in service can also be estimated via Little's law, i.e.

$$E\left[SR_i^l\right] = \lambda_i^l / \mu_i^l$$

We can use this as a test to check the accuracy of our numerical approximations in the next section.

The expected value of R_i^l is composed of these three terms:

$$E\left[R_i^l\right] = E\left[SR_i^l\right] + E\left[q_i^l\right] + E\left[PS_i^l\right]$$

Next, Little's law can be applied to calculate performance indicators as the mean waiting time until the *first* time that an item enters service $E\left[W_i^l\right]$, the mean postponement time $E\left[PsTime_i^l\right]$ and the mean sojourn time $E\left[SJTime_i^l\right]$:

$$\begin{aligned}
\lambda_i^l E\left[W_i^l\right] &= E\left[q_i^l\right] \\
\lambda_i^l E\left[PsTime_i^l\right] &= E\left[PS_i^l\right] \\
\lambda_i^l E\left[SJTime_i^l\right] &= E\left[R_i^l\right]
\end{aligned}$$

and, of course,

$$E\left[SJTime_i^l\right] = \frac{1}{\mu_i^l} + E\left[W_i^l\right] + E\left[PsTime_i^l\right]$$

Note that $E\left[PsTime_i^l\right]$ has to be interpreted as: the expected *total* time that an item of type *i* spends in a postponed state between the moment it leaves the queue and the moment its service is completed. Further, we should note that even though preemption occurs, the expected total service time equals $1/\mu_i^l$ due to memoryless property of the exponential distributions of the service times.

Let us now focus on another interesting quantity: $E\left[nrPreemptEvent_i^l\right]$, the expected number of preemption events per item of type *i*. In order to compute it, we need the arrival rate of a low priority item into the postponed state. It can be calculated using the state probabilities as derived in the previous sections. The arrival rate of low priority items *i* into postponed state is equal to the arrival rate of high priority items multiplied by the probability that item *i* was withdrawn from service:

$$\lambda_i^{ps} = \Lambda^h \sum_{\overline{s}^h, \overline{w}^l, \overline{s}^l, \overline{r}^l} \frac{s_i^l}{\left|\overline{s}^l\right|} P_{n,m}\left(0, \overline{s}^h, \overline{w}^l, \overline{s}^l, \overline{r}^l\right) = \Lambda^h \left\langle \chi_{n \le k, \hat{t} \ge k}^{\frac{s_i^l}{}}, v(1) \right\rangle$$

where the components of the vectors $\chi^{\frac{s_i^l}{}}_{n\leq k,\hat{t}\geq k}$ are equal to $\frac{s_j^l}{\left|\overline{s}^h\right|}$ of the corresponding vector component $\left(\overline{s}^h,\overline{s}^l,\overline{r}^l\right)$.

Comparing the number of preemption events with the number of arrivals over a long interval, it is clear that number of preemption events per item entering the system is equal to:

$$E\left[nrPreemptEvent_i^l\right]=\lambda_i^{ps}/\lambda_i^l$$

It is now also possible to compute the expected time between the moment an item of type i is postponed and the next moment when the service process was resumed again, i.e. the expected re-entrance into service time, $E\left[ReenterTime_i^l\right]$. Using Little's law again, we obtain

$$\lambda_i^{ps}E\left[ReenterTime_i^l\right]=E\left[PS_i^l\right]$$

Note that the number of preemption events per each item entering the system can be also calculated as:

$$E\left[nrPreemptEvent_i^l\right]=E\left[PsTime_i^l\right]\Big/E\left[ReenterTime_i^l\right]$$

Let us conclude this section on performance measures with some remarks on the calculation of the second moment of R_i^l. Here we should take into account the correlations between numbers of items in queue, in service and in the postponed states. This can be done analogous to computations for non-priority systems presented in Chapter 3. After a lengthy derivation we obtain:

$$\begin{aligned}
E\left[\left(R_i^l\right)^2\right] = {} & \left(a_i^l\right)^2 \left\langle \mathbf{1}_{n>k}, 2\left[\left(\mathbf{Z}(1)-\mathbf{I}\right)^{-1}\mathbf{Z}'(1)\left(\mathbf{Z}(1)-\mathbf{I}\right)^{-1}\mathbf{Z}'(1)\left(\mathbf{Z}(1)-\mathbf{I}\right)^{-1}\right.\right. \\
& \qquad\qquad \left.\left. -\left(\mathbf{Z}(1)-\mathbf{I}\right)^{-1}\mathbf{Z}''(1)\left(\mathbf{Z}(1)-\mathbf{I}\right)^{-1}\right] v_k(1)\right\rangle \\
& -2\left(a_i^l\right)^2 \left\langle \mathbf{1}_{n>k}, \left(\mathbf{Z}(1)-\mathbf{I}\right)^{-1}\mathbf{Z}'(1)\left(\mathbf{Z}(1)-\mathbf{I}\right)^{-1}\mathbf{Z}'(1)\left(\mathbf{Z}(1)-\mathbf{I}\right) v_k'(1)\right\rangle \\
& +\left(a_i^l\right)^2 \left\langle \mathbf{1}_{n>k}, \left(\mathbf{Z}(1)-\mathbf{I}\right)^{-1}\mathbf{Z}(1) v_k''(1)\right\rangle \\
& -a_i^l \left\langle \mathbf{1}_{n>k}, \left(\mathbf{Z}(1)-\mathbf{I}\right)^{-1}\mathbf{Z}'(1)\left(\mathbf{Z}(1)-\mathbf{I}\right)^{-1} v_k(1)\right\rangle + a_i^l \left\langle \mathbf{1}_{n>k}, \left(\mathbf{Z}(1)-\mathbf{I}\right)^{-1} v_k'(1)\right\rangle \\
& -2a_i^l \left\langle \chi_{n>k}^{r_i^l}, \left(\mathbf{Z}(1)-\mathbf{I}\right)^{-1}\mathbf{Z}'(1)\left(\mathbf{Z}(1)-\mathbf{I}\right)^{-1} v_k(1)\right\rangle \\
& +2a_i^l \left\langle \chi_{n>k}^{r_i^l}, \left(\mathbf{Z}(1)-\mathbf{I}\right)^{-1} v_k'(1)\right\rangle + \left\langle \chi_{n>k}^{\left(r_i^l\right)^2}, \left(\mathbf{I}-\mathbf{Z}^{-1}(1)\right)^{-1} v_k(1)\right\rangle \\
& +\left(a_i^l\right)^2 \left\langle \mathbf{1}_{n\le k,\hat{t}\ge k}, v''(1)\right\rangle + a_i^l \left(\left\langle \mathbf{1}_{n\le k,\hat{t}\ge k}, v'(1)\right\rangle + 2\left\langle \chi_{n\le k,\hat{t}\ge k}^{r_i^l+s_i^l}, v'(1)\right\rangle\right) \\
& +\left\langle \chi_{n\le k,\hat{t}\ge k}^{\left(r_i^l+s_i^l\right)^2}, v(1)\right\rangle + \sum_{\hat{t}=1}^{k-1}\left\langle \chi_{\hat{t}}^{\left(s_i^l\right)^2}, \mathbb{Q}_{\hat{t}}\cdots\mathbb{Q}_1\mathbb{P}_0\right\rangle
\end{aligned}$$

The second moments of SR_i^l, q_i^l and PS_i^l can be given analogously.

6.6 Numerical experiments

We implemented our algorithm in Borland Delphi 5 and we ran several experiments on a Pentium III-500 PC. In this section, we discuss the results of three sets of numerical experiments. First, we study the convergence speed and computer time requirements of our iterative procedure. Next, we examine the impact of the system parameters on the key performance characteristics for the low priority items as presented in the previous section. Finally, we compare the number of items in the system for priority queues to non-priority queues.

6.6.1 Convergence of iterative procedure and computation times

To examine the computational efforts necessary to obtain accurate results by the proposed algorithm, we first examine the estimates for the first and second moments of numbers of items in the system as function of the numbers of iterations $\hat{t}$ in our algorithm, see formula (6.19).

We chose the following parameter settings for our numerical experiments:

- k is equal to 4;
- N^h and N^l are fixed to 2;
- the arrival rates are fixed and equal to: $\lambda_1^h = 1.75$, $\lambda_2^h = 2.25$, $\lambda_1^l = 2.75$, $\lambda_2^l = 3.25$;
- the service rates satisfy: $\mu_i^h = \mu^h\left(1+\delta_i^h\right)$ and $\mu_i^l = \mu^h\left(1+\delta_i^l\right)$, where the $\delta_1^* = 0.5$ and $\delta_2^* = -0.5$;
- μ^h is varied so that ρ equals 0.75, 0.85 or 0.95.

As performance measures, we focus on the mean and variance of the number of low priority items in the system per class, because the characteristics of high priority items are not new (they can also be obtained using our method from Chapter 3).
First, we compared the results of our numerical procedure to some results obtained by discrete event simulation. We found that the deviation between the calculated and simulated performance measures already lies within 95% confidence interval after 10 iterations. Further, we have checked whether the test $E\left[SR_i^l\right] = \lambda_i^l / \mu_i^l$ from the previous section is satisfied. The experiments have shown that the average relative error of this test is ~0.2% after 20 iterations and ~0.04% after 40 iterations.

Next, we examine the convergence speed and computer time requirements of our iterative procedure. In Figure 6.2, we plot the maximum error for the mean and the variance of the numbers of low priority items in the system obtained by our algorithm as function of the number of iterations. As benchmark, we use the results after 80 iterations, because then the values of the performance measures hardly change anymore in all experiments.

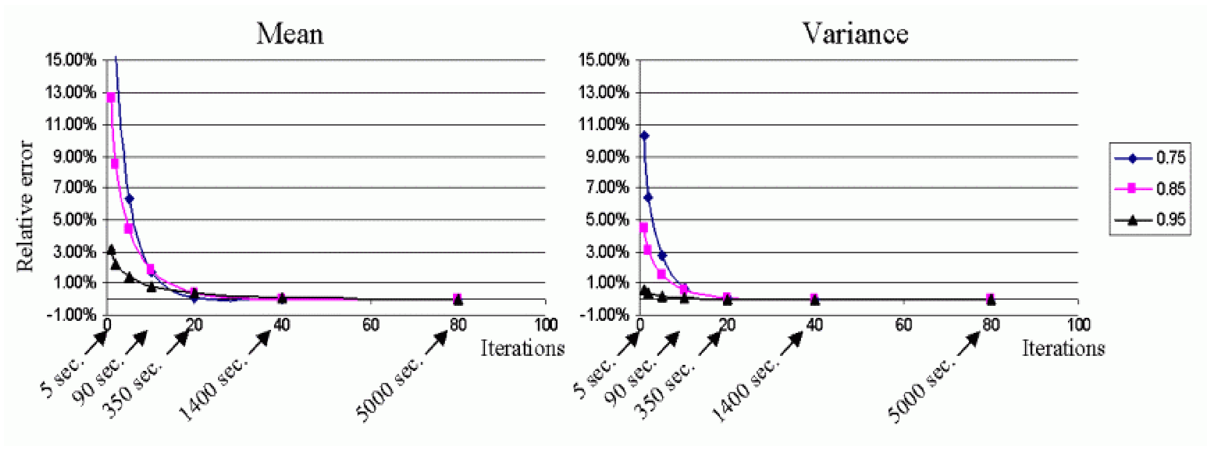

Figure 6.2 Error in performance estimators of low priority items against amounts of iterations and computation time for cases with 4 item types and 4 servers and different utilisation rates

Under the x-axis, we show the CPU time required for our calculations using a Pentium III-500 PC. We find that the CPU time is approximately a quadratic function of the number of iterations. The reason of this behaviour is that for t iterations the algorithm requires O(t^2) matrix operations. As a consequence, we need long run times to obtain extremely accurate results. Figure 6.2 also shows that the CPU-time requirements are modest if we accept a slight approximation error. For example, the maximum approximation error is less than 1% after 10 iterations, and we need only 90 seconds for these calculations.

Figure 6.2 also shows another interesting characteristic of our algorithm. We see that for systems with high utilisation rates we can obtain good results for low priority items with less iterations. These cases are most interesting for our application (spare part management). Further, we have an exact method to estimate the performance characteristics of high priority items that is independent of ρ as for computational effort (cf. Chapter 3). Together, we are able to obtain sufficiently accurate results for the performance characteristics required, especially for the most practical cases (high values for ρ).

6.6.2 Impact of system parameters on performance characteristics

Next, we study the influence of the most important parameters of the MCMS priority system on the performance measures. From queueing theory, we know that the total utilisation rate ρ and the number of servers k are such parameters for any queueing system. However, we have learned from the experiments with the MCMS non-priority queueing system (Chapter 3) that the fractions of arrival rates (a_i) and the perturbations of the service times (δ_i) might also seriously influence the performance characteristics of the queueing system.

We did computations for a large set of instances. Since the effects we want to discuss are already present for small systems, we shall only present the results on experiments for 3 servers and 3 items. One of items has high priority and two have low priority. The utilisation rate ρ is fixed to 95%. First of all, we want to see the influence of the difference between the service rates. This difference is defined by the parameters γ and $\frac{\mu_1^l}{\mu_2^l}$, where μ_1^l and μ_2^l are the service rates of the low priority classes. Also, we would like to see the influence of the utilisation rate for high priority items. Hence, we vary three parameters: ρ^h, γ and $\frac{\mu_1^l}{\mu_2^l}$. The other parameters are either fixed (e.g., the fractions of arrival rates within the group of low priority items are equal to $a_1^l = 0.3$ and $a_2^l = 0.7$), or they are completely defined by the other parameters (e.g. δ_1^l and δ_2^l). In this way, we have 18 experiments. We choose the values 0.5, 1 and 2 for both γ and $\frac{\mu_1^l}{\mu_2^l}$ and 20% and 60% for ρ^h. Thereby, we obtain 3×3×2=18 model runs.

Some interesting performance characteristics are presented in Table 6.1: the expected total number of items in the system for each low priority subclass $E\left[R_i^l\right]$, the expected number of items in the queue for each low priority subclass $E\left[q_i^l\right]$, the expected number of the postponed items for each low priority subclass $E\left[PS_i^l\right]$ and the expected number of preemption events per each low priority item entering the system $\frac{\lambda_i^{ps}}{\lambda_i^l}$.

Table 6.1 Impact of high priority utilisation and service rate ratios on the performance of low priority items; each cell in the table contains the performance measure for both low priority item classes

		$\rho^h = 20\%$				$\rho^h = 60\%$			
$\frac{\mu_1^l}{\mu_2^l}$	γ	$E\left[R_i^l\right]$	$E\left[q_i^l\right]$	$E\left[PS_i^l\right]$	$\lambda_i^{ps}/\lambda_i^l$	$E\left[R_i^l\right]$	$E\left[q_i^l\right]$	$E\left[PS_i^l\right]$	$\lambda_i^{ps}/\lambda_i^l$
0.5	0.5	6.21	5.03	0.14	0.73	4.34	3.38	0.47	2.95
		13.10	11.73	0.16	0.36	8.98	7.89	0.52	1.43
	1	6.74	5.59	0.11	0.37	5.85	4.96	0.40	1.49
		14.37	13.04	0.12	0.18	12.58	11.58	0.44	0.72
	2	7.78	6.66	0.07	0.18	8.87	8.06	0.33	0.75
		16.84	15.55	0.08	0.09	19.72	18.80	0.35	0.36
1	0.5	5.31	4.54	0.09	0.47	3.80	3.18	0.30	1.89
		12.38	10.59	0.21	0.47	8.86	7.43	0.69	1.89
	1	5.84	5.10	0.07	0.24	5.32	4.76	0.25	0.95
		13.63	11.90	0.15	0.24	12.42	11.11	0.58	0.95
	2	6.89	6.17	0.05	0.12	8.37	7.85	0.20	0.48
		16.08	14.40	0.11	0.12	19.52	18.31	0.47	0.48
2	0.5	5.27	4.83	0.05	0.27	3.65	3.30	0.17	1.08
		13.37	11.26	0.25	0.56	9.39	7.70	0.83	2.23
	1	5.82	5.39	0.04	0.14	5.20	4.88	0.14	0.54
		14.61	12.57	0.19	0.28	12.95	11.38	0.70	1.12
	2	6.88	6.46	0.03	0.07	8.27	7.97	0.11	0.27
		17.06	15.08	0.13	0.14	20.04	18.60	0.57	0.56

From Table 6.1, we can draw the following conclusions:

- The numbers of items in the postponed state increase with the utilisation rate of the high priority items ρ^h (this is not a trivial result since $\rho = \rho^h + \rho^l$ is constant, hence increasing ρ^h means decreasing ρ^l).
- The total number of postponed items hardly depends on the ratio $\frac{\mu_1^l}{\mu_2^l}$.
- However, the dependence of the total number of low priority items in queue on the ratio $\frac{\mu_1^l}{\mu_2^l}$ is remarkable. Namely, the numbers of items in queue (hence, the waiting times) are lower when service times of low priority items are equal ($\mu_1^l = \mu_2^l$), than

when perturbations of service times of low priority items occur. This can be interpreted as a sort of Pollaczek-Khintchine effect (cf. Tijms, 1994), i.e. the average waiting time increases when the variability of the service time increases.

- The number of low priority items in the queue (hence, the waiting times) is lower when the service times of high priority items are shorter (γ is smaller).
- The number of low priority items in the queue decreases with the utilisation rate of high priority items (ρ^h) when the service time of the high priority items is shorter or equal ($\gamma \leq 1$) and it increases when the service time of the high priority items is longer than the average service time of the low priority items ($\gamma > 1$).

It is also possible to derive from this table the waiting times in the queue and in the postponed state and the total time spent in the system. This can be done using Little's law as was shown in section 6.5.

6.6.3 Comparison between priority and non-priority queues.

To conclude this section, we sketch the effect of applying a priority queueing rule. Therefore, we vary ρ in the experiment with $N^h = 1$, $N^l = 2$, $k = 3$, $\mu_1^l / \mu_2^l = 0.5$, $\gamma = 0.5$ and $\rho^h = 0.6\rho$, $\rho^l = 0.4\rho$. We compare the total numbers of items in the system for cases with and without priority rules (Figure 6.3).

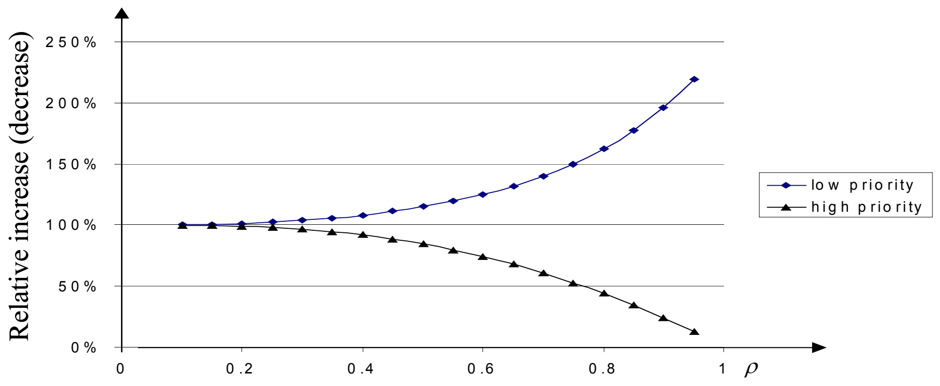

Figure 6.3 Relative increase (decrease) of the total number of low (high) priority items caused by introduction of priority rules into the queueing system

The picture shows not only that the introduction of priority rules cause increase (decrease) of the total number of low (high) priority items in the system, but also the scale of this increase (decrease). We see that there is already a significant impact of priority rule usage on the number of items in the system if the utilisation is only moderate (0.6-0.8). This confirms the conjecture that the appropriate use of priority rules may provide an opportunity for efficiency gain in spare part networks. We will address this issue in more detail in Chapter 8.

6.7 Conclusions and generalisations

In this chapter, we derived a method to analyse multi-class *M/M/k* priority queues with preemption and two priority groups (high and low). Each priority group can contain several classes of items with different arrival and service rates. The proposed method is based on solution of the stationary state equations. It uses an iteration algorithm. The computational effort to find good approximation depends on the number of types, number of servers and utilisation rate (since higher utilisation rate requires more iterations). For example, system with 4 subclasses, 4 servers and utilisation rate of 95% needs O(20) matrix operations, where dimension of each matrix is 238×238. Due to the increase of the size of matrices, the computational effort increases rapidly for large k, N^h and N^l. For example, in case of $k = 5$, $N^h = 2$ and $N^l = 3$, we have to deal with state space (and matrices) of dimension 1782×1782. Approximations are necessary then. As an approximation, we may replace groups of items with similar characteristic by one item with average service properties. Thereby, we answered partly research question 5 (Chapter 1), i.e. for models without outsourcing.

The presented queueing system assumes random preemption of low priority items and random resumption of the preempted items. Several other situations can be easily dealt with and the equilibrium equations for queueing systems with different preemption and resumption rules can be formulated and solved. As other preemption (resumption) rules, one can think of class-dependent preemption (resumption) probabilities or FIFO rules. However, the latter situations will lead to equilibrium equations of higher dimension than the ones presented in this chapter.

Our method can in principle be used as approximation for problems with more priority groups. This is possible iteratively due to preemption property. That is, we can estimate performance estimators for each priority group ignoring all classes with lower priorities and aggregating all classes with high priorities into one high priority group. However, in this case the dimension of the state space becomes extremely large and approximations for the problem with two priority classes are needed as well. This method has still to be explored.

Also, our algorithm can in principle be used for multi-class, multi-server priority queues where items have hyperexponential (H_x) service times. We can deal with these cases by representing each class as x classes with exponential distributed service time and adopting the performance estimators for the total number of items in the system among these x classes.

Chapter 7
An exact analysis of the multi-class *M/M/k* priority queue with outsourcing

Abstract

We consider a multi-server queueing model with two priority classes that consist of multiple item types each. The high priority items are outsourced if they cannot be served immediately upon arrival. Each item type has its own Poisson arrival and exponential service rate. We derive an exact method to estimate the steady state probabilities for both preemptive and non-preemptive priority disciplines. Based on these probabilities, we derive exact expressions for a wide range of relevant performance characteristics for each item type, such as the moment of the number of items in the queue and in the system, the expected postponement time and the fraction of items that are outsourced. We illustrate our method with some numerical examples. These results along with the results of Chapter 6 answer research question 5 of Chapter 1 completely.

This chapter is based on the paper:

"An exact analysis of the multi-class *M/M/k* priority queue with outsourcing". A. Sleptchenko, M.C. van der Heijden, A. van Harten. Submitted for publication.

7.1 Introduction

In this chapter, we consider a multi-class priority queueing model where high priority items are outsourced if they cannot be served immediately upon arrival. When repairing high priority items, delay because of lack of repair capacity may have serious consequences in terms of the availability of the installed base. An option to prevent delay is to outsource urgent (i.e. high priority) repair jobs that cannot be served immediately, even if this implies higher costs. To examine whether this is worthwhile, we need a model to determine the percentage of jobs that is outsourced under such a policy and to examine the impact on low priority items. After all, outsourcing of a part of the high priority jobs means that the effective server utilisation decreases, and thus the number of low priority items in the system decreases. As a consequence, outsourcing of a part of the high priority jobs leads to a reduced need for inventories of high *and* low priority items. Therefore, we analyse a multi-class *M/M/k* priority queue with outsourcing in this chapter. We consider both preemptive and non-preemptive priorities. We show that outsourcing high priority jobs reduces the complexity of the steady state equations. We present a method to solve the steady state equations exactly. Therefore, we are able to derive a wide range of performance characteristics, including the first two moments of the number of items in the system per item class that we need to analyse spare part networks. Our results can be used for other applications as well. For example, we can interpret outsourcing repair jobs as impatient customers that leave the system if there is no available server. Such queueing models arise in applications in computer and telecommunication systems.

Our contribution to the literature (cf. Chapter 1) is to give an exact solution for the equilibrium probabilities in an *M/M/k* priority queue with subclasses in an outsourcing situation. This model is useful for various applications, such as the analysis of spare part networks as mentioned above. In that case, we are primarily interested in the first two moments of the number of items in the system for each item class, where we include the number of outsourced items for the high priority group. Note that we can easily calculate these first two moments for high priority items if the repair times in the own repair shop and outsourced are identical, because this can be modelled as an $M/M/\infty$ queue. These first two moments are much more difficult to obtain for the low priority items. Therefore, we focus on low priority items in this chapter.

The remainder of this chapter is structured as follows. In section 7.2, we deal with the variant with non-preemptive priorities. We present the equilibrium equations for the continuous time Markov chain and we show how we can solve these equations exactly. Also, we discuss the calculation of performance measures from the steady state probabilities. In section 7.3, we do the same for the variant with preemptive priorities. We present numerical results using our methods in section 7.4, where we focus on the percentage of high priority jobs that are outsourced and the number of items in the system. We state our conclusions in section 7.5.

7.2 Non-preemptive priorities

7.2.1 Model and notations

In this section, we consider a model where customers are processed according to a non-preemptive priority rule, That is, a high priority item is outsourced if there is no server available on arrival. We denote the number of item classes with high (low) priority by N^h (N^l). High priority jobs from subclass i arrive according to a Poisson process with rate λ_i^h and low priority jobs from subclass j arrive with rate λ_j^l. The service times of the subclasses are exponentially distributed with rates μ_i^h and μ_j^l for high and low priority item classes, respectively. All servers are equal, and if multiple servers are available to process a job, each available server has an equal chance to get this job. We use ρ_i^h, ρ_j^l to denote the utilisation rates of high and low priority item classes in the system. We denote the total number of high priority items by n and the number of high priority items of type i by n_i. For low priority items, we will use the notation m and m_i respectively.

To characterise the system state, we define the following vectors:

$\overline{s}^h$ and $\overline{s}^l$	–	vectors with dimensions N^h and N^l containing the number of high and low priority items in standard (not outsourced) service per item class.
$\overline{w}^l$	–	vector with dimension N^l containing the number of low priority items in the queue per item class.

We describe the system state as a vector $(\overline{s}^h, \overline{w}^l, \overline{s}^l)$ and the corresponding state probabilities by $P_{n,m}\left(\overline{s}^h, \overline{w}^l, \overline{s}^l\right)$. We define $P\left(\overline{s}^h, \overline{w}^l, \overline{s}^l\right) = 0$ if one of the components of $\overline{s}^h, \overline{w}^l$ and/or $\overline{s}^l$ is negative. Note that the vector $\overline{w}^h$, the high priority items in queue, vanishes because all these items are outsourced if they cannot be served immediately. Further, we will use some general notation as in the previous chapter (section 6.2.1):

We describe the system state probabilities as $P\left(\overline{s}^h, \overline{w}^l, \overline{s}^l\right) = P'\left(\left|\overline{w}^l\right|, \overline{s}^h, \overline{s}^l\right) \times$ Pr{distribution of high priority items in queue over subclasses = $\overline{w}^l$ | number of high priority items in queue = $\left|\overline{w}^l\right|$}. The expression $P'\left(\left|\overline{w}^l\right|, \overline{s}^h, \overline{s}^l\right)$ describes the probability that the system is in the state $\left(\left|\overline{w}^l\right|, \overline{s}^h, \overline{s}^l\right)$. Because it holds that $\left|\overline{w}^l\right| = m - \left|\overline{s}^l\right|$, we can omit the first vector parameter and substitute $P'\left(\left|\overline{w}^l\right|, \overline{s}^h, \overline{s}^l\right)$ by the notation $P_{n,m}\left(\overline{s}^h, \overline{s}^l\right)$. Then, we can rewrite the equilibrium equations for the states with $m > k$ using the product form:

$$P\left(\overline{s}^h, \overline{w}^l, \overline{s}^l\right) = \left|\overline{w}^l\right|! \prod_{i=1}^{N^l} \frac{\left(a_i^l\right)^{w_i^l}}{w_i^l!} P_{n,m}\left(\overline{s}^h, \overline{s}^l\right) \tag{7.1}$$

where n corresponds to the number of high priority items in standard service and m to the total number of low priority items in the system.

7.2.2 Equilibrium equations

Now we define the equilibrium state equations for the continuous time Markov chain. Neighbours of a state $\left(\overline{s}^h, \overline{w}^l, \overline{s}^l\right)$ with n clients of high priority and m clients of low priority are states to/from which a one-step transition is possible, either by an item arrival or by a service completion. According to the numbers n and m of items in the system, the system states (so the equilibrium equations) can be divided into two areas (see Figure 7.1):

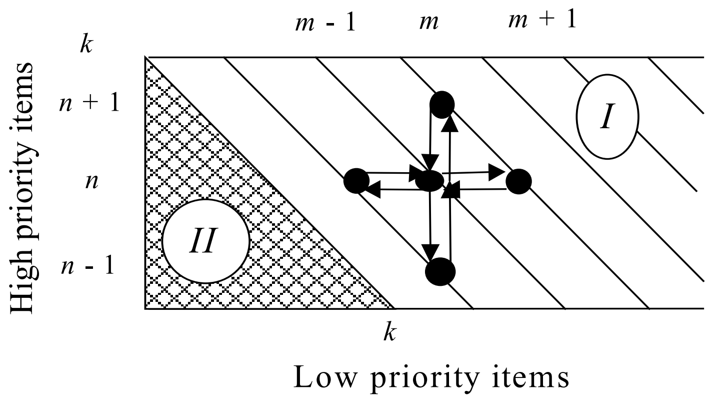

Figure 7.1 Two sets of probability states

I. all servers are busy ($n + m > k,\ n \leq k$),

II. there is at least one server available ($n + m < k$),

Note that the states with $n > k$ have zero probability, because outsourcing means that high priority jobs cannot wait in the queue. As a consequence, the analysis is simplified compared to the priority model without outsourcing as described in Chapter 6. Being outsourced, a high priority item does not influence the system anymore. A low priority item may enter service after the next job completion, even if high priority jobs are outsourced.

Let us now define the equilibrium equations for both areas as shown in Figure 7.1. Further, we separately address the equations for the boundary between the two regions ($m + n = k$).

In area I ($m + n > k,\ n \leq k$), all servers are occupied and there is no high priority item in the queue due to outsourcing. Transitions out of the state $(\overline{s}^h, \overline{s}^l)$ occur if:

(1) a low priority item arrives,

(2) a (high or low priority) item in service leaves the system.

A high priority item arriving is immediately outsourced, so it does not influence the system state. The transitions to the state $(\overline{s}^h, \overline{s}^l)$ from its neighbours are

(3) the arrival of a low priority job that enters the queue,

(4) a service completion of a (high or low priority) item type i, so a new service of a low priority item type j is started; note that subclass j is at the front of the queue with probability $\frac{w_j^l + 1}{|\overline{w}^l| + 1}$.

Then, the equilibrium equations are:

$$\begin{aligned}\left(\Lambda^l + \overline{\mu}\left(\overline{s}^h, \overline{s}^l\right)\right) P\left(\overline{s}^h, \overline{w}^l, \overline{s}^l\right) = & \sum_{i=1}^{N^l} \lambda_i^l P\left(\overline{s}^h, \overline{w}^l - \overline{e}_i^l, \overline{s}^l\right) \\ & + \sum_{i=1}^{N^h} \sum_{j=1}^{N^l} \frac{w_j^l + 1}{\left|\overline{w}^l\right| + 1} \left(s_i^h + 1\right) \mu_i^h P\left(\overline{s}^h + \overline{e}_i^h, \overline{w}^l + \overline{e}_j^l, \overline{s}^l - \overline{e}_j^l\right) \\ & + \sum_{i=1}^{N^l} \sum_{j=1}^{N^l} \frac{w_j^l + 1}{\left|\overline{w}^l\right| + 1} \left(s_i^l + 1 - e_{ij}^l\right) \mu_i^l P\left(\overline{s}^h, \overline{w}^l + \overline{e}_j^l, \overline{s}^l + \overline{e}_i^l - \overline{e}_j^l\right)\end{aligned} \tag{7.2}$$

To reduce the dimension of the state space, we divide these equations by $k\mu$ and replace $P\left(\overline{s}^h,\overline{w}^l,\overline{s}^l\right)$ by expression (7.1). This leads us to

$$\begin{aligned}\left(1+\gamma\rho^l+\overline{\delta}\left(\overline{s}^h,\overline{s}^l\right)\right)P_{n,m}\left(\overline{s}^h,\overline{s}^l\right) &= \gamma\rho^1 P_{n,m-1}\left(\overline{s}^h,\overline{s}^l\right)\\ &+\frac{1}{k}\sum_{i=1}^{N^h}\sum_{j=1}^{N^l}\left(s_i^h+1\right)\left(1+\delta_i^h\right)a_i^l P_{n+1,m}\left(\overline{s}^h+\overline{e}_i^h,\overline{s}^l-\overline{e}_j^{\,l}\right)\\ &+\frac{1}{k}\sum_{i=1}^{N^l}\sum_{j=1}^{N^l}\left(s_i^l+1-e_{ij}^{\,l}\right)\left(1+\delta_i^l\right)a_i^l P_{n,m+1}\left(\overline{s}^h,\overline{s}^l+\overline{e}_i^l-\overline{e}_j^{\,l}\right)\end{aligned}$$

These equations can be rewritten in vector form as:

$$\mathbb{D}\mathbb{P}_t=\gamma\rho^l\mathbb{P}_{t-1}+\mathbb{G}\mathbb{P}_{t+1} \tag{7.3}$$

In this equation, the matrices $\mathbb{D}$ and $\mathbb{G}$ are defined by

$$\mathbb{D}\mathbb{P}_t\left[\overline{s}^h,\overline{s}^l\right]=\left(1+\gamma\rho^l+\overline{\delta}\left(\overline{s}^h,\overline{s}^l\right)\right)\mathbb{P}_t\left[\overline{s}^h,\overline{s}^l\right], \qquad t>k$$

$$\begin{aligned}\mathbb{G}\mathbb{P}_{t+1}\left[\overline{s}^h,\overline{s}^l\right]&=\frac{1}{k}\sum_{i=1}^{N^h}\sum_{j=1}^{N^l}\left(s_i^h+1\right)\left(1+\delta_i^h\right)\mathbb{P}_{t+1}\left[\overline{s}^h+\overline{e}_i^h,\overline{s}^l-\overline{e}_j^{\,l}\right]\\ &+\frac{1}{k}\sum_{i=1}^{N^l}\sum_{j=1}^{N^l}a_j^l\left(s_i^l+1-e_{ij}^{\,l}\right)\left(1+\delta_i^l\right)\mathbb{P}_{t+1}\left[\overline{s}^h,\overline{s}^l+\overline{e}_i^l-\overline{e}_j^{\,l}\right], \qquad t>k\end{aligned}$$

The dimension of the vector $\mathbb{P}_t$ for $t>k$ equals

$$\dim\left(\mathbb{P}_t\right)=\frac{\left(N^h+N^l+k-1\right)!}{\left(N^h+N^l-1\right)!k!}, \quad t>k$$

In **area II** ($m + n < k$), there is no queue, and so the vector $\overline{w}^l$ is equal to zero. Transitions out of the state ($\overline{s}^h,\overline{s}^l$) occur if:

(1) a (high or low priority) item arrives,

(2) a (high or low priority) item in service leaves the system.

The transitions to the state ($\overline{s}^h,\overline{s}^l$) from its neighbours are:

(3) the arrival of a (high or low priority) job that enters a free server,

(4) the service completion of a (high or low priority) item type *i*.

Then, the equilibrium equations are:

$$\left(\Lambda^h+\Lambda^l+\overline{\mu}\left(\overline{s}^h,\overline{s}^l\right)\right)P\left(\overline{s}^h,0,\overline{s}^l\right)=\sum_{i=1}^{N^h}\lambda_i^h P\left(\overline{s}^h-\overline{e}_i^h,0,\overline{s}^l\right)+\sum_{i=1}^{N^l}\lambda_i^l P\left(\overline{s}^h,0,\overline{s}^l-\overline{e}_i^l\right)$$
$$+\sum_{i=1}^{N^h}\left(s_i^h+1\right)\mu_i^h P\left(\overline{s}^h+\overline{e}_i^h,0,\overline{s}^l\right)+\sum_{i=1}^{N^l}\left(s_i^l+1\right)\mu_i^l P\left(\overline{s}^h,0,\overline{s}^l+\overline{e}_i^l\right)$$

These equations can be rewritten in a matrix form as:

$$\mathbb{D}_t\mathbb{P}_t=\mathbb{F}_t\mathbb{P}_{t-1}+\mathbb{G}_t\mathbb{P}_{t+1} \tag{7.4}$$

where the matrices $\mathbb{D}_t$, $\mathbb{F}_t$ and $\mathbb{G}_t$ for $t\le k$ are defined by

$$\mathbb{D}_t\mathbb{P}_t\left[\overline{s}^h,\overline{s}^l\right]=\left(\tfrac{t}{k}+\rho^h+\gamma\rho^l+\overline{\delta}\left(\overline{s}^h,\overline{s}^l\right)\right)\mathbb{P}_t\left[\overline{s}^h,\overline{s}^l\right]$$

$$\mathbb{F}_t\mathbb{P}_{t-1}\left[\overline{s}^h,\overline{s}^l\right]=\rho^h\sum_{i=1}^{N^h}\mathbb{P}_{t-1}\left[\overline{s}^h-\overline{e}_i^h,\overline{s}^l\right]+\gamma\rho^l\sum_{i=1}^{N^h}\mathbb{P}_{t-1}\left[\overline{s}^h,\overline{s}^l-\overline{e}_j^{\,l}\right]$$

$$\mathbb{G}_t\mathbb{P}_{t+1}\left[\overline{s}^h,\overline{s}^l\right]=\frac{1}{k}\sum_{i=1}^{N^h}\left(s_i^h+1\right)\left(1+\delta_i^h\right)\mathbb{P}_{t+1}\left[\overline{s}^h+\overline{e}_i^h,\overline{s}^l\right]$$
$$+\frac{1}{k}\sum_{i=1}^{N^l}\left(s_i^l+1\right)\left(1+\delta_i^l\right)\mathbb{P}_{t+1}\left[\overline{s}^h,\overline{s}^l+\overline{e}_i^l\right]$$

The dimension of the vector $\mathbb{P}_t$ is equal to

$$\dim\left(\mathbb{P}_t\right)=\frac{\left(N^h+N^l+t-1\right)!}{\left(N^h+N^l-1\right)!t!},\quad t\le k$$

Finally, we specify the equations for the **border between area I and area II** ($m+n = k$). Then, all servers are occupied and the queue is empty. Transitions out of the state ($\overline{s}^h, 0, \overline{s}^l$) occur if

(1) a low priority item arrives,

(2) a (high or low priority) item in service leaves the system.

Similarly to area II, a high priority item arriving is immediately outsourced, so it does not influence the system state. The transitions from the neighbours of the state ($\overline{s}^h, 0, \overline{s}^l$) are:

(3) the arrival of a (low or high) priority job that enters service,

(4) a service completion of a (high or low priority) item type i while there is one low priority item type j in queue, so a new service of that low priority item is started.

Now, the equilibrium equations are:

$$\begin{aligned}\left(\Lambda^l+\overline{\mu}\left(\overline{s}^h,\overline{s}^l\right)\right)P\left(\overline{s}^h,0,\overline{s}^l\right)=&\sum_{i=1}^{N^h}\lambda_i^h P\left(\overline{s}^h-\overline{e}_i^h,0,\overline{s}^l\right)+\sum_{j=1}^{N^l}\lambda_j^l P\left(\overline{s}^h,0,\overline{s}^l-\overline{e}_j^l\right)\\+\sum_{i=1}^{N^h}\sum_{j=1}^{N^l}\left(s_i^h+1\right)\mu_i^h P\left(\overline{s}^h+\overline{e}_i^h,\overline{e}_j^l,\overline{s}^l-\overline{e}_j^l\right)&+\sum_{i=1}^{N^l}\sum_{j=1}^{N^l}\left(s_i^l+1-e_{ij}^l\right)\mu_i^l P\left(\overline{s}^h,\overline{e}_j^l,\overline{s}^l+\overline{e}_i^l-\overline{e}_j^l\right)\end{aligned} \tag{7.5}$$

This equations can be written in matrix form as:

$$\mathbb{D}\mathbb{P}_k=\mathbb{F}_k\mathbb{P}_{k-1}+\mathbb{G}\mathbb{P}_{k+1} \tag{7.6}$$

where the matrices $\mathbb{D}$, $\mathbb{G}$ and $\mathbb{F}_k$ are equal to those introduced before.

7.2.3 Solution of the equilibrium equations

The matrix equilibrium equation (7.3) can be solved via the quadratic eigenvalue equation

$$z^2\gamma\rho^l=z\mathbb{D}-\mathbb{G} \tag{7.7}$$

as it is described in Chapter 3. In this case, the vector $\mathbb{P}_t$ can be written in product form as

$$\mathbb{P}_t = \mathbb{Z}^{-(t-k)}\mathbb{P}_k$$

where $\mathbb{Z} = \Xi\Omega\Xi^{-1}$, Ω is the matrix containing the eigenvalues of equation (7.7) that exceed 1 and Ξ is the matrix containing the eigenvectors corresponding to these eigenvalues. The matrix $\mathbb{G}$ in this equation has a special structure, since in its definition the terms

$$\frac{1}{k}\sum_{i=1}^{N^l}\sum_{j=1}^{N^l} a_j{}^l\left(s_i{}^l + 1 - e_{ij}{}^l\right)\left(1+\delta_i{}^l\right)\mathbb{P}_{t+1}\left[\overline{s}^h, \overline{s}^l + \overline{e}_i^l - \overline{e}_j{}^l\right]$$

refer to the same $\overline{s}^h$ and the terms

$$\frac{1}{k}\sum_{i=1}^{N^h}\sum_{j=1}^{N^l}\left(s_i{}^h + 1\right)\left(1+\delta_i{}^h\right)\mathbb{P}_{t+1}\left[\overline{s}^h + \overline{e}_i{}^h, \overline{s}^l - \overline{e}_j{}^l\right]$$

refer to the "upper triangular" indices with $\left|\overline{s}^h + e_i^h\right| = \left|\overline{s}^h\right| + 1$. Then, due to the fact that the matrix $\mathbb{D}$ is diagonal, the eigenvalue equations decompose with respect to each $\overline{s}^h$. Each of these decomposed equations has the same structure as the equations for the multi-class queue without priorities (cf. Chapter 3). That is, we can apply the same theory and thus we obtain exactly dim($\mathbb{P}_t$) real eigenvalues greater than 1.

The equations (7.4) for $i \le k$ have a different dimension for different t and they can be solved recursively as in the previous sections. That is:

$$\mathbb{P}_t = \mathbb{Q}_t\mathbb{P}_{t-1}$$

where $\mathbb{Q}_t = \left(\mathbb{D}_t - \mathbb{G}_t\mathbb{Q}_{t+1}\right)^{-1}\mathbb{F}_t$ and $\mathbb{Q}_k = Z^{-1}\mathbb{F}_k$. We find the probability of an empty system P_0 from $\sum_{\overline{s}^h,\overline{w}^l,\overline{s}^l} P\left(\overline{s}^h, \overline{w}^l, \overline{s}^l\right) = 1$, which yields

$$P_0 = \left[1 + \left\langle \mathbf{1}_1, \mathbb{Q}_1 \right\rangle + \cdots + \left\langle \mathbf{1}_{k-1}, \mathbb{Q}_{k-1}\cdots\mathbb{Q}_1 \right\rangle + \left\langle \mathbf{1}_k, \left(\mathbf{I} - \mathbb{Z}^{-1}\right)^{-1}\mathbb{Q}_k\cdots\mathbb{Q}_1 \right\rangle\right]^{-1}$$

Here $\mathbf{1}_t$ is a vector of dimension

$$\dim(\mathbf{1}_t) = \frac{\left(N^h + N^l + t - 1\right)!}{\left(N^h + N^l - 1\right)!t!} \quad t \le k$$

with all components equal to 1. For $t > k$, we introduce $\mathbf{1}_{t>k} = \mathbf{1}_k$.

7.2.4 Performance measures

Since we can now calculate any system state probability exactly, we can also find the exact value of any related performance measure. As examples, we discuss the probability of high priority items to be outsourced, the mean number of low priority items in the queue and the first two moments of the number of low priority items in the system. Note that we can easily calculate the first two moments of the number of high priority items in the system (in own repair and outsourced) if the service times for own and outsourced repair are identical. Then the high priority items behave as in a *M/M/∞* queue, and so we can use the well-known result that the number of high priority items of type *i* in the *M/G/∞* queue is Poisson distributed with mean λ_i / μ_i, see Palm (1938). If the own repair rate and outsourced repair rate are different, we can still simply calculate the throughput time distribution and the mean number of high priority items in the system from the steady state probabilities. However, it is not straightforward to derive the second moment of the number of high priority items in the system, because the number of high priority items in own repair and being outsourced are correlated.

The **outsourcing probability** is the probability that an arriving high priority item finds all servers busy. This probability can be expressed as follows:

$$P_{iout} = a_i^h \left\langle \mathbf{1}_k, \left(\mathbf{I} - \mathbb{Z}^{-1}\right)^{-1} \mathbb{P}_k \right\rangle$$

The **mean number** of low priority items of subclass *i* **in queue** (area I) can be estimated via the total number of items in queue as follows:

$$E\left[q_i^l\right]=\sum_{n=0}^{k}\sum_{m=k-n}^{\infty}\sum_{\overline{s}^h,\overline{w}^l,\overline{s}^l} w_i^l\left|\overline{w}^l\right|!\prod_{j=1}^{N^l}\frac{\left(a_j^l\right)^{w_j^l}}{w_j^l!}P_{n,m}\left(\overline{s}^h,\overline{s}^l\right)$$
$$=a_i^l\sum_{n=0}^{k}\sum_{m=k-n}^{\infty}(m-n)\left(\sum_{j=1}^{N^l}a_j^l\right)^{m-n}\sum_{\overline{s}^h,\overline{s}^l}P_{n,m}\left(\overline{s}^h,\overline{s}^l\right)=a_i^l E\left[q^l\right]$$

Then, we can write down the mean total number of low priority items in queue as

$$E\left[q^l\right]=\sum_{n=0}^{k}\sum_{m=k-n}^{\infty}(m-n)\sum_{\substack{\overline{s}^h,\overline{w}^l,\overline{s}^l\\ \text{s.t. }\left|\overline{w}^l\right|=m-n}}\left|\overline{w}^l\right|!\prod_{j=1}^{N^l}\frac{\left(a_j^l\right)^{w_j^l}}{w_j^l!}P_{n,m}\left(\overline{s}^h,\overline{s}^l\right)$$
$$=\sum_{n=0}^{k}\sum_{m=k-n}^{\infty}(m-n)\left(\sum_{j=1}^{N^l}a_j^l\right)^{m-n}\sum_{\overline{s}^h,\overline{s}^l}P_{n,m}\left(\overline{s}^h,\overline{s}^l\right)=\sum_{t=k+1}^{\infty}(t-k)\left\langle\mathbf{1}_k,\mathbb{Z}^{-(t-k)}\mathbb{P}_k\right\rangle$$
$$=\left\langle\mathbf{1}_k,\left(\mathbf{I}-\mathbb{Z}^{-1}\right)^{-2}\mathbb{Z}^{-1}\mathbb{P}_k\right\rangle$$

The mean number of items of subclass *i* in queue is equal to

$$E\left[q_i^l\right]=a_i^l\left\langle\mathbf{1}_k,\left(\mathbf{I}-\mathbb{Z}^{-1}\right)^{-2}\mathbb{Z}^{-1}\mathbb{P}_k\right\rangle$$

The **mean number** of items of subclass *i* **in the system**, denoted by R_i^l, equals the mean number of items in queue plus the mean number of items in service λ_i^l/μ_i^l, since we deal with non-preemptive priorities. Then, the mean number of items in the system is:

$$E\left[R_i^l\right]=a_i^l\left\langle\mathbf{1}_k,\left(\mathbf{I}-\mathbb{Z}^{-1}\right)^{-2}\mathbb{Z}^{-1}\mathbb{P}_k\right\rangle+\frac{\lambda_i^l}{\mu_i^l}$$

The **second moment** of R_i^l can be found in a similar way (cf. Chapter 6):

$$E\left[\left(R_i^l\right)^2\right]=\sum_{w_i^l,s_i^l}\left(w_i^l+s_i^l\right)^2 P_2\left(w_i^l,s_i^l\right)+\sum_{s_i^l}\left(s_i^l\right)^2 P_3\left(s_i^l\right)$$

We calculate the first sum as:

$$\begin{aligned}\sum_{w_i^l,s_i^l}\left(w_i^l+s_i^l\right)^2 P_2\left(w_i^l,s_i^l\right) &= \sum_{w_i^l,s_i^l}\left(w_i^l\right)^2 P_2\left(w_i^l,s_i^l\right)+\sum_{w_i^l,s_i^l}2s_i^l w_i^l P_2\left(w_i^l,s_i^l\right)+\sum_{w_i^l,s_i^l}\left(s_i^l\right)^2 P_2\left(w_i^l,s_i^l\right)\\ &= 2\left(a_i^l\right)^2 a_i^l\left\langle \mathbf{1}_k,\left(\mathbf{I}-\mathbb{Z}^{-1}\right)^{-3}\mathbb{Z}^{-2}\mathbb{P}_k\right\rangle + a_i^l\left\langle \mathbf{1}_k,\left(\mathbf{I}-\mathbb{Z}^{-1}\right)^{-2}\mathbb{Z}^{-1}\mathbb{P}_k\right\rangle\\ &\quad + 2a_i^l\left\langle \chi_k^{s_i^l},\left(\mathbf{I}-\mathbb{Z}^{-1}\right)^{-2}\mathbb{Z}^{-1}\mathbb{P}_k\right\rangle + \left\langle \chi_k^{\left(s_i^l\right)^2},\left(\mathbf{I}-\mathbb{Z}^{-1}\right)^{-1}\mathbb{Z}^{-1}\mathbb{P}_k\right\rangle\end{aligned}$$

Here we use $\chi_{t>k}^{s_i^l}$ for a vector with the same dimension as the vector $\mathbb{P}_k$ and with components equal to the numbers of items in the service in the corresponding system states. The second sum is

$$\sum_{s_i^l}\left(s_i^l\right)^2 P_3\left(s_i^l\right)=\sum_{t=1}^{k}\left\langle \chi_t^{\left(s_i^l\right)^2},\mathbb{P}_t\right\rangle$$

This give us an expression for the second moment of R_i^l:

$$\begin{aligned}E\left[\left(R_i^l\right)^2\right] &= 2\left(a_i^l\right)^2 a_i^l\left\langle \mathbf{1}_{t>k},\left(\mathbf{I}-\mathbb{Z}^{-1}\right)^{-3}\mathbb{Z}^{-2}\mathbb{P}_k\right\rangle + a_i^l\left\langle \mathbf{1}_{t>k},\left(\mathbf{I}-\mathbb{Z}^{-1}\right)^{-2}\mathbb{Z}^{-1}\mathbb{P}_k\right\rangle\\ &\quad + 2a_i^l\left\langle \chi_{t>k}^{s_i^l},\left(\mathbf{I}-\mathbb{Z}^{-1}\right)^{-2}\mathbb{Z}^{-1}\mathbb{P}_k\right\rangle + \left\langle \chi_{t>k}^{\left(s_i^l\right)^2},\left(\mathbf{I}-\mathbb{Z}^{-1}\right)^{-1}\mathbb{Z}^{-1}\mathbb{P}_k\right\rangle + \sum_{t=1}^{k}\left\langle \chi_t^{\left(s_i^l\right)^2},\mathbb{P}_t\right\rangle\end{aligned}$$

7.3 Preemptive priorities

7.3.1 Model

In this section, we discuss a variant with preemptive priorities of the model in the previous section. This is also a variant of the model without outsourcing as presented in Chapter 6. Compared to the latter model, outsourcing reduces the state space, and therefore we find a simpler way to solve the equilibrium equations.

In the case of preemptive priorities, we extend the system state description from the previous section with a vector of postponed low priority items, $\overline{r}^l$. We need this vector, because item types with a longer processing time will be withdrawn (postponed) more often, and so their throughput times are affected more than those of item types with a shorter processing time. So, we describe the system state by a set of vectors $\overline{s}^h, \overline{w}^l, \overline{s}^l, \overline{r}^l$ that contain information on the number of items of each class in standard (not outsourced) service ($\overline{s}^h, \overline{s}^l$), the number of postponed items ($\overline{r}^l$) and the number of items in queue ($\overline{w}^l$) for each item type. Analogously to the previous section, we describe the state probabilities as

$$P\left(\overline{s}^h, \overline{w}^l, \overline{s}^l, \overline{r}^l\right) = \left|\overline{w}^l\right|! \prod_{i=1}^{N^l} \frac{\left(a_i^{\,l}\right)^{w_i^l}}{w_i^{\,l}\,!} P_{n,m}\left(\overline{s}^h, \overline{s}^l, \overline{r}^l\right) \tag{7.8}$$

7.3.2 Equilibrium equations

Again, we discuss the equilibrium equations per area as shown in Figure 7.1.

In area I ($n \le k$, $m + n > k$), all servers are occupied and there is no high priority item in the queue. Transitions out of the state ($\overline{s}^h, \overline{w}^l, \overline{s}^l, \overline{r}^l$) occur if

(1) a low priority item arrives that enters the queue,

(2) a high priority item arrives that can enter service immediately, thereby postponing a low priority item in service (only if $n < k$, otherwise the high priority item is immediately outsourced and so the system state does not change),

(3) a (high or low priority) item in service leaves the system.

The transitions to the state ($\overline{s}^h, \overline{w}^l, \overline{s}^l, \overline{r}^l$) from its neighbours are:

(4) the arrival of a low priority job that enters the queue,

(5) the arrival of a high priority job that preempts one of the low priority items with probability $\frac{s_j^{\,l}+1}{\left|\overline{s}^l\right|+1}$ and that enters the service,

(6) the service completion of a (high or low priority) item type i, so a new service of a low priority item type j is started; note that subclass j is at the front of the service with probability $\frac{w_j^{\,l}+1}{\left|\overline{w}^l\right|+1}$ or $\frac{r_j^{\,l}+1}{\left|\overline{r}^l\right|+1}$ if there are postponed items.

Then, we can write the equilibrium equations in vector form:

$$\mathbb{D}\mathbb{P}_t = \mathbb{F}\mathbb{P}_{t-1} + \mathbb{G}\mathbb{P}_{t+1} \tag{7.9}$$

In this equation, the matrices $\mathbb{D}_t$, $\mathbb{F}_t$ and $\mathbb{G}_t$ are defined by

$$\mathbb{D}\mathbb{P}_t\left[\bar{s}^h,\bar{s}^l,\bar{r}^l\right] = \left(1 + I\left(\left|\bar{s}^h\right| < k\right)\rho^h + \gamma\rho^l + \bar{\delta}\left(\bar{s}^h,\bar{s}^l\right)\right)\mathbb{P}_t\left[\bar{s}^h,\bar{s}^l,\bar{r}^l\right]$$

$$\mathbb{F}\mathbb{P}_t\left[\bar{s}^h,\bar{s}^l,\bar{r}^l\right] = \gamma\rho^l\mathbb{P}_{t-1}\left[\bar{s}^h,\bar{s}^l,\bar{r}^l\right] + \rho^h\sum_{i=1}^{N^h}\sum_{j=1}^{N^l}\frac{s_j^{\,l}+1}{\left|\bar{s}^l\right|+1}a_i^{\,h}\mathbb{P}_{t-1}\left[\bar{s}^h-\bar{e}_i^{\,h},\bar{s}^l+\bar{e}_j^{\,l},\bar{r}^l-\bar{e}_j^{\,l}\right]$$

$$\begin{aligned}\mathbb{G}\mathbb{P}_t\left[\bar{s}^h,\bar{s}^l,\bar{r}^l\right] = {} & \frac{1}{k}\sum_{i=1}^{N^h}\sum_{j=1}^{N^l}\frac{r_j^{\,l}+1}{\left|\bar{r}^l\right|+1}\left(s_i^{\,h}+1\right)\left(1+\delta_i^{\,h}\right)\mathbb{P}_{t+1}\left[\bar{s}^h+\bar{e}_i^{\,h},\bar{s}^l-\bar{e}_j^{\,l},\bar{r}^l+\bar{e}_j^{\,l}\right] \\ & + \frac{1}{k}\sum_{i=1}^{N^h}\sum_{j=1}^{N^l}\left(s_i^{\,h}+1\right)\left(1+\delta_i^{\,h}\right)a_j^{\,l}\mathbb{P}_{t+1}\left[\bar{s}^h+\bar{e}_i^{\,h},\bar{s}^l-\bar{e}_j^{\,l},0\right] \\ & + \frac{1}{k}\sum_{i=1}^{N^l}\sum_{j=1}^{N^l}\frac{r_j^{\,l}+1}{\left|\bar{r}^l\right|+1}\left(s_i^{\,l}+1-e_{ij}^{\,l}\right)\left(1+\delta_i^{\,l}\right)\mathbb{P}_{t+1}\left[\bar{s}^h,\bar{s}^l+\bar{e}_i^{\,l}-\bar{e}_j^{\,l},\bar{r}^l+\bar{e}_j^{\,l}\right] \\ & + \frac{1}{k}\sum_{i=1}^{N^l}\sum_{j=1}^{N^l}\left(s_i^{\,l}+1-e_{ij}^{\,l}\right)\left(1+\delta_i^{\,l}\right)a_j^{\,l}\mathbb{P}_{t+1}\left[\bar{s}^h,\bar{s}^l+\bar{e}_i^{\,l}-\bar{e}_j^{\,l},0\right]\end{aligned}$$

Here $I(x)$ is the indicator function that equals 1 if x is true and 0 otherwise. The dimension of the vector $\mathbb{P}_t$ for $t > k$ does not depend on t. It can be derived from the number of all combinations of $N^h + N^l$ items on k servers including all combinations of j postponed items, $j \le n \le k$, i.e.:

$$\dim\left(\mathbb{P}_t\right) = \sum_{i=0}^{k}\left[\sum_{n=i}^{k}\binom{N^h+n-1}{n}\binom{N^l+k-n-1}{k-n}\right]\binom{N^h+i-1}{i}, \qquad t > k$$

In **area II** ($n \le k$, $m + n < k$), the queue is empty. So, the vectors $\bar{w}^l$ and $\bar{r}^l$ are equal to zero. Transitions out of the state ($\bar{s}^h,0,\bar{s}^l,0$) occur if

(1) a (low or high priority) item arrives that enters service immediately,

(2) a (high or low priority) item in service leaves the system.

Transitions to the state ($\overline{s}^h, 0, \overline{s}^l, 0$) from its neighbours are specified by

(1) the arrival of a (high or low priority) job that enters a free server,

(2) the service completion of a (high or low priority) item type *i*.

Now, the equilibrium equations are:

$$\mathbb{D}_t \mathbb{P}_t = \mathbb{F}_t \mathbb{P}_{t-1} + \mathbb{G}_t \mathbb{P}_{t+1} \tag{7.10}$$

where the matrices $\mathbb{D}_t$, $\mathbb{F}_t$ and $\mathbb{G}_t$ for $t \le k$ are given by

$$\mathbb{D}_t \mathbb{P}_t\left[\overline{s}^h, \overline{s}^l, 0\right] = \left(\tfrac{t}{k} + \rho^h + \gamma\rho^l + \overline{\delta}\left(\overline{s}^h, \overline{s}^l\right)\right)\mathbb{P}_t\left[\overline{s}^h, \overline{s}^l, 0\right]$$

$$\mathbb{F}_t \mathbb{P}_{t-1}\left[\overline{s}^h, \overline{s}^l, 0\right] = \rho^h \sum_{i=1}^{N^h} \mathbb{P}_{t-1}\left[\overline{s}^h - \overline{e}_i^h, \overline{s}^l, 0\right] + \gamma\rho^l \sum_{i=1}^{N^h} \mathbb{P}_{t-1}\left[\overline{s}^h, \overline{s}^l - \overline{e}_j^l, 0\right]$$

$$\begin{aligned}\mathbb{G}_t \mathbb{P}_{t+1}\left[\overline{s}^h, \overline{s}^l, 0\right] = {} & \frac{1}{k}\sum_{i=1}^{N^h}\left(s_i^h + 1\right)\left(1 + \delta_i^h\right)\mathbb{P}_{t+1}\left[\overline{s}^h + \overline{e}_i^h, \overline{s}^l, 0\right] \\ & + \frac{1}{k}\sum_{i=1}^{N^l}\left(s_i^l + 1\right)\left(1 + \delta_i^l\right)\mathbb{P}_{t+1}\left[\overline{s}^h, \overline{s}^l + \overline{e}_i^l, 0\right]\end{aligned}$$

The dimension of the vector $\mathbb{P}_t$ is equal to

$$\dim\left(\mathbb{P}_t\right) = \frac{\left(N^h + N^l + t - 1\right)!}{\left(N^h + N^l - 1\right)!\,t!}, \quad t \le k$$

Finally, we specify the equations for the **border between area I and area II** ($m + n = k$). Then, all servers are occupied and the queue is empty. Transitions out of ($\overline{s}^h, 0, \overline{s}^l, 0$) occur if:

(1) a low priority item arrives that enters the queue,

(2) a high priority item arrives that can enter service immediately, thereby postponing a low priority item in service (only if $n < k$, otherwise the high priority item is immediately outsourced and so the system state does not change),

(3) a (high or low priority) item in service leaves the system.

The transitions to the state ($\overline{s}^h, 0, \overline{s}^l, 0$) from its neighbours are:

(1) the arrival of a (low or high) priority job that enters service immediately,

(2) a service completion of a (high or low priority) item type i while there is one low priority item type j is in queue, so a new service of that low priority item is started.

Now the equilibrium equations are:

$$\mathbb{D}\mathbb{P}_k = \mathbb{F}_k\mathbb{P}_{k-1} + \mathbb{G}\mathbb{P}_{k+1} \tag{7.11}$$

with $\mathbb{D}$, $\mathbb{G}$ and $\mathbb{F}_k$ as before.

7.3.3 Solution of equilibrium equations

Again, the matrix equilibrium equation (7.9) can be solved via the quadratic eigenvalue equation

$$z^2\mathbb{F} = z\mathbb{D} - \mathbb{G}, \tag{7.12}$$

as in the previous section. In this case, the vector $\mathbb{P}_t$ can be written in product form as

$$\mathbb{P}_t = \mathbb{Z}^{-(t-k)}\mathbb{P}_k$$

where $\mathbb{Z} = \Xi\Omega\Xi^{-1}$ with Ω a matrices containing the eigenvalues of the equation (7.12) that are greater than 1 and Ξ a matrix containing the eigenvectors corresponding to those eigenvalues.

This equation has a similar structure to the one for multi-class queue without priority rules (Chapter 3). That is, we can apply the same theory and we obtain that they have exactly dim($\mathbb{P}_t$) real eigenvalues greater than 1.

The equations (7.10) for $t \leq k$ have different dimensions for different t and can be solved recursively as in the previous sections. That is:

$$\mathbb{P}_t = \mathbb{Q}_t\mathbb{P}_{t-1}$$

where $\mathbb{Q}_t = \left(\mathbb{D}_t - \mathbb{G}_t \mathbb{Q}_{t+1}\right)^{-1} \mathbb{F}_t$ and $\mathbb{Q}_k = \mathbb{Z}^{-1}\mathbb{F}_k$. The probability of an empty system P_0 is found from $\sum_{\vec{s}^h, \vec{w}^l, \vec{s}^l} P\left(\vec{s}^h, \vec{w}^l, \vec{s}^l\right) = 1$, which yields

$$P_0 = \left[1 + \langle \mathbf{1}_1, \mathbb{Q}_1 \rangle + \cdots + \langle \mathbf{1}_{k-1}, \mathbb{Q}_{k-1} \cdots \mathbb{Q}_1 \rangle + \left\langle \mathbf{1}_k, \left(\mathbf{I} - \mathbb{Z}^{-1}\right)^{-1} \mathbb{Q}_k \cdots \mathbb{Q}_1 \right\rangle \right]^{-1}$$

where $\mathbf{1}_t$ is a vector with all components equal to 1, as before.

7.3.4 Performance measures

Now that we have the exact state probabilities, we can also derive a variety of performance measures exactly. Similar to section 7.2, we discuss the outsourcing probability for high priority items, the mean number of low priority items in queue and the first two moments of the number of low priority items in the system. For the number of high priority items in the system, we can use the result for the *M/G/∞* queue again if the service times for own and outsourced repair are identical, see section 7.2.4.

The **probability to be outsourced** is the probability that the arriving high priority item finds all servers busy. This probability can be expressed as follows:

$$P_{i_{out}} = a_i^h \left\langle \mathbf{1}_k, \left(\mathbf{I} - \mathbb{Z}^{-1}\right)^{-1} \mathbb{P}_k \right\rangle$$

The **mean number** of items of subclass *i* **in queue** (area II) can be estimated via the total number of items in queue as follows:

$$E\left[q_i^l\right] = a_i^l \left\langle \mathbf{1}_{t>k}, \left(\mathbf{I} - \mathbb{Z}^{-1}\right)^{-2} \mathbb{Z}^{-1} \mathbb{P}_k \right\rangle$$

The **mean number** of items of subclass *i* **in the system** equals the mean number of items in queue, in service and postponed. The number of postponed items can be derived as in section 7.2.4 using the vector $\chi_k^{r_i^l}$:

$$E\left[PS_i^l\right]=\left\langle \chi_k^{r_i^l},\left(\mathbf{I}-\mathbb{Z}^{-1}\right)^{-1}\mathbb{Z}^{-1}\mathbb{P}_k\right\rangle$$

Here $\chi_k^{r_i^l}$ is a vector with the same dimension as vector $\mathbb{P}_k$ and with components equal to the numbers of postponed items in the corresponding system states.

Together with the mean number of low priority items in service (λ_i^l/μ_i^l), this gives us an expression for the mean total number of items in the system:

$$E\left[R_i^l\right]=a_i^l\left\langle \mathbf{1}_k,\left(\mathbf{I}-\mathbb{Z}^{-1}\right)^{-2}\mathbb{Z}^{-1}\mathbb{P}_k\right\rangle+\left\langle \chi_k^{r_i^l},\left(\mathbf{I}-\mathbb{Z}^{-1}\right)^{-1}\mathbb{Z}^{-1}\mathbb{P}_k\right\rangle+\frac{\lambda_i^l}{\mu_i^l}$$

The **second moment** of the total number of low priority items of type *i* **in the system,** we now find to be

$$\begin{aligned}E\left[\left(R_i^l\right)^2\right]&=2\left(a_i^l\right)^2\left\langle \mathbf{1}_k,\left(I-\mathbb{Z}^{-1}\right)^{-3}\mathbb{Z}^{-2}\mathbb{P}_k\right\rangle+a_i^l\left\langle \mathbf{1}_k,\left(\mathbf{I}-\mathbb{Z}^{-1}\right)^{-2}\mathbb{Z}^{-1}\mathbb{P}_k\right\rangle\\&+2a_i^l\left\langle \chi_k^{s_i^l+r_i^l},\left(\mathbf{I}-\mathbb{Z}^{-1}\right)^{-2}\mathbb{Z}^{-1}\mathbb{P}_k\right\rangle+\left\langle \chi_k^{\left(s_i^l+r_i^l\right)^2},\left(\mathbf{I}-\mathbb{Z}^{-1}\right)^{-1}\mathbb{Z}^{-1}\mathbb{P}_k\right\rangle+\sum_{t=1}^{k}\left\langle \chi_t^{s_i^l},\mathbb{P}_t\right\rangle\end{aligned}$$

7.4 Numerical results

The equations presented in the previous sections are exact. Still, we checked the validity of our results by comparison to discrete simulation events. We found no significant deviations.

In this numerical section, we first examine the impact of outsourcing in section 7.4.1. We do this by comparison of model with preemptive priorities and outsourcing (section 7.3) to the model without outsourcing (Chapter 6). We do not examine the impact of outsourcing for the case with non-preemptive priorities, because as far as we know there are no results for the equivalent model without outsourcing, i.e. when the high or low priority class consists of multiple item subclasses. In section 7.4.2, we investigate the difference between preemptive and non-preemptive priorities in our model with outsourcing.

7.4.1 Impact of outsourcing of high priority items

We consider a queueing system with three servers and three item types. One of the items has high priority and two have low priority. The total utilisation rate ρ is fixed to 0.95. We vary three parameters: ρ^h, $\gamma(=\mu^l/\mu^h)$ and μ_1^l/μ_2^l. We choose the values 0.5, 1 and 2 for both γ and μ_1^l/μ_2^l and 0.2 and 0.6 for ρ^h, thereby obtaining 3×3×2 = 18 model runs. The arrival fractions within the group of low priority items are fixed and equal to $a_1^l = 0.3$ and $a_2^l = 0.7$. The values of δ_1^l and δ_2^l are completely defined by the other parameters.

In Table 7.1, we show the outsourcing probability for high priority items P_{out}, and the expected number of items in the system, $E\left[R_i^l\right]$, and the expected number of postponed items, $E\left[PS_i^l\right]$, per item subclass. Each cell in the table contains three numbers, referring to the high priority item type and the two low priority item types, respectively. Note that the number of high priority items in the system includes the items outsourced. Recall that we can easily calculate this number from the *M/G/∞* queue (see section 7.2.4). We choose this definition, because we need the *total* number of items in process for the application that we have in mind (inventory control in spare part networks). Of course, we can also calculate the number of high priority items in the own repair from the steady state probabilities.

Table 7.1. The impact of outsourcing of high priority items in preemptive priority queueing system; each cell in the table contains the performance measure for all item classes

$\frac{\mu_1^l}{\mu_2^l}$	γ	$\rho^h = 20\%$					$\rho^h = 60\%$				
		$E\left[R_i^l\right]$		$E\left[PS_i^l\right]$		P_{out}	$E\left[R_i^l\right]$		$E\left[PS_i^l\right]$		P_{out}
		outsr.	*non-outsr.*	*outsr.*	*non-outsr.*		*outsr.*	*non-outsr*	*outsr.*	*non-outsr*	
0.5	0.5	0.60	0.61	0.00	0.00	0.02	1.80	2.33	0.00	0.00	0.18
		5.73	6.21	0.14	0.14	–	1.23	4.34	0.24	0.47	–
		12.00	13.10	0.15	0.16	–	2.00	8.98	0.27	0.52	–
	1	0.60	0.60	0.00	0.00	0.02	1.80	2.33	0.00	0.00	0.18
		6.16	6.74	0.10	0.11	–	1.33	5.85	0.20	0.40	–
		13.04	14.37	0.11	0.12	–	2.30	12.58	0.23	0.44	–
	2	0.60	0.60	0.00	0.00	0.02	1.80	2.32	0.00	0.00	0.18
		7.01	7.78	0.07	0.07	–	1.55	8.87	0.17	0.33	–
		15.06	16.84	0.08	0.08	–	2.85	19.72	0.18	0.35	–
1	0.5	0.60	0.61	0.00	0.00	0.02	1.80	2.33	0.00	0.00	0.18
		4.87	5.31	0.09	0.09	–	0.93	3.80	0.15	0.30	–
		11.37	12.38	0.20	0.21	–	2.17	8.86	0.36	0.69	–
	1	0.60	0.60	0.00	0.00	0.02	1.80	2.33	0.00	0.00	0.18
		5.31	5.84	0.06	0.07	–	1.05	5.32	0.13	0.25	–
		12.39	13.63	0.15	0.15	–	2.45	12.42	0.30	0.58	–
	2	0.60	0.60	0.00	0.00	0.02	1.80	2.32	0.00	0.00	0.18
		6.17	6.89	0.04	0.05	–	1.27	8.37	0.10	0.20	–
		14.39	16.08	0.10	0.11	–	2.97	19.52	0.24	0.47	–
2	0.5	0.60	0.61	0.00	0.00	0.02	1.80	2.33	0.00	0.00	0.18
		4.82	5.27	0.05	0.05	–	0.76	3.65	0.09	0.17	–
		12.30	13.37	0.24	0.25	–	2.42	9.39	0.43	0.83	–
	1	0.60	0.60	0.00	0.00	0.02	1.80	2.33	0.00	0.00	0.18
		5.27	5.82	0.04	0.04	–	0.89	5.20	0.07	0.14	–
		13.32	14.61	0.18	0.19	–	2.69	12.95	0.36	0.70	–
	2	0.60	0.60	0.00	0.00	0.02	1.80	2.32	0.00	0.00	0.18
		6.14	6.88	0.02	0.03	–	1.13	8.27	0.06	0.11	–
		15.31	17.06	0.12	0.13	–	3.21	20.04	0.29	0.57	–

From Table 7.1, we see that the mean numbers of items in the system as well as the mean number of postponed low priority items decrease if outsourcing is allowed. The impact of outsourcing on the number of items in the system is high, particularly for low priority items and for a high utilisation of high priority items ($\rho^h = 0.6$). But even if $\rho^h = 0.2$, when only 2% of the high priority jobs is outsourced, we see some impact on the number of items in the system. The main reason is, of course, the high total utilisation of ρ = 0.95. In that situation, the number of items in the system is rather sensitive to changes in the effective system utilisation (i.e., the utilisation corrected for outsourcing). Further, we note that the sensitivity of the number of low priority items in the system to $\gamma(=\mu^l/\mu^h)$ and particularly to μ_1^l/μ_2^l is relatively small.

7.4.2 Comparison between priority and non-priority queues

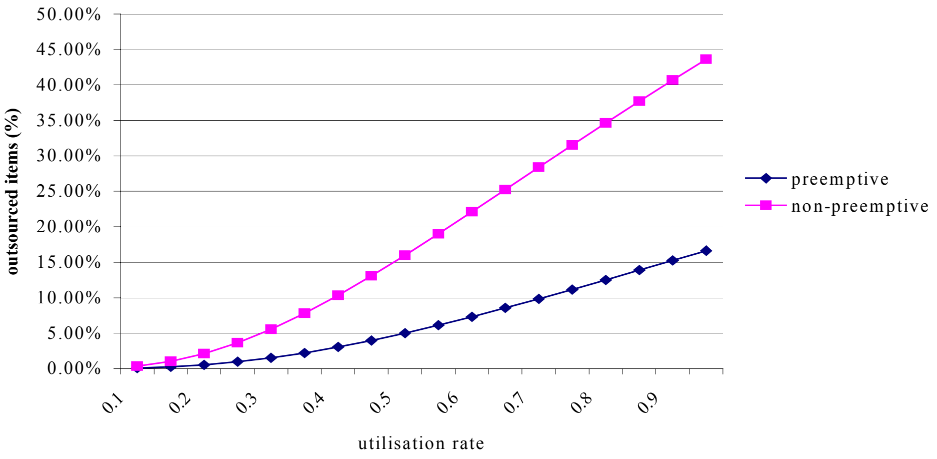

Figure 7.2. Percentage of outsourced high priority items as function of the total utilisation ρ.

In this section, we examine the impact of the priority discipline on the number of items in the system per class and on the fraction of high priority items that is outsourced. We vary the total utilisation ρ in the experiments with $N^h = 1$, $N^l = 2$, $k = 3$, $\mu_1^l/\mu_2^l = 0.5$, $\gamma = 0.5$ and $\rho^h = 0.6\rho$, $\rho^l = 0.4\rho$. In Figure 7.2, we show the percentage of high priority items that is outsourced as a function of the total utilisation ρ for both preemptive and non-preemptive priorities. The mean number of items in each class as a function of ρ is depicted in Figure 7.3 for both priority disciplines.

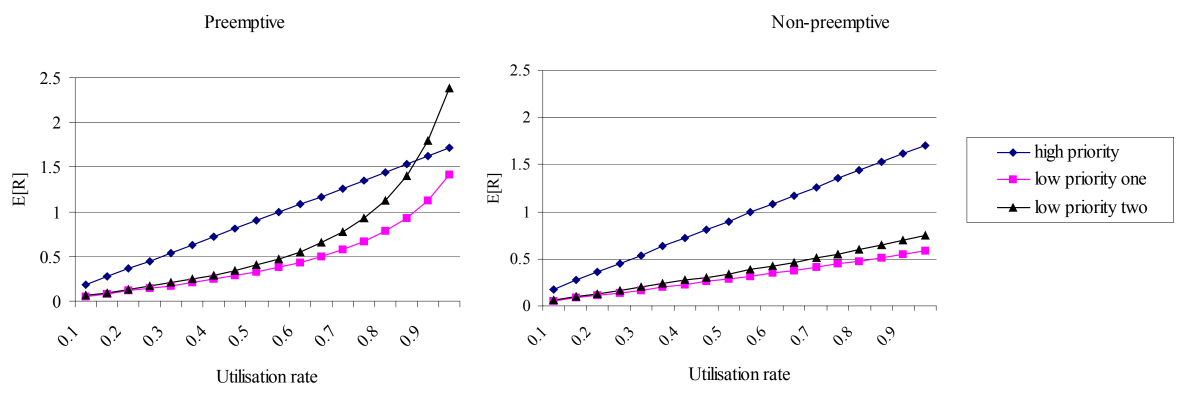

Figure 7.3. Mean number of items in the system per class as function of the total utilisation ρ in case of preemptive (left) and non-preemptive (right) priorities.

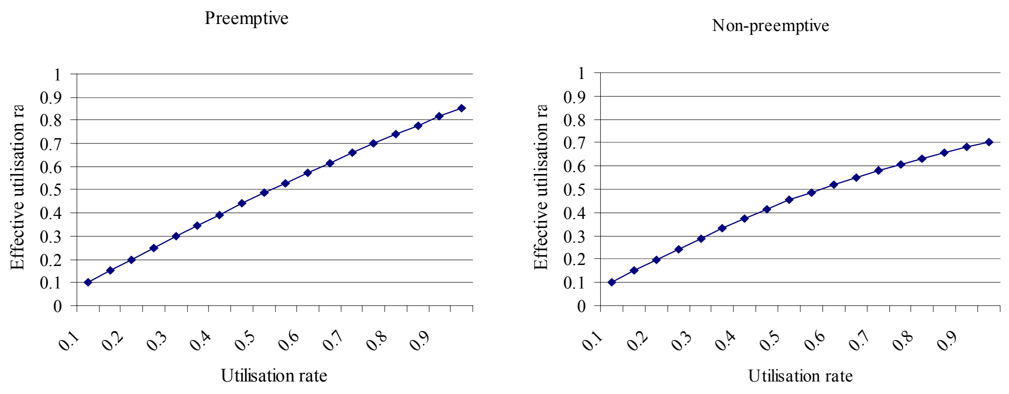

Figure 7.4. Effective repair shop utilisation in case of preemptive (left) and non-preemptive (right) priorities.

From these figures, we see that the priority discipline can have a significant impact on the system performance, especially if the repair shop utilisation is high. The percentage of high priority items that is outsourced depends strongly on the priority discipline (Figure 7.2). The number of low priority items in the system is not very sensitive to the priority discipline if the total utilisation rate is not too high (say $\rho \leq 0.7$). Otherwise, preemption leads to considerably more low priority items in the system in this example. This is caused by the fact that less high priority items are outsourced, and so the *effective* utilisation of the own repair shop is higher, see Figure 7.4. Further, we see from Figure 7.3 that the number of high priority items in the system is completely insensitive to the priority discipline as they should. Referring to our application on spare part inventories, we conclude that the priority discipline may have a significant impact on the inventory requirement of low priority items (which is related to the number of items in the queueing system) if the repair shop utilisation is relatively high.

7.5 Conclusions

In this chapter, we presented an exact method to analyse the multi-class *M/M/k* priority queue with outsourcing of high priority items that cannot be served immediately. We considered both preemptive and non-preemptive priorities. From the steady state probabilities, we can calculate the commonly used relevant performance measures. We tested our model in a numerical experiment and found that both outsourcing and the priority discipline can have a major impact on the number of low items in the system if the total system utilisation is relatively high. For the application that we had in mind when analysing this model, inventory optimisation in spare part networks, this means that the inventory requirements for low priority items can be significantly influenced by outsourcing and priority discipline. Thus, research question 5 (Chapter 1) is answered completely.

Finally, we note that we can also use our method to analyse a priority model with hyperexponential (H_x) service times and no item subclasses per priority group. In this case, we represent each priority class by x item classes with exponential service times and we adopt the performance estimators for the total number of items in the system.

Chapter 8
Using repair priorities to reduce stock investment in spare part networks

Abstract

In this chapter, we examine the impact of repair priorities in spare part networks. Several heuristics for assigning priorities to items as well as optimising stock levels are developed, extending the well-known VARI-METRIC method. We model repair shops by multi-class, multi-server priority queues. A proper priority setting may lead to a significant reduction in the inventory investment required to attain a target system availability (usually 10-20%). The saving opportunities are particularly high if the utilisation of the repair shops is high and if the item types sharing the same repair shop have clearly different characteristics (price, repair time). For example, we find an investment reduction of 73% for a system with single server repair shops with an utilisation of 0.90 that handle five different items. In this chapter we answer the last research question (question 6) of Chapter 1.

This chapter is based on the paper:

“Using repair priorities to reduce stock investment in spare part networks.” A. Sleptchenko, M.C. van der Heijden, A. van Harten. Submitted for publication.

8.1 Introduction

One of the key parameters determining stock levels of repairable items in spare part networks is the repair shop throughput time. It is clear that less inventory is needed to obtain a certain system availability if the throughput times can be reduced. If a repair shop handles a variety of item types and its capacity is limited, we can reduce the throughput times of expensive, critical items by giving the corresponding repair jobs high priority. As a consequence, other item types having low priority will have to face longer throughput times. This means that priority setting leads to reduced stock levels for high priority items and increased stock levels for low priority items. This provides us an opportunity for inventory investment reduction while maintaining the same system availability: if we choose expensive slow movers as high priority items and cheap fast movers as low priority items, the reduction of inventory investment in high priority items will probably be more than the increase of investment in low priority items. Equivalently, we can increase the system availability with the same budget by introducing adequate priority rules.

In this chapter, we examine the impact of using repair priorities in spare part distribution networks. We focus on *static* priorities, i.e. an a priori assignment of item types to priority classes for each repair shop, which is state-independent. We need such priorities, because we have to choose stock levels a priori as well. In fact, we focus on logistic decisions at a tactical level. At the operational level, one may consider using *dynamic* priorities additionally. That is, we may sequence repair jobs based on the specific system state at a specific point in time. For example, we may consider giving high priority to an item for which no stock is available, even if it has been classified as low priority originally. Such an operational finetuning of priorities can further improve the system performance, but it is out of the scope of our research. We will address the following research questions in the remainder of this chapter:

1) To which extent can we reduce inventory investment using a priori priority setting of items?
2) Which rule should we use to classify items as high priority or low priority?
3) Which impact has priority setting on the stock levels of individual items?

To answer these questions, we need (1) an algorithm for assigning priorities to items sharing the same repair capacity, and (2) an algorithm for spare part inventory optimisation based on these priorities. We will discuss both sequential optimisation and integral priority assignment / stock optimisation. In our approach, we model the repair shops as multi-class, multi-server queues with preemptive priorities. Our model contains two priority groups (high and low), each consisting of one or more subclasses. Hence priority assignment boils down to assigning either high or low priority to each item. To analyse these queueing models, we use the method as developed in Chapter 6. Note that our priority assignment heuristics do *not* lead to optimal solutions. Even the case without priorities for capacitated repair shops is already that complex, that optimal stock levels cannot be guaranteed except for some special cases.

In the remainder of this chapter, we first explore the potential of repair priorities using a variant of VARI-METRIC for given priorities in section 8.2. In section 8.3, we discuss an algorithm for integral priority assignment and stock optimisation and a sequential algorithm. We compare the various heuristics in a numerical experiment in section 8.4. As performance criteria, we use run time and solution quality, that is, the inventory investment needed to achieve a target availability. Finally, we give our conclusions in section 8.5.

8.2 The impact of repair priorities

In this section, we will first discuss how VARI-METRIC can be extended to finite capacity repair shops and with job priorities. Using this model, we can then optimise spare part stock levels for a given assignment of items to (fixed) priority classes. Next, we will compare the inventory investment required for various priority classifications to the investment requirements if no priorities are used. In all cases, we consider finite capacity repair shops.

8.2.1 VARI-METRIC with fixed priority classes

Similarly to Chapter 4, other queueing models for the repair shops can be embedded in VARI-METRIC if the first two moment of the number of items in the queueing system can be evaluated. The latter condition is not satisfied for many queueing models that have been discussed in the literature. Often, queueing analysis is limited to a few core performance characteristics like the mean waiting time and the mean queue length. Even if the variance of the queue length can be evaluated, this is not sufficient for use in VARI-METRIC, because we need the variance of the *total* number of items in the system. Note that the number of items in queue and the number of items in repair are correlated, so that we cannot simply add up both variances.

A suitable priority queueing model for our purpose is the multi-class *M/M/k* priority model that has been analysed in Chapter 6, being an extension of the non-priority model as has been analysed in Chapter 3. Basic characteristics are:

- repair jobs are classified in item classes $i = 1 \ldots I$;
- each item class is assigned either high or low priority; hence we have two priority groups that each consist of one or more item classes;
- repair job arrivals are Poisson distributed with an item class dependent arrival rate λ_i;
- repair times are exponentially distributed with an item class dependent service rate μ_i;
- all jobs are handled by k identical servers;
- a low priority job is interrupted if a high priority job arrives finding all servers busy; if multiple low priority are in service, we select the job to be postponed randomly with equal probabilities.

In a spare part network, we may have dedicated repair capacity for certain job types. In that case, we model each dedicated repair facility as a multi-class *M/M/k* priority model.

8.2.2 Comparing priority and non-priority repair shops

In this section, we will use two variants of VARI-METRIC to examine the impact of repair priorities, namely the variant based on a multi-class *M/M/k* queue without priorities and the variant based on the multi-class *M/M/k* priority queue. Using some numerical experiments, we will show that the inventory investment can be reduced significantly if we use repair priorities. In this section, we simply optimise stock levels for fixed priority classes and vary the priority assignment in our experiments to see the impact of proper priority setting.

We use the two-indenture product structure and the two-echelon distribution structure as shown in previous chapters (e.g. Figure 5.3). The network consists of one central depot and four frigates. On each frigate, two pumps are installed (A and B). A frigate is available if both pumps are in operation. The pumps have two critical parts each. For pump A, these are a valve and a piston and for pump B these are a flange and a piston. The failure rates of pump A and pump B are 10 and 15 failures per pump per year, respectively. The failure cause for pump A is either the valve (with probability 0.3) or the piston (with probability 0.4) or there is a combination of factors, meaning that the whole pump has to be repaired or replaced (with probability 0.3). For pump B, the failure causes are 0.4 for both the flange and the piston and 0.2 for the pump itself. Repair shops are present at the depot and the frigates. Every repair job can be repaired at the frigate with probability 0.2 (simple repairs), whereas it should be forwarded to the depot with probability 0.8 (complex repairs). The order-and-ship time between depot and frigates is zero for all items. Table 8.1 shows the item prices and the mean repair times. The mean repair times are expressed relatively to the mean repair time of pump A. Because we will vary repair shop capacities and utilisation in our experiments, we will choose the repair time of pump A such, that a given utilisation is reached.

Table 8.1. The relative mean service times and item prices

Item	Pump A	Pump B	Valve	Flange	Piston
Mean service time $E[S_i]$ relative to the mean service time of pump A	1	2	2	4	3
Prices (euro) c_i	5000	7100	200	300	500

We vary the following parameters in our experiments:

1. The assignment of items to repair shops: (a) all items are assigned to the same repair shop, so we have a single five-class repair shop at all locations, (b) one repair shop is dedicated to pump A and pump B (two-class model) and one repair shop is dedicated to the valve, the flange and the piston (three-class model).
2. The assignment of items to priority classes. We consider all options, so we have 2^5-1 = 31 options for a single five-class repair shop and $(2^2-1)*(2^3-1)$ = 21 options for the combination of a two-class and a three-class repair shop. Note that we subtract one option per repair shop, because it does not make a difference whether all items have high priority or all items have low priority.
3. The number of servers per repair shop: $k = 1$ or $k = 3$.
4. The repair shop utilisation: $\rho = 0.8$ or $\rho = 0.9$.

All repair shops in the system have the same number of servers and the same utilisation. We vary the number of servers and the utilisation in our experiments, because these parameters appeared to be critical in the finite capacity analysis presented in Chapter 4. Other parameters, such as failure rates, the number of echelons and indenture levels, appear to be less crucial. In the same paper, it is also shown that the impact of finite capacity is especially high if the utilisation is high. Therefore, we do not include low utilisation rates in our experiments.

In Table 8.2, we show the key results of our numerical experiments.

Table 8.2. Investments in inventories (x 1000 euro) for different combinations of priorities, number of servers and utilisation rates

Priority per item (L= low, H=high)					*One five-class repair shop*				*One two-class and one three-class repair shop*			
					$\rho = 0.8$		$\rho = 0.9$		$\rho = 0.8$		$\rho = 0.9$	
Pump A	Pump B	Valve	Flange	Piston	$k = 1$	$k = 3$	$k = 1$	$k = 3$	$k = 1$	$k = 3$	$k = 1$	$k = 3$
No priorities					*326*	*328*	*697*	*697*	*517*	*539*	*1090*	*1110*
L	L	H	L	L	346	341	732	727	516	537	1094	1112
L	L	L	H	L	404	396	882	877	534	555	1125	1142
L	L	H	H	L	429	416	943	932	537	558	1133	1151
L	L	L	L	H	420	412	917	913	515	536	1081	1104
L	L	H	L	H	449	434	987	978	515	536	1083	1102
L	L	L	H	H	574	563	1318	1313	535	552	1144	1166
L	L	H	H	H	625	614	1490	1478	—	—	—	—
H	L	L	L	L	235	245	473	486	452	479	919	941
H	L	H	L	L	244	253	494	505	452	479	919	942
H	L	L	H	L	310	304	633	624	476	490	976	986
H	L	H	H	L	329	319	680	669	480	500	990	997
H	L	L	L	H	308	311	641	631	448	475	901	928
H	L	H	L	H	333	329	684	679	**439**	**473**	**892**	**918**
H	L	L	H	H	449	442	1006	995	475	499	1025	1055
H	L	H	H	H	498	494	1151	1141	—	—	—	—
L	H	L	L	L	236	246	468	473	640	643	1344	1335
L	H	H	L	L	252	258	495	499	641	644	1353	1338
L	H	L	H	L	317	315	650	633	653	657	1387	1371
L	H	H	H	L	351	336	714	699	663	663	1412	1395
L	H	L	L	H	318	320	652	643	638	640	1332	1317
L	H	H	L	H	352	345	726	719	637	641	1341	1328
L	H	L	H	H	495	491	1148	1147	646	649	1357	1353
L	H	H	H	H	559	568	1417	1410	—	—	—	—
H	H	L	L	L	**125**	**148**	**190**	**212**	—	—	—	—
H	H	H	L	L	132	153	201	219	—	—	—	—
H	H	L	H	L	189	191	290	305	—	—	—	—
H	H	H	H	L	204	219	333	343	—	—	—	—
H	H	L	L	H	171	185	251	268	—	—	—	—
H	H	H	L	H	183	205	286	304	—	—	—	—
H	H	L	H	H	293	318	543	573	—	—	—	—
Savings					**62%**	**55%**	**73%**	**70%**	**15%**	**12%**	**18%**	**17%**

The first line contains the inventory investment that is required to obtain an average availability of 95% if all items have the same priority. We show the investment corresponding to the best priority assignment in bold figures. Note that the investment reductions are only comparable within the same column and *not* between columns. The reason is that the item repair times vary between columns, because these times are chosen such, that the specified utilisation rate is attained for the given assignment of items to repair shops and the given number of servers *k*. Further, some of the assignments of items to priority classes are not relevant for the two-repair shop experiments, because it does not make a difference whether all items within a repair shop have high priority or all have low priority. Therefore, some of the cells in Table 8.2 are empty.

From Table 8.2, we see that proper priority setting may reduce the inventory investment requirement substantially (12%-73%). The investment reduction decreases with the number of servers and increases with the repair shop utilisation rate. However, the impact of the assignment of items to repair shops is dominant. The potential investment reduction is particularly high if items with clearly different characteristics share the same repair capacity. In that case, the increase in stock investment for (cheap) low priority items is much less than the decrease in stock investment for (expensive) high priority items. The latter is illustrated in Table 8.3, that shows the modification in stock levels per item at both the depot and each frigate if the optimal priority assignment is used compared to a non-priority system. We only present the results for 1 server and utilisation 0.9, but the results for the other cases are similar.

We see that the stock levels of high priority items decrease and those of low priority items increase, as expected. The main cause for the huge investment reduction if all items share the same repair capacity is the difference in item costs. For example, the number of pumps B on stock decreases from 4 (frigates) x 13 + 1 (at depot) = 53 pumps to (3 x 2 + 1 x 3) + 0 = 9 pumps. The investment reduction is 44 x 7100 euro = 312.400 euro. Similarly, we see that we save 230.000 euro on the other high priority item, pump A. Although the stock levels of the low priority items increase significantly, these are all cheap, so that we need only 45.500 euro additional investment in valves, flanges and pistons. As a consequence, the investment reduction is 496.600 euro.

Table 8.3. Comparison of stock allocation between the optimal priority system and a non-priority system (*c*=1, ρ = 0.9); the mark * means that one frigate gets one additional item on stock in the optimal solution to obtain an *average* system availability of at least 0.95.

	Location →	Depot					Frigates				
Repair shop structure	Item →	Pump A	Pump B	Valve	Flange	Piston	Pump A	Pump B	Valve	Flange	Piston
One five-class repair shop	*No priorities*	1	1	6	10	13	13*	13	4	7	9
	Optimal priorities	0	0	8	15	21	2	2*	9	15	22
One two-class and one three-class repair shop	No priorities	2	3	6	10	13	21*	20	7	13	17
	Optimal priorities	1	4	2	24	3	4	25*	3	33	5

Note that the reduction in pump investment is also high compared to the increase in investment in cheap items, because the repair times of pumps are shorter in these examples (see Table 8.1). Therefore, the delay of low priority items is not too much. Intuitively, it is clear that item costs and mean repair times are important parameters influencing investment reduction because of priority setting. Giving high priority to items with a relatively small repair time has a similar impact as using the Shortest Processing Time (SPT) rule: the average number of items in the system decreases.

8.3 Integral priority assignment and stock optimisation

Finding an optimal assignment of items to priority classes and determining the corresponding optimal stock levels is a very complex problem because of the large number of integer decision variables and the complex relation between stock levels and backorders. As discussed in the previous sections, the latter relation includes complex queueing models for each finite capacity repair shop. Therefore, we will discuss two heuristic optimisation approaches in this section:

- nested optimisation, i.e. we optimise the stock levels for each priority assignment considered;
- sequential priority assignment and stock optimisation (decomposition).

8.3.1 Nested optimisation

A straightforward method to optimise priority assignment and stock levels is to extend VARI-METRIC with priority assignment. VARI-METRIC subsequently adds stock for those items at those locations that give the highest improvement in the goal function (mean backorders) per invested dollar. It is difficult to mix the stock level decision with priority assignment, because changing priorities requires a new stock optimisation. However, we can construct a layer of priority assignment around the VARI-METRIC stock optimisation procedure. This leads to the following nested optimisation heuristic:

<u>Nested stock optimisation and priority assignment</u>

1. *Initialisation:* Set all item priorities at all repair shops equal, at the low level; optimise stock levels for equal priorities according to Chapter 4.
2. *Priority improvement:* Examine the impact of assigning high priority to a single item that is currently low priority for all repair shops while keeping the same priority for all other items:
   a. Re-optimise the stock levels for each modified priority assignment and calculate the investment reduction resulting from the modified priority assignment using the extended VARI-METRIC algorithm as discussed in section 8.2.1.
   b. Select the priority modification causing the highest inventory reduction; if this reduction is positive, implement the new priority assignment and repeat step 2; otherwise, keep the previous solution as the optimal one and stop.

This is a nested optimisation, because all stock levels are re-optimised every time the priority assignment is modified. Note that this is a heuristic procedure, because (1) VARI-METRIC is generally not an exact optimisation procedure, (2) items are only moved from the low priority group to the high priority group and not vice versa; there is no guarantee that this will lead to an optimal priority assignment. Still we examine many options for priority assignment and stock levels, so that we will probably find a good (although not optimal) solution. Because an optimal solution is not possible for most practical problems (even not for a non-priority system), it is impossible to give an indication for the quality of the optimisation procedure. We can however compare priority optimisation to an optimised system without priorities and analyse the investment reduction, see section 8.4.

Unfortunately, this procedure may be time consuming, because the number of stock optimisations may be excessive. In each iteration, the number of stock optimisations has order $N^{loc} \times N^{item}$, where N^{loc} represents the number of stock locations in the network and N^{item} represents the number of items (assemblies and subassemblies) in the system. The number of iterations clearly depends on the parameters, but cannot exceed $N^{loc} \times N^{item}$, because then all items have been moved to the high priority class at all locations. Hence, the algorithm may require $O((N^{loc} \times N^{item})^2)$ stock optimisations. The number of evaluations of a multi-class *M/M/k* priority queue is bounded by $O(N^{loc} \times (N^{item})^2)$. This number is smaller, because we only have to evaluate $O(N^{item})$ new priority models for the repair shop where we moved an item to the high priority class in the previous iteration. Still, we conclude that the nested optimisation approach may take a lot of run time, so that it is only practical for small problem instances. Therefore, we search for a faster heuristic.

8.3.2 Sequential priority assignment and stock optimisation

The nested optimisation procedure considers many modifications in priority assignment. It is clear that a part of these assignments is probably not advantageous, because the item characteristics do not justify high priority. For example, items having relatively low costs and/or relatively long service times are candidates for low priority rather than high priority, as we also concluded from the numerical experiments in section 8.2.2. Therefore, we can try to reduce the number of priority modifications under consideration.

A simple way to reduce the run time needed for the optimisation is to decompose the problem in priority assignment and stock optimisation. We first assign items to priority classes in each repair shop without considering stock optimisation. Next, we optimise stock levels *only once* for the (heuristically found) priority assignment. An a priori assignment of items to priority classes is not straightforward, because it depends on many parameters of all items sharing the same repair capacity, such as the total repair shop utilisation, the variation in items costs, the variation in mean service times and the variation in arrival rates. Some preliminary experiments that we do not discuss here showed that the high priority class may use 50-60% of the repair shop capacity in some cases and only 10-20% in other cases. The precise relation between the optimal assignment and all relevant parameters is not clear. Only the number of servers seems to be less relevant for the optimal priority assignment.

As a solution, we estimate the impact of priority assignment on investment using a goal function that (1) is a good indicator for stock investment and that (2) can easily be evaluated. It is clear that stock levels tend to be high (low) if the number of items in the repair shop are high (low). Therefore, we propose to use as goal function for priority assignment *the total expected costs of being in the system for all job classes in a single server, multi-class priority queue*, i.e. $\sum_{i=1}^{I} c_i h_i E[R_i]$, where N_i denotes the number of items of type i in the repair shop, c_i the costs of item i and h_i the holding cost rate for item i. The hypothesis is that the stock levels are approximately proportional to the number of items in the systems. Obviously, this is not true, but we hope that this indicative goal function is sufficient for priority assignment. We will test the validity of this heuristic approach in some numerical experiments (section 8.4). Note that we may simplify the goal function to $\sum_{i=1}^{I} c_i E[R_i]$ if the holding cost rate h_i is the same for all items

An advantage of this approximate goal function is that we can evaluate it quickly, because the state dimension of the *M/M/1* multi-class queue remains within reasonable limits, see Chapter 4. Because computation times increase significantly if k>1, we propose to approximate the system by a single server queue with a server that works k times as fast if the actual repair shop has $k \geq 2$ servers (*only* for priority assignment).

One option of finding the optimal priority assignment for the approximate goal function is to enumerate all options and choose the priority assignment with minimum costs. This is possible for a moderate number of item classes, say 12 at most (giving 2^{12}-1 = 4095 options). However, we can also construct a heuristic from Buzacott and Shantikumar's (1993) result on the optimal priority assignment in the multi-class *M/G/1* model with *non*preemptive priorities. They order jobs according to a *priority index* that is assigned to each class. They assume that each class contains one item and proof that $c_i/E[ST_i]$ is the optimal priority index, where S_i and c_i denote the service time and the costs per time unit in the system for item i, respectively. That is, the item with the highest index $c_i/E[ST_i]$ should have the highest priority. It is not difficult to show that this result is also valid for *preemptive* multi-class *M/G/1* model systems (see Appendix B).

Note that our model is different, because we have *two* priority groups with *multiple* items (classes) within each priority group. Hence we should not only order items, but also decide *how many* items should be allocated to the high priority group. Based on the result above, a reasonable heuristic is to order items according to the index $c_i/E[ST_i]$. Starting with low priority items only, we move items to the high priority class until the cost criterion cannot be improved anymore. In the numerical section, we will check the quality of this priority assignment heuristic by comparison to a full enumeration of all priority assignments.

Summarised, the sequential heuristic consists of the following steps:

<u>**Sequential priority assignment and stock optimisation**</u>

1. Priority assignment per repair shop,
   a. Initialisation:
      i) If the number of servers in the repair shop is $k \geq 2$, then modify the mean service time for priority assignment as $E[ST_i] := E[ST_i]/k$.
      ii) Sort the items in decreasing order of their cost / service time ratio $c_i/E[ST_i]$.
      iii) Assign low priority to all items in the repair shop.
      iv) Calculate the expected number of items $E[R_i]$ in the *M/M/1* multi-class queue for each item type i.
      v) Calculate the minimum total expected costs of being in the system for all job classes, $\sum_{i=1}^{I} c_i E[R_i]$.
   b. Iteration:
      i) Assign high priority to the low priority item i^* that is highest on the ordered list.
      ii) Recalculate the expected number of items $E[R_i]$ in the *M/M/1* multi-class priority queue for each item and the corresponding total expected costs $\sum_{i=1}^{I} c_i E[R_i]$.
      iii) If the total costs are less than in the previous iteration, continue with the next iteration; otherwise, assign low priority to item i^* and stop.
2. Optimise the stock levels and calculate the investment reduction resulting from the priority assignment using the extended VARI-METRIC algorithm from section 8.2.1.

8.4 Numerical analysis

In this section, we will test both optimisation heuristics. Evaluation criteria are (1) the investment reduction obtained compared to a non-priority finite capacity model and (2) the computer run time requirements. We conduct three sets of experiments. In the first set, we compare the investment reduction obtained using the two heuristics for priority assignment and stock optimisation as presented in the previous section. It will turn out that sequential priority assignment and stock optimisation (see section 8.3.2) is as good as nested optimisation and much faster in most of our numerical experiments. Therefore, we conduct a second set of experiments based on sequential optimisation to examine priority assignment in more detail (i.e., experiments in which many items share the same repair capacity). In the third set we validate our findings by running experiments for a completely different data set as given in Hausman and Scudder (1982).

8.4.1 Comparing the algorithms for priority assignment and stock optimisation

The data for our numerical experiments are similar to section 8.2.2, except for the following:

- We only consider the cases with $k = 3$ servers, because we see from Table 8.2 that the results for $k = 1$ server are similar.
- We consider both a two-indenture and a three-indenture system, see Figure 8.1; the item costs and the mean service times for the three indenture models are shown in Table 8.4.

We have two scenarios for the failure causes, see Figure 8.1 (the probability that an assembly failure is caused by a specific subassembly is displayed as a number at the connecting link); the first scenario is similar to section 8.2.2 with an extension for the three-indenture variant; the second scenario is displayed between parentheses.

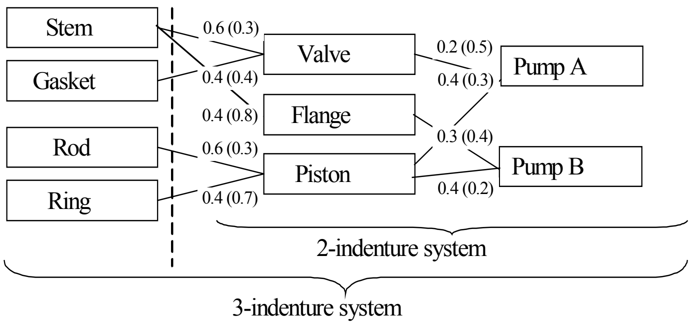

Figure 8.1. The indenture structure with two scenarios for the failure cause probabilities

Table 8.4. The relative mean service times and item prices

Item	Pump A	Pump B	Valve	Flange	Piston	Stem	Gasket	Rod	Ring
Mean service time $E[S_i]$ relative to the mean service time of pump A	1	2	2	4	3	2	4	1	3
Prices (euro) c_i	5000	7100	200	300	500	120	60	50	80

In our first set of experiments, each repair facility is dedicated to the items from the same level in the indenture structure. That is, each location has two respectively three (multi-class, multi-server) repair shops in the two- respectively three-indenture model. One repair shop is dedicated to pump A and pump B, one repair shop is dedicated to the valve, the piston and the flange and the third repair shop (only in the 3-indenture model) is dedicated to the other items.

Summarised, we vary three parameters: the number of indentures, the failure cause probabilities and the repair shop utilisation rates (i.e., we have $2^3 = 8$ cases). We optimise all cases using 4 algorithms:

1. inventory optimisation without repair priorities;
2. nested priority assignment and inventory optimisation;
3. sequential priority assignment and inventory optimisation in two variants:
    a. enumeration of all options in the priority assignment step;
    b. priority assignment based on the priority index $c_i/E[S_i]$.

In Table 8.5, we present for the various experiments the stock investment required to obtain an average system availability of 95%, the relative reduction using repair priorities and the algorithm run time (based on a PC with Pentium-III 800 Mhz processor). Similar to Table 8.2, investment reductions are only comparable within the same row and *not* between rows, because the item repair times vary between rows (they are chosen such, that the specified utilisation rate is attained for the given assignment of items to repair shops).

We see that the sequential algorithm performs only slightly worse than the nested algorithm in terms of investment reduction, whereas it is much faster. Therefore, sequential optimisation seems to be an attractive heuristic. The difference between both variants of the sequential algorithm are minor, because the number of items per repair shop is always small (at most 4).

Table 8.5. Inventory investments (x 1000 euro), investment reduction using repair priorities and run times for various optimisation algorithms

Indentures	Failure cause scenario	Utilisation Rates	No priority	Nested procedure	Sequential, enumeration	Sequential, priority index
Two	One	$\rho = 0.8$	814	716	718	718
			—	12.0%	11.8%	11.8%
			2 s.	23 m. 52 s.	22 s.	19 sec.
		$\rho = 0.9$	1667	1381	1387	1387
			—	17.2%	16.8%	16.8%
			8 s.	43 m. 30 s.	23 s.	23 sec.
	Two	$\rho = 0.8$	822	729	729	729
			—	11.3%	11.3%	11.3%
			3 s.	25 m. 41 s.	12 s.	11 sec.
		$\rho = 0.9$	1688	1415	1415	1415
			—	16.2%	16.2%	16.2%
			9 s.	45 m. 30 s.	16 s.	17 sec.
Three	One	$\rho = 0.8$	848	751	755	755
			—	11.5%	11.0%	11.0%
			9 s.	11 h. 17 m. 9 s.	4 m. 44 s.	4 m. 47 s.
		$\rho = 0.9$	1750	1443	1468	1468
			—	17.6%	16.1%	16.1%
			30 s.	26 h. 16 m. 41 s.	5 m. 47 s.	5 m. 42 s.
	Two	$\rho = 0.8$	851	748	748	754
			—	12.1%	12.1%	11.3%
			7 s.	8 h. 11 m. 8 s.	4 m. 14 s.	4 m. 17 s.
		$\rho = 0.9$	1745	1451	1451	1478
			—	16.9%	16.9%	15.3%
			25 s.	23 h. 28 m.35 s.	5 m. 40 s..	5 m. 33 s.
Average investment reduction			**0.0%**	**14.4%**	**14.0%**	**13.7%**

8.4.2 Investigating priority assignment

To examine the impact of the priority assignment algorithm in more detail, we need to run experiments in which more items share the same repair shop capacity. Therefore, we focus on the three-indenture models. We only include both variants of the sequential algorithm, because the nested procedure requires excessive run times (as can also be seen from Table 8.5). To keep run times small, we focus on repair shops with $k = 1$ server. We take scenario two for the failure cause probabilities and a repair shop utilisation of $\rho = 0.90$. We consider four variants for the assignment of items to repair shops:

1. three repair shops, one for each indenture level (i.e. with 2, 3 and 4 items respectively);

2. two repair shops, one for indenture level 1 and 2 and one for indenture level 3 (i.e., with 5 and 4 items respectively);
3. two repair shops, one for indenture level 1 and one for indenture level 2 and 3 (i.e., with 2 and 7 items respectively);
4. one repair shop for al items (i.e., 9 items sharing the same capacity)

Table 8.6. Priority assignment, investment reduction and algorithm run times for various repair shop structures

Repair shop structure	Priority assignment	Investment reduction (run time)	Pump A	Pump B	Valve	Flange	Piston	Stem	Gasket	Rod	Ring
Three repair shops:	enumeration	18.5% (23 s.)	H	L	H	L	L	H	L	H	H
2, 3 and 4 items	ordering	17.1% (23 s.)	H	L	L	L	L	H	L	H	H
Two repair shops:	enumeration	58.0% (44 s.)	H	H	L	L	L	H	L	H	H
5 and 4 items	ordering	58.0% (13 s.)	H	H	L	L	L	H	L	H	H
Two repair shops:	enumeration	17.7% (3 m. 33 s.)	H	L	H	H	L	H	L	H	L
2 and 7 items	ordering	17.7% (13 s.)	H	L	L	L	L	L	L	L	L
One repair shop:	enumeration	65.9% (56 m. 15 s.)	H	H	L	L	L	L	L	L	L
9 items	ordering	65.9% (17 sec.)	H	H	L	L	L	L	L	L	L

We show the results of these experiments in Table 8.6. For both variants of priority assignment, we give the investment reduction compared to a non-priority system, the run time needed for the optimisation algorithm and the assignment of priorities to items, H(igh) or L(ow).

We see that the investment reduction is almost the same for both enumeration of priorities and ordering based on the index $c_i/E[ST_i]$. Sometimes the assignment of items to priority classes is different, but then the investment reduction is almost the same. However, priority assignment using ordering is much faster, particularly when many items share the same repair shop capacity. Even for repair shops modelled as *M/M/1* priority queues with 9 classes, the run time is still relatively small (17 seconds). Therefore we conclude that the sequential heuristic based on the priority index is a suitable heuristic for large models.

Further, we observe that most investment reduction is possible if the two pumps share repair capacity with many cheaper items. Both pumps have relatively short repair times, and so repair priority for these pumps diminish the number of pumps in repair (and so the required inventories) considerably. Because the pumps are also much more expensive than the other items, this yields a large investment reduction. Hence repair priorities are especially attractive if many items with completely different characteristics (item costs, service times) share repair shops with a high utilisation.

8.4.3 Validation of findings

Because the structure of the data is similar in the previous experiments, our findings may be case-dependent. Therefore, we examine whether we find similar results using a different data set. To this end, we use the data as provided by Hausman and Scudder (1982) on engines at a commercial airline. They construct a simulation model for a three indenture, single location model with a single repair facility. All engine failures are caused by failure of one of the 23 components (third indenture items). The repair facility consists of 10 machines. Each item type passes some, but generally not all, machines following an item-specific route. The processing time at each machine is deterministic and differs for each item/machine combination. Obviously, this model is different from ours. Therefore, we only use the data from the repair shop by Hausman and Scudder (1992). We model the repair facility by a number of multi-server repair shops that are dedicated to a subset of items. We use the total machine times from Hausman and Scudder as mean repair times and we assume that these repair times are exponentially distributed.

We consider five variants for the assignment of items to repair shops, see Table 8.7. As we have mentioned in the introduction, our method to analyse priority queues is able to handle a limited number of item classes and servers only. Therefore, we are not able to consider a single repair shop for all items, which would require a 10-server priority queue, where both priority classes have totally 23 subclasses (components). Then the state space becomes too large, so new and fast approximations have to be developed to handle such large repair shops.

Table 8.7. Repair shop assignments and characteristics (numbers of servers and utilisation rates)

	Comp.11	Comp.12	Comp.13	Comp.14	Comp.21	Comp.22	Comp.23	Comp.31	Comp.32	Comp.33	Comp.34	Comp.35	Comp.36	Comp.37	Comp.41	Comp.42	Comp.43	Comp.44	Comp.51	Comp.52	Comp.53	Comp.54	Comp.55
Var. 1	$k = 2, \rho = 0.84$				k=1, ρ=0.88			$k = 2, \rho = 0.91$							$k = 2, \rho = 0.89$				$k = 2, \rho = 0.83$				
Var. 2	$k = 3, \rho = 0.86$							$k = 2, \rho = 0.91$							$k = 2, \rho = 0.89$				$k = 2, \rho = 0.83$				
Var. 3	$k = 2, \rho = 0.84$				$k = 2, \rho = 0.73$						$k = 2, \rho = 0.68$					k=2, ρ=0.83			$k = 2, \rho = 0.83$				
Var. 4	$k = 1$, ρ=0.79		$k = 1$, ρ=0.90		$k = 1$, $\rho = 0.88$			$k = 1$, $\rho = 0.96$				$k = 1$, ρ=0.77		$k = 1$, $\rho = 0.71$			k=1, ρ=0.69	$k = 1$, ρ=0.78		$k = 1$, ρ=0.74		$k = 1$, ρ=0.62	
Var. 5	$k = 3, \rho = 0.84$						$k = 3, \rho = 0.73$						$k = 2, \rho = 0.85$					$k = 3, \rho = 0.71$					

In all experiments, we assume that engines fail with a frequency of 0.50 per day. In Table 8.8, we present the remaining data for our experiments, i.e. the mean repair times, prices and probabilities that a failure is caused by a specific component. The key results are presented in Table 8.9.

Table 8.8. Prices (x $1000), repair times and cause probabilities of the single-echelon, multi-indenture model by Hausman and Scudder [1982]

Operating units		Modules			Componets			
Name	Price (x $1000)	Name	Price (x $1000)	Cause prob.	Name	Price (x $1000)	Cause prob.	Mean repair time (days)
Engine	2000	Module 1	800	0.1667	Component 11	135	0.2903	14.10
					Component 12	85	0.1936	27.70
					Component 13	150	0.2903	14.59
					Component 14	110	0.2258	29.07
		Module 2	308	0.0833	Component 21	80	0.3333	14.98
					Component 22	100	0.6000	25.05
					Component 23	60	0.0667	16.08
		Module 2	630	0.2500	Component 31	160	0.1026	17.40
					Component 32	110	0.0256	18.00
					Component 33	115	0.1538	15.45
					Component 34	90	0.2308	13.15
					Component 35	120	0.2051	18.16
					Component 36	85	0.2308	10.63
					Component 37	100	0.0513	14.60
		Module 2	490	0.2778	Component 41	140	0.0800	10.60
					Component 42	100	0.3200	11.15
					Component 43	80	0.2000	24.95
					Component 44	150	0.4000	8.46
		Module 2	575	0.2222	Component 51	140	0.3200	8.60
					Component 52	90	0.1600	14.55
					Component 53	100	0.2400	18.00
					Component 54	130	0.0800	15.95
					Component 55	110	0.2000	21.64

Table 8.9. Investment reduction using repair priorities with priority assignment for the Hausman and Scudder (1982) example (H=high priority, L=low priority)

	Invest. Reduc.	Comp.11	Comp.12	Comp.13	Comp.14	Comp.21	Comp.22	Comp.23	Comp.31	Comp.32	Comp.33	Comp.34	Comp.35	Comp.36	Comp.37	Comp.41	Comp.42	Comp.43	Comp.44	Comp.51	Comp.52	Comp.53	Comp.54	Comp.55
Var. 1	17.9%	H	L	H	L	H	L	L	H	L	H	L	L	H	L	H	H	L	H	H	H	L	H	L
Var. 2	19.9%	H	L	H	L	H	L	L	H	L	H	L	L	H	L	H	H	L	H	H	H	L	H	L
Var. 3	20.7%	H	L	H	L	H	L	L	H	L	H	L	L	H	L	H	H	L	H	H	H	L	H	L
Var. 4	11.1%	H	L	H	L	H	L	L	H	L	H	L	L	H	L	H	L	L	H	L	H	L	H	L
Var. 5	14.7%	H	L	H	L	H	L	L	H	L	H	L	L	H	H	H	H	L	H	H	L	L	L	L
Price / S.Time		9.6	3.1	10.3	3.8	5.3	4.0	3.7	9.2	6.1	7.4	6.8	6.6	8.0	6.8	13.2	9.0	3.2	17.7	16.3	6.2	5.6	8.2	5.1

In this table, we show the investment reduction, the assignment of priorities to items and the price / service time ratio. The variants for repair shop assignment are shown using vertical lines. That is, all components between two successive vertical lines share the same repair shop are assigned. Note that we do not distinguish between the two variants of sequential optimisation, enumeration and based on price \ service time ratio, because they give the same results in all five experiments.

We see that the investment reduction is not excessive, because the price/service time ratios of items sharing the same repair capacity do not vary too much. Still, we find 10-20% investment reduction, similar to the non-extreme cases from the previous sections. As we expected, high priority is always assigned to items with the highest price / service times ratios in the same repair shop. These results confirm our findings from the previous section.

8.5 Conclusion

We showed that repair priorities provide an opportunity to reduce investments in repairable spare part networks (usually 10%-20%), especially if the repair shop utilisation is high and many items with different costs and repair times share the same repair capacity. We presented an algorithm for priority assignment and stock optimisation and we showed that a sequential algorithm performs very well in terms of the investment reduction found and algorithm run times. We summarised our preferred sequential heuristic at the end of section 8.3.2. With the results presented in this chapter, we answered the last research question (Chapter 1).

Chapter 9
Conclusions and further research

Abstract

This chapter summarises the results from this thesis. We present the answers to the research questions that we have posed in Chapter 1. Further, we indicate directions for future research.

9.1 Introduction

The goal of this thesis is the extension of spare parts inventory models with finite repair capacities and repair job priorities and gaining insight in the relations between inventories, repair capacities and repair priorities. To this aim, we have formulated six research questions (section 1.2.2) concerning the models of the repair shops (questions 2 and 5) and concerning the extensions of the spare parts supply models (questions 1, 3, 4 and 6). To be able to answer the research questions, we have developed several tools. In the next subsection, we briefly discuss these tools. Next, we present our conclusions by answering the research questions that we posed in Chapter 1 (9.3). Finally, we provide some directions for further research (9.4).

9.2 Tools

First, we have built a simulation model of the multi-echelon, multi-indenture spare parts supply system to validate the mathematical models developed in this thesis. We used the simulation software eM-Plant^{TM} (Tecnomatix, 2001) because of its object-oriented nature, which allows for easy extension and modification (number of echelons and indentures, repair priorities). We used the object classes for the repair shops separately to validate the queueing models that we developed.

Next, using Borland Delphi, we have developed a library of numerical procedures to estimate the performance measures of our queueing models. We used these algorithms to build a software prototype of VARI-METRIC with finite capacity repair shops. This prototype includes the optimisation heuristics for stocks, repair capacities and repair priorities optimisation that we proposed in this thesis.

We used our simulation models and VARI-METRIC prototype to execute several series of numerical experiments, which helped us to draw conclusions.

9.3 Conclusions

Let us present the conclusions of this thesis by answering our research questions:

1. *Which model for finite capacity repair shops is appropriate to be embedded in the (VARI-)METRIC framework (Chapter 2)?*

We examined the requirements for a finite capacity repair shop model that can be embedded in the (VARI-)METRIC framework. We chose for the multi-class *M/M/k* queue to model repair shops. In case of repair priorities, we use two priority classes with a fixed assignment of items to priority classes per repair shop and a preemptive resume queueing discipline. As performance characteristics, we need the first two moments of the number of items in the queueing system for each item type.

2. *How do we analyse the selected repair shop model and how do we derive the performance characteristics needed for (VARI-)METRIC (Chapter 3)?*

We derived the steady state equations for the multi-class *M/M/k* queue (FCFS) and we presented an exact method to solve these equations. Using the steady state probabilities, we can derive the first two moments of the number of items in the system for each item type. However, the exact analysis requires solving matrix equations of high dimensions. Therefore, we derived approximations with smaller dimensions of the matrix equations. The approximations require solving linear equations with dimensions less then or equal to the number of item classes. The approximation errors of the mean and variance of the number of items in the system are at most 4% and 7% respectively for utilisation rates exceeding 0.85. However, the average approximation errors of these two performance estimators in our numerical experiments are 1.8% and 3.6% respectively. The errors decrease with the increase of repair shop utilisation rate. Finding an exact solution requires the solution of linear equations with much higher dimension, which grows with the number of servers.

3. *To which extent can we improve (VARI-)METRIC by introducing our queueing model for the finite capacity repair shops (Chapter 4)?*

The queueing results have allowed us to extend the existing (VARI-) METRIC models with finite capacity of repair shops and to improve the estimates of the system availability. In our numerical experiments, we found that the average absolute deviation between estimated and simulated availability is less than 1% and decreases with the repair shop utilisation rate. The classical VARI-METRIC method overestimates the system availability with 4.8% on average and leads to incorrect stock levels distributions. Our improved VARI-METRIC method underestimates the system availability, thereby guaranteeing that the required system availability will be reached.

Finite repair capacities have also impact on the stock distribution. We have found that the stocks of the highest indenture items at the most downstream locations increase with the repair shop utilisations. In other words, the stocks are shifting such, that more ready-for-use modules are stocked close to the installed base.

4. *How can we choose an appropriate combination of spare part inventories and repair capacities (Chapter 5)?*

Having the improved (VARI-)METRIC model, we were capable to include repair capacities as decision variables and to allocate budget between spare part inventories and repair capacities. For a better optimisation, we have provided a simple procedure to analyse repair shops with a non-integer number of servers, representing part time or overtime work. Our experimental results have shown that the optimal utilisation of repair shops varies between 0.8 and 0.95. These utilisation rates increase with the price of servers relative to the price of items and with the number of servers. An interesting result is that the optimal distribution of budget between inventory and service capacities hardly depends on the prices of repair capacity relative to the price of items. That is, the increase in the price of repair capacity leads to a decrease in repair capacity and an increase in spare part inventories, but the percentage of budget spent to both inventories and capacities remains almost constant.

5. *How do we extend and analyse our repair shop model to deal with repair priorities (Chapter 6 and 7)?*

Analogously to the multi-class *M/M/k* queue, we derived the steady state equations for the multi-class *M/M/k* queue with two priority classes and preemptive priorities, where each priority class may consist of item subclasses with different arrival and service rates. To solve these equations approximately, we assumed that the probability to have more than t items in the queue in total is equal to zero. Then, choosing the parameter t high enough, we obtain quite good approximations of the key performance characteristics. Our numerical experiments have shown that it is enough to choose $t \approx 20$ to obtain an approximation error less than 1% for the first and second moment of the number of low priority items in the system. We found that the approximations have a higher accuracy for high utilisation rates (0.85-0.95). The presented iterative method has also shown good results being applied to the multi-class, multi-server queues without priorities, which shows its generality.

As a side result, we derived an exact solution procedure for a variant in which high priority jobs are outsourced if they cannot be served immediately upon arrival (Chapter 7).

For this variant, we were capable to analyse both variants with preemptive and non-preemptive priorities. Further, the approximations provided for the non-priority multi-class, multi-server model can also be applied to the multi-class queueing problem with non-preemptive priorities and with outsourcing.

6. *To which extent can we reduce inventory investment by introducing repair priories, and how should we allocate items to priority classes then (Chapter 8)?*

To answer this last research question, we embedded our priority queueing model in VARI-METRIC and we developed a few heuristics for simultaneous inventory optimisation and priority assignment. Using this tool, we could examine the impact of priority assignment.

From our numerical experiments, we found that repair priorities provide an opportunity to reduce investments in repairable spare parts networks. We found the highest investments reduction when items with totally different costs and service times characteristics share the same repair capacity. An important indicator in this respect is the ratio between the price and the mean repair time of an item. In our numerical experiments, we found an investment reduction up to 73% when single server repair shops with utilisation of 0.9 are shared by five items whose price / service time ratios differ by a factor 67. In these cases, the items with the highest price / service time ratio receive high priority and therefore their stock levels can be drastically reduced. As a consequence, the total investment reduces drastically as well. However, an investment reduction of 10%-20% is more common.

Further, we used the property that items with high ratio between price and mean service time should have high repair priorities to construct a sequential priority assignment / stock optimisation heuristic. This heuristic first assigns items to priority classes, minimising the total costs of items in repair, and next optimises spare part stock levels once for the chosen priority assignment. We found that this heuristic usually provides similar results as a nested procedure where the priority assignment is optimised by a local search procedure and stock levels are optimised for each step in this local search procedure separately. Usually, our sequential heuristic leads to the same inventory reduction compared to a non-priority system. In a few cases, the inventory reduction was up to 10% less. On the other hand, our sequential heuristic usually does not need too much computation time. For example, the nested optimisation procedure requires up to 26 hours on a Pentium III – 800MHz PC for a three indenture system with 90% server utilisation, while the sequential procedure finds an 9% smaller reduction in 6 minutes.

9.4 Further research

We see possibilities for further research along three lines: (1) further development of multi-class queueing models (extensions and approximations), (2) further development of VARI-METRIC with finite repair capacities (extensions and approximations), and (3) other applications of multi-class, multi-server queues. We will discuss these three research lines in more detail below.

9.4.1 Multi-class queues

In this area, we distinguish three research directions.

First, the analysis of multi-class queues requires the solution of matrix equations with high dimensions. Our current approximations are based on finding some of eigenvalues without solving the whole eigenvalue problem. However, these approximations are only available for the non-priority model and the non-preemptive priority model with outsourcing. Besides, the approximations become worse for larger systems (with many servers or priority classes) and with lower utilisation rates. Therefore, the first direction of the future research is the further development of approximations for priority and non-priority models, particularly for larger systems.

A second research direction is the extension of our model with two priority groups to more priority groups. We can extend of the multi-class priority model for preemptive priority and without outsourcing using the following approximate approach. The performance of the two highest priority groups can be estimated without taking into account lower priority items. Next, the performance of lower priority groups can be estimated iteratively by joining all groups with higher priority into one class and ignoring other priority groups. We expect that this iterative procedure will give a good approximation, although we still have to test it.

A third direction is to develop a solution for non-preemptive priority queues without outsourcing. This will extend the range of processing disciplines, which we can use in our spare parts model, and it makes this model more flexible. This priority model is technically more complicated than the models as discussed in this thesis and it requires the solution of larger matrix equations.

As a fourth point of further research, we should mention, of course, other possible applications of the obtained queueing models different from the application presented here. These can be applications in:

- other supply chain models
- production systems

- telecommunication systems

9.4.2 VARI-METRIC with finite repair capacities

Concerning the VARI-METRIC models with finite capacity repair shops, there are four areas to be explored.

First, the analysis of the spare part network model with repair priorities is done for the repair shops with a preemptive priority rule and two priority classes. An open issue is which further investment reduction is possible using more detailed priorities (more than two priority groups).

Second, we considered a fixed assignment of items to priority classes, because we focus on the tactical decision of initial inventory investment. At the operational level, the system performance can possibly be improved taking into account information on the specific system state at a specific point in time (cf. Hausman and Scudder, 1982; Pyke, 1990). For example, the current backorder and inventory levels are relevant when deciding on the next job that will be taken into repair. Further research is needed on such operational dynamic priority rules and their impact on the system availability, given the stock levels from our static priority models. This can be done by testing such priority rules using simulation.

Third, the outsourcing policy presented in this thesis assumes outsourcing of high priority items if they cannot be served upon arrival. One can think of other outsourcing criteria, e.g. based on the queue of high priority items or the waiting time that has expired since arrival. However, the analysis of such queueing models is not straightforward.

Fourth, the presented spare parts models do not take into account the specifics of preventive maintenance. This is relevant for our finite capacity models if the repair capacities are used for both preventive maintenance and corrective maintenance. The timing and capacity requirement for preventive maintenance is known in advance within certain limits. This is not true for corrective maintenance. Also, corrective maintenance tasks will generally be more urgent, cf. Keizers (2000). Preventive maintenance can influence the demand process as well, because demands become more predictable.

Fifth, we have found that the improved VARI-METRIC model still has approximation errors for the average system availability in case of high repair shop utilisations. This error is caused by correlations between the numbers of items in the pipeline for various item types sharing the same repair capacity. Therefore, one of the directions of further research is reduction of the approximation error of the system availability.

References

1. Aboud, N.E., 1996, "The Markovian two-echelon repairable item provisioning problem", *Journal of the Operational Research Society* 47, 284-296.

2. Adan, I.J.B.F., 2000, *Lecture notes on Queueing Theory,* http://www.win.tue.nl/~iadan/s1/

3. Adan, I.J.B.F., M.J.A. van Eenige and J.A.C. Resing, 1996, “Fitting discrete distributions on the first two moments”, *Probability in the Engineering and Informational Sciences* 9, 623-632.

4. Adan, I. and J. van der Wal, 1998, *Difference and differential equations in Stochastic Operations Research*, tutorial, Eindhoven University of Technology, The Netherlands.

5. Albright, S.C. and A. Gupta, 1993, “Steady-state approximation of a multi-echelon multi-indentured repairable-item inventory system with a single repair facility”, *Naval Research Logistics* 40(4), 479-493.

6. Albright, S.C. and A. Soni, 1988, “Markovian multi-echelon repairable inventory system”*, Naval Research Logistics Quarterly* 35, 49-61.

7. Alfredsson, P. and J. Verrijdt, 1999, "Modeling emergency supply flexibility in a two-echelon inventory system", *Management Science* 45, 1417-1431.

8. Ashayeri, J., R. Heuts, A. Jansen and B. Sczerba, 1996, "Inventory management of repairable service parts for personal computers", *International Journal of Operations and Production Management* 16(2), 74-94.

9. Avsar Z.M. and W.H.M. Zijm, 2000, “Resource-constrained two-echelon inventory models for repairable item systems”, working paper, University of Twente, Enschede, The Netherlands.

10. Axsäter, S., 1990, "Modeling emergency lateral transshipments in inventory systems", *Management Science* 36, 1329-1338.

11. Bertsimas, D. and G. Mourtzinou, 1997, "Multiclass Queueing Systems in Heavy Traffic: An Asymptotic Approach Based on Distributional and Conservation Laws", *Operations Research*,vol. 45(3), 470-487.

12. Bramson, M., 1998, State space collapse with application to heavy traffic limits for multiclass queueing networks, *Queueing Systems* 30, 89-148.

13. Büyükkurt, M.D. and M. Parlar, 1993, "A comparison of allocation policies in a two-echelon repairable-item inventory model", *International Journal of Production Economics* 29, 291-302.

14. Buzacott, J.A. and J.G. Shanthikumar, 1993, *Stochastic models of manufacturing systems*, Prentice Hall, Englewood Cliffs, NJ

15. Buzen, J.P. and A.B. Bondi, 1993, "The response times of priority classes under preemptive resume in *M/M/m* queues", *Operations Research* 31(3), 456-465.

16. Cho, D.I., 2001, "An approximation to a dynamic inventory model for repairable items", *International Journal of Sytems Science* 32(7), 879-888.

17. Cho, D.I., R. Abad and M. Parlar, 1996, "A Markov decision process approach to repairable-item inventory problems", in: *Probability Models and Statistics*, New Age International, New Delhi.

18. Chua, R.C.H., G.D. Scudder and A.V. Hill, 1993, "Batching policies for a repair shop with limited spares and finite capacity", *European Journal of Operational Research* 66, 135-147.

19. Cohen, J.W., 1982, “On the *M/G/*2 queueing model”, *Stoch. Proc. Appl*., 12, 231-248

20. Cohen, M.A., P.R. Kleindorfer and H.L. Lee, 1986, "Optimal stocking policies for low usage items in multi-echelon inventory systems", *Naval Research Logistics Quarterly* 33, 17-38.

21. Cohen, M.A., P.R. Kleindorfer and H.L. Lee, 1988, "Service constrained (s, S) inventory systems with priority demand classes and lost sales", *Management Science* 34, 482-499.

22. Cohen, M.A., P.R. Kleindorfer and H.L. Lee, 1989, "Near-optimal service constrained stocking policies for spare parts", *Operations Research* 37, 104-117.

23. Cohen, M.A., P.R. Kleindorfer, H.L. Lee and A. Tekerian, 1990 , "Optimizer: IBM's multi-echelon inventory system for managing service logistics", *Naval Research Logistics* 39, 561-577.

24. Cohen, M.A., P.R. Kleindorfer, H.L. Lee and D.F. Pyke, 1992 , "Multi-item service-constrained (s, S) policies for spare parts logistics systems", *Naval Research Logistics* 39, 561-577.

25. Cohen, M.A., Y-S. Zheng and V. Agrawal, 1997, "Service parts logistics: a benchmark analysis"*, IIE Transactions* 29, 627-639.

26. Cohen, M.A., Y-S Zheng and Y. Wang, 1999, "Identifying opportunities for improving Teradyne service-parts logistics system", *Interfaces* 29(4), 1-18.

27. Dada, M., 1992, "A two-echelon inventory system with priority shipments", Management Science 38(8), 1140-1153.

28. Dekker, R., M.J. Kleijn and P.J. de Rooij, 1998, "A spare parts stocking policy based on equipment criticality", *International Journal of Production Economics* 56-57, 67-77.

29. Diaz, A. and M.C. Fu, 1997, "Models for multi-echelon repairable item inventory systems with limited repair capacity", *European Journal of Operational Research* 97, 480-492.

30. Fisher, W.W. amd J.J. Brennan, 1986, "The performance of cannibalization policies in a maintenance system with spares, repair and resource constraints", *Naval Research Logistics Quarterly* 31, 1-15.

31. Fleishmann, M., J.M. Bloemhof-Ruwaard, R. Dekker, E. van der Laan, J.A.E.E. van Nunen and L.N. van Wassenhove, 1997, "Quantitative models for reverse logistics: a review", *European Journal of Operational Research* 103, 1-17.

32. Gail, H.R., S.L. Hantler, and B.A. Taylor, 1988, "Analysis of a non-preemptive priority multiserver queue", *Advances in Applied Probability* 20(4), 852-879.

33. Gail, H.R., S.L. Hantler, and B.A. Taylor, 1992, "On preemptive Markovian queue with multiple servers and two priority classes", *Mathematics of Operations Research* 17(2), 365-391.

34. Golub, G.H., C.F. van Loan. 1996, *Matrix computations (Third edition).* John Hopkins university Press, Baltimore and London

35. Grahovac, J. and A. Chakravarty, 2001, "Sharing and lateral transshipment of inventory in a supply chain with expensive low-demand items", *Management Science* 47(4), 579-594.

36. Graves, S.C., 1985, “A multi-echelon inventory model for a repairable item control with one-for-one replenishment”, *Management Science* 31, 1247-1256.

37. Gross, D. and C.M. Harris, 1998, *Fundamentals of queueing theory*, Wiley, New York

38. Gross, D. and J.F. Ince, 1978, “Spares provisioning for repairable items: Cyclic queues in light traffic”, *AIIE Transactions* 10(3), 1978, 307-314.

39. Gross, D., D.R. Miller and R.M. Soland, 1983, “A closed queueing network model for multi-echelon repairable item provisioning”, *IIE Transactions* 15(4), 344-352.

40. Guide Jr, V.D.R. and R. Srivastava, 1997. “Reparable inventory theory: Models and applications”, *European Journal of Operational Research* 102, 1-20.

41. Guide Jr, V.D.R., R. Srivastava and M.E. Kraus, 2000, "Priority scheduling policies for repair shops", *International Journal of Production Research* 38(4), 929-950.

42. Gupta, A. and S.C. Albright, 1992, “Steady-state approximations for multi-echelon multi-indentured repairable-item inventory system”, *European Journal of Operational Research* 97(3), 340-353.

43. Haas, H.F.M. de, 1995, *The coordination of initial stock and flexible manpower in repairable item systems*, Ph.D. thesis, Eindhoven University of Technology.

44. Haas, H.F.M. de and J.H.C.M. Verrijdt, 1997, “Target setting for the departments in an aircraft repairable item system”, *European Journal of Operational Research* 99, 596-602.

45. Hall, R.W., 1991, *Queueing methods for services and manufacturing*, Prentice Hall, Englewood Cliffs, NJ

46. Harten, A. van and A. Sleptchenko, 2000, “On multi-class, multi-server queueing and spare part management”, working paper, University of Twente, Enschede, The Netherlands (submitted for publication).

47. Hausman, W., and G. Scudder, 1982, "Priority scheduling rules for repairable inventory systems", *Management Science* 28, 1215-1232.

48. Hooghiemstra, G., M. Keane and S. van de Ree, 1988, "Power series for stationary distributions of coupled processor models", *SIAM Journal on Applied Mathematics*, 48(5), 1159-1166.

49. Hopp, W.J., R.Q. Zhang and M.L. Spearman, 1999, "An easily implementable hierarchical heuristic for a two-echelon spare parts distribution system", *IIE Transactions* 31, 977-988.

50. Inderfurth, K., and R.H. Teunter, 2001, "Production planning and control of closed-loop supply chains", Econometric Institute report no. EI 2001-39, Erasmus University Rotterdam, The Netherlands.

51. Kabir, A.B.M.Z., and A.S. Al-Olayan, 1996a, "A stocking policy for spare part provisioning under age based preventive replacement", *European Journal of Operational Research* 90, 171-181.

52. Kabir, A.B.M.Z., and A.S. Al-Olayan, 1996b, "Joint optimization of age replacement and continuous review spare provisioning policy", *International Journal of Production Economics* 14(7), 53-69.

53. Kanadia, A.S., M.F. Kazmi and A.C. Mitchell, 1984, "Analysis of a finite capacity non-preemptive priority queue", *Computers and Operations Research* 11(3), 337-343.

54. Kao, E.P.C. and K.S. Narayanan, 1990, "Computing steady state probabilities of a nonpreemptive priority queue", *ORSA Journal on Computing* 2, 211-218.

55. Kao, E.P.C. and S.D. Wilson, 1999, "Analysis of nonpreemptive priority queues with multiple servers and two priority classes", *European Journal of Operational Research* 118, 181-193.

56. Keizers, J., 2000, *Subcontracting as a capacity management tool in multi-project repair shops*, PhD. thesis, Eindhoven University of technology, The Netherlands.

57. Keizers, J, I. Adan and J.van der Wal, 2001, "A Queueing Model for Due Date Control in a Multiserver Repair Shop", *Naval Research Logistics* 48(4), 281-292.

58. Kella, O. and U. Yechiali., 1985, "Waiting times in the non-preemptive priority *M/M/c* queue", *Communications in statistics - Stochastic models* 1, 257-262.

59. Kennedy, W.J., J.W. Patterson and L.D. Fredendall, 2002, “An overview of recent literature on spare parts inventories”, *International Journal of Production Economics* 76, 201-215

60. Kim, J-S, K-C Shin and S-K Park, 2000, “An optimal algorithm for repairable-item inventory systems with depot spares”, *Journal of the Operational Research Society* 51, 350-357.

61. Klein Haneveld, W.K. and R.H. Teunter, 1997, "Optimal provisioning for slow moving spare parts with small lead times", *Journal of the Operational Society*, 184-194.

62. Kleinrock, L., 1975, *Queueing systems*, Wiley, New York

63. Kumar, N, P. Vrat and Sushil, 1994, "A simulation study of unit exchange spares management of diesel locomotives in the Indian Railways", *International Journal of Production Economics* 33, 225-236.

64. Law, M.A. and W.D. Kelton, 2000, *Simulation modeling and analysis*. Third edition. McGraw-Hill, Singapore.

65. Lee, H.L., 1987, "A multi-echelon inventory model for repairable items with emergency lateral transshipments", *Management Science* 33, 1302-1316.

66. Miller, B.L., 1974, "Dispatching from depot repair in a recoverable item inventory system: on the optimality of a heuristic rule", *Management Science* 21(3), 316-325.

67. Mitrani, I. and P.J.B. King, 1981, "Multiprocessor systems with preemptive priorities", *Performance Evaluation* 1, 118-125.

68. Muckstadt, J, 1973, "A model for a multi-item, multi-indenture inventory system", *Management Science* 20, 472-481.

69. Palm, C., 1938, "Analysis of the Erlang traffic formulae for busy-signal arrangements", *Ericsson Technics* 4, 39-58.

70. Papadopolous H.T., C. Heavey and J. Browne, 1993, *Queueing theory in manufacturing systems analysis and design*, Chapman & Hall, London

71. Perlman, Y., A. Mehrez and M. Kaspi, 2001, “Setting expediting repair policy in a multi-echelon repairable-item inventory system with limited repair capacity”, *Journal of the Operational Research Society* 52, 198-209.

72. Pinedo, M. and X. Chao, 1999, *Operations scheduling with applications in manufacturing and services*, McGraw-Hill, Boston.

73. Pyke, D.F., 1990, “Priority repair and dispatch policies for repairable-item logistics systems”, *Naval Research Logistics* 37, 1-30.

74. Rustenburg, W.D., 2000, *A system approach to budget-constrained spare parts management*, PhD thesis, University of Twente, Enschede, The Netherlands.

75. Rustenburg, W.D., G.J van Houtum and W.H.M Zijm, 2000, "Spare parts management for technical systems: resupply of spare parts under limited budgets", *IIE Transactions* 32(10), 1013-1026

76. Rustenburg, W.D., G.J. van Houtum and W.H.M. Zijm, 2001, “Spare parts management at complex technology-based organizations: An agenda for research”, *International Journal of Production Economics* 71, 177-193.

77. Saad, Y. 1992, *Numerical methods for large eigenvalue problems: Theory and algorithms*. John Willey and Sons, New York.

78. Sarker, R. and A. Haque, 2000, "Optimization of maintenance and spare provisioning policy using simulation", *Applied Mathematical Modelling* 24, 751-760.

79. Scudder, G, 1984, "Priority scheduling and spares stocking policies for a repair shop: the multiple failure case", *Management Science* 30, 739-749.

80. Scudder, G., and W. Hausman, 1982, "Spares stocking policies for repairable inventory systems", *Naval Research Logistics Quarterly* 29, 303-322.

81. Seyboth, A., 1997, "A simulation of inventory management for spare part supply of trucks", *American Journal of Mathematical and Management Sciences* 17 (3,4), 263-297.

82. Sherbrooke, C.C., 1968, “METRIC: Multi-Echelon Technique for Recoverable Item Control”, *Operations Research* 16, 122-141.

83. Sherbrooke, C.C., 1986, "VARI-METRIC: improved approximations for multi-indenture, multi-echelon availability models", *Operations Research* 34, 311-319.

84. Sherbrooke, C.C., 1992, *Optimal inventory modelling of systems: Multi-echelon techniques*, Wiley, New York.

85. Shibuya, T., T. Dohi and S. Osaki, 1998, "Optimal continuous review policies for spare part provisioning with random lead times", *International Journal of Production Economics* 55, 257-271.

86. Slay, F.M., 1984, “VARY-METRIC: An Approach to Modeling Multi-Echelon Resupply when the Demand Process is Poisson with Gamma Prior”, Report AF301-3, Logistic Management Institute, Washington, D.C.

87. Sleptchenko, A., M.C. van der Heijden and A. van Harten, 2002a, “Effects of finite repair capacity in multi-echelon, multi-indenture service part supply systems”, *International Journal of Production Economics* 79(3), 209-230.

88. Sleptchenko, A., A. van Harten and M.C. van der Heijden, 2002b, "Analyzing multi-class, multi-server queueing systems with preemptive priorities", working paper University of Twente, Faculty of Technology and Management, Enschede, the Netherlands (submitted for publication).

89. Sleptchenko, A., A. van Harten and M.C. van der Heijden, 2002c, “Trade-off between inventory and repair capacity in spare parts networks”, accepted for publication in *Journal of Operational Research Society.*

90. Smit, J.H.A. de, 1983a, “The queue *GI/M/s* with customers of different types or the queue *$GI/H_m/s$*”, *Advanced Applied Probabilities* 15, 392-419.

91. Smit, J.H.A. de, 1983b, “A numerical solution for the multi-server queue with hyper-exponential service time”, *Operations research letters* 2(5), 217-224.

92. Tecnomatix, 2000, *eM-Plant object manual and reference manual, version 5.5*, Tecnomatix Technologies GmbH & Co., Germany.

93. Tedone, M.J., "Repairable part management", *Interfaces* 19(4), 61-68.

94. Tijms, H.C., 1994, *Stochastic models: An algorithmic approach*. John Willey and Sons, Chichester.

95. Verrijdt, J., I. Adan and A.G..de Kok, 1998, "A trade off between emergency repair and inventory investment", *I.I.E. transactions* 30(2), 119-132.

96. Verrijdt, J.H.C.M., 1997, *Design and control of service part distribution systems*, Ph.D. thesis, Eindhoven University of Technology, The Netherlands.

97. Wagner, D., 1997, "Waiting time of a finite-capacity multi-server model with non-preemptive priorities", *European Journal of Operational Research*, 102, 227-241.

98. Wagner, D., 1997, "Analysis of mean values of a multi-server model with non-preemptive priorities and non-renewal input", *Communications in statistics - Stochastic models* 13(1), 67-84.

99. Wagner, D., 1998, "A finite-capacity multi-server multi-queueing model with non-renewal input", *Annals of Operational Research* 79, 63-82.

100. Walker, J., "Base stock level determination for 'insurance type' spares", 1997, *International Journal of Quality and Reliability Management* 14(6), 569-574.

101. Whitt, W., 1993, “Approximations for the *GI/G/c* queue”, *Production and Operations Management* 2, 144-161.

102. Williams, R.J., 1998, “Diffusion approximations for open class queueing networks: sufficient conditions involving state space collapse”, *Queueing systems* 30, 27-88.

103. Winston, W., 1997, *Operations Research Applications and Algorithms*, 3rd Edition, Duxbury Press.

Appendices

Appendix A. Notations

The system layout:

N^{ech}	—	number of echelons in the system.
ECH(n)	—	set of locations belonging to the echelon n.
N^{loc}	—	number of locations in the system.
CUS(m)	—	set of customers (supported locations) of location *m*.
SUP(m)	—	the location that supplies items to location *m* (i.e. the direct supplier of location *m*)
N^{ind}	—	number of levels (indentures) in the item hierarchy.
IDN(i)	—	set of items belonging to indenture *i*.
N^{item}	—	number of items in the system.
SA(j)	—	set of subassemblies of item *j*.
AS(j)	—	set of assemblies, which have item *j* as a subassembly.

Repair shop characteristics

h	—	repair shop index, $h = 1,\dots,N^{shop}$, where N^{shop} denotes the total number of repair shops in the system;
SH(m)	—	set of repair shops at location *m*;
p_{mjh}^{shop}	—	probability that item *j* will be repaired by repair shop *h*, given that it will be repaired at location *m*;
λ_{mjh}	—	arrival rate of item *j* at repair shop *h* at location *m*.
ST_{mhj}	—	repair time of item *j* at repair shop *h* at location *m*, a random variable with mean $\mathrm{E}[ST_{mhj}]$ and squared coefficient of variation $C_{ST_{mhj}}^{2}$, where the

		coefficient of variation is defined as the ratio between the standard deviation and the mean
Q_{mhj}	—	number of item *j* in the repair queue of repair shop *h* at the location *m*.
R_{mhj}	—	number of type *j* under repair (in queue and in service) in repair shop *h* at location *m*.
k_{hm}	—	number of servers in repair shop *h* at location *m*, $\overline{k}$ — matrix of all repair capacities in the system.

Routing characteristics

p_{mj}^{rep}	—	probability that item *j* can be repaired at location *m*.
p_{mjk}^{cause}	—	probability that a failure of item *j* at location *m* is caused by a failure of subassembly *k* ($k \in SA(j)$)

Other notation VARI-METRIC notations.

st_{mj}	—	stock level of item *j* at location *m*. $\overline{st}$ – matrix of all stock levels in the system.
PL_{mj}	—	number of type *j* items in the pipeline at the location *m*, i.e. under repair or in resupply.
O_{mj}	—	Order-and-ship time of item *j* to location *m* from its supplier, a random variable with mean $E\left[O_{mj}\right]$ and squared coefficient of variation $C_{O_{mj}}^{2}$.
BO_{mj}	—	number of backorders for item *j* at location *m*: $BO_{mj} = \max\left\{PL_{mj} - ST_{mj}, 0\right\}$
c_{j}^{stock}	—	price of item *j*.
c_{hm}^{cap}	—	price of repair capacity in repair shop *h* at location *m*.

Notations of MCMS queueing systems

N	—	number of items in the queueing system.
λ_i	—	arrival rate of item *i*.
μ_i	—	service rate of item *i*.

ρ_i	— utilisation rate of item *i*, $\rho_i = \lambda_i / \mu_i$
Λ, μ	— overall arrival and service rates, i.e. $\Lambda = \sum_{i=1}^{N} \lambda_i$, and $\mu = \Lambda \Big/ \sum_{i=1}^{N} \frac{\lambda_i}{\mu_i} = \Lambda / k\rho$, where the utilisation rate is $\rho = \Lambda / k\mu$.
a_i	— fractions of arrival rates, i.e. $a_i = \lambda_i / \Lambda$.
δ_i	— perturbations of service rates, i.e. $(1+\delta_i) = \mu_i / \mu$.
$\overline{s}$	— vector containing the number of items in service per item class.
$\overline{w}$	— vector containing the number of items in the queue per item class.
$P_n(\overline{w},\overline{s})$	— probability of system state $(\overline{w},\overline{s})$ with *n* items in the system
$\mu(\overline{s})$	— sum of service rates of all items in service, i.e. $\mu(\overline{s}) = \sum_{i=1}^{N} s_i \mu_i$.
$\overline{\delta}(\overline{s})$	— average perturbation of service rates of all items in service, i.e. $\overline{\delta}(\overline{s}) = \frac{1}{k} \sum_{i=1}^{N} s_i \delta_i$.
x_i	— the *i*th component of any vector $\overline{x}$.
$\overline{e}_i$	— a vector of dimension *N* with component *i* equal to 1 and all other components equal to 0; this vector is used to indicate the changes in vectors $\overline{w}$ and $\overline{s}$ during transitions from state to state.
e_{ij}	— denotes the *j*th component of the vector $\overline{e}_i$, so e_{ij} = 1 if *i* = *j* and 0 otherwise.
$\lvert\overline{x}\rvert$	— denotes the sum of all components of any vector $\overline{x}$.
$\mathbf{P}_n$	— matrix of server states probabilities given that there are *n* items in the queueing system.
$\mathbf{Z}^{-1}$	— matrix of transition probabilities from server states with *n* – 1 server to *n* servers ($n > k$).
$\mathbf{Q}_n$	— matrix of transition probabilities from server states with *n* – 1 server to *n* servers ($n \le k$).

$\mathbf{I}_n^{i,m}$ — diagonal matrix indicating states with m items of type i with 1 and 0 elsewhere given n items in the system.

$\chi_n^{s_i}$ — diagonal matrix indicating numbers of items of type i in the service in corresponding service state and given n numbers in the system

R_i — numbers of items of type i in the system.

S_i — numbers of items of type i in the service.

q_i — numbers of items of type i in the queue.

Notations of MCMS priority queueing systems

N^h (N^l) — numbers of high (low) priority items in the system.

λ_i^h, (λ_j^l) — arrival rates of high (low) priority jobs of class *i* (*j*) into the system.

μ_i^h, (μ_j^l) — service rates of high (low) priority jobs of class *i* (*j*).

ρ_i^h, (ρ_j^l) — utilisation rates of high (low) priority jobs *i* (*j*) the system, $\rho_i^h = \lambda_i^h / \mu_i^h$ ($\rho_i^l = \lambda_i^l / \mu_i^l$).

Λ^h, (Λ^l), μ^h, (μ^l) — overall arrival and service rates of high (low) priority class, i.e. $\Lambda^h = \sum_{i=1}^{N^h} \lambda_i^h$ ($\Lambda^l = \sum_{i=1}^{N^l} \lambda_i^l$) and $\mu^h = \Lambda^h \big/ \sum_{i=1}^{N^h} \frac{\lambda_i^h}{\mu_i^h} = \Lambda^h / k\rho^h$, ($\mu^l = \Lambda^l \big/ \sum_{i=1}^{N^l} \frac{\lambda_i^l}{\mu_i^l} = \Lambda^l / k\rho^l$), where the overall utilisation rates for each priority class are $\rho^h = \Lambda^h / k\mu^h$, $\rho^l = \Lambda^l / k\mu^l$ and the total utilisation rate is $\rho = \rho^h + \rho^l$.

a_i^h, (a_j^l) — fractions of arrival rates, i.e. $a_i^h = \lambda_i^h / \Lambda^h$, ($a_j^l = \lambda_j^l / \Lambda^l$).

δ_i^h, (δ_j^l) — perturbations of service rates, i.e. $\left(1+\delta_i^h\right) = \mu_i^h / \mu^h$, $\left(1+\delta_j^l\right) = \mu_j^l / \mu^l$.

γ — ratio f the service rates for high and low priority items, i.e. $\gamma = \mu^l / \mu^h$.

n_i (m_j)	— number of high (low) priority items of type i (j) and by n (m) the total number of high (low) priority items in the system.
$\overline{s}^h$ ($\overline{s}^l$)	— vectors containing the number of high (low) priority items in service per item class.
$\overline{w}^h$ ($\overline{w}^l$)	— vectors containing the number of high (low) priority items in the queue waiting for first service per item class (these vectors *exclude* postponed items).
$\overline{r}^l$	— vector containing the number of postponed low priority items per item class.
$P_{n,m}\left(\overline{w}^h,\overline{s}^h,\overline{w}^l,\overline{s}^l,\overline{r}^l\right)$	— probability of system state $\left(\overline{w}^h,\overline{s}^h,\overline{w}^l,\overline{s}^l,\overline{r}^l\right)$ with n (m) items of high (low) priority in the system
$\mu\left(\overline{s}^h,\overline{s}^l\right)$	— sum of service rates of all items in service, i.e. $\mu\left(\overline{s}^h,\overline{s}^l\right)=\sum_{i=1}^{N^h} s_i^h\mu_i^h+\sum_{i=1}^{N^l} s_i^l\mu_i^l$.
$\overline{\delta}\left(\overline{s}^h,\overline{s}^l\right)$	— average perturbation of service rates of all items in service, i.e. $\overline{\delta}\left(\overline{s}^h,\overline{s}^l\right)=\frac{1}{k}\left(\sum_{i=1}^{N^h} s_i^h\delta_i^h+\sum_{i=1}^{N^l} s_i^l\delta_i^l\right)$.
x_i	— the ith component of any vector $\overline{x}$.
$\overline{e}_i^h$ ($\overline{e}_i^l$)	— a vector of dimension N^h (N^l) with component i equal to 1 and all other components equal to 0; this vector is used to indicate the changes in vectors $\overline{w}^h$ and $\overline{s}^h$ ($\overline{w}^l$, $\overline{s}^l$ and $\overline{r}^l$) during transitions from state to state.
e_{ij}^h (e_{ij}^l)	— denotes the jth component of the vector $\overline{e}_i^h$ ($\overline{e}_i^l$), so e_{ij}^h (e_{ij}^l) = 1 if $i = j$ and 0 otherwise.
$\lvert\overline{x}\rvert$	— denotes the sum of all components of any vector $\overline{x}$.
$\mathbf{P}_{n,m}$	— vector server states probabilities given that there are n(m) high(low) priority items in the queueing system and $n > k$.

$\mathbb{P}_t$	— vector server states probabilities given that there are t items in the queueing system and there is no high priority items in the queue.
$v(\xi)$	— generating vector-function of server states probabilities given there is no high priority items in the queue.
$\mathbb{Q}_t$	— matrix of transition probabilities from server states with $t - 1$ items in the system to t items in the system (no high priority items in the queue).
χ_n^x	— diagonal matrix indicating numbers x in the corresponding service state given n numbers in the system
R_i^h	— numbers of high priority items of type i in the system.
R_i^l	— numbers of low priority items of type i in the system.
S_i^h	— numbers of high priority items of type i in the service.
S_i^l	— numbers of low priority items of type i in the service.
q_i^h	— numbers of high priority items of type i in the queue.
q_i^l	— numbers of low priority items of type i in the queue.

Appendix B. Optimal priority assignment

In this appendix, we prove that under certain conditions the optimal priority assignment in a preemptive priority queueing system should be done according to the price / service time ratio (cf. Chapter 8). We prove it only for a single-server preemptive priority queueing system with multiple priority groups and one item class per priority group. We follow the line of the proof by Buzacott and Shantikumar (1993) for the non-preemptive priority *M/M/1* queue.

Consider a single-server queue with N item types. Each unit of time that an item of type i spends in the system (waiting and service) costs c_i[3]. The system has multiple priorities and each priority class contains only one type of item (so, we have N priority classes). We denote

[3] If the holding cost rate is identical for all items, we can replace the holding costs per unit of time by the item procurement costs, as we did in Chapter 8.

by $\pi(i)$ the priority index of item i, where item i has priority over item j if $\pi(i) < \pi(j)$. So, the highest priority item has index $\pi(i) = 1$ and the lowest priority item has index $\pi(i) = N$. Let $E\left[R_i^{\pi(i)}\right]$ be the expected number of type i items in the system, given that item i has priority index $\pi(i)$. Then, the total expected cost per unit of time of all items in the system as function of the priority assignment π can be expressed as follows:

$$C = \sum_{i=1}^{N} c_i E\left[R_i^{\pi(i)}\right]$$

Without loss of generality, we assume that all items are numbered in decreasing order of their cost / service time ration $c_i/E[S_i]$. Then, we have to prove that the priority assignment π is optimal if and only if it is done according to the cost / service time ratio, i.e. $\pi(j) = j$ (j = 1,…,N).

To prove this, we proceed as follows. Consider an arbitrary priority assignment $\pi' \neq \pi$. If we can show that the priority assignment π' can always be improved, we have proven that π is optimal.

Given that $\pi' \neq \pi$, there exists at least one index j, such that $\pi'(j) > \pi'(j+1)$, i.e. item j has lower priority than item j + 1. Now we derive a priority assignment π'' from π' by interchanging the priorities of item j and item j + 1, i.e. $\pi''(j)$ = $\pi'(j+1)$ and $\pi''(j+1)$ = $\pi'(j)$. Let us compare the total costs prices of all items in the system for the two priority assignments π' and π'':

$$C' = \sum_{i=1}^{N} c_i E\left[R_i^{\pi'(i)}\right] \text{ and } C'' = \sum_{i=1}^{N} c_i E\left[R_i^{\pi''(i)}\right].$$

The number of type j items in a preemptive priority *M/M/1* queue, given that the priority assignment coincides with the order of items, can be calculated as (Gross and Harris, 1998):

$$E\left[R_j^{\pi'(j)}\right] = \frac{\lambda_j \sum_{i=1}^{\pi'(j)} \rho_{\pi'(i)^{-1}} \Big/ \mu_{\pi'(i)^{-1}}}{\left(1 - \sum_{i=1}^{\pi'(j)} \rho_{\pi'(i)^{-1}}\right)\left(1 - \sum_{i=1}^{\pi'(j)-1} \rho_{\pi'(i)^{-1}}\right)} + \frac{\lambda_j / \mu_j}{\left(1 - \sum_{i=1}^{\pi'(j)-1} \rho_{\pi'(i)^{-1}}\right)}$$

Let us define $W_{j-1} = \sum_{i=1}^{\pi'(j)-1} \rho_{\pi'(i)^{-1}} \big/ \mu_{\pi'(i)^{-1}}$ and $\mathrm{P}_{j-1} = \sum_{i=1}^{\pi'(j)-1} \rho_{\pi'(i)^{-1}}$. Then the difference between the total cost of the priority assignments π' and π'' can be expressed as follows:

$$\begin{aligned}
C' - C'' &= c_j E\left[R_j^{\pi'(j)}\right] + c_{j+1} E\left[R_{j+1}^{\pi'(j+1)}\right] - c_j E\left[R_j^{\pi''(j)}\right] - c_{j+1} E\left[R_{j+1}^{\pi''(j''+1)}\right] \\
&= \frac{c_j \lambda_j \left(W_{j-1} + \rho_j / \mu_j\right)}{\left(1-\mathrm{P}_{j-1} - \rho_j\right)\left(1-\mathrm{P}_{j-1}\right)} + \frac{c_j\, \lambda_j / \mu_j}{\left(1-\mathrm{P}_{j-1}\right)} \\
&\quad + \frac{c_{j+1}\lambda_{j+1}\left(W_{j-1} + \rho_j/\mu_j + \rho_{j+1}/\mu_{j+1}\right)}{\left(1-\mathrm{P}_{j-1} - \rho_j - \rho_{j+1}\right)\left(1-\mathrm{P}_{j-1} - \rho_j\right)} + \frac{c_{j+1}\, \lambda_{j+1}/\mu_{j+1}}{\left(1-\mathrm{P}_{j-1} - \rho_j\right)} \\
&\quad - \frac{c_{j+1}\lambda_{j+1}\left(W_{j-1} + \rho_{j+1}/\mu_{j+1}\right)}{\left(1-\mathrm{P}_{j-1} - \rho_{j+1}\right)\left(1-\mathrm{P}_{j-1}\right)} - \frac{c_{j+1}\, \lambda_{j+1}/\mu_{j+1}}{\left(1-\mathrm{P}_{j-1}\right)} \\
&\quad - \frac{c_j\lambda_j\left(W_{j-1} + \rho_j/\mu_j + \rho_{j+1}/\mu_{j+1}\right)}{\left(1-\mathrm{P}_{j-1} - \rho_{j+1}\right)\left(1-\mathrm{P}_{j-1} - \rho_j - \rho_{j+1}\right)} + \frac{c_j\, \lambda_j/\mu_j}{\left(1-\mathrm{P}_{j-1} - \rho_{j+1}\right)}
\end{aligned}$$

After lengthy computations with the help of Maple$^{\mathrm{TM}}$, we obtain

$$\begin{aligned}
C' - C'' = \left(c_{j+1}\mu_{j+1} - c_j\mu_j\right)\rho_j\rho_{j+1} \Bigg\{ & W_{j-1} \frac{\left(1-\mathrm{P}_{j-i}\right) + \left(1-\mathrm{P}_{j-i} - \rho_j - \rho_{j+1}\right)}{\left(1-\mathrm{P}_{j-i}\right)\left(1-\mathrm{P}_{j-i} - \rho_j - \rho_{j+1}\right)\left(1-\mathrm{P}_{j-i} - \rho_{j+1}\right)\left(1-\mathrm{P}_{j-i} - \rho_j\right)} \\
& + \frac{\left(1-\mathrm{P}_{j-i}\right)\left(1-\mathrm{P}_{j-i} - \rho_{j+1}\right)/\mu_j + \left(1-\mathrm{P}_{j-i}\right)\left(1-\mathrm{P}_{j-i} - \rho_j\right)/\mu_{j+1}}{\left(1-\mathrm{P}_{j-i}\right)\left(1-\mathrm{P}_{j-i} - \rho_j - \rho_{j+1}\right)\left(1-\mathrm{P}_{j-i} - \rho_{j+1}\right)\left(1-\mathrm{P}_{j-i} - \rho_j\right)} \Bigg\}
\end{aligned}$$

It is not difficult to see that the expression in braces is always greater then 0. Then, the costs C' are lower than the costs C'' if and only if the cost / service time ratio $\frac{c_{j+1}}{E\left[ST_{j+1}\right]} = c_{j+1}\mu_{j+1}$ of item *j* + 1 is lower than cost / service time ratio of item *j*.

So, we conclude that we can improve any priority assignment $\pi' \neq \pi$, unless two items have the same cost / service time ratio. In the latter case, we can find an alternative assignment π'' having the same costs by interchanging two items as described above, such that the priority index of item *j* is decreased. By iteration, we either can find an improved assignment of π', or that π' has the same costs as *π*. This completes our proof.

Acknowledgements

Four years ago I have started the research results of which you can find in this thesis. It could not been completed without guidance, support and encouragements of many people.

First of all, I would like to thank my supervisors Prof. Dr. Aart van Harten and Dr. Matthieu van der Heijden for entrusting me this interesting project and continuous guidance during these years. I appreciate all the efforts and time they devoted helping me to finish this work and from whom I learned a lot. I am particularly grateful to Matthieu van der Heijden, for willingness to listen any moment and for his continues help not only in obtaining but also in clear explaining of the results described in this thesis.

During my Ph.D. studies I was honored to be a member of OMST group. I would like to thank the current and former staff of this group for the pleasant and enjoyable company and relaxing coffee breaks. I was also very lucky to share my office with Xiaoqing and Marisela. Especially I am thankful to Marisela, who was not only my office-mate but also one of the best friends, for her unpredictability, which saved me from burnouts in these years and for continues supply of cookies in the office.

My life in Holland would not be so comfortable without all the Russian, Romanian, Cuban, etc. friends whom I knew before coming here and whom I met here. Especially I would like to mention my former course-mates from Novosibirsk State University Arthur Iordanidis and Alexander Netchaev.

At last but not least, I want to thank my parents, my sister, Adriana and other members of my family for their constant support, without which my stay abroad would have been much more difficult.

It is very difficult to write an acknowledgement section, since it is impossible to mention all the people you would like to thank. So, want to thank all other people whom I met during these years and who helped me to finish this thesis.

About the author

Andrei Sleptchenko was born on January 9, 1974 in Barabinsk, Russia. In 1991 he began Applied Mathematics at Novosibirsk State University. He graduated in 1995, with specialisation in Operations Research. Two years later he have got a master degree in Mathematics from the same university.

During the academic year 1997/1998 he attended the master class programme in Operations Research organised by the Mathematical Research Institute (MRI) in The Netherlands.

Since 1998 he was a Ph.D. student at the Technology & Management Faculty of the University of Twente. His research concerned mathematical models for repairable spare parts supply systems for with capacity restrictions at repair facilities and resulted in this thesis.