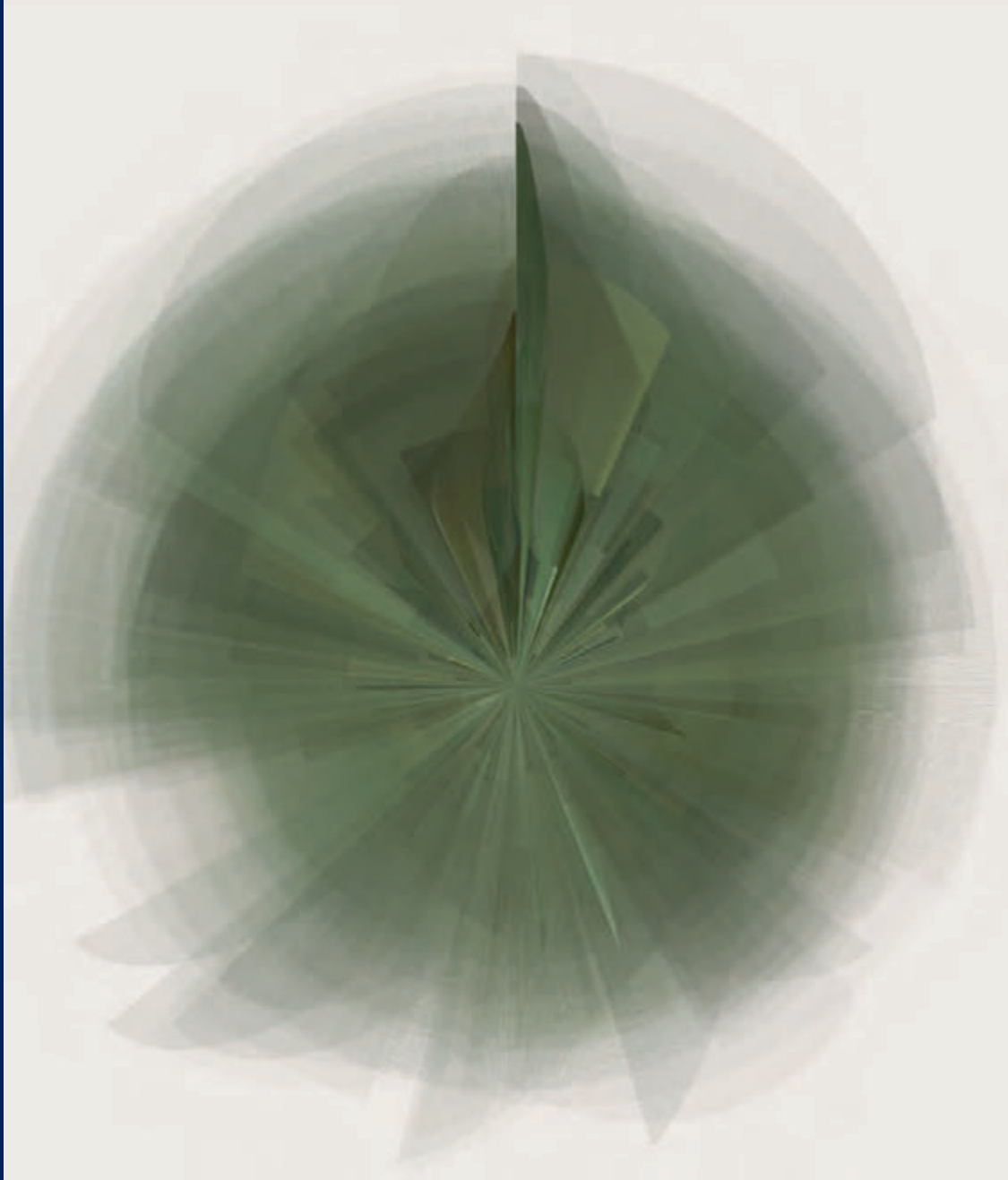
JELMER PIER VAN DER GAAST

Stochastic models for order picking systems



Stochastic Models for Order Picking Systems

Stochastic Models for Order Picking Systems

Stochastische modellen voor orderverzamelsystemen

Thesis

to obtain the degree of Doctor from

Erasmus University Rotterdam

by the command of

rector magnificus

Prof.dr. H.A.P. Pols

and in accordance with the decision of the Doctoral Board

The public defense shall be held on

Thursday 8 September 2016 at 13:30 hours

by

Jelmer Pier van der Gaast

born in Rotterdam, the Netherlands

Erasmus University Rotterdam

Doctoral Committee

Promoters:	Prof.dr.ir. M.B.M. de Koster
	Prof.dr.ir. I.J.B.F. Adan
Other members:	Prof.dr.ir. R. Dekker
	Prof.dr.-ing. K. Furmans
	Dr.ir. J.A.C. Resing

Erasmus Research Institute of Management – ERIM

The joint research institute of the Rotterdam School of Management (RSM) and the Erasmus School of Economics (ESE) at the Erasmus University Rotterdam

Internet: `http://www.erim.eur.nl`

ERIM Electronic Series Portal: `http://repub.eur.nl/pub/`

ERIM PhD Series in Research in Management, 398

ERIM reference number: EPS-2016-398-LIS

ISBN 978-90-589-2455-1

Design: PanArt en advies, `www.panart.nl`

This publication (cover and interior) is printed by Tuijtel on recycled paper, BalanceSilk®. The ink used is produced from renewable resources and alcohol free fountain solution. Certifications for the paper and the printing production process: Recycle, EU Ecolabel, FSC®, ISO14001. More info: `www.tuijtel.com`

Acknowledgments

While writing this dissertation I received tremendous support from a lot of different people without whom I simply could not have written this dissertation. First, I would like to thank my supervisors René de Koster and Ivo Adan for their enthusiasm and all the advice and feedback I received over the years.

Second, I would like to thank Rommert Dekker, Kai Furmans, Jacques Resing, for serving on the inner committee, as well as, the members of the plenary committee. Especially, I would like to thank Jacques for all the help and the many lunches we had together with Ivo.

Next, I would like to thank Albert Wagelmans, Adriana Gabor, Yingqian Zhang, Onno Boxma, Arie Quist, Bram Verweij, and David Huygens for supporting me during the beginning of my PhD trajectory. Also, Niels Rietveld for all the times when we were still students and the regular meetings we had afterwards.

I would also like to thank all my (former) colleagues from vakgroep 6, vakgroep 1, EURANDOM, and the Econometric Institute. Also thanks to my office mates and Carmen Mirasol-Meesters for the help with administrative work and all our non-work related talks.

I would like to thank my friends and family for all the support and especially both my sisters Eline and Myrthe for being my paranymphs. Finally, Jing, I am glad I met you and I hope we will be happy together for a very long time.

Jelmer Pier van der Gaast
Rotterdam, 2016

Contents

1 Introduction

Warehouses are a key factor in any supply chain. Their main function is to buffer against variability between supply and demand caused by factors such as seasonality in demand, production scheduling, transportation, consolidation of items, and value-added-processes (Gu et al., 2010). In order to stay competitive in a dynamic business environment full of uncertainties, warehouse operations nowadays need lower delivery costs, shorter customer response times, and higher customer service. Thus, a better understanding of the impact of uncertainties that occur in warehouse processes can contribute to the success of any supply chain.

(a) Automated picking **(b)** Batch picking **(c)** Zone picking

Figure 1.1: Various warehousing systems (photographs by author).

In Figure 1.1 various warehousing systems are shown. A warehouse is typically divided into functional areas to support daily operations. Figure 1.2 shows a schematic representation of the main functional areas; receiving, reserve and forward storage, and shipping. Activities in the *receiving* area include the unloading of products from transport carriers; quantity and quality control; repackaging (e.g. full pallets to cases, or standardized bins); and the transfer of products either to storage or directly to the shipping area via cross-docking. The *storage* area often consists of two parts: the reserve and forward storage area. In the reserve area products are stored in bulk (e.g. pallet racking system), whereas the forward area is used for picking

high-demand, fast moving products. In the order picking process, products are retrieved from their storage locations and this process can either be done manually or (semi-)automated. Depending on the demand and product characteristics, different order picking methods can be applied, e.g. pallet picking, case picking, or broken case picking. In case of a stock-out in the forward area, products are replenished from the bulk stock stored in the reserve area. After the picking process, the orders are sorted, accumulated, and packed. Finally, at the *shipping* area, the orders are consolidated and loaded onto transport carriers.

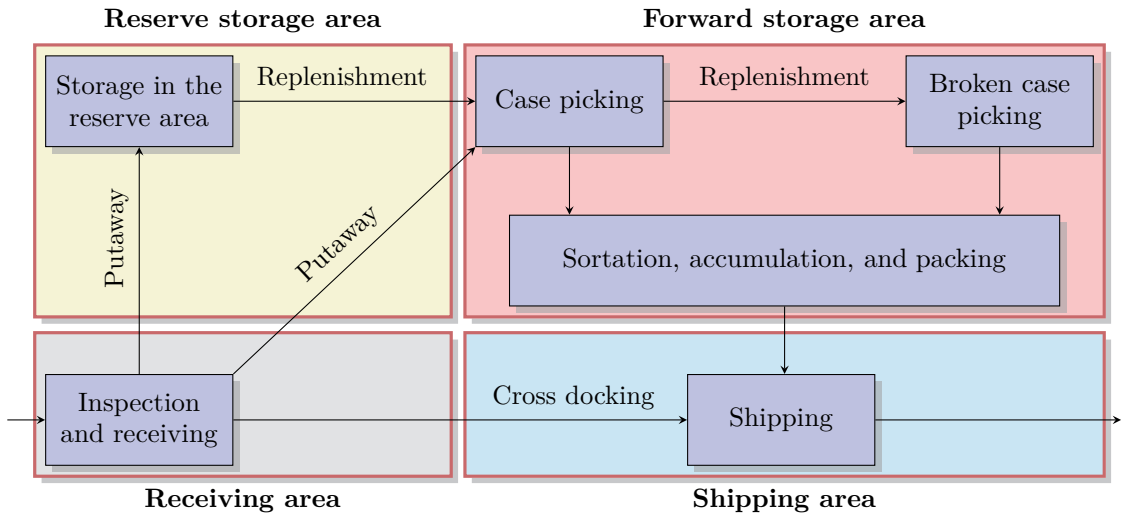


Figure 1.2: Typical warehouse functions and flows (Tompkins et al., 2003).

Out of all these activities, order picking is the most labor-intensive and costly activity in warehouses due to its high contribution (about 55%) to the total operating cost (Drury, 1988). In order for a warehouse to operate efficiently, the order picking process needs to be robustly designed and optimally controlled (De Koster et al., 2007). Any under performance in order picking can lead to unsatisfactory service and high operational cost for the warehouse, and consequently for the whole supply chain. Therefore, many warehouses invest heavily in state-of-the-art order picking solutions to increase productivity and reduce any uncertainty associated with order picking. However, this has increased the complexity of today's warehouse operations (Frazelle, 2001). As a consequence, conventional models studying the order picking process are no longer sufficient.

In this thesis, we develop new stochastic models for the performance evaluation of several highly state-of-the-art warehousing systems that, in particular, adequately describe and predict the consequences of variability in, e.g. order arrivals, and picking times on the performance of a warehousing system. Stochastic models provide an indispensable tool for this task and have already proved to be extremely valuable for areas such as manufacturing, communication, and computer systems. Also for warehousing systems, the stochastic models provide valuable guidance in the rapid comparison of key features of different design alternatives and allow operations to be optimized in order to meet prespecified performance targets.

This chapter is structured as follows. Section 1.2 will describe the various factors that contribute to uncertainty in the order picking process. Section 1.2 and Section 1.3 introduce the two order picking solutions, *zone picking* and *milkrun order picking*, studied in this thesis and discuss their popularity in practice, the causes and consequences of variability within these solutions, and finally address the research questions. Section 1.4 presents the contributions of the thesis and gives an overview of the following chapters.

1.1 Uncertainties in order picking

Due to the growing popularity and availability of new complex warehouse systems, there is a strong need to make these systems manageable for warehouse operators. In particular, a clear understanding of causes and consequences of the variability in the various warehouse processes is essential, since uncertainty propagates throughout the whole supply chain and leads to inefficient processing and non-value adding activities (Van der Vorst & Beulens, 2002).

Gong & De Koster (2011) classify three groups of types of uncertainty that occur in warehouse operations; *unpredictable rare events* (e.g. strikes, natural disasters, political changes), *predictable events* (e.g. demand seasonality), and *internal variability*. Much of the internal variability is caused by stochastic behavior in the order picking process, e.g. varied order batches, differences in order picking times, etc. Therefore, in order to improve productivity and reduce cost within a warehouse and the entire supply chain, studying the order picking process is essential.

Order picking is the process of retrieving customer orders from their storage locations. A customer order can consist of multiple order lines, that determine which products and in what quantity they need to be picked. Nowadays, a single warehouse often consists of multiple order picking systems to satisfy customer demand (De Koster et al., 2007). These systems mainly vary in the degree of automation employed, in the type of products that need to be handled by the system, and in the number of orders and order lines that need to be processed from the system. The most common solutions are *picker-to-parts* systems where an order picker has to travel/drive to the pick locations containing the stored products in, e.g. bin shelves or flow racks. Triggered by the need for faster customer response times and lower internal variability, automated systems have grown in popularity in recent years. In *parts-to-picker* systems the products to be picked are automatically brought to the order picker. Examples are carousels and miniload automated storage-and-retrieval systems. Recently more *automated* order picking systems have become available on the market, e.g. automated A-frame dispenser systems, and robot picking (Marchet et al., 2015).

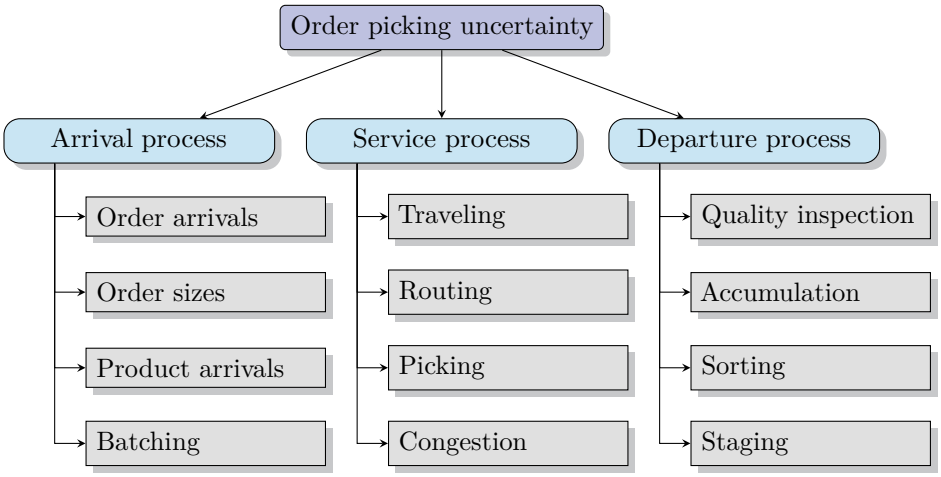


Figure 1.3: Examples of uncertainties that influence the order picking process.

Within all these systems variability plays an important role. From a stochastic viewpoint, uncertainty in the order picking process can occur in the arrival, service, and/or departure process. In terms of order picking, the arrival process is mainly associated with incoming products and customer orders, whereas the service process concerns the internal warehouse activities. Finally, the departure process consists of the outbound flow in the warehouse. In Figure 1.3 the same three groups are used

to classify sources of uncertainty that influence the order picking process. In the *arrival process*, incoming product arrivals can vary because of the inventory levels at suppliers and the mode of transportation used. On the other hand, customer orders arrive typically at irregular intervals with a varying demand for products and the arrival rate can change over time. In addition, orders might need to be batched until the next pick tour starts, e.g. due to relatively long walking distances. In the *service process* variability is caused by varying traveling distances, e.g. of the order picker walking/driving to the current pick locations, of an order tote traveling on a conveyor to its next station, or of a robot picking up a specific order bin and bringing it to a workstation. Also, the time to reach the destination strongly depends on the routing mechanism being used. The actual picking times at the pick locations depend on the time to find the right amount of products, extracting the products, putting them down, and verifying that the correct amount of products has been picked. Finally, especially when the order picking system is heavily utilized, congestion and even blocking can occur during the order picking process. Examples include order pickers that cannot pass each other because of narrow aisles, or pick zones become saturated. Lastly, the *departure process* depends on how long it takes until all the customer orders are accumulated and sorted for a shipment, making sure no mistakes were made during the order picking process, and waiting in the staging area for the shipment to depart.

In the academic literature different stochastic models have been proposed to study different kinds of order picking systems. An excellent overview is given by De Koster et al. (2007). The authors clearly state the importance of stochastic models, since deterministic models may lead to wrong conclusions if underlying processes are variable and can reduce the warehouse operator's competitiveness significantly. Still, the area of order picking is a rapidly growing industry and since the processes associated are inherently stochastic, stochastic modeling of these systems should be explored even further.

In this thesis we will focus on order picking systems that are mainly left unexplored, but that are highly relevant in practice. Two of such systems are *zone picking* and *milkrun picking*. In the next two sections, we show how uncertainty affects the performance of these systems and how it can be modeled in order to optimally design and control these systems.

1.2 Zone picking systems

Zone picking is one of the most popular picker-to-parts order picking methods used in practice. It is particularly popular in companies with a fairly large number of customer orders, picked from a large assortment of relatively small-sized products, and low to moderate number of picks per order. In such a system the order picking area is zoned, where in each zone an order picker is responsible for picking products from his or her dedicated part of the warehouse. Because of zoning, the pickers have to spend little traveling time between locations to pick the required products and have an increased familiarity with the products in the zone (Gu et al., 2010). Zones are usually connected with automated conveyors and products are typically stored in flow racks or shelves.

Zone picking systems can be categorized in systems with parallel and sequential zone picking. In a *parallel zone picking system*, a batch of customer orders, which each can consist of several order lines, is picked simultaneously in multiple zones and a downstream sorting process consolidates the picked order lines into the customer orders after the picking process has finished. In *sequential zone picking* (or pick-and-pass picking), shown in Figure 1.4, an order is assigned to an order tote or order carton that travels on the conveyor and sequentially enters the buffer of a zone where products are stored that should be added to the order. Order totes travel between the zones and visit only those zones where the required products are stored. At a zone, each picker picks for only one tote at a time. The advantage of sequential zone picking is that order integrity is maintained and no sorting and product consolidation is required. However, since an order has to visit the zones in a sequential order the order throughput time is usually higher than in parallel zone picking due to the transport times between zones and the individual set-up times at each zone.

In most practical environments zone picking systems exhibit highly variable behavior, due to differences of work profiles of the orders in the various zones. Some orders may require much work at one particular zone, whereas other orders may require more work at other zones. Even depending on the time or day, work profiles can vastly vary. In sequential zone picking this can lead to congestion and even blocking situations; order totes on the conveyor cannot be diverted to a zone, thereby (temporarily blocking) the conveyor, which can lead to reduced throughput and causes unpredictable throughput

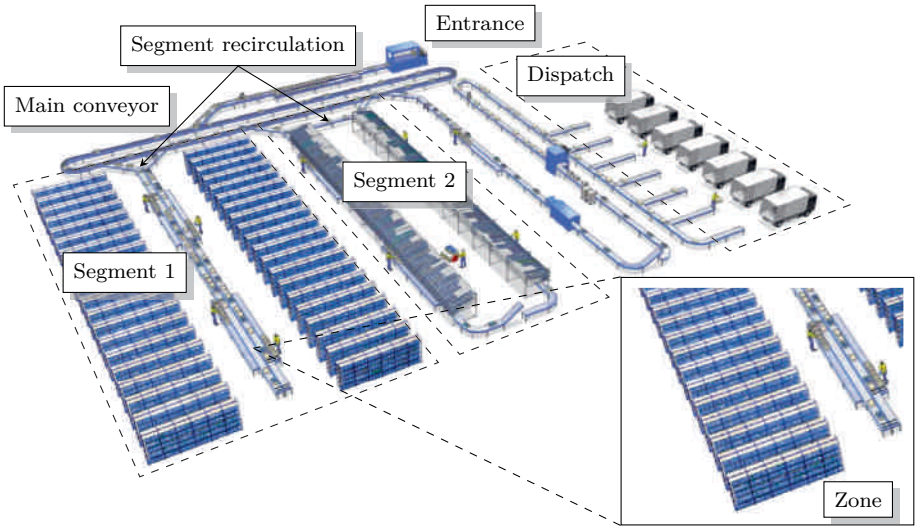


Figure 1.4: A zone picking system with multiple segments (Vanderlande, 2007).

times. Often, blocking on the conveyor is avoided by providing a recirculation option for blocked totes. The automated conveyor will transport totes to another zone where other products can be picked. Eventually the totes return to the blocked zone that might then be unblocked in the meantime. Blocking also can occur when two conveyor flows merge into one single flow, e.g. totes leaving a zone merge with the totes on the conveyor. The tote must wait for a sufficiently large space on the conveyor in order to prevent collisions. Under low utilization, the time required for a sufficiently large space on the conveyor to show up is negligible. However, many systems are highly utilized during peak hours. In such a case, this space can become very scarce leading to long merge times and a loss in overall system performance. In addition, the waiting totes can stop the order picker from continuing to work on the next tote in line.

Despite its popularity in practice, sequential zone picking has not received much attention in the literature. The previous works assume that no blocking occurs in the system, e.g. Yu & De Koster (2009) and Melacini et al. (2010). This is, however, not very realistic. In peak periods zones can become congested, leading to blocking of totes which may propagate over the entire network. Such blocking effects have serious impact on the performance of a zone picking system. Besides blocking there are many open questions from industry which are highly relevant for zone picking

systems (Apple, Jr. et al., 2010). These include questions like, should a tote pick (or not) and pass through all zones or bypass zones without activity, and should a zone provide recirculation or provide long accumulation buffers for totes?

In this thesis we will answer the following research questions related to zone picking systems;

- How to determine the performance measures of a given zone picking system, e.g. zone utilization, the average overall order throughput, system throughput time, or the probability of a tote blocked by a full zone?
- How much influence do congestion and blocking related to limited zone buffer sizes and conveyor merges have on the performance of a zone picking system?
- Given an order profile, what is the best product allocation policy for a zone picking system such that it maximizes the performance measures and what are the trade-offs?

In Chapters 2, 3 and 4 of this thesis we study zone picking systems, with the following features;

- Zone skipping of order totes; order totes that encounter a full zone, skip the zone and return later (Chapter 2 and Chapter 3).
- Single- or multi-segment routing (Chapter 2).
- Priority merging of multiple conveyor flows; order totes with the least priority should wait until they can merge on the conveyor (Chapter 3).
- The effect of different product allocation policy methods (Chapter 4).

The corresponding models will describe a zone picking system more accurately than any of the previous models currently available in literature. Using queueing networks and verified by discrete event simulation techniques we will develop an approximation model for analyzing and evaluating the performance of a zone picking system. The method can be used to rapidly analyze and design zone picking systems for a particular performance. In Chapter 4 the performance measures of the approximation model will be compared to a real-world system.

1.3 Milkrun picking systems

Recent technological advances and trends in distribution and manufacturing have led to a growth in complexity of warehousing systems. In order to stay competitive and flexible, warehouse managers have started to incorporate the concepts of *lean* in their warehouse to continuously improve their operations and eliminate as much *waste* as possible. One of the biggest sources of waste in warehouses is associated with material handling, e.g. a customer order waiting for the next pick cycle to start so that it can be picked, waiting of pallets for pickup and transport, and order pickers waiting for work due to blocking at upstream processes. A way of reducing non-value added processes is by standardizing material flows by incorporating a *milkrun*.

A milkrun refers to the scheduled pickup or supply of materials to a number of customers or suppliers by a single vehicle which visits them according to a fixed round trip (Baudin, 2004). The advantage is that the milkrun reduces transportation costs due to consolidated transportation, allows for better synchronization with the customers or suppliers, and improves general response times and system efficiency (Brar & Saini, 2011). The same milkrun concept can also be used for internal logistics, e.g. manufacturing or warehousing, to transport raw materials, work-in-process, and finished goods between different locations within the facility.

Most internal milkrun systems use tugger trains to transport materials between storage and production areas. The tugger train, pulling multiple trailers, follows a fixed delivery route where at different locations on the route materials are loaded or unloaded given the demand of the particular location. Compared to a system that uses forklifts to supply materials in assembly or production lines, tugger trains are more efficient, safer because of less traffic, and significantly reduce the number of empty runs. Furthermore, a milkrun system can lead to considerable savings in labor costs and operating costs, which have led that milkrun systems are also being used for order picking (Gong & De Koster, 2008).

In a *milkrun order picking system*, an order picker picks orders that arrive in real time during the picking process, by dynamically changing the stops on the picker's current picking route. The picker is constantly moving through the warehouse and receives, using modern order-picking aids like pick-by-voice techniques, or a handheld terminal, new pick instructions that allow new orders or order lines to be included

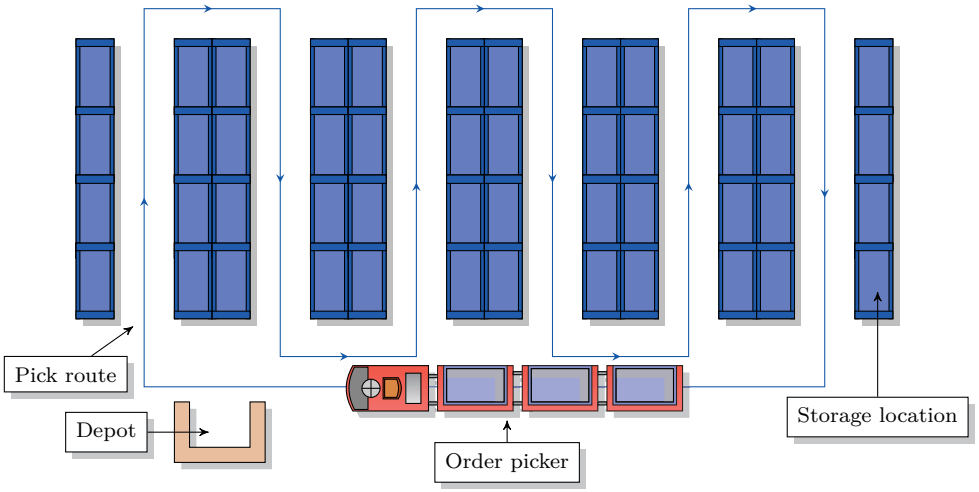


Figure 1.5: A milkrun picking system.

in the current pick route. After a pick cycle has been completed, the order picker disposes all the picked order lines centrally at the depot where the orders can be sorted and packed and immediately starts a new pick cycle. The advantage of milkrun picking is that it reduces order picking set-up time and worker travel time compared to conventional batch picking systems. However, despite the fixed pick route, a milkrun picking system still exhibits variability, not only in order arrivals, but also in the pick process.

In the literature milkrun picking systems are scarcely researched. Gong & De Koster (2008) studied a milkrun picking system (referred as a dynamic order picking system in the paper) using polling models and showed that the use of a milkrun picking system has a considerable advantage over conventional batch picking. Boon et al. (2010) considered an efficient enhancement to an ordinary milkrun picking system that allows products stored at multiple locations. The location of the picker would then determine where specific order lines need to be picked. However, both papers only considered waiting times of order lines, which is the time between the arrival of a customer order and the start of a pick of a product unit within in the picking area. A key for the quality of a milkrun picking system is that the system achieves low (average) *order throughput times*, i.e. the time between a customer order, consisting often of multiple order lines, entering the system and when the whole order is delivered at the depot. Short order throughput times in milkrun systems can allow for faster

customer response and improved customer satisfaction. Currently, the analysis of order throughput times in milkrun picking systems is unexplored in the literature.

The modeling and analysis of order throughput times in a milkrun picking system considerably benefits from the available theory on polling systems. There is considerable literature on polling models (see Vishnevskii & Semenova (2006) and Boon et al. (2011)), which are used to model an abundant set of systems in a wide field of different applications. On the other hand, studying order throughput times (or batch sojourn-times in a general context) can also enrich the literature on polling systems by providing new insights such as under which conditions the throughput times are minimized.

Finally, the order throughput time strongly depends on the product allocation, since a customer order often contains several order lines, each for a different product that can only be stored at certain locations within the order picking area. A product (or storage) assignment method is a set of rules used to assign products to storage locations. In order to achieve short order throughput times, it is essential to take the correlation between order lines into consideration and place order lines that are strongly correlated in an optimal way in order to reduce the probability that the order will be completed in the next pick tour. Assigning the products to the optimal location would lead to shorter order throughput times, and consequently to faster customer response.

In this thesis we will answer the following research questions related to milkrun picking systems;

- How to calculate order throughput times in a milkrun system?
- How much does the picking policy (e.g. pick all outstanding order lines at a location, or pick all outstanding order lines and just arrived incoming order lines at a location before moving to the next location) influence the order throughput times?
- Given an order profile, what is the best storage assignment method that achieves the shortest order throughput times?

In Chapters 5 and 6 of this thesis we study milkrun systems, with the following features;

- The general framework used to analyze the order throughput times in a milkrun system (Chapter 5).
- The effect of different service disciplines on order throughput times (Chapter 5 and Chapter 6).
- Optimal product allocation in a milkrun picking system (Chapter 6).

These chapters create the opportunity to model and analyze a milkrun picking system accurately and the corresponding model can be used to rapidly analyze and design a milkrun picking system and determine optimal product allocations.

1.4 Contribution and thesis outline

The systems that we analyze in this thesis using stochastic models are commonly used in warehouse practice. First, often the real-world cases that have been implemented have not been optimized; i.e. implementations primarily take care of feasibility rather than optimal performance. Second, most of the existing models used in practice to analyze these systems rarely take into account the inherent variable behavior that typically occurs. The models developed in the thesis will help both designers and managers to create optimal design and control methods to improve the performance of such systems. They can also help designers to avoid major mistakes, because they expand the designer's intuitive understanding of what determines system performance. These models will help to improve the performance of these systems and, on other hand, enrich the current literature on warehousing systems.

We summarize the following chapters of this thesis as follows:

Chapter 2: Modeling and performance analysis of sequential zone picking systems

This chapter develops an analytical model of sequential zone picking systems. The systems belong to the most popular internal transport and order picking systems in practice, due to their scalability, flexibility, high-throughput ability, and fit-for-use for a wide range of products and order profiles. The major disadvantage of such systems, though, is congestion and blocking under heavy use, leading to long order lead times. In order to diminish blocking and congestion most systems make use

of a dynamic block-and-recirculate protocol. The various elements of the system, like conveyor lanes and the pick zones, are modeled as a network of queues with multiple order classes and with capacity constraints on subnetworks, including the dynamic block-and-recirculate protocol. Due to this protocol, however, the stationary distribution of the queueing network is highly intractable. Therefore, an innovative approximation method, using jump-over blocking is proposed to accurately assess key performance statistics such as throughput and recirculation. Multi-class jump-over networks admit a product-form stationary distribution, and can be efficiently evaluated by Mean Value Analysis (MVA) and use of Norton's theorem. The method is most suitable to support rapid and optimal design of complex zone picking systems, in terms of number of segments, number and length of zones, buffer capacities, and storage allocation of products to zones, in order to meet prespecified performance targets. Comparison of the approximation results to simulation show that for a wide range of parameters the mean relative error in the system throughput is typically less than 1%.

Chapter 3: An accurate model for conveyor merges in zone picking systems

Sequential zone picking systems are popular conveyor-based picker-to-parts order picking systems that divide the order picking area in work zones. When designing a zone picking system, it is important to know whether the throughput capability of the system is able to meet customer demand. However, the performance and maximum throughput capability of a zone picking system is largely determined by congestion and blocking that occurs at the various conveyor merges in the system. In this chapter we develop an analytical model to study the impact of conveyor merges in sequential zone picking systems. Due to finite buffers, blocking, recirculation, and merging, the resulting queueing model does not have a product-form stationary queue-length distribution which makes exact analysis practically unfeasible. Therefore, we develop an approximate solution by using an aggregation technique and matrix-geometric methods to study the throughput capability of the system. The model is suitable to support rapid and optimal design of complex zone picking systems, in terms of number and length of zones, input and output buffer capacities, and storage allocation of products to zones, in order to meet prespecified performance targets. Comparison of the approximation results to simulation show that for a wide range of parameters

the mean relative error in the system throughput is typically less than a few percent. The model accurately predicts the loss in throughput due to congestion and blocking at the merges, and can be used to allocate input and output buffer spaces in order to maximize the throughput capability of the system.

Chapter 4: Case study: product allocation methods in zone picking systems

When designing a new zone picking system many decisions have to be taken on different strategic, tactical, and operational levels. However due to the complex nature of a zone picking system, these decisions are not always in line with each other and affect the performance of the system in different ways. In addition, when in practice a new zone picking system is designed and implemented its feasibility, i.e. within budget and structure wise, is often of main concern rather than optimal performance and the inherent variability of the system. In this chapter we investigate the performance of a current zone picking system of a large wholesaler supplying non-food items to supermarkets with the analytical methods we developed in the previous chapters. We test a product allocation method that minimizes the number of segments a tote on average has to visit and a method that applies workload balancing between segments in order to reduce congestion and potential blocking in the system. In addition, we combine both methods into a single method that tries to minimize simultaneously the average number of segments a tote visits and applies workload balancing between segments. In particular, a product allocation that only applies workload balancing between segments reduces the system throughput on average by 8% compared to a product allocation that minimizes the number of segments a tote on average has to visit. On the other hand, blocking of zones and segments is significantly reduced by a product allocation that applies workload balancing, e.g. segments are blocked 7.6% on average when a product allocation that minimizes the number of segments is used to 0.3% on average in the other case. As such, it provides a valuable tool for initial product allocation decisions for zone picking systems and the consequences strategic decisions can have on the overall performance of the system.

Chapter 5: The analysis of batch sojourn-times in polling systems

We consider a cyclic polling system with general service times, general switch-over times, and simultaneous batch arrivals. This means that at an arrival epoch, a batch of customers may arrive simultaneously at the different queues of the system. For the locally-gated, globally-gated, and exhaustive service disciplines, we study the batch sojourn-time, which is defined as the time from an arrival epoch until service completion of the last customer in the batch. We obtain for the different service disciplines exact expressions for the Laplace-Stieltjes transform of the steady-state batch sojourn-time distribution, which can be used to determine the moments of the batch sojourn-time, and in particular, its mean. However, we also provide an alternative, more efficient way to determine the mean batch sojourn-time, using Mean Value Analysis. Finally, we compare the batch sojourn-times for the different service disciplines in several numerical examples. Our results show that the best performing service discipline, in terms of minimizing the batch sojourn-time, depends on the system load and the ratio between service/switch-over times.

Chapter 6: Optimizing product allocation in a milkrun picking system

E-commerce fulfillment competition evolves around cheap, speedy, and time-definite delivery. Milkrun order picking systems have proven to be very successful in providing handling speed for a large, but highly variable, number of orders. In this system, an order picker picks orders that arrive in real time during the picking process; by dynamically changing the stops on the picker's current picking route. The advantage of milkrun picking is that it reduces order picking set-up time and worker travel time compared to conventional batch picking systems. This paper is the first in studying order throughput times of multi-line orders in a milkrun picking system. We model this system as a cyclic polling system with general service times, general switch-over times, and simultaneous batch arrivals. We determine the mean order throughput time for three picking strategies; exhaustive, locally-gated, and globally-gated. These results allow us to study the effect of different product allocations in an optimization framework. We show in several numerical examples that, depending on the system parameters, the picking strategy that achieves the shortest order throughput times varies. In addition, for a real world application we show that milkrun order picking reduces the order throughput time significantly compared to conventional batch picking.

Research statement

This PhD dissertation has been written during the author's work at the Erasmus University Rotterdam. The author is solely responsible for formulating the research questions, building the analytical models, analyzing the results, and writing all the chapters of this thesis. While carrying out the research, the author received feedback from the doctoral advisors and other members of the doctoral committee which subsequently increased the quality of research.

The data in Chapter 4 was provided by a large Dutch wholesaler supplying non-food items to supermarkets.

Chapters 2, 3, 5 and 6 have been submitted to scientific journals and are currently at various stages of the review process.

2 Modeling and performance analysis of sequential zone picking systems

2.1 Introduction

Order picking, the process of picking products to fill customer orders, is the most labor-intensive and costly activity in warehouses due to its high contribution (about 55%) to the total operating cost (Drury, 1988). Recent trends in distribution and manufacturing, like e-commerce, have increased the importance of efficient order picking even more (Le-Duc & De Koster, 2007). The focus of this chapter is on the modeling and (approximate) analysis of sequential zone picking systems, with single or multi-segment routing.

Zone picking is one of the most popular picker-to-parts order picking method, where the order picking area is zoned. In each zone, an order picker is responsible for picking from his or her dedicated part of the warehouse (Petersen, 2002; Gu et al., 2010). In practice, the zones are often connected by conveyors to reduce travel times. Major advantages of zone picking systems are high-throughput ability, scalability and flexibility in handling both small and large order volumes, and fit-for-use for different product sizes, with a different number of order pickers. These systems are often applied in warehouses handling customer orders with a large number of order lines and with a large number of different products kept in stock (Park, 2012). A disadvantage of such systems, however, is congestion and blocking under heavy use, leading to long order throughput times.

Zone picking systems can be categorized in systems with parallel or sequential zone picking (De Koster et al., 2007). In a parallel zone picking system, a customer order, which can consist of several order lines, is picked simultaneously in multiple

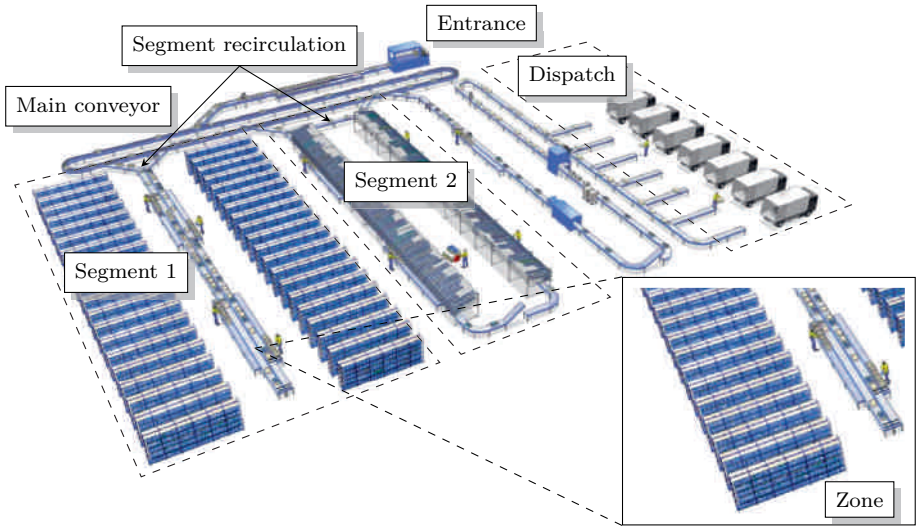


Figure 2.1: A sequential multi-segment zone picking system (Vanderlande, 2007).

zones and a downstream sorting process consolidates the picked order lines into the customer orders after the picking process has finished. In sequential zone picking (or pick-and-pass picking), shown in Figure 2.1, an order is assigned to an order tote or order carton that travels on the conveyor. Upon arrival at a zone, the tote enters the buffer if the zone stores products that should be added to the order, or it is passed to the next zone. At a zone, each picker picks for only one tote at a time. The advantage of sequential zone picking is that order integrity is maintained and no sorting and product consolidation is required (Petersen, 2000).

There are two types of sequential zone picking systems that can be distinguished: *single-segment routing* and *multi-segment routing*. In single-segment routing, the conveyor forms one circular loop that connects all the zones, whereas in multi-segment routing, zones are grouped in segments and per segment the zones are connected to a conveyor with a recirculation loop, like in Figure 2.1. The different segments are then connected by a central (or main) conveyor that diverts totes to the required segments. Multi-segment routing improves system throughput significantly due to shorter conveyor loops that avoid unnecessary long order tote travel times. However, the investment costs and space requirements are higher compared to single-segment routing.

De Koster (1994), Yu & De Koster (2008, 2009), and Melacini et al. (2010) model a zone picking system as a network of queues. In order to estimate performance statistics, such as the utilization, throughput rate of a zone, and, the mean and standard deviation of the throughput time of the totes, they use Whitt's queueing network analyzer (Whitt, 1982). A crucial aspect, however, that is not taken into consideration is blocking. In most environments, the workloads of the zones exhibit variability, due to differences in work profiles of the orders. In peak periods, zones can become congested, leading to blocking of order totes which may propagate through the entire network, such that zones become starved and order throughput times significantly increase. Such blocking effects may have considerable impact on the performance of a zone picking system and should not be ignored. Identification and quantification of the effects of blocking is challenging and crucial in the design of a zone picking system.

In a zone picking system blocking and congestion occurs at zones as well as at segments. Zones can become congested due to finite buffer space, since only a limited number of totes can be stored before they are processed by the order picker. Also segments can become congested if too many totes visit the segment at once. To resolve this, workload control is applied by which the system will prevent incoming totes from entering the segment until sufficiently many totes have left. In both cases, zone picking systems use a dynamic *block-and-recirculate* protocol: a blocked tote recirculates on the conveyor loop when the destination buffer is full or a segment is congested in order to avoid temporarily blocking all upstream totes. The tote potentially visits other zones or segments before attempting to enter the place where it was blocked previously. Queueing networks that attempt to model systems with the block-and-recirculate protocols are highly intractable: no exact results for the stationary distribution exist. In the literature, different blocking protocols have been investigated for various types of applications (see Schmidt & Jackman (2000), Hsieh & Bozer (2005), and Osorio & Bierlaire (2009) for recent references in manufacturing systems with automated conveyors). For an extensive review on blocking in queueing networks, the reader is referred to the books of Perros (1994), Balsamo et al. (2001), and Papadopoulos et al. (1993). A variation of the block-and-recirculate protocol for flexible manufacturing systems (FMS) was first studied by Yao & Buzacott (1987). According to their definition, a blocked part returns to the end of the queue

where it came from such that it can try to enter for a second time. The authors derive product-form solutions for FMS networks with finite buffers and recirculation, including networks with a central server (e.g. the material handling system) and networks with zero-buffer stations. These networks, however, are not suitable to model zone picking systems, since a blocked tote does not try to enter the buffer of the zone immediately again, but is transported to the next zone in line where it tries to enter this zone's buffer if still products need to be picked from this zone. By recirculation, the tote tries to enter the blocked station again. This process repeats until the order is fully picked.

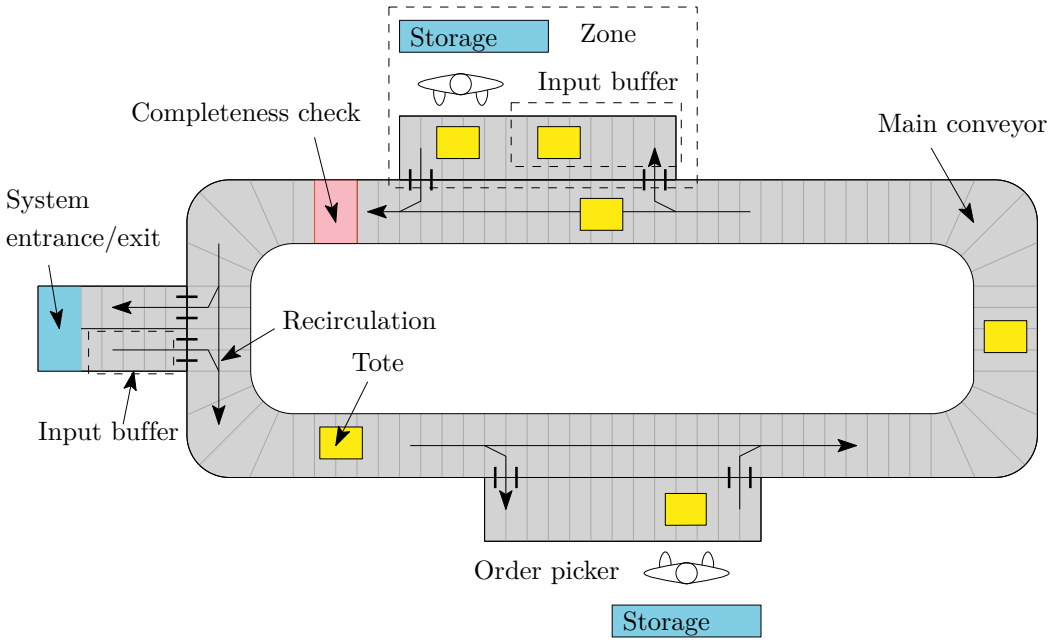
The objective of this chapter is to develop an analytical model for sequential zone picking systems (hereafter zone picking) with either single-segment routing, or multi-segment routing, and with finite buffers and segment capacities. This model is used to study the effects of design choice, loading, and storage on blocking and congestion of this commonly used order picking method. It considerably extends the models of De Koster (1994); Yu & De Koster (2008, 2009) and Melacini et al. (2010) that only consider zone picking systems with single-segment routing and no blocking. We develop a queueing model that incorporates the dynamic block-and-recirculate protocol and use the model to estimate the key performance statistics. Because an exact analysis of the queueing model with blocking is not feasible, we iteratively estimate the blocking probabilities from a multi-class queueing network, with *jump-over* blocking (Van Dijk, 1988; Economou & Fakinos, 1998). We show that this Markovian blocking protocol admits a product-form stationary queue-length distribution for a network with both jump-over nodes and with jump-over sub-networks. Key to the approximation is to equip the jump-over queueing network with Markovian routing that correctly reflects the relation to the block-and-recirculate queueing network, i.e., the flows in both networks should match. It appears that the jump-over queueing network provides very accurate estimates of the key performance statistics and allows us to study the sources of blocking and congestion in complex systems containing many zones and many different order classes. As such, it provides a powerful tool for system design, e.g., to determine the required number of segments, number and size of the zones, buffer capacities, or storage allocation of products to zones, in order to meet target performance levels.

The organization of this chapter is as follows. Section 2.2 presents the model for single-segment routing zone picking systems and discusses the corresponding approximation and analysis. The model is generalized to multi-segment routing zone picking systems in Section 2.3. We extensively analyze the results of both models in Section 2.4 via computational experiments for a range of parameters and validate them for a real-life system. In the final section we conclude and suggest some extensions of the model and further research topics.

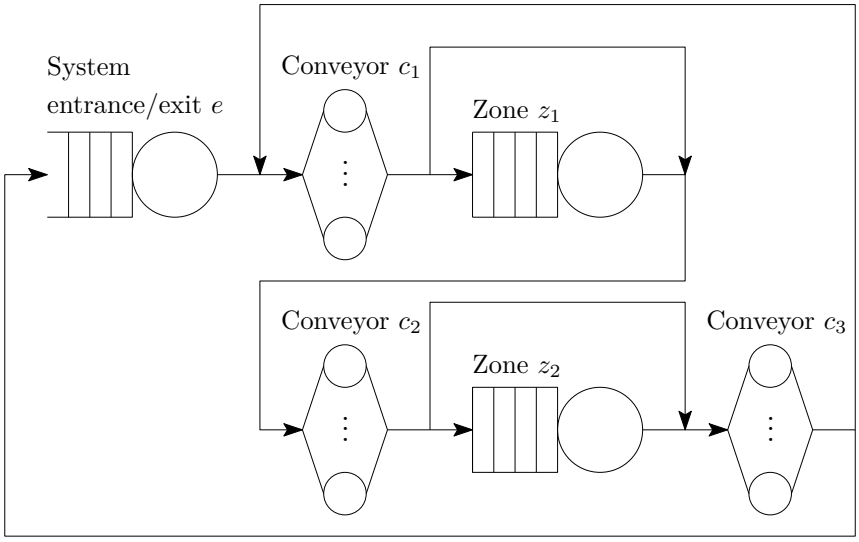
2.2 Single-segment zone picking systems

In a zone picking system with single-segment routing, three different elements can be distinguished: the entrance/exit of the system, the conveyors, and the zones. These elements of a single-segment zone picking system with two zones are shown in Figure 2.2a. For now, the assumption is made that totes enter and leave the system at the same location.

At the entrance of the system, a customer order is assigned to a new order tote. The tote is released into the system as soon as it is allowed by the workload control mechanism (Park, 2012). This mechanism sets an upper bound on the number of totes in the system and only releases a new order (with a tote) when a tote with all required order lines leaves the system. This workload control mechanism prevents the conveyor to become the bottleneck of the system. After release of an order it is merged with a tote at the entrance station and it receives all necessary paperwork (e.g. a packing list). The tote then moves to the buffer of a requested zone and enters if the buffer of that zone is not full. A blocked tote will stay on the conveyor and visits potentially other zones before returning. When the picking process has finished in a zone, the picker pushes the tote back on the conveyor. The waiting time for a sufficiently large space on the conveyor is considered to be negligible due to the workload control mechanism. This assumption is relaxed in Chapter 3. The conveyor then transports the tote to the next zone to be visited. A completeness check at the end of the conveyor loop ensures that the tote contains all the required order lines; otherwise, the tote is sent back to the beginning of the loop such that the tote will return to the zones it was blocked previously. When the tote has visited all the required zones, it leaves the system at the exit and a new order tote is immediately



(a) A zone picking system with single-segment routing and two zones



(b) The corresponding queueing network with system entrance/exit station e, $\mathcal{C} = \{c_1, c_2, c_3\}$ and, $\mathcal{Z} = \{z_1, z_2\}$

Figure 2.2: A zone picking system with single-segment routing and its corresponding queueing network.

released into the system. This is a valid assumption for zone picking system design, which aims at studying the throughput capacity of the system. Operational issues, like the effect of varying customer order arrival rates and customer order waiting times are not in the scope of this chapter, although they could be studied using a similar approach.

To model this system, a queueing network is proposed, the topology of which is shown in Figure 2.2b in case of two zones. The zone picking system is modeled as a closed queueing network with one entrance/exit, M zones and $M+1$ nodes that describe the conveyor between either two adjacent zones or between the entrance/exit and the first or last zone. The nodes are labeled in the following manner: the system entrance/exit is denoted as e, $\mathcal{Z} = \{z_1, \ldots, z_M\}$ denotes the set of zones, and $\mathcal{C} = \{c_1, \ldots, c_{M+1}\}$ is the set of conveyors in the network. Finally, let $\mathcal{S} = \{e\} \cup \mathcal{C} \cup \mathcal{Z}$ be the union of all the nodes in the network. The following assumptions are adopted for the network:

- There is an infinite supply of totes at the entrance of the system. This means that a leaving tote can always be replaced immediately by a new tote. Each tote has a class $\boldsymbol{r} \subseteq \mathcal{Z}$, e.g., $\boldsymbol{r} = \{z_2, z_3\}$ means that the tote has to visit the second and third zone.
- The total number of totes or totes to be released in the system is constant N. As long as the total number of totes in the zones and conveyor nodes is less than N, new totes are released one-by-one at an exponential rate μ_e at the system entrance, which is the rate at which a tote is prepared to enter the system (unfolding, adding labels and packing list, etc.).
- The conveyor nodes are assumed to be delay nodes with a deterministic delay time $1/\mu_i$, $i \in \mathcal{C}$.
- Each zone has $d_i\,(\geq 1)$, $i \in \mathcal{Z}$ servers, which represent the order pickers in the zone. The order picking time is assumed to be exponentially distributed with rate μ_i, $i \in \mathcal{Z}$, that captures both variations in the pick time per tote and variations in the number of order lines to be picked; this assumption is relaxed in Remark 2.3.
- When the order pickers are busy, incoming totes are stored in a finite input buffer of size q_i, $i \in \mathcal{Z}$. Incoming totes are blocked when the total number of totes in the buffer equals q_i.

The state of the queueing network is defined as $x = (x_i : i \in \mathcal{S})$, where $x_i = (\boldsymbol{r}_{i1}, \ldots, \boldsymbol{r}_{il}, \ldots, \boldsymbol{r}_{in_i})$ represents the state of node i with $\boldsymbol{r}_{i1}$ as the class of the first tote in the node, and $\boldsymbol{r}_{in_i}$ as the class of the last tote in the node. Let $\mathbb{S}(N)$ be the state space of the network, i.e., the sets of states x for which the number of totes in the system is equal to $\sum_{i \in \mathcal{S}} n_i = N$ and the number of totes in each zone satisfies $n_i \leq d_i + q_i$, $i \in \mathcal{Z}$.

The routing of totes through the network proceeds as follows. A new tote of class $\boldsymbol{r} \subseteq \mathcal{Z}$ is released at the system entrance with probability $\psi_{\boldsymbol{r}}$. These release probabilities correspond to a known order profile that can be obtained using e.g., historical order data or forecasts. After release, a tote of class $\boldsymbol{r}$ moves from the system entrance to the first conveyor node c_1. In general, after conveyor node c_i, the tote will either enter the input buffer of zone i if $z_i \in \boldsymbol{r}$ and the buffer is not full, or move to the next conveyor node c_{i+1}. In case the tote needs to enter and the buffer is full, the tote skips the zone and also moves to the next conveyor c_{i+1}, while it keeps the same class. If the buffer is not full, the tote enters the buffer of the zone and, after possibly waiting some time in the buffer, the order picker picks the required order lines. After all picks are completed, the tote will enter the next conveyor node in line and changes its class to $\boldsymbol{s} = \boldsymbol{r} \backslash \{z_i\}$. After visiting the last conveyor node c_{M+1}, all the totes with $\boldsymbol{r} \neq \emptyset$ are routed to the first conveyor node c_1; the other totes move to the exit and are immediately replaced by a new tote which is waiting for release at the entrance.

Summarizing, the state dependent routing probability $p_{i\boldsymbol{r},j\boldsymbol{s}}(x)$ that a tote of class $\boldsymbol{r}$ is routed from node i to node j which it enters as a class $\boldsymbol{s}$ tote, given that the network is in state x, can be specified as follows:

$$p_{e\emptyset,c_1\boldsymbol{r}}(x) = \psi_{\boldsymbol{r}}, \tag{2.1}$$

$$p_{c_i\boldsymbol{r},z_i\boldsymbol{r}}(x) = 1, \quad i = 1, \ldots, M,\ z_i \in \boldsymbol{r} \text{ and } n_{z_i} < d_{z_i} + q_{z_i}, \tag{2.2}$$

$$p_{c_i\boldsymbol{r},c_{i+1}\boldsymbol{r}}(x) = 1, \quad i = 1, \ldots, M,\ z_i \notin \boldsymbol{r} \text{ or } n_{z_i} = d_{z_i} + q_{z_i}, \tag{2.3}$$

$$p_{z_i\boldsymbol{r},c_{i+1}\boldsymbol{s}}(x) = 1, \quad i = 1, \ldots, M,\ \boldsymbol{s} = \boldsymbol{r} \backslash \{z_i\}, \tag{2.4}$$

$$p_{c_{M+1}\emptyset,e\emptyset}(x) = 1, \tag{2.5}$$

$$p_{c_{M+1}\boldsymbol{r},c_1\boldsymbol{r}}(x) = 1, \quad \boldsymbol{r} \neq \emptyset, \tag{2.6}$$

where the other routing probabilities are equal to 0.

The stationary distribution of this queueing network is intractable due to the finite buffers (Stidham, Jr., 2002) and the block-and-recirculate mechanism. This justifies the attempt to develop an approximate analysis of this queueing network.

In order to accurately estimate the performance statistics of the queueing network of Section 2.2, it is approximated in Section 2.2.1 by a network with the jump-over blocking protocol. This jump-over network exhibits a product-form steady state distribution, as will be shown in Section 2.2.2. Another property of the network, shown in Section 2.2.3, is that closed-form formulas of the visit ratios exist. The performance statistics of the jump-over network can be easily calculated exactly using e.g., mean value analysis (MVA), as explained in Section 2.2.4. Section 2.2.5 shows how the jump-over network is used to approximate the original network. Finally, the quality of the single-segment approximation is presented in Section 2.2.6.

2.2.1 Jump-over network

Assume a tote that intends to visit zone z_i is "tagged" after z_i with either the label *visited* z_i or *skipped* z_i. In the real system, and hence in the queueing network of Section 2.2, a tote is tagged as *visited* z_i if the tote entered z_i and received service. On the other hand, a tote is tagged as *skipped* z_i if the tote skipped the zone because the buffer was full.

The idea of the jump-over network is to introduce a Bernoulli process that tags each tote that intends to visit z_i randomly, and independent of whether the tote actually visited z_i or not. The probability of tagging with either one of the two labels is taken as the fraction of totes receiving a specific tag in the original queueing network, such that the fraction of totes that are tagged with *skipped* z_i equals the blocking probability b_i, $i \in \mathcal{Z}$ of the original network that a tote that tries to enter zone i but encounters it to be full ($n_i = d_i + q_i$).

Naturally, blocking probabilities are not known in advance, but will be estimated iteratively after an initial guess from the approximation, as shown in Section 2.2.5. Hereafter, b_i is assumed to be known beforehand in the jump-over network.

The routing probabilities in the jump-over network are *state independent*. In particular, the routing probabilities (2.1)-(2.6) in the block-and-recirculate network are

replaced by state independent probabilities in the jump-over network. That is, after c_i each tote with $z_i \in \boldsymbol{r}$ is routed to z_i regardless whether the buffer of the zone is full. The tote will enter the buffer if it is not full, otherwise the tote instantaneously skips the zone. Then for each class $\boldsymbol{r}$ tote, independent of whether the tote visited or skipped z_i (because of a full buffer), $p_{z_i\boldsymbol{r},c_{i+1}\boldsymbol{r}} = b_{z_i}$, $i = 1, \ldots, M$. This means that a tote of class $\boldsymbol{r}$ is tagged as *skipped* z_i and routed to the next conveyor node c_{i+1} with the same class with probability b_{z_i}, and otherwise, with probability $1 - b_{z_i}$, the tote is tagged as *visited* z_i and the class of the tote changes to $\boldsymbol{s} = \boldsymbol{r} \backslash \{z_i\}$. Summarizing, the routing probabilities (2.2)-(2.4) are replaced by

$$p_{c_i\boldsymbol{r},z_i\boldsymbol{r}} = 1, \qquad i = 1, \ldots, M,\ z_i \in \boldsymbol{r}, \tag{2.7}$$

$$p_{c_i\boldsymbol{r},c_{i+1}\boldsymbol{r}} = 1, \qquad i = 1, \ldots, M,\ z_i \notin \boldsymbol{r}, \tag{2.8}$$

$$p_{z_i\boldsymbol{r},c_{i+1}\boldsymbol{r}} = b_{z_i}, \qquad i = 1, \ldots, M, \tag{2.9}$$

$$p_{z_i\boldsymbol{r},c_{i+1}\boldsymbol{s}} = 1 - b_{z_i}, \quad i = 1, \ldots, M,\ \boldsymbol{s} = \boldsymbol{r} \backslash \{z_i\}\,. \tag{2.10}$$

Since the tagging process is made independent of the state of the buffer, the block-and-recirculate protocol is replaced by the *jump-over* blocking protocol (Van Dijk, 1988). Under this protocol, each tote of class $\boldsymbol{r}$ leaving z_i, either after service or skipping, continues to follow the same Markovian routing. The advantage of the jump-over blocking protocol, also known as "overtake full stations, skipping, and blocking and rerouting", is that closed-form analytic results for single-class queueing networks are available in the literature (Pittel, 1979; Schassberger, 1984; Van Dijk, 1988; Economou & Fakinos, 1998).

2.2.2 Product-form of the stationary distribution

A powerful tool to prove the existence of product-form solutions in the multi-class jump-over network is the concept of *quasi-reversibility* (Kelly, 1979); when a queueing network with Markovian routing can be decomposed in terms of nodes that are quasi-reversible, the stationary distribution of the network can be written as the product of the stationary distributions of these individual nodes.

Let $\lambda_{i\boldsymbol{r}}$ be the visit ratio of a class $\boldsymbol{r}$ tote to node i satisfying the traffic equations

$$\lambda_{i\boldsymbol{r}} = \sum_{j\in\mathcal{S}} \sum_{\boldsymbol{s}\subseteq\mathcal{Z}} \lambda_{j\boldsymbol{s}} p_{j\boldsymbol{s},i\boldsymbol{r}}, \quad i \in \mathcal{S}, \boldsymbol{r} \subseteq \mathcal{Z}. \tag{2.11}$$

The equations (2.11) determine the visit ratios $\lambda_{i\boldsymbol{r}}$ up to a multiplicative constant.

Theorem 2.1. *The jump-over network with state space $\mathbb{S}(N)$ has a product-form stationary distribution of the form*

$$\pi(x) = \frac{1}{G} \prod_{i\in\mathcal{S}} \pi_i(x_i), \tag{2.12}$$

where G is a normalization constant and $\pi_i(x_i)$ are the marginal probabilities defined as

$$\pi_i(x_i) = \begin{cases} \prod_{l=1}^{n_i} \left(\frac{\lambda_{i\boldsymbol{r}_{il}}}{\mu_i}\right), & i = e, \\ \prod_{l=1}^{n_i} \left(\frac{\lambda_{i\boldsymbol{r}_{il}}}{\mu_i}\right) \frac{1}{n_i!}, & i \in \mathcal{C}, \\ \prod_{l=1}^{n_i} \left(\frac{\lambda_{i\boldsymbol{r}_{il}}}{\mu_i}\right) \frac{1}{\gamma(n_i)}, & i \in \mathcal{Z}, \end{cases} \tag{2.13}$$

with $\lambda_{i\boldsymbol{r}_{il}}$ is a solution of the traffic equations (2.11) *and $\gamma(n_i)$ given as*

$$\gamma(n_i) = \begin{cases} n_i!, & \text{if } n_i \le d_i, \\ d_i!\,(d_i)^{n_i - d_i}, & \text{if } n_i > d_i. \end{cases}$$

Proof. The proof is restricted to the case of conveyor nodes with exponential delays. The extension to deterministic delays is standard. Similar as in the BCMP theorem (Baskett et al., 1975), the deterministic delay of a conveyor node is approximated by a tandem of k conveyor nodes, each with an exponential delay with rate $k\mu_i$, and then k is taken to infinity.

To verify that the jump-over network with transition rates $q(x, y)$, $x, y \in \mathbb{S}(N)$ has a product-form stationary distribution of form (2.12), it is sufficient to find non-negative numbers $\bar{q}(y, x)$ and a collection of positive numbers $\pi(x)$ summing to

unity, such that the following two conditions are fulfilled (Kelly, 1979)

$$\bar{q}(x) = q(x), \qquad x \in \mathbb{S}(N), \tag{2.14}$$

$$\pi(x)\, q(x,y) = \pi(y)\, \bar{q}(y,x), \quad x,y \in \mathbb{S}(N), \tag{2.15}$$

where $q(x) = \sum_{y \in \mathbb{S}(N)} q(x,y)$ and $\bar{q}(x) = \sum_{y \in \mathbb{S}(N)} \bar{q}(x,y)$. Then $\bar{q}(y,x)$ are the time-reversed transition rates and $\pi(x)$ is the product-form stationary distribution of the network.

Whenever state x does not contain a blocked zone, $\bar{q}(y,x)$ is defined similarly as in a regular multi-class queueing network without jump-over blocking (Nelson, 1995). Each node in the jump-over network has exponential service times and is either a multi-class single/multi-server node, or a multi-class infinite server node which are well-known to be quasi-reversible. Then it can easily be verified that conditions (2.14) and (2.15) hold and $\pi(x)$ is given by the product form of Equation (2.12). In case state x contains a blocked zone and a transition that involves a tote skipping a zone occurs, the transition rates $q(x,y)$ and $\bar{q}(y,x)$ can be described as follows. When a tote of class $\boldsymbol{r}$ departs from the lth position of conveyor node i and moves to zone i with rate μ_{c_i}, it immediately jumps over the zone since it encounters a full buffer. The tote will move to the next conveyor node c_{i+1} with probability 1, where it joins the end of the node. When arriving in c_{i+1} the tote is either tagged as *visited* z_i with probability $1 - b_i$ such that its class becomes $\boldsymbol{s} = \boldsymbol{r} \backslash \{z_i\}$ or as *skipped* z_i with probability b_i while the class of the tote remains the same.

The transition rates of a tote skipping zone i are given as follows,

$$q\left(x, x - \boldsymbol{r}_{c_i l} + \boldsymbol{r}_{c_{i+1} n_{c_{i+1}}+1}\right) = \mu_{c_i} b_{z_i}, \qquad l = 1, \ldots, n_{c_i}, \tag{2.16}$$

$$q\left(x, x - \boldsymbol{r}_{c_i l} + \boldsymbol{s}_{c_{i+1} n_{c_{i+1}}+1}\right) = \mu_{c_i}(1 - b_{z_i}), \quad l = 1, \ldots, n_{c_i}, \tag{2.17}$$

where state $x - \boldsymbol{r}_{c_i l} + \boldsymbol{s}_{c_{i+1} n_{c_{i+1}}+1}$ denotes the removal of a class $\boldsymbol{r}$ tote at the lth position in c_i and an arrival of a class $\boldsymbol{s}$ tote in c_{i+1} at the last position.

In the time-reversed process a tote of class $\boldsymbol{r}$ or $\boldsymbol{s}$ departs from the last position in c_{i+1} with rate $\left(n_{c_{i+1}} + 1\right)\mu_{c_{i+1}}$ and joins at position l in c_i as class $\boldsymbol{r}$ with probability $\lambda_{c_i \boldsymbol{r}_{c_i l}} b_{z_i} / \lambda_{c_{i+1} \boldsymbol{r}_{c_{i+1} n_{c_{i+1}}+1}}$ if it was tagged as *skipped* z_i and with probability

$\lambda_{c_i \boldsymbol{r}_{c_i l}} (1 - b_{z_i}) / \lambda_{c_{i+1} \boldsymbol{s}_{c_{i+1} n_{c_{i+1}}+1}}$ otherwise,

$$\bar{q}\left(x - \boldsymbol{r}_{c_i l} + \boldsymbol{r}_{c_{i+1} n_{c_{i+1}}+1}, x\right) = \mu_{c_{i+1}} \frac{n_{c_{i+1}} + 1}{n_{c_i}} \frac{\lambda_{c_i \boldsymbol{r}_{c_i l}} b_{z_i}}{\lambda_{c_{i+1} \boldsymbol{r}_{c_{i+1} n_{c_{i+1}}+1}}}, \quad l = 1, \ldots, n_{c_i}, \tag{2.18}$$

$$\bar{q}\left(x - \boldsymbol{r}_{c_i l} + \boldsymbol{s}_{c_{i+1} n_{c_{i+1}}+1}, x\right) = \mu_{c_{i+1}} \frac{n_{c_{i+1}} + 1}{n_{c_i}} \frac{\lambda_{c_i \boldsymbol{r}_{c_i l}} (1 - b_{z_i})}{\lambda_{c_{i+1} \boldsymbol{s}_{c_{i+1} n_{c_{i+1}}+1}}}, \quad l = 1, \ldots, n_{c_i}. \tag{2.19}$$

Inserting (2.12)-(2.13) and (2.16)-(2.19) and canceling identical terms, it can be verified that (2.15) holds. By considering also transitions that do not skip a zone, it can be shown that $q(x) = \bar{q}(x) = \mu_e I_{(n_e>0)} + \sum_{i \in \mathcal{Z}} \min\{d_i, n_i\} \mu_i + \sum_{i \in \mathcal{C}} n_i \mu_i$, where $I_{(.)}$ is an indicator function. As a result, this means that the jump-over network has a product-form stationary distribution of the form of Equation (2.12). □

Theorem 2.1 provides a detailed description of the state of the network by specifying the order of totes in the nodes. However, knowledge of the aggregate state, i.e. the total number of totes in a node, is sufficient to determine performance statistics as the throughput and waiting times in the zones. For the description of the aggregate state, it is convenient to transform the class dependent visit ratios $\lambda_{i\boldsymbol{r}}$ into chain visit ratios

$$V_i = \frac{\sum_{\boldsymbol{r} \subseteq \mathcal{Z}} \lambda_{i\boldsymbol{r}}}{\sum_{\boldsymbol{r} \subseteq \mathcal{Z}} \lambda_{e\boldsymbol{r}}}, \quad i \in \mathcal{S}, \tag{2.20}$$

where V_i can be interpreted as the average number of times an arbitrary tote visits node i before moving to the exit node. Note that the jump-over network only has one chain of classes, with a population of N totes, due to the fact that every tote will be replaced by a tote of a possibly different class at the exit.

Corollary 2.1. *The jump-over network with state space $\mathbb{S}(N)$ has a product-form stationary distribution, in aggregated form given by*

$$\pi(\bar{n}) = \frac{1}{G} \prod_{i \in \mathcal{S}} \left(\frac{V_i}{\mu_i}\right)^{n_i} \prod_{i \in \mathcal{C}} \frac{1}{n_i!} \prod_{i \in \mathcal{Z}} \frac{1}{\gamma(n_i)}, \tag{2.21}$$

where G is a normalization constant and $\bar{n} = (n_i : i \in \mathcal{S})$ with n_i as the number of totes in node i and $\sum_{i \in \mathcal{S}} n_i = N$.

Remark 2.1. The definition of quasi-reversibility by Kelly (1979) was further generalized by e.g. Chao & Miyazawa (2000) and Henderson & Taylor (2001). They show that quasi-reversibility can also be applied to obtain product form results for queueing networks with signals, negative customers, transitions involving three or more nodes, and batch movements. This framework can also be used to describe the jump-over network.

2.2.3 Chain visit ratios

To obtain the chain visit ratios V_i, $i \in \mathcal{S}$, first linear system (2.11) must be solved. This might, however, require a large computational effort if the queueing network consists of many nodes. Also, the number of different tote classes, 2^M, grows exponentially with the number of zones. Another way to obtain the chain visit ratios V_i is to calculate them directly per node type, i.e. entrance/exit, conveyor node or zone. Clearly, V_e is 1 by (2.20). Then the chain visit ratios of the conveyors nodes and the zones can be calculated as follows.

2.2.3.1 Conveyor nodes.

A tote visits all the conveyor nodes the same number of times during its stay in the system. As a result, the chain visit ratios of the conveyor nodes V_i, $i \in \mathcal{C}$ are equal and given by the average number of circulations an arbitrary tote makes in the system before moving to the exit.

To calculate the average number of circulations, absorbing Markov chain $\{X_l, l \geq 0\}$ with a state space consisting of all subsets of $\mathcal{Z}$ and transition matrix Φ is defined. The chain starts in state $X_0 = \boldsymbol{r}$ with probability $\psi_{\boldsymbol{r}}$, which is the probability that a tote of class $\boldsymbol{r}$ is released in the system. State X_l defines the class of the tote at the last conveyor node after the lth circulation. Then the average number of circulations is equal to the expected number of transitions before entering the absorbing state $\emptyset$, i.e., the class for which all the order lines have been picked such that the tote leaves the system. The transition probability from state $\boldsymbol{r}$ at the start of the circulation to

$\boldsymbol{s}$ at the end of the circulation, $\Phi_{\boldsymbol{r},\boldsymbol{s}}$, is given by

$$\Phi_{\boldsymbol{r},\boldsymbol{s}} = \prod_{j\in\boldsymbol{s}} b_j \prod_{i\in\boldsymbol{r}\setminus\boldsymbol{s}} (1-b_i), \quad \boldsymbol{s}\subseteq\boldsymbol{r}\subseteq\mathcal{Z},$$

and zero otherwise.

Markov chain $\{X_l, l\geq 0\}$ has one absorbing state, and transitions are only possible to a state with fewer zones to visit, i.e., the states of transition matrix Φ can be ordered in such a way that Φ is an upper triangular matrix. This implies that transition matrix Φ can be rewritten in canonical form as

$$\Phi = \begin{bmatrix} \Theta & \Upsilon \\ \mathbf{0} & 1 \end{bmatrix}, \tag{2.22}$$

where Θ is an upper triangular sub-matrix of the transition probabilities between the transient states, and Υ is a column vector of the transition probabilities between the transient states and the absorbing state. The last row of Φ corresponds to the absorbing state of the Markov chain. The expected number of transitions until absorption in a Markov chain with one absorbing state is given by (Wolff, 1989)

$$V_i = \psi\left(I-\Theta\right)^{-1}\mathbf{1}, \quad i\in\mathcal{C}, \tag{2.23}$$

where I is an identity matrix, $\mathbf{1}$ a column vector with ones, and $\psi = (\psi_{\boldsymbol{r}} : \boldsymbol{r}\subseteq\mathcal{Z}\setminus\emptyset)$ a row vector with the initial release probabilities ordered in the same way as Θ. Since $(I-\Theta)$ is an upper triangular matrix, its inverse can easily be determined by back-substitution. Denote $\omega = \left(I-\Theta\right)^{-1}\mathbf{1}$, then the jth element of ω can be found by the following recursion,

$$\omega_j = (1+\sum_{k=j+1}^{2^M-1}\Theta_{j,k}\omega_k)/\left(1-\Theta_{j,j}\right), \quad j = 2^M-1, 2^M-2, \ldots, 1.$$

Remark 2.2. When the location of the system entrance and exit do not coincide, the chain visit ratios of the conveyor nodes are not all equal. Now, a tote visits the conveyor nodes from the system entrance going to the system exit a single time more often than the conveyor nodes going the opposite way. V_i, $i\in\mathcal{C}$ can still be found using the previous analysis, except that the Markov chain will now start in state

$X_0 = \boldsymbol{r}'$ with corresponding probability $\tilde{\psi}_{\boldsymbol{r}'}$, where $\boldsymbol{r}' \subseteq \mathcal{Z}$ is the class of the tote when reaching the system exit for the first time. Then, Equation (2.23) will give the average number of recirculations in the system, which equals the chain visit ratios of the conveyor nodes from the system exit to the system entrance. These should be incremented with 1 for the other conveyor nodes.

2.2.3.2 Zones.

The chain visit ratios of the zones V_i, $i \in \mathcal{Z}$ are equal to the mean number of times an arbitrary tote visits zone i before leaving the system. In the jump-over network, the number of times the tote visits zone i follows a geometric distribution with a probability of actually being tagged as *visited zone* i equal to $1 - b_i$. Hence, the chain visit ratios of zone i are

$$V_i = \sum_{\boldsymbol{r}:\, i \in \boldsymbol{r} \subseteq \mathcal{Z}} \frac{\psi_{\boldsymbol{r}}}{1 - b_i}, \quad i \in \mathcal{Z}. \tag{2.24}$$

2.2.4 Mean value analysis

A mean value analysis (MVA) algorithm (Reiser & Lavenberg, 1980) can be formulated that efficiently computes exactly the key performance statistics of the jump-over network by iterating over the total number of totes $n = 1, \ldots, N$ in the system. MVA is based on the arrival theorem which can be shown to hold also for multi-class jump-over networks by exploiting their product-form distribution. The algorithm iteratively calculates the mean throughput time $E(T_i(n))$, which is the expected time a tote will spend in node i per visit given that there are n totes in the system, the system throughput $X(n)$, the mean number of totes in a node $E(L_i(n))$, and the marginal queue length probabilities $\pi_i(j|n)$ of having j totes in zone i given that there are n totes in the network.

First, initialize $E(L_i(0)) = 0$, $i \in \mathcal{S}$ and $\pi_i(0|0) = 1$, $\pi_i(j|0) = 0$ for $j = 1, \ldots, d_i + q_i$ if $i \in \mathcal{Z}$. Then, the mean throughput time $E(T_i(n))$ of the entrance/exit and conveyor

nodes can be calculated by

$$E\left(T_i\left(n\right)\right) = \begin{cases} \frac{1}{\mu_i}\left(1 + E\left(L_i\left(n-1\right)\right)\right), & \text{if } i = e, \\ \frac{1}{\mu_i}, & \text{if } i \in \mathcal{C}. \end{cases} \tag{2.25}$$

This directly follows from the arrival theorem and the fact that the entrance/exit is a single-server node and the conveyor nodes are infinite server nodes.

The mean throughput time of the zones can be calculated by

$$E\left(T_i\left(n\right)\right) = \sum_{j=d_i}^{d_i+q_i-1}\left(j+1-d_i\right)\frac{1}{d_i\mu_i}\pi_i\left(j|n-1\right) + \frac{1}{\mu_i}\left(1-\pi_i\left(d_i+q_i|n-1\right)\right), \quad i \in \mathcal{Z}. \tag{2.26}$$

The first term of Equation (2.26) denotes the average waiting time conditioned on the number of totes, j, in the zone on arrival, and the second term is the tote's own average service time. When the buffer of the zone is full, the throughput time is 0, since the tote skips the zone.

The system throughput $X\left(n\right)$ can be calculated using $E\left(T_i\left(n\right)\right)$, $i \in \mathcal{S}$, (Reiser & Lavenberg, 1980)

$$X\left(n\right) = \frac{n}{\sum_{i\in\mathcal{S}} V_i E\left(T_i\left(n\right)\right)}, \tag{2.27}$$

where the denominator denotes the average time a tote spends in the system, i.e. system throughput time.

Applying Little's law gives the mean number of totes in a node

$$E\left(L_i\left(n\right)\right) = V_i X\left(n\right) E\left(T_i\left(n\right)\right), \quad i \in \mathcal{S}. \tag{2.28}$$

Finally, the marginal queue length probabilities can be determined by balancing the number of transitions per time unit between state $j-1$ and j, where j is the number of totes in zone i. The rate from j to $j-1$ is given by $\min\left(j, d_i\right)\mu_i\pi_i\left(j|n\right)$ and, by

the arrival theorem, the rate from $j-1$ to j is $V_i X(n)\,\pi_i(j-1|n-1)$. Hence,

$$\pi_i(j|n) = \frac{V_i X(n)}{\mu_i \min(j, d_i)} \pi_i(j-1|n-1), \quad j = 1, \ldots, d_i + q_i,\ i \in \mathcal{Z}, \tag{2.29}$$

and where $\pi_i(0|n)$ follows from normalization

$$\pi_i(0|n) = 1 - \sum_{j=1}^{d_i+q_i} \pi_i(j|n), \quad i \in \mathcal{Z}. \tag{2.30}$$

Equation (2.30) has often been reported as the cause of numerical instability in MVA (Chandy & Sauer, 1980). An alternative approach is to use Equation (27) of Reiser (1981) which is known to be numerically stable.

Sequentially applying Equations (2.25)-(2.30) allows for an iterative procedure for obtaining the performance statistics. In the last step of the MVA, the system throughput time, the fraction of time a tote encounters zone i blocked, i.e. $b_i = \pi_i(d_i + q_i|N-1)$, and the utilization of the nodes ρ_i can be obtained, which is given by

$$\rho_i = \begin{cases} X(N)/\mu_i, & \text{if } i = e, \\ V_i X(N)/\mu_i, & \text{if } i \in \mathcal{C}, \\ 1 - \sum_{j=0}^{d_i-1} ((d_i - j)/d_i)\,\pi_i(j|N), & \text{if } i \in \mathcal{Z}, \end{cases} \tag{2.31}$$

where ρ_e and ρ_i, $i \in \mathcal{Z}$, are the fraction of time the entrance/exit or the pickers in the zones are busy and where ρ_i, $i \in \mathcal{C}$ is the average number of totes at a conveyor node.

Remark 2.3. In a jump-over network with non-exponential picking times, MVA will no longer be exact. Still, closed queueing networks are known to be robust to the service distribution of a node (Bolch et al., 2006). Hence, the arrival theorem can be adopted as an approximation. When the picking times are non-exponentially distributed, the throughput time of a zone is equal to the tote's own service time if not all the order pickers are busy. If all the order pickers are busy, then a newly arriving tote has to wait for the first departure at the zone and then continues to wait for as many departures as there were totes waiting on arrival before it is served. The throughput time of a tote that encounters the zone full is still zero. Combining

these results, Equation (2.26) is replaced in the approximate MVA with,

$$E\left(T_i\left(n\right)\right) = Q_i\left(n-1\right)\frac{E\left(R_i\right)}{d_i} + \sum_{j=d_i}^{d_i+q_i-1}\left(j+1-d_i\right)\frac{E\left(B_i\right)}{d_i}\pi_i\left(j|n-1\right) + E\left(B_i\right)\left(1-\pi_i\left(d_i+q_i|n-1\right)\right), \quad i \in \mathcal{Z}, \quad (2.32)$$

where $E\left(B_i\right)$ is the expected service time of zone i, $E\left(R_i\right) = E\left(B_i^2\right)/\left(2E\left(B_i\right)\right)$ is the expected residual service time of zone i, and $Q_i\left(n-1\right) = \sum_{j=d_i}^{d_i+q_i}\pi_i\left(j|n-1\right)$ is the probability that all order pickers are busy in zone i upon an arrival instant. When the picking times are generally distributed, $\pi_i\left(j|n\right)$ can be approximated by the corresponding probabilities in a zone where each order picker has an exponential service rate $1/E\left(B_i\right)$.

2.2.5 Iterative algorithm for calculating blocking probabilities

In the jump-over network, totes are tagged independently from the state of the buffer using blocking probabilities b_i, $i \in \mathcal{Z}$. However, these blocking probabilities are not known in advance. The probability b_i can be iteratively estimated by the probability that the buffer of zone i is full using the following algorithm.

First, initialize the blocking probabilities $b_i^{(1)}$, $i \in \mathcal{Z}$ to 0. Then, calculate the marginal queue length probabilities using Equations (2.29) and (2.30) and take the fraction of totes that find the zone containing $d_i + q_i$ totes as a new estimate for the blocking probability. Thus, by the arrival theorem,

$$b_i^{(m+1)} = \pi_i^{(m)}\left(d_i+q_i|N-1\right), \quad i \in \mathcal{Z}, \quad (2.33)$$

where the superscript indicates in which iteration the quantities have been calculated. Based on this new estimate for b_i, the routing probabilities and subsequently the chain visit ratios are updated for the zones and conveyor nodes. Equation (2.33) can be evaluated again after applying MVA in order to get a better estimate of the blocking probabilities and so on. By repeating this process until for all i the differences between $b_i^{(m+1)}$ and $b_i^{(m)}$ is less than a small ϵ, the algorithm terminates and the performance statistics are calculated. In our experience, convergence is reached fast, and does not depend on the initial starting values of b_i.

2.2.6 Example of the single-segment routing model

In order to illustrate the performance and accuracy of the iterative algorithm of Section 2.2.5, consider the zone picking system with two zones shown in Figure 2.2. In total there are 2^2 different tote classes, due to class changing after a zone or at the entrance. The release probabilities are set to $\psi_{\emptyset} = 0$ and $\psi_{\{z_1\}} = \psi_{\{z_2\}} = \psi_{\{z_1,z_2\}} = 1/3$, the service times for the entrance/exit are exponentially distributed with mean $\mu_e^{-1} = 5$ seconds, the travel times on the conveyor nodes are deterministic, $\mu_{c_1}^{-1} = \mu_{c_2}^{-1} = \mu_{c_3}^{-1} = 100$ seconds, and the service times in the zones are exponentially distributed with mean $\mu_{z_1}^{-1} = \mu_{z_2}^{-1} = 15$ seconds. The number of order pickers in both zones $d_{z_1} = d_{z_2}$ are equal to 1 and the buffer sizes of the zones are respectively $q_{z_1} = 2$ and $q_{z_2} = 1$.

Table 2.1 gives the average time in seconds a tote spends on the conveyor $E(T_{\mathcal{C}}(N))$ and at the zones $E(T_{\mathcal{Z}}(N))$, and the overall throughput rate per hour $X(N)$. These statistics are shown for the jump-over network (*Jump*), the same closed queueing network but with infinite buffers in the zones (*CQN*) and the approximation of Yu & De Koster (2008) (*YdK*). *YdK* uses an open queueing network in its analysis. Using bisection, the arrival rate of this approximation is set such that the average number of totes in the open network is equal to N. The results show that the jump-over network produces very accurate results compared to the simulation of the original queueing network (*Sim*), where the half width of the 95% confidence interval is given between brackets and the errors are calculated by; $(x - Sim)/Sim \times 100\%$ where x is *Jump*, *CQN*, or *YdK*. In all cases, the algorithm stopped after 5 iterations with $\epsilon = 10^{-3}$. Both *CQN* and *YdK* assume infinite buffers, which means that they cannot estimate the blocking probabilities. The run times for the *Jump*, *CQN*, and *YdK* methods is less than a second on a Core i7 with 2.4 GHz and 8 GB of RAM, whereas the simulation for $N = 100$ takes around 30 seconds.

When the total number of totes in the system N is small, the errors of the jump-over network are negligible and relatively small for *CQN* and *YdK*. This is obvious since almost no blocking occurs in the system, i.e. only 5% of the totes that intends to visit the second zone are blocked. This means the jump-over network will have almost the same performance as *CQN* and the original queueing network. However, a higher N increases the blocking probability and the number tote recirculations.

Table 2.1: The performance statistics obtained for the example with varying number of totes N.

N	$X(N)$ (in hours^{-1})							$E(T_{\mathcal{Z}}(N))$ (in seconds)						
					Err (%)							Err (%)		
	Sim	*Jump*	*CQN*	*YdK*	*Jump*	*CQN*	*YdK*	*Sim*	*Jump*	*CQN*	*YdK*	*Jump*	*CQN*	*YdK*
10	104.4 (±0.16)	104.5	108.2	107.9	0.13	3.62	3.39	25.2 (±0.06)	25.3	27.1	28.6	0.12	7.27	13.15
20	182.8 (±0.25)	182.9	206.5	204.9	0.04	12.93	12.06	29.9 (±0.08)	29.8	41.8	46.4	0.28	40.00	55.29
30	234.3 (±0.52)	235.3	283.5	276.2	0.44	21.03	17.92	33.3 (±0.13)	33.1	72.8	86.0	0.58	119.07	158.47
40	268.8 (±0.58)	269.8	326.2	313.4	0.40	21.39	16.61	35.5 (±0.08)	35.5	132.4	154.5	0.02	273.31	335.67
50	291.5 (±0.39)	293.0	342.0	330.1	0.52	17.34	13.24	37.3 (±0.08)	37.3	216.8	240.4	0.06	481.65	544.82
100	336.4 (±0.65)	338.6	354.9	350.1	0.67	5.50	4.07	42.4 (±0.22)	42.3	704.6	723.4	0.17	1563.16	1607.49

N	$E(T_{\mathcal{C}}(N))$ (in seconds)							b_{z_1}			b_{z_2}		
					Err (%)								
	Sim	*Jump*	*CQN*	*YdK*	*Jump*	*CQN*	*YdK*	*Sim*	*Jump*	Err (%)	*Sim*	*Jump*	Err (%)
10	313.7 (±0.52)	313.4	300.0	300.0	0.08	4.37	4.37	0.01 (±0.00)	0.01	1.97	0.05 (±0.00)	0.05	0.62
20	357.4 (±0.54)	357.3	300.0	300.0	0.04	16.07	16.07	0.07 (±0.00)	0.07	1.92	0.18 (±0.00)	0.18	0.06
30	420.1 (±1.05)	418.7	300.0	300.0	0.34	28.59	28.59	0.16 (±0.00)	0.15	4.14	0.31 (±0.00)	0.31	0.27
40	491.9 (±0.96)	490.4	300.0	300.0	0.32	39.02	39.02	0.24 (±0.00)	0.23	3.40	0.41 (±0.00)	0.41	0.23
50	571.6 (±0.72)	568.8	300.0	300.0	0.49	47.51	47.51	0.32 (±0.00)	0.31	3.11	0.50 (±0.00)	0.50	0.41
100	1017.7 (±1.93)	1011.5	300.0	300.0	0.61	70.52	70.52	0.57 (±0.01)	0.55	4.41	0.72 (±0.00)	0.73	1.01

When the number of totes in the system equals $N = 40$ or 50, blocking becomes more prominent. Since every zone is visited with the same frequency, the totes are more often blocked at zone 2 than at zone 1, due to the buffer sizes of the zones. Moreover, the system throughput time starts to increase rapidly, while the throughput rate stabilizes because all the zones become saturated. *CQN* and *YdK* produce large errors in the average time a tote spends at the zones and conveyor nodes, which is due to the assumption of infinite buffers in the zones. This does not happen in the jump-over network. Because of recirculation, the conveyor nodes act as buffers for totes that cannot enter a zone. When $N = 100$, blocking seriously impacts the performance of the system and totes spend twice as long in the system compared to $N = 50$.

2.3 Multi-segment zone picking systems

In this section, the single-segment routing model is extended to multi-segment routing. In a zone picking system with multi-segment routing, each segment consists of a number of zones connected by a conveyor with recirculation. The segments are connected to the main conveyor, which forms the center of the zone picking system. In order to analyze the system, again three different types of elements can be distinguished: the entrance/exit stations, the conveyors, and the zones. Furthermore, the entrance/exit stations are divided in the system entrance/exit and the segment entrance stations, and the conveyor nodes are split into main and segment conveyor nodes. An example of a zone picking system with multi-segment routing is shown in Figure 2.3, which has six segments that contain a different number of zones.

A zone picking system with multi-segment routing works very similar to the system described in Section 2.2. Upon release at the system entrance, a tote is transported to the first segment where order lines have to be picked. The tote enters this segment via the segment entrance station and stays in the segment until it has visited all the required zones within the segment. When finished, the tote leaves the segment and is transported either to another segment or to the system exit in case the picking process has finished. When a segment is considered in isolation, it is equivalent to the single-segment model, except that the entrance station is now modeled as a conveyor.

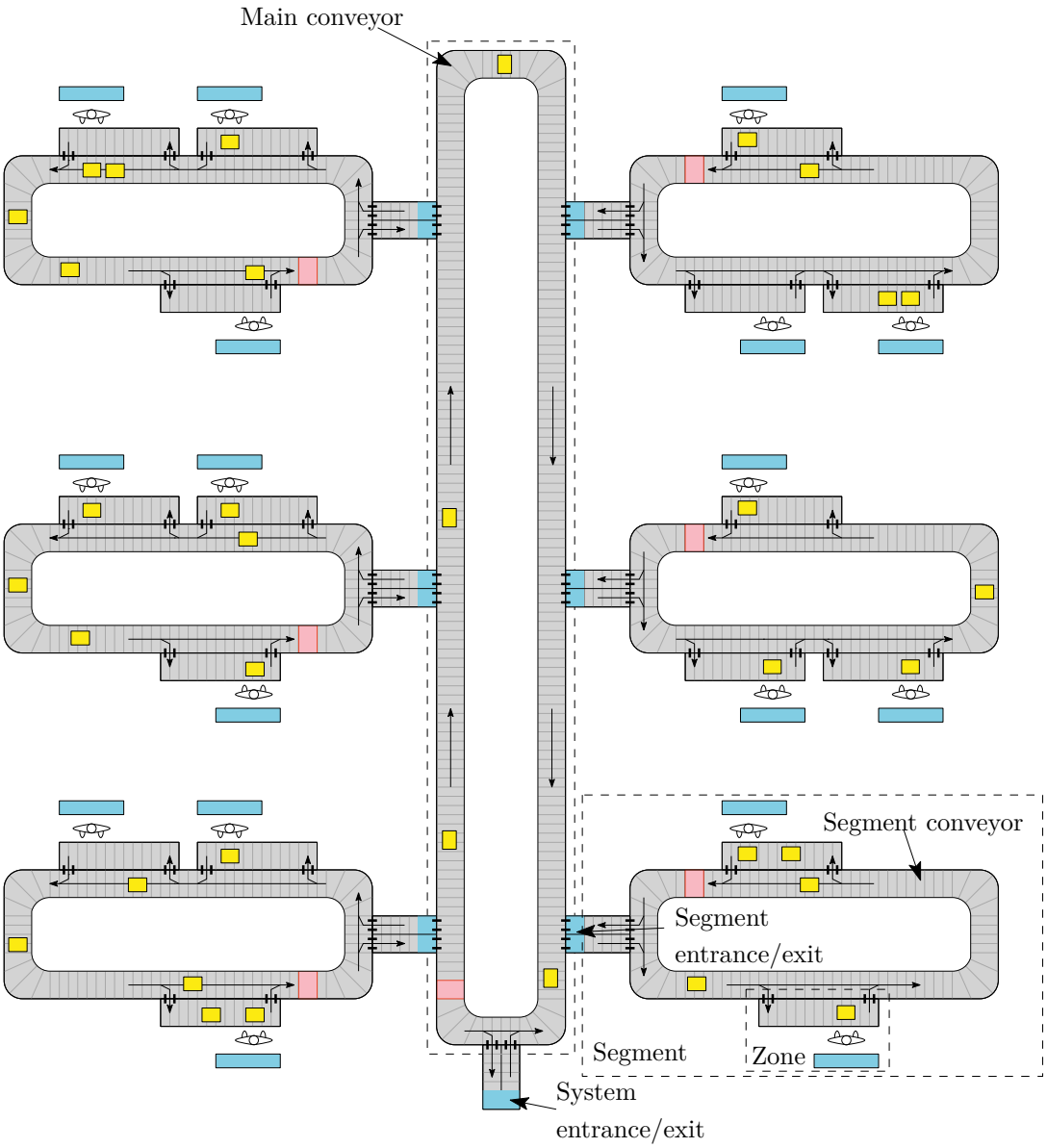


Figure 2.3: A multi-segment zone picking system with six segments and seventeen zones.

The workload control mechanism controls both the maximum number of totes in the system and, in addition, the maximum number of totes within each segment. If a tote tries to enter a segment that is fully saturated, the control mechanism withholds the tote from entering. The blocked tote will skip the segment and stay on the main conveyor, potentially visiting other segments before again attempting to enter this segment. This is very similar to the situation when a tote is blocked by a zone, but now blocking depends on the number of totes within an entire segment instead that of a single zone.

The queueing network of Section 2.2 is extended to a zone picking system with multi-segment routing. Let the extended model consist of K segments. Denote the entrance/exit stations by $\mathcal{E} = \{e_0, e_1, \dots e_k, \dots, e_K\}$ where e_0 is the system entrance/exit station and e_k the entrance station of segment k, that represents the conveyor connecting the segment with the main conveyor. Let $\mathcal{Z} = \cup_{k=1}^{K} \mathcal{Z}^k$ be the union of zones, where $\mathcal{Z}^k = \left\{z_1^k, \dots, z_{m^k}^k\right\}$ are the zones within segment k, such that $\sum_{k=1}^{K} m^k = M$. Denote $\mathcal{C} = \cup_{k=0}^{K} \mathcal{C}^k$ as the union of the conveyor nodes where $\mathcal{C}^0 = \left\{c_1^0, \dots, c_{K+1}^0\right\}$ are the main conveyor nodes and $\mathcal{C}^k = \left\{c_1^k, \dots, c_{m^k+1}^k\right\}$ the segment conveyor nodes within segment k. Finally, let $\mathcal{S} = \mathcal{E} \cup \mathcal{C} \cup \mathcal{Z}$ be the union of all the nodes in the network. Figure 2.4 shows the topology of the multi-segment routing queueing network with K segments.

The system is partitioned into $K+1$ subsystems: $\left\{\mathcal{H}^0, \mathcal{H}^1, \dots, \mathcal{H}^k, \dots, \mathcal{H}^K\right\}$, where $\mathcal{H}^0 = \{e_0\} \cup \mathcal{C}^0$ consists of the system entrance/exit and the nodes on the main conveyor, and $\mathcal{H}^k = \{e_k\} \cup \mathcal{C}^k \cup \mathcal{Z}^k$ the set of nodes belonging to the kth segment. The following additional assumptions are adopted for the network:

- Each tote has a class $\boldsymbol{r} \subseteq \mathcal{Z}$ defining the zones the tote should visit. Let $\boldsymbol{r}^k \subseteq \mathcal{Z}^k$, $k = 1, \dots, K$ describe the zones a class $\boldsymbol{r}$ tote has to visit within segment k. A tote will enter segment k if and only if $\boldsymbol{r}^k \neq \emptyset$.
- The entrance station e_k to segment k is assumed to be an infinite-server node with a deterministic service of rate μ_{e_k}, $k = 1, \dots, K$ that accounts for the time the tote needs for entering and leaving the segment.
- The maximum number of totes allowed in segment k, $k = 1, \dots, K$ is $N^k \leq N$, which is controlled by the workload control mechanism.

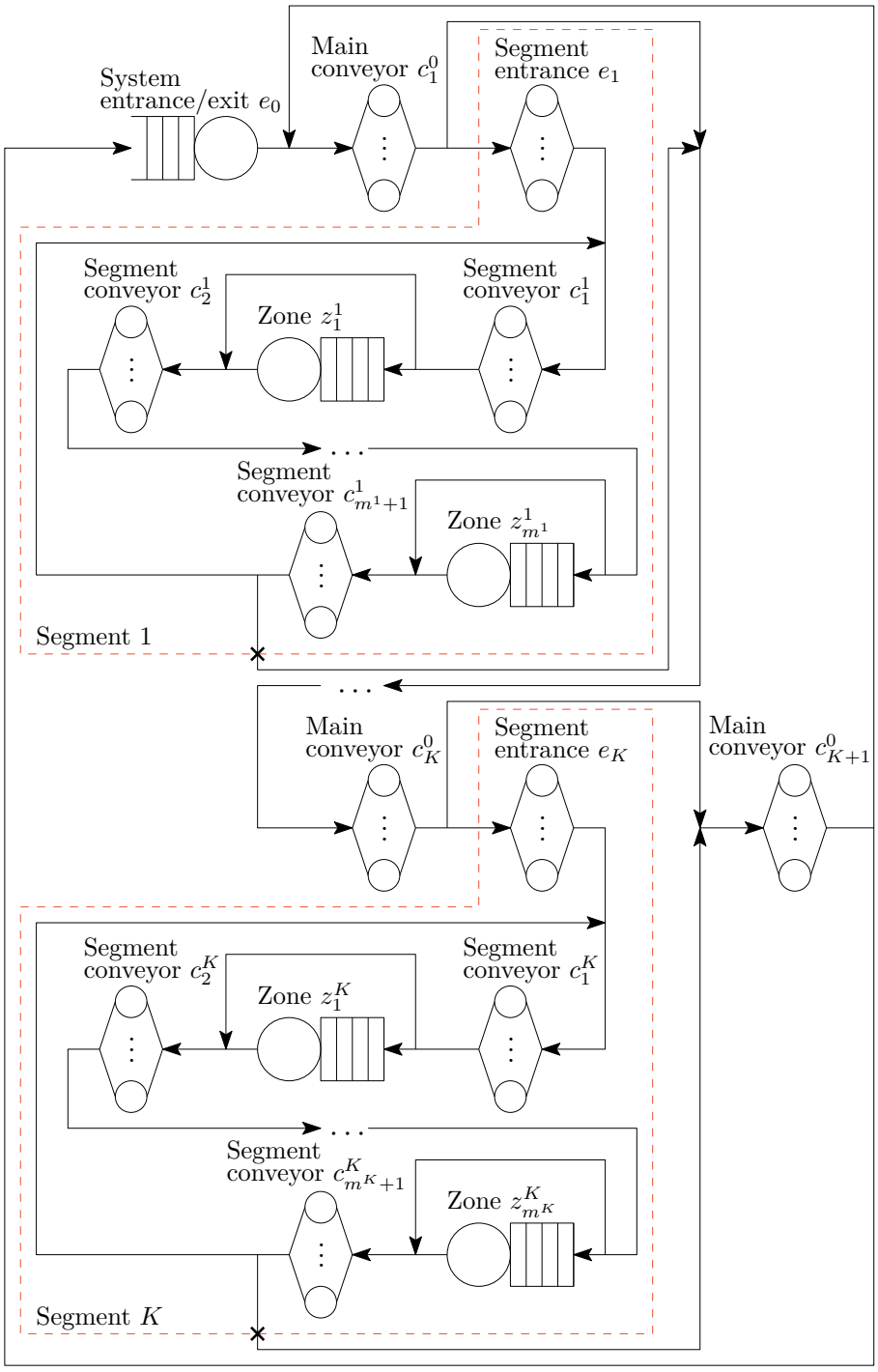


Figure 2.4: A multi-segment zone picking queueing network with K segments.

At the system entrance, new totes of class $\boldsymbol{r} \subseteq \mathcal{Z}$ are released with probability $\psi_{\boldsymbol{r}}$. After release, a tote of class $\boldsymbol{r}$ is transported from the system entrance to the first main conveyor node c_1^0. From c_k^0, the tote is either transported to segment entrance e_k if $\boldsymbol{r}^k \neq \emptyset$ or move to the next main conveyor node c_{k+1}^0. Whenever the number of totes in segment k equals N^k, the tote skips the segment and also moves to c_{k+1}^0, while its class remains the same. In case the tote actually enters the segment, it resides in the segment until it has visited all the required zones. Then the tote leaves the segment via $c_{m^k+1}^k$ and its class has changed from $\boldsymbol{r}$ to $\boldsymbol{s} = \boldsymbol{r} \backslash \boldsymbol{r}^k$. After visiting the last main conveyor node c_{K+1}^0, all totes with $\boldsymbol{r} \neq \emptyset$ are routed to the first main conveyor node c_1^0; the other totes are transported to the exit and are immediately replaced by a new tote which waits for release at the entrance.

Summarizing, the state dependent routing probability for the multi-segment system are defined as follows:

$$p_{e_0\emptyset, c_1^0\boldsymbol{r}}(x) = \psi_{\boldsymbol{r}}, \tag{2.34}$$

$$p_{c_k^0\boldsymbol{r}, e_k\boldsymbol{r}}(x) = 1,\ k = 1, \ldots, K,\ \boldsymbol{r}^{\boldsymbol{k}} \neq \emptyset \text{ and } \textstyle\sum_{i \in \mathcal{H}^k} n_i \leq N^k, \tag{2.35}$$

$$p_{c_k^0\boldsymbol{r}, c_{k+1}^0\boldsymbol{r}}(x) = 1,\ k = 1, \ldots, K,\ \boldsymbol{r}^k = \emptyset \text{ or } \textstyle\sum_{i \in \mathcal{H}^k} n_i = N^k, \tag{2.36}$$

$$p_{c_{K+1}^0\boldsymbol{r}, e_0\boldsymbol{r}}(x) = 1,\ \boldsymbol{r} = \emptyset, \tag{2.37}$$

$$p_{c_{K+1}^0\boldsymbol{r}, c_1^0\boldsymbol{r}}(x) = 1,\ \boldsymbol{r} \neq \emptyset, \tag{2.38}$$

$$p_{e_k\boldsymbol{r}, c_1^k\boldsymbol{r}}(x) = 1,\ k = 1, \ldots, K, \tag{2.39}$$

$$p_{c_i^k\boldsymbol{r}, z_i^k\boldsymbol{r}}(x) = 1,\ k = 1, \ldots, K,\ i = 1, \ldots, m^k,\ z_i^k \in \boldsymbol{r} \text{ and } n_{z_i^k} < d_{z_i^k} + q_{z_i^k}, \tag{2.40}$$

$$p_{c_i^k\boldsymbol{r}, c_{i+1}^k\boldsymbol{r}}(x) = 1,\ k = 1, \ldots, K,\ i = 1, \ldots, m^k,\ z_i^k \notin \boldsymbol{r} \text{ or } n_{z_i^k} = d_{z_i^k} + q_{z_i^k}, \tag{2.41}$$

$$p_{z_i^k\boldsymbol{r}, c_{i+1}^k\boldsymbol{s}}(x) = 1,\ k = 1, \ldots, K,\ i = 1, \ldots, m^k,\ \boldsymbol{s} = \boldsymbol{r} \backslash \left\{z_i^k\right\}, \tag{2.42}$$

$$p_{c_{m^k+1}^k\boldsymbol{r}, c_{k+1}^0\boldsymbol{r}}(x) = 1,\ k = 1, \ldots, K,\ \boldsymbol{r}^k = \emptyset, \tag{2.43}$$

$$p_{c_{m^k+1}^k\boldsymbol{r}, c_1^k\boldsymbol{r}}(x) = 1,\ k = 1, \ldots, K,\ \boldsymbol{r}^k \neq \emptyset. \tag{2.44}$$

Just as before, the first step of the analysis is to approximate the multi-segment queueing network of Section 2.3 by a network with jump-over blocking in Section 2.3.1 and Section 2.3.2. In Section 2.3.3 it is shown that the visit ratios of the multi-segment jump-over network again have closed form expressions. The performance statistics of this jump-over network are calculated in Section 2.3.4 using flow equivalent servers

(Chandy et al., 1975) and MVA. The iterative algorithm for estimating the blocking probabilities is presented in Section 2.3.5.

2.3.1 Jump-over network

In the multi-segment queueing network, totes can be blocked either by a zone or by a segment. A similar approach as described in Section 2.2 can be used to approximate segment blocking. In the real system totes of class $\boldsymbol{r}$ that have to visit segment k are "tagged" after segment k with the labels *visited segment k* or *skipped segment k* depending on whether they received service or skipped the segment. This is now approximated by tagging a tote randomly and independently of whether the tote actually visited segment k or not. This approximation renders a jump-over network, where a tote skipping segment k will immediately move from the start to the end of the segment, which is indicated by a cross in Figure 2.4. Here, the skipping tote will act as a "regular" tote that actually visited the segment so $\boldsymbol{r}^k$ is set to $\emptyset$.

When a tote is tagged as *skipped segment k,* the class of the tote should revert to its class when it entered the segment. However, for all totes leaving the segment there is no knowledge about which zones the tote visited in the segment. Therefore, in the jump-over network, the classes are extended such that they also include the initial class of the tote when it entered the system. Denote the new classes by $\bar{\boldsymbol{r}} = \{\boldsymbol{h}, \boldsymbol{r}\}$, where $\boldsymbol{h} \subseteq \mathcal{Z}$ is the initial class of the tote and $\boldsymbol{r} \subseteq \mathcal{Z}$ the current set of zones the tote still needs to visit. The initial class $\boldsymbol{h}$ only changes when the tote is replaced by a new tote, whereas $\boldsymbol{r}$ changes to $\boldsymbol{s} = \boldsymbol{r} \backslash \left\{z_i^k\right\}$ if zone i in segment k is tagged as visited.

Let blocking probability B_k, $k = 1, \ldots, K$ be the fraction of totes that receive the *skipped segment k* tag in the real system. Then, for each class $\bar{\boldsymbol{s}} = \{\boldsymbol{h}, \boldsymbol{s}\}$ tote leaving segment k, i.e., when $\boldsymbol{s}^k = \emptyset$, independent of whether the tote visited segment k or not (because of a fully saturated segment), $p_{c^k_{m^k+1}\bar{\boldsymbol{s}}, c^0_{k+1}\bar{\boldsymbol{r}}} = B_k$, $k = 1, \ldots K$, where $\bar{\boldsymbol{r}} = \left\{\boldsymbol{h}, \boldsymbol{s} \cup \boldsymbol{h}^k\right\}$ and $\boldsymbol{h}^k$ are the zones the tote was required to visit in segment k. This means that a tote is tagged as *skipped segment k* and is routed to next main conveyor node c^0_{k+1} leaving segment k with the same class $\bar{\boldsymbol{r}}$ as it entered the segment. Otherwise, the tote is tagged as *visited segment k* and the class of the tote does not change, i.e, $p_{c^k_{m^k+1}\bar{\boldsymbol{s}}, c^0_{k+1}\bar{\boldsymbol{s}}} = 1 - B_k$, $k = 1, \ldots K$. Just as in Section 2.2.1, the blocking

probabilities B_k are not known in advance and need to be estimated, as is shown in Section 2.3.5.

2.3.2 Product-form of the stationary distribution

In case of the multi-segment jump-over network, a state of the network x is defined in the same way as in Section 2.2.2. Let $\bar{\mathbb{S}}(N)$ be the state space of the network where in each state x the number of totes in the system is N and where the number of totes in each zone and segment satisfy both $n_i \leq d_i + q_i$, $i \in \mathcal{Z}$ and $\sum_{i \in \mathcal{H}^k} n_i \leq N^k, k = 1, \ldots, K$ respectively. The existence of a product-form solution in the network can again be proven using conditions (2.14) and (2.15).

Theorem 2.2. *The jump-over network with state space $\bar{\mathbb{S}}(N)$ has a product-form stationary distribution of the form*

$$\pi(x) = \frac{1}{G} \prod_{i \in \mathcal{S}} \pi_i(x_i), \tag{2.45}$$

where G is a normalization constant, $\pi_i(x_i)$ is the stationary distribution of node i, $i \in \mathcal{S}$ given by (2.13)*, where $\lambda_{i\boldsymbol{r}_{il}}$ is replaced by $\lambda_{i\bar{\boldsymbol{r}}_{il}}$.*

Proof. As in the proof of Theorem 2.1, we restrict the proof to the case of conveyor nodes with an exponential delay with rate μ_i, $i \in \mathcal{C}$. Whenever state x does not contain a blocked segment, it was shown in Theorem 2.1 that conditions (2.14) and (2.15) hold. In case state x contains a blocked segment and a transition that involves a tote skipping a segment occurs, the transition rate $q(x, y)$ is given as follows. A tote of class $\bar{\boldsymbol{r}}$ departs from the lth position of main conveyor node c_k^0 with rate $\mu_{c_k^0}$ and it immediately jumps over the entire segment. The tote will move to the next main conveyor node c_{k+1}^0 with probability 1, where it joins the end of the node. When arriving in c_{k+1}^0 the tote is tagged with either as *visited segment k* or *skipped segment k*. Hence,

$$q\left(x, x - \bar{\boldsymbol{r}}_{c_k^0 l} + \bar{\boldsymbol{r}}_{c_{k+1}^0 n_{c_{k+1}^0}+1}\right) = \mu_{c_k^0} B_k, \qquad l = 1, \ldots, n_{c_k^0}, \tag{2.46}$$

$$q\left(x, x - \bar{\boldsymbol{r}}_{c_k^0 l} + \bar{\boldsymbol{s}}_{c_{k+1}^0 n_{c_{k+1}^0}+1}\right) = \mu_{c_k^0}(1 - B_k), \qquad l = 1, \ldots, n_{c_k^0}. \tag{2.47}$$

The time-reversed transition rates $\bar{q}(y, x)$ are analogous to (2.18) and (2.19). Then it can be verified, similarly as in Theorem 2.1, that conditions (2.14) and (2.15) hold and the jump-over network has a product-form stationary distribution of the form of Equation (2.45). □

Theorem 2.2 again provides a detailed description of the state of the jump-over network. Since performance statistics as the throughput and mean waiting times in the zones are of interest, the aggregated state of the network will suffice and can be obtained similar as in Corollary 2.1.

2.3.3 Chain visit ratio

The chain visit ratios of the jump-over network can be computed directly per node type similar as in Section 2.2.3. Let the chain visit ratio of the system entrance/exit be normalized to $V_{e_0} = 1$. Next, the chain visit ratios are derived in the following order for the main conveyor nodes, segment entrances, segment conveyor nodes, and the zones.

2.3.3.1 Main conveyor nodes and segment entrances.

The chain visit ratios of the main conveyor nodes and the segment entrances can be computed as the conveyor nodes and the zones in the single-segment routing model. This follows from the fact that a tote needs to visit all the main conveyor nodes the same number of times during its stay in the network, and the number of visits to the segment entrances depends on the number of times a tote intends to visit a segment. The difference is now that the visit ratios depend on the blocking probabilities of the segments B_k, instead of those of the zones. The chain visit ratios of the main conveyor nodes V_i, $i \in \mathcal{C}^0$ are given by Section 2.2.3.1 and the segment entrances V_i, $i \in \mathcal{E} \setminus \{e_0\}$ by Section 2.2.3.2. For both, the tote classes $\boldsymbol{r} \subseteq \mathcal{Z}$ are replaced by the aggregated segment classes $\boldsymbol{k} \subseteq \{1, \ldots, K\}$ that define the segments a tote should visit. The corresponding release probabilities of the segment classes $\hat{\psi}_{\boldsymbol{k}}$ can be obtained by summing over all the class specific release probabilities of totes that need to visit the segments contained in $\boldsymbol{k}$. Finally, the blocking probabilities of the zones b_i are replaced by the blocking probabilities of the segments B_k.

2.3.3.2 Segment conveyor nodes and zones.

Within a segment, the network behaves exactly the same as in the single-segment routing model. This means that the chain visit ratios of the nodes in $\mathcal{H}^k$, only depend on the blocking probabilities of the zones in $\mathcal{Z}^k$. The chain visit ratios of the segment conveyor nodes $V_i, i \in C^k$ are given by Section 2.2.3.1 where the tote classes are now replaced by $\boldsymbol{r}^k \subseteq \mathcal{Z}^k$. The corresponding release probabilities for segment k are $\psi^k = \left(\psi^k_{\boldsymbol{r}^k} : \boldsymbol{r}^k \subseteq \mathcal{Z}^k \backslash \emptyset\right)$, where $\psi^k_{\boldsymbol{r}^k} = \sum_{\boldsymbol{s} \subseteq \mathcal{Z}} \psi_{\boldsymbol{s}} I_{\left(\boldsymbol{s}^k = \boldsymbol{r}^k\right)} / \sum_{\boldsymbol{s} \subseteq \mathcal{Z}} \psi_{\boldsymbol{s}} I_{\left(\boldsymbol{s}^k \neq \emptyset\right)}$ is the normalized sum of all tote classes that need to visit the zones $\boldsymbol{r}^k \subseteq \mathcal{Z}^k$ and $I_{(.)}$ an indicator function. Then by calculating the expected number of transitions until entering the absorbing state, i.e. when the tote has to leave the segment, $V_i, i \in C^k$ is obtained by multiplying the number of times the tote intends to visit the segment V_{e_k} with the average number of circulations a tote makes within segment k

$$V_i = V_{e_k} \psi^k \left(I - \Theta^k\right)^{-1} \mathbf{1}, \quad i \in \mathcal{C}^k,\ k = 1, \ldots, K, \tag{2.48}$$

where Θ^k is defined similar as in (2.22).

A similar argument holds for the chain visit ratios of the zones V_i, $i \in \mathcal{Z}$ which are given by Section 2.2.3.2. Hence,

$$V_i = V_{e_k} \sum_{\boldsymbol{r}^k :\, i \in \boldsymbol{r}^k \subseteq \mathcal{Z}^k} \frac{\psi^k_{\boldsymbol{r}^k}}{1 - b_i}, \quad i \in \mathcal{Z}^k,\ k = 1, \ldots, K. \tag{2.49}$$

2.3.4 Aggregation technique

In order to analyze the jump-over network, an extended version of the MVA presented in Section 2.2.4 can be formulated. However, it is more efficient to aggregate the jump-over network by replacing all segments by flow equivalent server centers with load-dependent service rates (Chandy et al., 1975). Norton's theorem states that the stationary distribution of the rest of the network remains unchanged after this modification (Chandy et al., 1975; Walrand, 1983; Boucherie, 1998).

Based on the product-form of Section 2.3.2 and the fact that each tote enters or leaves a segment via a single input/output node, an equivalent queueing network can be analyzed along the same lines as the analysis of the single-segment routing model.

The first step of the aggregation technique is to replace all segments by flow equivalent servers with load-dependent service rates. These rates can be determined by calculating the throughput $X^k(n)$ of subsystem $\mathcal{H}^k$ in isolation when varying the number of totes n from 1 up to N^k. The isolated subsystem can be obtained by short-circuiting all nodes that are not in $\mathcal{H}^k$. As a result, a tote leaving subsystem $\mathcal{H}^k$ will instantaneously be routed back to the entrance of the subsystem. Since every subsystem $\mathcal{H}^k$, $k = 1, \ldots, K$ in isolation is (almost) equivalent to the single-segment routing model, it can be analyzed using the MVA presented in Section 2.2.4, where only the entrance station is now an infinite server node.

Next, an equivalent queueing network can be constructed by replacing each subsystem $\mathcal{H}^k$, $k = 1, \ldots, K$ in the jump-over network by a flow equivalent server center. The server rates of the kth flow equivalent server are equal to the throughputs $X^k(n)$ of the isolated subsystems, so

$$\mu_{FES_k}(n) = X^k(n), \quad n = 1, \ldots, N^k,\ k = 1, \ldots, K. \tag{2.50}$$

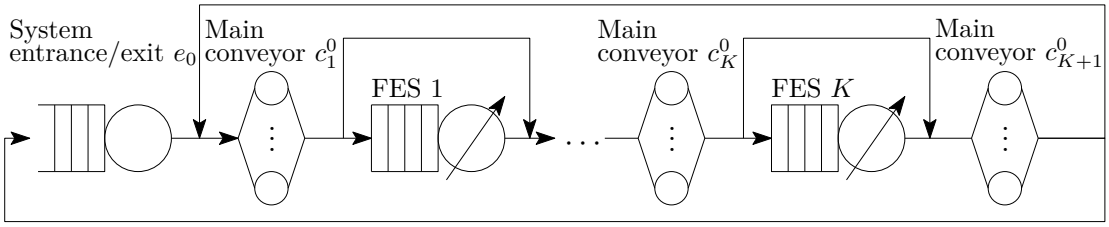


Figure 2.5: The equivalent network of the jump-over network. Segments are replaced by flow equivalent service centers with load-dependent service rates.

In Figure 2.5 the equivalent network of Figure 2.4 is shown, which is identical to Figure 2.2b except that the zones are replaced by flow equivalent service centers. The MVA of Section 2.2.4 can be applied to analyze the system, where (2.26) should be replaced by the mean throughput time of a tote in subsystem $\mathcal{H}^k$ (Reiser, 1981),

$$E(T_{FES_k}(n)) = \sum_{j=1}^{N^k-1} \frac{j}{\mu_{FES_k}(j)} \Pi_{FES_k}(j-1|n-1), \quad k = 1, \ldots, K, \tag{2.51}$$

where the marginal queue length probabilities $\Pi_{FES_k}(j|n)$ of having j totes in the kth flow equivalent server in a network with n circulating totes. These can be obtained

similar as in (2.29) by

$$\Pi_{FES_k}(j|n) = \frac{V_{FES_k} X(n)}{\mu_{FES_k}(j)} \Pi_{FES_k}(j-1|n-1), \quad j = 1, \ldots, N^k,\ k = 1, \ldots, K, \quad (2.52)$$

where V_{FES_k} is the visit ratio of the kth flow equivalent server, which is equal to the visit ratio of segment entrance e_k. Equation (2.51) is obtained by application of Little's law to the kth flow equivalent server and substitution of (2.52).

The performance statistics obtained from the equivalent (aggregate) network correspond with the aggregated performance statistics of the segments in the original jump-over network, e.g. the mean throughput time of subsystem $\mathcal{H}^k$, $E(T_{FES_k}(N))$, and the marginal queue length probabilities $\Pi_{FES_k}(j|N)$ of having j totes in subsystem $\mathcal{H}^k$ when there are N totes in the system.

The detailed performance statistics of the nodes within the segments can be obtained by a disaggregation step. Let the marginal queue length probabilities of subsystem $\mathcal{H}^k$ analyzed in isolation be $\pi_i^k(j \mid n)$, where j is the number of totes in node i in segment k, given the number of totes in segment k is n. Then, the detailed marginal queue length probabilities $\pi_i(j|N)$ are given by (Baynat & Dallery, 1993)

$$\pi_i(j|N) = \begin{cases} \Pi_i(j|N), & \text{if } i \in \mathcal{H}^0, \\ \sum_{l=1}^{N^k} \pi_i^k(j \mid l)\, \Pi_{FES_k}(l|N), & \text{if } i \in \mathcal{H}^k. \end{cases} \quad (2.53)$$

The performance statistics of all the nodes can now be calculated. The utilization of the system entrance (with $d_i = 1$) and zones can be calculated as

$$\rho_i = 1 - \sum_{j=0}^{d_i-1} \frac{d_i - j}{d_i} \pi_i(j|N), \quad i \in e_0 \cup \mathcal{Z}. \quad (2.54)$$

The mean number of totes in a node is given by

$$E(L_i(N)) = \sum_{j=1}^{\sigma_i} j \pi_i(j|N), \quad i \in \mathcal{S}, \quad (2.55)$$

where σ_i equals the number of totes in the system N if $i \in \mathcal{H}^0$, the segment capacity N^k if $i \in \mathcal{H}^k \backslash \mathcal{Z}^k$ and $d_i + q_i$ if $i \in \mathcal{Z}$.

Applying Little's law gives the mean throughput time in a node

$$E\left(T_i\left(N\right)\right) = E\left(L_i\left(N\right)\right) / V_i X\left(N\right), \quad i \in \mathcal{S}, \tag{2.56}$$

where $X\left(N\right)$ is the overall throughput rate from the equivalent aggregate network.

2.3.5 Iterative algorithm for calculating the blocking probabilities

As in Section 2.2.4, totes are tagged independently from the state of the network using the blocking probabilities; b_i, $i \in \mathcal{Z}$ and B_k, $k = 1, \ldots, K$. These blocking probabilities are not known in advance, but they can be iteratively estimated by the probabilities that the buffer of the zone is full or a segment is saturated.

First, blocking probabilities $b_i^{(1)}$, $i \in \mathcal{Z}$ and $B_k^{(1)}$, $k = 1, \ldots, K$ are initialized to 0. Then, calculate the marginal queue length probabilities of the equivalent network and use the fraction of arrivals that see a segment being saturated as a new estimate for the blocking probabilities of the segments, so by the arrival theorem,

$$B_k^{(m+1)} = \Pi_{FES_k}^{(m)}\left(N^k | N-1\right), \quad k = 1, \ldots, K, \tag{2.57}$$

where the superscript denotes the iteration number. Using the detailed marginal queue length probabilities of Equation (2.53) and by calculating the fraction of arrivals in segment k that encounter a full buffer in zone i, the new estimates for the blocking probabilities of the zones are given by

$$\begin{aligned} b_i^{(m+1)} &= \pi_i^{(m)}\left(d_i + q_i | N^k - 1, N-1\right), \\ &= \sum_{l=1}^{N^k-1} \pi_i^k\left(d_i + q_i \mid l\right) \Pi_{FES_k}\left(l | N-1\right), \quad i \in \mathcal{Z}. \end{aligned} \tag{2.58}$$

which is the probability of encountering a full buffer in zone i in a network containing $N-1$ totes, where in segment k a maximum of $N^k - 1$ totes is allowed. Note the

remarkable feature that a tote arriving at a zone sees the network in equilibrium without itself *and* in which the capacity of the segment is reduced by 1.

With (2.57) and (2.58), the routing probabilities and subsequently the visit ratios are updated for all nodes in the network. Applying the MVA equations, (2.57) and (2.58) are updated to obtain better estimates. By repeating this process until for all i, the differences $B_i^{(m+1)} - B_i^{(m)}$ and $b_i^{(m+1)} - b_i^{(m)}$are less than a small ϵ, the algorithm terminates and the performance statistics are calculated.

2.4 Numerical results

In order to investigate the performance of the jump-over network, the approximation is evaluated for a large test set and compared with the results of a discrete-event simulation of the real queueing network. This section is split into two parts. Section 2.4.1 and Section 2.4.2 discuss the accuracy of the approximation for a large test set for single-segment and for multi-segment routing systems.

Both the jump-over network and the discrete-event simulation were implemented in Java. For each case, the simulation model was run 10 times for 1,000,000 seconds, preceded by 10,000 seconds of initialization for the system to become stable, which guaranteed that the 95% confidence interval width of the system throughput time is less than 1% of the mean value for all the cases. All experiments are run on Core i7 with 2.4 GHz and 8 GB of RAM.

2.4.1 Single-segment models

In this section the performance of the approximation is investigated for the single-segment routing model. A test set was generated for which the parameters are listed in Table 2.2. The number of zones in the system M varied between 1 and 8 and the number of totes N between 10 and 80. Furthermore, it is first assumed that every zone and every conveyor node are identical to the other nodes of the same type and that all possible tote classes are released into the system with the same probability. This ensures that the work-load of all zones in the system is balanced. In the test set, the mean conveyor times, $\mu_i^{-1}, i \in \mathcal{C}$ are varied between 20 and 60 seconds and

the mean zone times, $\mu_i^{-1}, i \in \mathcal{Z}$ between 10 and 30 seconds. The number of order pickers $d_i, i \in \mathcal{Z}$ and buffer places $q_i, i \in \mathcal{Z}$ varied between 1 and 3, and 0 and 1, respectively. In total, this leads to $8 \times 8 \times 5 \times 5 \times 3 \times 2 = 9,600$ different cases.

In addition, the effect of work-load imbalance among order pickers and zones is tested. Imbalance can be introduced by either changing the release probabilities or the parameters of a zone, e.g., the mean zone times. The latter approach was chosen for the imbalanced test set. In this test set, the mean conveyor times $\mu_i^{-1}, i \in \mathcal{C}$ are equal to 30 seconds and both the number of order pickers $d_i, i \in \mathcal{Z}$ and buffer places $q_i, i \in \mathcal{Z}$ are equal to 1. Four different scenarios were created for the mean zone times. In the first scenario, the mean zone times are equal, whereas in the other three scenarios they increase by either 2, 5, or 10 seconds per subsequent zone. This leads to an additional $7 \times 8 \times 4 = 224$ cases. The run time per case for the analytical model is less than a second, whereas the simulation takes at most 30 seconds in case of the larger systems.

Table 2.2: Parameters of the single-segment routing model test set.

(a) Balanced test set (9,600 cases)

Parameter	Value
Nr. of zones, M	$1, 2, 3, 4, 5, 6, 7, 8$
Nr. of totes, N	$10, 20, \ldots, 80$
Mean conveyor times, $\mu_i^{-1}, i \in \mathcal{C}$	$20, 30, 40, 50, 60$
Mean zone times, $\mu_i^{-1}, i \in \mathcal{Z}$	$10, 15, 20, 25, 30$
Nr. of order pickers, $d_i, i \in \mathcal{Z}$	$1, 2, 3$
Buffer size of a zone, $q_i, i \in \mathcal{Z}$	$0, 1$

(b) Imbalanced test set (224 cases)

Parameter	Value
Nr. of zones, M	$2, 3, 4, 5, 6, 7, 8$
Nr. of totes, N	$10, 20, \ldots, 80$
Mean zone times, $\mu_i^{-1}, i \in \mathcal{Z}$	1) $10, 10, 10, \ldots$
	2) $10, 12, 14, \ldots$
	3) $10, 15, 20, \ldots$
	4) $10, 20, 30, \ldots$

The results of the balanced test set are summarized in Table 2.3, Table 2.4, Table 2.5, and Table 2.6. Each table lists the average of the relative error between the approximation and the simulation for the system throughput (thr) in hour^{-1}, the average number of circulations a tote makes in the system before moving to the exit, and the mean of the sum of throughput times of the zones; $(x - Sim)/Sim \times 100\%$. Each table also gives the percentage of cases that fall in three different error-ranges. From the tables it can be concluded that the approximation produces very accurate results for the three performance statistics. The overall average error in the system throughput is 0.54%, it is 0.65% for the mean number of circulations, and 0.30%

for the average mean throughput times of the zones. Almost all of these errors fall between $0 - 1\%$, with only a few larger than 5%.

Table 2.3 and Table 2.4 show that the largest errors occur when the system has three or four zones and when the number of totes in the system is high. An explanation is that the blocking probabilities increase when the number of zones M decreases or if the number of totes in the system N increases. Moreover, if blocking is prevalent, a higher M means the approximation needs to estimate more blocking probabilities, which creates more room for error. Eventually, M is high enough that blocking is almost fully absent for any N. The approximation becomes exact since the network will behave precisely as the original queueing network where totes are never blocked.

Table 2.5 and Table 2.6 show that the largest errors occur with low mean conveyor times and high mean zone times. Here the product-form assumption that each node can be analyzed in isolation does not describe the real behavior sufficient. For example, if a tote is blocked by a zone, it can circulate through the whole system and eventually encounter the zone still working on the same tote. This will create dependencies between successive visits to the nodes which are not captured by the approximation. However, this situation is very unlikely in practice. The total recirculation time is usually much higher than the time a tote will spend in a zone.

Table 2.7 presents the results of the imbalanced test set. The errors of the three performance statistics are slightly larger than those of the balanced test set. In particular, the errors increase when there is more imbalance between the zones. Totes that need to visit the slowest zone now spend more time in the system since the probability of being blocked is higher, which increases errors, as seen in the previous tables. Still, on average the errors for the three statistics are well below 1%.

2.4.2 Multi-segment models

For the multi-segment routing model, a new test set is created, the parameters of which are listed in Table 2.8. In all test cases, the number of zones M equals 18, but the number of zones per segment m_k can vary between 3, 6, and 9. Furthermore, it is assumed that within every segment the zones and conveyor nodes are identical, i.e., $\mu_i^{-1} = 30, i \in \mathcal{C}\backslash\mathcal{C}^0$, $\mu_i^{-1} = 15, i \in \mathcal{Z}$, and $q_i = d_i = 1, i \in \mathcal{Z}$. The release probabilities $\psi_{\boldsymbol{r}}$ are assumed to be the same for all $\boldsymbol{r}$, and the service means of all entrances are

Table 2.3: Results balanced test set with a varying number of zones M.

M	Error (%) in system thr.				Error (%) in nr. of circulations				Error (%) in thr. times zones			
	Avg	$0-1$	$1-5$	>5	Avg	$0-1$	$1-5$	>5	Avg	$0-1$	$1-5$	>5
1	0.08	100.0	0.0	0.0	0.08	100.0	0.0	0.0	0.09	100.0	0.0	0.0
2	0.67	70.0	29.8	0.2	0.78	69.0	29.8	1.3	0.44	83.9	16.1	0.0
3	0.78	68.2	31.7	0.2	0.94	67.2	30.3	2.5	0.44	86.2	13.8	0.0
4	0.73	71.9	27.8	0.3	0.90	71.3	25.9	2.8	0.38	90.3	9.8	0.0
5	0.64	76.6	23.3	0.2	0.80	75.0	22.4	2.6	0.32	93.2	6.8	0.0
6	0.54	80.4	19.5	0.1	0.68	78.6	18.9	2.5	0.28	94.9	5.1	0.0
7	0.45	83.8	16.2	0.0	0.57	82.4	15.8	1.8	0.25	96.9	3.1	0.0
8	0.38	86.7	13.3	0.0	0.48	85.2	13.5	1.3	0.23	97.7	2.3	0.0

Table 2.4: Results balanced test set with a varying number of totes N.

N	Error (%) in system thr.				Error (%) in nr. of circulations				Error (%) in thr. times zones			
	Avg	$0-1$	$1-5$	>5	Avg	$0-1$	$1-5$	>5	Avg	$0-1$	$1-5$	>5
10	0.24	95.0	4.8	0.2	0.29	93.1	5.9	1.0	0.21	99.8	0.3	0.0
20	0.40	86.8	12.9	0.3	0.53	85.5	12.3	2.2	0.21	97.9	2.1	0.0
30	0.52	81.7	18.1	0.3	0.67	80.0	17.5	2.5	0.24	95.6	4.4	0.0
40	0.59	77.6	22.3	0.2	0.74	76.7	20.8	2.6	0.28	93.0	7.0	0.0
50	0.62	75.3	24.8	0.0	0.77	74.2	23.7	2.2	0.32	91.8	8.3	0.0
60	0.64	74.1	25.9	0.0	0.76	73.0	25.1	1.9	0.36	89.6	10.4	0.0
70	0.64	73.6	26.4	0.0	0.75	72.9	25.7	1.4	0.38	88.8	11.3	0.0
80	0.64	73.6	26.4	0.0	0.72	73.3	25.7	1.0	0.41	86.7	13.3	0.0

Table 2.5: Results balanced test set with varying mean conveyor times $\mu_i^{-1}, i \in \mathcal{C}$.

μ_i^{-1}	Error (%) in system thr.				Error (%) in nr. of circulations				Error (%) in thr. times zones			
	Avg	$0-1$	$1-5$	>5	Avg	$0-1$	$1-5$	>5	Avg	$0-1$	$1-5$	>5
20	0.91	66.5	33.0	0.5	1.21	65.3	28.1	6.7	0.47	85.4	14.6	0.0
30	0.64	74.1	25.9	0.0	0.79	72.7	25.2	2.2	0.33	91.0	9.0	0.0
40	0.47	81.4	18.6	0.0	0.55	80.3	19.4	0.4	0.27	94.4	5.6	0.0
50	0.36	86.3	13.7	0.0	0.41	85.5	14.5	0.0	0.23	96.3	3.8	0.0
60	0.29	90.2	9.8	0.0	0.31	89.2	10.8	0.0	0.21	97.3	2.7	0.0

Table 2.6: Results balanced test set with varying mean zone times $\mu_i^{-1}, i \in \mathcal{Z}$.

μ_i^{-1}	Error (%) in system thr.				Error (%) in nr. of circulations				Error (%) in thr. times zones			
	Avg	$0-1$	$1-5$	>5	Avg	$0-1$	$1-5$	>5	Avg	$0-1$	$1-5$	>5
10	0.24	93.0	7.0	0.0	0.23	92.7	7.3	0.0	0.17	98.2	1.8	0.0
15	0.40	84.9	15.1	0.0	0.46	83.5	16.1	0.4	0.23	95.5	4.5	0.0
20	0.55	78.4	21.6	0.0	0.67	76.8	21.7	1.5	0.30	92.7	7.3	0.0
25	0.68	72.9	27.0	0.2	0.86	71.8	25.3	2.9	0.37	90.1	9.9	0.0
30	0.80	69.3	30.3	0.4	1.05	68.1	27.3	4.5	0.43	87.8	12.2	0.0

Table 2.7: Results imbalanced test set with varying mean zone times $\mu_i^{-1}, i \in \mathcal{Z}$.

μ_i^{-1}	Error (%) in system thr.				Error (%) in nr. of circulations				Error (%) in thr. times zones			
	Avg	$0-1$	$1-5$	>5	Avg	$0-1$	$1-5$	>5	Avg	$0-1$	$1-5$	>5
10,10,10,...	0.20	100.0	0.0	0.0	0.20	100.0	0.0	0.0	0.15	100.0	0.0	0.0
10,12,14,...	0.23	100.0	0.0	0.0	0.24	100.0	0.0	0.0	0.16	100.0	0.0	0.0
10,15,20,...	0.35	98.2	1.8	0.0	0.36	94.6	5.4	0.0	0.21	100.0	0.0	0.0
10,20,30,...	0.40	89.3	10.7	0.0	0.45	85.7	14.3	0.0	0.32	100.0	0.0	0.0

equal to $\mu_i^{-1} = 5, i \in \mathcal{E}$. The number of totes in the system is varied between 10 and 80 and the capacities of the segments N^k between 10 and 40 totes as long as $N \geq N^k$. Finally, the mean main conveyor times, $\mu_i^{-1}, i \in \mathcal{C}^0$ varied between 10 and 60. This leads to 1,320 different test cases. The run time per case for the analytical model is around 10 seconds, whereas the simulation takes at most 1 minute for the largest systems with $N = 80$.

The results of the multi-segment test set are summarized in Table 2.9 and Table 2.10. The overall average error in the system throughput is 0.21%, for the mean number of circulations on the main conveyor 0.93%, and for the average throughput times of the zones 0.24%. The tables show that the errors are the largest in cases with a low number of segments and a fast main conveyor. In these cases, totes are more likely to be blocked by a segment such that they need to recirculate on the main conveyor multiple times. As seen in the previous results, the errors increase when there is more blocking in the system. When varying the segment capacities N^k similar results can be seen.

Table 2.8: Parameters of the multi-segment routing model test set (1,320 cases).

Parameter	Value
Nr. of segments, K	$2, 3, 4, 5, 6$
Nr. of totes, N	$10, 20, \ldots, 80$
Mean main conveyor times, $\mu_i^{-1}, i \in \mathcal{C}^0$	$10, 20, \ldots, 60$
Segment capacity, N^k, $k = 1, \ldots, K$	$10, 15, \ldots, 40$

Parameter	Value
Nr. of zones per segment, m_k	1) $9, 9$
	2) $6, 6, 6$
	3) $3, 6, 3, 6$
	4) $3, 3, 6, 3, 3$
	5) $3, 3, 3, 3, 3, 3$

Table 2.9: Results multi-segment routing test set with a varying number of zones per segment m_k.

m_k	Error (%) in system thr.				Error (%) in nr. of circulations				Error (%) in thr. times zones			
	Avg	$0-1$	$1-5$	>5	Avg	$0-1$	$1-5$	>5	Avg	$0-1$	$1-5$	>5
9,9	0.21	99.6	0.4	0.0	2.27	66.3	20.1	13.6	0.23	100.0	0.0	0.0
6,6,6	0.25	100.0	0.0	0.0	1.23	72.0	21.6	6.4	0.23	100.0	0.0	0.0
6,3,6,3	0.19	100.0	0.0	0.0	0.43	84.8	15.2	0.0	0.24	100.0	0.0	0.0
3,3,6,3,3	0.16	100.0	0.0	0.0	0.28	89.8	10.2	0.0	0.25	100.0	0.0	0.0
3,3,3,3,3,3	0.23	99.2	0.8	0.0	0.43	86.7	12.5	0.8	0.23	100.0	0.0	0.0

Table 2.10: Results multi-segment routing test set with varying mean conveyor times $\mu_i^{-1}, i \in \mathcal{C}^0$.

μ_i^{-1}	Error (%) in system thr.				Error (%) in nr. of circulations				Error (%) in thr. times zones			
	Avg	$0-1$	$1-5$	>5	Avg	$0-1$	$1-5$	>5	Avg	$0-1$	$1-5$	>5
10	0.28	99.1	0.9	0.0	2.23	66.8	22.3	10.9	0.23	100.0	0.0	0.0
20	0.24	100.0	0.0	0.0	1.15	75.5	18.6	5.9	0.23	100.0	0.0	0.0
30	0.21	100.0	0.0	0.0	0.77	77.3	18.2	4.5	0.24	100.0	0.0	0.0
40	0.19	99.5	0.5	0.0	0.58	83.6	14.5	1.8	0.24	100.0	0.0	0.0
50	0.18	100.0	0.0	0.0	0.45	86.8	11.8	1.4	0.24	100.0	0.0	0.0
60	0.16	100.0	0.0	0.0	0.37	89.5	10.0	0.5	0.24	100.0	0.0	0.0

Comparing the systems with each other, in case the main conveyors are slow ($\mu_i^{-1} \geq 50$ seconds) and the number of totes in the system is low ($N \leq 20$), the systems with $m^k = 9$, $k = 1, 2$ obtain the highest throughput, because of less conveying times. On the other side, if the segment capacity $N^k \leq 20$ is low, then $m^k = 3$, $k = 1, \ldots, 6$ has the highest throughput, because the probability that a tote is blocked by segment is lower and less recirculation of totes is required.

2.5 Conclusion and further research

In this chapter, we developed an analytical model for sequential zone picking systems with either single-segment, or multi-segment routing. The model provides a valuable tool for rapid design of complex zone picking systems in order to meet specific performance levels and it can be used to study and reduce the sources of blocking and congestion. We developed a queueing model to estimate the performance of the system. Because an exact analysis of this queueing model is not feasible, we approximate the blocking behavior with the jump-over protocol which yield product-form results, and we use MVA, and an aggregation technique to obtain rapidly very accurate estimates of the key performance statistics of a zone picking system. Comparison of the approximation results to simulation for a wide range of parameters showed that the mean relative error for statistics as the system throughput and the mean number of circulations in the system is less than 1%.

The model lends itself to several modifications and extensions left for future research. The approximation can be used to evaluate and compare operational policies such as order batching and order splitting on system performance, like in Yu & De Koster (2008). In addition, a general optimization framework can be formulated for allocating products to zones in order to maximize, for example, the system throughput. Also, when the zone picking system is too heavily loaded, congestion occurs when two conveyor streams merge, e.g. totes flowing out of a zone with totes on the conveyor. Totes should wait for a sufficiently large space on the conveyor before merging, which decreases the performance of the system due to long waiting times. It is also possible to incorporate this type of blocking-after-service to set achievable targets, e.g. the system throughput time, and predict when the system is overloaded. Another relevant extension is the situation where order pickers can help each other when the

workload in one zone is high or leave when there is little work such that one order picker becomes responsible for picking products at multiple zones. Furthermore, the model may provide a starting point in order to approximate higher moments or the distribution of performance statistics such as the zone, segment, and, system throughput time and as a building block for a semi-open queueing analysis approach (closed for totes and open for orders in the system) when studying a zone picking in an operational setting.

Our approach to model and analyze queueing networks with finite capacities and the dynamic block-and-recirculate protocol has shown to give very accurate approximations. There are many potential applications beyond zone picking systems where our method might be applied successfully, e.g., end-of-aisle picking systems, AGV transportation systems, and vehicle-based compact storage systems.

3 An accurate model for conveyor merges in zone picking systems

3.1 Introduction

Conveyor systems are a critical component of many order picking and sorting systems, responsible for moving products from one location to another. One of the most important functions of conveyor systems is to consolidate multiple flows of products into one single flow (a *merge* operation). These merges are often potential points of congestion which can lead to blocking and increased order throughput times. Obviously, the performance of the merges strongly influences the performance of the overall system. In *sequential zone picking*, a very popular order picking method in practice, conveyor merges occur frequently and need to be considered when determining the maximum throughput capability of the system.

Zone picking is a picker-to-parts order picking method, which divides the order picking area in work zones, each operated by one or multiple order pickers (Petersen, 2002; Gu et al., 2010). The major advantages of zone picking systems are the high-throughput ability, scalability, and flexibility in handling various order volumes and product sizes, with a varying number of order pickers (Van der Gaast et al., 2012). These systems are often applied in e-commerce warehouses handling customer orders with a large number of order lines and with a large number of different products kept in stock (Park, 2012).

In a sequential zone picking system, orders are transported via a tote on a conveyor to the zones. When an order reaches a zone, it is diverted from the conveyor into the zone whenever required products are stored within the zone. Orders that do not require products from the zone remain on the conveyor and are transported

to the next zone in line. After picking, the orders in the zone need to be merged back onto the conveyor. Under heavy loads, congestion and blocking can occur at these conveyor merges due to limited free space on the conveyor. This congestion leads to reduced throughput and causes unpredictable throughput times. As a direct consequence, orders cannot be shipped on time, which leads to delayed customer deliveries and loss in revenue. Especially e-commerce warehouse companies deal with very strict delivery lead times since the customer's demand fast delivery, often within 24 hours.

Besides the delivery lead times, *throughput* is one of the key performance indicators in zone picking systems. Throughput, measured as the number of completed orders or order lines per period of time, is used to judge whether the order picking system is capable of meeting a certain customer demand. It is also used to determine the cut-off time so that orders are guaranteed to be shipped during the next delivery cycle. However, estimating the throughput of a zone picking system, or any conveyor system with one or multiple merges is very complicated due to congestion at the merges and proliferation of congestion over the merges. This holds even stronger in the presence of job variability, variability of the performance of components, and of human operators.

The objective of this chapter is to quantify the impact of merge operations on the throughput of zone picking systems in order to determine their maximum throughput capability. Zone picking systems with merges can be analyzed by simulation models and by testing various scenarios. Simulation allows for very accurate modeling, but it is time-consuming to build and evaluate each scenario or lay-out design, especially when the system is highly utilized and blocking occurs frequently. Also, the accuracy of the simulation depends strongly on the quality of the calibration data (Osorio & Bierlaire, 2009). Another approach to analyze zone picking systems are queueing networks. Queueing networks are in general much faster, easier to modify, and less data expensive in estimating the performance of a zone picking system. They can be used as evaluation tools in the initial design phase to help designers quickly evaluate many design alternatives and to narrow down the available design space (Gu et al., 2010). They can also be used to optimize the system in later design (or control) phases in terms of order release rules and workload allocation.

The system is therefore modeled as a closed queueing network that describes the conveyor, the pick zones, and the merge locations. We assume that the arrival rate of new orders is higher than the rate at which totes are completed, such that each completed order can be immediately replaced by a new one. This is a valid assumption for zone picking system design, which aims at finding the maximum throughput capacity of a system. In addition, when the order arrival rate is low, zone picking systems are not used (or inappropriate), because they tend to be expensive, and cheaper and more effective order picking methods exist for these situations (e.g. sort-while-pick parallel picking, see De Koster et al. (2007)). Due to finite buffers, blocking, recirculation and merging, the resulting queueing model does not have a product-form stationary queue-length distribution which makes exact analysis practically infeasible. Therefore, we approximate the performance of the model using an aggregation technique (Chandy et al., 1975) and matrix-geometric methods (Latouche & Ramaswami, 1999). We show that the approximation model produces very accurate estimates of the maximum throughput capability of a zone picking system with merge operations when compared with simulation. The model is, in particular, well suited to evaluate many design alternatives, in terms of number of zones, zone input and output buffer lengths, and maximum number of totes in the systems. Our results show that throughput drops dramatically when congestion and blocking at the merges increase, and that if the number of totes in the system increases, it becomes more beneficial to increase the size of the output buffer rather than the input buffer of the zones.

The organization of this chapter is as follows. In Section 3.2, we discuss zone picking systems. An overview of existing models for both zone picking and conveyor systems with recirculating loops and merge operations is given in Section 3.3. The queueing model is presented in Section 3.4. In Section 3.5 we explain our approximation method and verify its performance in Section 3.6 via computational experiments for a range of parameters. In the final section, we conclude and suggest some extensions of the model.

3.2 Zone picking systems

In zone picking systems, the order picking area is zoned so that each order picker is responsible for picking products only from his or her zone. Zone picking systems can be categorized in either *parallel* or *sequential* zone picking (De Koster et al., 2007).

In a parallel zone picking system, multiple pickers, in multiple zones, can work simultaneously on one order (or a batch of orders). The picked products are sent downstream to a designated consolidation area where they are combined into orders. In sequential zone picking (also called pick-and-pass systems), an order is assigned to an order tote or order carton that travels on the conveyor and is passed sequentially to the next zone where order lines that need to be added to the order may (or may not) be stored. At a zone, each picker picks for only one tote at a time and after each pick the tote is merged back on the main conveyor. The advantage of sequential zone picking is that order integrity is maintained and no sorting and product consolidation is required (Petersen, 2000). These advantages make sequential zone picking systems highly popular in practice, especially in e-commerce warehouses. In this chapter, we only consider sequential zone picking (hereafter zone picking).

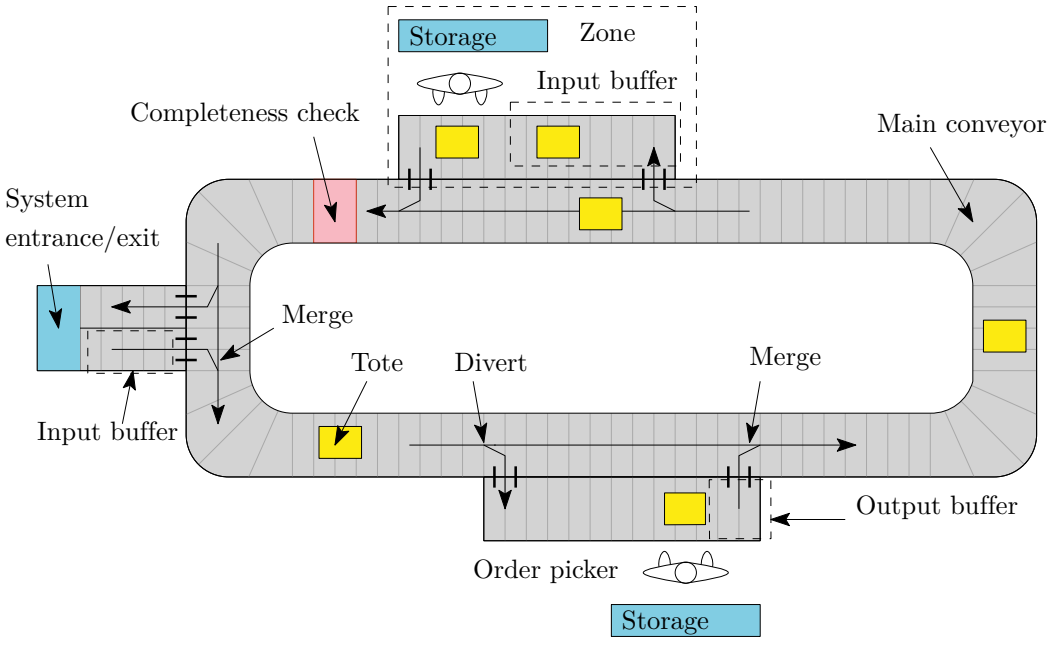


Figure 3.1: A zone picking system with two zones.

In Figure 3.1 is a schematic representation of a zone picking system, where the picking area is divided into two zones. These zones are connected by conveyors enabling automatic transportation of customer orders through the system. A customer order is released into the system at the system entrance as an order tote, which contains a list of products to be picked and their locations within the picking area. The tote only enters the system when the workload control mechanism allows it to do so (Park, 2012). This mechanism sets an upper bound on the number of totes in the system and only releases a new tote when a picked tote leaves the system. After release, the tote travels on the main conveyor to the required zones, i.e. where a product that still needs to be picked for the tote's order is stored and enters the input buffer when reaching one of these zones. The order picker then starts picking the required products that are stored within the zone. After all picks are completed, the order picker places the tote in the output buffer, where the tote waits until there is enough space on the main conveyor so that it can travel to a next zone. When the tote has visited all the required zones, it leaves the system at the exit and, if available, a new order tote is immediately released into the system.

Many different configurations of zone picking systems exist. These variants include, e.g. workstation type, pick-face design, buffer lengths, storage system lay-out, and conveyor configuration. Especially the conveyor configuration is of great importance, since it affects how and when totes arrive at the zones. In most zone picking systems, a tote can skip a zone if it does not need to pick up order lines from the zone. Also, combined with a closed-loop conveyor, totes can skip a zone if the zone's input buffer is fully occupied. The tote can then return to this zone after visiting other zones or after recirculating on the conveyor (a completeness check at the end of the conveyor ensures that the tote does not leave the system before visiting all the required zones). The advantage of this *dynamic block-and-recirculate* protocol is that it prevents congestion on the main conveyor and balances workload across the various zones. For a detailed analysis of this protocol in zone picking, the reader is referred to Van der Gaast et al. (2012).

The maximum throughput capability of a zone picking system is largely determined by the performance of the merges. After entering the system and after each zone, totes merge on the main conveyor in order to move to their next location. At a merge location, totes that are already on the main conveyor have absolute priority (the

main conveyor moves continuously without possibilities for accumulation). Therefore, in order to allow a tote to leave the output buffer, its predecessors from the same buffer should have left, and a sufficiently large space on the main conveyor must be available to prevent collisions. Under low utilization, the time required for a sufficiently large space on the conveyor to show up is negligible when determining the performance of the system. However, many systems are highly utilized during peak hours, e.g. in e-commerce environments. In such a case, this space can become very scarce leading to long merge times and a loss in overall system performance. In addition, the output buffer can become full and stop the order picker or the entrance station from continuing to work on the next tote in line. The order picker or entrance station can resume its work only if there is at least one empty place in the output buffer. Finally, some zone picking systems do not have an output buffer; in these cases, the order picker or entrance station must always wait until the processed tote has entered the main conveyor before starting to work on the next tote in line.

3.3 Existing literature

Literature on zone picking systems is still very limited; however, it has gained popularity in recent years. Gray et al. (1992) used a hierarchical approach to evaluate economic tradeoffs of equipment selection, storage assignment, number of zones, picker routing, and order batching when designing a zone picking system. De Koster (1994) modeled a zone picking system without recirculation as a Jackson queueing network, which allows for fast early-stage estimation of design alternatives in terms of order throughput times and average work-in-process. Malmborg (1996) developed a model to study the tradeoffs in space requirements and retrieval costs with dedicated and randomized storage in a zone picking system. Jane (2000) studies workload balancing in zone picking systems and proposed several heuristic methods to adjust the number of zones so that each picker remains balanced. Petersen (2002) performed a simulation study to investigate the shape of the zone and showed that the size or storage capacity of a zone, the number of items on the pick list, and the storage policy have a significant effect on average walking distances. Jewkes et al. (2004) studied the assignment of products to zones and the location of the picker home base in order to minimize the expected order cycle time. The authors developed a

dynamic programming algorithm for fixed product locations that optimally determines the product and server locations. Yu & De Koster (2008) analyzed zone picking systems without recirculation and presented an approximation method based on a $G/G/m$ queueing network. Eisenstein (2008) analyzed product assignments and depot locations in a zone picking system where single or dual depots are allowed along the pick line. Pan & Wu (2009) used Markov chain analysis and proposed three heuristics that optimally allocate items to a single picking zone, a picking line with unequal-sized zones, and a picking line with equal-sized zones. Melacini et al. (2010) modeled a zone picking system as a network of queues. In order to estimate performance statistics, such as the utilization, throughput rate of a zone, and the mean and standard deviation of the throughput time of the totes, they used Whitt's queueing network analyzer (Whitt, 1982). Van der Gaast et al. (2012) studied a single/multi-segment zone picking system with the block-and-recirculation protocol. They showed that the system can be very accurately approximated by a related product-form queueing network with the jump-over protocol.

In contrast, the analysis of conveyor systems has received much more attention. The models from the literature can be categorized as either deterministic or stochastic. Deterministic conveyors models were studied by for example, Kwo (1958); Muth (1977); Bastani & Elsayed (1986); and Bastani (1988) who investigated feasibility conditions, such as loading/unloading rates and conveyor lengths, for various simple closed-loop conveyors. However, these models fail to capture the effects that random fluctuations in either the input and/or output can have on the design and performance of a conveyor system.

For stochastic models, Disney (1962) studied the behavior of a conveyor system as a multichannel queueing system with ordered entry. This model served as the basis for many other studies about conveyor systems (see Muth & White (1979) for a survey of these models). Sonderman (1982) studied a conveyor system with a single loading and unloading station where loads can recirculate. The author uses Whitt's queueing network analyzer (Whitt, 1982) to approximate the output process at the unloading station and to study the effect of recirculation. Sonderman & Pourbabai (1987) extended the model of Sonderman (1982) by allowing random access on the conveyor.

Coffman, Jr et al. (1988) studied a conveyor system for flexible manufacturing systems and investigated the effect of the distance of input and output points of workstations at which items leave and rejoin the conveyor. In order to study the performance of the system, the conveyor queue was modeled by a Markov process. Schmidt & Jackman (2000) modeled a recirculating conveyor as an open network of queues. The system consists of one loading station, one unloading station, with two servers performing the same service on loads entering the system, and a loop conveyor divided into segments. Zijm et al. (2000) analyzed an automated kit transportation system and studied a number of key elements of the system separately and subsequently combined the results of this analysis in an Approximate Mean Value Analysis (AMVA) algorithm. Bozer & Hsieh (2005) and Hsieh & Bozer (2005) modeled a conveyor as a unidirectional closed loop consisting of discrete spaces or windows of equal size, which hold at most one load or unit. They considered different machines that are located around the conveyor with a pair of unloading and loading stations per machine, modeling them as output and input queues.

All these papers analyze only particular aspects of a zone picking system, such as recirculation and merging conveyor flows. In the next section, we integrate these various aspects into a single model.

3.4 Queueing model for zone picking systems

Figure 3.2 shows the model for zone picking systems with merges, for the case of two zones. Van der Gaast et al. (2012) studied a similar model, but they did not model the merges. The zone picking system is modeled as a closed queueing network with one entrance/exit, W zones, $W+1$ merges, and $W+1$ nodes that describe the conveyor between a merge location and a zone or the entrance/exit. The nodes are labeled in the following manner: the system entrance/exit is denoted as e, $\mathcal{Z} = \{z_1, \ldots, z_W\}$ denotes the set of zones, $\mathcal{M} = \{m_1, \ldots, m_{W+1}\}$ denotes the set of merges, and $\mathcal{C} = \{c_1, \ldots, c_{W+1}\}$ is the set of conveyors in the network. Finally, set $\mathcal{S} = \{e\} \cup \mathcal{C} \cup \mathcal{M} \cup \mathcal{Z}$ is the union of all the nodes in the network. The following assumptions are adopted for the network:

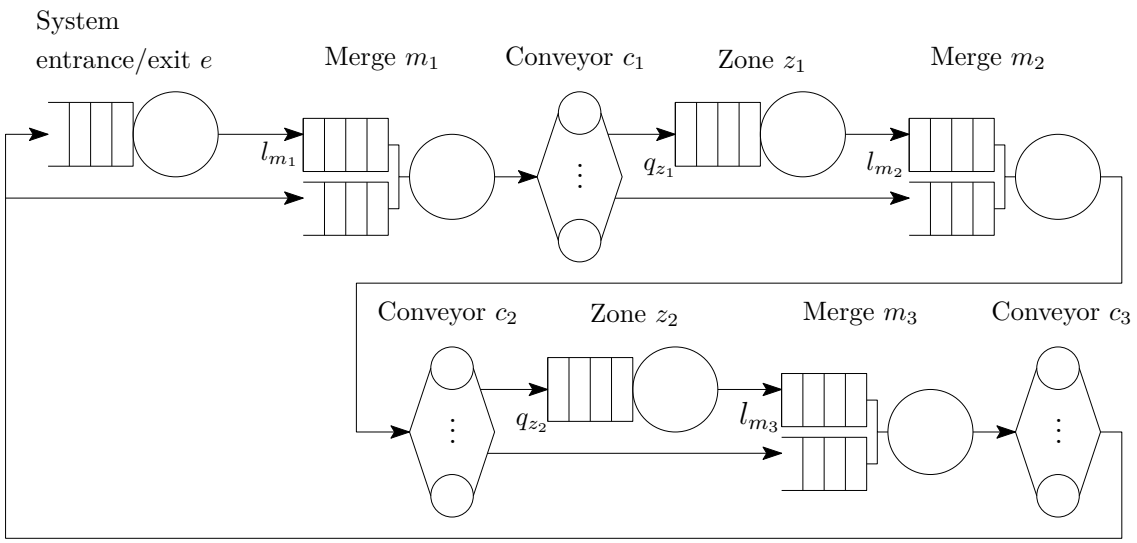


Figure 3.2: The corresponding queueing network with system entrance/exit station e, conveyors $\mathcal{C} = \{c_1, c_2, c_3\}$, merges $\mathcal{M} = \{m_1, m_2, m_3\}$ and, zones $\mathcal{Z} = \{z_1, z_2\}$.

- There is an infinite supply of totes at the entrance of the system. This means that a leaving tote can always be replaced immediately by a new tote. Each tote has a class $\boldsymbol{r} \subseteq \mathcal{Z}$, e.g., $\boldsymbol{r} = \{z_2, z_3\}$ means that the tote has to visit the second and third zone.
- The total number of totes in the system or to be released is constant, N. As long as the total number of totes in the zones, merges, and conveyor nodes is less than N, new totes are released one-by-one at the system entrance where the inter-release times are exponentially distributed with rate μ_e. The assumption of a constant number of totes in the system is not restrictive, since our objective is to study the maximum throughput capability of the system. In addition, zone picking systems, especially in e-commerce warehouses, are typically heavily utilized. This means that sufficiently many customer orders needing to be picked at any point in time are available.
- The conveyor nodes are assumed to be infinite-server nodes with a deterministic delay $1/\mu_i$, $i \in \mathcal{C}$.
- Each zone has one order picker. The order picking time is assumed to be exponentially distributed with rate μ_i, $i \in \mathcal{Z}$, that captures both variations in the pick time per order line and variations in the number of order lines to be picked. The assumption of the exponential distribution can be relaxed to a

phase-type distribution at the cost of a more complex state space. The same holds for the number of order pickers per zone.

- When the order picker is busy, incoming totes at his or her station are stored in a finite input buffer of size $q_i\,(\geq 0)$, $i \in \mathcal{Z}$. Incoming totes are blocked when the total number of totes in the input buffer equals q_i.
- The merge nodes are assumed to be single server preemptive non-identical repeat priority stations where totes on the main conveyor (*high priority*) have absolute priority over the totes flowing out of the zones/entrance (*low priority*). Whenever a high priority tote enters the merge, it will preempt any low priority tote currently in service. After the high priority tote has left and no other tote of high priority is currently at the merge, the low priority tote will repeat its service. The time required to pass the merge, either for low or high priority totes, is assumed to be exponentially distributed with rate μ_i^H for high priority totes and μ_i^L for low priority totes, $i \in \mathcal{M}$. Similar as for the zones, the assumption of the exponential distribution can be relaxed.
- Each merge has a limited capacity of size $l_i\,(\geq 0)$, $i \in \mathcal{M}$ to store low priority totes. This corresponds with the limited output buffer found after the zones/entrance. When there are l_i low priority totes waiting at the merge node, no incoming low priority tote will be accepted by the merge node and the low priority tote has to wait at its current node, subsequently blocking the order picker/entrance station from starting to work on the next tote in line. We can distinguish two distinct cases for the unblocking procedure. In case $l_i \geq 1$, the tote leaves its current node and unblocks the order picker/entrance station, only when there is at least one open position for low priority totes at the merge node. Whenever there is no output buffer ($l_i = 0$), the order picker/entrance station only unblocks after the current tote has passed the merge.

Let $\mathbb{S}\,(N)$ be the state space of the network, i.e. the set of states $\boldsymbol{x} = (\boldsymbol{x}_i;\, i \in \mathcal{S})$ for which the number of totes in the system equals $\sum_{i\in\mathcal{S}} n_i = N$. The state of node $i \in \mathcal{C}$ is $\boldsymbol{x}_i = (\boldsymbol{r}_{i1}, \ldots, \boldsymbol{r}_{in_i})$, with $\boldsymbol{r}_{i1}$ as the class of the first tote in the node, and $\boldsymbol{r}_{in_i}$ as the class of the last tote in the node. The state of node $i \in \{e\} \cup \mathcal{Z}$ is $\boldsymbol{x}_i = (\boldsymbol{r}_{i1}, \ldots, \boldsymbol{r}_{in_i}; y_i)$ where $y_i = 1$ if the order picker/entrance is blocked since a tote has finished service but cannot leave the zone since the merge is occupied and $y_i = 0$ otherwise. The state of merge node $i \in \mathcal{M}$, is defined as $\boldsymbol{x}_i = \left(\boldsymbol{r}^H_{i1}, \ldots, \boldsymbol{r}^H_{in_i^H}; \boldsymbol{r}^L_{i1}, \ldots, \boldsymbol{r}^L_{in_i^L}\right)$ with

$n_i = n_i^H + n_i^L$ as the number of totes with high and low priority respectively. The number of low priority totes in each merge should not exceed the capacity of the merge's output buffer l_i; $n_i^L \leq l_i$, $i \in \mathcal{M}$. Finally, the number of totes in each zone satisfies the capacity constraint $n_i \leq q_i + 1$, $i \in \mathcal{Z}$, which implies that a tote cannot enter the zone if the input buffer is full and the order picker is occupied/blocked.

A new tote of class $\boldsymbol{r} \subseteq \mathcal{Z}$ is released at the system entrance with given probability $\psi_{\boldsymbol{r}}$. These release probabilities correspond to a known order profile that can be obtained using, e.g., historical order data or forecasts. After release, a tote of class $\boldsymbol{r}$ moves from the system entrance to the first merge node m_1 with low priority. In general after merging, a tote travels to conveyor node c_i. After transport at conveyor node c_i, the tote will either enter the input buffer of zone z_i if $z_i \in \boldsymbol{r}$ and its input buffer is not full, or move to the next merge m_{i+1} with high priority. In case the tote needs to enter and the buffer is full, the tote skips the zone and also moves to the next merge m_{i+1}, while it keeps the same class and again with high priority. If the buffer is not full, the tote enters the buffer of the zone and, after possibly waiting some time in the buffer, the order picker picks the required order lines. After all picks are completed, the tote will enter the merge node with low priority and changes its class to $\boldsymbol{s} = \boldsymbol{r} \backslash \{z_i\}$. When the tote successfully passes the merge, it is routed to conveyor node c_{i+1}. After visiting the last conveyor node c_{W+1}, all the totes with $\boldsymbol{r} \neq \emptyset$ are routed to the first merge node m_1 with high priority; the other totes with $\boldsymbol{r} = \emptyset$ move to the exit and are immediately replaced by a new tote which is waiting for release at the entrance. Denote by $p_{i\boldsymbol{r},j\boldsymbol{s}}(\boldsymbol{x})$ the state dependent routing probability that a tote of class $\boldsymbol{r}$ is routed from node i to node j and enters as a class $\boldsymbol{s}$ tote given that the network is in state $\boldsymbol{x}$. Then, the routing probabilities are:

$$p_{e\emptyset,m_1\boldsymbol{r}}(\boldsymbol{x}) = \psi_{\boldsymbol{r}}, \tag{3.1}$$

$$p_{m_i\boldsymbol{r},c_i\boldsymbol{r}}(\boldsymbol{x}) = 1, \quad i = 1,\ldots,W, \tag{3.2}$$

$$p_{c_i\boldsymbol{r},z_i\boldsymbol{r}}(\boldsymbol{x}) = 1, \quad i = 1,\ldots,W,\ z_i \in \boldsymbol{r} \text{ and } n_{z_i} < q_{z_i} + 1, \tag{3.3}$$

$$p_{c_i\boldsymbol{r},m_{i+1}\boldsymbol{r}}(\boldsymbol{x}) = 1, \quad i = 1,\ldots,W,\ z_i \notin \boldsymbol{r} \text{ or } n_{z_i} = q_{z_i} + 1, \tag{3.4}$$

$$p_{z_i\boldsymbol{r},m_{i+1}\boldsymbol{s}}(\boldsymbol{x}) = 1, \quad i = 1,\ldots,W,\ \boldsymbol{s} = \boldsymbol{r} \backslash \{z_i\}, \tag{3.5}$$

$$p_{c_{W+1}\emptyset,e\emptyset}(\boldsymbol{x}) = 1, \tag{3.6}$$

$$p_{c_{W+1}\boldsymbol{r},m_1\boldsymbol{r}}(\boldsymbol{x}) = 1, \quad \boldsymbol{r} \neq \emptyset, \tag{3.7}$$

where every other probability is equal to 0.

Exact analytic methods to analyze queueing networks are only known for a very limited set of models that satisfy certain conditions. The majority of these models have a *product-form* stationary distribution (Jackson, 1963; Gordon & Newell, 1967; Baskett et al., 1975). For these models, it can be proven that the stationary distribution of the network can be expressed as a product of factors describing the state of each node. Based on this independence assumption, exact efficient analysis algorithms such as the convolution algorithm (Buzen, 1973) and the mean-value analysis (MVA) (Reiser & Lavenberg, 1980) can be applied to analyze the models.

However, the previously described queueing network does not have a product-form stationary distribution, because of the priorities at the merge nodes (Bryant et al., 1984), and because of the dynamic block-and-recirculate protocol (Van der Gaast et al., 2012). Also, direct analysis of the resulting underlying Markov chain is not feasible due to state-space explosion which prevents analysis of the Markov chain within reasonable time and storage. Usually, non-product-form queueing networks are studied using approximation analysis. An overview of many general techniques is presented in Bolch et al. (2006).

Van der Gaast et al. (2012) show that the queueing network without merges can be very accurately approximated by a related product-form queueing network with the *jump-over* protocol. The idea of the approximation is to replace the state dependent routing with state independent routing in such a way that the flows in the new network match the flows of the original network. This is done by introducing a Bernoulli process that randomly determines for every tote that intends to visit z_i, $i \in \mathcal{Z}$ and independently of whether the tote actually visited z_i or not, whether the tote should return to z_i. The probability b_i of the Bernoulli process that a tote should return to z_i is chosen in such a way that it corresponds with the probability that a tote is blocked by a zone in the original network. Naturally, the blocking probabilities are not known in advance, but they are estimated iteratively after an initial guess from the approximation.

The queueing network with merges and the dynamic block-and-recirculate protocol can be transformed into a queueing network with jump-over blocking as follows. First, routing probabilities (3.1)-(3.7) become state independent. This means after service at c_i, each tote with $z_i \in \boldsymbol{r}$ is routed to z_i regardless of whether the buffer of

the zone is full (3.8)-(3.9). The tote will enter the buffer if it is not full; otherwise the tote instantaneously skips the node. Then for each class $\boldsymbol{r}$ tote, independent of whether the tote visited or skipped z_i (because of a full buffer), $p_{z_i\boldsymbol{r},m_{i+1}\boldsymbol{r}} = b_{z_i}$ and $p_{z_i\boldsymbol{r},m_{i+1}\boldsymbol{s}} = 1 - b_{z_i}$, $i = 1, \ldots, M$, where $\boldsymbol{s} = \boldsymbol{r} \backslash \{z_i\}$. This means that a tote of class $\boldsymbol{r}$ is tagged as *skipped* z_i and routed to the next merge node m_{i+1} with the same class with probability b_{z_i}, and otherwise, the tote is tagged as *visited* z_i with probability $1 - b_{z_i}$ and the class of the tote changes to $\boldsymbol{s} = \boldsymbol{r} \backslash \{z_i\}$. Summarizing, the routing probabilities (3.3)-(3.5) are replaced by

$$p_{c_i\boldsymbol{r},z_i\boldsymbol{r}} = 1, \qquad i = 1, \ldots, W,\ z_i \in \boldsymbol{r}, \tag{3.8}$$

$$p_{c_i\boldsymbol{r},m_{i+1}\boldsymbol{r}} = 1, \qquad i = 1, \ldots, W,\ z_i \notin \boldsymbol{r}, \tag{3.9}$$

$$p_{z_i\boldsymbol{r},m_{i+1}\boldsymbol{r}} = b_{z_i}, \qquad i = 1, \ldots, W, \tag{3.10}$$

$$p_{z_i\boldsymbol{r},m_{i+1}\boldsymbol{s}} = 1 - b_{z_i}, \quad i = 1, \ldots, W,\ \boldsymbol{s} = \boldsymbol{r} \backslash \{z_i\}\,. \tag{3.11}$$

Since the recirculation process is made independent of the state of the buffer, essentially the block-and-recirculate protocol is replaced by the *jump-over* blocking protocol (Van Dijk, 1988). Under this protocol, each tote of class $\boldsymbol{r}$ leaving z_i, either after service or skipping, continues to follow the same Markovian routing. The advantage of the jump-over blocking protocol, also known as "overtake full stations, skipping, and blocking and rerouting", is that closed-form analytic results for single-class queueing networks are available in the literature (Pittel, 1979; Schassberger, 1984; Van Dijk, 1988; Economou & Fakinos, 1998).

However, this jump-over network still has no product-form due to the finite capacity priority queues. In this chapter, we develop a decomposition-based approximation by studying each merge and zone location in isolation. Our method progressively aggregates parts of the network and replaces the aggregated subnetwork by a flow equivalent single node. The approximation directly solves the global balance equations of the underlying Markov chain of the subnetworks for its steady-state distribution.

3.5 Approximate aggregation method

3.5.1 Aggregation technique

The aggregation technique was introduced by Chandy et al. (1975) to study the performance of product-form queueing networks (Baskett et al., 1975). The technique has been extended for more general multi-class queueing networks by Kritzinger et al. (1982), Walrand (1983), Hsiao & Lazar (1989), and Boucherie & van Dijk (1993). Based on Norton's theorem, the idea of the aggregation technique is to decompose the queueing network into subnetworks and to replace each subnetwork by a flow equivalent single server with load-dependent service rates. The rates of the flow equivalent server (FES) are obtained by studying the subnetwork in isolation, i.e. by short-circuiting all nodes that are not in the subnetwork. The service rate of the kth FES f_k when n totes are present is taken equal to $X^k(n)$ the throughput of the closed subnetwork with population n;

$$\mu_{f_k}(n) = X^k(n), \quad n = 1, \ldots, N. \tag{3.12}$$

The aggregation method is proven to be exact if the queueing network has a product-form stationary distribution (Chandy et al., 1975), and can be used as a basis to analyze non-product form queueing networks (Bolch et al., 2006).

Figure 3.3a presents the queueing network of Section 3.4. It is analyzed by the approximate aggregation method as shown in Figure 3.3b, where the nodes are partitioned into $W + 2$ subnetworks as follows;

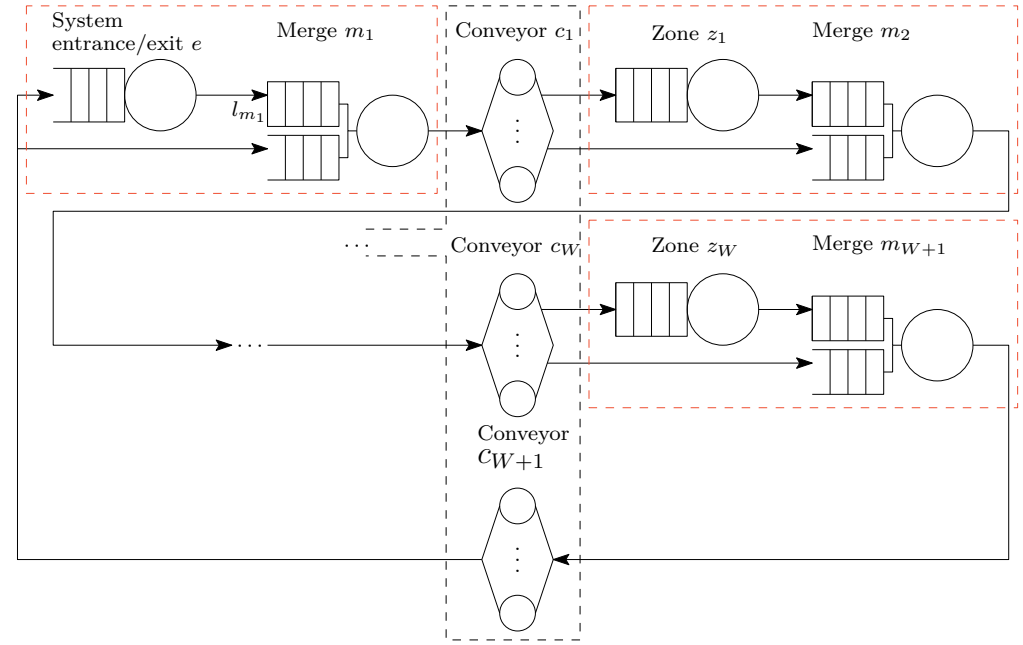
$$\mathcal{H}^0 = \mathcal{C}, \tag{3.13}$$

$$\mathcal{H}^1 = \{f_0\} \cup \{e\} \cup \{m_1\}, \tag{3.14}$$

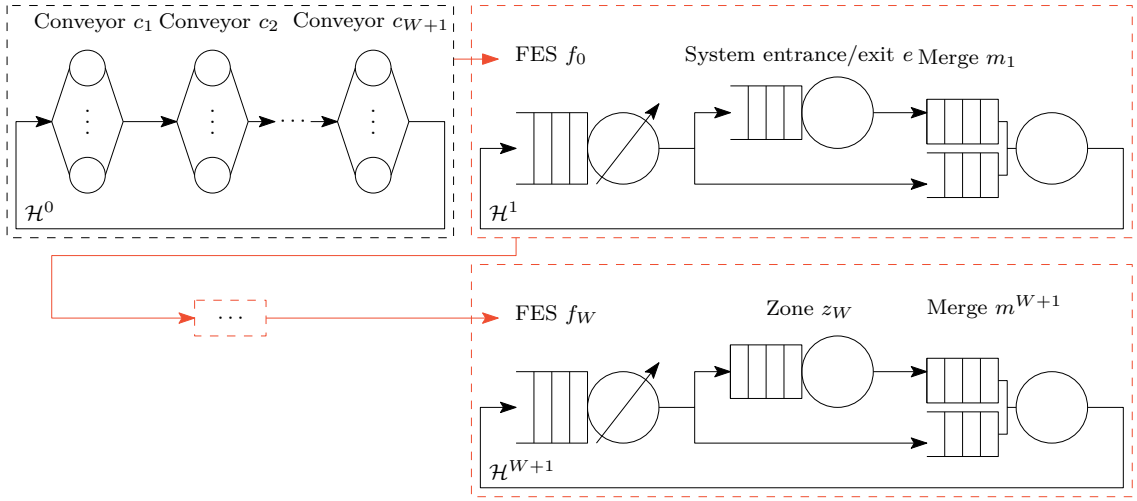
$$\mathcal{H}^{k+1} = \{f_k\} \cup \{z_k\} \cup \{m_{k+1}\}, \quad k = 1, \ldots W. \tag{3.15}$$

The first step of the approximation is to determine the chain visit ratios V_i that a tote visits node i;

$$V_i = \frac{\sum_{r \subseteq \mathcal{Z}} \lambda_{ir}}{\sum_{r \subseteq \mathcal{Z}} \lambda_{er}}, \quad i \in \mathcal{S}, \tag{3.16}$$



(a) Original queueing network



(b) Approximation steps

Figure 3.3: The approximate aggregation technique applied to a zone picking system with W zones, where each subnetwork is replaced by a flow equivalent server with load-dependent service rates.

where $\lambda_{i\boldsymbol{r}}$ is the class dependent visit ratio of a class $\boldsymbol{r}$ tote to node i satisfying the traffic equations (up to a multiplicative constant)

$$\lambda_{i\boldsymbol{r}} = \sum_{j \in \mathcal{S}} \sum_{\boldsymbol{s} \subseteq \mathcal{Z}} \lambda_{j\boldsymbol{s}} p_{j\boldsymbol{s},i\boldsymbol{r}}, \quad i \in \mathcal{S}, \boldsymbol{r} \subseteq \mathcal{Z}. \tag{3.17}$$

The next step is to study subnetwork $\mathcal{H}^0$ in isolation. Since subnetwork $\mathcal{H}^0$ only consists of conveyor nodes with deterministic service, the average throughput of the subnetwork with population n is simply given by;

$$X^0(n) = \sum_{i \in \mathcal{H}^0} \frac{n}{V_i/\mu_i}, \quad n = 1, \ldots, N. \tag{3.18}$$

The marginal queue-length probabilities $\pi_i^0(j|n)$, $n = 1, \ldots, N$, $i \in \mathcal{H}^0$, can be calculated in a similar fashion as a service center of Type-3 (infinite server with general distributed service times) in a BCMP network (Baskett et al., 1975).

Then for each subsequent subnetwork $\mathcal{H}^k$, the previous subnetwork is aggregated into FES f_{k-1} with service rates given by (3.12) and analyzed in isolation together with the nodes in $\mathcal{H}^k$. For each of these networks, we only need to know the visit ratio of totes that visit the entrance/zone or not, and the ratio of totes that skip the zone, which depends on b_{z_i} (see Section 3.3.2 of Van der Gaast et al. (2012)). This process is repeated until the last subnetwork $\mathcal{H}^{W+1}$ from which the overall performance statistics such as the throughput are obtained. The performance of the individual nodes can now be calculated by disaggregating the network using the marginal queue length probabilities obtained from each subnetwork (see Section 3.5.3).

Our approximation method differs from other aggregation heuristics, e.g. Marie (1979) and Neuse & Chandy (1982). These heuristics start by replacing each node that does not satisfy the product-form assumption by an equivalent product-form node. Then these heuristics solve the subnetwork iteratively and better estimates for the equivalent node are found. This is repeated until it resembles the original node up to a certain prespecified threshold. However, convergence might be slow and many iterations may be required, while the number of iterations of our approach is equal to the number of subnetworks, and the underlying Markov chain is solved only once.

Section 3.5.2 shows how the other subnetworks $\mathcal{H}^k$, $k = 1, \ldots, W+1$ can be studied. The full approximate method is presented in Section 3.5.3.

3.5.2 Solving subnetworks $k \geq 1$

In this section, we describe the analysis of subnetwork $\mathcal{H}^1$ till $\mathcal{H}^{W+1}$. Each of these subnetworks consists of a node with preemptive non-identical repeat priority; the merge, and thus cannot be analyzed using conventional product-form solution techniques (Bryant et al., 1984). Using aggregation, we reduce the size of the problem to a small system that we can efficiently model as a finite Markov process and directly solve the global balance equations of the underlying Markov chain for its steady-state. This allows us to calculate for a given $n = 1, \ldots, N$ the throughput $X^k(n)$ of subnetwork $\mathcal{H}^k$.

For each subnetwork $\mathcal{H}^k$, $k = 1, \ldots, W+1$ we define a Markov process with state space $\mathbb{W}^k(n)$ with states (i, j, l) and the number of totes in the subnetwork is n. The state variable i denotes the number of totes waiting at the input buffer or in service either in e or z_{k-1}, state variable j represents the number of totes with low priority at merge m_k and includes the totes that finished service in e or z_{k-1} but cannot enter the output buffer or cross the merge. Finally, state variable l denotes the number of totes with high priority at the merge. Note that the number of totes at the FES f_{k-1} for any state is implicitly given by $u = n - i - j - l$.

Let the transition rates from state (i, j, l) to state (i', j', l') be given by $q_{(i,j,l)(i',j',l')}$. For a tote leaving FES f_k, the rates for $\mathcal{H}^k$, $k > 1$ can be written as follows:

$$q_{(i,j,l)(i+1,j,l)} = V_{z_{k-1}} \mu_{f_{k-1}}(u), \qquad u < n,\ i + y_{z_{k-1}} < q_{z_{k-1}} + 1, \quad (3.19)$$

$$q_{(i,j,l)(i,j,l+1)} = \left[V_{m_k} - V_{z_{k-1}}\right] \mu_{f_{k-1}}(u), \qquad u < n,\ i + y_{z_{k-1}} < q_{z_{k-1}} + 1, \quad (3.20)$$

$$q_{(i,j,l)(i,j,l+1)} = V_{m_k} \mu_{f_{k-1}}(u), \qquad u < n,\ i + y_{z_{k-1}} = q_{z_{k-1}} + 1. \quad (3.21)$$

Transition rate (3.19) is the rate at which a tote from FES f_{k-1} enters the zone. A tote can only enter the input buffer if the number of totes at the zone is lower than the zone's maximum capacity $q_{z_{k-1}} + 1$. The number of totes currently in z_{k-1} is equal to i plus an additional tote $y_{z_{k-1}} = 1_{\{j = l_{m_k} + 1\}}$, where the indicator function $1_{\{.\}}$ is equal to one if there is a tote which just received service and is waiting to leave the

zone, but cannot since the output buffer is full ($j = l_{m_k} + 1$). Transition rate (3.20) denotes the rate at which a tote moves to merge m_k if the tote does not need to visit the zone. If the zone is blocked ($i + y_{z_{k-1}} = q_{z_{k-1}} + 1$), the totes that are supposed to go to the zone are directly transported to the merge (3.21) with high priority. The rates for $\mathcal{H}^1$ are defined similarly, except (3.21) is not defined since the input buffer of the entrance station is assumed to be infinite.

The rate at which a tote leaves the zone or the merge is given as follows:

$$q_{(i,j,l)(i-1,j+1,l)} = \mu_{z_{k-1}}, \quad i > 0,\ j < l_{m_k} + 1, \tag{3.22}$$

$$q_{(i,j,0)(i,j-1,0)} = \mu^L_{m_k}, \quad j > 0, \tag{3.23}$$

$$q_{(i,j,l)(i,j,l-1)} = \mu^H_{m_k}, \quad l > 0. \tag{3.24}$$

Transition rate (3.22) denotes the rate of a service completion of a tote at zone z_{k-1}, whereas (3.23) and (3.24) denote the rate of a service completion of a tote at merge m_k for low and high priority totes, respectively. A low priority tote can only complete its service when there is no high priority tote at the merge. Again, the rates for $\mathcal{H}^1$ are defined similarly.

Figure 3.4 shows the Markov chain of subnetwork $\mathcal{H}^k$ on state space $\mathbb{W}^k(n)$ where the number of output buffer places equals $l_{m_i} = 1$. From Figure 3.4, we can see that the Markov chain is irreducible and that it is possible to partition state space $\mathbb{W}^k(n)$ such that

$$\mathbb{W}^k(n) = \bigcup_{l=0}^{n} \mathbb{W}^k_l(n), \quad \mathbb{W}^k_l(n) = \left\{ w^k_{l,1}(n), \ldots, w^k_{l,s_l}(n) \right\},$$

$$\mathbb{W}^k_l(n) \bigcap \mathbb{W}^k_{l'}(n) = \emptyset \quad \text{for } l \neq l',$$

where partition $\mathbb{W}^k_l(n)$ consists of s_l states where the number of high priority totes at merge m_k is equal to l.

Then the states of $\mathbb{W}^k(n)$ can be arranged such that the generator matrix of (3.19)-(3.24) is of block-tridiagonal form, that

$$Q = \begin{pmatrix} Q_{00} & Q_{01} & & & & & \\ Q_{10} & Q_{11} & Q_{12} & & & & \\ & Q_{21} & Q_{22} & Q_{23} & & & \\ & & \ddots & \ddots & \ddots & & \\ & & & Q_{n-1,n-2} & Q_{n-1,n-1} & Q_{n-1,n} \\ & & & & Q_{n,n-1} & Q_{nn} \end{pmatrix}, \tag{3.25}$$

where $Q_{ij} \in \mathbb{R}^{s_i \times s_j}$. In this case, the Markov process is said to be a finite level-dependent quasi-birth-and-death (LDQBD) process, and, $l = 1, \ldots, n$, the subset $\mathbb{W}_l^k(n)$ of states is referred to as the process level with level number l (Latouche & Ramaswami, 1999). In Bright & Taylor (1995) an efficient procedure is given to compute the stationary distribution π of a LDQBD process. The procedure partitions stationary distribution $\pi = (\pi_0, \pi_1, \ldots, \pi_i, \ldots, \pi_n)$, with $\pi_i = \left(\pi^k(0,0,i), \pi^k(1,0,i), \pi^k(0,1,i), \ldots\right)$ and uses the fact that

$$\pi_{i+1} = \pi_i R_i, \quad i \geq 0, \tag{3.26}$$

with non-negative matrices $R_i \in \mathbb{R}^{s_i \times s_{i+1}}$ that depend on the level. The basic idea behind the procedure is to exploit the fact that these matrices satisfy the infinite recurrence scheme

$$R_i = -Q_{i,i+1}\left(Q_{i+1,i+1} + R_{i+1}Q_{i+2,i+1}\right)^{-1}, \tag{3.27}$$

and where π_0 is a solution of the equation

$$\pi_0\left(Q_{00} + R_0 Q_{10}\right) = 0. \tag{3.28}$$

Bright & Taylor (1995) show that the inverse matrices in (3.27) exist, and that $Q_{00} + R_0 Q_{10}$ has the characteristics of a generator matrix of an irreducible CTMC with a finite state space. The solution of π_0 is unique up to a multiplicative constant and can be chosen to be non-negative.

The boundary condition at maximum level n is given by

$$R_{n-1} = -Q_{n-1,n}Q_{nn}^{-1}. \tag{3.29}$$

Thus, an exact algorithm for computing the stationary distribution of finite LDQBDs proceeds as follows. Determine R_{n-1}, then compute R_i for $i = n-2, \ldots, 0$ using (3.27). Determine π_0 by choosing a nontrivial solution of (3.28), and use (3.26) for computing $\pi_1, \pi_2, \ldots, \pi_n$. Finally, normalize π.

After obtaining the marginal distribution of the Markov chain with transition matrix (3.25) it is now possible to calculate the marginal distribution of the nodes in subnetwork $\mathcal{H}^k$. Let $\pi_i^k(m|n)$ be the marginal distribution of node i in subnetwork $\mathcal{H}^k$ where there are m totes in the node and the population size is n. The marginal distribution for the three nodes in the subnetwork is given as follows:

$$\pi_{f_{k-1}}^k(m|n) = \sum_{w \in \mathbb{W}^k(n):\, m=n-i-j-l} \pi^k(i,j,l), \tag{3.30}$$

$$\pi_{z_{k-1}}^k(m|n) = \sum_{w \in \mathbb{W}^k(n):\, m=i+y_{z_{k-1}}} \pi^k(i,j,l), \tag{3.31}$$

$$\pi_{m_k}^k(m|n) = \sum_{w \in \mathbb{W}^k(n):\, m=l+j-y_{z_{k-1}}} \pi^k(i,j,l). \tag{3.32}$$

Given the marginal queue-length probabilities, the average throughput of the subnetwork with population n is given by;

$$X^k(n) = \sum_{j=1}^{n} \pi_{f_k}^k(j|n)\, \mu_{f_k}(j), \quad n = 1, \ldots, N, \tag{3.33}$$

which is used as input to analyze subnetwork $\mathcal{H}^{k+1}$.

After analyzing the last subnetwork, the marginal queue-length probability for each node in the queueing network can be obtained by a disaggregation step using the marginal queue-length probability of the subnetworks. The marginal probability $\pi_i(j|l)$ of j totes present at node i given that the number of totes in the system is l

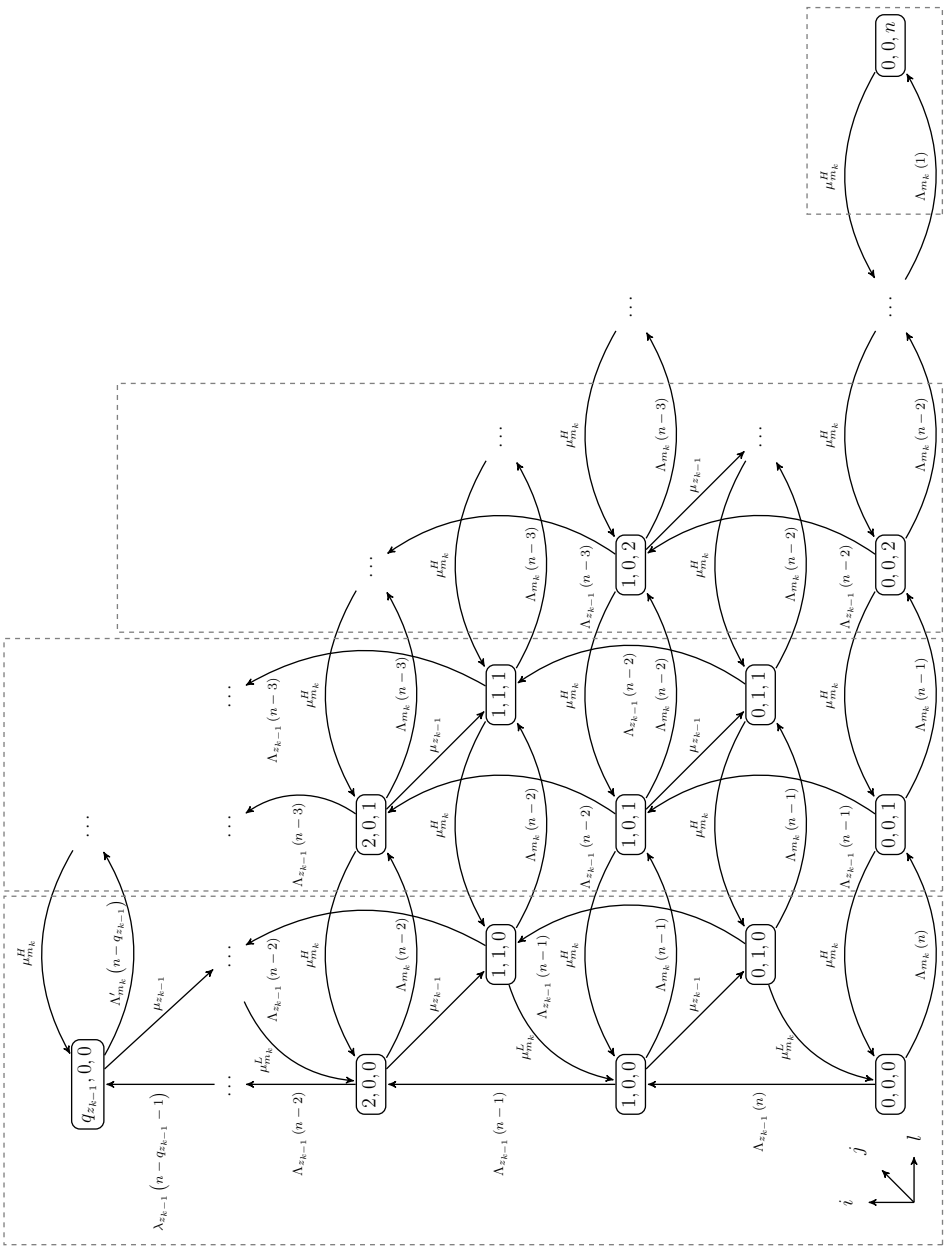


Figure 3.4: The Markov chain of subnetwork $\mathcal{H}^k$ on state space $\mathbb{W}^k(n)$, $k = 2, \ldots, W+1$, with $l_{m_i} = 1$ and where $\Lambda_{z_{k-1}}(n)$, $\Lambda_{m_k}(n)$, and $\Lambda'_{m_k}(n)$ are shorthand for (3.19), (3.20), and (3.21) respectively.

can be obtained as follows:

$$\pi_i(j|l) = \begin{cases} \sum_{m_k=0}^{N} \pi_i^k(j|m_k) \prod_{p=k+1}^{W} \left[\sum_{m_p=0}^{N} \pi_{f_p}^p(m_{p-1}|m_p) \right] \pi_{f_W}^{W+1}(m_W|l), & i \in \mathcal{H}^k, \\ \pi_i^{W+1}(j|l), & i \in \mathcal{H}^{W+1}. \end{cases} \tag{3.34}$$

Using the marginal queue-length probabilities performance statistics, such as the system throughput, node utilization, queue lengths can now easily be calculated.

3.5.3 Algorithm

In this section, we summarize the approximation procedure for analyzing the queueing network of Section 3.4. As shown in Section 3.5.1 and Section 3.5.2, we analyzed the queueing network by progressively aggregating parts of the network. However, in the first step of the algorithm, the visit ratios are calculated using the jump-over approximation and the unknown blocking probabilities of the zones b_i, $i \in \mathcal{Z}$, (see Section 3.4). In order to analyze the queueing network, we use a modified version of the algorithm presented in Van der Gaast et al. (2012). Similar as in the original algorithm, the blocking probabilities b_i, $i \in \mathcal{Z}$ are initialized by 0 and are subsequently updated until all the differences between the current and the previous blocking probability are smaller than a small value ϵ. In each iteration of the current version of the algorithm, the blocking probabilities are obtained by analyzing the subnetworks $\mathcal{H}^k$, $k = 0, \ldots, W+1$.

The approximation procedure can now be summarized as follows:

Step 1: Analyze the subnetwork $\mathcal{H}^0$ for different population sizes $n = 1, 2, \ldots, N$. For each n, obtain the marginal queue length probabilities $\pi_i^0(m|n)$, $i \in \mathcal{H}^0$ and throughput $X^0(n)$ (3.18).

Step 2: For $k = 1, \ldots, W+1$. Construct FES f_{k-1} using (3.12) and analyze $\mathcal{H}^k$ for different population size $n = 1, 2, \ldots, N$. Obtain the marginal queue length probabilities $\pi_i^k(m|n)$ and throughput $X^k(n)$ (3.33).

Step 3: The throughput rate of the system is given by $X(N) = X^{W+1}(N)$ and the blocking probabilities are $b_i = \pi_i(q_i + 1|N-1)$, $i \in \mathcal{Z}$ (3.34).

Step 4: Go back to Step 1 with the new estimates for blocking probabilities b_i, continue until convergence of all the blocking probabilities.

3.6 Numerical results

In this section, we compare the results of our approximation method with a discrete-event simulation of the real queueing network. We test the performance of the approximation method for a zone picking system without recirculation in Section 3.6.1 and test it with recirculation in Section 3.6.2. In Section 3.6.3, we analyze whether the order in which the subnetworks are analyzed in the approximation method has a significant effect on the performance statistics. Finally, in Section 3.6.4, we study the effect of additional places in either the input or output buffer of a zone on the performance of the system.

For each run, the simulation model was run 10 times for 1,000,000 seconds, preceded by 10,000 seconds of initialization for the system to become stable, which guaranteed that the 95% confidence interval width of the average throughput is less than 1% of the mean value for all the runs. In the algorithm $\epsilon = 10^{-3}$, and convergence usually occurs within a few iterations.

3.6.1 Zone picking system without recirculation

In order to study the performance and accuracy of the algorithm of Section 3.5.3, we start by considering a zone picking system with 2, 4, or 6 zones without recirculation. The iterative algorithm of Van der Gaast et al. (2012) does not need to be used in case of no recirculation, since $b_i = 0$, $i \in \mathcal{Z}$. In a system with W zones, a tote can visit a total of 2^W possible combinations of zones. We assume that each combination of zones (a class) has the same probability of being released into the system, except the empty set, e.g., $\psi_{\emptyset} = 0$ and $\psi_{\boldsymbol{r}} = 1/\left(2^W - 1\right)$. Furthermore, we assume that each zone, merge, and conveyor is identical to a node of the same type. The time required to prepare a new tote to be launched into the system at the entrance station is equal to $\mu_e^{-1} = 5$ seconds. Each conveyor node requires a fixed deterministic time to cross of $\mu_i^{-1} = 60$ seconds, $i \in \mathcal{C}$, whereas the time required to pass a merge node

is equal to $\left(\mu_i^L\right)^{-1} = \left(\mu_i^H\right)^{-1} = 3$ seconds, $i \in \mathcal{M}$. The time to pick products for a tote at a zone is $\mu_i^{-1} = 30$ seconds, $i \in \mathcal{Z}$. The number of order pickers in each zone is equal to 1 and the input buffer sizes of each zone is respectively $q_i = \infty$, $i \in \mathcal{Z}$, which means that an incoming tote is always accepted by the buffer of the zone. Finally, it is assumed that there is no output buffer after a zone and the entrance ($l_i = 0,\ i \in \mathcal{M}$). The order picker or the entrance station can only start to work on the next tote in line when the current tote has crossed the merge.

Table 3.1: Results of the average throughput $X(N)$ per hour of the approximation model and simulation for a zone picking system with 2, 4, and 6 zones without recirculation.

	2 zones			4 zones			6 zones		
N	Approx	Simulation	Error	Approx	Simulation	Error	Approx	Simulation	Error
5	70.77	71.03(±0.10)	-0.37	45.39	45.39(±0.03)	-0.01	32.77	32.77(±0.03)	0.02
10	119.34	120.49(±0.32)	-0.95	86.20	86.23(±0.08)	-0.04	63.47	63.50(±0.07)	-0.05
15	141.93	142.91(±0.40)	-0.69	120.84	120.83(±0.15)	0.01	91.56	91.45(±0.07)	0.12
20	150.14	150.82(±0.40)	-0.45	148.03	148.19(±0.20)	-0.11	116.49	116.54(±0.10)	-0.04
25	153.70	153.89(±0.32)	-0.13	167.62	167.69(±0.42)	-0.04	137.84	137.86(±0.15)	-0.01
30	155.64	155.94(±0.22)	-0.19	180.88	181.07(±0.36)	-0.11	155.41	155.49(±0.29)	-0.05
35	156.87	157.12(±0.34)	-0.16	189.69	189.91(±0.40)	-0.12	169.36	169.24(±0.34)	0.07
40	157.71	157.99(±0.47)	-0.18	195.68	195.88(±0.30)	-0.10	180.18	180.13(±0.28)	0.02
45	158.33	158.66(±0.38)	-0.21	199.91	199.95(±0.52)	-0.02	188.47	188.67(±0.35)	-0.11
50	158.80	158.83(±0.43)	-0.02	203.02	202.86(±0.65)	0.08	194.84	194.96(±0.33)	-0.06
55	159.17	159.22(±0.35)	-0.03	205.39	205.60(±0.42)	-0.10	199.80	199.79(±0.39)	0.00
60	159.47	159.47(±0.42)	0.00	207.25	207.28(±0.33)	-0.02	203.72	203.76(±0.41)	-0.02

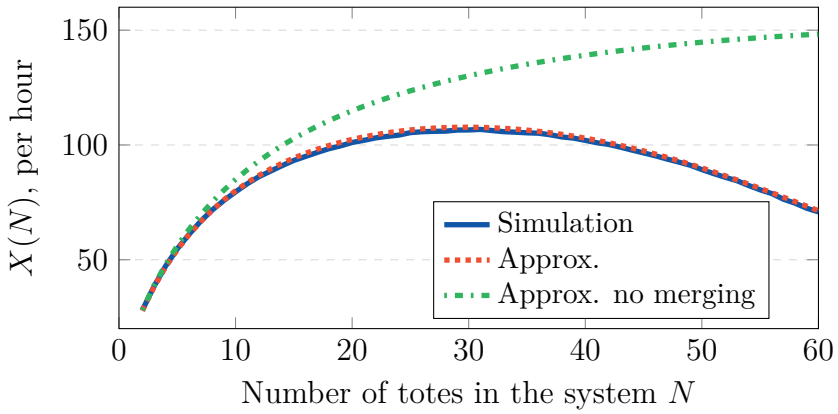
Table 3.1 presents the results of both the approximation method and the simulation method in terms of the average throughput $X(N)$ per hour for different numbers of totes in the system. The numbers in parentheses represent the standard deviations of the ten different runs of the simulation model and the column *error* shows the relative error between the approximation and the simulation; $(\text{Approx} - \text{Simulation})/\text{Simulation} \times 100\%$. The results show that the approximation method accurately predicts the average throughput of the system for each of the three configurations since all the errors are within 1%. Also, for any N, the average throughput will never decrease due to the assumption of infinite input buffers for the zones.

3.6.2 Zone picking system with recirculation

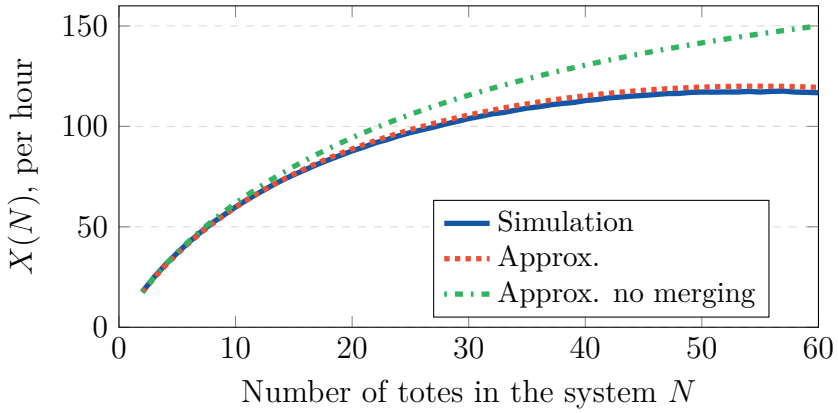
For the next comparison, we test the performance and accuracy of the algorithm of Section 3.5.3 for a zone picking system with recirculation. It is assumed that all the parameters are the same as in Section 3.6.2, with the exception that each zone now has a finite input buffer of size $q_i = 3$, $i \in \mathcal{Z}$.

Table 3.2 presents the results of the three configurations for both the approximation method and the simulation in terms of the average throughput $X(N)$ per hour. The approximation slightly overestimates the average throughput when reaching the maximum average throughput capability of the system. For example, in the configuration with two zones, the maximum throughput capability that can reached is ± 106 totes per hour if $N = 30$. Afterwards the average throughput starts to decrease because totes flowing out of a zone have to wait a long time until they can be merged on the main conveyor, and therefore prevent the order picker from continuing his/her work on the next tote in line and prevent the entrance station from releasing new totes. On the other hand, totes on the main conveyor recirculate until there is an open position in the input buffer of the zone. A similar effect can be seen in the configuration with 4 and 6 zones.

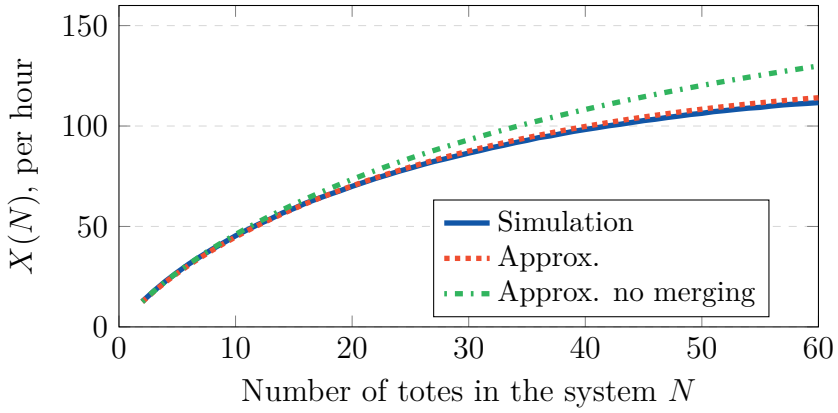
Figure 3.5 shows the same results for the average throughput, as well as, the results of the approximation method where the merge node is replaced by a single-server queueing node with an infinite buffer and with the same service distribution as the current merge node. This means that totes entering the merge are served on a first-come first-served basis and the order picker/entrance station is never blocked because of a full output buffer. The figure shows that for the approximation without the merges large errors are made when the number of totes N becomes large. In fact, the throughput will never decrease since any additional tote that enters the system can always enter the conveyor. Eventually, the throughput stabilizes at a point when the utilization of the order pickers equals 1. It can be concluded that modeling the merge operation in detail is of great importance because otherwise the maximum throughput capability of the system cannot be determined correctly. This can lead to the expectation that the system has a much higher throughput capability than what is possible in reality.



(a) 2 zones



(b) 4 zones



(c) 6 zones

Figure 3.5: Results of the average throughput $X(N)$ per hour of the approximation model and the simulation model with and without merges modeled for 2, 4, and 6 zones without recirculation.

Table 3.2: Results of the average throughput $X(N)$ per hour of the approximation model and simulation for a zone picking system with 2, 4, and 6 zones with recirculation.

	2 zones			4 zones			6 zones		
N	Approx	Simulation	Error	Approx	Simulation	Error	Approx	Simulation	Error
5	54.33	54.73(±0.15)	-0.74	36.60	36.99(±0.11)	-1.05	26.68	27.17(±0.10)	-1.80
10	80.04	79.56(±0.21)	0.60	59.67	59.85(±0.17)	-0.30	44.76	45.34(±0.11)	-1.29
15	94.31	93.26(±0.22)	1.13	76.28	75.95(±0.29)	0.45	58.81	58.95(±0.13)	-0.24
20	102.42	101.08(±0.30)	1.33	88.77	87.79(±0.24)	1.12	70.20	69.98(±0.12)	0.32
25	106.55	105.37(±0.34)	1.12	98.33	96.91(±0.20)	1.47	79.64	79.07(±0.17)	0.71
30	107.74	106.53(±0.32)	1.13	105.65	103.89(±0.23)	1.70	87.54	86.54(±0.15)	1.16
35	106.46	105.28(±0.30)	1.12	111.19	109.07(±0.17)	1.94	94.20	92.89(±0.21)	1.41
40	102.96	102.00(±0.25)	0.95	115.23	112.84(±0.18)	2.12	99.80	98.28(±0.26)	1.54
45	97.40	96.54(±0.26)	0.89	117.98	115.40(±0.22)	2.23	104.50	102.70(±0.18)	1.75
50	89.98	89.28(±0.29)	0.79	119.54	117.10(±0.32)	2.09	108.39	106.32(±0.28)	1.95
55	81.10	80.45(±0.34)	0.80	120.01	117.12(±0.32)	2.47	111.57	109.29(±0.22)	2.09
60	71.36	70.84(±0.14)	0.73	119.45	116.75(±0.26)	2.31	114.09	111.66(±0.17)	2.18

3.6.3 Order of solving the subnetworks

In the algorithm of Section 3.5.3, Section 3.6.1, and Section 3.6.2, it was assumed that the subnetworks were solved starting from the subnetwork with all the conveyors, then the subnetwork with the entrance/exit station until the subnetwork with the last zone. However, any other sequence of analyzing the subnetwork is also feasible, but based on Norton's theorem this will not lead to the exact same results because the queueing network does not have a product-form solution. In this section, we test how the sequence of solving the subnetworks has an effect on the average throughput of the system.

We ran experiments for a zone picking system with $W = 4$ zones with a varying number of totes in the system $N = 2, \ldots, 40$. The input/output buffer sizes of the zones are assumed to be equal, and varied between 1, 2, and 3 positions. All the other parameters are similar as in Section 3.6.1. We test two extreme cases; solve the subnetworks starting the conveyor subnetwork up to the subnetwork with the last zone (*forward*) and the reverse situation where the conveyor subnetwork is again analyzed first, but then the subnetwork with the last zone up till the subnetwork

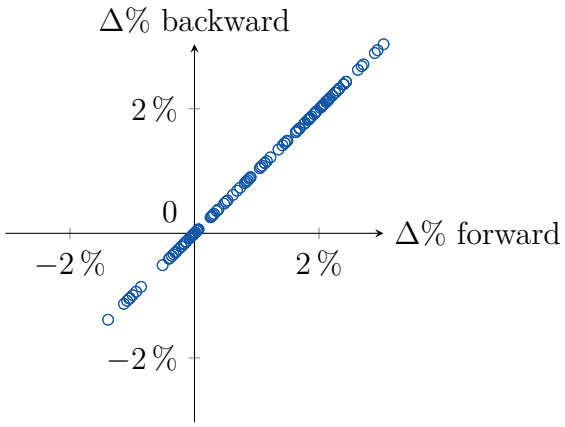


Figure 3.6: Relative errors in the average throughput $X(N)$ for solving the subnetworks in a forward or backward sequence for a zone picking system with $W = 4$ compared with the simulation model.

with the entrance/exit are analyzed (*backward*). In total, this gives 234 different cases (117/117 cases forward/backward).

Figure 3.6 shows the relative errors $\Delta\%$ for both the forward and the backward method with simulation for the average throughput $X(N)$. Both approximations obtain results that are all within 3% compared to the simulation results. If we compare the results from the forward and backward method, we can see that they almost fit perfectly on an increasing 45 degree line. This means that even when analyzing the subnetworks in a different sequence, almost exactly the same results are obtained for the average throughput. This also holds for other node specific statistics such as the utilization and average queue lengths.

3.6.4 Effect of buffer sizes of the zones

Finally, we test the effect of additional input and output positions in the buffer of the zones on the performance of a zone picking system with $W = 2$ zones. This is important for warehouse managers since additional conveyor space is expensive and requires space. Therefore, deciding on the optimal number of input and output positions is essential for system performance, as well as, for budget constraints. We assume that zones are located parallel to the main conveyor (as shown in Figure 3.1) and that adding each additional buffer position increases the time required to travel

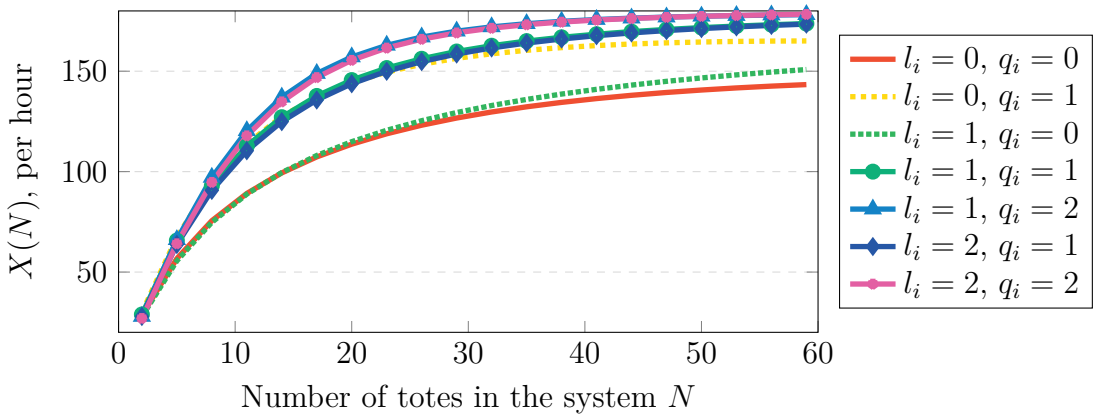
the conveyor by 3 seconds. The number of positions in the input/output buffer of the zones, q_i and l_i, is varied from 0 to 2 and are assumed to be the same across all zones in the system. All the other parameters are the same as in Section 3.6.2, except that the time required to pass a merge node $\left(\mu_i^L\right)^{-1} = \left(\mu_i^H\right)^{-1}$, $i \in \mathcal{M}$ varies between 1, 3, and 5 seconds.

Figure 3.7 presents the results for the average throughput for the three different merge times and the seven configurations of input/output buffer sizes. Only the results from the approximation model are shown, but the relative errors compared to simulation model are of the same magnitude as in Section 3.6.2. If we compare the three figures, we can see the maximum throughput capability increases as the merge times decrease when comparing the same input/output buffer positions. In addition, in Figure 3.7c the average throughput decreases much faster than in Figure 3.7a and Figure 3.7b.

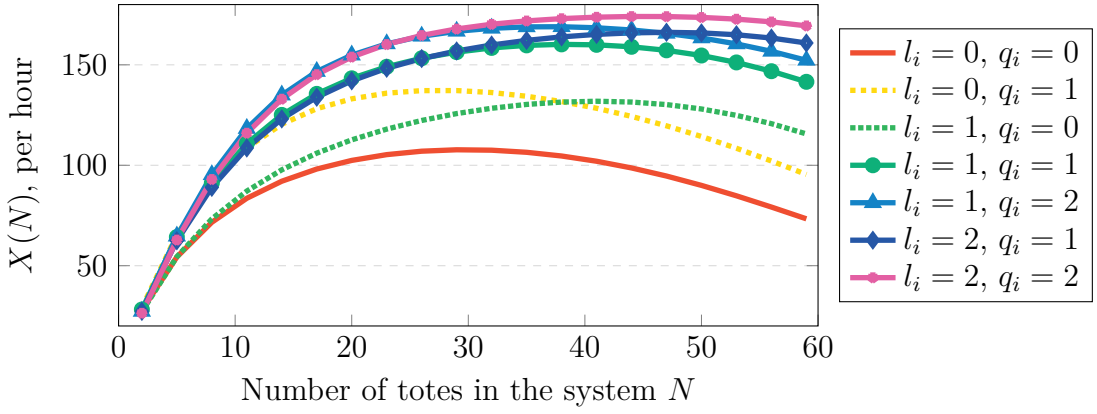
Also, in all three figures it can be seen that when N is low it is more beneficial to have an additional position in the input buffer, since it decreases the possibility that a tote is rejected from entering the buffer of the zone and has to recirculate on the main conveyor. However, when N increases, it becomes more attractive to have an additional output buffer position, since the average time required to merge and the fact the order picker is stopped more often becomes higher than the time it takes for a tote to recirculate once. Also, when the system is heavily utilized, the supply of new totes to the zones stalls due to congestion at the merges. As a consequence, increasing the length of the output buffer is more attractive than increasing the size of the input buffer. This can especially be seen when comparing $l_i = 0,\ q_i = 1$ with $l_i = 1,\ q_i = 0$, $i \in \mathcal{Z}$.

3.7 Conclusion and further research

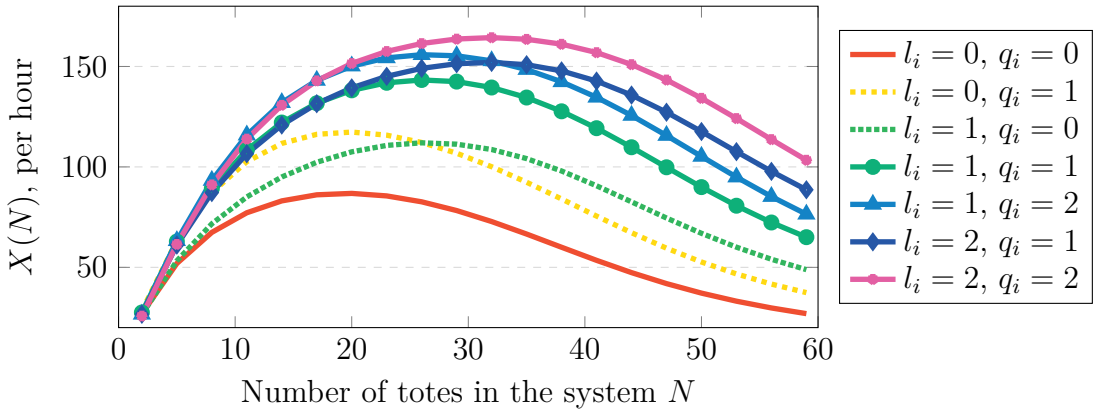
In this chapter, we developed an analytical model for studying the merge operation in zone picking systems. A decomposition-based approximation was used to study each merge and zone in isolation. Our method progressively aggregated parts of the network and replaced the aggregated subnetwork by a flow equivalent single node. The approximation directly solved the global balance equations of the underlying



(a) Merge time 1 second



(b) Merge time 3 seconds



(c) Merge time 5 seconds

Figure 3.7: Results of the average throughput $X\,(N)$ per hour of the approximation model and the simulation model for a zone picking system with 2 zones with different merging times and different input/output buffer sizes q_i and l_i.

Markov chain of the subnetworks for its steady-state distribution. The results show that for a wide range of parameters, the approximation was able to predict the maximum throughput capability of a zone picking system very accurately compared to simulation. The model is capable of predicting the loss in throughput given the level of congestion and blocking in the system, and can be used to allocate input and output buffer spaces in order to maximize the throughput capability of the system. A topic for further research would be to apply the approximation to study conveyor merges in other order picking and sorting systems.

4 Case study: product allocation methods in zone picking systems

4.1 Introduction

Due to the need for lower delivery costs, shorter customer response times, and higher customer service, the importance of warehousing in the supply chain has grown significantly. In order to increase warehouse performance and to achieve higher customer satisfaction, a warehouse should be adequately and robustly designed and controlled (De Koster et al., 2007). Out of all the activities carried out in a warehouse, *order picking*, the process of retrieving customer orders from their storage locations, is the most labor-intensive and costly of them all.

When designing a new order picking system and, afterwards, controlling the system, many decisions have to be taken on different strategic, tactical, and operational levels (Rouwenhorst et al., 2000). *Strategic decisions* include the level of automation, how to organize the material flow, as well as, which storage and material handling system to use. *Tactical decisions*, on the other hand, consist of the layout dimensions, and the product allocation. Finally, *operational decisions* include, e.g., which picking policies to use (discrete picking, batching, or zoning), the routing of order pickers (e.g. S-shaped, largest gap), and how and when to release a new customer order that needs to be picked. However due to the complex nature of an order picker system, these decisions are not always in line with each other and affect the performance of the order picking system in different ways. For example, the efficiency of the order picking process can be increased by large pick batches, but at the same time the responsiveness of the order picking process will decrease since customer orders have to wait longer before the next picking cycle starts. In addition, when in practice

a new order picking system is designed and implemented, feasibility is often of main concern rather than optimal performance. Moreover, the performance of an order picking system is largely dependent on the variability that occurs within the system, e.g. by varying customer order arrivals, by different order profiles, and by varying pick times. Therefore, analytical models that properly describe variability in order picking systems are extremely valuable in early design phases to test various design alternatives and decisions. In addition, in later phases they can help both designers and managers to create optimal design and control methods to improve the performance of the systems.

In this chapter, we investigate how the performance of a current order picking system of a large wholesaler supplying non-food items to supermarkets is affected by different product allocation methods. The system consists of a zone picking system with dynamic storage. *Zone picking* is one of the most popular picker-to-parts order picking methods for companies with a fairly large number of customer orders, picked from a large assortment of relatively small-sized products, and low to moderate number of picks per order (Van der Gaast et al., 2012). *Dynamic storage* refers to the situation that per zone only a fraction of the products is stored in the picking area (Yu & De Koster, 2010). An automated Storage and Retrieval (S/R) machine then retrieves products, when they are requested, from the bulk storage area which is located behind the zone. In this chapter, we compare, in particular, different product allocation methods and test their influence on system performance. We test a product allocation method that minimizes the number of segments a tote on average has to visit and a method that applies workload balancing between segments, i.e. zones connected by a recirculating conveyor, in order to reduce congestion and potential blocking in the system. In addition, we combine both methods into a single method that tries to minimize simultaneously the average number of segments a tote visits and applies workload balancing between segments. Using the analytical framework of Van der Gaast et al. (2012) and verifying with simulation, we find that product allocation methods can significantly influence the performance of a zone picking system. In particular, a product allocation that only applies workload balancing between segments reduces the system throughput on average by 8% compared to a product allocation that minimizes the number of segments a tote on average has to visit. On the other hand, blocking of zones and segments is significantly reduced

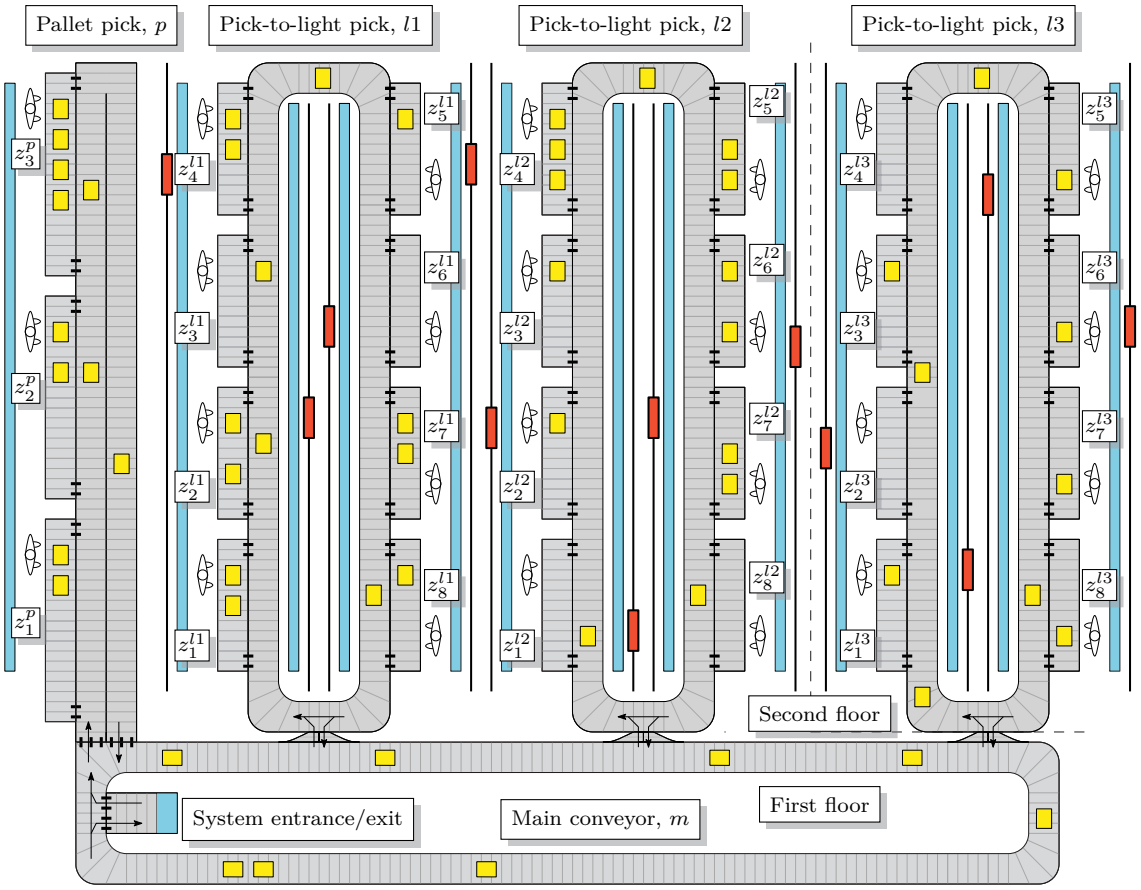
by a product allocation that applies workload balancing, e.g. segments are blocked 7.6% on average when a product allocation that minimizes the number of segments is used to 0.3% on average in the other case. As such, it provides a valuable tool for initial product allocation decisions for zone picking systems and the consequences strategic decisions can have on the overall performance of the system.

The organization of this chapter is as follows. In Section 4.2 a detailed description of the real-life system is given. In Section 4.3 several different methods of product allocation are presented in order to test their influence on system performance. We extensively analyze the results of the product allocation methods in Section 4.4 for the real-life system. Finally, in Section 4.5 we conclude and suggest some extensions of the model and further research topics.

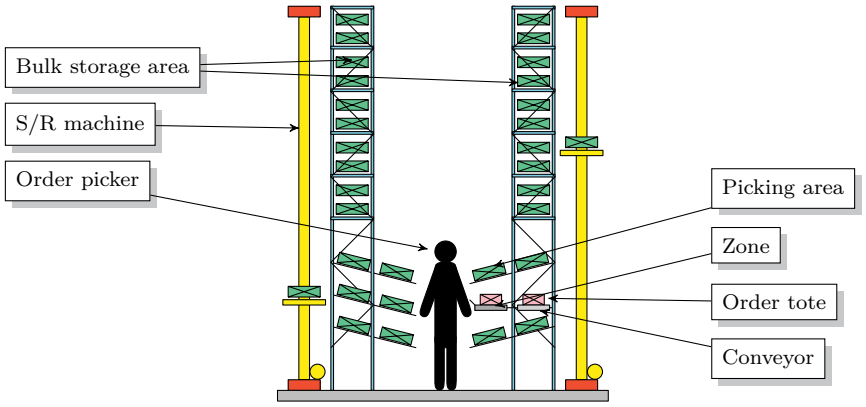
4.2 System description

The lay-out of the order picking area in the warehouse studied in this chapter is shown in Figure 4.1. The part of the warehouse dedicated to picking is divided into four interconnected closed-loop segments of which three are on the first floor and one is on the second floor; see Figure 4.1a. When a new customer tote is released into the system at the entrance station, the first segment it encounters is segment p, which consists of three zones, $\{z_1^p, z_2^p, z_3^p\}$. In these zones the order pickers pick products from pallets. The other three segments, $1, 2, 3$, each consists of eight zones, $\{z_1^1, \ldots, z_8^1\}$, $\{z_1^2, \ldots, z_8^2\}$, $\{z_1^3, \ldots, z_8^3\}$, where each zone uses pick-to-light and products are picked from product bins that are reshuffled when needed by the dynamic storage system. In each zone an order picker is responsible for picking products from his or her dedicated part of the system. On a normal working day order picking lasts for 9 hours and there are in total 220 totes simultaneously in the system that need to be served. On busy days, this number can increase to 280 before the conveyor becomes congested and long merge times start to occur. In addition, the workload control mechanism sets an upper limit of 95 totes that can be present in each segment at the same time.

In Figure 4.1b a zone is shown together with the dynamic storage system. Products are stored at two locations; the *picking area* and the *bulk storage area*. Products stored in the picking area are easily accessible by the order picker and are used to



(a) Zone picking system



(b) Dynamic storage

Figure 4.1: Overview of the system lay-out of the order picking system.

fulfill customer totes that are waiting at the zone. All the other products are stored in the bulk storage area, which is situated behind the picking area. Whenever a product is needed for picking, the Automated Storage and Retrieval (S/R) machine retrieves it from the bulk area, just in time (Yu & De Koster, 2010). Whenever the product is not needed anymore, the S/R machine brings it back to the bulk area. The advantage of a dynamic storage system is that only a fraction of the products is stored in a compact picking area which leads to reduced walking distances and increased ergonomics as it requires less manual lifting and carrying products than a conventional zone picking system. Also, the S/R machine can be used for automated replenishment reducing the chances of congestion in aisles by manual replenishment (Yu & De Koster, 2010).

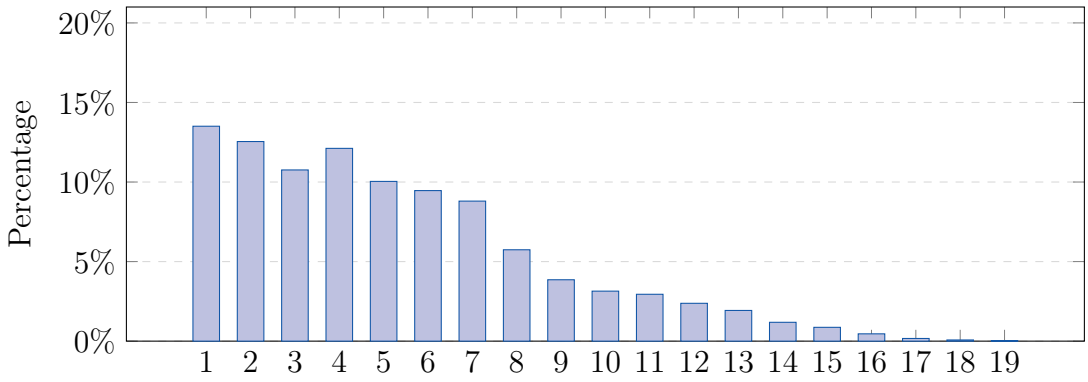
For five representative picking days (9th-10th and 13th-15th of February 2012), data about release probabilities and service times of the zones were obtained after analyzing system logs from the Warehouse Management System (WMS). In Table 4.1 for each of the five picking days the number of order lines picked per zone and the number of times a tote visited a zone are given. It can be seen that the zones in the three pick-to-light segments have about the same number of picks and visits. However, the number of picks and visits in the zones of the pallet pick segment p is considerably lower, mainly due to the longer time it requires to pick products from pallets compared to the zones in the pick-to-light segments. Also, the pallet pick segment contains less popular products and oddly shaped products which decreases the probability that a tote has to visit this segment and which increases the picking effort.

Figure 4.2 shows for all the orders of the five picking days the percentages of how many zones are visited (Figure 4.2a), the number of order lines (unique products) are picked (Figure 4.2b), and the number of individual units of products picked per customer order (Figure 4.2c). The number of zones visited per customer order is on average 5.3, whereas the number of order lines to pick is 8.7. Finally, the average number of product units required per customer order is 46.7. Note that this large number of units can result in a substantially smaller number of picks per line, since products may be packed in boxes of 10–12 units.

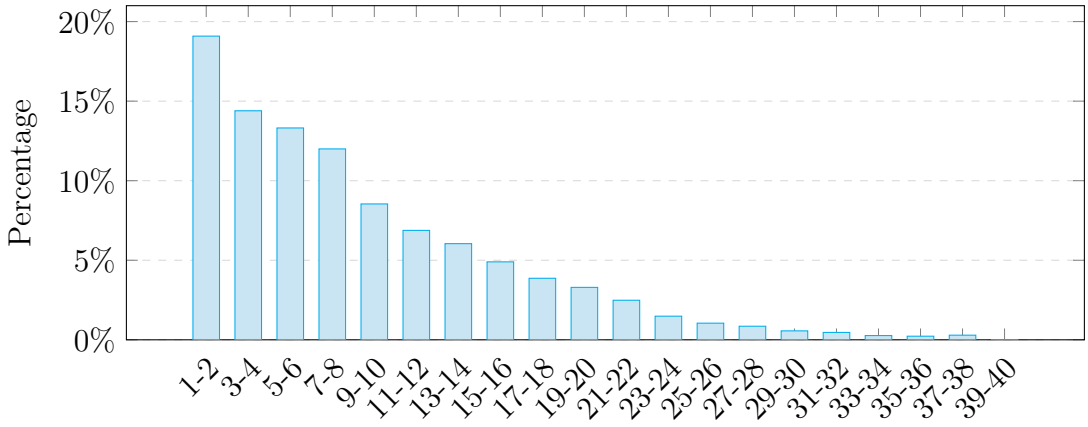
Table 4.2 shows the fraction of all orders that visit a particular zone (given in a row), given that it also visits one of the other zones (given in a column). From

Table 4.1: Number of order lines picked per zone and the number of times a tote visited a zone for the five picking days.

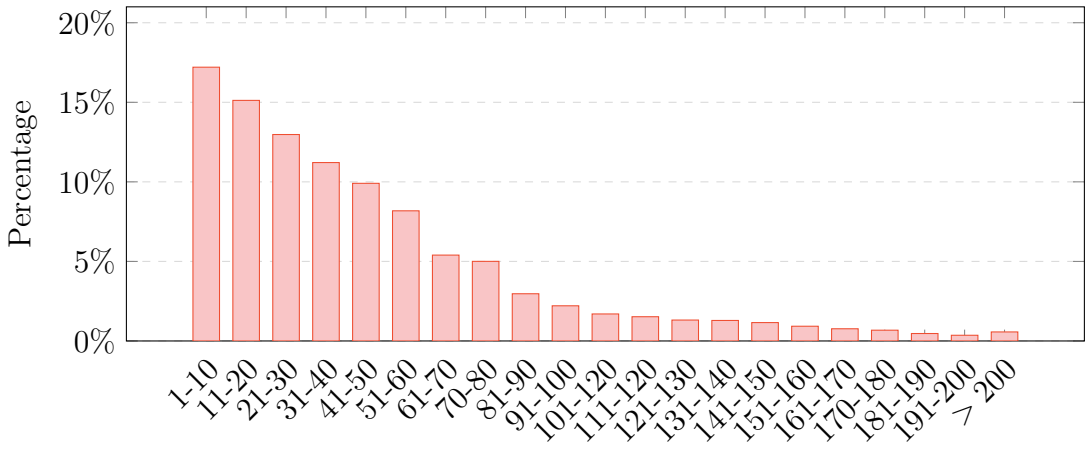
Zone	Day 1		Day 2		Day 3		Day 4		Day 5		Overall	
	Picks	Visits	Picks	Visits	Picks	Visits	Picks	Visits	Picks	Visits	Picks	Visits
z_1^p	260	224	222	180	270	232	323	273	272	236	1347	1145
z_2^p	362	298	230	198	340	275	412	344	365	290	1709	1405
z_3^p	276	249	168	154	272	249	287	261	241	220	1244	1133
z_1^{l1}	1561	962	1124	718	1268	830	1817	1125	2141	1233	7911	4868
z_2^{l1}	1506	1016	1150	796	1696	1069	1829	1191	1910	1248	8091	5320
z_3^{l1}	1073	734	1114	749	1521	969	1369	903	1424	933	6501	4288
z_4^{l1}	1478	940	972	657	1498	987	1676	1084	1769	1125	7393	4793
z_5^{l1}	1256	883	929	645	1348	941	1356	957	1340	947	6229	4373
z_6^{l1}	1366	928	970	698	1333	945	1447	1008	1500	1043	6616	4622
z_7^{l1}	1497	942	1023	701	1430	972	1496	994	1720	1114	7166	4723
z_8^{l1}	979	719	788	575	992	764	1332	951	1407	1025	5498	4034
z_1^{l2}	1163	765	786	542	1043	698	1298	867	1484	959	5774	3831
z_2^{l2}	1594	976	1134	734	1471	959	1538	968	1684	1094	7421	4731
z_3^{l2}	1188	829	884	604	1356	936	1194	826	1323	904	5945	4099
z_4^{l2}	1595	1034	1126	736	1381	918	1677	1029	1763	1132	7542	4849
z_5^{l2}	1423	940	1113	746	1390	864	1542	1032	1617	1080	7085	4662
z_6^{l2}	1103	737	887	624	1074	767	1271	853	1533	1000	5868	3981
z_7^{l2}	1560	1014	1243	815	1284	847	1554	1016	1666	1088	7307	4780
z_8^{l2}	1074	792	910	675	1139	844	1353	973	1488	1040	5964	4324
z_1^{l3}	965	522	655	387	939	509	907	469	1205	704	4671	2591
z_2^{l3}	1636	835	1136	617	1357	719	1411	726	1752	915	7292	3812
z_3^{l3}	1323	640	919	472	1162	596	1243	630	1798	892	6445	3230
z_4^{l3}	1560	817	1129	562	1526	753	1427	742	2051	1032	7693	3906
z_5^{l3}	1573	843	1392	689	1563	789	1624	808	1905	989	8057	4118
z_6^{l3}	1985	839	1342	566	1823	742	1681	727	2391	1049	9222	3923
z_7^{l3}	2177	957	1642	797	1927	874	2087	897	2496	1180	10329	4705
z_8^{l3}	2747	907	2026	771	2514	819	2694	957	3456	1247	13437	4701
Total	36,280	21,342	27,014	16,408	34,917	20,867	37,845	22,611	43,701	25,719	179,757	106,947



(a) Percentage of number of zones visited per customer order



(b) Percentage of number of order lines required per customer order



(c) Percentage of number of product units required per order

Figure 4.2: Descriptive statistics of all the orders of the five picking days.

the table it can be clearly seen that totes often need to visit multiple zones within the same segment. Totes that visit the pallet pick segment also regularly visit the first pick-to-light segment, whereas a tote that needs to visit the first pick-to-light segment also has a high chance to visit the second pick-to-light segment and vice versa. The totes that need to visit the third pick-to-light segment normally only visit this segment, mainly because the company stores a distinct product range only on the second floor.

Finally, in Table 4.3a the mean and coefficient of variation of the empirical order picking time distribution obtained from the log files for each zone is given, as well as, the number of order pickers and input buffer sizes per zone. In each zone the number of order pickers d_i equals 1, while the input buffer size q_i depends on the location of the zone. For the zones that use pick-to-light, $q_i = 11$, except for the first and last zone in each segment for which q_i is either 8 or 9. The buffer sizes of the zones in the first segment are 12, 17, and, 19 respectively. Order pickers have to manually push the order totes back on the conveyor, which means that there is no output buffer after a zone. In addition, the mean picking times vary between 18 seconds in the pick-to-light zones and 33 seconds in the pallet pick zones.

In Table 4.3b the deterministic conveying times between the various components in the system are given. The times vary between 25 up to 180 seconds per conveyor segment depending on the location of the conveyor. The time spent in the segment entrance stations equals 5 seconds.

4.3 Product allocation in zone picking systems

In this section we present the different product allocation methods. In particular, we change the product allocation by interchanging zones between the three pick-to-light segments. We exclude the zones in the pallet pick segment since they cannot be shifted to a pick-to-light segment because of the unique characteristics of the products in these zones and the fact that pallets cannot be stored in the dynamic storage system and vice versa. Also, we assume that product allocation to a shifted zone remains the same, because otherwise the empirical picking distribution of Table 4.3a is not valid anymore and the S/R machine might start to become a bottleneck. Also,

Table 4.2: The percentage of co-occurrence that when a customer order needs to visit a zone (column) it also needs to visit another zone (row).

	Pallet pick			Pick-to-light 1								Pick-to-light 2								Pick-to-light 3							
	z_1^p	z_2^p	z_3^p	z_1^1	z_2^1	z_3^1	z_4^1	z_5^1	z_6^1	z_7^1	z_8^1	z_1^2	z_2^2	z_3^2	z_4^2	z_5^2	z_6^2	z_7^2	z_8^2	z_1^3	z_2^3	z_3^3	z_4^3	z_5^3	z_6^3	z_7^3	z_8^3
z_1^p	1.00	0.40	0.28	0.39	0.17	0.27	0.43	0.12	0.36	0.19	0.32	0.05	0.06	0.05	0.06	0.06	0.05	0.05	0.04	0.00	0.03	0.02	0.01	0.01	0.01	0.02	0.03
z_2^p	0.33	1.00	0.23	0.37	0.20	0.23	0.41	0.13	0.32	0.18	0.33	0.10	0.10	0.07	0.10	0.11	0.11	0.09	0.08	0.00	0.03	0.04	0.02	0.01	0.01	0.02	0.03
z_3^p	0.29	0.28	1.00	0.32	0.22	0.24	0.33	0.18	0.28	0.20	0.28	0.13	0.13	0.10	0.15	0.13	0.12	0.12	0.10	0.01	0.05	0.05	0.03	0.02	0.03	0.03	0.05
z_1^1	0.09	0.11	0.07	1.00	0.52	0.39	0.46	0.40	0.43	0.47	0.42	0.34	0.39	0.34	0.40	0.40	0.35	0.38	0.34	0.01	0.13	0.09	0.09	0.06	0.04	0.13	0.10
z_2^1	0.04	0.05	0.05	0.48	1.00	0.38	0.46	0.45	0.40	0.50	0.38	0.37	0.45	0.38	0.46	0.43	0.38	0.42	0.39	0.03	0.16	0.09	0.12	0.10	0.05	0.14	0.11
z_3^1	0.07	0.08	0.06	0.45	0.48	1.00	0.45	0.38	0.44	0.44	0.35	0.29	0.39	0.36	0.41	0.39	0.35	0.40	0.36	0.02	0.10	0.06	0.08	0.07	0.03	0.10	0.07
z_4^1	0.10	0.12	0.08	0.47	0.51	0.40	1.00	0.44	0.45	0.48	0.37	0.35	0.41	0.35	0.42	0.38	0.33	0.36	0.34	0.02	0.15	0.07	0.12	0.10	0.04	0.13	0.09
z_5^1	0.03	0.04	0.05	0.44	0.54	0.38	0.48	1.00	0.42	0.49	0.36	0.40	0.47	0.38	0.48	0.44	0.37	0.42	0.38	0.02	0.17	0.08	0.13	0.12	0.05	0.14	0.09
z_6^1	0.09	0.10	0.07	0.45	0.46	0.41	0.46	0.40	1.00	0.43	0.36	0.34	0.42	0.36	0.40	0.38	0.33	0.36	0.35	0.02	0.11	0.07	0.08	0.08	0.03	0.10	0.09
z_7^1	0.05	0.05	0.05	0.49	0.56	0.40	0.49	0.46	0.42	1.00	0.38	0.39	0.47	0.43	0.48	0.46	0.39	0.43	0.38	0.02	0.16	0.08	0.13	0.10	0.04	0.13	0.09
z_8^1	0.09	0.11	0.08	0.50	0.50	0.37	0.44	0.39	0.42	0.44	1.00	0.36	0.40	0.33	0.39	0.40	0.36	0.37	0.34	0.02	0.12	0.09	0.09	0.05	0.04	0.09	0.10
z_1^2	0.02	0.04	0.04	0.44	0.51	0.32	0.44	0.46	0.41	0.49	0.37	1.00	0.55	0.47	0.58	0.51	0.49	0.52	0.46	0.02	0.15	0.10	0.11	0.06	0.05	0.12	0.13
z_2^2	0.01	0.03	0.03	0.40	0.51	0.35	0.42	0.43	0.41	0.47	0.34	0.45	1.00	0.48	0.55	0.54	0.48	0.55	0.48	0.03	0.18	0.09	0.14	0.12	0.05	0.15	0.10
z_3^2	0.01	0.02	0.03	0.41	0.49	0.38	0.40	0.40	0.40	0.50	0.33	0.44	0.55	1.00	0.53	0.52	0.45	0.49	0.46	0.04	0.17	0.10	0.12	0.09	0.05	0.14	0.10
z_4^2	0.01	0.03	0.04	0.40	0.50	0.37	0.41	0.44	0.39	0.47	0.32	0.46	0.54	0.44	1.00	0.52	0.44	0.55	0.47	0.03	0.17	0.09	0.14	0.12	0.05	0.17	0.11
z_5^2	0.02	0.03	0.03	0.42	0.49	0.36	0.39	0.41	0.38	0.47	0.35	0.42	0.55	0.46	0.54	1.00	0.45	0.55	0.44	0.03	0.18	0.09	0.14	0.12	0.06	0.16	0.12
z_6^2	0.01	0.04	0.03	0.43	0.51	0.38	0.40	0.40	0.39	0.46	0.37	0.47	0.57	0.46	0.54	0.53	1.00	0.57	0.52	0.03	0.14	0.11	0.10	0.06	0.05	0.11	0.12
z_7^2	0.01	0.03	0.03	0.38	0.47	0.36	0.37	0.39	0.35	0.43	0.31	0.42	0.55	0.42	0.56	0.53	0.47	1.00	0.47	0.03	0.18	0.09	0.15	0.12	0.05	0.17	0.11
z_8^2	0.01	0.03	0.03	0.38	0.48	0.36	0.38	0.39	0.37	0.41	0.31	0.41	0.52	0.44	0.53	0.47	0.48	0.52	1.00	0.05	0.16	0.10	0.12	0.10	0.06	0.14	0.11
z_1^3	0.00	0.00	0.01	0.03	0.07	0.03	0.03	0.03	0.03	0.04	0.03	0.03	0.05	0.06	0.06	0.06	0.04	0.05	0.08	1.00	0.56	0.61	0.66	0.65	0.65	0.50	0.56
z_2^3	0.01	0.01	0.01	0.16	0.22	0.11	0.19	0.19	0.14	0.20	0.12	0.15	0.22	0.18	0.22	0.23	0.15	0.23	0.18	0.38	1.00	0.53	0.64	0.51	0.47	0.46	0.44
z_3^3	0.01	0.02	0.02	0.14	0.15	0.08	0.11	0.11	0.10	0.12	0.11	0.12	0.14	0.13	0.14	0.13	0.13	0.14	0.13	0.49	0.63	1.00	0.67	0.51	0.60	0.44	0.52
z_5^3	0.00	0.01	0.01	0.11	0.16	0.08	0.15	0.15	0.10	0.15	0.09	0.11	0.17	0.12	0.17	0.17	0.11	0.18	0.14	0.44	0.62	0.56	1.00	0.54	0.55	0.46	0.45
z_5^3	0.00	0.00	0.00	0.07	0.13	0.07	0.12	0.12	0.08	0.11	0.05	0.06	0.14	0.09	0.14	0.14	0.06	0.14	0.11	0.41	0.48	0.40	0.51	1.00	0.56	0.57	0.56
z_6^3	0.00	0.00	0.01	0.05	0.07	0.03	0.05	0.05	0.04	0.05	0.04	0.05	0.06	0.05	0.07	0.07	0.05	0.06	0.07	0.43	0.46	0.49	0.55	0.59	1.00	0.54	0.63
z_7^3	0.01	0.00	0.01	0.13	0.16	0.09	0.13	0.13	0.10	0.13	0.07	0.10	0.15	0.12	0.17	0.16	0.09	0.17	0.13	0.28	0.38	0.30	0.38	0.50	0.45	1.00	0.65
z_8^3	0.01	0.01	0.01	0.11	0.13	0.06	0.09	0.08	0.08	0.09	0.08	0.10	0.10	0.09	0.11	0.12	0.10	0.11	0.10	0.31	0.36	0.36	0.38	0.49	0.53	0.65	1.00

Table 4.3: Overview parameters zones and conveyor nodes.

(a) Zone parameters, d_i; number of order pickers, q_i; input buffer size, μ_i^{-1}; empirical mean order picking time in seconds, cv_i; coefficient of variation of the empirical order picking time

Zone	d_i	q_i	Empr. dist.		Zone	d_i	q_i	Empr. dist.		Zone	d_i	q_i	Empr. dist.	
			μ_i^{-1}	cv_i				μ_i^{-1}	cv_i				μ_i^{-1}	cv_i
z_1^p	1	12	26.5	1.06	z_7^1	1	11	21.5	0.98	z_8^2	1	8	24.5	0.91
z_2^p	1	17	28.4	1.11	z_8^1	1	8	23.1	0.96	z_1^3	1	8	25.2	0.89
z_3^p	1	19	32.9	1.16	z_1^2	1	9	24.6	0.87	z_2^3	1	11	26.7	0.86
z_1^1	1	9	18.3	0.79	z_2^2	1	11	25.4	0.87	z_3^3	1	11	26.0	0.92
z_2^1	1	11	19.8	0.89	z_3^2	1	11	26.2	0.92	z_4^3	1	11	25.3	0.85
z_3^1	1	11	21.6	0.95	z_4^2	1	11	24.2	0.83	z_5^3	1	11	22.2	0.78
z_4^1	1	11	21.0	0.93	z_5^2	1	11	24.6	0.87	z_6^3	1	11	21.9	0.88
z_5^1	1	11	22.5	0.82	z_6^2	1	11	24.0	0.90	z_7^3	1	11	23.4	0.83
z_6^1	1	11	21.2	0.91	z_7^2	1	11	26.4	0.91	z_8^3	1	8	25.4	0.84

(b) Conveyor node parameters, μ_i^{-1}; empirical conveying time in seconds (deterministic)

Conveyor	μ_i^{-1}	Conveyor	μ_i^{-1}	Conveyor	μ_i^{-1}	Conveyor	μ_i^{-1}
c_1^m	37	c_1^1	37	c_1^2	37	c_1^3	37
c_2^m	37	c_2^1	37	c_2^2	37	c_2^3	37
c_3^m	70	c_3^1	37	c_3^2	37	c_3^3	37
c_4^m	180	c_4^1	37	c_4^2	37	c_4^3	37
c_5^m	180	c_5^1	62	c_5^2	62	c_5^3	62
c_1^p	43	c_6^1	37	c_6^2	37	c_6^3	37
c_2^p	43	c_7^1	37	c_7^2	37	c_7^3	37
c_3^p	43	c_8^1	37	c_8^2	37	c_8^3	37
c_4^p	130	c_9^1	25	c_9^2	25	c_9^3	25

we only consider the assignment of zones to a segment, instead assigning the zones to a specific location. Within the segment we can neglect the location assignment, because the order the zones are visited within a segment has no significant effect on the performance of the system. This is due to the unidirectional design of the conveyor in the segment that requires totes to visit all zones before leaving the segment. Finally, we assume that the order picker never has to wait for the S/R machine to reshuffle a product bin from the bulk storage area to the picking area. This will allow us to study the performance of the zone picking system, without the interference of other related processes.

This section is structured as follows. In Section 4.3.1 the notation used in the allocation methods is given. The next three sections describe three product allocation methods, where the first minimizes the average number of segments a tote has to visit. The second method balances the total number of visits per segment on average, and whereas the third method combines the previous two models into a single model.

4.3.1 Notation

$\mathcal{K}$: The set of pick-to-light segments, $\{1, 2, 3\}$.

$\mathcal{Z}$: The set of pick-to-light zones, $\{1, 2, \ldots, 24\}$.

$\mathcal{N}$: The set of customer tote classes. The class of a customer tote $n \in \mathcal{N}$ indicates which zones the tote has to visit, e.g., $n = \{1, 3\}$ means that the tote has to visit the first and third zone.

D_n: The fraction of customer tote class $n \in \mathcal{N}$ per day or over the five picking days.

M: A large number (big M).

δ_{ni}: $\begin{cases} 1, & \text{if customer tote class } n \in \mathcal{N} \text{ needs to visit zone } i \in \mathcal{Z}, \\ 0, & \text{otherwise.} \end{cases}$

x_{ik}: $\begin{cases} 1, & \text{if zone } i \in \mathcal{Z} \text{ is assigned to segment } k \in \mathcal{K}, \\ 0, & \text{otherwise.} \end{cases}$

v_{nk}: $\begin{cases} 1, & \text{if customer tote class } n \in \mathcal{N} \text{ needs to visit segment } k \in \mathcal{K}, \\ 0, & \text{otherwise.} \end{cases}$

4.3.2 Minimized segment visits model

In the current situation the zones are allocated to the segments such that the number of segments a tote has to visit on average is minimized. The advantage of this method is that it reduces the total order tote travel distance, since the conveyor topology allows that large parts of the system can be cut short. However, it may cause congestion in a segment, since certain zones with popular items are visited more often than others. If these zones are placed in the same segment, the probability of a tote being blocked by a segment increases.

The model used to minimize the number of segments a tote visits is defined as the following integer programming model;

$$\text{minimize} \quad \sum_{n\in\mathcal{N}}\sum_{k\in\mathcal{K}} v_{nk}D_n \tag{4.1}$$

$$\text{subject to} \quad \sum_{k\in\mathcal{K}} x_{ik} = 1 \qquad \forall i \in \mathcal{Z} \tag{4.2}$$

$$\sum_{i\in\mathcal{Z}} x_{ik} = 8 \qquad \forall k \in \mathcal{K} \tag{4.3}$$

$$\sum_{i\in\mathcal{Z}} x_{ik}\delta_{ni} \leq M v_{nk} \qquad \forall n \in \mathcal{N},\, k \in \mathcal{K} \tag{4.4}$$

$$x_{ik}, v_{nk} \in \{0,1\} \qquad \forall i \in \mathcal{Z},\, j \in \mathcal{N},\, k \in \mathcal{K} \tag{4.5}$$

The objective of the first model (4.1) is to minimize the average number of segments a customer tote has to visit. Constraints (4.2) ensure that each zone is assigned to one of the three segments. On the other hand, constraints (4.3) define that each segment should contain eight assigned zones. Constraints (4.4) ensure if order class $n \in \mathcal{N}$ has to visit segment $k \in \mathcal{K}$, then v_{nk} is equal to one, otherwise v_{nk} will be zero. Since each segment should contain eight zones, the smallest big M possible is 8. Finally, constraints (4.5) are the integrality constraints.

A problem encountered when trying to solve the program is that the IP is symmetric, because the zone allocation to segments can be permuted in such a way the structure of the problem does not change. Therefore, in order to reduce symmetry, a common approach is to add symmetry-breaking constraints (Sherali & Smith, 2001). In our case we add the following constraints that ensure that zone i can be assigned to segment k only if zone $1, \ldots, i-1$ are assigned to segment $1, \ldots, k-1, k$. This gives

the following symmetry breaking constraint,

$$\sum_{l=1}^{i} 2^{i-l} x_{l,k-1} \geq \sum_{l=1}^{i} 2^{i-l} x_{lk} \qquad i \in \mathcal{Z}, \quad k \in \mathcal{K} \backslash \{1\}\,. \tag{4.6}$$

A proof of the validity of this constraint in a more general context is given in Proposition 2 in Sherali & Smith (2001).

4.3.3 Balanced workload model

The second method is to store products in the zones such that the average workload between the segments is balanced. The product allocation balances the workload between segments by that it increases or decreases the number of totes that visit a particular segment on average. This will lower the probability that a segment and its zones become a bottleneck and, ultimately, will increase the performance of the order picking system. A downside of this policy is that a tote can spend more time in the system, since the probability of visiting more segments increases.

The model used to balance the average workloads between the segments is defined as the following integer programming model;

$$\text{minimize} \quad \sum_{k=1}^{2} \sum_{k'=k+1}^{3} \left| \sum_{n \in N} v_{n,k} D_n - \sum_{n \in N} v_{n,k'} D_n \right| \tag{4.7}$$

$$\text{subject to} \quad \sum_{k \in \mathcal{K}} x_{ik} = 1 \qquad \forall i \in \mathcal{Z} \tag{4.8}$$

$$\sum_{i \in \mathcal{Z}} x_{ik} = 8 \qquad \forall k \in \mathcal{K} \tag{4.9}$$

$$\sum_{i \in \mathcal{Z}} x_{ik} \delta_{ni} \leq M v_{nk} \qquad \forall n \in \mathcal{N},\ k \in \mathcal{K} \tag{4.10}$$

$$\sum_{i \in \mathcal{Z}} x_{ik} \delta_{ni} \geq v_{nk} \qquad \forall n \in \mathcal{N},\ k \in \mathcal{K} \tag{4.11}$$

$$x_{ik}, v_{nk} \in \{0, 1\} \qquad \forall i \in \mathcal{Z},\ j \in \mathcal{N},\ k \in \mathcal{K} \tag{4.12}$$

The objective of the second model (4.7) is to minimize the differences between the total number of totes that visit the three segments. Constraints (4.8)-(4.12) have the same interpretation as constraints (4.2)-(4.5). Additionally constraints (4.11) are

needed to make sure that v_{nk} is only set to 1 if totes of tote class n actually visits segment k.

The model cannot be directly solved by a IP solver, because of the absolute value the objective function (4.7) is nonlinear. However, we can easily linearize the model by introducing non-negative variables y_k^+, y_k^-, $k \in \mathcal{K}$ and by modifying the objective function to

$$\text{minimize} \quad \sum_{k \in \mathcal{K}} y_k^+ + y_k^- \tag{4.13}$$

and by adding the following constraints to the model,

$$\sum_{n \in N} v_{n,1} D_n - \sum_{n \in N} v_{n,2} D_n = y_1^+ - y_1^- \tag{4.14}$$

$$\sum_{n \in N} v_{n,1} D_n - \sum_{n \in N} v_{n,3} D_n = y_2^+ - y_2^- \tag{4.15}$$

$$\sum_{n \in N} v_{n,2} D_n - \sum_{n \in N} v_{n,3} D_n = y_3^+ - y_3^- \tag{4.16}$$

$$y_k^+ + y_k^- \geq 0 \quad \forall k \in \mathcal{K} \tag{4.17}$$

Finally, symmetry breaking constraint (4.6) can also be applied in this model.

4.3.4 Combined model

The last model combines the objective function of the previous two models given a prespecified scaling factor α. The objective of this model is given by

$$\text{minimize} \quad \alpha \sum_{k=1}^{2} \sum_{k'=k+1}^{3} \left| \sum_{n \in N} v_{n,k} D_n - \sum_{n \in N} v_{n,k'} D_n \right| + (1-\alpha) \sum_{n \in \mathcal{N}} \sum_{k \in \mathcal{K}} v_{nk} D_n \tag{4.18}$$

The constraints of the model are the same for the first model. Again, because of the absolute value the objective has to be linearized as in Section 4.3.3. In addition, constraints (4.6) are used for improved solver performance.

4.4 Numerical results

In this section the performance effects of the different product allocation methods will be tested for the real zone picking system. In Section 4.4.1 the results of zone allocation of the three methods is presented and in Section 4.4.2 the different allocation will be tested in a simulation of the real zone picking system and compared with the analytical model of Van der Gaast et al. (2012).

All the results were obtained on a Core i7 with 2.4 GHz and 8 GB of RAM. The IP-models were solved in Gurobi 6.0.4. The discrete-event simulation was implemented in Java. The simulation model was run 15 times for 1,000,000 seconds, preceded by 10,000 seconds of initialization for the system to become stable, which guaranteed that the 95% confidence interval width of the system throughput time is less than 1% of the mean value for all the cases.

4.4.1 Product allocation methods

In this section the results of the three product allocation methods are presented. First, for each of the five picking days the number of unique customer classes $n \in \mathcal{N}$ and their occurrence were obtained from the log files. In Table 4.4 the number of unique customer classes are shown for each of the five picking days, as well as, the overall where all the picking data was aggregated over all the days.

Table 4.4: The number of unique customer tote classes per picking day and overall.

	Day 1	Day 2	Day 3	Day 4	Day 5	Overall
$\lvert\mathcal{N}\rvert$	392	271	436	385	408	1596

The results of solving the three models for each individual day, as well as, aggregated over all the picking days is shown in Table 4.5. *Model 1* corresponds with the minimized segment visits model of Section 4.3.2, *Model 2* with the balanced workload model of Section 4.3.3, and *Model 3* with the combined model of Section 4.3.4. All the models were solved to optimality within a couple of minutes for the individual days and two hours for the overall model. It can be seen that for *Model 1* the product allocation is almost identical every day and is similar to current allocation

presented in Section 4.2. However, the allocation of *Model 2* varies a lot per day as it tries to balance the workload on a day-to-day basis which can differ significantly as seen from Table 4.1. Finally, for *Model 3* two different scaling factors α were used. Similar as *Model 1*, both allocations differ slightly per day, as well as, for all the picking days combined. This implies that both scaling factors favor the second part of objective (4.18) more.

Table 4.5: Results of the zone allocation to segments of the different product allocation methods.

		Zones																							
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
Day 1	Model 1	1	1	1	1	1	1	1	1	2	2	2	2	2	2	2	2	3	3	3	3	3	3	3	3
	Model 2	1	2	1	3	3	1	3	1	3	2	3	1	1	3	2	1	2	2	3	2	2	2	3	1
	Model 3 ($\alpha = 0.5$)	1	1	2	1	1	2	1	1	2	1	2	2	2	1	2	2	3	3	3	3	3	3	3	3
	Model 3 ($\alpha = 0.2$)	1	1	2	1	1	2	1	1	2	1	2	2	2	1	2	2	3	3	3	3	3	3	3	3
Day 2	Model 1	1	1	2	1	1	1	1	1	2	2	2	1	2	2	2	2	3	3	3	3	3	3	3	3
	Model 2	1	1	1	2	3	2	3	1	2	2	3	1	1	1	2	2	2	3	3	3	1	2	3	3
	Model 3 ($\alpha = 0.5$)	1	1	2	1	2	1	1	1	1	2	2	2	2	1	2	2	3	3	3	3	3	3	3	3
	Model 3 ($\alpha = 0.2$)	1	1	2	1	2	1	1	1	1	2	2	2	2	1	2	2	3	3	3	3	3	3	3	3
Day 3	Model 1	1	1	1	1	1	1	1	1	2	2	2	2	2	2	2	2	3	3	3	3	3	3	3	3
	Model 2	1	2	3	2	1	3	3	3	1	2	3	1	3	2	2	3	2	1	1	2	2	3	1	1
	Model 3 ($\alpha = 0.5$)	1	1	1	2	1	2	1	2	2	2	1	2	1	2	1	2	3	3	3	3	3	3	3	3
	Model 3 ($\alpha = 0.2$)	1	1	1	2	1	2	1	2	2	2	1	2	1	2	1	2	3	3	3	3	3	3	3	3
Day 4	Model 1	1	1	1	1	1	1	1	1	2	2	2	2	2	2	2	2	3	3	3	3	3	3	3	3
	Model 2	1	2	3	2	1	2	1	1	2	3	2	1	1	2	2	1	3	3	3	3	3	2	1	3
	Model 3 ($\alpha = 0.5$)	1	2	1	1	2	2	1	2	1	1	2	1	2	2	1	2	3	3	3	3	3	3	3	3
	Model 3 ($\alpha = 0.2$)	1	1	1	2	1	2	1	2	1	2	2	1	1	2	2	2	3	3	3	3	3	3	3	3
Day 5	Model 1	1	1	1	1	1	1	1	1	2	2	2	2	2	2	2	2	3	3	3	3	3	3	3	3
	Model 2	1	1	2	2	1	1	3	3	3	2	2	3	2	2	1	3	1	1	3	3	2	1	3	2
	Model 3 ($\alpha = 0.5$)	1	1	2	2	1	2	1	1	1	1	2	1	2	2	2	2	3	3	3	3	3	3	3	3
	Model 3 ($\alpha = 0.2$)	1	1	2	2	1	2	1	1	1	2	2	1	1	2	2	2	3	3	3	3	3	3	3	3
Overall	Model 1	1	1	1	1	1	1	1	1	2	2	2	2	2	2	2	2	3	3	3	3	3	3	3	3
	Model 2	1	1	2	1	3	3	2	3	1	3	3	1	3	1	2	2	2	1	2	2	3	3	1	2
	Model 3 ($\alpha = 0.5$)	1	2	2	1	1	2	1	1	1	1	2	2	1	2	2	2	3	3	3	3	3	3	3	3
	Model 3 ($\alpha = 0.2$)	1	1	2	2	2	1	1	1	1	2	2	1	2	1	2	2	3	3	3	3	3	3	3	3

Figure 4.3 shows for the four allocations the visiting probability $\tilde{\psi}_k = \sum_{\boldsymbol{r} \subseteq \mathcal{Z}} \psi_{\boldsymbol{r}} I_{(\boldsymbol{r}^k \neq 0)}$ of pick-to-light segment k for the aggregated picking data. On average, totes will visit 1.61 pick-to-light segments in case of the allocation of *Model 1*, 2.33 for *Model 2*, and 1.70 for both $\alpha = 0.5$ and $\alpha = 0.2$ of *Model 3*. The aggregated visiting probabilities $\tilde{\psi}_k$ of the allocation of *Model 1* show that totes on average visit the first and third pick-to-light segment on average 0.43 and 0.49 times respectively. However, the second pick-to-light segment is only visited on average 0.29 times. The objective of *Model 2* was to balance the workload between segments; from the results it can be clearly seen that this is accomplished by having totes visiting each pick-to-light segment on average 0.68 times. The results for allocations of *Model 3* are similar to *Model 1* except that the second pick-to-light segment is visited now more often.

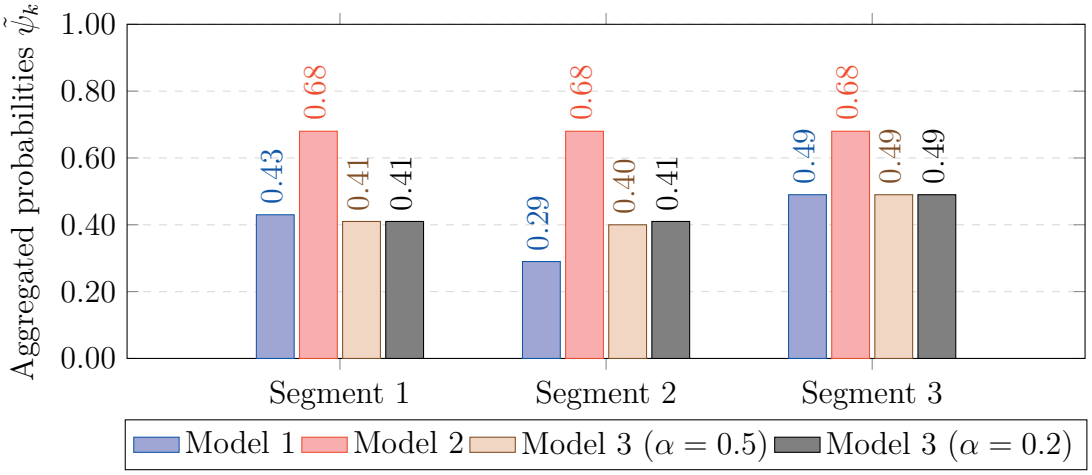


Figure 4.3: The aggregated probabilities $\tilde{\psi}_k$ of a tote visiting pick-to-light segment k for the four overall product allocations.

4.4.2 Validation with real zone picking system

The performance of the four product allocations is compared by varying the number of totes in the system N from 220 up to 300 for the aggregated picking data. The simulation uses the empirical picking time distributions of Table 4.3. Figure 4.4 shows for the four allocations the impact on the system throughput $X(N)$ when increasing the number of totes in the system. The 95% confidence interval is shown for the simulation results of each policy. The mean average error in the system throughput

is 1.26% for the allocation of *Model 1*, 0.67% for the *Model 2*, and 1.23% and 1.18% for $\alpha = 0.5$ and $\alpha = 0.2$ for *Model 3*. The errors are slightly higher than in the previous chapters, due to the use of empirically instead of exponentially distributed picking times in the simulation (cf. Remark 2.2). For the four allocations, for high N the approximated system throughput is always higher than the one obtained from simulation, since the approximation tends to underestimate the blocking probabilities as seen in the previous chapters.

Table 4.6: The mean and maximum blocking percentages for $N = 300$ for the zones and segments for the MVA and the simulation.

(a) Zones

	MVA		Simulation	
	Mean	Max	Mean	Max
Model 1	5.2	50.5	6.0	51.9
Model 2	2.2	8.0	3.0	14.7
Model 3 ($\alpha = 0.5$)	4.3	43.0	5.2	42.4
Model 3 ($\alpha = 0.2$)	3.7	22.8	4.8	36.8

(b) Segments

	MVA		Simulation	
	Mean	Max	Mean	Max
Model 1	1.9	5.0	7.7	19.8
Model 2	0.0	0.2	0.0	0.4
Model 3 ($\alpha = 0.5$)	1.2	3.4	4.7	13.4
Model 3 ($\alpha = 0.2$)	0.4	2.0	1.6	7.9

When $N = 220$ zones and segments become rarely congested. This implies that minimizing the mean number of segments a tote has to visit, which also minimizes the mean total travel distance of a tote, yields the highest throughput. Clearly, this is the case for the allocation of *Model 1*, whereas the throughput of the other three allocations are lower due to totes traveling larger distances. Increasing N, increases the probability that a tote is blocked by a full zone or a segment. When $N = 300$, the system throughput of *Model 1* and *Model 3* with $\alpha = 0.2$ are close to each other, whereas the other two allocations obtain significantly lower throughput rates. Especially, in *Model 2* the system throughput on average is 8% lower than *Model 1*.

In Table 4.6 the mean and maximum blocking probability for the zones and segments when the number of totes $N = 300$ for the MVA and the simulation of the four allocations is shown. In this case the number of totes in the system is higher than the real-world system limit (280 totes) set by management. From the results it can be clearly seen that some zones and segments are blocked too often than what would be acceptable in practice. Still, under the allocation of *Model 1*, totes are more often

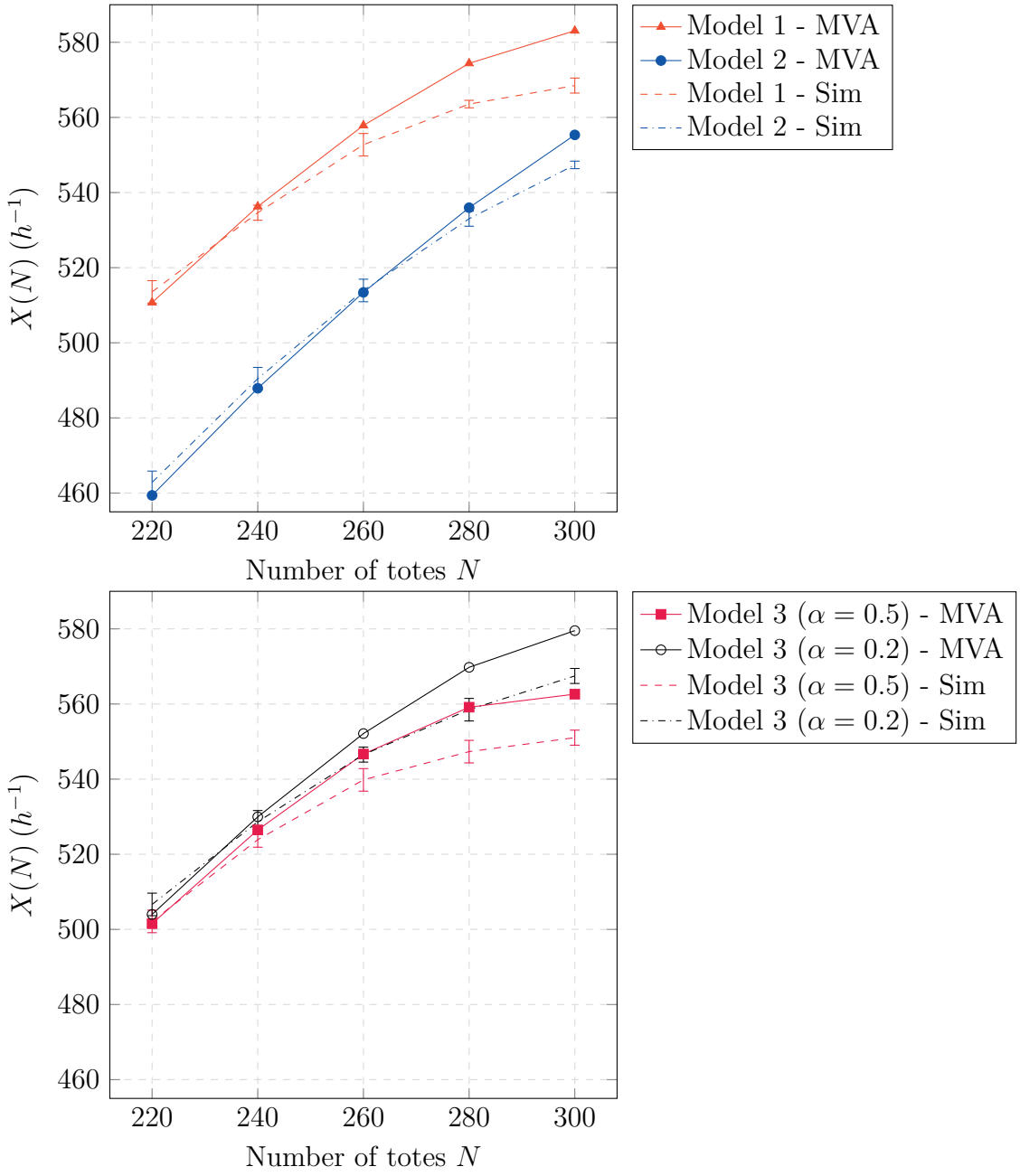


Figure 4.4: The system throughput of the four product allocations for the approximation and the simulation when varying the number of totes N.

blocked by zones and segments than for the other allocations. Since a blocked tote has to recirculate on the main conveyor, the mean total travel distance of a tote increases and the system throughput decreases. By balancing the work-load between segments, congestion starts to occur later, which is especially true for *Model 2*.

4.5 Conclusion and further research

In this chapter, we compared different product allocation methods and tested their influence on system performance on a real-world zone picking system. We tested three product allocation methods and found that a product allocation that only applies workload balancing between segments reduces the average system throughput on average 7% compared to a product allocation minimizing the number of segments a tote on average has to visit. On the other hand, blocking of zones and segments is significantly reduced by a product allocation that applies workload balancing, e.g. segments are blocked 7.6% on average when a product allocation that minimizing the number of segments is used to 0.3% on average in the other case. As such, it provides a valuable tool for initial product allocation decisions for zone picking systems and the consequences strategic decisions can have on the overall performance of the system.

A topic for further research would be to investigate product allocation to zones instead of shifting entire zones to another segment. This would make it possible to balance the workload better between the zones, instead of only between segments. Furthermore, the S/R machine in the dynamic storage has proven to significantly influence the performance of the zone picking system when not properly controlled (Yu & De Koster, 2010). For this it would be interesting to study how to reshuffle the product bins to make sure the order picker never has to wait for an incoming product bin.

5 The analysis of batch sojourn-times in polling systems

5.1 Introduction

Polling models are multi-queue systems in which a single server cyclically visits queues in order to serve waiting customers, typically incurring a switch-over time when moving to the next queue. Polling systems have been extensively used for decades to model a wide variety of applications in areas such as computer and communication systems, production systems, and traffic and transportation systems (Takagi, 2000; Boon et al., 2011). In the majority of the literature on polling systems, it is assumed that in each queue new customers arrive via independent Poisson processes. However, in many applications these arrival processes are not necessarily independent; customers arrive in batches and batches of customers may arrive at different queues simultaneously (Van der Mei, 2002). It is important to consider the correlation structure in the arrival processes for these applications, because neglecting it may lead to strongly erroneous performance predictions, and, consequently, to improper decisions about system performance. In this chapter, we study the *batch sojourn-time* in polling systems with simultaneous arrivals, that is, the time until all the customers in a single batch are served after an arrival epoch.

Batch sojourn-times are of great interest in many applications of polling systems with simultaneous arrivals. Below we describe some examples in manufacturing, warehousing, and communication. The first example is the *stochastic economic lot scheduling problem*, which is used to study the production of multiple products on a single machine with limited capacity, under uncertain demands, production times, and setup times (Federgruen & Katalan, 1999; Winands et al., 2011). In

case of a cyclic policy, there is a fixed production sequence such that the order in which products are manufactured is always known to the manufacturer. Whenever a customer has placed an order for one or multiple products, the machine starts production. After the requested number of products has been produced, including possible demand for the same product of orders that just came in, the machine starts to process the next product in the sequence. In this way, the machine polls the buffers of the different product categories to check whether production is required. In this example, the server represents the machine, a customer represents a unit of demand for a given product, and a batch arrival corresponds to the order itself. The batch sojourn-time is defined as the total time required for manufacturing an entire order.

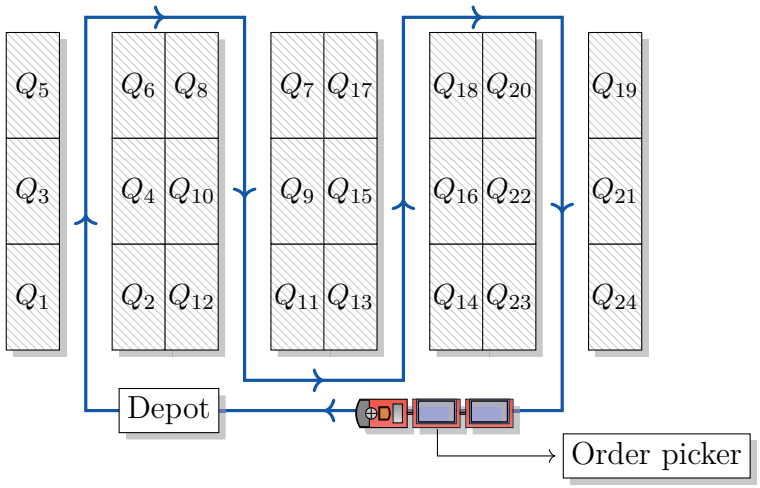


Figure 5.1: A milkrun order picking system with one order picker and 24 different storage locations.

The second example comes from the area of warehousing. In a *milkrun order picking system*, an order picker is constantly moving through the warehouse (e.g. with a tugger train) and receives, using modern order-picking aids like pick-by-voice, pick-by-light, or hand-held terminals, new pick instructions that allow new orders to be included in the current pick route (Gong & De Koster, 2011). In Figure 5.1 a milkrun order picking system is shown, where different products are stored at locations $Q_1, \ldots, Q_N$. Assume that a single order picker is constantly traveling through the aisles with the S-shape routing policy (Roodbergen & De Koster, 2001) and picks all outstanding orders in one pick route to a pick cart, which has sufficient

capacity. An order consists of multiple products that have to be picked at multiple locations in the warehouse. Demand for products that are located upstream of the order picker will be picked in the next picking cycle. When the order picker reaches the depot, the picked products are disposed and sorted per customer order (using a pick-and-sort system) and a new picking cycle will start. The server is represented by the order picker, a new customer order by a batch arrival, a product within an order by a customer in the polling system. The batch sojourn-time is the time required to pick a customer order.

The last example from the area of computer-communication systems is an *I/O subsystem* of a web server. Web servers are required to perform millions of transaction requests per day at an acceptable Quality of Service (QoS) level in terms of client response time and server throughput (Van der Mei et al., 2001). When a request for a web page from the server is made, several file-retrieval requests are made simultaneously (e.g., text, images, multimedia, etc). In many implementations these incoming file-retrieval requests are placed in separate I/O buffers. The I/O controller continuously polls, using a scheduling mechanism, the different buffers to check for pending file-retrieval requests to be executed. The web page will be fully loaded when all its file-retrieval requests are executed. In this application, the server represents the I/O controller, a customer represents an individual file-retrieval request, a batch of customers that arrive simultaneously corresponds to each web page request, and the batch sojourn-time is the time required to fully load a web page.

In the literature, polling systems with simultaneous arrivals have not been studied intensively. Shiozawa et al. (1990) study a two-queue polling system where customers arrive at each station according to an independent Poisson process and, in addition, customers can arrive in pairs at the system and each join a different queue. The authors derive the Laplace-Stieltjes transform of the waiting time distribution of an individual customer and the response time distribution of a pair of customers that arrive simultaneously. Levy & Sidi (1991) study polling models with simultaneous batch arrivals. For models with gated or exhaustive service, they derive a set of linear equations for the expected waiting time at each of the queues. They also provide a pseudo-conservation law for the system, i.e., an exact expression for a specific weighted sum of the expected waiting times at the different queues. Chiarawongse & Srinivasan (1991) also derive pseudo-conservation laws, but in their model all

customers in a batch join the same queue. Finally, Van der Mei (2001) considers an asymmetric cyclic polling model with mixtures of gated and exhaustive service and general service time and switch-over time distributions and studies the heavy traffic behavior. The results were further generalized in Van der Mei (2002).

The objective of this chapter is to analyze the batch sojourn-time in a cyclic polling system with simultaneous batch arrivals. The contribution of this chapter is that we obtain exact expressions for the Laplace-Stieltjes transform of the steady-state batch sojourn-time distribution for the locally-gated, globally-gated, and exhaustive service disciplines, which can be used to determine the moments of the batch sojourn-time, and in particular, its mean. However, we provide an alternative, more efficient way to determine the mean batch sojourn-time by extending the Mean Value Analysis approach of Winands et al. (2006) for the cases of exhaustive and locally-gated service disciplines. We compare the batch sojourn-times for the different service disciplines in several numerical examples and show that the best performing service discipline, minimizing the batch sojourn-time, depends on system characteristics. From the results we conclude that there is no unique best service discipline that minimizes the expected batch sojourn-time. As such, our results provide a starting point for a framework to minimize batch sojourn-times for a given polling system.

The organization of this chapter is as follows. In Section 5.2 a detailed description of the model and the corresponding notation used in this chapter is given. Section 5.3 analyzes the batch sojourn-time for exhaustive service, Section 5.4 does this for locally-gated service, and in Section 5.5 for globally-gated service. We extensively analyze the results of our model in Section 5.6 via computational experiments for a range of parameters. Finally, in Section 5.7 we conclude and suggest some further research topics.

5.2 Model description

Consider a polling system consisting of $N \geq 2$ infinite buffer queues $Q_1, \ldots, Q_N$ served by a single server that visits the queues in a fixed cyclic order. For the ease of presentation, all references to queue indices greater than N or less than 1 are implicitly assumed to be modulo N, e.g., Q_{N+1} is understood as Q_1. Assume

that a new batch of customers arrives according to a Poisson process with rate λ. Each batch of customers is of size $\boldsymbol{K} = (K_1, \ldots, K_N)$, where K_i represents the number of customers entering the system at Q_i, $i = 1, \ldots, N$. The random vector $\boldsymbol{K}$ is assumed to be independent of past and future arriving epochs and at least one element of vector $\boldsymbol{K}$ is larger than 0 and the other elements are larger than or equal to 0, i.e. each batch contains at least one customer. The support with all possible realizations of $\boldsymbol{K}$ is denoted by $\mathcal{K}$ and let $\boldsymbol{k} = (k_1, \ldots, k_N)$ be a realization of $\boldsymbol{K}$. The joint probability distribution of $\boldsymbol{K}$, $\pi(\boldsymbol{k}) = \mathbb{P}(K_1 = k_1, \ldots, K_N = k_N)$ is arbitrary and its corresponding probability generating function (PGF) is given by $\widetilde{K}(\boldsymbol{z}) = E\left(z_1^{K_1} z_2^{K_2} \ldots z_N^{K_N}\right)$. The PGF of the marginal batch size distribution at Q_i is denoted by $\widetilde{K}_i(z) = \widetilde{K}(1, \ldots, 1, z, 1, \ldots, 1)$, $|z| \leq 1$, where the z occurs at the i-th entry. The arrival rate of customers to Q_i is $\lambda_i = \lambda E(K_i)$, and let $E(K_{ij}) = E(K_i K_j)$ for $i \neq j$ and $E(K_{ii}) = E(K_i^2) - E(K_i)$. The total arrival rate of customers arriving in the system is given by $\Lambda = \sum_{i=1}^{N} \lambda_i$.

The service time of a customer in Q_i is a generally distributed random variable B_i with Laplace-Stieltjes transform (LST) $\widetilde{B}_i(.)$, and with first and second moment $E(B_i)$ and $E(B_i^2)$, respectively. The workload at queue Q_i, $i = 1, \ldots, N$ is defined by $\rho_i = \lambda_i E(B_i)$; the overall system load by $\rho = \sum_{i=1}^{N} \rho_i$. In order for the system to be stable, a necessary and sufficient condition is that $\rho < 1$ (Takagi, 1986). In the remainder of this chapter, it is assumed that the condition for stability holds. When the server switches from Q_i to Q_{i+1}, it incurs a generally distributed switch-over time S_i with LST $\tilde{S}_i(.)$, and first and second moment $E(S_i)$ and $E(S_i^2)$. Let $E(S) = \sum_{i=1}^{N} E(S_i)$ be the total switch-over time in a cycle and $E(S^2) = \sum_{i=1}^{N} E(S_i^2) + \sum_{i \neq j} E(S_i) E(S_j)$ its second moment.

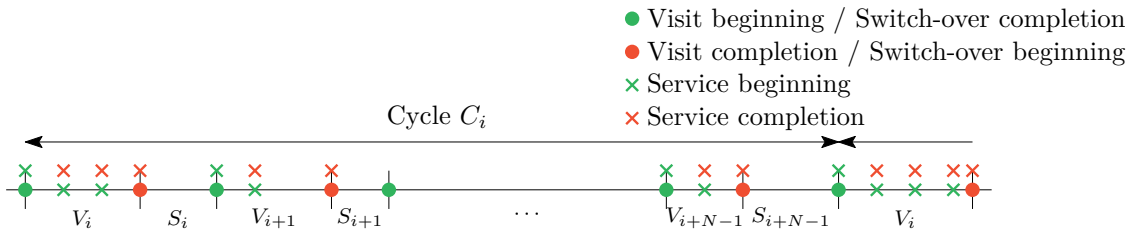


Figure 5.2: Description of a cycle, visit periods, and switch-over times.

The cycle time C_i of Q_i is defined as the time between two successive visits beginning of the server at this queue. A cycle consists of N visit periods each followed by

a switch-over time; $V_i, S_i, V_{i+1}, \ldots, V_{i+N-1}, S_{i+N-1}$ (see Figure 5.2). A visit period, V_i, starts whenever there are customers waiting at Q_i with a service beginning and ends with a service completion. Its duration equals the sum of service times of the customers served during the current visit to Q_i. By definition, a visit beginning always corresponds with a switch-over completion, whereas a visit completion corresponds with a switch-over beginning. In case there are no customers waiting at Q_i, these two epochs coincide. It is well known that the mean cycle length is independent of the queue involved (and the service discipline) and is given by (see, e.g., Takagi (1986)) $E(C) = E(S) / (1 - \rho)$.

In this chapter three different service policies are considered that satisfy the branching property (Resing, 1993). Under the *exhaustive policy*, when a visit beginning starts at Q_i the server continues to work until the queue becomes empty. Any customer that arrives during the server's visit to Q_i is also served within the current visit. However, under the *locally-gated policy*, the server only serves the customers that were present at Q_i at its visit beginning; all customers that arrive during the course of the visit are served in the next visit to Q_i. The final policy is the *globally-gated policy*; according to this policy the server will only serve the customers who were present at all queues at the visit beginning of a reference queue, which is normally assumed to be Q_1. Customers arriving after this visit beginning will only be served after the server has finished its current cycle. This policy strongly resembles the locally-gated policy, except that all queues are gated at the same time instead of one per visit beginning.

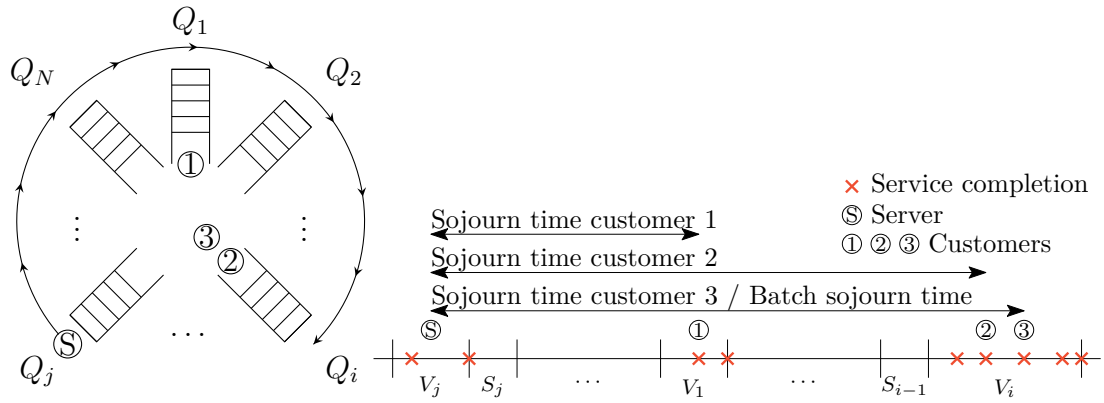


Figure 5.3: Description of the batch sojourn-time.

The batch sojourn-time of a specific customer batch $\boldsymbol{k}$, denoted by $T_{\boldsymbol{k}}$ and its LST by $\widetilde{T}_{\boldsymbol{k}}(.)$, is defined as the time between its arrival epoch until the service completion of the last customer in the arrived batch; see Figure 5.3. In this example assume that when the server is in a visit period of Q_j, a batch of three customers arrives in Q_1 and Q_i. Then the batch sojourn-time of this batch equals the residual time in V_j, switch-over times $S_j, \ldots, S_{i-1}$, visit periods $V_{j+1}, \ldots, V_{i-1}$, and the time until service completion of the last customer of the batch in V_i. By definition, the batch sojourn-time corresponds with the sojourn-time of the last customer that is served within the batch. It is important to realize that the queue where the batch finishes service *depends* on the location of the server of the arrival of the batch and there is no fixed order in which the customers need to be served. The order in which the customers are served in this example is the same for the three service policies, but varies between disciplines depending on the location of the server. Finally, the batch sojourn-time of an arbitrary customer batch is denoted by T and its corresponding LST by $\widetilde{T}(.)$.

Throughout this chapter we make references to the server path from Q_i to Q_j, which should be understood in a cyclic sense; e.g. $Q_i, Q_{i+1}, \ldots, Q_j$ if $i \leq j$, and otherwise $Q_i, Q_{i+1}, \ldots, Q_N, Q_1, \ldots, Q_j$ if $i > j$. For the ease of notation, we define a *cyclic sum* and, analogously, a *cyclic product* as (Boxma et al., 1990)

$$\sum_{l=i}^{j}{}' x_l := \begin{cases} \sum_{l=i}^{j} x_l, & \text{if } i \leq j, \\ \sum_{l=i}^{N} x_l + \sum_{l=1}^{j} x_l, & \text{if } i > j, \end{cases} \qquad \prod_{l=i}^{j}{}' x_l := \begin{cases} \prod_{l=i}^{j} x_l, & \text{if } i \leq j, \\ \prod_{l=i}^{N} x_l \times \prod_{l=1}^{j} x_l, & \text{if } i > j, \end{cases}$$

and alternatively,

$$\sum_{l=0}^{j-i}{}' x_{i+l} := \begin{cases} \sum_{l=0}^{j-i} x_{i+l}, & \text{if } i \leq j, \\ \sum_{l=0}^{j+N-i} x_{i+l}, & \text{if } i > j, \end{cases} \qquad \prod_{l=0}^{j-i}{}' x_l := \begin{cases} \prod_{l=0}^{j-i} x_{i+l}, & \text{if } i \leq j, \\ \prod_{l=0}^{j+N-i} x_{i+l}, & \text{if } i > j. \end{cases}$$

Finally, let $\mathcal{K}_{i,j}$ be a subset of support $\mathcal{K}$ where the last customer of an arbitrary arriving customer batch is served in Q_j and all its other customers are served in

$Q_i, \ldots, Q_j$. By definition, a batch will complete its service in one of the queues, such that $\bigcup_{j=1}^{N} \mathcal{K}_{i,j} = \mathcal{K}$, $i = 1, \ldots, N$. The corresponding probability of subset $\mathcal{K}_{i,j}$ is given by,

$$\pi\left(\mathcal{K}_{i,j}\right) = \begin{cases} \mathbb{P}\left(K_j > 0, K_{j+1} = 0, \ldots, K_{i-1} = 0\right), & j = 1, \ldots, N,\ i \neq j+1, \\ \mathbb{P}\left(K_j > 0\right), & \text{otherwise.} \end{cases}$$

In addition, let $E\left(K_l|\mathcal{K}_{i,j}\right)$ be the conditional expected number of customers that have arrived in Q_l, $l = 1, \ldots, N$ given subset $\mathcal{K}_{i,j}$. We define $\widetilde{K}\left(\boldsymbol{z}|\mathcal{K}_{i,j}\right)$ as the conditional PGF of the distribution of the number of customers that arrive in $Q_i, \ldots, Q_j$ given $\mathcal{K}_{i,j}$,

$$\widetilde{K}\left(\boldsymbol{z}|\mathcal{K}_{i,j}\right) = \sum_{\boldsymbol{k} \in \mathcal{K}_{i,j}} \frac{\pi\left(\boldsymbol{k}\right)}{\pi\left(\mathcal{K}_{i,j}\right)} \prod_{l=i}^{j}{}' z_l^{k_l}, \tag{5.1}$$

such that $\widetilde{K}\left(\boldsymbol{z}\right) = \sum_{j=1}^{N} \pi\left(\mathcal{K}_{i,j}\right) \widetilde{K}\left(\boldsymbol{z}|\mathcal{K}_{i,j}\right)$, $i = 1, \ldots, N$.

5.3 Exhaustive service

In this section, we start by deriving the LST of the batch sojourn-time distribution of a specific batch of customers in case of exhaustive service. The batch sojourn-time distribution is found by conditioning on the numbers of customers present in each queue at an arrival epoch and then studying the evolution of the system until all customers within the batch have been served. For this analysis, we first study the joint queue-length distribution at several embedded epochs in Section 5.3.1. We use these results to determine the LST of the batch sojourn-time distribution for both a specific and an arbitrary batch of arriving customers in Section 5.3.2, and present a Mean Value Analysis (MVA) to calculate the mean batch sojourn-time in Section 5.3.3.

5.3.1 The joint queue-length distribution

In the polling literature, the probability generating function (PGF) of the joint queue-length distribution at various epochs is extensively studied (e.g. Takagi (1986); Kleinrock & Levy (1988); Levy & Sidi (1990)). Let $\widetilde{LB}^{(V_i)}(\boldsymbol{z})$ and $\widetilde{LC}^{(V_i)}(\boldsymbol{z})$ be the joint queue-length PGF at *visit* beginnings and completions at Q_i, where $\boldsymbol{z} = (z_1, \ldots, z_N)$ is an N-dimensional vector with $|z_i| \leq 1$. Similarly, let $\widetilde{LB}^{(S_i)}(\boldsymbol{z})$ and $\widetilde{LC}^{(S_i)}(\boldsymbol{z})$ be the joint queue-length PGFs at *switch-over* beginnings and completions at Q_i, respectively. Because of the branching property (Resing, 1993), these PGFs can be related to each other as follows,

$$\begin{aligned}
\widetilde{LC}^{(V_i)}(\boldsymbol{z}) =& \widetilde{LB}^{(V_i)}(z_1, \ldots, z_{i-1}, \\
& \widetilde{BP}_i\left(\lambda - \lambda\widetilde{K}(z_1, \ldots, z_{i-1}, 1, z_{i+1}, \ldots, z_N)\right), z_{i+1}, \ldots, z_N\Big), \qquad (5.2)\\
\widetilde{LB}^{(S_i)}(\boldsymbol{z}) =& \widetilde{LC}^{(V_i)}(\boldsymbol{z}), \qquad (5.3)\\
\widetilde{LC}^{(S_i)}(\boldsymbol{z}) =& \widetilde{LB}^{(S_i)}(\boldsymbol{z})\,\widetilde{S}_i\left(\lambda - \lambda\widetilde{K}(\boldsymbol{z})\right), \qquad (5.4)\\
\widetilde{LB}^{(V_{i+1})}(\boldsymbol{z}) =& \widetilde{LC}^{(S_i)}(\boldsymbol{z}), \qquad (5.5)
\end{aligned}$$

where $i = 1, \ldots, N$ and $\widetilde{BP}_i(.)$ is the LST of a busy-period in Q_i equals that of an $M^X/G/1$ queue initiated by the service of a customer and is given by,

$$\widetilde{BP}_i(\omega) = \widetilde{B}_i\left(\omega + \lambda - \lambda\widetilde{K}_i\left(\widetilde{BP}_i(\omega)\right)\right). \qquad (5.6)$$

Equations (5.2)-(5.5) are referred in the polling literature as the *laws of motion*. The interpretation of (5.2) is that the queue-length in Q_j, $j \neq i$ at the end of visit period V_i is given by the number of customers already at Q_j at the visit beginning plus all the customers that arrive in the system during visit period V_i. For Q_i, all customers that are already in Q_i or arrive during V_i will be served before the end of the visit completion, and, therefore, Q_i will contain no customers at the end of the visit period. Equation (5.3) simply states that the PGF of a visit completion corresponds to the PGF of the next switch-over beginning (see also Figure 5.2). Finally, the queue-length vector at a switch-over completion corresponds to the sum of customers already present at the switch-over beginning plus all the customers that arrive during this switch-over period (5.4), and by definition the queue-length

vector at a switch-over completion is the same for the next visit beginning (5.5). Note that equations (5.2)-(5.5) can be differentiated with respect to $z_1, \ldots, z_N$ to compute moments of the queue-length distributions on embedded points (Levy & Sidi, 1991) or numerically inverted for the queue-length probability distributions (e.g. Choudhury & Whitt (1996) for the case for non-simultaneously arrivals).

Let $\widetilde{LB}^{(B_i)}(\boldsymbol{z})$ and $\widetilde{LC}^{(B_i)}(\boldsymbol{z})$ be the joint queue-length PGFs at *service* beginnings and completions at Q_i. Eisenberg (1972) proved, that besides the laws of motion, there exists a simple relation between the joint queue-length distributions at *visit*- and *service* beginnings and completions. He observed that each visit beginning either starts with a service beginning, or with a visit completion in case there are no customers at the queue. Similarly, each visit completion coincides with either a visit beginning or a service completion. Eisenberg (1972) only considered polling systems either with exhaustive or gated service at all queues and individual arriving customers, but Boxma et al. (2011) has proven that the relation is not restricted to a particular service discipline and also holds for general branching-type service disciplines. In this section, we generalize this result for the case of simultaneous batch arrivals. Similar as in Eisenberg (1972), the four PGFs are related as follows,

$$\widetilde{LB}^{(V_i)}(\boldsymbol{z}) + \lambda_i E(C)\,\widetilde{LC}^{(B_i)}(\boldsymbol{z}) = \lambda_i E(C)\,\widetilde{LB}^{(B_i)}(\boldsymbol{z}) + \widetilde{LC}^{(V_i)}(\boldsymbol{z}), \tag{5.7}$$

where the term $1/(\lambda_i E(C))$ is the long-run ratio between the number of service beginnings/completions and visit beginnings/completions in Q_i, for every $i = 1, \ldots, N$.

Furthermore, the joint queue-length distribution at service beginnings and completions are related via,

$$\widetilde{LC}^{(B_i)}(\boldsymbol{z}) = \widetilde{LB}^{(B_i)}(\boldsymbol{z}) \left[\widetilde{B}_i\left(\lambda - \lambda\widetilde{K}(\boldsymbol{z})\right)/z_i\right]. \tag{5.8}$$

Substituting (5.8) in (5.7) and rearranging terms, the joint queue-length distribution at a service beginning can be written as,

$$\widetilde{LB}^{(B_i)}(\boldsymbol{z}) = \frac{z_i\left(\widetilde{LC}^{(V_i)}(\boldsymbol{z}) - \widetilde{LB}^{(V_i)}(\boldsymbol{z})\right)}{\lambda_i E(C)\left(\widetilde{B}_i\left(\lambda - \lambda\widetilde{K}(\boldsymbol{z})\right) - z_i\right)}. \tag{5.9}$$

Next, we can find the PGFs of the joint queue-length distributions at an arbitrary moment during V_i and S_i, denoted by $\tilde{L}^{(V_i)}(\boldsymbol{z})$ and $\tilde{L}^{(S_i)}(\boldsymbol{z})$, by noticing that the queue-length at an arbitrary moment in V_i or S_i is equal to the queue length at service/switch-over beginning plus the number of customers that arrived in the past service/switch-over time,

$$\tilde{L}^{(V_i)}(\boldsymbol{z}) = \widetilde{LB}^{(B_i)}(\boldsymbol{z}) \frac{1-\widetilde{B}_i\left(\lambda-\lambda\widetilde{K}(\boldsymbol{z})\right)}{E(B_i)\left(\lambda-\lambda\widetilde{K}(\boldsymbol{z})\right)}, \tag{5.10}$$

$$\tilde{L}^{(S_i)}(\boldsymbol{z}) = \widetilde{LB}^{(S_i)}(\boldsymbol{z}) \frac{1-\widetilde{S}_i\left(\lambda-\lambda\widetilde{K}(\boldsymbol{z})\right)}{E(S_i)\left(\lambda-\lambda\widetilde{K}(\boldsymbol{z})\right)}. \tag{5.11}$$

Finally, let $\tilde{L}(\boldsymbol{z})$ be the PGF of the joint queue-length distribution at an arbitrary moment. By conditioning on periods $V_1, S_1, \ldots, V_N, S_N$ and using (5.10) and (5.11) $\tilde{L}(\boldsymbol{z})$ can be written as,

$$\tilde{L}(\boldsymbol{z}) = \frac{1}{E(C)} \sum_{i=1}^{N} \left(E(V_i)\, \tilde{L}^{(V_i)}(\boldsymbol{z}) + E(S_i)\, \tilde{L}^{(S_i)}(\boldsymbol{z}) \right), \tag{5.12}$$

with $E(V_i) = \rho_i E(C)$ as the expected visit time to Q_i.

The conditioning approach of Equation (5.12) will also be used in the next section to determine the batch sojourn-time distribution. The next theorem will show how (5.12) can be reformulated and used to find the marginal queue-length distributions.

Theorem 5.1. *Let $\tilde{L}(\boldsymbol{z})$ be the probability generating function of the joint queue-length distribution at an arbitrary time in steady-state. Then, $\tilde{L}(\boldsymbol{z})$ can be written as follows,*

$$\tilde{L}(\boldsymbol{z}) = \sum_{i=1}^{N} \frac{\lambda_i (1-z_i)\, \widetilde{LC}^{(B_i)}(\boldsymbol{z})}{\lambda - \lambda\widetilde{K}(\boldsymbol{z})}. \tag{5.13}$$

Proof. First, we start by rewriting (5.10) and (5.11). Equation (5.10) can be rewritten using (5.9). Hence,

$$\tilde{L}^{(V_i)}(\boldsymbol{z}) = \frac{z_i\left(\widetilde{LC}^{(V_i)}(\boldsymbol{z}) - \widetilde{LB}^{(V_i)}(\boldsymbol{z})\right)}{\lambda_i E(C)\left(\widetilde{B}_i\left(\lambda-\lambda\widetilde{K}(\boldsymbol{z})\right) - z_i\right)} \frac{1-\widetilde{B}_i\left(\lambda-\lambda\widetilde{K}(\boldsymbol{z})\right)}{E(B_i)\left(\lambda-\lambda\widetilde{K}(\boldsymbol{z})\right)}. \tag{5.14}$$

Similarly, (5.11) can be rewritten using (5.3)-(5.5),

$$\tilde{L}^{(S_i)}(\boldsymbol{z}) = \frac{\widetilde{LC}^{(V_i)}(\boldsymbol{z}) - \widetilde{LB}^{(V_{i+1})}(\boldsymbol{z})}{E(S_i)\left(\lambda - \lambda\widetilde{K}(\boldsymbol{z})\right)}. \tag{5.15}$$

Substituting (5.14) and (5.15) into (5.12) gives

$$\tilde{L}(\boldsymbol{z}) = \frac{1}{E(C)}\sum_{i=1}^{N}\left[\frac{z_i\left(\widetilde{LC}^{(V_i)}(\boldsymbol{z}) - \widetilde{LB}^{(V_i)}(\boldsymbol{z})\right)}{\widetilde{B}_i\left(\lambda - \lambda\widetilde{K}(\boldsymbol{z})\right) - z_i}\frac{1 - \widetilde{B}_i\left(\lambda - \lambda\widetilde{K}(\boldsymbol{z})\right)}{\lambda - \lambda\widetilde{K}(\boldsymbol{z})} + \frac{\widetilde{LC}^{(V_i)}(\boldsymbol{z}) - \widetilde{LB}^{(V_{i+1})}(\boldsymbol{z})}{\lambda - \lambda\widetilde{K}(\boldsymbol{z})}\right]. \tag{5.16}$$

Next, (5.16) can be rewritten into (5.13) as follows. First, by rearrangement it holds that,

$$\sum_{i=1}^{N}\frac{\widetilde{LC}^{(V_i)}(\boldsymbol{z}) - \widetilde{LB}^{(V_{i+1})}(\boldsymbol{z})}{\left(\lambda - \lambda\widetilde{K}(\boldsymbol{z})\right)} = \sum_{i=1}^{N}\frac{\widetilde{LC}^{(V_i)}(\boldsymbol{z}) - \widetilde{LB}^{(V_i)}(\boldsymbol{z})}{\left(\lambda - \lambda\widetilde{K}(\boldsymbol{z})\right)}.$$

Then, using (5.16), (5.8), and (5.9),

$$\begin{aligned}
&\sum_{i=1}^{N}\left[\frac{z_i\left(\widetilde{LC}^{(V_i)}(\boldsymbol{z}) - \widetilde{LB}^{(V_i)}(\boldsymbol{z})\right)}{\widetilde{B}_i\left(\lambda - \lambda\widetilde{K}(\boldsymbol{z})\right) - z_i}\frac{1 - \widetilde{B}_i\left(\lambda - \lambda\widetilde{K}(\boldsymbol{z})\right)}{\lambda - \lambda\widetilde{K}(\boldsymbol{z})} + \frac{\widetilde{LC}^{(V_i)}(\boldsymbol{z}) - \widetilde{LB}^{(V_{i+1})}(\boldsymbol{z})}{\lambda - \lambda\widetilde{K}(\boldsymbol{z})}\right] \\
&= \sum_{i=1}^{N}\left(1 + \frac{z_i\left(1 - \widetilde{B}_i\left(\lambda - \lambda\widetilde{K}(\boldsymbol{z})\right)\right)}{\widetilde{B}_i\left(\lambda - \lambda\widetilde{K}(\boldsymbol{z})\right) - z_i}\right)\frac{\left(\widetilde{LC}^{(V_i)}(\boldsymbol{z}) - \widetilde{LB}^{(V_i)}(\boldsymbol{z})\right)}{\lambda - \lambda\widetilde{K}(\boldsymbol{z})} \\
&= \sum_{i=1}^{N}\frac{(1 - z_i)\,\widetilde{B}_i\left(\lambda - \lambda\widetilde{K}(\boldsymbol{z})\right)}{\widetilde{B}_i\left(\lambda - \lambda\widetilde{K}(\boldsymbol{z})\right) - z_i}\frac{\left(\widetilde{LC}^{(V_i)}(\boldsymbol{z}) - \widetilde{LB}^{(V_i)}(\boldsymbol{z})\right)}{\lambda - \lambda\widetilde{K}(\boldsymbol{z})} \\
&= \sum_{i=1}^{N}\frac{(1 - z_i)\,\lambda_i E(C)\,\widetilde{LC}^{(B_i)}(\boldsymbol{z})}{\lambda - \lambda\widetilde{K}(\boldsymbol{z})},
\end{aligned}$$

and multiplying with $1/E(C)$ gives (5.13). □

Remark 5.1. Boxma et al. (2011) derived $\tilde{L}(\boldsymbol{z})$ for a polling system with individually arriving customers. In case of individually arriving customers, $\lambda - \lambda\widetilde{K}(\boldsymbol{z})$ reduces to $\sum_{i=1}^{N} \lambda_i (1 - z_i)$ in (5.13), which corresponds with Equation (10) in Boxma et al. (2011).

Remark 5.2. From Theorem 5.1 the marginal queue-length distributions $\widetilde{L}_i(z)$ immediately follows by setting $z_i = z$ and $z_j = 1$, for $j \neq i$. Then, from (5.13),

$$\begin{aligned}\tilde{L}_i(z) &= \frac{\lambda_i (1-z)\, \widetilde{LC}_i^{(B_i)}(z)}{\lambda - \lambda \widetilde{K}_i(z)} \\ &= \frac{E(K_i)(1-z)}{1 - \widetilde{K}_i(z)} \widetilde{LC}_i^{(B_i)}(z). \end{aligned} \tag{5.17}$$

where $\widetilde{LC}_i^{(B_i)}(z) = \widetilde{LC}^{(B_i)}(1, \ldots, 1, z, 1, \ldots, 1)$, where the z occurs at the i-th entry.

Remark 5.3. When $N = 1$, the system reduces to a $M^X/G/1$ queueing system with multiple vacations (Baba, 1986). Batches of customers arrive at the system according to a compound Poisson process. As soon as the system becomes empty, the server takes an uninterruptible vacation (switch-over time) for a random length of time. After returning from that vacation, the server keeps on taking vacations until there is at least one customer in the system. With use of (5.8), (5.9), and (5.13) it is easy to determine the PGF of the stationary queue size distribution of the $M^X/G/1$ multiple vacation model,

$$\begin{aligned}\tilde{L}(z) &= \frac{\lambda E(K)(1-z)\, \widetilde{LC}^{(B)}(z)}{\lambda - \lambda \widetilde{K}(z)} \\ &= \frac{(1-\rho)(1-z)\, \widetilde{B}\left(\lambda - \lambda\widetilde{K}(z)\right)\left(\widetilde{LC}^{(V)}(z) - \widetilde{LB}^{(V)}(z)\right)}{\left(\lambda - \lambda\widetilde{K}(z)\right) E(S) \left(\widetilde{B}\left(\lambda - \lambda\widetilde{K}(z)\right) - z\right)} \\ &= \left[\frac{(1-\rho)(1-z)\, \widetilde{B}\left(\lambda - \lambda\widetilde{K}(z)\right)}{\widetilde{B}\left(\lambda - \lambda\widetilde{K}(z)\right) - z}\right]\left[\frac{1 - \widetilde{S}\left(\lambda - \lambda\widetilde{K}(z)\right)}{E(S)\left(\lambda - \lambda\widetilde{K}(z)\right)}\right], \end{aligned} \tag{5.18}$$

where we use the fact that $\widetilde{LC}^{(V)}(z) = 1$, since the server only goes on a vacation if the queue is empty. Equation (5.18) can be interpreted as follows. The first term is the PGF of the stationary queue-length distribution of the standard $M^X/G/1$ queue

without vacations, whereas the second term is the PGF of the number of customers that arrive during the residual duration of the vacation time (Choudhury, 2002).

5.3.2 Batch sojourn-time distribution

In order to determine the LST of the steady-state batch sojourn-time distribution, we follow the method of Boon et al. (2012) by conditioning on the location of the server and determining the time it takes until the last customer in a specific batch is served. These results are then used to determine the batch sojourn-time distribution of an arbitrary batch. Boon et al. (2012) developed this method to study the steady-state waiting time distribution for polling systems with rerouting. For these kinds of models, the distributional form of Little's Law (Keilson & Servi, 1988) cannot be applied, since the combined processes of internal and external arrivals do not necessary form a Poisson process. However, by studying the evolution of the system after a customer arrival this problem can be avoided and the waiting time distribution can be obtained. Important in their analysis is the concept of *descendants* from the theory of branching processes, which is defined as all the customers who arrive during the service of a tagged customer, plus the customers who arrive during the service of those customers, etc (i.e. the total progeny of the tagged customer).

The approach of Boon et al. (2012) is very suited to determine the steady-state batch sojourn-time distribution, since for a specific customer batch the location where the last customer in the batch will be served varies on the location of the server at the arrival of the batch (e.g. in Figure 5.3 depending of the location of the server the batch is either fully served in Q_1 or Q_i). Similar as in (5.12) we explicitly condition on the location of the server; the LST of the batch sojourn-time distribution of a specific customer batch $\boldsymbol{k}$ can be written as,

$$\widetilde{T}_{\boldsymbol{k}}(\omega) = \frac{1}{E(C)} \sum_{j=1}^{N} \left(E(V_j)\, \widetilde{T}_{\boldsymbol{k}}^{(V_j)}(\omega) + E(S_j)\, \widetilde{T}_{\boldsymbol{k}}^{(S_j)}(\omega) \right), \tag{5.19}$$

where $\widetilde{T}_{\boldsymbol{k}}^{(V_j)}(.)$ is the LST of the batch sojourn-time for customer batch $\boldsymbol{k}$ *given* that the batch arrived during V_j, and whereas $\widetilde{T}_{\boldsymbol{k}}^{(S_j)}(.)$ is *given* when the customer batch arrived during S_j. The remainder of this section will focus on how to determine $\widetilde{T}_{\boldsymbol{k}}^{(V_j)}(.)$, $\widetilde{T}_{\boldsymbol{k}}^{(S_j)}(.)$, and the LST of an arbitrary batch $\widetilde{T}(.)$.

From the theory of branching processes, we denote $B_{j,i}$ $i, j = 1, \ldots, N$, as the service of a tagged customer in Q_j plus all its descendants that will be served before or during the next visit to Q_i. Combining this gives the following recursive function,

$$B_{j,i} = \begin{cases} BP_j, & \text{if } i = j, \\ BP_j + \sum_{l=j+1}^{i}{}' \sum_{m=1}^{N_l(BP_j)} B_{l_m,i}, & \text{otherwise,} \end{cases} \tag{5.20}$$

where BP_j is the busy period initiated by the tagged customer in Q_j, $N_l\left(BP_j\right)$ denotes the number of customers that arrive in Q_l during this busy-period in Q_j, and $B_{l_m,i}$ is a sequence of (independent) $B_{l,i}$'s. Let $\widetilde{B}_{j,i}\left(.\right)$ be the LST of $B_{j,i}$, which is given by,

$$\widetilde{B}_{j,i}\left(\omega\right) = \widetilde{BP}_j\left(\omega + \lambda(1 - \widetilde{K}(\boldsymbol{B_{j+1,i}}))\right), \tag{5.21}$$

where $\boldsymbol{B_{j+1,i}}$ is an N-dimensional vector defined as follows,

$$(\boldsymbol{B_{j,i}})_l = \begin{cases} \widetilde{B}_{l,i}\left(\omega\right), & \text{if } l = j, \ldots, i, \text{ and } j \neq i+1, \\ 1, & \text{otherwise.} \end{cases} \tag{5.22}$$

A similar LST can also be formulated for a switch-over time S_j and the service of all its descendants that will be served before the end of the visit to S_i,

$$\widetilde{S}_{j,i}\left(\omega\right) = \widetilde{S}_j\left(\omega + \lambda(1 - \widetilde{K}(\boldsymbol{B_{j+1,i}})\right), \tag{5.23}$$

Finally, let $\boldsymbol{B^*_{j,i}}$ be an N-dimensional vector defined as,

$$(\boldsymbol{B^*_{j,i}})_l = \begin{cases} \widetilde{B}_i\left(\omega\right), & \text{if } l = i, \\ (\boldsymbol{B_{j,i-1}})_l, & \text{otherwise.} \end{cases} \tag{5.24}$$

The key difference with (5.22) is that (5.24) excludes any new customer arrivals in Q_i. This is needed to omit customers that arrive in Q_i after the batch arrival; these customers do not influence the batch sojourn-time of the arriving customer batch since they will be served afterwards.

We first focus on the batch sojourn-time of a customer batch that arrives during a visit period. Assume than an arriving customer batch $\boldsymbol{k}$ enters the system while the server is currently within visit period V_j and the last customer in the batch will be served in Q_i. Formally, this means $k_i > 0$ and all the other customer arriving in the same batch should be served before the next visit to Q_i; $k_l \geq 0$, $l = j, \ldots, i-1$, and $k_l = 0$ elsewhere. Whenever all the customers arrive in the same queue that is currently visited; then $k_i = k_j > 0$, and $k_l = 0$ elsewhere.

The batch sojourn-time of customer batch $\boldsymbol{k}$ consists of the (i) residual service time in Q_j, (ii) the service of all the customers already in the system in $Q_j, \ldots, Q_i$, (iii) the service of all new customer arrivals that arrive after customer batch $\boldsymbol{k}$ in $Q_j, \ldots, Q_{i-1}$ before the server reaches Q_i, (iv) switch-over times $S_j, \ldots, S_{i-1}$, and (v) the service of the customers in the customer batch $\boldsymbol{k}$. From (5.10) we know that at the arrival of the customer batch, the PGF of the joint queue-length distribution is the equal to the queue lengths at a service beginning, $\widetilde{LB}^{(B_j)}(.)$, plus the number of customers that arrived in the past part of the service time, $\widetilde{B}_j^P(.)$. On the other hand, we also need to consider the residual part of the service time, $\widetilde{B}_j^R(.)$, and if $i \neq j$ the arrivals that occur in $Q_j, \ldots, Q_{i-1}$ during this period as well. Therefore similar as in Boon et al. (2012), we need to consider the PGF-LST of the joint queue-length distribution at an arrival epoch *and* the residual service time; $\widetilde{L}^{(V_i)}(\boldsymbol{z}, \omega)$. First, since the number of customers that arrive in the past and residual part of the service time are independent of each other and from the queue lengths at a service beginning, we can write the LST of the joint distribution of $\widetilde{B}_j^P(.)$ and $\widetilde{B}_j^R(.)$ as (Cohen, 1982)

$$\widetilde{B}_j^{PR}(\omega_P, \omega_R) = \frac{\widetilde{B}_j(\omega_P) - \widetilde{B}_j(\omega_R)}{E(B_j)(\omega_R - \omega_P)},$$

Then because of independence between $\widetilde{B}_j^{PR}(\omega_P, \omega_R)$ and $\widetilde{LB}^{(B_j)}(\boldsymbol{z})$, we have

$$\widetilde{L}^{(V_j)}(\boldsymbol{z}, \omega) = \widetilde{LB}^{(B_j)}(\boldsymbol{z})\, \widetilde{B}_j^{PR}\left(\lambda - \lambda \widetilde{K}(\boldsymbol{z}), \omega\right). \tag{5.25}$$

Proposition 5.1. *The LST of the batch sojourn-time distribution of batch* $\boldsymbol{k}$ *conditioned that the server is in visit period* V_j *and the last customer in the batch will be served in* Q_i *is given by,*

$$\widetilde{T}_{\boldsymbol{k}}^{(V_j)}(\omega) = \widetilde{L}^{(V_j)}\left(\boldsymbol{B}^*_{\boldsymbol{j,i}},\, \omega + \lambda(1 - \widetilde{K}(\boldsymbol{B}_{\boldsymbol{j,i-1}}))\right) \times \prod_{l=1}^{i-j}{}' \widetilde{S}_{j+l-1,i-1}(\omega) \frac{1}{(\boldsymbol{B}^*_{\boldsymbol{j,i}})_j} \prod_{l=j}^{i}{}'(\boldsymbol{B}^*_{\boldsymbol{j,i}})_l^{k_l}. \quad (5.26)$$

Proof. Consider the system just before the arrival of the customer batch and assume that the batch does not finish service in the current visit period, i.e. $i \neq j$. Then, let $n_1, n_2, \ldots, n_N$ be the number of customers present in the system at the arrival epoch of the customer batch and $k_1, \ldots, k_N$ be the number of customers per queue that arrived in batch $\boldsymbol{k}$. Since the batch arrives in V_j, it first has to wait for the residual service time of the customer currently in service. During this period, new customers can arrive before the next visit to Q_i which bring in additional work with $\lambda(1 - \widetilde{K}(\boldsymbol{B}_{\boldsymbol{j,i-1}}))$. Afterwards, each customer already in the system at the arrival of the customer batch in $Q_j, \ldots, Q_i$ and each customer in batch $\boldsymbol{k}$ will make a contribution of $(\boldsymbol{B}^*_{\boldsymbol{j,i}})_l$, $l = j, \ldots, i$ to the batch sojourn-time. Finally, in the switch-over periods between Q_j and Q_i, new customers can arrive that will be served before the service of the last customer in the batch. Combining this, gives the LST of the batch sojourn-time distribution of batch $\boldsymbol{k}$ conditioned that $n_1, n_2, \ldots, n_N$ customers are already present in the system, the server is in visit period V_j, and the last customer in the batch will be served in Q_i,

$$E(e^{-\omega T_{\boldsymbol{k}}^{(V_j)}} | n_1, n_2, \ldots, n_N) = \widetilde{B}_j^R\left(\omega + \lambda(1 - \widetilde{K}(\boldsymbol{B}_{\boldsymbol{j,i-1}}))\right) \widetilde{B}_{j,i-1}(\omega)^{n_j - 1 + k_j} \times \prod_{l=j+1}^{i-1}{}' \widetilde{B}_{l,i-1}(\omega)^{n_l + k_l} \prod_{l=j}^{i-1}{}' \widetilde{S}_{l,i-1}(\omega)\, \widetilde{B}_i(\omega)^{n_i + k_i}. \quad (5.27)$$

Unconditioning this equation gives (5.26). □

Now, consider a customer batch that arrives during a switch-over period. Assume an arriving customer batch $\boldsymbol{k}$ enters the system while the server is currently within switch-over period S_{j-1} and the last customer in the batch will be served in Q_i. The

reason that we consider S_{j-1} is that batch $\boldsymbol{k}$ will finish service in the same queue had it arrived in V_j because of the exhaustive service discipline.

In this case, the batch sojourn-time consists of the same components (ii), (iii), (iv), and (v) on page 126. Component (i) is however different and is now defined as the residual switch-over time between Q_{j-1} and Q_j. Similarly, we define $\widetilde{L}^{(S_{j-1})}(\boldsymbol{z},\omega)$ as the PGF-LST of the joint queue-length distribution of customers present in the system at an arbitrary moment during S_{j-1} *and* the residual switch-over time $\widetilde{S}^{R}_{j-1}(.)$. From (5.11) we have the joint queue-length distribution at a switch-over beginning, $\widetilde{LB}^{(S_{j-1})}(.)$, and the number of customers that arrived in the past part of the switch-over time, $\widetilde{S}^{P}_{j-1}(.)$. Similar to $\widetilde{B}^{PR}_{j}(.)$, we define $\widetilde{S}^{PR}_{j-1}(\omega_R,\omega_P)$ as the LST of the joint distribution of the past and residual switch-over time S_{j-1} as

$$\widetilde{S}^{PR}_{j-1}(\omega_P,\omega_R)=\frac{\widetilde{S}_{j-1}(\omega_P)-\widetilde{S}_{j-1}(\omega_R)}{E(S_{j-1})(\omega_R-\omega_P)}.$$

Then due to independence, the PGF-LST of the joint queue-length distribution present at an arbitrary moment during S_{j-1} and the residual switch-over time is given by,

$$\widetilde{L}^{(S_{j-1})}(\boldsymbol{z},\omega)=\widetilde{LB}^{(S_{j-1})}(\boldsymbol{z})\,\widetilde{S}^{PR}_{j-1}\left(\lambda-\lambda\widetilde{K}(\boldsymbol{z}),\omega\right). \tag{5.28}$$

Proposition 5.2. *The LST of the batch sojourn-time distribution of batch* $\boldsymbol{k}$ *conditioned that the server is in switch-over period* S_{j-1} *and the last customer in the batch will be served in* Q_i *is given by,*

$$\widetilde{T}^{(S_{j-1})}_{\boldsymbol{k}}(\omega)=\widetilde{L}^{(S_{j-1})}\left(\boldsymbol{B}^{*}_{\boldsymbol{j,i}},\,\omega+\lambda(1-\widetilde{K}(\boldsymbol{B}_{\boldsymbol{j,i-1}}))\right)\times\prod_{l=1}^{i-j}{}'\,\widetilde{S}_{j+l-1,i-1}(\omega)\prod_{l=j}^{i}{}'(\boldsymbol{B}^{*}_{\boldsymbol{j,i}})^{k_l}_{l}. \tag{5.29}$$

Proof. Similar as in Proposition 5.1, let $n_1,n_2,\ldots,n_N$ be the number of customers present in the system at the arrival epoch, $k_1,\ldots,k_N$ be the number of customers per queue in batch $\boldsymbol{k}$, and $i\neq j$. Then, the first component of the batch sojourn-time is the residual switch-over time in S_{j-1} and the contribution of the arrival of potential new customers before the next visit to Q_i with $\lambda(1-\widetilde{K}(\boldsymbol{B}_{\boldsymbol{j,i-1}}))$. Afterwards, each customer in $Q_j,\ldots,Q_i$ already in the system at the arrival of the customer batch

and each customer in batch $\boldsymbol{k}$ will make a contribution of $(\boldsymbol{B}^*_{\boldsymbol{j,i}})_l$, $l = j, \ldots, i$ to the batch sojourn-time. Finally, in the switch-over periods between Q_j and Q_i, new customers can arrive that will be served before the service of the last customer in the batch. Combining this, gives the LST of the batch sojourn-time distribution of batch $\boldsymbol{k}$ conditioned that $n_1, n_2, \ldots, n_N$ customers are already present in the system, the server is in visit period S_{j-1}, and the last customer in the batch will be served in Q_i,

$$E(e^{-\omega T_{\boldsymbol{k}}^{(S_{j-1})}} | n_1, n_2, \ldots, n_N) = \widetilde{S}^R_{j-1}\left(\omega + \lambda(1 - \widetilde{K}(\boldsymbol{B_{j,i-1}}))\right) \times \prod_{l=j}^{i-1}{}' \widetilde{B}_{l,i-1}(\omega)^{n_l+k_l} \prod_{l=j}^{i-1}{}' \widetilde{S}_{l,i-1}(\omega)\, \widetilde{B}_i(\omega)^{n_i+k_i}. \quad (5.30)$$

Unconditioning this equation gives (5.29). □

From Proposition 5.1 and Proposition 5.2, it can be seen that the LST of the batch sojourn-time distribution of batch $\boldsymbol{k}$ conditioned on a visit/switch-over period is comprised of two terms; a term independent of batch $\boldsymbol{k}$ *and* a term that corresponds to the additional contribution batch $\boldsymbol{k}$ makes to the batch sojourn-time;

$$\widetilde{T}_{\boldsymbol{k}}^{(V_j)}(\omega) = \sum_{i=1}^{N} 1_{(\boldsymbol{k}\in\mathcal{K}_{j,i})} \widetilde{W}_i^{(V_j)}(\omega) \prod_{l=j}^{i}{}' (\boldsymbol{B}^*_{\boldsymbol{j,i}})_l^{k_l}, \quad (5.31)$$

$$\widetilde{T}_{\boldsymbol{k}}^{(S_{j-1})}(\omega) = \sum_{i=1}^{N} 1_{(\boldsymbol{k}\in\mathcal{K}_{j,i})} \widetilde{W}_i^{(S_{j-1})}(\omega) \prod_{l=j}^{i}{}' (\boldsymbol{B}^*_{\boldsymbol{j,i}})_l^{k_l}, \quad (5.32)$$

where $1_{(\boldsymbol{k}\in\mathcal{K}_{j,i})}$ is an indicator function that is equal to one if for batch $\boldsymbol{k}$ all its customers are served in $Q_j, \ldots, Q_i$ and the last customer will be served in Q_i, and zero otherwise. The terms $\widetilde{W}_i^{(V_j)}(\omega)$ and $\widetilde{W}_i^{(S_{j-1})}(\omega)$ can be considered as the time between the batch arrival epoch and the service completion of the last customer in Q_i that was already in the system at the arrival of the customer batch, excluding batch $\boldsymbol{k}$ and any arrivals to Q_i after the arrival epoch, conditioned on the location of the server. In case there are only individually arriving customers this would correspond to the LST of the waiting time distribution of a customer arriving in Q_i conditioned that the server is in a visit or switch-over period.

The LST of the batch sojourn-time distribution of a specific customer batch $\boldsymbol{k}$ can now be calculated using (5.19), and alternatively using (5.31) and (5.32) by,

$$\widetilde{T}_{\boldsymbol{k}}(\omega) = \frac{1}{E(C)} \sum_{j=1}^{N} \sum_{i=1}^{N} 1_{(\boldsymbol{k} \in \mathcal{K}^{j,i})} \left(E(V_j)\, \widetilde{W}_i^{(V_j)}(\omega) + E(S_{j-1})\, \widetilde{W}_i^{(S_{j-1})}(\omega) \right) \times \prod_{l=j}^{i}{}' (\boldsymbol{B}^*_{\boldsymbol{j,i}})_l^{k_l}. \quad (5.33)$$

Finally, we focus on the LST of the batch sojourn-time of an arbitrary batch $\widetilde{T}(.)$.

Theorem 5.2. *The LST of the batch sojourn-time distribution of an arbitrary batch* $\widetilde{T}(\boldsymbol{.})$*, in case of exhaustive service, is given by:*

$$\widetilde{T}(\omega) = \sum_{\boldsymbol{k} \in \mathcal{K}} \pi(\boldsymbol{k})\, \widetilde{T}_{\boldsymbol{k}}(\omega), \quad (5.34)$$

where $\widetilde{T}_{\boldsymbol{k}}(\omega)$ *is given by* (5.19) *or* (5.33)*. Alternatively, we can write* (5.34) *as,*

$$\widetilde{T}(\omega) = \frac{1}{E(C)} \sum_{j=1}^{N} \sum_{i=1}^{N} \left(E(V_j)\, \widetilde{W}_i^{(V_j)}(\omega) + E(S_{j-1})\, \widetilde{W}_i^{(S_{j-1})}(\omega) \right) \times \pi(\mathcal{K}_{j,i})\, \widetilde{K}\left(\boldsymbol{B}^*_{\boldsymbol{j,i}} | \mathcal{K}_{j,i} \right). \quad (5.35)$$

Proof. It can be easily seen that (5.34) follows by enumerating all possible realizations of customer batches and the law of total probability.

Next for (5.35), we can partition $\mathcal{K}$ into $\mathcal{K}_{j,i}$ and write (5.34) using (5.19) as,

$$\widetilde{T}(\omega) = \frac{1}{E(C)} \sum_{i=1}^{N} \sum_{j=1}^{N} \sum_{\boldsymbol{k} \in \mathcal{K}_{j,i}} \pi(\boldsymbol{k}) \left(E(V_j)\, \widetilde{T}_{\boldsymbol{k}}^{(V_j)}(\omega) + E(S_{j-1})\, \widetilde{T}_{\boldsymbol{k}}^{(S_{j-1})}(\omega) \right). \quad (5.36)$$

From (5.31) and (5.32) it can be seen that when the server is either in S_{j-1} or V_j, then for two different customer batches that both finish service in the same queue their LST of the batch sojourn-time distribution only varies in the contribution the batch makes to the batch sojourn-time.

Then, by (5.33) and (5.1), we have by rearrangement

$$\begin{aligned}
&\sum_{\boldsymbol{k}\in\mathcal{K}_{j,i}} \pi\left(\boldsymbol{k}\right)\left(E\left(V_j\right)\widetilde{T}_{\boldsymbol{k}}^{(V_j)}\left(\omega\right)+E\left(S_{j-1}\right)\widetilde{T}_{\boldsymbol{k}}^{(S_{j-1})}\left(\omega\right)\right)\\
&=\left(E\left(V_j\right)\widetilde{W}_i^{(V_j)}\left(\omega\right)+E\left(S_{j-1}\right)\widetilde{W}_i^{(S_{j-1})}\left(\omega\right)\right)\pi\left(\mathcal{K}_{j,i}\right)\sum_{\boldsymbol{k}\in\mathcal{K}_{j,i}}\frac{\pi\left(\boldsymbol{k}\right)}{\pi\left(\mathcal{K}_{j,i}\right)}\prod_{l=j}^{i}{}'(\boldsymbol{B}_{\boldsymbol{j},\boldsymbol{i}}^{*})_l^{k_l}\\
&=\left(E\left(V_j\right)\widetilde{W}_i^{(V_j)}\left(\omega\right)+E\left(S_{j-1}\right)\widetilde{W}_i^{(S_{j-1})}\left(\omega\right)\right)\pi\left(\mathcal{K}_{j,i}\right)\widetilde{K}\left(\boldsymbol{B}_{\boldsymbol{j},\boldsymbol{i}}^{*}|\mathcal{K}_{j,i}\right).
\end{aligned}$$

Substituting the last equation in (5.36) gives (5.35). □

Differentiating (5.35) will give the mean batch sojourn-time, however in the next section an alternative, more efficient way to determine the mean batch sojourn-time is presented.

5.3.3 Mean batch sojourn-time

In this section, we derive the mean batch sojourn-time of a specific batch and an arbitrary batch using *Mean Value Analysis* (MVA). MVA for polling systems was developed by Winands et al. (2006) to study mean waiting times in systems with exhaustive, gated service, or mixed service. The main advantage of MVA is that it has a pure probabilistic interpretation and is based on standard queueing results, i.e., the Poisson arrivals see time averages (PASTA) property (Wolff, 1982) and Little's Law (Little, 1961). Furthermore, MVA evaluates the polling system at arbitrary time periods and not on embedded points such as visit beginnings, like in the buffer occupancy method (Takagi, 1986) and the descendant set approach (Konheim et al., 1994).

Central in the MVA of Winands et al. (2006) is the derivation of $E\left(\bar{L}_i^{(S_{j-1},V_j)}\right)$, the mean queue-length at Q_i (excluding the potential customer currently in service) at an arbitrary epoch within switch-over period S_{j-1} and visit period V_j;

$$\begin{aligned}
E\left(\bar{L}_i^{(S_{j-1},V_j)}\right)=\frac{E\left(S_{j-1}\right)}{E\left(S_{j-1}\right)+E\left(V_j\right)}E\left(\bar{L}_i^{(S_{j-1})}\right)\\
+\frac{E\left(V_j\right)}{E\left(S_{j-1}\right)+E\left(V_j\right)}E\left(\bar{L}_i^{(V_j)}\right),
\end{aligned}\quad(5.37)$$

where $E\left(\bar{L}_i^{(S_{j-1})}\right)$ and $E\left(\bar{L}_i^{(V_j)}\right)$ are the expected queue-length in Q_i during, respectively, a switch-over/visit period. Subsequently, with $E\left(\bar{L}_i^{(S_{j-1};V_j)}\right)$ the mean queue-length $E\left(\bar{L}_i\right)$ in Q_i can be determined,

$$E\left(\bar{L}_i\right) = \sum_{j=1}^{N} \frac{E\left(S_{j-1}\right) + E\left(V_j\right)}{E\left(C\right)} E\left(\bar{L}_i^{(S_{j-1},V_j)}\right), \quad i = 1, \ldots, N, \tag{5.38}$$

and by Little's law, also the mean waiting time $E\left(W_i\right)$ of a random customer in Q_i, which is defined as the time in steady-state from the customer's arrival until the start of his/her service.

For notation purposes we introduce θ_j as shorthand for the intervisit period (S_{j-1}, V_j); the expected duration of this period $E\left(\theta_j\right)$ is given by,

$$E\left(\theta_j\right) = E\left(S_{j-1}\right) + E\left(V_j\right), \quad j = 1, \ldots, N. \tag{5.39}$$

Notice that $\sum_{j=1}^{N} E\left(\theta_j\right) = E\left(C\right)$. In addition, we define $\theta_{j,i}$ as the duration of an intervisit period starting in θ_j and ending in θ_i, the expected duration of this period $E\left(\theta_{j,i}\right)$ is equal to,

$$E\left(\theta_{j,i}\right) = \sum_{l=j}^{i}{}' E\left(\theta_l\right), \quad i = 1, \ldots, N,\ j = 1, \ldots, N, \tag{5.40}$$

and where $E\left(\theta_{j,i}^R\right) = E\left(\theta_{j,i}^2\right) / 2E\left(\theta_{j,i}\right)$ is the mean residual duration of this period. However, $E\left(\theta_{j,i}^2\right)$ is unknown and not straightforward to derive directly. In the MVA, based on probabilistic arguments, $E\left(\theta_{j,i}^2\right)$ will be expressed in terms of $E\left(\bar{L}_i^{(\theta_j)}\right)$.

We denote $E\left(B_{j,i}\right)$ as the mean service of a customer in Q_j and all its descendants *before* the server starts serving Q_i. Let $E\left(B_{j,j}\right) = E\left(B_j\right)$ and $E\left(B_{j,j+1}\right) = E\left(B_j\right) / \left(1 - \rho_j\right)$ be the expected busy-period initiated by a customer in Q_j. Then, $E\left(B_{j,j+2}\right)$ equals the busy-period in Q_j plus all the customers that arrive during this busy period in Q_{j+1} and the busy periods that they trigger,

$$E\left(B_{j,j+2}\right) = \frac{E\left(B_j\right)}{1 - \rho_j}\left(1 + \frac{\rho_{j+1}}{1 - \rho_{j+1}}\right) = \frac{E\left(B_j\right)}{\left(1 - \rho_j\right)\left(1 - \rho_{j+1}\right)}.$$

In general we can write $E(B_{j,i})$ for $i \neq j$,

$$E(B_{j,i}) = \frac{E(B_j)}{\prod_{l=j}^{i-1}(1-\rho_l)}, \quad i = 1,\ldots,N,\ j = 1,\ldots,N. \tag{5.41}$$

Also, let $E(S_{j,i})$ is denoted by the switch-over in Q_j and the service of all the customers that arrive during $E(S_j)$ and their descendants *before* the server starts serving Q_i. Then $E(S_{j,j+1}) = E(S_j)$ and, in general, for $i \neq j+1$,

$$E(S_{j,i}) = \frac{E(S_j)}{\prod_{l=j+1}^{i-1}(1-\rho_l)}, \quad i = 1,\ldots,N,\ j = 1,\ldots,N. \tag{5.42}$$

Finally, $E\left(B_{j,i}^R\right)$ is the mean residual service of a customer in Q_j and all its descendants *before* the server starts serving Q_i and is given by replacing $E(B_j)$ by $E\left(B_j^R\right) = E\left(B_j^2\right)/2E(B_j)$ in $E(B_{j,i})$. In addition, $E\left(S_{j,i}^R\right)$ is defined as $E(S_{j,i})$ and by replacing $E(S_j)$ by $E\left(S_j^R\right) = E\left(S_j^2\right)/2E(S_j)$.

In the MVA a set of N^2 linear equations is derived for $E\left(\bar{L}_i\right)$ in terms of unknowns $E\left(\bar{L}_i^{(\theta_j)}\right)$. For this we have to consider the waiting time of an arbitrary customer and make use of the arrival relation and the PASTA property. Assume that an arbitrary customer enters the system in Q_i. The waiting time of the customer consists of (i) the service of $E\left(\bar{L}_i\right)$ customers already at Q_i upon its arrival to the system, (ii) the service of $E(K_{ii})/2E(K_i)$ customers that arrived in the same customer batch, but are placed before the arbitrary customer in Q_i, (iii) if the server is currently in intervisit period θ_i, then the arbitrary customer has to wait with probability ρ_i for the residual service time $E\left(B_i^R\right)$ and with probability $E(S_{i-1})/E(C)$ for the residual switch-over time $E\left(S_{i-1}^R\right)$. Finally, (iv) whenever the server is not in intervisit period θ_i, the arbitrary customer has to wait for the expected residual duration before the server returns at Q_i. Based on these components, the mean waiting time $E(W_i)$ of a customer in Q_i, $i = 1,\ldots,N$ is given by,

$$\begin{aligned} E(W_i) = {} & E\left(\bar{L}_i\right)E(B_i) + \frac{E(K_{ii})}{2E(K_i)}E(B_i) + \rho_i E\left(B_i^R\right) \\ & + \frac{E(S_{i-1})}{E(C)}E\left(S_{i-1}^R\right) + \left(1 - \frac{E(\theta_i)}{E(C)}\right)\left(E\left(\theta_{i+1,i-1}^R\right) + E(S_{i-1})\right). \end{aligned} \tag{5.43}$$

The next step to derive the equations is to relate unknowns $E\left(\theta^R_{i+1,i-1}\right)$ to $E\left(\bar{L}_i^{(\theta_j)}\right)$. Consider $E\left(\theta^R_{j,i}\right)$ the expected residual duration of an intervisit period starting in θ_j and ending in θ_i given that an arbitrary customer batch just entered the system. Then with probability $E(\theta_l)/E(\theta_{j,i})$, the server is during this period in intervisit period θ_l, $l = j, \ldots, i$, and the expected residual duration until the intervisit ending of θ_i, conditioned that the server is in intervisit period θ_l, is defined as follows. First, with probability $E(V_l)/E(\theta_l)$ the server is busy serving a customer in Q_l and with probability $E(S_{l-1})/E(\theta_l)$ the server is in switch-over period S_{l-1}. During the residual service/switch-over time new customers can arrive that will be served before the intervisit ending in θ_i, which equals $E\left(B^R_{l,i+1}\right)$ and $E\left(S^R_{l-1,i+1}\right)$ respectively. In addition, the expected number of customers in Q_n given the server is in θ_l, $E\left(\bar{L}_n^{(\theta_l)}\right)$, and the expected number of customers $E(K_{nl})/E(K_n)$ that arrived in Q_n in the arbitrary customer batch will increase the duration of $E\left(\theta^R_{j,i}\right)$ by $E(B_{n,i+1})$. Finally, the customer also has to wait for all the switch-over times $E(S_{n,i+1})$, $n = j, \ldots, i$ between Q_n to Q_{n+1} plus the customers that arrive during the switch-over times and their descendants that will be served before the end of $E\left(\theta^R_{j,i}\right)$. Combining this gives the following expression for $i \neq j-1$,

$$E\left(\theta^R_{j,i}\right) = {\sum_{l=j}^{i}}' \frac{E(\theta_l)}{E(\theta_{j,i})}\left(\frac{E(V_l)}{E(\theta_l)}E\left(B^R_{l,i+1}\right) + \frac{E(S_{l-1})}{E(\theta_l)}E\left(S^R_{l-1,i+1}\right) + {\sum_{n=l}^{i}}'\left[\frac{E(K_{nl})}{E(K_n)} + E\left(\bar{L}_n^{(\theta_l)}\right)\right]E(B_{n,i+1}) + {\sum_{n=1}^{i-l}}' E(S_{l+n-1,i+1})\right), \quad (5.44)$$

It is now possible to set up a set of N^2 linear equations. First, after the server has visited Q_i, there will be no customers present in the queue. Therefore, the number of customers in Q_i given an arbitrary moment in an intervisit period starting in θ_{i+1} and ending in θ_j equals the number of Poisson arrivals during the age of this period (Winands et al., 2006). Because the age is equal to the residual time in distribution, we have for $i = 1, \ldots, N$, $j = 1, \ldots, N$, and $i \neq j$,

$${\sum_{l=i+1}^{j}}' \frac{E(\theta_l)}{E(\theta_{i+1,j})}E\left(\bar{L}_i^{(\theta_l)}\right) = \lambda_i E\left(\theta^R_{i+1,j}\right). \quad (5.45)$$

Second, by (5.43) and using Little's Law, $\lambda_i E(W_i) = E\left(\bar{L}_i\right)$, into (5.38) gives, for $i = 1, 2 \ldots, N$,

$$\sum_{j=1}^{N} \frac{E(\theta_j)}{E(C)} E\left(\bar{L}_i^{(\theta_j)}\right) = \frac{\lambda_i}{1-\rho_i} \left(\frac{E(K_{ii})}{2E(K_i)} E(B_i) + \rho_i E\left(B_i^R\right) \right.$$
$$\left. \frac{E(S_{i-1})}{E(C)} E\left(S_{i-1}^R\right) + \left(1 - \frac{E(\theta_i)}{E(C)}\right) \left(E\left(\theta_{i+1,i-1}^R\right) + E(S_{i-1}) \right) \right). \quad (5.46)$$

With (5.45) and (5.46) a set of N^2 linear equations for unknowns $E\left(\bar{L}_i^{(\theta_j)}\right)$ are now defined. Solving the set of linear equations and by (5.38) and (5.43) will give the expected queue-lengths and waiting times.

In order to derive the mean batch sojourn-time $E(T_{\boldsymbol{k}})$ of customer batch $\boldsymbol{k}$, $E\left(\bar{L}_i^{(\theta_j)}\right)$ also plays an integral role. Similar as in (5.19), in order to calculate the expected batch sojourn-time distribution of a specific customer batch $\boldsymbol{k}$, we explicitly condition on the location on the server,

$$E(T_{\boldsymbol{k}}) = \frac{1}{E(C)} \sum_{j=1}^{N} E(\theta_j) E\left(T_{\boldsymbol{k}}^{(\theta_j)}\right), \quad (5.47)$$

where $E\left(T_{\boldsymbol{k}}^{(\theta_j)}\right)$ is the expected batch sojourn-time distribution of a specific customer batch $\boldsymbol{k}$ given that the server is in intervisit period θ_j. $E\left(T_{\boldsymbol{k}}^{(\theta_j)}\right)$ can be derived in a similar way as (5.44). First, if the last customer will be served in Q_i, then with probability $E(V_j)/E(\theta_j)$ and $E(S_{j-1})/E(\theta_j)$ the arriving customer batch has to wait for the residual service/switch-over time during which new customers can arrive that will be served before the visit beginning in Q_i. Note that the customers arriving at Q_i during these residual times will not affect the batch sojourn-time of batch $\boldsymbol{k}$ since they will be served after the last customer in the batch is served. Then each customer already in the system and in batch $\boldsymbol{k}$ in Q_l, $l = j, \ldots, i$ will make a contribution of $E(B_{l,i})$ to the batch sojourn-time. Finally, the batch also has to wait for all the switch-over times between Q_j to Q_i and all their descendants that will be

served before the server reaches Q_i. This gives the following expression,

$$E\left(T_{\boldsymbol{k}}^{(\theta_j)}\right) = \frac{E\left(V_j\right)}{E\left(\theta_j\right)} E\left(B_{j,i}^{R}\right) + \frac{E\left(S_{j-1}\right)}{E\left(\theta_j\right)} E\left(S_{j-1,i}^{R}\right) + {\sum_{l=j}^{i}}' E\left(\bar{L}_l^{(\theta_j)}\right) E\left(B_{l,i}\right) + {\sum_{l=j}^{i}}' k_l E\left(B_{l,i}\right) + {\sum_{n=1}^{i-j}}' E\left(S_{j+n-1,i}\right), \quad (5.48)$$

Note that the same decomposition as (5.31) and (5.32) also holds for the expected batch sojourn-time,

$$E\left(T_{\boldsymbol{k}}^{(\theta_j)}\right) = \sum_{i=1}^{N} 1_{(\boldsymbol{k}\in\mathcal{K}^{j,i})} \left[E\left(W_i^{(\theta_j)}\right) + {\sum_{l=j}^{i}}' k_l E\left(B_{l,i}\right) \right],$$

where $E\left(W_i^{(\theta_j)}\right)$ is the expected time between the batch arrival epoch and the service completion of the last customer in Q_i that is already in the system, excluding any arrivals to Q_i after the arrival epoch. The term ${\sum}'_{l=j}^{i} k_l E\left(B_{l,i}\right)$ can be interpreted as the total contribution batch $\boldsymbol{k}$ makes to the batch sojourn-time.

Finally, the expected batch sojourn-time of an arbitrary customer batch is obtained by multiplying $E\left(T_{\boldsymbol{k}}\right)$ with the probability that a particular batch $\boldsymbol{k}$ enters the system,

$$E\left(T\right) = \sum_{\boldsymbol{k}\in\mathcal{K}} \pi\left(\boldsymbol{k}\right) E\left(T_{\boldsymbol{k}}\right). \quad (5.49)$$

However if there are many different realizations of customer batches possible, (5.49) might not be computational feasible to consider, since for every $\boldsymbol{k}$ we have to determine the mean batch sojourn-time given that the server starts in intervisit period θ_j and ends in θ_i; in total there are then $|\mathcal{K}| \times N \times N$ combinations to consider, where $|\mathcal{K}|$ denotes the size of support $\mathcal{K}$. Instead, by using $E\left(K_l|\mathcal{K}_{j,i}\right)$ we can rewrite (5.49) as follows,

$$\begin{aligned} E\left(T\right) &= \frac{1}{E\left(C\right)} \sum_{j=1}^{N}\sum_{i=1}^{N}\sum_{\boldsymbol{k}\in\mathcal{K}_{j,i}} \pi\left(\boldsymbol{k}\right) E\left(\theta_j\right) E\left(T_{\boldsymbol{k}}^{(\theta_j)}\right) \\ &= \frac{1}{E\left(C\right)} \sum_{j=1}^{N}\sum_{i=1}^{N} E\left(\theta_j\right) \sum_{\boldsymbol{k}\in\mathcal{K}_{j,i}} \pi\left(\boldsymbol{k}\right) \left(E\left(W_i^{(\theta_j)}\right) + {\sum_{l=j}^{i}}' k_l E\left(B_{l,i}\right) \right) \end{aligned}$$

$$= \frac{1}{E(C)} \sum_{j=1}^{N} \sum_{i=1}^{N} E(\theta_j)\, \pi(\mathcal{K}_{j,i}) \left(E\left(W_i^{(\theta_j)}\right) + \sum_{l=j}^{i}{}' E(K_l|\mathcal{K}_{j,i})\, E(B_{l,i}) \right).$$

The advantage is that the number of combinations reduces to $N \times N$, and $\pi(\mathcal{K}_{j,i})$ can be determined in $|\mathcal{K}|$ steps.

5.4 Locally-gated service

In this section, we study batch sojourn-times in a polling system with locally-gated service. In Section 5.4.1 and Section 5.4.2 we will study the joint queue-length distribution and the LST of the batch sojourn-time distribution. Instead of providing a thorough analysis, we present the differences with the analysis of Section 5.3. Finally, in Section 5.4.3 a Mean Value Analysis (MVA) is presented to calculate the mean batch sojourn-time.

5.4.1 The joint queue-length distributions

Similar as in Section 5.3.1, we start by defining the laws of motions in case of locally-gated service. For this we distinguish between customers that are standing behind of the gate and those who are standing before the gate (Boon et al., 2012). Customers that are standing behind the gate will be served in the current cycle, whereas customers before the gate will only be served in the next cycle. Let $\widetilde{LB}^{(V_i)}(\boldsymbol{z})$, $\widetilde{LB}^{(S_i)}(\boldsymbol{z})$, $\widetilde{LC}^{(S_i)}(\boldsymbol{z})$, and $\widetilde{LC}^{(V_i)}(\boldsymbol{z})$ be the joint queue-length PGF at *visit/switch-over* beginnings and completions at Q_i, for $i = 1, \ldots, N$, where $\boldsymbol{z} = (z_1, \ldots, z_N, z_G)$ is an $N+1$ dimensional vector. The first N elements correspond with the number of customers that are standing behind gate Q_i, $i = 1, \ldots, N$, whereas element $N+1$, z_G, is used during visit periods to correspond with the number of customers that are currently standing before the gate at the queue that is currently being visited.

Then the law of motions for locally-gated service are as follows,

$$\begin{aligned}\widetilde{LC}^{(V_i)}(\boldsymbol{z}) =& \widetilde{LB}^{(V_i)}\left(z_1, \ldots, z_{i-1}, \widetilde{B}_i\left(\lambda - \lambda \widetilde{K}(z_1, \ldots, z_{i-1}, z_G, z_{i+1}, \ldots, z_N)\right),\right. \\ & \left. z_{i+1}, \ldots, z_N, z_G\right), \end{aligned} \tag{5.50}$$

$$\widetilde{LB}^{(S_i)}(\boldsymbol{z}) = \widetilde{LC}^{(V_i)}(z_1, \ldots, z_N, z_i), \tag{5.51}$$

$$\widetilde{LC}^{(S_i)}(\boldsymbol{z}) = \widetilde{LB}^{(S_i)}(\boldsymbol{z})\, \widetilde{S}_i\left(\lambda - \lambda \widetilde{K}(z_1, \ldots, z_{i-1}, z_i, z_{i+1}, \ldots, z_N)\right), \tag{5.52}$$

$$\widetilde{LB}^{(V_{i+1})}(\boldsymbol{z}) = \widetilde{LC}^{(S_i)}(\boldsymbol{z}), \tag{5.53}$$

Equation (5.50) states that the queue-length in Q_j, $j \neq i$ at the end of visit period V_i is composed of the number of customers already at Q_j at the visit beginning plus all the customers that arrived in the system during the current visit period. However for Q_i, only the customers that were standing behind the gate are served before the end of the visit completion; customers that arrived to Q_i during this visit period are placed before the gate and will be served during the next visit to Q_i. In (5.51) it can be seen that the PGF of a visit completion corresponds to the PGF of the next switch-over beginning, except that the customer standing before the gate in Q_i are now placed behind the gate. Finally, the interpretation of (5.52) and (5.53) is the same as for (5.4) and (5.5).

In order to define the PGF of the joint queue-length distribution, Eisenberg's relationship (5.7) is also valid for locally-gated service. However, the joint queue-length distribution at service beginnings and completions (5.8) should be modified to,

$$\widetilde{LC}^{(B_i)}(\boldsymbol{z}) = \widetilde{LB}^{(B_i)}(\boldsymbol{z}) \times \left[\widetilde{B}_i\left(\lambda - \lambda \widetilde{K}(z_1, \ldots, z_{i-1}, z_G, z_{i+1}, \ldots, z_N)\right)/z_i\right], \tag{5.54}$$

since during a service period in Q_i arriving customers who join Q_i are placed before the gate. A similar modification also applies for the PGF of the joint queue-length distributions at an arbitrary moment during V_i,

$$\tilde{L}^{(V_i)}(\boldsymbol{z}) = \widetilde{LB}^{(B_i)}(\boldsymbol{z})\, \frac{1 - \widetilde{B}_i\left(\lambda - \lambda \widetilde{K}(z_1, \ldots, z_{i-1}, z_G, z_{i+1}, \ldots, z_N)\right)}{E(B_i)\left(\lambda - \lambda \widetilde{K}(z_1, \ldots, z_{i-1}, z_G, z_{i+1}, \ldots, z_N)\right)}. \tag{5.55}$$

Then, all the other results from Section 5.3.1 can be easily modified for locally-gated service.

5.4.2 Batch sojourn-time distribution

In the following section we derive the LST of the steady-state batch sojourn-time distribution for locally-gated service. Assume an arriving customer batch $\boldsymbol{k}$ enters the system while the server is currently within visit period V_{j-1} or switch-over period S_{j-1} such that the last customer in the batch will be served in Q_i. This means $k_i > 0$ and all the other customers arriving in the same batch should be served before the next visit to Q_i; $k_l \geq 0$, $l = j, \ldots, i-1$, and $k_l = 0$ elsewhere. Whenever a customer arrives in the same queue that is currently being visited, then this customer will be placed before the gate. As a consequence, this customer will be served last in the batch since the server will visit first all the other queues before serving this customer.

Similar as for exhaustive service, let $B_{j,i}$ $i, j = 1, \ldots, N$, be the service of a tagged customer in Q_j plus all its descendants that will be served before or during the next visit to Q_i. Since during a service period in Q_j incoming customers to Q_j are placed before the gate, we have

$$B_{j,i} = \begin{cases} B_j & \text{if } i = j, \\ B_j + \sum_{l=j+1}^{i}{}' \sum_{m=1}^{N_l(B_j)} B_{l_m,i}, & \text{otherwise,} \end{cases} \tag{5.56}$$

where B_j is the service time of the tagged customer in Q_j, $N_l(B_j)$ denotes the number of customers that arrive in Q_l during the service time of the tagged customer in Q_j, and $B_{l_m,i}$ is a sequence of (independent) $B_{l,i}$'s. Let $\widetilde{B}_{j,i}(.)$ be the LST which is given by,

$$\widetilde{B}_{j,i}(\omega) = \widetilde{B}_j\left(\omega + \lambda(1 - \widetilde{K}(\boldsymbol{B_{j+1,i}}))\right), \tag{5.57}$$

where $\boldsymbol{B_{j+1,i}}$ is an N-dimensional vector similar defined as (5.22). We define $\boldsymbol{B^*_{j,i}}$ as an $N+1$-dimensional vector defined as follows,

$$(\boldsymbol{B^*_{j,i}})_l = \begin{cases} \widetilde{B}_i(\omega), & \text{if } l = i, \\ 1, & \text{if } l = N+1, \\ (\boldsymbol{B_{j,i-1}})_l, & \text{otherwise.} \end{cases} \tag{5.58}$$

Finally, let $\boldsymbol{B}^{\boldsymbol{G}}_{\boldsymbol{j,i}}$, $i,j = 1,\ldots,N$, be an $N+1$-dimensional vector defined as for $j \neq i$,

$$(\boldsymbol{B}^{\boldsymbol{G}}_{\boldsymbol{j,i}})_l = \begin{cases} (\boldsymbol{B}_{\boldsymbol{j,i}})_l & \text{if } l = j,\ldots,i, \\ 1, & \text{otherwise,} \end{cases} \tag{5.59}$$

and for $j = i$,

$$\begin{aligned} \boldsymbol{B}^{\boldsymbol{G}}_{\boldsymbol{i,i}} = \Big(&\widetilde{B}_{1,i-1}(\omega),\ldots,\widetilde{B}_{i-1,i-1}(\omega), \\ &\widetilde{B}_i\Big(\omega+\lambda(1-\widetilde{K}(\widetilde{B}_{1,i-1}(\omega),\ldots,\widetilde{B}_i(\omega),\ldots,\widetilde{B}_{N,i-1}(\omega)))\Big) \\ &\qquad ,\widetilde{B}_{i+1,i-1}(\omega),\ldots,\widetilde{B}_{N,i-1}(\omega),\widetilde{B}_i(\omega)\Big), \end{aligned} \tag{5.60}$$

The interpretation of $\boldsymbol{B}^{\boldsymbol{G}}_{\boldsymbol{j,i}}$, $j \neq i$ is similar to (5.22). On the other hand, $\boldsymbol{B}^{\boldsymbol{G}}_{\boldsymbol{i,i}}$ contains the service times of a complete cycle starting in Q_i. This includes the service times of all the customers that are standing behind the gate in Q_i, the service times of all the customers in $Q_{i+1},\ldots,Q_{i-1}$ that were already in the system on the arrival of the customer batch or entered the system before the next visit to Q_i, and when the server reaches Q_i again the service times of all the customers that were standing before the gate when the cycle in Q_i started.

We first focus on the batch sojourn-time of a customer batch that arrives during a visit period V_{j-1}. The batch sojourn-time of customer batch $\boldsymbol{k}$ that arrives when the server is in visit period V_{j-1} consists of the (i) residual service time in Q_{j-1}, (ii) the service of all the customers behind the gate in $Q_{j-1},\ldots,Q_i$, (iii) the service of all new customer arrivals that arrive after customer batch $\boldsymbol{k}$ in $Q_j,\ldots,Q_{i-1}$ before the server reaches Q_i, (iv) switch-over times $S_{j-1},\ldots,S_{i-1}$, (v) the service of the customers in customer batch $\boldsymbol{k}$, and (vi) if $i = j-1$ also the customers before the gate in Q_i. Because incoming customers are placed before the gate when the server is in visit period V_{j-1}, we have to modify (5.25) to,

$$\begin{aligned} \widetilde{L}^{(V_{j-1})}(\boldsymbol{z},\omega) = &\widetilde{LB}^{(B_{j-1})}(\boldsymbol{z}) \\ &\times \widetilde{B}^{PR}_{j-1}(\lambda-\lambda K(z_1,\ldots,z_{j-2},z_G,z_j,\ldots,z_N),\omega). \end{aligned} \tag{5.61}$$

Then, the LST of batch sojourn-time distribution of batch $\boldsymbol{k}$ given that the server is in visit period V_{j-1} is given in the next proposition.

Proposition 5.3. *The LST of the batch sojourn-time distribution of batch* $\boldsymbol{k}$ *conditioned that the server is in visit period* V_{j-1} *and the last customer in the batch will be served in* Q_i *is given by,*

$$\widetilde{T}_{\boldsymbol{k}}^{(V_{j-1})}(\omega) = \widetilde{L}^{(V_{j-1})}\left(\boldsymbol{B}^{\boldsymbol{G}}_{\boldsymbol{j-1,i}},\, \omega + \lambda(1 - \widetilde{K}(\boldsymbol{B}_{\boldsymbol{j,i-1}}))\right) \times \prod_{l=j-1}^{i-1}{}' \widetilde{S}_{l,i-1}(\omega)\, \frac{1}{(\boldsymbol{B}^{\boldsymbol{G}}_{\boldsymbol{j-1,i}})_{j-1}} \prod_{l=j}^{i}{}'(\boldsymbol{B}^{*}_{\boldsymbol{j,i}})_l^{k_l}. \quad (5.62)$$

Proof. During visit period V_{j-1} incoming customers to Q_{j-1} are placed before the gate and will be served in the next visit. Taken this into account, the same steps as in the proof of Proposition 5.1 can be used to derive (5.62). □

Next, we derive the LST of batch sojourn-time distribution of batch $\boldsymbol{k}$ given that the server is in switch-over period S_{j-1}. For this we modify (5.28) to,

$$\widetilde{L}^{(S_{j-1})}(\boldsymbol{z}, \omega) = \widetilde{LB}^{(S_{j-1})}(\boldsymbol{z}) \times \widetilde{S}^{PR}_{j-1}\left(\lambda - \lambda\widetilde{K}\left(z_1, \ldots, z_{j-2}, z_{j-1}, z_j, \ldots, z_N\right), \omega\right). \quad (5.63)$$

Proposition 5.4. *The LST of the batch sojourn-time distribution of batch* $\boldsymbol{k}$ *conditioned that the server is in switch-over period* S_{j-1} *and the last customer in the batch will be served in* Q_i *is given by*

$$\tilde{T}_{\boldsymbol{k}}^{(S_{j-1})}(\omega) = \widetilde{L}^{(S_{j-1})}\left(\boldsymbol{B}^{*}_{\boldsymbol{j,i}},\, \omega + \lambda(1 - \widetilde{K}(\boldsymbol{B}_{\boldsymbol{j,i-1}}))\right) \times \prod_{l=1}^{i-j}{}' \widetilde{S}_{j+l-1,i-1}(\omega) \prod_{l=j}^{i}{}'(\boldsymbol{B}^{*}_{\boldsymbol{j,i}})_l^{k_l}. \quad (5.64)$$

Proof. Similarly, the same steps as in the proof of Proposition 5.2 can be used to derive (5.64). □

From Proposition 5.3 and Proposition 5.4, it can be seen that the LST of the batch sojourn-time distribution of batch $\boldsymbol{k}$ conditioned on a visit/switch-over period can be decomposed into two terms;

$$\widetilde{T}_{\boldsymbol{k}}^{(V_{j-1})}(\omega) = \sum_{i=1}^{N} 1_{(\boldsymbol{k}\in\mathcal{K}_{j,i})} \widetilde{W}_i^{(V_{j-1})}(\omega) \prod_{l=j}^{i}{}'(\boldsymbol{B}^{*}_{\boldsymbol{j,i}})_l^{k_l}, \quad (5.65)$$

$$\widetilde{T}_{\boldsymbol{k}}^{(S_{j-1})}(\omega) = \sum_{i=1}^{N} 1_{(\boldsymbol{k}\in\mathcal{K}_{j,i})} \widetilde{W}_i^{(S_{j-1})}(\omega) \prod_{l=j}^{i}{}' (\boldsymbol{B}^*_{\boldsymbol{j,i}})_l^{k_l}, \tag{5.66}$$

where $\widetilde{W}_i^{(V_{j-1})}(\omega)$ and $\widetilde{W}_i^{(S_{j-1})}(\omega)$ can be considered as the time between the batch arrival epoch and the service completion of the last customer in Q_i that is already in the system, excluding any arrivals to Q_i after the arrival epoch and contribution of the batch.

The LST of the batch sojourn-time distribution of a specific customer batch $\boldsymbol{k}$ can now be calculated using (5.19) or alternatively by (5.19),

$$\widetilde{T}_{\boldsymbol{k}}(\omega) = \frac{1}{E(C)} \sum_{j=1}^{N}\sum_{i=1}^{N} 1_{(\boldsymbol{k}\in\mathcal{K}_{j,i})} \left(E(V_{j-1})\, \widetilde{W}_i^{(V_{j-1})}(\omega) + E(S_{j-1})\, \widetilde{W}_i^{(S_{j-1})}(\omega)\right) \times \prod_{l=j}^{i}{}' (\boldsymbol{B}^*_{\boldsymbol{j,i}})_l^{k_l}. \tag{5.67}$$

Finally, we focus on the LST of the batch sojourn-time of an arbitrary batch $\widetilde{T}(.)$.

Theorem 5.3. *The LST of the batch sojourn-time distribution of an arbitrary batch* $\widetilde{T}(\boldsymbol{.})$*, if this queue receives locally-gated service, is given by:*

$$\widetilde{T}(\omega) = \sum_{\boldsymbol{k}\in\mathcal{K}} \pi(\boldsymbol{k})\, \widetilde{T}_{\boldsymbol{k}}(\omega), \tag{5.68}$$

where $\widetilde{T}_{\boldsymbol{k}}(\omega)$ *is given by* (5.19) *or* (5.67)*. Alternatively, we can write* (5.68) *as,*

$$\widetilde{T}(\omega) = \frac{1}{E(C)} \sum_{j=1}^{N}\sum_{i=1}^{N} \left(E(V_{j-1})\, \widetilde{W}_i^{(V_{j-1})}(\omega) + E(S_{j-1})\, \widetilde{W}_i^{(S_{j-1})}(\omega)\right) \times \pi(\mathcal{K}_{j,i})\, \widetilde{K}\left(\boldsymbol{B}^*_{\boldsymbol{j,i}}|\mathcal{K}_{j,i}\right). \tag{5.69}$$

Proof. Using the definition of $\mathcal{K}_{j,i}$, the proof is almost identical to the one of Theorem 5.2. □

5.4.3 Mean value analysis

In this section, we will use MVA again to derive the mean batch sojourn-time of a specific batch and an arbitrary batch. Central in the MVA for locally-gated service is $E\left(\bar{L}_i^{(V_j,S_j)}\right)$, the mean queue-length at Q_i (excluding the potential customer currently in service) at an arbitrary epoch within visit period V_j and switch-over period S_j. First, for notation purposes we introduce θ_j as shorthand for intervisit period (V_j, S_j); the expected duration of this period $E(\theta_j)$ is given by,

$$E(\theta_j) = E(V_j) + E(S_j), \quad j = 1, \ldots, N. \tag{5.70}$$

The big difference with Section 5.3.3 is that we know have to consider the customers that stand before the gate and those who stand behind. For this we introduce variables $E\left(\tilde{L}_i^{(\theta_j)}\right)$ as the expected number of customers standing before the gate the gate in Q_i during intervisit period θ_j and $E\left(\hat{L}_i^{(\theta_i)}\right)$ as the expected number of customers standing behind the gate the gate in Q_i during intervisit period θ_i. In the MVA all incoming customers are placed before the gate, and only placed behind the gate when a visit period begins. Note this is a slight difference with Section 5.4.1 where only customers arriving to the same queue that is being visited are placed before the gate. Then the mean queue-length in Q_i, $E\left(\bar{L}_i^{(\theta_j)}\right)$, given that the server is not in intervisit period θ_i, i.e. $i \neq j$, is equal to the mean number of customers standing before the gate $E\left(\tilde{L}_i^{(\theta_j)}\right)$. Otherwise, when $i = j$ the mean queue length in Q_i is the sum of the number of customers standing in front and behind the gate. Thus we can write $E\left(\bar{L}_i^{(\theta_j)}\right)$ as,

$$E\left(\bar{L}_i^{(\theta_j)}\right) = \begin{cases} E\left(\tilde{L}_i^{(\theta_j)}\right) + E\left(\hat{L}_i^{(\theta_i)}\right), & i = j, \\ E\left(\tilde{L}_i^{(\theta_j)}\right), & \text{otherwise.} \end{cases}$$

Subsequently, the mean queue-length in Q_i is given by,

$$E\left(\bar{L}_i\right) = \sum_{j=1}^{N} \frac{E(\theta_j)}{E(C)} E\left(\tilde{L}_i^{(\theta_j)}\right) + \frac{E(\theta_i)}{E(C)} E\left(\hat{L}_i^{(\theta_i)}\right), \quad i = 1, \ldots, N. \tag{5.71}$$

We denote by $E(B_{j,i})$ the mean duration of a service time B_j and its descendants before the server starts service in Q_i given that the server is currently in Q_j. Let

$E(B_{j,j+1}) = E(B_j)$ be the expectation of B_j and $E(B_{j,j+2}) = E(B_j)(1+\rho_{j+1})$ be the sum of the service time B_j and the service of all the customers that arrive in Q_{j+1} during this service. In general we can write $E(B_{j,i})$ for $i \neq j+1$ as,

$$E(B_{j,i}) = E(B_j) \prod_{l=j+1}^{i-1}{}' (1+\rho_l), \quad i = 1,\ldots,N,\ j = 1,\ldots,N. \tag{5.72}$$

Finally, $E(S_{j,i})$, $E\left(B_{j,i}^R\right)$, and $E\left(S_{j,i}^R\right)$ are given by $E(B_{j,i})$ and replacing $E(B_j)$ with $E(S_j)$, $E\left(B_j^R\right)$, and $E\left(S_j^R\right)$ respectively.

Again, we consider the waiting time $E(W_i)$ of an arbitrary customer and make extensively use of Little's Law and the PASTA property. When the customer enters the system at Q_i, it has to wait for the next visit to Q_i. Even if the customer enters the system while the server is in intervisit period θ_i, the customer is placed before the gate and will only be served when the server returns to this queue in the next cycle. The average duration of the server returning to Q_i equals $E\left(\theta_{i,i-1}^R\right)$. Then at Q_i, the customer first has to wait for the service of the average number of customers $E\left(\tilde{L}_i\right) = \sum_{j=1}^N \left[E(\theta_j)/E(C)\right] E\left(\tilde{L}_i^{(\theta_j)}\right)$ that are in front of the customer when it arrived in the system, as well as, the service of $E(K_{ii})/2E(K_i)$ customers that arrived in the same customer batch, but are placed before the arbitrary customer in Q_i. This gives the following expression for the mean waiting time $E(W_i)$,

$$E(W_i) = E\left(\tilde{L}_i\right) E(B_i) + \frac{E(K_{ii})}{2E(K_i)} E(B_i) + E\left(\theta_{i,i-1}^R\right), \tag{5.73}$$

Applying Little's law gives,

$$E\left(\bar{L}_i\right) = \rho_i E\left(\tilde{L}_i\right) + \rho_i \frac{E(K_{ii})}{2E(K_i)} + \lambda_i E\left(\theta_{i,i-1}^R\right). \tag{5.74}$$

The next step to derive the equations is to relate unknowns $E\left(\theta_{i,i-1}^R\right)$ to $E\left(\tilde{L}_i^{(\theta_j)}\right)$ and $E\left(\hat{L}_i^{(\theta_i)}\right)$. Consider $E\left(\theta_{j,i}^R\right)$ the expected residual duration of an intervisit period starting in θ_j and ending in θ_i given that an arbitrary customer batch just entered the system. Then with probability $E(\theta_l)/E(\theta_{j,i})$, the server is during this period in intervisit period θ_l, $l = j,\ldots,i$, and the expected residual duration until the intervisit ending of θ_i, conditioned that the server is in intervisit period θ_l, is defined as follows.

First, with probability $E(V_l)/E(\theta_l)$ the customer has to wait for the server serving a customer in Q_l and switch-over period S_l and with probability $E(S_l)/E(C)$ the customer has to wait for a residual switch-over period in S_l. Also, $E\left(\hat{L}_l^{(\theta_j)}\right)$ customers are standing behind the gate in Q_l that need to be served. During this period new descendants can arrive in the system that will be served before the intervisit ending in θ_j. In addition, for each queue Q_n, $n = j+1, \ldots, i$, the expected number of customers in the Q_n given that the server is in θ_l, $E\left(\tilde{L}_n^{(\theta_l)}\right)$, and the expected number of customers that arrived in Q_n in the arbitrary customer batch $E(K_{nl})/E(K_n)$ will increase the duration of $E\left(\theta_{j,i}^R\right)$ by $E(B_{n,i+1})$. Finally, the switch-over times between Q_n to Q_{n+1} plus all its descendants that will be served before the end of the period contribute with $E(S_{n,i+1})$. Combining this gives the following expression,

$$\begin{aligned} E\left(\theta_{j,i}^R\right) = {}& {\sum_{l=j}^{i}}' \frac{E(\theta_l)}{E(\theta_{j,i})}\left(\frac{E(V_l)}{E(\theta_l)}\left(E\left(B_{l,i+1}^R\right) + E(S_{l,i+1})\right)\right. \\ & + \frac{E(S_l)}{E(\theta_l)} E\left(S_{l,i+1}^R\right) + E\left(\hat{L}_l^{(\theta_l)}\right) E(B_{l,i+1}) \\ & \left. + {\sum_{n=1}^{i-l}}' \left(\frac{E(K_{l+n,l})}{E(K_{l+n})} + E\left(\tilde{L}_{l+n}^{(\theta_l)}\right)\right) E(B_{l+n,i+1}) + E(S_{l+n,i+1})\right). \end{aligned} \tag{5.75}$$

It is now possible to set up a set of $N(N+1)$ linear equations in terms of unknowns $E\left(\tilde{L}_i^{(\theta_j)}\right)$ and $E\left(\hat{L}_i^{(\theta_i)}\right)$. First, the number of customers in Q_i before the gate given an arbitrary moment in an intervisit period starting in θ_i and ending in θ_j equals the number of Poisson arrivals during the age of this period. Since the age is in distribution equal to the residual time, the following equation holds, $i = 1, \ldots, N$, $j = 1, \ldots, N$,

$${\sum_{l=i}^{j}}' \frac{E(\theta_l)}{E(\theta_{i,j})} E\left(\tilde{L}_i^{(\theta_l)}\right) = \lambda_i E\left(\theta_{i,j}^R\right). \tag{5.76}$$

Second, by (5.73) and using Little's Law $\lambda_i E(W_i) = E\left(\bar{L}_i\right)$ into (5.74) gives, for $i = 1, 2 \ldots, N$,

$$\begin{aligned} (1-\rho_i) \sum_{j=1}^{N} \frac{E(\theta_j)}{E(C)} E\left(\tilde{L}_i^{(\theta_j)}\right) + \frac{E(\theta_i)}{E(C)} E\left(\hat{L}_i^{(\theta_i)}\right) - \rho_i \frac{E(K_{ii})}{2E(K_i)} \\ = \lambda_i E\left(\theta_{i,i-1}^R\right). \end{aligned} \tag{5.77}$$

With (5.76) and (5.77) a set of $N(N+1)$ linear equations are now defined. Solving the set of linear equations and by (5.74) and (5.73) will give the expected queue-lengths and waiting times.

It is now possible to derive the mean batch time $E(T_{\boldsymbol{k}})$ of customer batch $\boldsymbol{k}$ using (5.47). For this we need to calculate $E\left(T_{\boldsymbol{k}}^{(\theta_{j-1})}\right)$. When customer batch $\boldsymbol{k}$ enters the system and the server is in intervisit period θ_{j-1}, then with probability $E(V_{j-1})/E(\theta_{j-1})$ and $E(S_{j-1})/E(\theta_{j-1})$ the arriving customer batch has to wait for the residual service and a switch-over or a residual switch-over time during in which new customer can arrive that will be served before the visit completion in Q_{i-1}. Then each customer already in the system and in batch $\boldsymbol{k}$ in Q_l, $l = j-1, \ldots, i$ and their descendants will increase the batch sojourn-time. Finally, the batch also has to wait for all the switch-over times between Q_j to Q_{i-1} and all their descendants that will be served before the server reaches Q_i. This gives the following expression,

$$\begin{aligned} E\left(T_{\boldsymbol{k}}^{(\theta_{j-1})}\right) &= \frac{E(V_{j-1})}{E(\theta_{j-1})}\left(E\left(B_{j-1,i}^{R}\right) + E(S_{j-1,i})\right) + \frac{E(S_{j-1})}{E(\theta_{j-1})}E\left(S_{j-1,i}^{R}\right) \\ &+ E\left(\hat{L}_{j-1}^{(\theta_{j-1})}\right)E(B_{j-1,i}) + {\sum_{l=1}^{i-j}}'\left(E\left(\tilde{L}_{j+l-1}^{(\theta_{j-1})}\right) + k_{j+l-1}\right)E(B_{j+l-1,i}) \\ &\qquad + E(S_{j+l-1,i}) + \left(\left(\tilde{L}_{i}^{(\theta_{j-1})}\right) + k_i\right)E(B_i), \end{aligned} \tag{5.78}$$

Notice that the same decomposition as (5.31) and (5.32) also holds for the expected batch sojourn-time,

$$E\left(T_{\boldsymbol{k}}^{(\theta_{j-1})}\right) = E\left(W_{i}^{(\theta_{j-1})}\right) + {\sum_{l=1}^{i-j}}' k_{j+l-1}E(B_{j+l-1,i}) + k_i E(B_i), \tag{5.79}$$

where $E\left(W_{i}^{(\theta_{j-1})}\right)$ is the expected time between the batch arrival epoch and the service completion of the last customer in Q_i that is already in the system, excluding any arrivals to Q_i after the arrival epoch.

Finally, the expected batch sojourn-time of an arbitrary customer batch is given by (5.49). Similarly, we can rewrite (5.49) by taking the expectation of $\mathcal{K}_{j,i}$ and using (5.79),

$$E(T) = \frac{1}{E(C)} \sum_{j=1}^{N} \sum_{i=1}^{N} E(\theta_j)\, \pi(\mathcal{K}_{j,i}) \left(E\left(W_i^{(\theta_{j-1})}\right) + \sum_{l=1}^{i-j}{}' E(K_{j+l-1}|\mathcal{K}_{j,i})\, E(B_{j+l-1,i}) + E(K_i|\mathcal{K}_{j,i})\, E(B_i)\right).$$

5.5 Globally-gated service

In this section the batch sojourn distribution under globally-gated service is studied in Section 5.5.1, and the mean batch sojourn-times in Section 5.5.2.

5.5.1 Batch sojourn distribution

Under the globally-gated service discipline all the customers that were present at the visit beginning of reference queue Q_1 will be served during the coming cycle. Meanwhile, customers that arrive in the system during this cycle have to wait and will be served in the next cycle. The advantage of the globally-gated service discipline is that closed-form expressions can be easily derived for the delay distribution compared to exhaustive and locally-gated (Boxma et al., 1992).

Let random variables $n_1, \ldots n_N$ denote the number of customers in the queues at the beginning of an arbitrary cycle C and let $\widetilde{C}(\omega) = E\left(e^{-\omega C}\right)$ be its LST. Then, the length of the current cycle will equal the sum of all switch-over times and the total sum of all the service times of the customers present at the beginning of the cycle. Combining this gives,

$$E\left(e^{-\omega C}|n_1, \ldots, n_N\right) = \widetilde{S}(\omega) \prod_{j=1}^{N} \widetilde{B}_j^{n_j}(\omega), \tag{5.80}$$

where $\widetilde{S}(\omega) = \prod_{j=1}^{N} \widetilde{S}_j(\omega)$.

On the other hand, the length of a cycle determines the joint queue-length distribution at the beginning of the next cycle (Boxma et al., 1992),

$$\begin{aligned} E\left(z_1^{n_1}\cdots z_N^{n_N}\right) &= E\left(E\left(z_1^{n_1}\cdots z_N^{n_N}|C=t\right)\right) \\ &= E\left(\exp\left(-\left(\lambda-\lambda\widetilde{K}\left(\boldsymbol{z}\right)\right)t\right)\right) = \widetilde{C}\left(\lambda-\lambda\widetilde{K}\left(\boldsymbol{z}\right)\right). \end{aligned} \tag{5.81}$$

With use of (5.80) and (5.81), we have

$$\begin{aligned} \widetilde{C}\left(\omega\right) &= \widetilde{S}\left(\omega\right)E\left(\widetilde{B}_1^{n_1}\left(\omega\right)\cdots\widetilde{B}_N^{n_N}\left(\omega\right)\right) \\ &= \widetilde{S}\left(\omega\right)\widetilde{C}\left(\lambda-\lambda\widetilde{K}\left(\widetilde{B}_1\left(\omega\right),\ldots,\widetilde{B}_N\left(\omega\right)\right)\right). \end{aligned} \tag{5.82}$$

Let C_P and C_R be the past and residual time, respectively, of a cycle. We can write the LST of the joint distribution of C^P and C^R as (Cohen, 1982),

$$\widetilde{C}^{PR}\left(\omega_P,\omega_R\right) = \frac{\widetilde{C}\left(\omega_R\right)-\widetilde{C}\left(\omega_P\right)}{E\left(C\right)\left(\omega_P-\omega_R\right)}, \tag{5.83}$$

and

$$\widetilde{C}^{P}\left(\omega\right) = \widetilde{C}^{R}\left(\omega\right) = \frac{1-\widetilde{C}\left(\omega\right)}{\omega E\left(C\right)}. \tag{5.84}$$

Finally, let $\boldsymbol{B_{j,i}}$ be an N-dimensional vector with the LST of the service times of Q_l on elements $l=j,\ldots,i$,

$$\boldsymbol{B_{j,i}} = \left(1,\ldots,\widetilde{B}_j\left(\omega\right),\widetilde{B}_{j+1}\left(\omega\right),\ldots,\widetilde{B}_i\left(\omega\right),1,\ldots,1\right).$$

With the previous results, we can now derive the LST of the batch sojourn distribution of specific batch of customers.

Proposition 5.5. *The LST of the batch sojourn-time distribution of batch* $\boldsymbol{k}$ *is given by,*

$$\widetilde{T}_{\boldsymbol{k}}(\omega) = \frac{1}{E(C)} \left[\frac{\widetilde{C}\left(\lambda - \lambda\widetilde{K}(\boldsymbol{B_{1,i}})\right) - \widetilde{C}\left(\lambda - \lambda\widetilde{K}(\boldsymbol{B_{1,i-1}}) + \omega\right)}{\omega - \lambda\left(1 - \widetilde{K}(\boldsymbol{B_{i,i}})\right)} \right] \prod_{j=1}^{i-1} \widetilde{S}_j(\omega) \times \prod_{j=1}^{i} k_i \widetilde{B}_j(\omega). \quad (5.85)$$

Proof. Assume an arbitrary customer batch $\boldsymbol{k}$ where the number of customer arrivals per queue is $k_1 \geq 0, \ldots, k_i > 0$ and $k_{i+1} = 0, \ldots, k_N$. Due to the globally-gated service discipline, any arriving customer batch will be totally served in the next cycle, which implies that the customer batch will be fully served after its last customer in Q_i is served. Then, the batch sojourn-time of customer batch $\boldsymbol{k}$ is composed of; (i) the residual cycle time C^R, (ii) the service times of all customers who arrive at $Q_1, \ldots, Q_{i-1}$ during the cycle in which the new customer batch arrives, (iii) the switch-over times of the server between $Q_1, \ldots, Q_{i-1}$, (iv) the service times of all the customers who arrive at Q_i during the past part C^P of the cycle in which the customer batch arrives, and (v) the service times of all the customers in the batch at $Q_1, \ldots, Q_i$. Combining this gives,

$$T_{\boldsymbol{k}} = C^R + \sum_{j=1}^{i-1} \sum_{m=1}^{N_j\left(C^P+C^R\right)} B_{j_m} + \sum_{j=1}^{i-1} S_j + \sum_{m=1}^{N_i\left(C^P\right)} B_{i_m} + \sum_{j=1}^{i} \sum_{m=1}^{k_j} B_{j_m}, \quad (5.86)$$

where $N_j\left(C^P + C^R\right)$ denotes number of arrivals in Q_j during the past and residual time of the current cycle and $N_i\left(C^P\right)$ denotes the number of arriving customers in Q_i during C^P. Note that the mean cycle length in which the customer batch arrives is not equal to $E(C)$, but is atypical of size $E\left(C^P\right) + E\left(C^R\right)$ (Boxma et al., 1992). By taking the LST of (5.86) we obtain,

$$\widetilde{T}_{\boldsymbol{k}}(\omega) = \prod_{j=1}^{i-1} \widetilde{S}_j(\omega) \int_{t_P=0}^{\infty} \int_{t_R=0}^{\infty} e^{-\omega t_R} e^{-\left(\lambda - \lambda K\left(\boldsymbol{B_{1,i-1}}\right)\right)(t_P+t_R)} \times e^{-\left(\lambda - \lambda\widetilde{K}\left(\boldsymbol{B_{i,i}}\right)\right)t_P} \mathrm{d}Pr\left(C^P < t_P,\, C^R < t_R\right) \prod_{j=1}^{i} k_j \widetilde{B}_j(\omega)$$

$$= \prod_{j=1}^{i-1} \widetilde{S}_j(\omega)\, E\left(\exp\left(-\left(\lambda - \lambda\widetilde{K}\left(\boldsymbol{B_{1,i}}\right)\right) C^P - \left(\lambda - \lambda\widetilde{K}\left(\boldsymbol{B_{1,i-1}}\right) + \omega\right) C^R\right)\right) \times \prod_{j=1}^{i} k_j \widetilde{B}_j(\omega),$$

Using the LST of the joint distribution of C_P and C_R of (5.83), we obtain (5.85). □

We can now find the LST of the batch sojourn-time distribution of an arbitrary batch.

Theorem 5.4. *The LST of the batch sojourn-time distribution of an arbitrary batch* $\widetilde{T}(\mathbf{.})$*, if this queue receives globally-gated service, is given by:*

$$\widetilde{T}(\omega) = \sum_{\boldsymbol{k}\in\mathcal{K}} \pi(\boldsymbol{k})\, \widetilde{T}_{\boldsymbol{k}}(\omega), \tag{5.87}$$

where $\widetilde{T}_{\boldsymbol{k}}(\omega)$ *is given by* (5.19)*. Alternatively, we can write* (5.85) *as,*

$$\widetilde{T}(\omega) = \frac{1}{E(C)} \sum_{i=1}^{N} \left[\frac{\widetilde{C}\left(\lambda - \lambda\widetilde{K}\left(\boldsymbol{B_{1,i}}\right)\right) - \widetilde{C}\left(\lambda - \lambda\widetilde{K}\left(\boldsymbol{B_{1,i-1}}\right) + \omega\right)}{\omega - \lambda\left(1 - \widetilde{K}\left(\boldsymbol{B_{i,i}}\right)\right)} \right] \times \prod_{j=1}^{i-1} \widetilde{S}_j(\omega)\, \pi(\mathcal{K}_{1,i})\, \widetilde{K}\left(\boldsymbol{B_{1,i}}|\mathcal{K}_{1,i}\right). \tag{5.88}$$

Proof. In case of globally-gated an incoming customer batch can only be served in the next cycle. Therefore, independently on the location of the server the last customer in the batch to be served is located in the queue that is the farthest located from the reference queue. Thus, we can write

$$\widetilde{T}(\omega) = \sum_{\boldsymbol{k}\in\mathcal{K}} \sum_{i=1}^{N} 1_{(\boldsymbol{k}\in\mathcal{K}_{1,i})} \pi(\boldsymbol{k})\, \widetilde{T}_{\boldsymbol{k}}(\omega).$$

Finally, by inserting (5.85) and (5.1) we obtain (5.88). □

5.5.2 Mean batch sojourn-time

In this section we determine $E(T_{\boldsymbol{k}})$, the expected batch sojourn-time for a specific customer batch $\boldsymbol{k}$. Instead of using MVA, as was the case for exhaustive and locally-

gated, we can directly calculate $E(T_{\boldsymbol{k}})$ similar as for the mean waiting time (Boxma et al., 1992). Taking the expectation of (5.86) gives the following expression,

$$E(T_{\boldsymbol{k}}) = E\left(C^R\right) + \sum_{j=1}^{i-1} \lambda_j E(B_j)\left(E\left(C^P\right) + E\left(C^R\right)\right) + \sum_{j=1}^{i-1} E(S_j) + \rho_i E\left(C^P\right) + \sum_{j=1}^{i} k_j E(B_j). \quad (5.89)$$

What is left is to derive the mean past and residual time of the cycle time, $E(C_P)$ and $E(C_R)$. Differentiating (5.82) once and twice yields closed-form expressions for the first two moment of the cycle time,

$$E(C) = \frac{E(S)}{(1-\rho)}, \quad (5.90)$$

$$E\left(C^2\right) = \frac{1}{(1-\rho^2)}\left[E\left(S^2\right) + 2\rho E(S) E(C) + \sum_{j=1}^{N} \lambda_j E\left(B_j^2\right) E(C) + \sum_{i=1}^{N}\sum_{j=1}^{N} \lambda E(K_{ij}) E(B_i) E(B_j) E(C)\right]. \quad (5.91)$$

and the expected past and residual cycle time is given by

$$E\left(C^P\right) = E\left(C^R\right) = \frac{E(C^2)}{2E(C)} = \frac{1}{(1+\rho)}\left[\frac{E(S^2)}{2E(S)} + \frac{\rho E(S)}{(1-\rho)} + \frac{\sum_{j=1}^{N} \lambda_j E\left(B_j^2\right) + \sum_{i=1}^{N}\sum_{j=1}^{N} \lambda E(K_{ij}) E(B_i) E(B_j)}{2(1-\rho)}\right]. \quad (5.92)$$

Using (5.92), we can rewrite (5.89) as follows,

$$E(T_{\boldsymbol{k}}) = \left[1 + 2\sum_{j=1}^{i-1} \rho_j + \rho_i\right] \frac{E(C^2)}{2E(C)} + \sum_{j=1}^{i-1} E(S_j) + \sum_{j=1}^{i} k_j E(B_j). \quad (5.93)$$

Finally, we can derive $E(T)$ the expected batch sojourn-time of an arbitrary customer batch. Multiplying $E(T_{\boldsymbol{k}})$ with all possible realizations of $\boldsymbol{k}$ and using $\mathcal{K}_{1,i}$ gives,

$$E(T) = \sum_{i=1}^{N} \sum_{\boldsymbol{k} \in \mathcal{K}_{1,i}} \pi(\boldsymbol{k}) \left(\left[1 + 2\sum_{l=1}^{i-1} \rho_l + \rho_i\right] \frac{E(C^2)}{2E(C)} + \sum_{j=1}^{i-1} E(S_j) + \sum_{j=1}^{i} k_j E(B_j)\right)$$

$$= \sum_{i=1}^{N} \pi\left(\mathcal{K}_{1,i}\right) \left(\left[1 + 2 \sum_{l=1}^{i-1} \rho_l + \rho_i \right] \frac{E\left(C^2\right)}{2E\left(C\right)} + \sum_{j=1}^{i-1} E\left(S_j\right) \right) + \sum_{j=1}^{N} E\left(K_j\right) E\left(B_j\right)$$

$$= \frac{E\left(C^2\right)}{2E\left(C\right)} + \sum_{i=1}^{N} \left(\rho_i \frac{E\left(C^2\right)}{E\left(C\right)} + E\left(S_i\right) \right) \cdot \left(1 - \sum_{j=1}^{i} \pi\left(\mathcal{K}_{1,j}\right) \right)$$

$$+ \rho_i \frac{E\left(C^2\right)}{2E\left(C\right)} \pi\left(\mathcal{K}_{1,i}\right) + E\left(K_i\right) E\left(B_i\right).$$

5.6 Numerical results

In the following section we investigate the batch sojourn-times for the three server disciplines. In Section 5.6.1 we study a symmetrical polling system with two queues and derive a closed form solution for the expected batch sojourn-times and show under which parameters settings, which service discipline has the lowest expected batch sojourn-time. In Section 5.6.2 we study asymmetrical systems and show that the service discipline that achieves the lowest expected batch sojourn-time depends on the system parameters.

5.6.1 A symmetrical polling system with two exponential queues

Consider a symmetrical polling system with two queues where all customers arrive in pairs and each of them joins another queue as shown in Figure 5.4. Assume that the arrival rate is λ, the expected service time of a customer in Q_1 or Q_2 is $E\left(B_1\right) = E\left(B_2\right) = b$, and the expected switch-over time from Q_1 to Q_2 and vice versa is $E\left(S_1\right) = E\left(S_2\right) = s$. In addition, we make the assumption that both service times and switch-over times are exponentially distributed; i.e. $E\left(B_1^R\right) = E\left(B_2^R\right) = b$ and $E\left(S_1^R\right) = E\left(S_2^R\right) = s$. Since customers arrive in pairs, $E\left(K_1\right) = E\left(K_2\right) = 1$, and $E\left(K_{12}\right) = E\left(K_{21}\right) = 1$ and $E\left(K_{11}\right) = E\left(K_{22}\right) = 0$. Finally, the overall system load is $\rho = \rho_1 + \rho_2 = 2b\lambda$.

First, consider the expected batch sojourn-time $E\left(T^{EX}\right)$ in case of *exhaustive service*. When a new pair of customers enter the system, they will encounter with equal probability the system either in intervisit period $\theta_1 = (S_2, V_1)$ or $\theta_2 = (S_1, V_2)$. Because of exhaustive service, the first customer will be served within the current intervisit period, whereas the second will be served in the following intervisit period. Because the queues are symmetrical, with probability ρ the pair of customers should

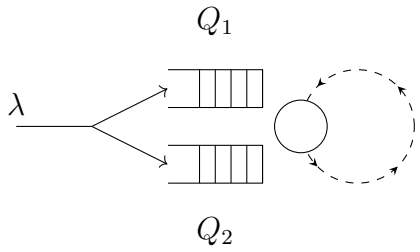


Figure 5.4: A symmetrical polling system with two exponential queues.

wait for the remaining service of a customer and the service of new arrivals to the same queue total duration of which is $b/(1-0.5\rho)$ and with probability $1-\rho$ they should wait for the remaining duration of a switch-over period and the busy period it triggers of duration $s/(1-0.5\rho)$. In addition, there are $\bar{L}^S = \bar{L}_1^{(\theta_1)} = \bar{L}_2^{(\theta_2)}$ customers waiting at the queue that are served within the current intervisit period each of which trigger a busy period of $b/(1-0.5\rho)$ and, in addition, one of the arriving customers will be taken into service and trigger a busy period of $b/(1-0.5\rho)$. After this, the server moves to the other queue which takes a switch-over time s. Then at the other queue, first the customers that were already in the queue before the pair of customers arrived at the system $\bar{L}^O = \bar{L}_1^{(\theta_2)} = \bar{L}_2^{(\theta_1)}$ will be served and afterwards the other arriving customer is served. Hence, the average batch sojourn-time in case of exhaustive service is given as follows,

$$E\left(T^{EX}\right) = \underbrace{\frac{1}{1-0.5\rho}\left[\rho b + (1-\rho)\,s + b + \bar{L}^S b\right]}_{\text{Intervisit 1}} + \underbrace{s + \bar{L}^O b + b.}_{\text{Intervisit 2}} \tag{5.94}$$

Solving the linear equations of (5.45) and (5.46) gives,

$$\bar{L}^S = \frac{\lambda(1.5\rho b - 1.5\rho s + 2s)}{1-\rho}, \qquad \bar{L}^O = \frac{\lambda\,(0.5\rho b - 0.5\rho s + b + s)}{1-\rho},$$

and by substituting $\bar{L}^S$ and $\bar{L}^O$ in (5.94), we obtain the expected batch sojourn-time in case of exhaustive service,

$$E\left(T^{EX}\right) = \frac{0.25\rho^2 b - 0.25\rho^2 s - \rho s + 2b + 2s}{1-\rho}. \tag{5.95}$$

Second, consider the expected batch sojourn-time in case of *locally-gated* service. In this case, neither of the arriving customers will be served during the current intervisit period, since both customers are placed before a gate. The residual duration of the current intervisit period is $\rho\left(b+s\right)+\left(1-\rho\right)s+\hat{L}^{S}b$, where $\hat{L}^{S}=\hat{L}_{1}^{(\theta_1)}=\hat{L}_{2}^{(\theta_2)}$ are the average number of customers standing before the gate on the arriving of the customer pair. Then, in the next intervisit period, $\tilde{L}^{O}=\tilde{L}_{1}^{(\theta_2)}=\tilde{L}_{2}^{(\theta_1)}$ customers will be served, as well as, all the customers that arrived to this queue during the previous intervisit period and the one of the arriving customers. Afterwards, the server returns to the other queue again and serves first the $\tilde{L}^{S}=\tilde{L}_{1}^{(\theta_1)}=\tilde{L}_{2}^{(\theta_2)}$ customers that were standing before the gate when the pair of customers entered the system and finally the other arriving customer. Then, the average batch sojourn-time in case of locally-gated service is given as follows,

$$E\left(T^{LG}\right)=\underbrace{\left[\rho\left(b+s\right)+\left(1-\rho\right)s^{R}+\hat{L}^{S}b\right]}_{\text{Intervisit 1}}+ \\ +\underbrace{\left[\rho\left(b+s\right)+\left(1-\rho\right)s+\hat{L}^{S}b\right]0.5\rho+\tilde{L}^{O}b+b+s}_{\text{Intervisit 2}}+\underbrace{\tilde{L}^{S}b+b}_{\text{Intervisit 3}}, \quad (5.96)$$

Solving the linear equations of (5.76) and (5.77) gives,

$$\hat{L}^{S}=\frac{\left(0.5\rho^{3}+0.25\rho^{2}+1.5\rho s\lambda\right)}{\left(1+0.5\rho\right)\left(1-\rho\right)}, \qquad \tilde{L}^{O}=\frac{\lambda\left(0.5\rho b-0.5\rho s+b+2s\right)}{1-\rho}, \\ \tilde{L}^{S}=\frac{\lambda\left(-0.25\rho^{2}b+0.25\rho^{2}s+\rho b-0.5\rho s+s\right)}{\left(1+0.5\rho\right)\left(1-\rho\right)},$$

and by substituting $\hat{L}^{S}$, $\tilde{L}^{S}$, and $\tilde{L}^{O}$ in (5.96), we obtain the expected batch sojourn-time in case of locally-gated service,

$$E\left(T^{LG}\right)=\frac{-0.125\rho^{3}b+0.125\rho^{3}s+0.25\rho^{2}b-0.5\rho^{2}s+0.5\rho b+\rho s+2b+2s}{\left(1+0.5\rho\right)\left(1-\rho\right)}. \quad (5.97)$$

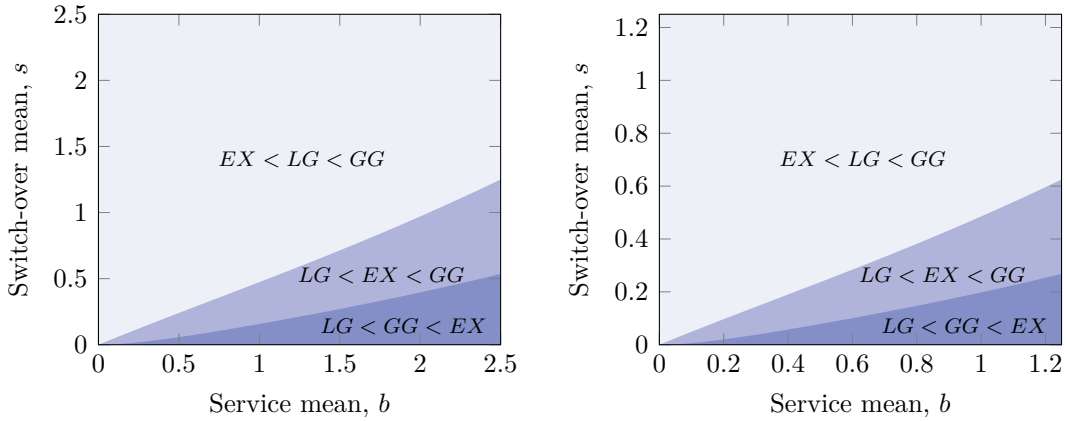
Finally, consider the expected batch sojourn-time in case of *globally-gated* service.

$$E\left(T^{GG}\right)=\left(1+1.5\rho\right)\frac{E\left(C^{2}\right)}{2E\left(C\right)}+s+2b, \quad (5.98)$$

Then by (5.92), we obtain the expected batch sojourn-time in case of globally-gated service,

$$E\left(T^{GG}\right) = \frac{0.5\rho^2 b - 0.5\rho^2 s + 3\rho b + 5.5\rho s + 4b + 5s}{2\left(1+\rho\right)\left(1-\rho\right)}. \tag{5.99}$$

Now, we can compare the expected batch sojourn-times $E\left(T^{EX}\right)$, $E\left(T^{LG}\right)$, and $E\left(T^{GG}\right)$ and investigate under which parameters settings which service discipline achieves the lowest expected batch sojourn-time. Figure 5.5 shows for two different total arrival rates, Λ, the areas where a specific service discipline achieves the lowest expected batch sojourn-time. From the figures it can be seen that when the switch-over times are longer compared to the service times, the exhaustive service discipline achieves the lowest expected batch sojourn-time, since it is more beneficial to serve all customers at the current queue first before moving to the other queue. However, if the service times are longer than the switch-over times it is better to switch to the other queue more often, because otherwise the server will spend too much time serving customers in one queue and it will take a long time before a customer batch is completely served. In this case, both gated policies perform better than exhaustive service. For both Λ the same pattern can be observed.



(a) $\Lambda = 0.4$ **(b)** $\Lambda = 0.8$

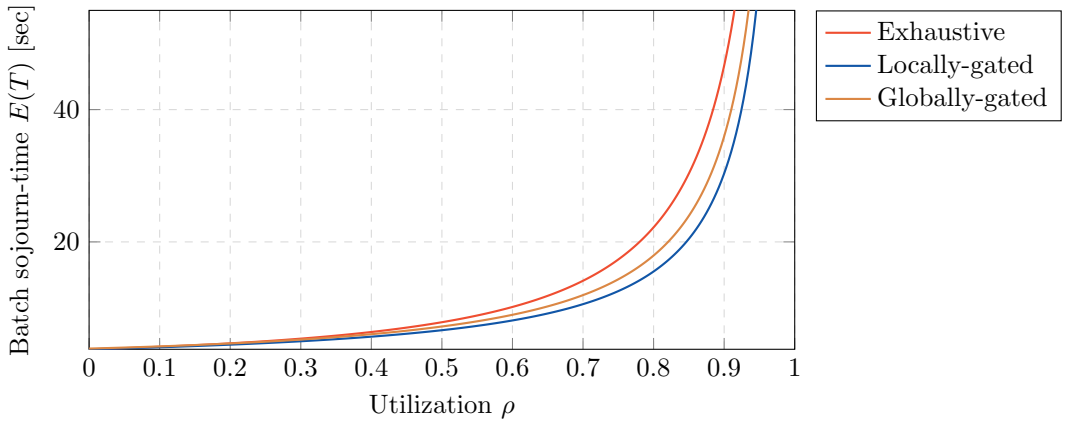
Figure 5.5: The expected batch sojourn-time for symmetrical polling system with two queues.

5.6.2 Asymmetrical polling systems with multiple queues

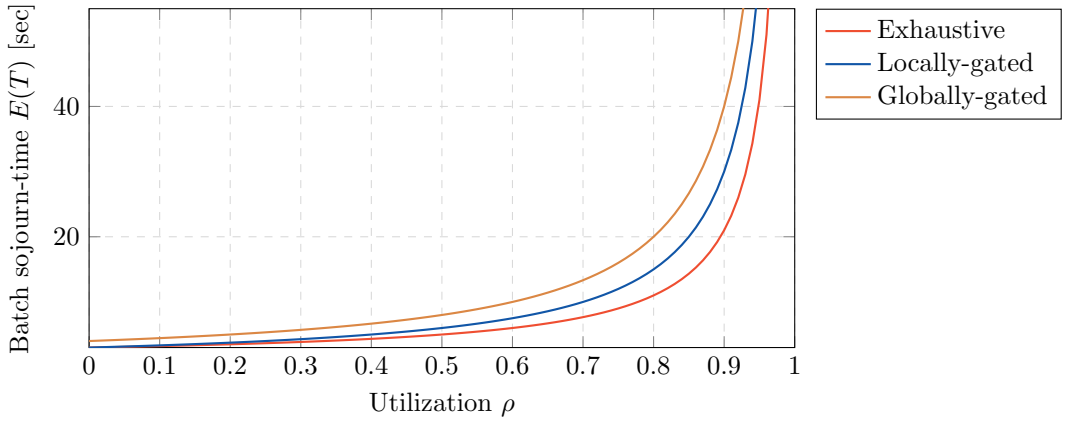
Table 5.1: Parameters for three polling models.

	Model a			Model b			Model c		
Q_i	1	2	3	1	2	3	1	2	3
$E(B_i)$	1.00	1.00	1.00	1.00	1.00	1.00	0.10	0.40	0.90
$E\left(B_i^{(2)}\right)$	2.00	2.00	2.00	2.00	2.00	2.00	1.00	1.00	1.00
$E(S_i)$	0.10	0.10	0.10	1.00	1.00	1.00	1.00	1.00	1.00
$E\left(S_i^{(2)}\right)$	0.02	0.02	0.02	2.00	2.00	2.00	1.00	1.00	1.00
$\boldsymbol{k} \in \mathcal{K}$	$\pi(1,1,0) = 1/4$			$\pi(1,0,0) = 1/3$			$\pi(1,1,0) = 4/5$		
	$\pi(3,0,1) = 3/4$			$\pi(0,1,0) = 1/3$			$\pi(1,0,3) = 1/5$		
				$\pi(0,0,1) = 1/3$					

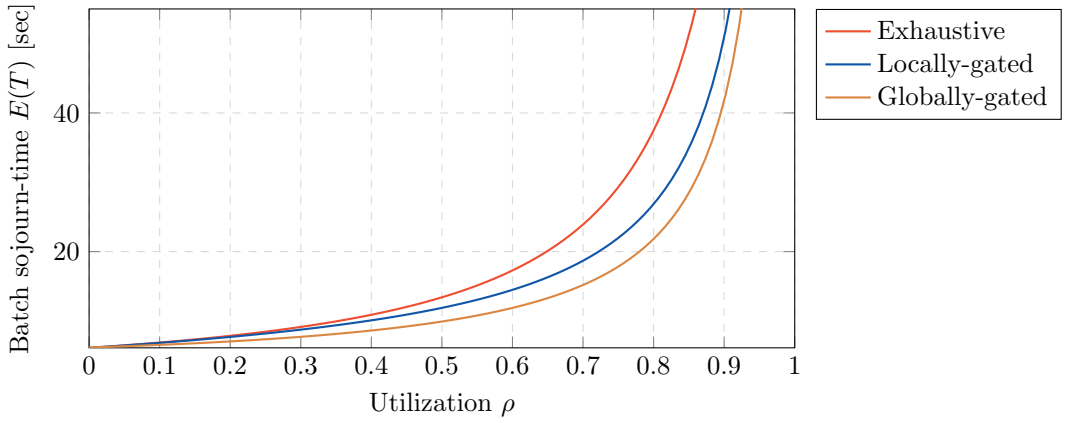
In the previous section, we have shown that depending on the system parameters exhaustive service or locally-gated service minimizes the expected batch sojourn-time. However, it can be shown that any of the three service disciplines studied in this chapter can minimize the expected batch sojourn-time. In Table 5.1 the parameters of three systems with $N = 3$ are given. *Model a* has short switch-over times, *Model b* is a system with individual arriving customers and equal switch-over times and service times, and in *Model c* the last queue is the slowest and receives most of the work. Using the results of Section 5.3.3, Section 5.4.3, and Section 5.5.2 the expected batch sojourn-times for the three different models can be calculated. The batch sojourn-times are shown in Figure 5.6 for $0 \leq \rho < 1$. The results of *Model a* in Figure 5.6a show that locally-gated achieves the lowest expected batch sojourn-times, which is similar as in Section 5.6.1 when the switch-over times were short. From the results of *Model b* shown in Figure 5.6b, it can be seen that exhaustive service has the lowest expected batch sojourn-times. Here it is beneficial to serve a customer arriving to the same queue that is currently being served, since otherwise this customer has to wait a full cycle which increases the mean batch sojourn-time. Finally, *Model c* in Figure 5.6c shows that globally-gated service achieves the lowest expected batch sojourn-times, since for this policy the server will switch more often between the queues and finish service for all customers in a batch during one cycle, compared to the other disciplines.



(a) Locally-gated minimizes the expected batch sojourn-time



(b) Exhaustive minimizes the expected batch sojourn-time



(c) Globally-gated minimizes the expected batch sojourn-time

Figure 5.6: The expected batch sojourn-time for various utilizations for three different systems.

5.7 Conclusion and further research

In this chapter we analyzed the batch sojourn-time in a cyclic polling system with simultaneous batch arrivals and obtained exact expressions for the Laplace-Stieltjes transform of the steady-state batch sojourn-time distribution for the locally-gated, globally-gated, and exhaustive service disciplines. Also, we provided a more efficient way to determine the mean batch sojourn-time using the Mean Value Analysis. We compared the batch sojourn-times for the different service disciplines in several numerical examples and showed that the best performing service discipline, minimizing the batch sojourn-time, depends on system characteristics.

A further research topic would be to determine for each of the three policies, under what conditions for the system parameters, its mean batch sojourn-time is smaller than that of the other two and whether alternative service disciplines can achieve even lower batch sojourn-times. Another interesting further research topic would be to study how the customers of an arriving customer batch should be allocated over the various queues in order to minimize the batch sojourn-times.

6 Optimizing product allocation in a milkrun picking system

6.1 Introduction

Recent technological advances and trends in distribution and manufacturing have led to a growth in complexity of warehousing systems. Today's warehouse operations face challenges like the need for shorter lead times, for real-time response, to handle a larger number of orders with greater variety, and to deal with flexible processes (Gong & De Koster, 2011). Staying competitive requires efficient and flexible order picking systems in warehouses.

Batch picking is a common way to organize the picking process, in case daily a large number of customer orders needs to be picked. Batch picking is a picker-to-parts order picking system in which the demand from multiple orders is used to form so-called pick batches (De Koster et al., 2007). Eventually, a pick batch is released and an order picker collects all the individual products that form the batch with a pick cart while traveling through the warehouse in an efficient picking path. When finished, he or she disposes the picked products centrally where they are sorted per customer order. A drawback of this approach is that batch formation takes time, and, as the number of daily orders to be processed increases and as the required lead time becomes shorter, more efficient ways to organize the order picking process exist. In this chapter we study an alternative method of order picking served by *milkrun logistics* that allows shorter order throughput times compared to conventional batch picking systems particular for high order arrival rates.

Milkrun logistics refers to the scheduled pickup or delivery of materials at a number of customers or suppliers by a single vehicle that visits them according to a fixed

round trip (Baudin, 2004). Originating from the dairy industry, a milkrun reduces costs due to consolidated transportation. It allows for better pick up or drop off coordination with the customers or suppliers, and it improves general response times and system efficiency (Brar & Saini, 2011). The same milkrun concept can also be used for internal logistics, e.g. manufacturing or warehousing, for the transport of raw materials, work in process, or finished goods between different locations within the building (Kovács (2011), Kilic et al. (2012), Bozer & Ciemnoczolowski (2013), and Staab et al. (2015)). Furthermore, milkrun systems can lead to considerable savings in labor costs and operating costs and, compared to a system that uses forklifts to transport materials in assembly or production lines, a milkrun is more efficient, safer because of less traffic, and significantly reduces the number of empty runs (Kilic et al., 2012). Therefore, order picking can also benefit from milkrun logistics (Gong & De Koster, 2008).

In a *milkrun picking system*, an order picker picks orders in batches that arrive in real-time and integrates them in the current picking cycle. This subsequently changes dynamically the stops on the order picker's picking route. The picker is constantly traveling through the warehouse and receives, using modern order-picking aids like pick-by-voice techniques or by a handheld terminal, new pick instructions that allow new orders to be included in the current picking cycle. This way of order picking saves set-up time and worker travel time, particularly for high order arrival rates which are often experienced in warehouses of e-commerce companies (Gong & De Koster, 2011).

A key for the quality of any order picking system is that the system achieves short (average) *order throughput times*, i.e. the time between a customer order entering the system and when the whole order is delivered at the depot from where it can be subsequently shipped to the customer. Short order throughput times in milkrun systems allow for faster customer response and improved customer satisfaction, which is important as companies are inclined to set their order cut-off times as late as possible while still guaranteeing that orders can be delivered next day or in some cases even the same day. It is important to note that the order throughput time strongly depends on the product allocation in the order picking area. A product (or storage) allocation method is a set of rules used to assign products to storage locations. Typically, an incoming customer order consists of one or more order

lines, each for a product stored at a different location within the order picking area. Therefore, in order to achieve short order throughput times in a milkrun picking system, it is essential to take the correlation between products that are ordered simultaneously into consideration and to place products that are strongly correlated in an optimal way in order to increase the probability that an incoming customer order can be included and fully picked in the current picking cycle.

In this chapter, we will study the average order throughput time in milkrun picking systems, i.e. the time between a customer order entering the system until the whole order is delivered at the depot. We determine the mean order throughput time of a customer order for three picking strategies; *exhaustive*, *locally-gated*, and *globally-gated*. We do this by applying and extending the framework of Van der Gaast et al. (2015) for studying batch sojourn-times in polling systems with simultaneous arrivals. Next, we propose an optimization framework for product allocation in a milkrun picking system in order to minimize the average order throughput time and compare the various strategies with each other. This chapter is the first in studying the average order throughput time of multi-line orders in a milkrun picking system. It considerably extends the work of Gong & De Koster (2008) who only considered waiting times of single-line orders, which is the time between the arrival of a customer order and the start of the pick of the single order line within the picking area. We model the milkrun picking system more accurately, which allows to help both designers and managers to create optimal design and control methods to improve the performance.

The organization of this chapter is as follows. In Section 6.2, milkrun picking systems are discussed and in Section 6.3 an overview of existing models for milkrun picking systems and product allocation in order picking systems is presented. In Section 6.4 a detailed description of the model and the corresponding notation used in this chapter is given. In Section 6.5 the analysis of the mean order throughput time for different picking strategies is provided. In Section 6.6 and Section 6.7 the optimization framework is presented which is used to decide how products should be allocated to the various storage positions in order to minimize the order throughput time of an incoming customer order. We extensively analyze the results of our model and optimization framework in Section 6.8 via computational experiments for a range of

parameters. Finally, in Section 6.9 we conclude and suggest some extensions of the model and further research topics.

6.2 Milkrun picking systems

Milkrun picking systems are a new method of order picking and are characterized by an order picker who is constantly traveling a fixed route along the aisles of a part or the entire order picking area, while picking all outstanding customer orders in batches that arrive in real-time. The order picker uses a pick cart or a tugger-, tow-train which typically has sufficient capacity to accommodate all these picks. In the case of online retailers, the route often finishes before the cart or train is full (Gong & De Koster, 2008). On the route the picker stops at all locations where products are located that need to be picked to fulfill the current outstanding customer orders. Afterwards, when the order picker reaches the depot, the products are disposed and sorted per customer order (i.e. a pick-and-sort system is used) and a new picking cycle will start.

In a milkrun system new picking instructions can be included in the current picking cycle that will dynamically change the locations where the order picker needs to stop. Using modern order-picking aids (e.g. a pick-by-voice system), the picker constantly receives updated information about incoming customer orders. In case a line of an incoming customer order is either located at the current stop or further downstream of the picking route, the picker can pick this line during the current picking cycle which allows for increased customer response. Otherwise, the requested order line is located upstream and will be picked in the next picking cycle.

Figure 6.1 highlights the differences between order throughput time, waiting time, and order sojourn time. When a new customer order is received, it generates demand for multiple order lines. The waiting time of an order line corresponds with the time between the customer order arrival and the start of the order picker picking the order line in the picking area. The order sojourn time corresponds with the time required picking all the order lines and, by definition, corresponds with the sojourn time of the last order line that is picked by the order picker (Van der Gaast et al., 2015). Finally, the order throughput time is the sum of the order sojourn time and the amount of time required after the last pick for the order picker to return to the depot.

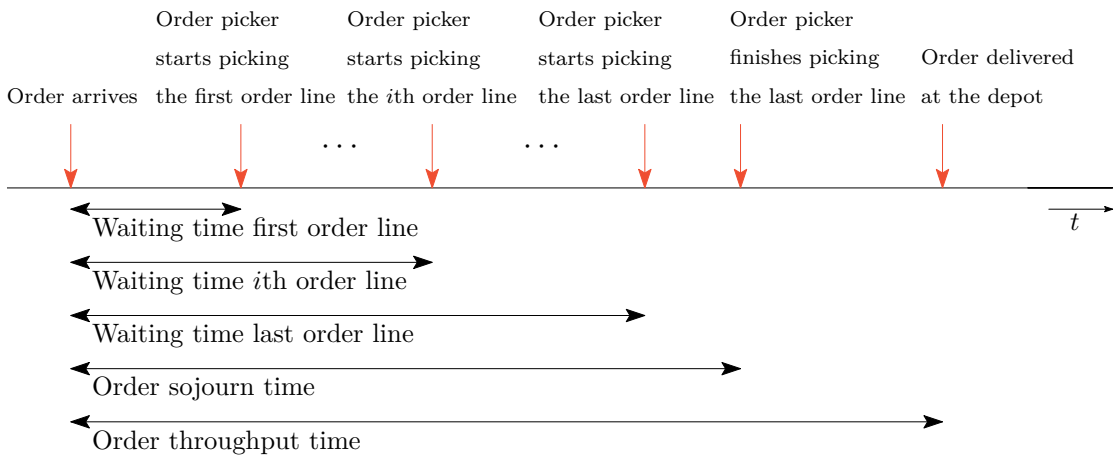
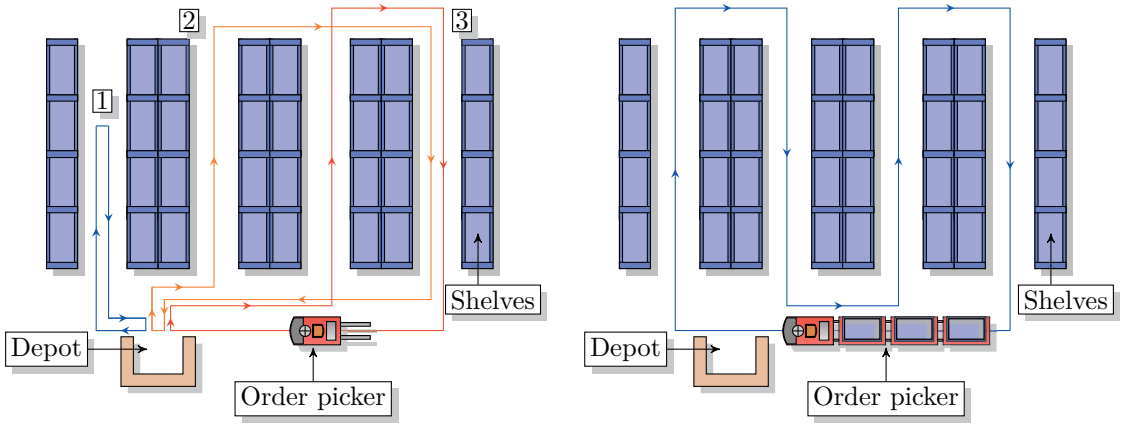


Figure 6.1: The differences between waiting times, order sojourn times, and order throughput times in a milkrun picking system.

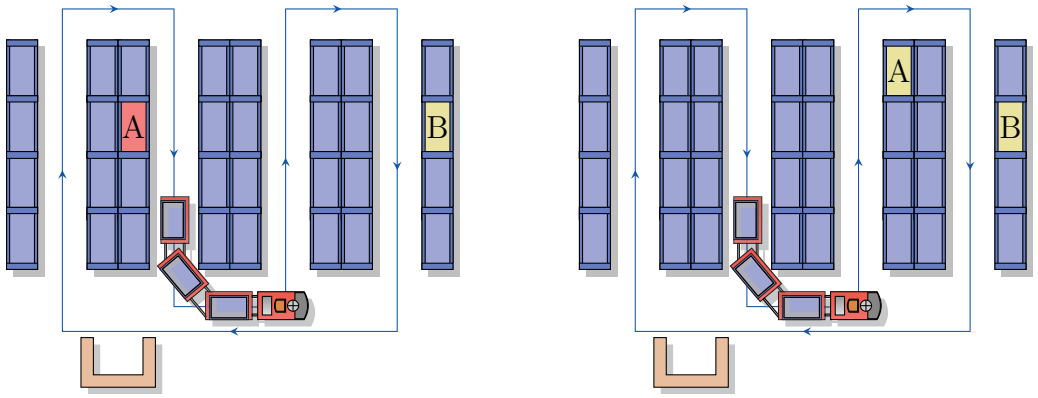
Figure 6.2 compares a milkrun picking system with a batch picking system. In case of batch picking the order picker waits at the depot until sufficient customer orders have entered the system and then starts a new picking tour. Typically, the route is constructed to minimize the total travel time traveled by the order picker, e.g. in Gademann & Van de Velde (2005). Examples of such picking routes are denoted by 1, 2, and 3 in Figure 6.2a. In Figure 6.2b the milkrun picking system is shown. In this case, the order picker visits the picking locations according to a strict S-shape routing policy in a cyclic sequence and immediately starts a new cycle after dropping off the picked products at the depot (Gong & De Koster, 2008). This means that every aisle is completely traversed during a picking cycle, because new customer orders can enter the system in real-time which subsequently change the stops where the order picker needs to pick products. Therefore, the order picker cannot skip entering an aisle like in conventional batch picking. The advantage of this is that after a picking cycle has started, a new arrived customer order can still be included in the current picking cycle if all its picking locations are downstream in the picking cycle. In a batch picking system the incoming customer order would only be picked in one of the following picking cycles. A drawback of a milkrun picking system can be increased travel distances for the order picker. However, when the arrival rate of new customer orders is high, like for e-commerce companies, the order picker in a batch picking system most likely also has to visit all aisles similar as in the milkrun picking system which will result in similar travel distances.



(a) Batch picking

(b) Milkrun picking

Figure 6.2: Comparison of batch and milkrun picking.



(a) Product A will be picked in the next picking cycle, whereas product B will be picked in the current picking cycle

(b) Product A and B will be picked in the current picking cycle

Figure 6.3: Product allocation in a milkrun picking system, where ▫ indicates products which are picked during the current picking cycle and ▫ indicates products which are picked during the next picking cycle.

A milkrun picking system with multi-line customer orders arriving in real-time can be accurately modeled using polling systems with simultaneous batch arrivals. Polling systems are multi-queue systems served by a single server who cyclically visits the queues in order to serve the customers waiting at these queues. Typically, when moving from one queue to another the server incurs a switch-over time. Polling systems have been applied in many areas such as computer and communication systems, production systems, and traffic and transportation systems (Takagi, 2000; Boon et al., 2011). In a milkrun picking system, the server is represented by the order picker and a queue by a storage location. At an arrival epoch, multiple customers enter the system and arrive in different queues at once. In the milkrun picking system this corresponds with an arriving multi-line customer order for which the order picker needs to pick multiple order lines at various storage locations in the order picking area.

In a milkrun picking system, the product allocation in the order picking area is important for performance. The product allocation method determines how products are assigned to the storage locations and directly influences the required time to pick a customer order. For example, in Figure 6.3, an incoming customer order arrives at the system which consists of demand for two products, A and B. In Figure 6.3a product A is located at a storage position which has already been visited in the current picking cycle and, therefore, cannot be picked anymore in the current picking cycle. Only at the end of the next picking cycle the incoming customer order will be delivered at the depot. On the other hand, in Figure 6.3b both locations where A and B are stored still need to be visited. Therefore, the incoming customer order can be included in the current picking cycle which allows the customer order to be picked faster. From this example it seems intuitive to store products or combinations of products that are requested frequently at the end of the picking route in order to increase the probability an incoming customer order can be fully picked during the current picking cycle. However, when the number of storage locations and different types of customer orders increases, finding the best product allocation becomes more complicated and requires the use of sophisticated methods as will be shown in Section 6.6 and Section 6.7.

6.3 Literature review

In the literature milkrun picking systems have not been researched often. Most papers on milkrun systems for internal logistics concern a production setting and in Section 6.3.1 we present a brief overview of these papers. Afterwards, in Section 6.3.2 we discuss product allocation in order picking systems, which has been studied intensively over the last decades. We will focus on correlated product assignment strategies for multi-line orders, similar as the setting studied in this chapter.

6.3.1 Milkrun systems for internal logistics

The literature on milkrun systems for internal logistics can be categorized into papers that study system performance using simulation methods, and the ones that use analytical models. Simulation methods are by far the most applied. Hanson & Finnsgård (2014) performed a case study in the automotive industry where forklifts were replaced by tow-trains operating according to a milkrun, as well as, where pallets were replaced by smaller plastic containers. The authors found that the milkrun increased smoothness in the material flows and increased the utilization of the warehouse workers. Staab et al. (2015) also studied a milkrun in the automotive industry and focused on different routing strategies to decrease the probability of congestion and blocking of tow-trains in order to improve the system performance. Korytkowski & Karkoszka (2015) examined a milkrun in an assembly line and studied the robustness of the system under various disturbances, such as delays in the supply cycle. They conclude that a milkrun system leads to a very stable supply of parts to the assembly line.

Analytical models for milkrun systems for internal logistics are more scarce. Bozer & Ciemnoczolowski (2013) and Ciemnoczolowski & Bozer (2013) analyzed a milkrun system that uses a Kanban system that tells which and how many materials should be delivered next to the work centers. The authors derive stability conditions for the system, since the number of materials that can be delivered per cycle is limited. They also studied the distribution of the number of containers delivered per milkrun, the effect of the number of Kanbans, and work station starvation. Emde & Boysen (2012) developed a nested dynamic programming procedure for studying the joint tow train

routing and scheduling problem in milkrun system for an automobile manufacturer. In Kilic & Durmusoglu (2013) a mixed-integer linear program was developed to analyze a milkrun system with equal cycle times. Due to the combinatorial structure of the problem, the authors propose a heuristic to minimize the work in process and transportation costs. Kovács (2011) studied storage assignment optimization in warehouses served by milkrun logistics. He showed that the milkrun system can achieve up to 36-38% improvement in order cycle time. In addition, the average picking effort was reduced compared if the classical cube-per-order index based strategy is used to allocate products..

The first to study milkrun picking systems (referred as a *dynamic order picking system* in the paper) is Gong & De Koster (2008). The authors used polling models and showed that the use of a milkrun picking system has a considerable advantage in on-time service completion over traditional batch picking. Boon et al. (2010) considered an efficient enhancement to an ordinary milkrun picking system that allows products stored at multiple locations. The location of the picker would then determine where specific order lines need to be picked. However, both papers only considered waiting times of order lines (or single-lined orders), which is the time between the arrival of a customer order and the start of a pick of an arbitrary order line within in the picking area. This statistic, however, does not capture the required time that is necessary for the order picker to return to the depot, neither does it provide the required time to pick a multi-line order. In this chapter a detailed analysis of the order throughput time will be provided.

6.3.2 Product allocation in order picking

A product allocation strategy is a set of rules used to assign products to their storage locations. Multiple strategies exist, each having a different impact on the performance of the order picking system, e.g. the required travel time to retrieve customer orders. In the literature, four strategies can be identified; randomized storage, dedicated storage, class-based storage, and correlated storage (Van den Berg, 1999). Especially the last policy is of great interest in case of multi-line orders, since information is used of which products are ordered together such that they can be stored together in order to reduce travel times for order picking.

Frazelle (1990) was the first to study correlated storage where pairs of products that are ordered the most frequent are stored close together. In order to solve the problem, the author formulated an integer program and proposed a two-phase heuristic algorithm. Kim (1993) also studied correlated storage and jointly determined storage locations and space requirements. The author showed that the algorithm developed in this paper outperforms the algorithm of Frazelle (1990). Garfinkel (2005) developed a mathematical model to minimize multi-zone orders while considering small-sized orders which can be picked in one route. The author proposed a Lagrangian relaxation solution approach and several heuristics to calculate upper bounds for the model. Lastly, Xiao & Zheng (2011) studied a correlated storage policy that stores products with demand dependencies together in order to minimize zone visits when picking materials or parts in a production line.

6.4 Model description

Consider a milkrun picking system as shown in Figure 6.4. We assume the order picking area to have a parallel aisle lay-out, with A aisles and L storage positions at each side of an aisle (a rack). Within an aisle, the order picker applies two-sided picking, i.e. simultaneous picking from the right and left sides within an aisle (De Koster et al., 1999). We denote the storage locations by $Q_1, \ldots, Q_N$, where the number of storage locations N equals $2AL$. Each storage location can be considered as a queue for order lines requesting the product stored on that location. Without loss of generality, we assume that the number of storage locations equals the number of different products stored in the warehouse, which can be accomplished by either increasing or decreasing the size of the order picking area. For the ease of presentation, all references to queue indices greater than N or less than 1 are implicitly assumed to be modulo N, e.g., Q_{N+1} is understood as Q_1. The order picker visits all queues in a cyclic sequence and picks all required products for the outstanding customer orders to a pick cart or tow-train. We assume the number of products the order picker can pick per picking cycle is unconstrained, as for online retailers the route often finishes before the cart or train is full (Gong & De Koster, 2008). Therefore, we model the milkrun picking system as a polling system with simultaneous batch arrivals (Van der Gaast et al., 2015).

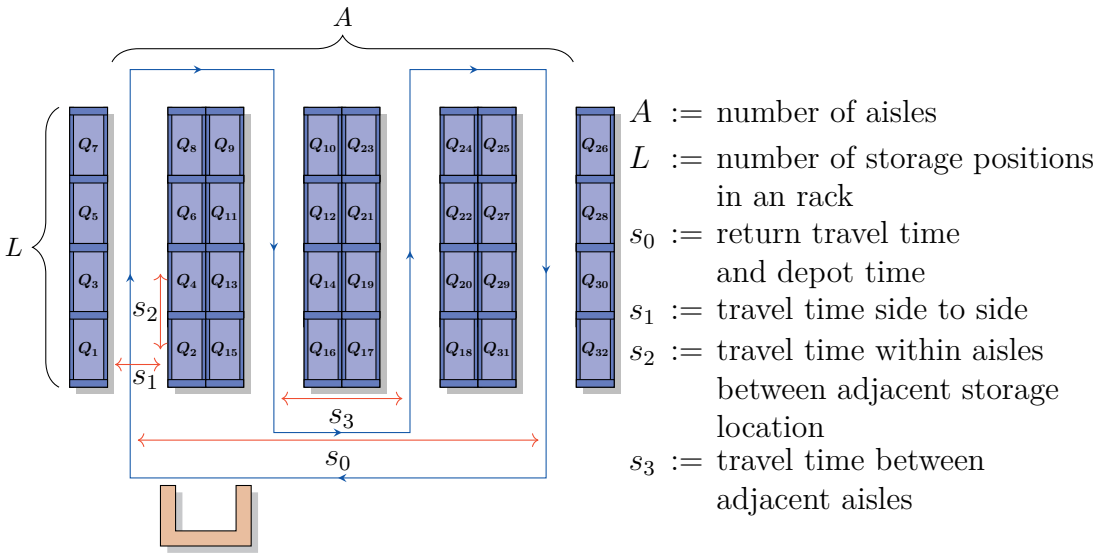


Figure 6.4: Overview of the milkrun picking system.

New customer orders arrive at the system according to a Poisson process with rate λ. Each customer order is of size $\boldsymbol{D} = (D_1, \ldots, D_N)$, where D_j, $j = 1, \ldots, N$ represents the number of units of product j is requested. Let $\boldsymbol{K} = \Phi(\boldsymbol{D})$, where $\Phi : \mathbb{N}^N \to \mathbb{N}^N$. Mapping Φ defines the product allocation of the products to their storage locations and is given as follows,

$$\Phi(\boldsymbol{D}) = \boldsymbol{D}x, \tag{6.1}$$

where $x_{ij} \in \mathbb{N}^{N \times N}$ with $x_{ij} = 1$ if product j is allocated to storage location i and 0 otherwise. Then, for each order $\boldsymbol{K} = (K_1, \ldots, K_N)$ where K_i represents the number of units that need to be picked at Q_i, $i = 1, \ldots, N$. The random vector $\boldsymbol{K}$ is assumed to be independent of past and future arriving epochs and for every realization at least one product needs to be picked. The support with all possible realizations of $\boldsymbol{K}$ is denoted by $\mathcal{K}$ and let $\boldsymbol{k} = (k_1, \ldots, k_N)$ be a realization of $\boldsymbol{K}$. The joint probability distribution of $\boldsymbol{K}$ is denoted by $\pi(\boldsymbol{k}) = \mathbb{P}(K_1 = k_1, \ldots, K_N = k_N)$. The arrival rate of products that need to be picked at Q_i is denoted by $\lambda_i = \lambda E(K_i)$. The total arrival rate of products to be picked for the customer orders arriving in the system is given by $\Lambda = \sum_{i=1}^{N} \lambda_i$. The order throughput time of a customer order $\boldsymbol{k} = (k_1, \ldots, k_N)$ is denoted by $T_{\boldsymbol{k}}$ and is defined as the time between its arrival

epoch until the order has been fully picked and delivered at the depot. The order throughput time of an arbitrary customer order is denoted by T.

At each queue, the picker picks the product units on a First-Come First-Served (FCFS) basis. The picking times are assumed to be independent and identically distributed random variables with finite first and second moments. The picking time of a product unit in Q_i is a generally distributed random variable B_i with first and second moment $E(B_i)$ and $E(B_i^2)$, respectively. The workload at Q_i, $i = 1, \ldots, N$ is defined by $\rho_i = \lambda_i E(B_i)$; the overall system load by $\rho = \sum_{i=1}^{N} \rho_i$. For the system to be stable a necessary and sufficient condition is that $\rho < 1$ (Takagi, 1986), which is assumed to be the case in the remainder of this chapter.

When the order picker moves from Q_i to Q_{i+1}, he or she takes a generally distributed travel time S_i with first and second moment $E(S_i)$ and $E(S_i^2)$. Without loss of generality, we assume that the travel times from side to side are independent and identically distributed with mean s_1 and second moment s_1^2, the travel times within aisles between two adjacent storage locations have mean s_2 and second moment s_2^2, whereas the time required to travel from one aisle to the next one has mean s_3 and second moment s_3^2. Finally, after visiting the last queue the order picker returns to the first queue to start a new cycle. On the way, the order picker visits the depot where he or she will drop off the picked products so that other operators can sort and transport them. We assume that this time is independent of the number of products the picker picked and is included in s_0 and its second moment s_0^2. Summarizing we have,

$$E(S_i) = \begin{cases} s_1, & \text{if } i = 2jL+1, 2jL+3, \ldots, 2L(j+1)-1,\ j = 0, 1, \ldots, A-1, \\ s_2, & \text{if } i = 2jL+2, 2jL+4, \ldots, 2L(j+1)-2,\ j = 0, 1, \ldots, A-1, \\ s_3, & \text{if } i = 2jL,\ j = 1, \ldots, A-1, \\ s_0, & \text{if } i = N, \end{cases}$$

and for the second moments,

$$E\left(S_i^2\right) = \begin{cases} s_1^2, & \text{if } i = 2jL+1, 2jL+3, \ldots, 2L\left(j+1\right)-1,\ j = 0, 1, \ldots, A-1, \\ s_2^2, & \text{if } i = 2jL+2, 2jL+4, \ldots, 2L\left(j+1\right)-2,\ j = 0, 1, \ldots, A-1, \\ s_3^2, & \text{if } i = 2jL,\ j = 1, \ldots, A-1, \\ s_0^2, & \text{if } i = N. \end{cases}$$

Let $E\left(S\right) = \sum_{i=1}^{N} E\left(S_i\right)$ be the total expected travel time in a cycle and $E(S^2) = \sum_{i=1}^{N} E(S_i^2) + \sum_{i \neq j} E\left(S_i\right) E\left(S_j\right)$ its second moment.

We define a picking cycle from the service beginning at the first queue until the order picker has delivered all the picked products at the depot and arrives at the first queue again. Therefore, a picking cycle C consists of N visit periods, V_i, each followed by a travel time S_i; $V_1, S_1, V_2, \ldots, V_i, S_i, \ldots, V_N, S_N$. A visit period V_i starts with a pick of a product unit and ends after the last product has been picked given that product units need to be picked at Q_i, afterwards the order picker travels to the next picking location of which the duration is S_i. In case no product units need to be picked at Q_i the order picker immediately travels to the next picking location. The total mean duration of a picking cycle is independent of the queues involved (and the service discipline) and is given by (see, e.g., Takagi (1986)) $E\left(C\right) = E\left(S\right) / \left(1-\rho\right)$. Finally, we assume replenishment is not required in a picking cycle, and each queue has infinite buffer capacity.

We consider in this chapter three different picking strategies; exhaustive, locally-gated, and globally-gated. In Figure 6.5 the differences between the three strategies are shown for an example. Assume a new customer order enters the system with demand for three products, 1, 2, and 3, stored at queues Q_j, Q_i, and Q_N respectively. The order picker is currently busy picking products at Q_j, which is also where the product units for product 1 need to be picked.

Under the *exhaustive strategy* as shown in Figure 6.5a, the order picker picks all product units at the current queue until no product units need to be picked anymore. This also includes demand for the product that arrives while the picker is busy picking at this queue. Therefore, in the example the order picker will pick all three products within the current cycle. Thus, the order throughput time of this order equals the residual time in V_j, visit periods $V_{j+1}, \ldots, V_N$, and travel times $S_j, \ldots, S_N$.

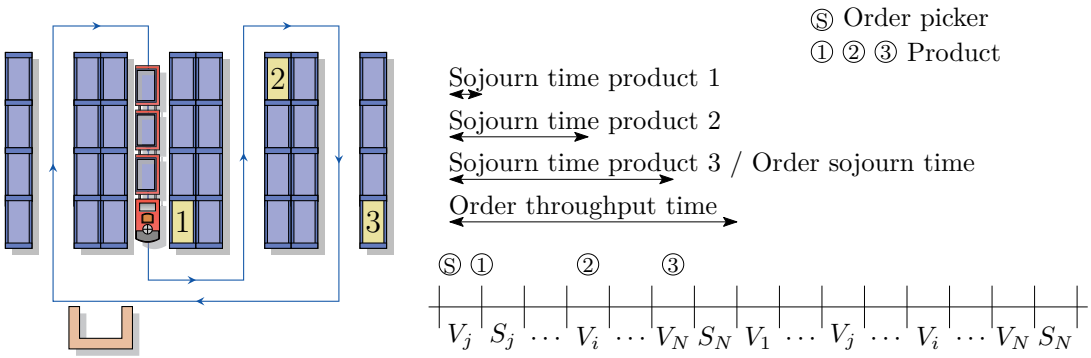
On the other hand, under the *locally-gated strategy* shown in Figure 6.5b, the order picker only picks the product units that need to be picked at the start of the first pick at a queue; all demand that arrives during the course of the visit will be picked in the next visit. In this example, this means products 2 and 3 will be picked in the current cycle and product 1 in the next cycle. Therefore, the order throughput time of this order equals the residual time in V_j, visit periods $V_{j+1}, \ldots, V_N, V_1, \ldots, V_N$, and travel times $S_j, \ldots, S_N, S_1, \ldots, S_N$.

Finally, for the *globally-gated strategy* shown in Figure 6.5c the picker will not pick any products of incoming customer orders that arrived during the current picking cycle. Only after the start of the next picking cycle these incoming orders will be picked. Note that this strategy is identical to classical batch picking, given that the order picker has to visit all the picking locations during a picking tour and immediately goes on a next tour after delivering the products at the depot. Similar as in batch picking no orders can be included during the current picking cycle. In the example, no products will be picked during the current cycle and the customer order will be delivered at the depot at the end of the next cycle.

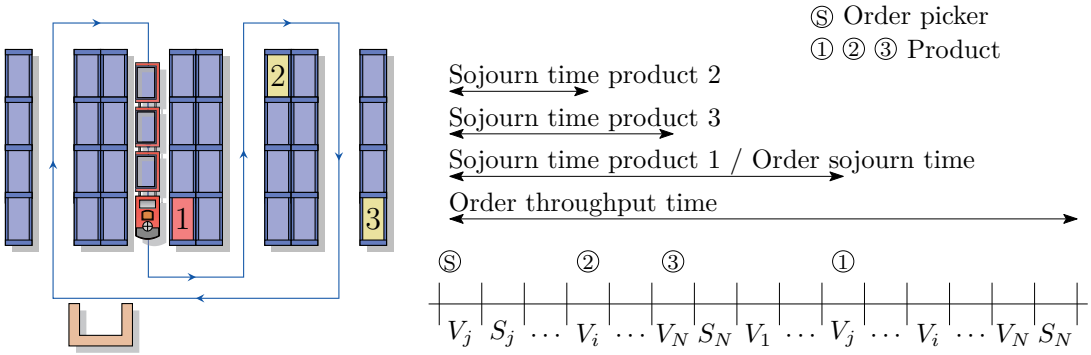
In the previous example we have seen that whether a customer order is fully picked in the same cycle it arrives or otherwise in the next cycle *depends* on the location of the server and the service discipline. Therefore, let $\mathcal{K}_j^0$ and $\mathcal{K}_j^1$, $j = 1, \ldots, N$ be subsets of support $\mathcal{K}$, defined as

$$\mathcal{K}_j^0 = \{k_1 = 0, \ldots, k_{j-1} = 0, k_j \geq 0, k_{j+1} \geq 0, \ldots, k_N \geq 0\} \in \mathcal{K},$$

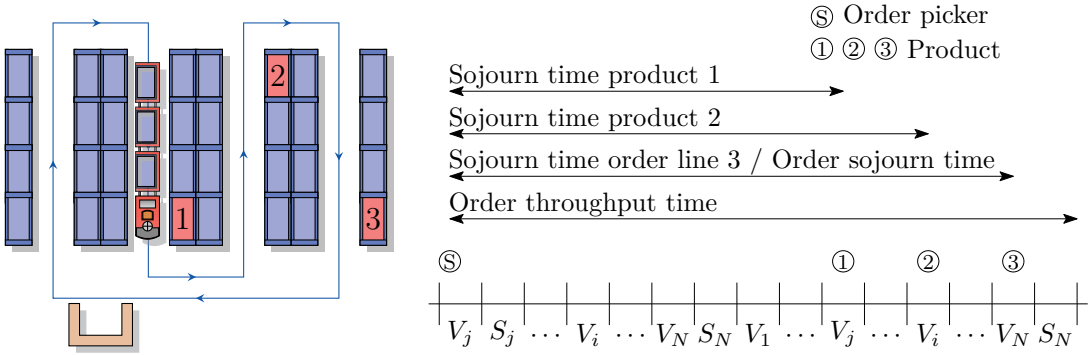
and $\mathcal{K}_j^1 = \left(\mathcal{K}_j^0\right)^c$ as its complement such that for $j = 1, \ldots, N$ we have $\mathcal{K}_j^0 \cup \mathcal{K}_j^1 = \mathcal{K}$ and let the associated probabilities be $\pi\left(\mathcal{K}_j^0\right)$ and $\pi\left(\mathcal{K}_j^1\right)$. The interpretation of $\boldsymbol{k} \in \mathcal{K}_j^0$ is that for an incoming customer order all the products need to be picked at $Q_j, \ldots, Q_N$. For example, in case of the exhaustive strategy this means if the order picker is at $Q_1, \ldots, Q_j$ a customer order $\boldsymbol{k} \in \mathcal{K}_j^0$ can be included in the current picking cycle, whereas if $\boldsymbol{k} \in \mathcal{K}_j^1$ the order will be completed in the next cycle. Finally, let $E\left(K_i|\mathcal{K}_j^0\right)$ and $E\left(K_i|\mathcal{K}_j^1\right)$ be the conditional mean number of product units that need to picked in Q_i, $i = 1, \ldots, N$ given subset $\mathcal{K}_j^0$ or $\mathcal{K}_j^1$.



(a) Exhaustive strategy; all the products are picked during the current cycle



(b) Locally-gated strategy; product 2 and 3 are picked in the current cycle and product 1 in the next cycle



(c) Globally-gated strategy; all the products are picked during the next cycle

Figure 6.5: Description of the order throughput time for different picking strategies, where ▭ indicates products which are picked during the current picking cycle and ▭ indicates products which are picked during the next picking cycle.

6.5 Mean order throughput time

In this section the mean order throughput time will be derived in case of the exhaustive strategy in Section 6.5.1, locally-gated strategy in Section 6.5.2, and globally-gated strategy in Section 6.5.3. In Van der Gaast et al. (2015) the batch sojourn time (order sojourn time) for polling systems with simultaneous batch arrivals has been studied, but in this chapter we will extend this framework for analyzing the order throughput time.

6.5.1 Exhaustive strategy

In order to derive the mean order throughput time for the exhaustive strategy we can use the *Mean Value Analysis* (MVA) developed by Van der Gaast et al. (2015) for polling systems with simultaneous arrivals. MVA for polling systems was originally developed by Winands et al. (2006) to study mean waiting times in systems with individually arriving customers. Van der Gaast et al. (2015) showed how this assumption can be relaxed and used to calculate other statistics like the batch sojourn-time. Compared to other methods as the buffer occupancy method (Takagi, 1986) and the descendant set approach (Konheim et al., 1994), MVA has the advantage that it has a pure probabilistic interpretation and is based on standard queueing results, i.e., the Poisson arrivals see time averages (PASTA) property (Wolff, 1982) and Little's Law (Little, 1961).

In MVA a set of N^2 linear equations is derived for calculating $E\left(\bar{L}_i^{(S_{j-1},V_j)}\right)$ (Equations (5.45) and (5.46) in Chapter 5), the conditional mean queue-length at Q_i (excluding the potential product unit that is being picked) at an arbitrary epoch within travel period S_{j-1} and visit period V_j, that subsequently can be used to calculate the mean performance statistics for the system. Therefore, the MVA also allows us to determine the order throughput time for a specific and arbitrary customer order.

For notation purposes we introduce θ_j in this section as shorthand for intervisit period (S_{j-1}, V_j); the mean duration of this period $E(\theta_j)$ is given by,

$$E(\theta_j) = E(S_{j-1}) + E(V_j), \quad j = 1, \ldots, N, \tag{6.2}$$

where $E(V_j) = \rho_j E(C)$ and $\sum_{j=1}^{N} E(\theta_j) = E(C)$.

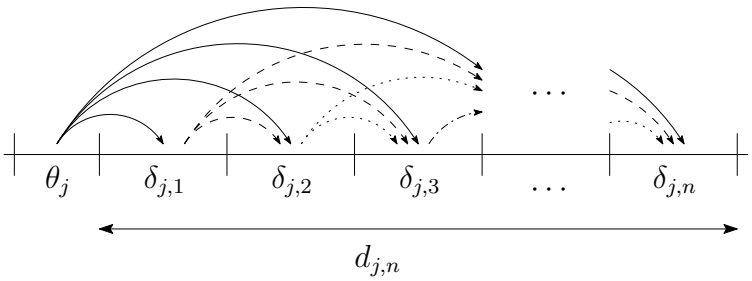


Figure 6.6: Description of $d_{j,n}$.

In addition, we denote by $D_{j,n}$ the total mean work of product units in $Q_{j+1}, \ldots, Q_{j+n}$ which originate from customer orders that arrive per unit of pick time B_j or travel time S_{j-1} and all the subsequent picks that are triggered by these picks before the picker finishes service in Q_{j+n}. For example, a single product pick in Q_j will generate on average additional work in $Q_{j+1}, \ldots, Q_{j+n}$ of duration $E(B_j)\, d_{j,n}$. Let $d_{j,0} = 0$ and for $n > 0$ we have,

$$d_{j,n} = \sum_{m=1}^{n} \delta_{j,m}, \quad j = 1, \ldots, N, \tag{6.3}$$

where $\delta_{j,m}$ is the contribution for Q_{j+m}. First, $\delta_{j,1} = \rho_{j+1}/(1-\rho_{j+1})$ includes the mean picking times and the consecutive busy periods in Q_{j+1} of product units that arrived during a product pick B_j or travel time S_{j-1}. Then, $\delta_{j,2} = (1+\delta_{j,1})\,\rho_{j+2}/(1-\rho_{j+2})$ contains the mean picking times of the product units that arrived in Q_{j+2} during B_j or S_{j-1} and the previous busy periods in Q_{j+1} plus all the busy periods that these picks generate in Q_{j+2}. In general we can write $\delta_{j,n}$ for $n > 0$ as (see Figure 6.6),

$$\delta_{j,n} = \sum_{m=1}^{\min(N-1,n)} \delta_{j,n-m} \frac{\rho_{j+n}}{1-\rho_{j+n}}, \quad j = 1, \ldots, N, \tag{6.4}$$

where $\delta_{j,0} = 1$. Note that $\delta_{j,n}$ only depends on at most $N-1$ previous $\delta_{j,n-m}$'s because if new demand arrives at the queue that is currently being visited it will be picked before the end of the current visit.

With the help of $E\left(\bar{L}_i^{(\theta_j)}\right)$, where $\theta_j = (S_{j-1}, V_j)$ and $D_{j,n}$ it is now possible to calculate the mean order throughput time $E(T_{\boldsymbol{k}})$ for customer order $\boldsymbol{k}$. We do this by explicitly conditioning on the location of the order picker and by studying the evolution of the system through time until the incoming customer order has been fully delivered at the depot. We can write $E(T_{\boldsymbol{k}})$ as,

$$E(T_{\boldsymbol{k}}) = \frac{1}{E(C)} \sum_{j=1}^{N} E(\theta_j) \left(1_{\left(\boldsymbol{k} \in \mathcal{K}_j^0\right)} E\left(T_{\boldsymbol{k}}^{(\theta_j,0)}\right) + 1_{\left(\boldsymbol{k} \in \mathcal{K}_j^1\right)} E\left(T_{\boldsymbol{k}}^{(\theta_j,1)}\right) \right), \tag{6.5}$$

where $1_{\left(\boldsymbol{k} \in \mathcal{K}_j^0\right)}$ and $1_{\left(\boldsymbol{k} \in \mathcal{K}_j^1\right)}$ are indicator functions that are equal to one if $\boldsymbol{k}$ is in subset $\mathcal{K}_j^0$ or $\mathcal{K}_j^1$ respectively; and zero otherwise. Whenever the order picker is at intervisit period θ_j and still can pick all the products of the incoming customer order (i.e. $\boldsymbol{k} \in \mathcal{K}_j^0$), then the order throughput time is equal to $E\left(T_{\boldsymbol{k}}^{(\theta_j,0)}\right)$ which is the mean time until the order picker reaches the depot during the current cycle while picking customer order $\boldsymbol{k}$. Otherwise, one or more products are located upstream and the order throughput time is equal to $E\left(T_{\boldsymbol{k}}^{(\theta_j,1)}\right)$ which is the expected time until the order picker reaches the depot in the next cycle while picking customer order $\boldsymbol{k}$.

First, we focus on the derivation of $E\left(T_{\boldsymbol{k}}^{(\theta_j,0)}\right)$. When the customer order enters the system in intervisit period θ_j with probabilities $E(V_j)/E(\theta_j)$ and $E(S_{j-1})/E(\theta_j)$ it has to wait for a residual picking time $E\left(B_j^R\right) = E\left(B_j^2\right)/(2E(B_j))$ or residual travel time $E\left(S_{j-1}^R\right) = E\left(S_{j-1}^2\right)/(2E(S_{j-1}))$. Also, it has to wait for $E\left(\bar{L}_j^{(\theta_j)}\right)$ product units that still need to be picked at Q_j, as well as at this queue k_j product units need be picked for the arriving customer order $\boldsymbol{k}$. Each of these picks triggers a busy period of $E(B_j)/(1-\rho_j)$ and generates additional picks that will be made before the end of the current cycle of duration $d_{j,N-j}E(B_j)/(1-\rho_j)$. This also applies for the residual picking time and residual travel time. Then, for each subsequent intervisit period θ_l, $l = j+1, \ldots, N$, the travel time from Q_{l-1} to Q_l will trigger a busy period and additional picks in $Q_l, \ldots, Q_N$ of duration $E(S_{l-1})(1+d_{l,N-l})/(1-\rho_l)$. Similarly, the product units that already needed to be picked before customer order $\boldsymbol{k}$ entered the system and the k_l product units need be picked for the arriving customer order $\boldsymbol{k}$ will increase the mean order throughput time by $\left[E\left(\bar{L}_l^{(\theta_j)}\right) + k_l\right] E(B_l)(1+d_{l,N-l})/(1-\rho_l)$. Finally, the picked order has to be delivered to the depot the duration of which is $E(S_N)$.

Combining this gives the following expression for the mean time until the order picker reaches the depot during the current cycle while picking customer order $\boldsymbol{k}$,

$$E\left(T_{\boldsymbol{k}}^{(\theta_j,0)}\right) = \left(\frac{E\left(V_j\right)}{E\left(\theta_j\right)}E\left(B_j^R\right) + \frac{E\left(S_{j-1}\right)}{E\left(\theta_j\right)}E\left(S_{j-1}^R\right) + \left[E\left(\bar{L}_j^{(\theta_j)}\right) + k_j\right]E\left(B_j\right)\right)$$
$$\times \frac{1+d_{j,N-j}}{1-\rho_j} + \sum_{l=1}^{N-j}\left(E\left(S_{j+l-1}\right) + \left[E\left(\bar{L}_{j+l}^{(\theta_j)}\right) + k_{j+l}\right]E\left(B_{j+l}\right)\right)$$
$$\times \frac{1+d_{j+l,N-j-l}}{1-\rho_{j+l}} + E\left(S_N\right). \quad (6.6)$$

Next we focus on $E\left(T_{\boldsymbol{k}}^{(\theta_j,1)}\right)$. The derivation is similar to the one of Equation (6.6), except that we should also consider the additional demand that is generated during a pick or a switch from queue to queue until the end of the next picking cycle. This gives the following expression,

$$E\left(T_{\boldsymbol{k}}^{(\theta_j,1)}\right) = \left(\frac{E\left(V_j\right)}{E\left(\theta_j\right)}E\left(B_j^R\right) + \frac{E\left(S_{j-1}\right)}{E\left(\theta_j\right)}E\left(S_{j-1}^R\right) + \left[E\left(\bar{L}_j^{(\theta_j)}\right) + k_j\right]E\left(B_j\right)\right)$$
$$\times\frac{1+d_{j,2N-j}}{1-\rho_j} + \sum_{l=1}^{N-1}\left(E\left(S_{j+l-1}\right) + \left[E\left(\bar{L}_{j+l}^{(\theta_j)}\right) + k_{j+l}\right]E\left(B_{j+l}\right)\right)$$
$$\times\frac{1+d_{j+l,2N-j-l}}{1-\rho_{j+l}} + \sum_{l=N}^{2N-j}E\left(S_{j+l-1}\right)\frac{1+d_{j+l,2N-j-l}}{1-\rho_{j+l}} + E\left(S_N\right), \quad (6.7)$$

From (6.6) and (6.7) we can see that both equations consist of terms independent of $\boldsymbol{k}$ and terms that depend on $\boldsymbol{k}$. Therefore, we can rewrite the expressions as follows,

$$E\left(T_{\boldsymbol{k}}^{(\theta_j,0)}\right) = E\left(C^{(\theta_j,0)}\right) + \sum_{l=0}^{N-j}k_{j+l}E\left(B_{j+l}\right)\frac{1+d_{j+l,N-j+l}}{1-\rho_{j+l}}, \quad (6.8)$$

$$E\left(T_{\boldsymbol{k}}^{(\theta_j,1)}\right) = E\left(C^{(\theta_j,1)}\right) + \sum_{l=0}^{N-1}k_{j+l}E\left(B_{j+l}\right)\frac{1+d_{j+l,2N-j+l}}{1-\rho_{j+l}}, \quad (6.9)$$

where $E\left(C^{(\theta_j,0)}\right)$ can be interpreted as the mean residual cycle time given that the order picker is at intervisit period θ_j, whereas $E\left(C^{(\theta_j,1)}\right)$ also includes the duration of the next cycle. Then, the second terms are the total contribution the incoming customer order makes to the order throughput time.

Finally, the mean order throughput time of an arbitrary customer order is obtained by multiplying $E(T_{\boldsymbol{k}})$ with the probability that a particular customer order $\boldsymbol{k}$ is placed,

$$E\left(T^{EX}\right) = \sum_{\boldsymbol{k}\in\mathcal{K}} \pi(\boldsymbol{k})\, E(T_{\boldsymbol{k}}). \tag{6.10}$$

By (6.5) and $\sum_{\boldsymbol{k}\in\mathcal{K}} \pi(\boldsymbol{k})\, 1_{\left(\boldsymbol{k}\in\mathcal{K}_j^0\right)} = \pi\left(\mathcal{K}_j^0\right)$ and $\sum_{\boldsymbol{k}\in\mathcal{K}} \pi(\boldsymbol{k})\, 1_{\left(\boldsymbol{k}\in\mathcal{K}_j^1\right)} = \pi\left(\mathcal{K}_j^1\right)$ we can rewrite (6.10) as follows,

$$\begin{aligned} E\left(T^{EX}\right) &= \sum_{\boldsymbol{k}\in\mathcal{K}} \pi(\boldsymbol{k}) \sum_{j=1}^{N} \frac{E(\theta_j)}{E(C)} \left(1_{\left(\boldsymbol{k}\in\mathcal{K}_j^0\right)} E\left(T_{\boldsymbol{k}}^{(\theta_j,0)}\right) + 1_{\left(\boldsymbol{k}\in\mathcal{K}_j^1\right)} E\left(T_{\boldsymbol{k}}^{(\theta_j,1)}\right)\right), \\ &= \frac{1}{E(C)} \sum_{j=1}^{N} E(\theta_j) \left(\pi\left(\mathcal{K}_j^0\right) E\left(T^{(\theta_j,0)}\right) + \pi\left(\mathcal{K}_j^1\right) E\left(T^{(\theta_j,1)}\right)\right), \end{aligned} \tag{6.11}$$

where for $j = 1, \ldots, N$, $E\left(T^{(\theta_j,0)}\right)$ and $E\left(T^{(\theta_j,1)}\right)$ are given as follows,

$$\begin{aligned} E\left(T^{(\theta_j,0)}\right) &= E\left(C^{(\theta_j,0)}\right) + \sum_{l=0}^{N-j} E\left(K_{j+l}|\mathcal{K}_j^0\right) E(B_{j+l}) \frac{1 + d_{j+l,N-j+l}}{1-\rho_{j+l}}, \\ E\left(T^{(\theta_j,1)}\right) &= E\left(C^{(\theta_j,1)}\right) + \sum_{l=0}^{N-1} E\left(K_{j+l}|\mathcal{K}_j^1\right) E(B_{j+l}) \frac{1 + d_{j+l,2N-j+l}}{1-\rho_{j+l}}. \end{aligned}$$

Then, $E\left(T^{EX}\right)$ can be evaluated by determining $\pi\left(\mathcal{K}_j^0\right)$, $\pi\left(\mathcal{K}_j^1\right)$, $E\left(K_i|\mathcal{K}_j^0\right)$, and $E\left(K_i|\mathcal{K}_j^1\right)$, $i, j = 1, \ldots, N$ and using the MVA for the conditional queue length probabilities $E\left(\bar{L}_i^{(\theta_j)}\right)$, $i, j = 1, \ldots, N$. Afterwards, $E\left(T^{(\theta_j,0)}\right)$ and $E\left(T^{(\theta_j,1)}\right)$ can be obtained such that $E\left(T^{EX}\right)$ can be calculated.

6.5.2 Locally-gated strategy

The mean order throughput time in case of the locally-gated strategy can also be determined using the MVA developed by Van der Gaast et al. (2015). For the locally-gated policy, per queue all incoming demand is placed before a gate. Only at the start of a visit period at a queue, all product units that need to be picked at this location are placed behind the gate which means that the order picker will

pick these product units in the current picking cycle. For this we slightly redefine $\mathcal{K}_j^0 = \{k_1 = 0, \ldots, k_{j-1} = 0, k_j = 0, k_{j+1} \geq 0, \ldots, k_N \geq 0\} \in \mathcal{K}$ to reflect this change.

First, we introduce θ_j in this section as shorthand for intervisit period (V_j, S_j); the expected duration of this period $E(\theta_j)$ is given by,

$$E(\theta_j) = E(V_j) + E(S_j), \quad j = 1, \ldots, N. \tag{6.12}$$

In contrast to the exhaustive strategy, we have to make a distinction between the mean number of product units before and behind the gate. We introduce variables $E\left(\tilde{L}_i^{(\theta_j)}\right)$, $i, j = 1, \ldots, N$ as the conditional mean queue-length of product units located before the gate in Q_i during intervisit period θ_j and $E\left(\hat{L}_i^{(\theta_i)}\right)$, $i = 1, \ldots, N$ as conditional mean queue-length of product units located behind the gate in Q_i during intervisit period θ_i. In MVA by Van der Gaast et al. (2015) a set of $N(N+1)$ linear equations (Equations (5.76) and (5.77) in Chapter 5) is derived for calculating these conditional mean queue-lengths, which we will use in order to determine the order throughput time.

Similar as for the exhaustive policy, we introduce $d_{j,n}$ which is defined as (6.3). Because of the different picking strategy, $\delta_{j,n}$ is different. First, $\delta_{j,1} = \rho_{j+1}$ contains the mean picking times of all product units in Q_{j+1} that arrive per unit of a product pick B_j or a travel time S_j, whereas $\delta_{j,2} = \rho_{j+2}(1 + \delta_{j,1})$ also includes the mean picking times of the product units that arrived in Q_{j+2} during a product pick B_j or a travel time S_j, as well as in $\delta_{j,1}$. In general we can write $\delta_{j,n}$ for $n > 0$ as,

$$\delta_{j,n} = \sum_{m=1}^{\min(N,n)} \delta_{j,n-m}\rho_{j+n}, \quad j = 1, \ldots, N. \tag{6.13}$$

where $\delta_{j,0} = 1$. In this case $\delta_{j,n}$ depends on N previous $\delta_{j,n-m}$'s because if new demand arrives at the queue that is currently being visited it will not be picked during the current cycle.

Conditioning on the location of the order picker, by (6.5) the order throughput time $E(T_{\boldsymbol{k}})$ of customer order $\boldsymbol{k}$ can now be determined. First, we consider $E\left(T_{\boldsymbol{k}}^{(\theta_j,0)}\right)$ in case customer order $\boldsymbol{k}$ arrives during intervisit period θ_j and will be fully picked and delivered to the depot at the end of the current cycle. With probability $E(V_j)/E(\theta_j)$ the arriving customer order has to wait for the order picker to finish the current pick

and the travel time to the next queue, whereas with probability $E(S_j)/E(\theta_j)$ the order only has wait for the residual travel time. Also, there are $E\left(\hat{L}_i^{(\theta_i)}\right)$ product units behind the gate that need still to be picked of which each pick has duration $E(B_j)$. During the residual time in θ_j new demand is generated at $Q_{j+1},\ldots,Q_N$ that will be picked before the end of the current picking cycle. Then, for Q_l, $l=j+1,\ldots,N$, $E\left(\tilde{L}_l^{(\theta_j)}\right)$ product units need to be picked for customer orders that entered the system before order $\boldsymbol{k}$, as well as k_l product units for order $\boldsymbol{k}$ where each pick has duration $E(B_l)$ and new demand is generated at the queues that still need to be visited during the current cycle. Similar during all the remaining travel times, new demand is generated that will be picked before the order picker reaches the depot. This gives the following expression,

$$E\left(T_{\boldsymbol{k}}^{(\theta_j,0)}\right) = \left(\frac{E(V_j)}{E(\theta_j)}\left(E\left(B_j^R\right)+E(S_j)\right)+\frac{E(S_j)}{E(\theta_j)}E\left(S_j^R\right)+E\left(\hat{L}_j^{(\theta_j)}\right)E(B_j)\right)$$
$$\times\left(1+d_{j,N-j}\right)+\sum_{l=1}^{N-j}\left(\left[E\left(\tilde{L}_{j+l}^{(\theta_j)}\right)+k_{j+l}\right]E(B_{j+l})+E(S_{j+l})\right)$$
$$\times\left(1+d_{j+l,N-j-l}\right). \quad (6.14)$$

The derivation of $E\left(T_{\boldsymbol{k}}^{(\theta_j,1)}\right)$ is similar to the one of (6.14), except that customer order $\boldsymbol{k}$ will be delivered at the depot the next picking cycle. Therefore, we should also consider the additional demand that is generated during a pick or a switch from queue to queue until the end of the next picking cycle. This gives the following expression,

$$E\left(T_{\boldsymbol{k}}^{(\theta_j,1)}\right) = \left(\frac{E(V_j)}{E(\theta_j)}\left(E\left(B_j^R\right)+E(S_j)\right)+\frac{E(S_j)}{E(\theta_j)}E\left(S_j^R\right)+E\left(\hat{L}_i^{(\theta_i)}\right)E(B_i)\right)$$
$$\times\left(1+d_{j,2N-j}\right)+\sum_{l=1}^{N}\left(\left[E\left(\tilde{L}_{j+l}^{(\theta_j)}\right)+k_{j+l}\right]E(B_{j+l})+E(S_{j+l})\right)$$
$$\times\left(1+d_{j+l,2N-j-l}\right)+\sum_{l=N+1}^{2N-j}E(S_{j+l})\left(1+d_{j+l,2N-j-l}\right). \quad (6.15)$$

The same decomposition as (6.8)-(6.9) also holds for (6.14) and (6.15),

$$E\left(T_{\boldsymbol{k}}^{(\theta_j,0)}\right) = E\left(C^{(\theta_j,0)}\right) + \sum_{l=1}^{N-j} k_{j+l} E\left(B_{j+l}\right)\left(1 + d_{j+l,N-j+l}\right),$$
$$E\left(T_{\boldsymbol{k}}^{(\theta_j,1)}\right) = E\left(C^{(\theta_j,1)}\right) + \sum_{l=1}^{N} k_{j+l} E\left(B_{j+l}\right)\left(1 + d_{j+l,2N-j+l}\right).$$

The mean order throughput time of an arbitrary customer order can also be obtained similarly as for the exhaustive strategy. Therefore using (6.5) and some rewriting, we have

$$E\left(T^{LG}\right) = \frac{1}{E\left(C\right)} \sum_{j=1}^{N} E\left(\theta_j\right)\left[\pi\left(\mathcal{K}_j^0\right) E\left(T^{(\theta_j,0)}\right) + \pi\left(\mathcal{K}_j^1\right) E\left(T^{(\theta_j,1)}\right)\right], \quad (6.16)$$

where for $j = 1, \ldots, N$, $E\left(T^{(\theta_j,0)}\right)$ and $E\left(T^{(\theta_j,1)}\right)$ are given as follows,

$$E\left(T^{(\theta_j,0)}\right) = E\left(C^{(\theta_j,0)}\right) + \sum_{l=1}^{N-j} E\left(K_{j+l}|\mathcal{K}_j^0\right) E\left(B_{j+l}\right)\left(1 + d_{j+l,N-j+l}\right),$$
$$E\left(T^{(\theta_j,1)}\right) = E\left(C^{(\theta_j,1)}\right) + \sum_{l=1}^{N} E\left(K_{j+l}|\mathcal{K}_j^1\right) E\left(B_{j+l}\right)\left(1 + d_{j+l,2N-j+l}\right).$$

6.5.3 Globally-gated strategy

The final strategy for which we derive the mean order throughput time is the globally-gated strategy. This strategy resembles locally-gated except that we only pick the product units that need to be picked during the start of a picking cycle, instead of the start of a visit period to a queue. This implies that every incoming customer order will only be picked during the next picking cycle. As a result, the analysis of this strategy is more straightforward compared to the other two strategies.

The mean order throughput time of a specific customer order $\boldsymbol{k}$ can be determined as follows. First, the incoming order first has to wait for the current residual cycle time. Then, the duration of the next picking cycle equals all the picks for incoming orders that already arrived at the system before order $\boldsymbol{k}$ in the same cycle and those that arrived during the residual cycle time. In addition, the duration of all the picks for order $\boldsymbol{k}$ and the total travel time in one cycle increase the mean order throughput

time. This gives the following expression,

$$E\left(T_{\boldsymbol{k}}\right) = E\left(C_R\right) + \sum_{j=1}^{N} \lambda_j E\left(B_j\right)\left(E\left(C_P\right) + E\left(C_R\right)\right) + \sum_{j=1}^{N} E\left(S_j\right) + \sum_{j=1}^{N} k_j E\left(B_j\right), \quad (6.17)$$

where $E\left(C_P\right)$ and $E\left(C_R\right)$ are the mean past and residual cycle time.

From Van der Gaast et al. (2015) we know that $E\left(C_P\right) = E\left(C_R\right) = E\left(C^2\right) / \left(2E\left(C\right)\right)$, where $E\left(C\right) = E\left(S\right) / \left(1 - \rho\right)$ and

$$E\left(C^2\right) = \frac{1}{\left(1-\rho^2\right)}\Big[E\left(S^2\right) + 2\rho E\left(S\right) E\left(C\right) + \sum_{j=1}^{N} \lambda_j E\left(B_j^2\right) E\left(C\right) + \sum_{i=1}^{N} \lambda \left(E\left(K_i^2\right) - E\left(K_i\right)\right) E\left(B_i\right)^2 E\left(C\right) + \sum_{i,j:i\neq j}^{N} \lambda E\left(K_i K_j\right) E\left(B_i\right) E\left(B_j\right) E\left(C\right)\Big]. \quad (6.18)$$

Now, we can rewrite (6.17) as follows,

$$E\left(T_{\boldsymbol{k}}\right) = \left(1 + 2\rho\right) \frac{E\left(C^2\right)}{2E\left(C\right)} + E\left(S\right) + \sum_{j=1}^{N} k_j E\left(B_j\right). \quad (6.19)$$

Finally, the mean order throughput time $E\left(T^{GG}\right)$ can obtained by multiplying $E\left(T_{\boldsymbol{k}}\right)$ with all possible demand realizations, which gives

$$E\left(T^{GG}\right) = \left(1 + 2\rho\right) \frac{E\left(C^2\right)}{2E\left(C\right)} + E\left(S\right) + \sum_{j=1}^{N} E\left(K_j\right) E\left(B_j\right). \quad (6.20)$$

6.6 Optimization model for product allocation

The performance of the milkrun picking system is largely dependent on the product allocation, as explained in Section 6.2. A good product allocation allows many customer orders being picked in the current picking cycle and delivering them to the depot as soon as possible in order to achieve short order throughput times. Therefore,

given the mapping of (6.1) we formulate an optimization model to find a product allocation x that minimizes the mean order throughput time. For each of the three picking strategies we minimize the mean order throughput time $E\left(T^d(x)\right)$, where $d \in \{EX, LG, GG\}$ denotes the picking strategy and x defines the product allocation (see (6.1)). As explained in Section 6.4, the mean order throughput time depends on allocation x since the allocation determines how many units on average should be picked per storage location, $E(K_i)$, $i = 1, \ldots, N$.

The model to minimize $E\left(T^d, x\right)$ is defined as the following integer programming model;

$$\text{minimize} \quad E\left(T^d(x)\right) \tag{6.21}$$

$$\text{subject to} \quad \sum_{j=1}^{N} x_{ij} = 1 \quad \text{for all } i \in N \tag{6.22}$$

$$\sum_{i=1}^{N} x_{ij} = 1 \quad \text{for all } j \in N \tag{6.23}$$

$$x_{ij} \in \{0, 1\} \quad \text{for all } i, j \in N \tag{6.24}$$

The objective of the model (6.21) is to minimize the mean order throughput time ((6.11), (6.16), or (6.20) evaluated for product allocation x) given picking strategy d. Constraints (6.22) ensure that each storage location has only one type of product assigned to it. On the other hand, constraints (6.23) define that each type of product should be stored at only one storage location. Finally, constraints (6.24) are the integrality constraints.

Since objective function (6.21) is nonlinear, we cannot apply standard integer programming techniques to find the product allocation that minimizes the mean order throughput time. Therefore, in the next section we introduce a meta-heuristic that overcomes this issue.

6.7 A meta-heuristic for product allocation

In order to solve the nonlinear optimization problem of Section 6.6 we apply a Genetic Algorithm to obtain a product allocation that minimizes the mean order throughput

time. Genetic Algorithms (GA) have been used successfully to solve nonlinear optimization problems for which exact or exhaustive methods are not feasible because of the prohibitive complexity of the problem and have already been applied in many different fields (see for examples in the context of order picking Tsai et al. (2008) and Bottani et al. (2012)). First developed by Holland (1975), a GA is an adaptive heuristic search algorithm inspired by the laws of natural selection and genetics. The idea of a GA is to start with an initial population of chromosomes, each representing a different product allocation, and to calculate the fitness of each chromosome by evaluating the objective function of (6.21). Then, the chromosomes with the highest level of fitness have the highest probability of surviving and producing offspring which in turn will form the basis of the next generation. Over the course of several generations, fitter chromosomes will be present within the population that will provide better product allocations.

The first step of the GA is to describe the population of chromosomes, as well as, how to calculate the fitness of each chromosome. We denote H^g as the g-th generation population, where

$$H^g = \{y_1^g, y_2^g, \ldots, y_l^g, \ldots, y_M^g\}, \tag{6.25}$$

consists in total of M different chromosomes each representing a product allocation and chromosome $y_l^g = \left\{y_{l,1}^g, y_{l,2}^g, \ldots, y_{l,j}^g, \ldots, y_{l,N}^g\right\}$, where allele $y_{l,j}^g$ denotes the allocated storage location for product j. In order to calculate the fitness of chromosome y_l^g, we determine its associated product allocation x_l^g such that we can evaluate $E\left(T^d, x_l^g\right)$ for a given picking strategy d. For this we define $x_l^g = \psi\left(y_l^g\right)$, where $\psi : \mathbb{N}^N \to \{0,1\}^{N\times N}$. The mapping ψ is given as follows,

$$\psi\left(y_l^g\right) = \left[e_{y_{l,1}^g}, e_{y_{l,2}^g}, \ldots, e_{y_{l,N}^g}\right], \tag{6.26}$$

where e_j denotes a column vector of length N with 1 in the j-th position and 0 in every other position. The fitness of population H^g, $F\left(H^g\right)$, can now be calculated by

$$\begin{aligned} F\left(H^g\right) &= \left\{F\left(y_1^g\right), F\left(y_2^g\right), \ldots, F\left(y_M^g\right)\right\} \\ &= \left\{E\left(T^d\left(\psi\left(y_1^g\right)\right)\right), E\left(T^d\left(\psi\left(y_2^g\right)\right)\right), \ldots, E\left(T^d\left(\psi\left(y_M^g\right)\right)\right)\right\}. \end{aligned} \tag{6.27}$$

In order to construct the next generation of chromosomes, we select using the current generation a survivor and an offspring population that together form the next generation. First, survivors are chromosomes that are selected from the current population and are then placed in the next generation. Second, offspring is created by mutating and/or recombining current chromosomes in order to create new product allocations. For the offspring population, we select chromosomes based on roulette-wheel selection, also known as stochastic sampling with replacement (Mitchell, 1998). This method determines for each chromosome a probability proportional to its fitness as follows,

$$p_l = \frac{F\left(y_l^g\right)}{\sum_{j=1}^{M} F\left(y_j^g\right)}, \quad l = 1, \ldots, M. \tag{6.28}$$

Then, chromosomes with a higher probability have a higher chance of being selected to be used to generate offspring. For the survivor population, we use tournament selection. In this method t_{size} chromosomes are randomly selected and then the chromosome with best fitness will be chosen to generate the survivor population. Finally, the size of the survivor and offspring population is controlled by parameter $0 \leq \alpha \leq 1$. In every generation $\lfloor \alpha M \rfloor$ chromosomes are selected to generate offspring, whereas $M - \lfloor \alpha M \rfloor$ selected chromosomes will become the survivor population.

The offspring is generated using a combination of two types of genetic operators; *recombination* and *mutation*. First, recombination generates new chromosomes by combining different parts of more than one parent chromosomes. The new child chromosome is constructed by selecting one or more crossover points in the parent chromosomes, splitting the chromosomes at these points, and then recombining the parts to construct the new chromosome while ensuring that the new chromosome provides a feasible product allocation. Second, mutation allows the population to be diversified which is essential to avoid that the search terminates at a local minimum. A mutation is carried out by altering one or more alleles from their original state of a single chromosome in order to form a new allocation.

In general, the structure of the GA can be described as follows;

In line 1 of the algorithm the generation index is set to zero and in line 2 the initial population of chromosomes is created. The population is initialized with random product allocations, like most GA applications (Reeves, 2003). The size of the

Algorithm 6.1 Description of the Genetic Algorithm.

1: $g \leftarrow 0$
2: Initialize the initial population, H^0
3: Calculate the initial fitness, $F(H^0)$
4: **while** maximum number of generations not met or no convergence achieved **do**
5: $\quad g \leftarrow g + 1$
6: $\quad H_s^g \leftarrow survivors(H^{g-1})$
7: $\quad H_o^g \leftarrow offspring(H^{g-1})$
8: $\quad H_g \leftarrow H_s^g \cup H_o^g$
9: $\quad$ Calculate the fitness, $F(H^g)$
10: **return** $\psi\left(y_{best}\right)$ ▷ The best product allocation over all generations

population equals M and should be big enough to allow enough variation between the chromosomes, but should not be too big since otherwise it would take a lot of time to find a good solution. In line 3 the initial fitness of the population is calculated using Equation (6.27). Then, the generation index is increased at line 5, whereas in line 6 and 7 the survivor and offspring population are generated. The genetic operators used to generate the offspring population are applied in sequence and each operator has a probability that determines how many chromosomes on average per generation the operator is applied on. Afterwards, both populations are combined in order to form the next generation. Lines 5-9 are repeated until the termination condition has been triggered. In our GA the algorithm stops if either the best solution found by the algorithm has not been improved for G_{stable} generations or if the generation index has reached G_{max}. Finally, at the last line the best product allocation over all the generations $\psi\left(y_{best}\right)$ is returned.

The details of each genetic operator will be discussed in the next three sections. These operators were carefully chosen after running an initial test to allow for sufficient recombination and mutation in every generation.

6.7.1 Swap mutation (SM)

The first operator used in the GA is the swap mutation (SM). The SM operator chooses one random allele in a chromosome and swaps it with one of the remaining alleles of the chromosome. The main function of this operator is to allow the heuristic to explore different parts of the search space.

Suppose that in chromosome y_l^g allele $y_{l,i}^g$ and $y_{l,j}^g$ are selected to be swapped. This results in a new chromosome $y_l^{g'}$;

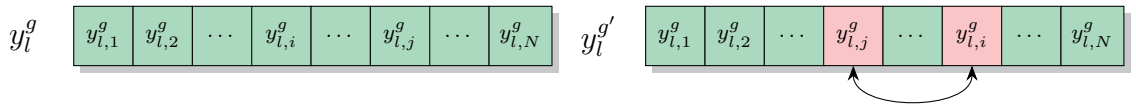
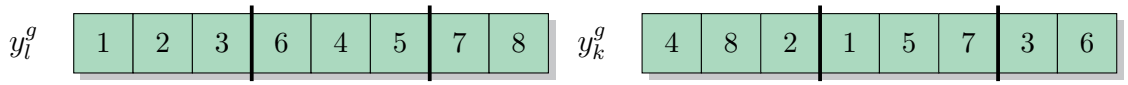


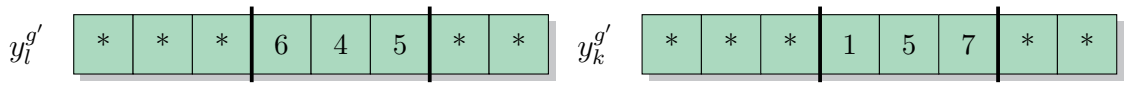
Figure 6.7: Example of the swap mutation operator.

6.7.2 Partially matched crossover (PMX)

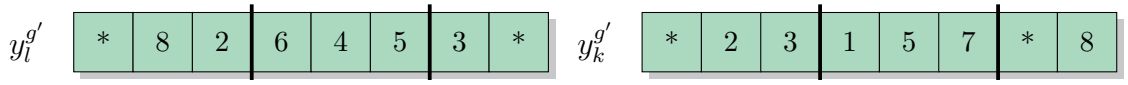
The second operator used is a partially matched crossover (PMX). PMX is recombination operator that uses a subset of alleles between two randomly chosen cut points from one parent and completes the remaining part of the child chromosome by preserving the order and positions of as many storage locations as possible from the other parent. In the following figure an example of the PMX operator is shown.



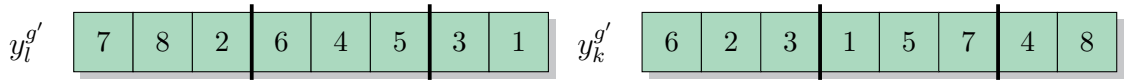
(a) Assume two parent chromosomes y_l^g and y_k^g each consisting of 8 alleles and two cut points per chromosome.



(b) By preserving the subset of alleles between the two cuts points two children $y_l^{g'}$ and $y_k^{g'}$ can be constructed with the following mapping: $6 \leftrightarrow 1$, $4 \leftrightarrow 5$, $5 \leftrightarrow 7$. Let $*$ denote an allele for which its storage location is not determined yet.



(c) The empty alleles of $y_l^{g'}$ can be determined (if possible) with alleles from y_k^g that are on the same positions and vice versa.



(d) Using the mapping the remaining alleles of $y_l^{g'}$ and $y_k^{g'}$ can be determined such that both child chromosomes provide a feasible product allocation.

Figure 6.8: Example of the partially matched crossover operator.

6.7.3 Edge recombination crossover (ERX)

The third operator is the edge recombination crossover (ERX). The idea of the ERX operator is to construct a new offspring that inherits as many edges (a combination of two subsequent alleles) as possible from its parent chromosomes. The first step of the operator is to create for the alleles an edge map based on their neighborhood. The neighborhood of an allele is defined as the alleles that are adjacent to it either in the first and/or the second parent. Afterwards, starting from an arbitrary allele, in each step the next allele is chosen that is in the neighborhood of the previous allele. If more than one allele is feasible, then randomly the allele with smallest neighborhood size is selected. This continues until the entire child chromosome is constructed. In the following figure an example of the ERX operator is shown.

(a) Assume two parent chromosomes y_l^g and y_k^g each consisting of 8 alleles.

1	*2, 5, 8*	3	*2, 6, 7*	5	*1, 4, 7*	7	*3, 5, 8*
2	*1, 3, 8*	4	*6, 5, 8*	6	*3, 4*	8	*1, 2, 4, 7*

(b) Based on the two parent chromosomes the edge map can be constructed which contains for each allele the adjacent alleles from the parent chromosomes.

(c) We start by constructing child $y_l^{g'}$ from the first allele of y_l^g. Then, the second allele is either 2, 5, or 8. Both 2 and 5 have three neighbors, whereas 8 has four neighbors. Assume that 2 is randomly chosen. In the same way 3 is chosen for the third allele.

(d) By continuing in the same manner, we finally obtain $y_l^{g'}$. In case we started with the first allele of y_k^g we would obtain $y_k^{g'}$.

Figure 6.9: Example of the edge recombination crossover operator.

6.8 Numerical results

In this section we study the mean order throughput time for the three picking strategies and under which product allocation policies these times are minimized.

Afterwards, we check for which range of system instances a particular picking strategy achieves the shortest mean order throughput times.

This section is split in two parts. Section 6.8.1 investigates for a large test set of different instances the solution quality and accuracy of the meta-heuristic of Section 6.7. Furthermore, we compare the results of the different picking strategies and discuss whether products that are often ordered together should be stored close to each other. Afterwards, Section 6.8.2 discusses a real world application for which we compare the three picking strategies and product allocation policies.

All the experiments were run on Core i7 with 2.5 GHz and 8 GB of RAM and the Genetic Algorithm was implemented in Java.

6.8.1 Comparison different system instances

In order to find out which product allocation policy minimizes the mean order throughput time given one of the picking strategies, a test set was generated for which the parameters are shown in Table 6.1.

Table 6.1: Parameters of the system instances test set.

Parameter	Values
Picking times, b	0.1 sec., 1.0 sec., 2 sec.
Traveling times, s	0.1 sec., 1.0 sec., 2 sec.
Number of different orders, $\lvert\mathcal{K}\rvert$	5 orders, 20 orders, 35 orders
Order sizes, $\sum_{i=1}^{N} k_i$	1–2 units, 2–5 units, 5–10 units
Overall system load, ρ	0.1, 0.5, 0.8, 0.95

First, for all instances the number of aisles A was assumed to be equal to 2 and the storage locations per rack in an aisle L was also equal to 2, which in total gives 8 different storage locations ($= 2AL$). We have chosen this number since it allows us to enumerate all possible product allocation policies ($8! = 40,320$ different combinations) in a reasonable time per instance in order to assess the solution quality and accuracy of the Genetic Algorithm (GA). Next, we assumed all picking times to be equal and exponentially distributed, i.e. $E(B_i) = b$ and $E\left(B_i^2\right) = 2b^2$ for $i = 1, \ldots, N$, and the values varied between 0.1, 1.0, and 2.0 seconds. The same assumption was also

made for the travel times between storage locations, $E(S_i) = s$ and $E\left(S_i^2\right) = 2s^2$ for $i = 1, \ldots, N$. Note that the actual values of the picking and traveling times are not of concern in this section, however we are interested in the situation that the picking times are faster than the traveling times or vice versa. Furthermore, the overall system load ρ was 0.1, 0.5, 0.8, or 0.95, such that for the arrival rate it holds that $\lambda = \rho / \left(b \sum_{i=1}^{N} E(K_i)\right)$ which is independent of the current product allocation and where $\sum_{i=1}^{N} E(K_i)$ is the expected order size. Next, we varied the number of different customer orders that arrive at the system at $|\mathcal{K}| = 5$, 20, or 35. For each of these orders we varied the demanded number of product units, $\sum_{i=1}^{N} k_i$, between only small order sizes (randomly chosen between 1–2 product units), medium order sizes (2–5 product units), or large order sizes (5–10 product units). In addition, we generated per number of customer orders $|\mathcal{K}|$ and order size $\sum_{i=1}^{N} k_i$ three sets of customer order probabilities $\pi(\boldsymbol{k})$ summing to 1, where each probability varied between 2% and 20%. In total this leads to 972 ($3 \times 3 \times 4 \times 3 \times 3 \times 3$) different (symmetric) instances.

In addition, we generated the same amount of (asymmetric) instances in which the picking and traveling times differ per location. The only difference with the symmetric instances is that each individual picking and traveling time was randomly perturbed between -10% and 10% of its current value while ensuring that a product allocation can be found such that the system is stable. Finally, note that because of different picking times per storage location the system load is now dependent on the product allocation ($\rho = \sum_{i=1}^{N} \lambda E(K_i) E(B_i)$).

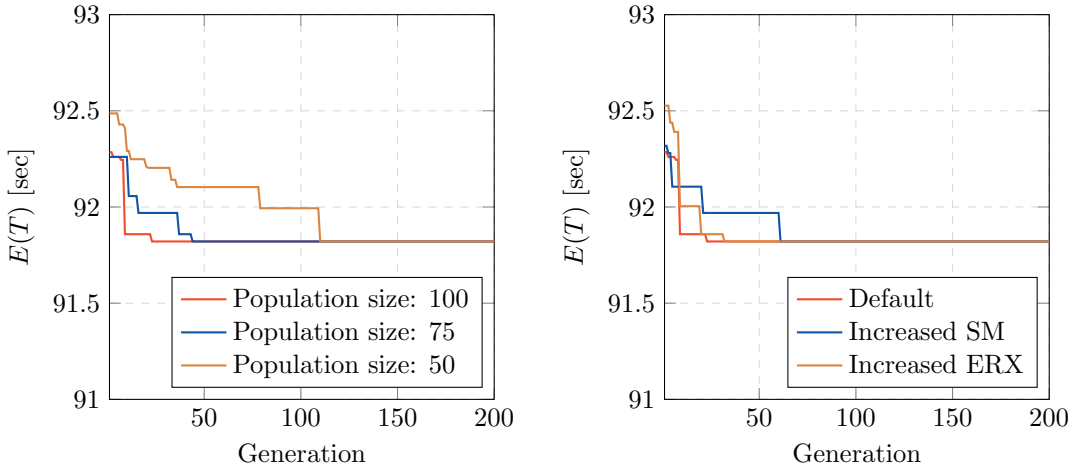
The parameters used in the GA are shown in Table 6.2.

Table 6.2: Parameters used in the Genetic Algorithm.

Parameter	Value	Parameter	Value
Population size, M	100	Probability p_{SM}	0.15
Stable generations, G_{stable}	150	Probability p_{PMX}	0.35
Maximum number of generations, G_{max}	1,000	Probability p_{ERX}	0.20
Tournament size, t_{size}	3	Offspring parameter α	0.6

In every generation the size of the population equals $M = 100$ which allows for enough variation between chromosomes. The two stopping criteria, G_{stable} and G_{max} are set equal to 150 and 1,000 respectively, and the tournament size t_{size} is set equal

to 3. Finally, each genetic operator has a probability that determines on how many chromosomes the operator is applied on average per generation. Since the operators are applied sequentially some chromosomes in the offspring population might not be modified and will remain unchanged in the next generation. The probabilities are 0.15 for the swap mutation, 0.35 for the partially matched crossover, and 0.20 for the edge recombination crossover. These parameters were obtained by running a sensitivity analysis on a preliminary data set of similar sized instances in order to avoid over-fitting on the current test set.



(a) Different population size M **(b)** Different genetic operator probabilities

Figure 6.10: The effect of different parameters used in the Genetic Algorithm.

Figure 6.10 shows the effect of different parameters used in the GA for a random instance in the test set. In Figure 6.10a we apply the GA to find the best allocation under the exhaustive strategy for three different population sizes M. In all three cases the optimal solution is found, but it can be seen that the larger the population size the faster the optimal solution is found in terms of number of generations. Increasing the population size even further does not lead to finding the optimal solution in a significant less amount of generations, however it mainly slows down the algorithm since it requires more time to evaluate all the chromosomes per generation. In Figure 6.10b the effect of different probabilities for the genetic operators are shown for the same instance, where default refers to the algorithm using the parameters listed in Table 6.2. First, by increasing the swap probability to 0.5 the operator diversifies the population substantially, but decreases the overall solution quality

on the long run, because good areas in the search space are less explored. When the probability of the edge recombination crossover is set to 0.5 we observe that the optimal solution is found slightly later. Typically, by increasing the level of recombination the algorithm will spend more time investigating a limited part of the search space, the consequence of which is that the optimal solution is often found later.

Table 6.3: Solution quality and accuracy of the Genetic Algorithm on the test set.

	Exhaustive	Locally-gated	Globally-gated
Symmetric instances			
Average GA time (sec.)	3.26	2.97	< 0.01
Average enumeration time (sec.)	22.19	20.30	< 0.01
Average number of generations	189.2	188.9	-
Solution quality (%)	94	93	100
Relative difference solution (%)	0.21	0.15	-
Asymmetric instances			
Average GA time (sec.)	3.12	2.76	0.24
Average enumeration time (sec.)	23.50	19.45	1.35
Average number of generations	184.2	183.3	177.7
Solution quality (%)	93	95	100
Relative difference solution (%)	0.30	0.32	-

In Table 6.3 the solution quality and accuracy of the GA for both the symmetric and asymmetric test set are shown. The average run time of the GA was around 3 seconds for the exhaustive and locally-gated strategy, whereas the average time to evaluate all the $40,320$ product allocations is around $19 - 24$ seconds. For the symmetric instances with the globally-gated strategy, all product allocations have the same mean order throughput time since in (6.20) both $E(C)$, $E(C^2)$, and $\sum_{j=1}^{N} E(K_j) E(B_j)$ will always be the same. Therefore, there is no need to run the GA nor to enumerate all possible allocations for these instances. For the asymmetric instances, this is not the case and the average run time of the GA is 0.24 seconds and 1.35 seconds for full enumeration. On average 40 generations are needed to find the best allocation plus the additional 150 iterations to ensure no better solution is found. In terms of solution quality, GA was able to find 94% and 93% of the optimal product allocations for the symmetric and asymmetric instances. For the cases where the optimal solution

was not found, GA still found solutions very close to the optimal solution; the average relative difference with the optimal solution value for these cases was 0.21% for the symmetric and 0.30% for the asymmetric instances. For locally-gated, GA found 93% and 95% of the optimal allocations for the symmetric and asymmetric instances, and the relative difference for the non-optimal solutions was 0.15% and 0.32%. Finally, GA was able to find all the optimal solutions for the asymmetric instances with globally-gated.

Table 6.4: The joint probabilities, σ_{ij}, that product i (row) and product j (column) are picked for a customer order.

(a) Symmetric instances (order size 1–2 product units, locally-gated and exhaustive strategy)

	Q_1	Q_2	Q_3	Q_4	Q_5	Q_6	Q_7	Q_8
Q_1	-	0.44	0.11	0.67	0.44	0.33	0.56	0.33
Q_2		-	0.33	0.44	0.22	0.44	0.56	0.67
Q_3			-	0.44	0.11	0.11	0.67	0.22
Q_4				-	0.44	0.44	0.33	0.67
Q_5					-	0.56	0.56	0.67
Q_6						-	0.67	0.33
Q_7							-	0.56
Q_8								-

(b) Asymmetric instances (order size 1–2 product units, locally-gated and exhaustive strategy)

	Q_1	Q_2	Q_3	Q_4	Q_5	Q_6	Q_7	Q_8
Q_1	-	0.67	0.11	0.78	0.33	0.11	0.22	0.33
Q_2		-	0.22	0.78	0.33	0.33	0.67	0.33
Q_3			-	0.22	0.33	0.33	0.56	0.44
Q_4				-	0.22	0.56	0.78	0.56
Q_5					-	0.11	0.56	0.33
Q_6						-	0.67	0.33
Q_7							-	0.67
Q_8								-

In Table 6.4 the probabilities, σ_{ij} are shown; σ_{ij} is the joint probability that product i (row) and product j (column) are picked for a customer order. These probabilities are shown for both the symmetric and asymmetric instances in case of an order size 1–2

products and locally-gated and exhaustive strategies. We excluded globally-gated, since in case of the symmetric cases all production allocations have the same average order throughput time. In addition, by only considering small order sizes we can easily investigate whether products that are often ordered together are stored close to each other. For the symmetric cases, it can be seen that products that are ordered together tend to be stored close to each other ($\sigma_{i,i+1}$), and also occur often with products at the last two storage locations ($\sigma_{i.7}$ and $\sigma_{i.8}$). This can mainly be explained by the trade-off between workload balancing (the server should not stay at a storage position too long) and allocating correlated products next to each other (increasing the probability an order can be picked in the same cycle it arrives). The previous results can also be observed for the asymmetric instances.

Finally, in Table 6.5 we investigate the range of instances a particular picking strategy achieves the shortest mean order throughput times. For given system load ρ, traveling time s, and picking time b, the table presents the fraction of times a particular picking strategy achieves the shortest mean order throughput times. The results from the full enumeration have been used to construct this table, however the same results are obtained if the GA would have been used. First, in Table 6.5a the results for the symmetric instances are presented. Note that the best allocation of products can differ per strategy. From the table it can be seen that when the system load is low and the picking and traveling times are the same the exhaustive strategy achieves the shortest mean order throughput times. This is also the case for all system loads when the traveling times are longer than the picking times. In these cases, it is more beneficial to stay longer at a picking location than to switch to another picking location. However, the opposite holds when the traveling times are shorter than the picking times. For these instances both gated strategies perform better and the higher the load of the system globally-gated performs the best. A reason for this is that in locally-gated the order picker will already pick many products for orders that will only be delivered at the depot next cycle, whereas in globally-gated only products will be picked for orders that will be delivered at the depot at the end of the cycle. Finally, the same patterns can also be observed for the asymmetric instances in Table 6.5b.

Table 6.5: For picking strategy exhaustive (EX), globally-gated (GG), and locally-gated (LG), the fraction of times this strategy achieves the minimal mean order throughput time given system load ρ, traveling time s, and picking time b.

(a) Symmetric instances

	b	0.10			1.00			2.00		
s	ρ	EX	GG	LG	EX	GG	LG	EX	GG	LG
0.10	0.10	0.96	0.00	0.04	0.00	0.41	0.59	0.00	0.63	0.37
	0.50	0.74	0.26	0.00	0.00	0.96	0.04	0.00	1.00	0.00
	0.80	0.67	0.30	0.04	0.00	1.00	0.00	0.00	1.00	0.00
	0.95	0.67	0.33	0.00	0.00	1.00	0.00	0.00	1.00	0.00
1.00	0.10	1.00	0.00	0.00	0.96	0.00	0.04	0.63	0.04	0.33
	0.50	1.00	0.00	0.00	0.74	0.26	0.00	0.37	0.33	0.30
	0.80	1.00	0.00	0.00	0.67	0.30	0.04	0.26	0.63	0.11
	0.95	1.00	0.00	0.00	0.67	0.33	0.00	0.11	0.63	0.26
2.00	0.10	1.00	0.00	0.00	1.00	0.00	0.00	0.96	0.00	0.04
	0.50	1.00	0.00	0.00	0.96	0.04	0.00	0.74	0.26	0.00
	0.80	1.00	0.00	0.00	0.85	0.15	0.00	0.67	0.30	0.04
	0.95	1.00	0.00	0.00	0.78	0.22	0.00	0.67	0.33	0.00

(b) Asymmetric instances

	b	0.10			1.00			2.00		
s	ρ	EX	GG	LG	EX	GG	LG	EX	GG	LG
0.10	0.10	1.00	0.00	0.00	0.00	0.37	0.63	0.00	0.59	0.41
	0.50	0.81	0.19	0.00	0.00	0.93	0.07	0.00	1.00	0.00
	0.80	0.70	0.26	0.04	0.00	1.00	0.00	0.00	1.00	0.00
	0.95	0.59	0.41	0.00	0.15	0.85	0.00	0.15	0.85	0.00
1.00	0.10	1.00	0.00	0.00	1.00	0.00	0.00	0.74	0.00	0.26
	0.50	1.00	0.00	0.00	0.81	0.19	0.00	0.48	0.30	0.22
	0.80	1.00	0.00	0.00	0.70	0.26	0.04	0.33	0.56	0.11
	0.95	1.00	0.00	0.00	0.59	0.41	0.00	0.41	0.48	0.11
2.00	0.10	1.00	0.00	0.00	1.00	0.00	0.00	1.00	0.00	0.00
	0.50	1.00	0.00	0.00	1.00	0.00	0.00	0.81	0.19	0.00
	0.80	1.00	0.00	0.00	0.89	0.11	0.00	0.70	0.26	0.04
	0.95	1.00	0.00	0.00	0.81	0.19	0.00	0.59	0.41	0.00

6.8.2 Real world application

In this section we investigate the effects of different product allocations for a real world milkrun picking system. For this we study the warehouse of an online Chinese retailer in consumer electronics, the same warehouse considered in case 2 in Gong & De Koster (2008). However, the authors only compared the product unit waiting times (see Section 6.2). The retailer sells over 20,000 products in 226 cities and provides deliveries within 2 hours upon order receipt in large cities. In order to meet this service level agreement, orders should start processing within 5 minutes on average after being received and the order throughput times should be as short as possible.

Table 6.6: Parameters of the China online shopping warehouse.

(a) Warehouse

Parameter	Value
Warehouse area	985 m^2
Aisles	8
Number of storage locations per aisle side	30

(b) Order pickers

Parameter	Value
Number of order pickers	30
Number of storage locations per picker, N	16
Number of aisles per picker, A	4
Number of storage locations per rack per picker, L	2

(c) Operations

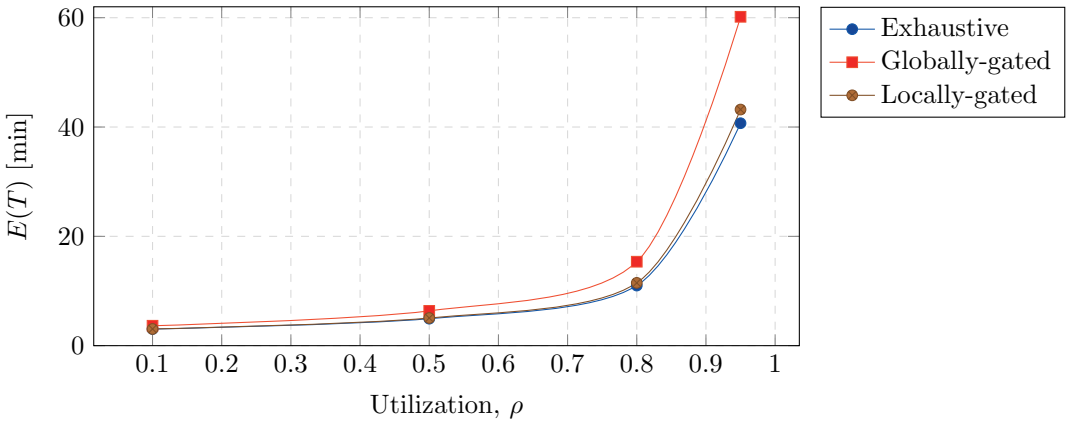
Parameter	Value
Travel speed of a picker	0.48 meter/sec.
Mean picking time, $E(B_i)$	1.51 sec.
Second moment picking time, $E(B_i^2)$	3.82
Mean traveling time (depot), s_0	63.0 sec.
Mean traveling time (side to side), s_1	2.00 sec.
Mean traveling time (adjacent storage locations), s_2	2.50 sec.
Mean traveling time (adjacent aisles), s_3	9.60 sec.

The company uses a milkrun picking system aided by an information system based on mobile technology and a call center (order processing center). In Table 6.6 an overview of the parameters of the warehouse is provided. The total area dedicated for the milkrun picking system is 985 m^2. The total number of aisles is 8 and each aisle has a width of 1 meter. On each side of the aisle there are 30 storage positions, where each storage position has a width and depth of 1.2 meter. Altogether there are 480 $(= 2 \cdot 8 \cdot 30)$ storage locations.

In total there are now 30 order pickers working per shift in the warehouse. Different from Gong & De Koster (2008) that assumes all order pickers visit sequentially every storage location and thus follow the same picking route, we assume that the order picking area is zoned and each picker is responsible for picking products from his or her zone. This means that there is no overlap in picking routes between order pickers. Picked products are brought to a central depot location where they are sorted per customer order. Additionally, we assume small sized orders (64% one product unit and 36% two product unit) and that every customer order can be fully picked in one zone. This allows us to study each zone in isolation.

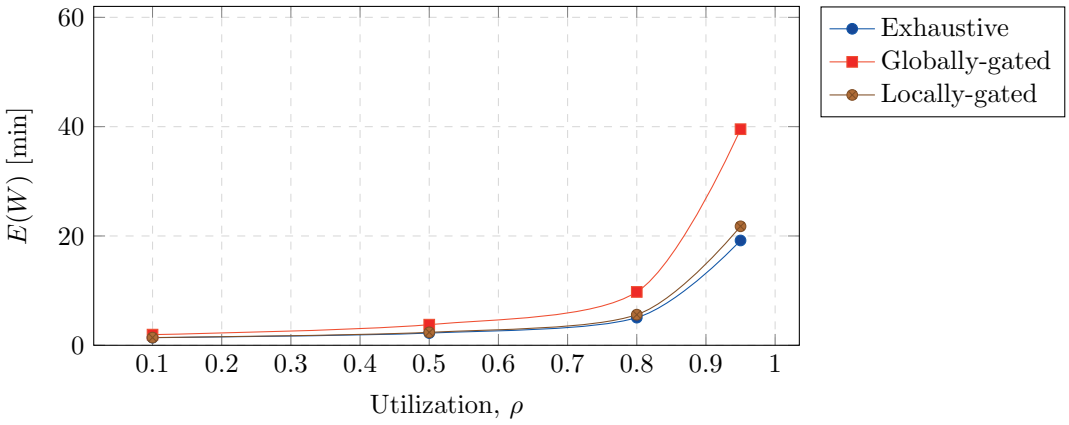
Then, a single order picker is responsible for $N = 16 = 2 \cdot 4 \cdot 2$ storage locations. The subsequent picking routes can be realized by adding additional cross-aisles to the order picking area. Each order picker has a traveling speed of 0.48 meter/seconds. The mean travel time side to side is $s_1 = 2$ seconds, the mean travel time within aisles between adjacent storage location is $s_2 = 2.50$ seconds, and the mean travel time between adjacent aisles is $s_3 = 9.60$ seconds. The average mean traveling times from the last storage location including the depot time is $s_0 = 63.0$ seconds for all the pickers. As a result, the total mean traveling time per cycle is $E(S) = 182.2$ seconds. All the second moments for the traveling times are $s_i^2 = 0$, $i = 0, 1, 2, 3$. Finally, for all storage locations the mean picking time per product unit is $E(B_i) = 1.51$ seconds and second moment of the picking time is $E(B_i^2) = 3.82$, $i = 1, \ldots, N$. In the rest of this section, we focus on one zone but the same conclusion can also be drawn for the other zones.

In Figure 6.11 the mean order throughput time and mean product unit waiting time is shown for the three picking strategies for different system utilization. The results were obtained after running the GA for which the parameters were identical as in Section 6.8.1. The run time of the algorithm was around 5 minutes per instance and



ρ	0.1	0.5	0.8	0.95
EX	3.0 min.	4.9 min.	11.0 min.	40.7 min.
GG	3.6 min.	6.4 min.	15.4 min.	60.2 min.
LG	3.1 min.	5.1 min.	11.5 min.	43.2 min.

(a) Mean order throughput time $E(T)$



ρ	0.1	0.5	0.8	0.95
EX	1.4 min.	2.2 min.	5.1 min.	19.2 min.
GG	2.0 min.	3.8 min.	9.8 min.	39.5 min.
LG	1.4 min.	2.4 min.	5.6 min.	21.8 min.

(b) Mean product unit waiting time $E(W)$

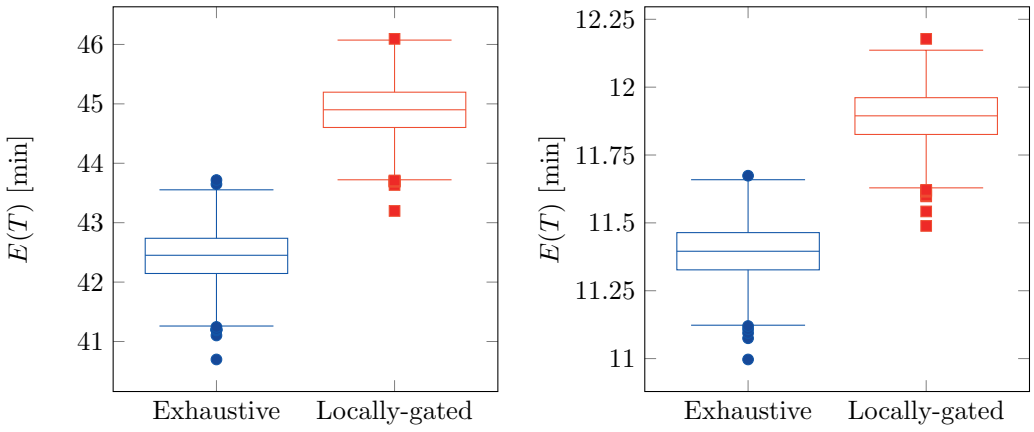
Figure 6.11: Results China online shopping warehouse for different utilization ρ and the three picking strategies.

around 500 generations were needed to find the best allocation. In Figure 6.11a the results for the mean order throughput time $E(T)$ is shown. The exhaustive strategy always achieves the lowest mean order throughput time, whereas the results of the locally-gated strategy are slightly above it. However, the globally-gated strategy performs significantly worse which shows that dynamically adding new customer orders to the picking cycle reduces the mean order throughput times considerably. From the results it can also be clearly seen that when the utilization increases, the higher the mean order throughput time becomes. For the average mean product unit waiting time $E(W) = \frac{1}{\Lambda}\sum_{i=1}^{N}\lambda_i E(W_i)$ in Figure 6.11b, similar conclusions can be drawn. On the other hand, comparing the results with the mean order throughput time it can be seen that the mean order throughput time is between 50% to 125% longer. This implies that when considering how long it takes to pick a customer order it is better to consider the order throughput time instead of product unit waiting times.

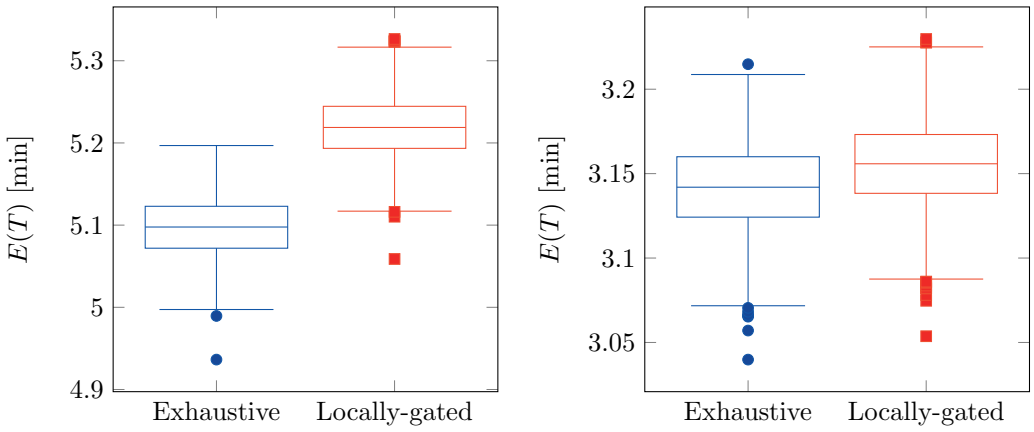
Figure 6.12 shows how much the mean order throughput time varies for several values of the utilization ρ for a randomly generated set of product allocations. We generated $3,000$ different allocations which also included the best allocation found in Figure 6.11 for which we calculated the mean order throughput time $E(T)$. We excluded globally-gated from this comparison since $E(B_i)$, $i = 1, \ldots, N$ is the same for every storage location, and therefore all product allocations have the same mean order throughput time. From the box plots it can be seen that the spread of mean order throughput times is around 3 minutes in case ρ is high to a couple of seconds when ρ is low and that the choice of picking strategy can lead to significantly shorter order throughput times.

6.9 Conclusion and further research

This chapter studied the order throughput time and product allocation in a milkrun picking system. In this system a picker picks orders that arrive in real time during the picking process, which subsequently changes dynamically the stops on the order picker's current picking route. This chapter is the first in studying order throughput times of multi-line orders in a milkrun picking system which provide better insights in the performance of the system and allows to study the effect of different product



(a) $\rho = 0.95$ **(b)** $\rho = 0.8$



(c) $\rho = 0.5$ **(d)** $\rho = 0.1$

Figure 6.12: Range of mean order throughput times for 3,000 different product allocations for various values of the utilization ρ.

allocations. It considerably extends the work of Gong & De Koster (2008) who only considered waiting times of single-line orders. We modeled the system as a cyclic polling system with general service times, general switch-over times, and simultaneous batch arrivals. For three picking strategies; exhaustive, locally-gated, and globally-gated, we determined the average order throughput time of a customer order by studying the evolution of the system through time from the order entering the system until all its order lines have been picked and delivered at the depot. Afterwards, we proposed an optimization framework for product allocation in a milkrun picking system in order to minimize the average order throughput time. Our results showed that the average order throughput time in a milkrun order system can significantly vary based on the chosen product allocation and picking strategy. In particular, we found that the exhaustive strategy obtains the lowest mean order throughput time when travel times between storage locations are long compared to the picking times, whereas both gated strategies perform better in the opposite situation.

The model and methods in this chapter lend themselves for further research. First, the model can be extended by including putaway and replenishment processes, similar as observed in a production setting. Other interesting topics are relaxing the assumption of an uncapacitated pick cart and investigating whether other or combinations of picking strategies can lead to increased picking performance. Also, it can be worthwhile to investigate whether a local backward routing policy, i.e. picking a product that arrived in the queue that just has been visited, might increase system performance. In addition, it is possible to further study a milkrun picking system with multiple pickers, where the order picking area is zoned and each picker is responsible for picking products from his/her zone. Interesting other research questions would be how many zones are required and how should products be allocated in order to minimize the order throughput time in case an order consists of demand for products located in multiple zones which all need to be send to a single depot location. Finally, the model can be generalized for the analysis of different warehouse systems such as carousels or paternosters, but also for production systems and communication networks.

7 Conclusions and future outlook

In this thesis, new stochastic models have been developed for the performance evaluation of several state-of-the-art warehousing systems that describe and predict the consequences of internal variability. Much of this variability is caused in the order picking process, e.g. variations in order arrivals, in picking times, and in availability of workers, pick totes, and other resources. The developed stochastic models provide valuable guidance in the rapid comparison of key features of different design alternatives and allow operations to be optimized in order to meet prespecified performance targets. In addition, they help designers to avoid major mistakes, because they expand the designer's intuitive understanding of what determines system performance.

We focused on two order picking systems that were mainly left unexplored in the literature, but that are highly relevant in practice namely; *zone picking* and *milkrun order picking*. We analyzed these systems from a stochastic point-of-view, mainly by using techniques from the field of queueing theory, e.g. closed product-form queueing networks and polling networks, and applied optimization techniques in order to compare different design alternatives.

7.1 Conclusions

In the first three chapters of this thesis we studied zone picking systems. Zone picking systems belong to the most popular conveyor-based picker-to-parts order picking methods used in practice. In such a system the order picking area is zoned, where in each zone an order picker is responsible for picking products from his or her dedicated part of the warehouse. However, from the industry there are many open questions about zone picking design issues that were not answered by the current

literature. In Chapter 2, we created an analytical model for zone picking systems with either single-segment, or multi-segment routing based on queueing theory. The model provides a valuable tool for rapid design of complex zone picking systems in order to meet specific performance levels and it can be used to study and reduce the sources of blocking and congestion. Because an exact analysis of the queueing model was not feasible, we approximated the blocking behavior with the jump-over protocol which yields product-form results. This result was proven using the concept of quasi-reversibility. The product-form results allowed us to efficiently evaluate the performance of the system using Mean Value Analysis and an aggregation technique. With these techniques we could rapidly obtain very accurate estimates of the key performance statistics such as zone utilization, average overall order throughput, mean system throughput time, and the probability of a tote blocked by a full zone. We evaluated the performance of the approximation on a large test set of different zone picking layouts, which varied e.g. on single- or multi-segment routing, number of zones, mean conveyor times, and the size of the input buffer of a zone. The results of the approximation were validated with a discrete-event simulation of the real queueing network. For both the single-segment and multi-segment routing lay-outs, almost all of the errors for the average system throughput, mean number of circulations, and the average mean throughput times of the zones fell between $0 - 1\%$, with only a few larger than 5%.

In Chapter 3, we extended the analysis by also studying the merge operation in zone picking systems. In case the system is under heavy load, congestion and blocking occur at conveyor merges due to limited free space on the conveyor. This congestion leads to reduced throughput and controls the maximum throughput capability of the system. A decomposition-based approximation was used to study each conveyor merge and zone in isolation. Our method progressively aggregated parts of the queueing network and replaced the aggregated subnetwork by a flow equivalent single node. The approximation directly solved the global balance equations of the underlying Markov chain of the subnetworks for its steady-state distribution. The results showed that the approximation is able to predict the maximum throughput capability of a zone picking system very accurately compared to simulation. Moreover, the model is capable of predicting the loss in throughput capacity given the level of congestion and blocking in the system. In addition, we used the model to study

the allocation of input and output buffer positions to zones in order to maximize the throughput capability of the system. When the system is not congested, it is more beneficial to have more input buffer positions, since it decreases the possibility that a tote is rejected from entering the buffer of the zone and has to recirculate on the conveyor. However, when congestion and blocking increases, it becomes more attractive to increase the number of output buffer positions, since the average time required to merge and the fact the order picker is stopped more often becomes higher than the time it takes for a tote to recirculate once. Also, when the system is heavily utilized, the supply of new totes to the zones stalls due to congestion at the merges.

Finally, in Chapter 4 we compared different product allocation methods and tested their influence on system performance for a real-world zone picking system. In order to assess the system performance, we extensively used the models of the previous chapters. We tested several product allocation methods and found that a product allocation that only applies workload balancing between zone/conveyor segments reduces the average system throughput on average 7% compared to a product allocation that minimizes the number of segments a tote on average has to visit. On the other hand, blocking of zones and segments is significantly reduced by a product allocation that applies workload balancing, e.g. segments are blocked 7.6% on average when a product allocation that minimizes the number of segments is used to 0.3% on average in the other case.

In the final two chapters we studied milkrun picking systems. In this system an order picker picks orders that arrive in real time during the picking process, which subsequently changes dynamically the stops on the order picker's current picking route. The advantage of milkrun picking is that it reduces order picking set-up time and worker travel time compared to conventional batch picking systems. In both chapters we focused extensively on the analysis of the order throughput time (or batch sojourn-time in a general context). This is the time lapse between the moment a customer order (consisting of multiple order lines) enters the system and the moment that the whole order is delivered at the depot. In these chapters, we modeled a milkrun picking system as a cyclic polling system with simultaneous batch arrivals. In addition, we investigated the effect of product allocation rules to study under which conditions a particular picking strategy, locally-gated, globally-gated, and exhaustive, minimizes the order throughput time. This is because the order

throughput time strongly depends on the product allocation, since a customer order often contains several order lines, each for a different product that can only be stored at certain locations within the order picking area.

The framework for analyzing the order throughput time was developed in Chapter 5. In this chapter we studied the batch sojourn-time for cyclic polling systems with simultaneous batch arrivals. We obtained exact expressions for the Laplace-Stieltjes transform of the steady-state batch sojourn-time distribution for the locally-gated, globally-gated, and exhaustive service disciplines. Also, we provided an alternative, more efficient way to determine the mean batch sojourn-time using Mean Value Analysis in case of exhaustive and locally-gated service disciplines. We compared the batch sojourn-times for the different service disciplines in several numerical examples and found that the best performing service discipline, minimizing the batch sojourn-time, depends on system characteristics. The results showed that when the switch-over times are longer compared to the service times, the exhaustive service discipline achieves the lowest mean batch sojourn-time, since it is more beneficial to serve all customers at the current queue first before moving to another queue. However, if the service times are longer than the switch-over times it is better to switch to another queue more often, because otherwise the server will spend too much time serving customers in one queue and it will take a long time before a customer batch is completely served. In this case, both gated policies perform better than exhaustive service.

Lastly, in Chapter 6 we study the order throughput time and product allocation in milkrun picking systems. By using and extending the framework of the previous chapter, we modeled the system as a cyclic polling system with simultaneous batch arrivals. For three picking strategies; exhaustive, locally-gated, and globally-gated, we determined the mean order throughput time of a customer order by studying the evolution of the system through time from the order entering the system until all its order lines have been picked and delivered at the depot. We also proposed an optimization framework for product allocation in a milkrun picking system in order to minimize the mean order throughput time. We compared the order throughput times for different service disciplines in several numerical examples and our results showed that the mean order throughput time in a milkrun order system can significantly vary based on the chosen product allocation and picking strategy. Similar as for the

mean batch sojourn-time, we find that the exhaustive strategy obtains the lowest mean order throughput time when travel times between storage locations are long compared to the pick times. Both gated strategies perform better in the opposite situation, where the pick times are longer compared to the travel times.

7.2 Future outlook

In this section, we discuss further research topics and potential extensions for the analytical models developed in this thesis. In addition, we briefly focus on recent warehousing trends which have an impact on the two order picking systems studied in this thesis.

First, zone picking systems will remain in the coming years one of the most popular order picking solutions, mainly because they can provide the level of flexibility often required by companies which many automated systems currently do not provide. This is especially the case for the picking process. Therefore, the analytical models studied in the first three chapters provide good starting points to evaluate and compare operational policies in zone picking systems, such as order batching, order splitting, order release, and more sophisticated product allocation methods, on system performance. Furthermore, the models may be extended in order to approximate higher moments or the distribution of performance statistics such as the on-time completion rate. In addition, when studying the system at an operational level the arrival rate of new customer orders may vary during the day. Sometimes it can be lower than the rate at which totes complete their service in the system. This situation can be analyzed using a semi-open queueing analysis approach (closed for totes and open for orders in the system), by aggregating the queueing network without the external queue by a flow equivalent server and then studying an aggregated Markov chain where its states represent the number of totes inside the network, as well as the number of waiting orders to be launched in the external queue. This approach resembles how the merge operation was studied in Chapter 4.

In practice it can be observed that order pickers cooperate when the workload in one zone is high, or they leave a zone when there is little work. One order picker then becomes responsible for picking products at multiple zones. It would be of great

interest to study at an operational level how many order pickers are required during daily operations, given the trade-off between maximizing performance and minimizing labor cost. Additionally, the performance of a zone picking system in combination with dynamic storage is still mainly left unexplored. With dynamic storage only a fraction of the products is stored in the picking area per zone and an automated storage and retrieval machine retrieves products, when they are requested, from the bulk storage area which is located behind the zone. This leads to significantly reduced walking distances for the order pickers. However, when the S/R machine is not properly controlled this would cause starvation of the pickers and ultimately reduce the performance of the system. For this, new algorithms need to be developed that decide on how and which product bin to reshuffle between the pick area and bulk storage to ensure the order picker never has to wait for an incoming product bin. A complicating factor would also be the fact that the S/R machine is often responsible for reshuffling bins for multiple zones.

On the whole, further research needs to be done to investigate the interactions of a zone picking system with other warehouse processes, e.g. receiving, putaway, replenishment, sorting, packing, and shipping. These interactions are extremely important since the performance of one process directly influences the next process in line and processing times can vastly vary during daily operations. In addition, we mainly focused on studying the order throughput (time), however warehouse managers are often also interested at the same time in measures such as the overall tardiness, picking cost, and balancing workload between pickers. For this, a comparative design study for zone picking systems can be carried out to find out which configurations are the most robust and flexible in terms of these measures. Moreover, performance increases are possible by dynamically balancing the segments and zones, instead of only trying to balance workload upon order release. Finally, there are many potential applications beyond zone picking systems where our methods to study the effects of blocking and to analyze order throughput times might also be applied successfully, e.g., end-of-aisle picking systems, sorting systems, AGV transportation systems, and vehicle-based compact storage systems.

The second system investigated in this thesis was milkrun picking. While the milkrun concept is still relatively new in order picking, it has already proved to be successful in manufacturing and production settings. Since picking aids like pick-by-voice

techniques or handheld terminals are becoming cheaper and reliable, milkrun picking systems will become more economically viable for companies.

For the general framework of Chapter 5 a further research topic would be to determine analytically for each of the three policies, under what conditions for the system parameters, its mean batch sojourn-time is smaller than that of the other two and whether alternative service disciplines can achieve even lower batch sojourn-times. Another interesting further research topic would be to study how to dynamically allocate an arriving customer batch over the various queues in order to minimize the batch sojourn-times. In other words, a central mechanism decides depending on the state of the system which queues the customers should join to ensure that the batch is served as soon as possible. In addition, it might be possible to find simple approximations for the batch sojourn-time based on low and heavy traffic analysis.

The model and methods for analyzing the order throughput time in milkrun picking system lend themselves for further research. The model can be extended by including putaway and replenishment processes, similar as observed in a production setting. Other interesting topics are relaxing the assumption of an uncapacitated pick cart and investigating whether other or combinations of picking strategies can lead to increased picking performance. Also, it can be worthwhile to investigate whether a local backward routing policy, i.e. picking a product that arrived in the queue that just has been visited, might increase system performance. In addition, it is possible to study a milkrun picking system with multiple pickers, where the order picking area is zoned and each picker is responsible for picking products from his/her zone. Interesting research questions would be how many zones are required, how should products be allocated, in order to minimize the order throughput time in case product units are located in multiple zones and should all be brought to a central depot location where they are sorted per customer order. Finally, the model may be generalized for analyzing different warehouse systems such as carousels or paternosters, but also for production systems and communication networks.

Bibliography

Apple, Jr., J. M., Meller, R. D., & White, Jr., J. A. (2010). Empirically-based warehouse design: Can academics accept such an approach? In *11th International Material Handling Research Colloquium*.

Baba, Y. (1986). On the $M^x/G/1$ queue with vacation time. *Operations Research Letters*, 5(2), 93–98.

Balsamo, S., De Nitto Personé, V., & Onvural, R. O. (2001). *Analysis of Queueing Networks with Blocking*, volume 31. Springer Netherlands.

Baskett, F., Chandy, K. M., Muntz, R. R., & Palacios, F. G. (1975). Open, closed, and mixed networks of queues with different classes of customers. *Journal of the ACM*, 22(2), 248–260.

Bastani, A. S. (1988). Analytical solution of closed-loop conveyor systems with discrete and deterministic material flow. *European Journal of Operational Research*, 35(2), 187–192.

Bastani, A. S. & Elsayed, E. A. (1986). Blocking in closed-loop conveyor systems connected in series with discrete and deterministic material flow. *Computers & Industrial Engineering*, 11(1), 40–45.

Baudin, M. (2004). *Lean Logistics: The Nuts and Bolts of Delivering Materials and Goods*. Productivity Press.

Baynat, B. & Dallery, Y. (1993). A unified view of product-form approximation techniques for general closed queueing networks. *Performance Evaluation*, 18(3), 205–224.

Bolch, G., Greiner, S., de Meer, H., & Trivedi, K. S. (2006). *Queueing Networks and Markov Chains: Modeling and Performance Evaluation with Computer Science Applications*. Wiley-Interscience, second edition.

Boon, M. A. A., Van der Mei, R. D., & Winands, E. M. M. (2011). Applications of polling systems. *Surveys in Operations Research and Management Science*, 16(2), 67–82.

Boon, M. A. A., Van der Mei, R. D., & Winands, E. M. M. (2012). Waiting times in queueing networks with a single shared server. *Queueing Systems*, 74(4), 403–429.

Boon, M. A. A., Van Wijk, A. C. C., Adan, I. J. B. F., & Boxma, O. J. (2010). A polling model with smart customers. *Queueing Systems*, 66(3), 239–274.

Bottani, E., Cecconi, M., Vignali, G., & Montanari, R. (2012). Optimisation of storage allocation in order picking operations through a genetic algorithm. *International Journal of Logistics Research and Applications*, 15(2), 127–146.

Boucherie, R. J. (1998). Norton's equivalent for queueing networks comprised of quasireversible components linked by state-dependent routing. *Performance Evaluation*, 32(2), 83–99.

Boucherie, R. J. & van Dijk, N. M. (1993). A generalization of Norton's theorem for queueing networks. *Queueing Systems*, 13(1-3), 251–289.

Boxma, O. J., Groenendijk, W. P., & Weststrate, J. A. (1990). A pseudoconservation law for service systems with a polling table. *IEEE Transactions on Communications*, 38(10), 1865–1870.

Boxma, O. J., Kella, O., & Kosinski, K. M. (2011). Queue lengths and workloads in polling systems. *Operations Research Letters*, 39(6), 401–405.

Boxma, O. J., Levy, H., & Yechiali, U. (1992). Cyclic reservation schemes for efficient operation of multiple-queue single-server systems. *Annals of Operations Research*, 35(3), 187–208.

Bozer, Y. A. & Ciemnoczolowski, D. D. (2013). Performance evaluation of small-batch container delivery systems used in lean manufacturing – Part 1: system stability and distribution of container starts. *International Journal of Production Research*, 51(2), 555–567.

Bozer, Y. A. & Hsieh, Y. J. (2005). Throughput performance analysis and machine layout for discrete-space closed-loop conveyors. *IIE Transactions*, 37(1), 77–89.

Brar, G. S. & Saini, G. (2011). Milk run logistics: literature review and directions. In *Proceedings of the World Congress on Engineering*, volume 1 (pp. 6–8).

Bright, L. & Taylor, P. G. (1995). Calculating the equilibrium distribution in level dependent quasi-birth-and-death processes. *Stochastic Models*, 11(3), 497–525.

Bryant, R. M., Krzesinski, A. E., Lakshmi, M. S., & Chandy, K. M. (1984). The MVA priority approximation. *ACM Transactions on Computer Systems*, 2(4), 335–359.

Buzen, J. P. (1973). Computational algorithms for closed queueing networks with exponential servers. *Communications of the ACM*, 16(9), 527–531.

Chandy, K. M., Herzog, U., & Woo, L. (1975). Parametric analysis of queuing networks. *IBM Journal of Research and Development*, 19(1), 36–42.

Chandy, K. M. & Sauer, C. H. (1980). Computational algorithms for product form queueing networks. *Communications of the ACM*, 23(10), 573–583.

Chao, X. & Miyazawa, M. (2000). Queueing networks with instantaneous movements: a unified approach by quasi-reversibility. *Advances in Applied Probability*, 32(1), 284–313.

Chiarawongse, J. & Srinivasan, M. M. (1991). On pseudo-conservation laws for the cyclic server system with compound Poisson arrivals. *Operations Research Letters*, 10(8), 453–459.

Choudhury, G. (2002). Analysis of the $M^X/G/1$ queueing system with vacation times. *Sankhya: The Indian Journal of Statistics*, 64(1), 37–49.

Choudhury, G. L. & Whitt, W. (1996). Computing distributions and moments in polling models by numerical transform inversion. *Performance Evaluation*, 25(4), 267–292.

Ciemnoczolowski, D. D. & Bozer, Y. A. (2013). Performance evaluation of small-batch container delivery systems used in lean manufacturing — Part 2: number of Kanban and workstation starvation. *International Journal of Production Research*, 51(2), 568–581.

Coffman, Jr, E. G., Gelenbe, E., & Gilbert, E. N. (1988). Analysis of a conveyor queue in a flexible manufacturing system. *European Journal of Operational Research*, 35(3), 382–392.

Cohen, J. W. (1982). *The single server queue*. North-Holland, Amsterdam.

De Koster, M. B. M. (1994). Performance approximation of pick-to-belt orderpicking systems. *European Journal of Operational Research*, 72(3), 558–573.

De Koster, M. B. M., Le-Duc, T., & Roodbergen, K. J. (2007). Design and control of warehouse order picking: A literature review. *European Journal of Operational Research*, 182(2), 481–501.

De Koster, M. B. M., Van der Poort, E. S., & Wolters, M. (1999). Efficient order-batching methods in warehouses. *International Journal of Production Research*, 37(7), 1479–1504.

Disney, R. L. (1962). Some multichannel queueing problems with ordered entry. *Journal of Industrial Engineering*, 13(1), 46–48.

Drury, J. (1988). *Towards more efficient order picking.* Technical Report 1, The Institute of Materials Management, Cranfield.

Economou, A. & Fakinos, D. (1998). Product form stationary distributions for queueing networks with blocking and rerouting. *Queueing Systems*, 30(3), 251–260.

Eisenberg, M. (1972). Queues with periodic service and changeover time. *Operations Research*, 20(2), 440–451.

Eisenstein, D. D. (2008). Analysis and optimal design of discrete order picking technologies along a line. *Naval Research Logistics*, 55(4), 350–362.

Emde, S. & Boysen, N. (2012). Optimally routing and scheduling tow trains for JIT-supply of mixed-model assembly lines. *European Journal of Operational Research*, 217(2), 287–299.

Federgruen, A. & Katalan, Z. (1999). The impact of adding a make-to-order item to a make-to-stock production system. *Management Science*, 45(7), 980–994.

Frazelle, E. H. (1990). *Stock location assignment and order batching productivity.* PhD thesis, Georgia Institute of Technology, Atlanta, GA.

Frazelle, E. H. (2001). *World-Class Warehousing and Material Handling.* McGraw Hill Professional.

Gademann, N. & Van de Velde, S. (2005). Order batching to minimize total travel time in a parallel-aisle warehouse. *IIE Transactions*, 37(1), 63–75.

Garfinkel, M. (2005). *Minimizing multi-zone orders in the correlated storage assignment problem*. PhD thesis, Georgia Institute of Technology, Atlanta, GA.

Gong, Y. & De Koster, M. B. M. (2008). A polling-based dynamic order picking system for online retailers. *IIE Transactions*, 40(11), 1070–1082.

Gong, Y. & De Koster, M. B. M. (2011). A review on stochastic models and analysis of warehouse operations. *Logistics Research*, 3(4), 191–205.

Gordon, W. J. & Newell, G. F. (1967). Closed queuing systems with exponential servers. *Operations Research*, 15(2), 254–265.

Gray, A. E., Karmarkar, U. S., & Seidmann, A. (1992). Design and operation of an order-consolidation warehouse: models and application. *European Journal of Operational Research*, 58(1), 14–36.

Gu, J., Goetschalckx, M., & McGinnis, L. F. (2010). Research on warehouse design and performance evaluation: A comprehensive review. *European Journal of Operational Research*, 203(3), 539–549.

Hanson, R. & Finnsgård, C. (2014). Impact of unit load size on in-plant materials supply efficiency. *International Journal of Production Economics*, 147, Part A, 46–52.

Henderson, W. & Taylor, P. G. (2001). State-dependent coupling of quasireversible nodes. *Queueing Systems*, 37(1), 163–197.

Holland, J. H. (1975). *Adaptation in natural and artificial systems*. University of Michigan Press.

Hsiao, M. T. & Lazar, A. A. (1989). An extension to Norton's equivalent. *Queueing Systems*, 5(4), 401–411.

Hsieh, Y. J. & Bozer, Y. A. (2005). Analytical modeling of closed-loop conveyors with load recirculation. In *Computational Science and Its Applications - ICCSA 2005*, volume 3483 of *Lecture Notes in Computer Science* (pp. 838–838). Springer Berlin.

Jackson, J. R. (1963). Jobshop-like queueing systems. *Management Science*, 10(1), 131–142.

Jane, C. C. (2000). Storage location assignment in a distribution center. *International Journal of Physical Distribution & Logistics Management*, 30(1), 55–71.

Jewkes, E., Lee, C., & Vickson, R. (2004). Product location, allocation and server home base location for an order picking line with multiple servers. *Computers & Operations Research*, 31(4), 623–636.

Keilson, J. & Servi, L. D. (1988). A distributional form of Little's Law. *Operations Research Letters*, 7(5), 223–227.

Kelly, F. P. (1979). *Reversibility and Stochastic Networks*. Wiley, first edition.

Kilic, H. S. & Durmusoglu, M. B. (2013). A mathematical model and a heuristic approach for periodic material delivery in lean production environment. *The International Journal of Advanced Manufacturing Technology*, 69(5-8), 977–992.

Kilic, H. S., Durmusoglu, M. B., & Baskak, M. (2012). Classification and modeling for in-plant milk-run distribution systems. *The International Journal of Advanced Manufacturing Technology*, 62(9-12), 1135–1146.

Kim, K. H. (1993). A joint determination of storage locations and space requirements for correlated items in a miniload automated storage-retrieval system. *International Journal of Production Research*, 31(11), 2649–2659.

Kleinrock, L. & Levy, H. (1988). The Analysis of Random Polling Systems. *Operations Research*, 36(5), 716–732.

Konheim, A. G., Levy, H., & Srinivasan, M. M. (1994). Descendant set: an efficient approach for the analysis of polling systems. *IEEE Transactions on Communications*, 42(234), 1245–1253.

Korytkowski, P. & Karkoszka, R. (2015). Simulation-based efficiency analysis of an in-plant milk-run operator under disturbances. *The International Journal of Advanced Manufacturing Technology*, In press, 1–11.

Kovács, A. (2011). Optimizing the storage assignment in a warehouse served by milkrun logistics. *International Journal of Production Economics*, 133(1), 312–318.

Kritzinger, P. S., Van Wyk, S., & Krzesinski, A. E. (1982). A generalisation of Norton's theorem for multiclass queueing networks. *Performance Evaluation*, 2(2), 98–107.

Kwo, T. T. (1958). A theory of conveyors. *Management Science*, 5(1), 51–71.

Latouche, G. & Ramaswami, V. (1999). *Introduction to matrix analytic methods in stochastic modeling*, volume 5. SIAM.

Le-Duc, T. & De Koster, M. B. M. (2007). Travel time estimation and order batching in a 2-block warehouse. *European Journal of Operational Research*, 176(1), 374–388.

Levy, H. & Sidi, M. (1990). Polling systems: Applications, modeling, and optimization. *IEEE Transactions on Communications*, 38(10), 1750–1760.

Levy, H. & Sidi, M. (1991). Polling systems with simultaneous arrivals. *IEEE Transactions on Communications*, 39(6), 823–827.

Little, J. D. C. (1961). A proof of the queuing formula $L = \lambda W$. *Operations Research*, 9(3), 383–387.

Malmborg, C. J. (1996). Storage assignment policy tradeoffs. *International Journal of Production Research*, 34(2), 363–378.

Marchet, G., Melacini, M., & Perotti, S. (2015). Investigating order picking system adoption: a case-study-based approach. *International Journal of Logistics Research and Applications*, 18(1), 82–98.

Marie, R. A. (1979). An approximate analytical method for general queueing networks. *IEEE Transactions on Software Engineering*, 5(5), 530–538.

Melacini, M., Perotti, S., & Tumino, A. (2010). Development of a framework for pick-and-pass order picking system design. *The International Journal of Advanced Manufacturing Technology*, 53(9), 841–854.

Mitchell, M. (1998). *An introduction to genetic algorithms*. USA: MIT Press, reprint, illustrated edition.

Muth, E. J. (1977). A model of a closed-loop conveyor with random material flow. *AIIE Transactions*, 9(4), 345–351.

Muth, E. J. & White, J. A. (1979). Conveyor theory: a survey. *AIIE Transactions*, 11(4), 270–277.

Nelson, R. (1995). *Probability, Stochastic Processes, and Queueing theory: The Mathematics of Computer Performance Modelling*. Springer.

Neuse, D. & Chandy, K. M. (1982). HAM: The heuristic aggregation method for solving general closed queueing network models of computer systems. In *ACM SIGMETRICS Performance Evaluation Review*, volume 11 (pp. 195–212).

Osorio, C. & Bierlaire, M. (2009). An analytic finite capacity queueing network model capturing the propagation of congestion and blocking. *European Journal of Operational Research*, 196(3), 996–1007.

Pan, J. C.-H. & Wu, M.-H. (2009). A study of storage assignment problem for an order picking line in a pick-and-pass warehousing system. *Computers & Industrial Engineering*, 57(1), 261–268.

Papadopoulos, H. T., Heavey, C., & Browne, J. (1993). *Queueing Theory in Manufacturing Systems Analysis and Design*. Springer.

Park, B. C. (2012). *Warehousing in the Global Supply Chain: Advanced Models, Tools and Applications for Storage Systems*. Springer-Verlag, first edition.

Perros, H. G. (1994). *Queueing Networks with Blocking*. Oxford University Press, Inc.

Petersen, C. G. (2000). An evaluation of order picking policies for mail order companies. *Production and Operations Management*, 9(4), 319–335.

Petersen, C. G. (2002). Considerations in order picking zone configuration. *International Journal of Operations & Production Management*, 22(7), 793–805.

Pittel, B. (1979). Closed exponential networks of queues with saturation: The Jackson-type stationary distribution and its asymptotic analysis. *Mathematics of Operations Research*, 4(4), 357–378.

Reeves, C. (2003). *Handbook of Metaheuristics*, chapter 3, (pp. 55 – 82). Springer Science & Business Media.

Reiser, M. (1981). Mean-value analysis and convolution method for queue-dependent servers in closed queueing networks. *Performance Evaluation*, 1(1), 7–18.

Reiser, M. & Lavenberg, S. S. (1980). Mean-value analysis of closed multichain queuing networks. *Journal of the ACM*, 27(2), 313–322.

Resing, J. A. C. (1993). Polling systems and multitype branching processes. *Queueing Systems*, 13(4), 409–426.

Roodbergen, K. J. & De Koster, M. B. M. (2001). Routing methods for warehouses with multiple cross aisles. *International Journal of Production Research*, 39(9), 1865–1883.

Rouwenhorst, B., Reuter, B., Stockrahm, V., Van Houtum, G. J., Mantel, R. J., & Zijm, W. H. M. (2000). Warehouse design and control: Framework and literature review. *European Journal of Operational Research*, 122(3), 515–533.

Schassberger, R. (1984). Decomposable stochastic networks: Some observations. *Modelling and Performance Evaluation Methodology*, 60, 135–150.

Schmidt, L. C. & Jackman, J. (2000). Modeling recirculating conveyors with blocking. *European Journal of Operational Research*, 124(2), 422–436.

Sherali, H. D. & Smith, J. C. (2001). Improving discrete model representations via symmetry considerations. *Management Science*, 47(10), 1396–1407.

Shiozawa, Y., Takine, T., Takahashi, Y., & Hasegawa, T. (1990). Analysis of a polling system with correlated input. *Computer Networks and ISDN Systems*, 20(1-5), 297–308.

Sonderman, D. (1982). An analytical model for recirculating conveyors with stochastic inputs and outputs. *International Journal Of Production Research*, 20(5), 591–605.

Sonderman, D. & Pourbabai, B. (1987). Single server stochastic recirculation systems. *Computers & Operations Research*, 14(1), 75–84.

Staab, T., Klenk, E., Galka, S., & Günthner, W. A. (2015). Efficiency in in-plant milk-run systems – The influence of routing strategies on system utilization and process stability. *Journal of Simulation*, 6. In press.

Stidham, Jr., S. (2002). Analysis, design, and control of queueing systems. *Operations Research*, 50(1), 197–216.

Takagi, H. (1986). *Analysis of polling systems*. MIT press.

Takagi, H. (2000). Analysis and application of polling models. In *Performance Evaluation: Origins and Directions* (pp. 423–442). Springer.

Tompkins, J. A., White, J. A., Bozer, Y. A., Frazelle, E. H., & Tanchoco, J. M. A. (2003). *Facilities Planning*. Hoboken, NJ: Wiley.

Tsai, C. Y., Liou, J. J. H., & Huang, T. M. (2008). Using a multiple-GA method to solve the batch picking problem: considering travel distance and order due time. *International Journal of Production Research*, 46(22), 6533–6555.

Van den Berg, J. P. (1999). A literature survey on planning and control of warehousing systems. *IIE Transactions*, 31(8), 751–762.

Van der Gaast, J. P., Adan, I. J. B. F., & De Koster, M. B. M. (2015). The analysis of batch sojourn-times in polling systems. Working paper.

Van der Gaast, J. P., De Koster, M. B. M., Adan, I. J. B. F., & Resing, J. A. C. (2012). Modeling and performance analysis of sequential zone picking systems. Working Paper.

Van der Mei, R. D. (2001). Polling systems with simultaneous batch arrivals. *Stochastic Models*, 17(3), 271–292.

Van der Mei, R. D. (2002). Waiting-time distributions in polling systems with simultaneous batch arrivals. *Annals of Operations Research*, 113(1-4), 155–173.

Van der Mei, R. D., Hariharan, R., & Reeser, P. K. (2001). Web server performance modeling. *Telecommunication Systems*, 16(3-4), 361–378.

Van der Vorst, J. G. A. J. & Beulens, A. J. M. (2002). Identifying sources of uncertainty to generate supply chain redesign strategies. *International Journal of Physical Distribution & Logistics Management*, 32(6), 409–430.

Van Dijk, N. M. (1988). On Jackson's product form with "jump-over" blocking. *Operations Research Letters*, 7(5), 233–235.

Vanderlande (2007). Zone picking met VISION.ZPS.

Vishnevskii, V. M. & Semenova, O. V. (2006). Mathematical methods to study the polling systems. *Automation and Remote Control*, 67(2), 173–220.

Walrand, J. (1983). A note on Norton's theorem for queuing networks. *Journal of Applied Probability*, 20(2), 442–444.

Whitt, W. (1982). Approximating a point process by a renewal process, I: Two basic methods. *Operations Research*, 30(1), 125–147.

Winands, E. M. M., Adan, I. J. B. F., & Van Houtum, G. J. (2006). Mean value analysis for polling systems. *Queueing Systems*, 54(1), 35–44.

Winands, E. M. M., Adan, I. J. B. F., & Van Houtum, G. J. (2011). The stochastic economic lot scheduling problem: A survey. *European Journal of Operational Research*, 210(1), 1–9.

Wolff, R. W. (1982). Poisson Arrivals See Time Averages. *Operations Research*, 30(2), 223–231.

Wolff, R. W. (1989). *Stochastic Modeling and the Theory of Queues*, volume 14. London: Prentice Hall, first edition.

Xiao, J. & Zheng, L. (2011). Correlated storage assignment to minimize zone visits for BOM picking. *The International Journal of Advanced Manufacturing Technology*, 61(5-8), 797–807.

Yao, D. D. & Buzacott, J. A. (1987). Modeling a class of flexible manufacturing systems with reversible routing. *Operations Research*, 35(1), 87–93.

Yu, M. & De Koster, M. B. M. (2008). Performance approximation and design of pick-and-pass order picking systems. *IIE Transactions*, 40(11), 1054–1069.

Yu, M. & De Koster, M. B. M. (2009). The impact of order batching and picking area zoning on order picking system performance. *European Journal of Operational Research*, 198(2), 480–490.

Yu, M. & De Koster, M. B. M. (2010). Enhancing performance in order picking processes by dynamic storage systems. *International Journal of Production Research*, 48(16), 4785–4806.

Zijm, W. H. M., Adan, I. J. B. F., Buitenhek, R., & Van Houtum, G. J. (2000). Capacity analysis of an automated kit transportation system. *Annals of Operations Research*, 93(1-4), 423–446.

About the author

Jelmer Pier van der Gaast was born in Rotterdam (The Netherlands) on July 21, 1988. He completed grammar school at the RSG Hoeksche Waard, Oud-Beijerland in June 2006. In October 2010 he received his master's degree in Econometrics and Management Science with distinction from Erasmus University Rotterdam. In 2011 he joined the Management of Technology & Innovation group at the Erasmus University Rotterdam.

His research interests include: warehouse operations, stochastic processes, and queueing theory. His research findings have been presented in many international conferences including INFORMS Annual Meeting, European Conference on Operational Research, SMMSO Conference on Stochastic Models of Manufacturing and Service Operations, and International Material Handling Research Colloquium.

Portfolio

Publications

Publications in journals:

Van der Gaast, J. P., Rietveld, C. A., Gabor, A. F., & Zhang, Y. (2014). A Tabu Search Algorithm for application placement in computer clustering. *Computers & Operations Research*, 50, 38-46.

Conference papers:

Van der Gaast, J. P., De Koster, M. B. M., Adan, I. J. B. F., & Resing, J. A. C. (2011). Application of closed queueing network in the performance of a pick-and-pass order picking system. In *Proceedings of the 8th International Conference on Stochastic Models of Manufacturing and Service Operations*

Book chapters:

Van der Gaast, J. P., De Koster, M. B. M., & Adan, I. J. B. F. (2014). Modeling Conveyor Merges In Zone Picking Systems. In *Progress in Material Handling Research: 2014*.

Tabuns, A., Van der Gaast, J. P., & De Koster, M. B. M. (2016). Robust vehicle dispatching rules for automated container terminals. In *Ports and Networks*.

Van der Gaast, J. P., De Koster, M. B. M, & Adan, I. J. B. F. (2016). Analyzing order throughput times in a milkrun picking system. In *Progress in Material Handling Research: 2016*, In press.

Working papers:

Van der Gaast, J. P., De Koster, M. B. M., Adan, I. J. B. F., & Resing, J. A. C. (2012). Modeling and performance analysis of sequential zone picking systems. EURANDOM Preprint series, 2012-026.

Van der Gaast, J. P., Adan, I. J. B. F., & De Koster, M. B. M. (2015). The analysis of batch sojourn-times in polling systems. arXiv:1607.03345, math.PR.

PhD Courses

Markov Decision Processes	Non-cooperative Games
Stochastic Programming	English (CPE) certificate
Queueing Theory	Research Methodology and Measurements
Advanced Queueing Theory	Publishing Strategy
Stochastic Dynamic Optimization	Statistical Methods
Combinatorial Optimization 2b	Interaction Performance Training/Coaching

Teaching

Lecturer:

Managing the supply chain 2015-2016

Operations & supply chain management 2015-2016

Supply chain simulation 2015-2016

Tutorial lecturer:

Primaire Processen 2011-2015

Operations Management 2011-2015

Guest lecturer:

Facility logistics management 2011-2016

Stochastic Models and Optimisation 2012-2013

Conferences attended

SMMSO 2011, Kusadasi, Turkey

LNMB Conference 2012, Lunteren, The Netherlands

INFORMS annual meeting 2012, Phoenix, USA

LNMB Conference 2013, Lunteren, The Netherlands

EURO 2013, Rome, Italy

OR 2013, Rotterdam, The Netherlands

LNMB Conference 2014, Lunteren, The Netherlands

LOGMS 2014, Rotterdam The Netherlands

IMHRC 2014, Cincinnati, USA

INFORMS annual meeting 2014, San Francisco, USA

IMHRC 2016, Karlsruhe, Germany

Summary

Order picking, the process of retrieving customer orders from their storage locations, is the most critical operation in a warehouse. In order for a warehouse to operate efficiently, the order picking process needs to be robustly designed and optimally controlled. Any under performance in order picking can lead to unsatisfactory service and high operational cost for the warehouse, and consequently for the whole supply chain. Therefore, many warehouses invest heavily in state-of-the-art order picking solutions to increase productivity and reduce any uncertainty associated with order picking, e.g. in order arrivals, and in picking times. However, this has increased the complexity of today's warehouse operations.

In this thesis, we develop new stochastic models for the performance evaluation of two state-of-the-art warehousing systems; *zone picking* and *milkrun order picking* that accurately describe and predict the consequences of variability, in order to create optimal design and control methods to improve the performance of these systems.

In Chapter 2 we develop an analytical model of zone picking systems, one of the most popular conveyor-based picker-to-parts order picking methods used in practice. We model the various elements of the system, like conveyor segments and the pick zones, as a network of queues with multiple order classes and with the dynamic block-and-recirculate protocol. An innovative approximation method is proposed to accurately assess key performance statistics and we show that for a wide range of parameters the mean relative error compared to simulation in the system throughput is typically less than 1%.

The previous model is extended in Chapter 3 in order to study the impact of conveyor merges in highly utilized zone picking systems which strongly influence the maximum performance of the overall system. Our approximation model, using an aggregation technique and matrix-geometric methods, produces very accurate estimates of the

maximum throughput capability of a zone picking system and shows that throughput drops dramatically when congestion and blocking at the merges increases.

In Chapter 4 the approximation models of the previous chapters are used to investigate the performance of a current zone picking system of a large wholesaler supplying non-food items to supermarkets. We test the impact of various storage allocation methods on system performance and find that a storage allocation that minimizes the average number of segments a tote has to visit achieves the highest system throughput.

In Chapter 5 a general framework is presented that can be used to analyze the order throughput time in a milkrun picking system. We consider a cyclic polling system with general service times, general switch-over times, and simultaneous batch arrivals. For different service disciplines, Exhaustive, Locally-gated, and Globally-gated, we obtain exact expressions for the Laplace-Stieltjes transform of the steady-state batch sojourn-time distribution, which can be used to determine the moments of the batch sojourn-time, and in particular, its mean. We also provide an alternative, more efficient way to determine the mean batch sojourn-time, using Mean Value Analysis. Our results show that the best performing service discipline, in terms of minimizing the batch sojourn-time, depends on system characteristics.

Finally, Chapter 6 studies milkrun picking systems. In such systems orders arrive in real time during the picking operation and change the stops in the current picking cycle dynamically. By using and extending the framework of the previous chapter, we analyze the order throughput times for various picking strategies and test how these times can be improved by choosing a better product allocation policy. For this we develop an optimization framework for product allocation to minimize the average order throughput time in the system. Our results show that the average order throughput time in a milkrun order system can significantly vary based on the chosen product allocation and picking strategy.

Samenvatting (Summary in Dutch)

Orderverzamelen, het proces van het verzamelen van klantorders van hun opslaglocaties, is de meest cruciale operatie in een magazijn. Om een magazijn efficiënt te exploiteren, moet het orderverzamelproces robuust zijn ingericht en optimaal worden aangestuurd. Elke onderprestatie in het orderverzamelen kan leiden tot onbevredigende service en hoge operationele kosten. Veel magazijnen hebben daarom geïnvesteerd in de nieuwste magazijnoplossingen om productiviteit te verhogen en stochastiek in het orderverzamelen te verminderen, bijvoorbeeld in order aankomsten en in verzameltijden. Dit heeft echter geleid tot meer complexiteit in de hedendaagse magazijnprocessen.

In dit proefschrift ontwikkelen wij nieuwe stochastische modellen voor de prestatie-evaluatie van twee innovatieve magazijnsystemen; zone picking en milkrun picking. Deze modellen beschrijven en voorspellen accuraat de consequenties van stochastiek in het orderverzamelen. Tevens kunnen ze worden gebruikt voor het beantwoorden van ontwerpvraagstukken en het ontwikkelen van controlemechanismes om systeemprestatie te verbeteren.

In hoofdstuk 2 ontwikkelen we een analytisch model om zone picking systemen te analyseren, één van de meest populaire conveyor-based picker-to-parts orderverzamelsystemen in de praktijk. We modelleren de verschillende elementen van het systeem, zoals de transportbaan en de zones, als een netwerk van wachtrijen met meerdere orderklassen en met het dynamische block-and-recirculate protocol. Een innovatieve approximatie techniek is ontworpen om accuraat de belangrijkste prestatie-statistieken te bepalen. We laten zien dat voor een groot aantal parameters de gemiddelde relatieve fout in systeem doorzet vergeleken met simulatie kleiner is dan 1%.

Het vorige model is uitgebreid in hoofdstuk 3 om de impact van het samenvoegen van meerdere transportbanen te bestuderen. In een zwaar belast zone picking

systeem wordt de maximale prestatie van het systeem sterk beïnvloed door de samenvoegpunten. Ons approximatie model maakt gebruik van een aggregatie techniek en matrix-geometrische methoden en geeft nauwkeurige schattingen voor de maximale doorzet. Ook laten de resultaten zien dat de doorzet significant afneemt als de congestie en blokkering toeneemt op samenvoegpunten.

In hoofdstuk 4 worden de approximatie modellen van de voorgaande hoofdstukken gebruikt om de prestatie van een huidig zone picking systeem van een grote distributeur van non-food producten aan supermarkten te bestuderen. We testen het effect op systeem prestatie van verschillende opslag allocatie methoden en tonen aan dat een allocatie die het aantal segment bezoeken voor een order bak minimaliseert de hoogste doorzet oplevert.

In hoofdstuk 5 presenteren wij een generiek framework om de orderdoorlooptijd in een milkrun picking systeem te analyseren. We bestuderen een cyclisch polling systeem met algemene bedieningstijden, algemene switch-over tijden, en simultane batch aankomsten. Voor drie verschillende service disciplines, Exhaustive, Locally-gated, en Globally-gated, bepalen wij exacte formules voor de Laplace-Stieltjes transformatie van de steady-state batch sojourn-time distributie. Deze kunnen gebruikt worden om de momenten van de batch-sojourn time te bepalen, waarvan de belangrijkste het gemiddelde is. Tevens formuleren wij een alternatieve en meer efficiënte manier om de gemiddelde batch sojourn-time te bepalen door middel van Mean Value Analysis. Onze resultaten laten zien dat de best presterende service discipline, in termen van het minimaliseren van de batch sojourn-time, afhangt van systeem karakteristieken.

Ten slotte, in hoofdstuk 6 staan milkrun picking systemen centraal. In een milkrun picking systeem komen in real-time klantorders binnen tijdens het orderverzamelen en kunnen in dezelfde pick cycle worden gepakt door dynamisch de stoppunten in de huidige verzamelroute aan te passen. Door gebruik te maken van de technieken van het vorige hoofdstuk, analyseren wij de orderdoorlooptijd voor drie verschillende verzamelstrategieën. Tevens onderzoeken we of een betere product allocatie deze tijd kan verminderen. Hiervoor ontwikkelen we een optimalisatie framework voor product allocatie om zodoende de gemiddelde orderdoorlooptijd in het systeem te minimaliseren. Onze resultaten laten zien dat de gemiddelde orderdoorlooptijd in een milkrun picking systeem significant kan verschillen gegeven de gekozen product allocatie en verzamelstrategie.

ERIM Ph.D. Series Research in Management

The ERIM PhD Series contains PhD dissertations in the field of Research in Management defended at Erasmus University Rotterdam and supervised by senior researchers affiliated to the Erasmus Research Institute of Management (ERIM). All dissertations in the ERIM PhD Series are available in full text through the ERIM Electronic Series Portal: `http://repub.eur.nl/pub`. ERIM is the joint research institute of the Rotterdam School of Management (RSM) and the Erasmus School of Economics at the Erasmus University Rotterdam (EUR).

Dissertations last five years

Abbink, E.J., *Crew Management in Passenger Rail Transport*, Promotor(s): Prof.dr. L.G. Kroon & Prof.dr. A.P.M. Wagelmans, EPS-2014-325-LIS, `http://repub.eur.nl/pub/76927`

Acar, O.A., *Crowdsourcing for Innovation: Unpacking Motivational, Knowledge and Relational Mechanisms of Innovative Behavior in Crowdsourcing Platforms*, Promotor(s): Prof.dr.ir. J.C.M. van den Ende, EPS-2014-321-LIS, `http://repub.eur.nl/pub/76076`

Akin Ates, M., *Purchasing and Supply Management at the Purchase Category Level: strategy, structure and performance*, Promotor(s): Prof.dr. J.Y.F. Wynstra & Dr. E.M. van Raaij, EPS-2014-300-LIS, `http://repub.eur.nl/pub/50283`

Akpinar, E., *Consumer Information Sharing*, Promotor(s): Prof.dr.ir. A. Smidts, EPS- 2013-297-MKT, `http://repub.eur.nl/pub/50140`

Alexander, L., *People, Politics, and Innovation: A Process Perspective*, Promotor(s): Prof.dr. H.G. Barkema & Prof.dr. D.L. van Knippenberg, EPS-2014-331-S&E, http://repub.eur.nl/pub/77209

Almeida e Santos Nogueira, R.J. de, *Conditional Density Models Integrating Fuzzy and Probabilistic Representations of Uncertainty*, Promotor(s): Prof.dr.ir. U. Kaymak & Prof.dr. J.M.C. Sousa, EPS-2014-310-LIS, http://repub.eur.nl/pub/51560

Bannouh, K., *Measuring and Forecasting Financial Market Volatility using High-frequency Data*, Promotor(s): Prof.dr. D.J.C. van Dijk, EPS-2013-273-F&A, http://repub.eur.nl/pub/38240

Ben-Menahem, S.M., *Strategic Timing and Proactiveness of Organizations*, Promotor(s): Prof.dr. H.W. Volberda & Prof.dr.ing. F.A.J. van den Bosch, EPS-2013-278-S&E, http://repub.eur.nl/pub/39128

Benning, T.M., *A Consumer Perspective on Flexibility in Health Care: Priority Access Pricing and Customized Care*, Promotor(s): Prof.dr.ir. B.G.C. Dellaert, EPS-2011-241-MKT, http://repub.eur.nl/pub/23670

Benschop, N, *Biases in Project Escalation: Names, frames & construal levels*, Promotors: Prof.dr. K.I.M. Rhode, Prof.dr. H.R. Commandeur, Prof.dr. M.Keil & Dr. A.L.P. Nuijten, EPS-2015-375-S&E, http://repub.eur.nl/pub/79408

Berg, W.E. van den, *Understanding Salesforce Behavior using Genetic Association Studies*, Promotor(s): Prof.dr. W.J.M.I. Verbeke, EPS-2014-311-MKT, http://repub.eur.nl/pub/51440

Betancourt, N.E., *Typical Atypicality: Formal and Informal Institutional Conformity, Deviance, and Dynamics*, Promotor(s): Prof.dr. B. Krug, EPS-2012-262-ORG, http://repub.eur.nl/pub/32345

Bliek, R. de, *Empirical Studies on the Economic Impact of Trust*, Promotor(s): Prof.dr. J. Veenman & Prof.dr. Ph.H.B.F. Franses, EPS-2015-324-ORG, http://repub.eur.nl/pub/78159

Blitz, D.C., *Benchmarking Benchmarks*, Promotor(s): Prof.dr. A.G.Z. Kemna & Prof.dr. W.F.C. Verschoor, EPS-2011-225-F&A, http://repub.eur.nl/pub/22624

Boons, M., *Working Together Alone in the Online Crowd: The Effects of Social Motivations and Individual Knowledge Backgrounds on the Participation and Perfor-*

mance of Members of Online Crowdsourcing Platforms, Promotor(s): Prof.dr. H.G. Barkema & Dr. D.A. Stam, EPS-2014-306-S&E, http://repub.eur.nl/pub/50711

Brazys, J., *Aggregated Marcoeconomic News and Price Discovery*, Promotor(s): Prof.dr. W.F.C. Verschoor, EPS-2015-351-F&A, http://repub.eur.nl/pub/78243

Burger, M.J., *Structure and Cooptition in Urban Networks*, Promotor(s): Prof.dr. G.A. van der Knaap & Prof.dr. H.R. Commandeur, EPS-2011-243-ORG, http://repub.eur.nl/pub/26178

Byington, E., *Exploring Coworker Relationships: Antecedents and Dimensions of Interpersonal Fit, Coworker Satisfaction, and Relational Models*, Promotor(s): Prof.dr. D.L. van Knippenberg, EPS-2013-292-ORG, http://repub.eur.nl/pub/41508

Camacho, N.M., *Health and Marketing: Essays on Physician and Patient Decision-Making*, Promotor(s): Prof.dr. S. Stremersch, EPS-2011-237-MKT, http://repub.eur.nl/pub/23604

Cancurtaran, P., *Essays on Accelerated Product Development*, Promotor(s): Prof.dr. F. Langerak & Prof.dr.ir. G.H. van Bruggen, EPS-2014-317-MKT, http://repub.eur.nl/pub/76074

Caron, E.A.M., *Explanation of Exceptional Values in Multi-dimensional Business Databases*, Promotor(s): Prof.dr.ir. H.A.M. Daniels & Prof.dr. G.W.J. Hendrikse, EPS-2013-296-LIS, http://repub.eur.nl/pub/50005

Carvalho, L. de, *Knowledge Locations in Cities: Emergence and Development Dynamics*, Promotor(s): Prof.dr. L. Berg, EPS-2013-274-S&E, http://repub.eur.nl/pub/38449

Consiglio, I., *Others: Essays on Interpersonal and Consumer Behavior*, Promotor: Prof.dr. S.M.J. van Osselaer, EPS-2016-366-MKT, http://repub.eur.nl/pub/79820

Cox, R.H.G.M., *To Own, To Finance, and To Insure - Residential Real Estate Revealed*, Promotor(s): Prof.dr. D. Brounen, EPS-2013-290-F&A, http://repub.eur.nl/pub/40964

Deichmann, D., *Idea Management: Perspectives from Leadership, Learning, and Network Theory*, Promotor(s): Prof.dr.ir. J.C.M. van den Ende, EPS-2012-255-ORG, http://repub.eur.nl/pub/31174

Deng, W., *Social Capital and Diversification of Cooperatives*, Promotor(s): Prof.dr. G.W.J. Hendrikse, EPS-2015-341-ORG, http://repub.eur.nl/pub/77449

Desmet, P.T.M., *In Money we Trust? Trust Repair and the Psychology of Financial Compensations*, Promotor(s): Prof.dr. D. de Cremer, EPS-2011-232-ORG, http://repub.eur.nl/pub/23268

Dollevoet, T.A.B., *Delay Management and Dispatching in Railways*, Promotor(s): Prof.dr. A.P.M. Wagelmans, EPS-2013-272-LIS, http://repub.eur.nl/pub/38241

Doorn, S. van, *Managing Entrepreneurial Orientation*, Promotor(s): Prof.dr. J.J.P. Jansen, Prof.dr.ing. F.A.J. van den Bosch, & Prof.dr. H.W. Volberda, EPS-2012-258-STR, http://repub.eur.nl/pub/32166

Douwens-Zonneveld, M.G., *Animal Spirits and Extreme Confidence: No Guts, No Glory?* Promotor(s): Prof.dr. W.F.C. Verschoor, EPS-2012-257-F&A, http://repub.eur.nl/pub/31914

Duca, E., *The Impact of Investor Demand on Security Offerings*, Promotor(s): Prof.dr. A. de Jong, EPS-2011-240-F&A, http://repub.eur.nl/pub/26041

Duyvesteyn, J.G. *Empirical Studies on Sovereign Fixed Income Markets*, Promotor(s): Prof.dr P.Verwijmeren & Prof.dr. M.P.E. Martens, EPS-2015-361-F&A, http://repub.eur.nl/pub/79033

Duursema, H., *Strategic Leadership: Moving Beyond the Leader-Follower Dyad*, Promotor(s): Prof.dr. R.J.M. van Tulder, EPS-2013-279-ORG, http://repub.eur.nl/pub/39129

Eck, N.J. van, *Methodological Advances in Bibliometric Mapping of Science*, Promotor(s): Prof.dr.ir. R. Dekker, EPS-2011-247-LIS, http://repub.eur.nl/pub/26509

Elemes, A, *Studies on Determinants and Consequences of Financial Reporting Quality*, Promotor: Prof.dr. E.Peek, EPS-2015-354-F&A, http://repub.eur.nl/pub/79037

Ellen, S. ter, *Measurement, Dynamics, and Implications of Heterogeneous Beliefs in Financial Markets*, Promotor(s): Prof.dr. W.F.C. Verschoor, EPS-2015-343-F&A, http://repub.eur.nl/pub/78191

Erlemann, C., *Gender and Leadership Aspiration: The Impact of the Organizational Environment*, Promotor: Prof.dr. D.L. van Knippenberg, EPS-2016-376-ORG, http://repub.eur.nl/pub/79409

Eskenazi, P.I., *The Accountable Animal*, Promotor(s): Prof.dr. F.G.H. Hartmann, EPS- 2015-355-F&A, http://repub.eur.nl/pub/78300

Essen, M. van, *An Institution-Based View of Ownership*, Promotor(s): Prof.dr. J. van Oosterhout & Prof.dr. G.M.H. Mertens, EPS-2011-226-ORG, http://repub.eur.nl/pub/22643

Evangelidis, I., *Preference Construction under Prominence*, Promotor(s): Prof.dr. S.M.J. van Osselaer, EPS-2015-340-MKT, http://repub.eur.nl/pub/78202

Faber, N., *Structuring Warehouse Management*, Promotor(s): Prof.dr. MB.M. de Koster, Prof.dr. Ale Smidts, EPS-2015-336-LIS, http://repub.eur.nl/pub/78603

Fernald, K., *The Waves of Biotechnological Innovation in Medicine: Interfirm Cooperation Effects and a Venture Capital Perspective*, Promotor(s): Prof.dr. E.Claassen, Prof.dr. H.P.G.Pennings & Prof.dr. H.R. Commandeur, EPS-2015-371-S&E, http://repub.eur.nl/pub/79120

Fourne, S.P., *Managing Organizational Tensions: A Multi-Level Perspective on Exploration, Exploitation and Ambidexterity*, Promotor(s): Prof.dr. J.J.P. Jansen & Prof.dr. S.J. Magala, EPS-2014-318-S&E, http://repub.eur.nl/pub/76075

Gharehgozli, A.H., *Developing New Methods for Efficient Container Stacking Operations*, Promotor(s): Prof.dr.ir. M.B.M. de Koster, EPS-2012-269-LIS, http://repub.eur.nl/pub/37779

Gils, S. van, *Morality in Interactions: On the Display of Moral Behavior by Leaders and Employees*, Promotor(s): Prof.dr. D.L. van Knippenberg, EPS-2012-270-ORG, http://repub.eur.nl/pub/38027

Ginkel-Bieshaar, M.N.G. van, *The Impact of Abstract versus Concrete Product Communications on Consumer Decision-making Processes*, Promotor(s): Prof.dr.ir. B.G.C. Dellaert, EPS-2012-256-MKT, http://repub.eur.nl/pub/31913

Gkougkousi, X., *Empirical Studies in Financial Accounting*, Promotor(s): Prof.dr. G.M.H. Mertens & Prof.dr. E. Peek, EPS-2012-264-F&A, http://repub.eur.nl/pub/37170

Glorie, K.M., *Clearing Barter Exchange Markets: Kidney Exchange and Beyond*, Promotor(s): Prof.dr. A.P.M. Wagelmans & Prof.dr. J.J. van de Klundert, EPS-2014-329-LIS, http://repub.eur.nl/pub/77183

Hekimoglu, M., *Spare Parts Management of Aging Capital Products*, Promotor: Prof.dr.ir. R. Dekker, EPS-2015-368-LIS, http://repub.eur.nl/pub/79092

Heij, C.V., *Innovating beyond Technology. Studies on how management innovation, co-creation and business model innovation contribute to firm's (innovation) performance*, Promotor(s): Prof.dr.ing. F.A.J. van den Bosch & Prof.dr. H.W. Volberda, EPS-2012-370-STR, http://repub.eur.nl/pub/78651

Heyde Fernandes, D. von der, *The Functions and Dysfunctions of Reminders*, Promotor(s): Prof.dr. S.M.J. van Osselaer, EPS-2013-295-MKT, http://repub.eur.nl/pub/41514

Heyden, M.L.M., *Essays on Upper Echelons & Strategic Renewal: A Multilevel Contingency Approach*, Promotor(s): Prof.dr.ing. F.A.J. van den Bosch & Prof.dr. H.W. Volberda, EPS-2012-259-STR, http://repub.eur.nl/pub/32167

Hoever, I.J., *Diversity and Creativity*, Promotor(s): Prof.dr. D.L. van Knippenberg, EPS-2012-267-ORG, http://repub.eur.nl/pub/37392

Hogenboom, A.C., *Sentiment Analysis of Text Guided by Semantics and Structure*, Promotor(s):Prof.dr.ir. U.Kaymak & Prof.dr. F.M.G. de Jong, EPS-2015-369-LIS, http://repub.eur.nl/pub/79034

Hogenboom, F.P., *Automated Detection of Financial Events in News Text*, Promotor(s): Prof.dr.ir. U. Kaymak & Prof.dr. F.M.G. de Jong, EPS-2014-326-LIS, http://repub.eur.nl/pub/77237

Hollen, R.M.A., *Exploratory Studies into Strategies to Enhance Innovation-Driven International Competitiveness in a Port Context: Toward Ambidextrous Ports*, Promotor(s) Prof.dr.ing. F.A.J. Van Den Bosch & Prof.dr. H.W.Volberda, EPS-2015-372-S&E, http://repub.eur.nl/pub/78881

Hoogendoorn, B., *Social Entrepreneurship in the Modern Economy: Warm Glow, Cold Feet*, Promotor(s): Prof.dr. H.P.G. Pennings & Prof.dr. A.R. Thurik, EPS-2011-246-STR, http://repub.eur.nl/pub/26447

Hoogervorst, N., *On The Psychology of Displaying Ethical Leadership: A Behavioral Ethics Approach*, Promotor(s): Prof.dr. D. de Cremer & Dr. M. van Dijke, EPS-2011-244-ORG, http://repub.eur.nl/pub/26228

Hout, D.H. van, *Measuring Meaningful Differences: Sensory Testing Based Decision Making in an Industrial Context; Applications of Signal Detection Theory and Thurstonian Modelling*, Promotor(s): Prof.dr. P.J.F. Groenen & Prof.dr. G.B. Dijksterhuis, EPS-2014-304-MKT, http://repub.eur.nl/pub/50387

Houwelingen, G.G. van, *Something To Rely On*, Promotor(s): Prof.dr. D. de Cremer & Prof.dr. M.H. van Dijke, EPS-2014-335-ORG, http://repub.eur.nl/pub/77320

Hurk, E. van der, *Passengers, Information, and Disruption*s, Promotor(s): Prof.dr. L.G. Kroon & Prof.mr.dr. P.H.M. Vervest, EPS-2015-345-LIS, http://repub.eur.nl/pub/78275

Hytonen, K.A., *Context Effects in Valuation, Judgment and Choice: A Neuroscientific Approach*, Promotor(s): Prof.dr.ir. A. Smidts, EPS-2011-252-MKT, http://repub.eur.nl/pub/30668

Iseger, P. den, *Fourier and Laplace Transform Inversion with Applications in Finance*, Promotor(s): Prof.dr.ir. R. Dekker, EPS-2014-322-LIS, http://repub.eur.nl/pub/76954

Jaarsveld, W.L. van, *Maintenance Centered Service Parts Inventory Control*, Promotor(s): Prof.dr.ir. R. Dekker, EPS-2013-288-LIS, http://repub.eur.nl/pub/39933

Jalil, M.N., *Customer Information Driven After Sales Service Management: Lessons from Spare Parts Logistics*, Promotor(s): Prof.dr. L.G. Kroon, EPS-2011-222-LIS, http://repub.eur.nl/pub/22156

Kappe, E.R., *The Effectiveness of Pharmaceutical Marketing*, Promotor(s): Prof.dr. S. Stremersch, EPS-2011-239-MKT, http://repub.eur.nl/pub/23610

Karreman, B., *Financial Services and Emerging Markets*, Promotor(s): Prof.dr. G.A. van der Knaap & Prof.dr. H.P.G. Pennings, EPS-2011-223-ORG, http://repub.eur.nl/pub/22280

Khanagha, S., *Dynamic Capabilities for Managing Emerging Technologies*, Promotor(s): Prof.dr. H.W. Volberda, EPS-2014-339-S&E, http://repub.eur.nl/pub/77319

Kil, J., *Acquisitions Through a Behavioral and Real Options Lens*, Promotor(s): Prof.dr. H.T.J. Smit, EPS-2013-298-F&A, http://repub.eur.nl/pub/50142

Klooster, E. van 't, *Travel to Learn: the Influence of Cultural Distance on Competence Development in Educational Travel*, Promotor(s): Prof.dr. F.M. Go & Prof.dr. P.J. van Baalen, EPS-2014-312-MKT, http://repub.eur.nl/pub/51462

Koendjbiharie, S.R., *The Information-Based View on Business Network Performance: Revealing the Performance of Interorganizational Networks*, Promotor(s): Prof.dr.ir. H.W.G.M. van Heck & Prof.mr.dr. P.H.M. Vervest, EPS-2014-315-LIS, http://repub.eur.nl/pub/51751

Koning, M., *The Financial Reporting Environment: The Role of the Media, Regulators and Auditors*, Promotor(s): Prof.dr. G.M.H. Mertens & Prof.dr. P.G.J. Roosenboom, EPS-2014-330-F&A, http://repub.eur.nl/pub/77154

Konter, D.J., *Crossing Borders with HRM: An Inquiry of the Influence of Contextual Differences in the Adoption and Effectiveness of HRM*, Promotor(s): Prof.dr. J. Paauwe & Dr. L.H. Hoeksema, EPS-2014-305-ORG, http://repub.eur.nl/pub/50388

Korkmaz, E., *Bridging Models and Business: Understanding Heterogeneity in Hidden Drivers of Customer Purchase Behavior*, Promotor(s): Prof.dr. S.L. van de Velde & Prof.dr. D. Fok, EPS-2014-316-LIS, http://repub.eur.nl/pub/76008

Kroezen, J.J., *The Renewal of Mature Industries: An Examination of the Revival of the Dutch Beer Brewing Industry*, Promotor(s): Prof.dr. P.P.M.A.R. Heugens, EPS-2014- 333-S&E, http://repub.eur.nl/pub/77042

Kysucky, V., *Access to Finance in a Cros-Country Context*, Promotor(s): Prof.dr. L. Norden, EPS-2015-350-F&A, http://repub.eur.nl/pub/78225

Lam, K.Y., *Reliability and Rankings*, Promotor(s): Prof.dr. Ph.H.B.F. Franses, EPS-2011-230-MKT, http://repub.eur.nl/pub/22977

Lander, M.W., *Profits or Professionalism? On Designing Professional Service Firms*, Promotor(s): Prof.dr. J. van Oosterhout & Prof.dr. P.P.M.A.R. Heugens, EPS-2012-253- ORG, http://repub.eur.nl/pub/30682

Langhe, B. de, *Contingencies: Learning Numerical and Emotional Associations in an Uncertain World*, Promotor(s): Prof.dr.ir. B. Wierenga & Prof.dr. S.M.J. van Osselaer, EPS-2011-236-MKT, http://repub.eur.nl/pub/23504

Lee, C.I.S.G, *Big Data in Management Research: Exploring New Avenues*, Promotor(s): Prof.dr. S.J. Magala & Dr W.A. Felps, EPS-2016-365-ORG, http://repub.eur.nl/pub/79818

Legault-Tremblay, P.O., *Corporate Governance During Market Transition: Heterogeneous responses to Institution Tensions in China*, Promotor: Prof.dr. B. Krug, EPS-2015-362-ORG, http://repub.eur.nl/pub/78649

Lenoir, A.S. *Are You Talking to Me? Addressing Consumers in a Globalised World*, Promotor(s) Prof.dr. S. Puntoni & Prof.dr. S.M.J. van Osselaer, EPS-2015-363-MKT, http://repub.eur.nl/pub/79036

Leunissen, J.M., *All Apologies: On the Willingness of Perpetrators to Apologize*, Promotor(s): Prof.dr. D. de Cremer & Dr. M. van Dijke, EPS-2014-301-ORG, http://repub.eur.nl/pub/50318

Li, D., *Supply Chain Contracting for After-sales Service and Product Support*, Promotor(s): Prof.dr.ir. M.B.M. de Koster, EPS-2015-347-LIS, http://repub.eur.nl/pub/78526

Li, Z., *Irrationality: What, Why and How*, Promotor(s): Prof.dr. H. Bleichrodt, Prof.dr. P.P. Wakker, & Prof.dr. K.I.M. Rohde, EPS-2014-338-MKT, http://repub.eur.nl/pub/77205

Liang, Q.X., *Governance, CEO Identity, and Quality Provision of Farmer Cooperatives*, Promotor(s): Prof.dr. G.W.J. Hendrikse, EPS-2013-281-ORG, http://repub.eur.nl/pub/39253

Liket, K., *Why 'Doing Good' is not Good Enough: Essays on Social Impact Measurement*, Promotor(s): Prof.dr. H.R. Commandeur & Dr. K.E.H. Maas, EPS-2014-307-STR, http://repub.eur.nl/pub/51130

Loos, M.J.H.M. van der, *Molecular Genetics and Hormones: New Frontiers in Entrepreneurship Research*, Promotor(s): Prof.dr. A.R. Thurik, Prof.dr. P.J.F. Groenen, & Prof.dr. A. Hofman, EPS-2013-287-S&E, http://repub.eur.nl/pub/40081

Lovric, M., *Behavioral Finance and Agent-Based Artificial Markets*, Promotor(s): Prof.dr. J. Spronk & Prof.dr.ir. U. Kaymak, EPS-2011-229-F&A, http://repub.eur.nl/pub/22814

Lu, Y., *Data-Driven Decision Making in Auction Markets*, Promotor(s): Prof.dr.ir. H.W.G.M. van Heck & Prof.dr. W. Ketter, EPS-2014-314-LIS, http://repub.eur.nl/pub/51543

Ma, Y., *The Use of Advanced Transportation Monitoring Data for Official Statistics*, Promotors: Prof. L.G. Kroon and Dr Jan van Dalen, EPS-2016-391-LIS, http://repub.eur.nl/pub/80174

Manders, B., *Implementation and Impact of ISO 9001*, Promotor(s): Prof.dr. K. Blind, EPS-2014-337-LIS, http://repub.eur.nl/pub/77412

Markwat, T.D., *Extreme Dependence in Asset Markets Around the Globe*, Promotor(s): Prof.dr. D.J.C. van Dijk, EPS-2011-227-F&A, http://repub.eur.nl/pub/22744

Mees, H., *Changing Fortunes: How China's Boom Caused the Financial Crisis*, Promotor(s): Prof.dr. Ph.H.B.F. Franses, EPS-2012-266-MKT, http://repub.eur.nl/pub/34930

Mell, J.N., *Connecting Minds: On The Role of Metaknowledge in Knowledge Coordination*, Promotor: Prof.dr.D.L. van Knippenberg, EPS-2015-359-ORG, http://repub.eur.nl/pub/78951

Meuer, J., *Configurations of Inter-firm Relations in Management Innovation: A Study in China's Biopharmaceutical Industry*, Promotor(s): Prof.dr. B. Krug, EPS-2011-228-ORG, http://repub.eur.nl/pub/22745

Micheli, M.R., *Business Model Innovation: A Journey across Managers' Attention and Inter-Organizational Networks*, Promotor(s): Prof.dr. J.J.P. Jansen, EPS-2015-344-S&E, http://repub.eur.nl/pub/78241

Mihalache, O.R., *Stimulating Firm Innovativeness: Probing the Interrelations between Managerial and Organizational Determinants*, Promotor(s): Prof.dr. J.J.P. Jansen, Prof.dr.ing. F.A.J. van den Bosch, & Prof.dr. H.W. Volberda, EPS-2012-260-S&E, http://repub.eur.nl/pub/32343

Milea, V., *News Analytics for Financial Decision Support*, Promotor(s): Prof.dr.ir. U. Kaymak, EPS-2013-275-LIS, http://repub.eur.nl/pub/38673

Naumovska, I., *Socially Situated Financial Markets: A Neo-Behavioral Perspective on Firms, Investors and Practices*, Promotor(s): Prof.dr. P.P.M.A.R. Heugens & Prof.dr. A. de Jong, EPS-2014-319-S&E, http://repub.eur.nl/pub/76084

Nielsen, L.K., *Rolling Stock Rescheduling in Passenger Railways: Applications in short term planning and in disruption management*, Promotor(s): Prof.dr. L.G. Kroon, EPS- 2011-224-LIS, http://repub.eur.nl/pub/22444

Nuijten, A.L.P., *Deaf Effect for Risk Warnings: A Causal Examination applied to Information Systems Projects*, Promotor(s): Prof.dr. G.J. van der Pijl, Prof.dr. H.R. Commandeur & Prof.dr. M. Keil, EPS-2012-263-S&E, http://repub.eur.nl/pub/34928

Oord, J.A. van, *Essays on Momentum Strategies in Finance*, Promotor: Prof. H.K. van Dijk, EPS-2016-380-F&A, http://repub.eur.nl/pub/80036

Osadchiy, S.E., *The Dynamics of Formal Organization: Essays on bureaucracy and formal rules*, Promotor(s): Prof.dr. P.P.M.A.R. Heugens, EPS-2011-231-ORG, http://repub.eur.nl/pub/23250

Ozdemir, M.N., *Project-level Governance, Monetary Incentives, and Performance in Strategic R&D Alliances*, Promotor(s): Prof.dr.ir. J.C.M. van den Ende, EPS-2011-235-LIS, http://repub.eur.nl/pub/23550

Peers, Y., *Econometric Advances in Diffusion Models*, Promotor(s): Prof.dr. Ph.H.B.F. Franses, EPS-2011-251-MKT, http://repub.eur.nl/pub/30586

Peters, M., *Machine Learning Algorithms for Smart Electricity Markets*, Promotor(s): Prof.dr. W. Ketter, EPS-2014-332-LIS, http://repub.eur.nl/pub/77413

Porck, J., *No Team is an Island: An Integrative View of Strategic Consensus between Groups*, Promotor(s): Prof.dr. P.J.F. Groenen & Prof.dr. D.L. van Knippenberg, EPS- 2013-299-ORG, http://repub.eur.nl/pub/50141

Porras Prado, M., *The Long and Short Side of Real Estate, Real Estate Stocks, and Equity*, Promotor(s): Prof.dr. M.J.C.M. Verbeek, EPS-2012-254-F&A, http://repub.eur.nl/pub/30848

Poruthiyil, P.V., *Steering Through: How organizations negotiate permanent uncertainty and unresolvable choices*, Promotor(s): Prof.dr. P.P.M.A.R. Heugens & Prof.dr. S.J. Magala, EPS-2011-245-ORG, http://repub.eur.nl/pub/26392

Pourakbar, M., *End-of-Life Inventory Decisions of Service Parts*, Promotor(s): Prof.dr.ir. R. Dekker, EPS-2011-249-LIS, http://repub.eur.nl/pub/30584

Pronker, E.S., *Innovation Paradox in Vaccine Target Selection*, Promotor(s): Prof.dr. H.J.H.M. Claassen & Prof.dr. H.R. Commandeur, EPS-2013-282-S&E, http://repub.eur.nl/pub/39654

Protzner, S. *Mind the gap between demand and supply: A behavioral perspective on demand forecasting*, Promotor(s): Prof.dr. S.L. van de Velde & Dr. L. Rook, EPS-2015-364-LIS, http://repub.eur.nl/pub/79355

Pruijssers, J.K., *An Organizational Perspective on Auditor Conduct*, Promotor(s): Prof.dr. J. van Oosterhout & Prof.dr. P.P.M.A.R. Heugens, EPS-2015-342-S&E, http://repub.eur.nl/pub/78192

Retel Helmrich, M.J., *Green Lot-Sizing*, Promotor(s): Prof.dr. A.P.M. Wagelmans, EPS- 2013-291-LIS, http://repub.eur.nl/pub/41330

Rietdijk, W.J.R. *The Use of Cognitive Factors for Explaining Entrepreneurship*, Promotor(s): Prof.dr. A.R. Thurik & Prof.dr. I.H.A. Franken, EPS-2015-356-S&E, http://repub.eur.nl/pub/79817

Rietveld, N., *Essays on the Intersection of Economics and Biology*, Promotor(s): Prof.dr. A.R. Thurik, Prof.dr. Ph.D. Koellinger, Prof.dr. P.J.F. Groenen, & Prof.dr. A. Hofman, EPS-2014-320-S&E, http://repub.eur.nl/pub/76907

Rijsenbilt, J.A., *CEO Narcissism: Measurement and Impact*, Promotor(s): Prof.dr. A.G.Z. Kemna & Prof.dr. H.R. Commandeur, EPS-2011-238-STR, http://repub.eur.nl/pub/23554

Rösch, D. *Market Efficiency and Liquidity*, Promotor: Prof.dr. M.A. van Dijk, EPS-2015-353-F&A, http://repub.eur.nl/pub/79121/

Roza-van Vuren, M.W., *The Relationship between Offshoring Strategies and Firm Performance: Impact of innovation, absorptive capacity and firm size*, Promotor(s): Prof.dr. H.W. Volberda & Prof.dr.ing. F.A.J. van den Bosch, EPS-2011-214-STR, http://repub.eur.nl/pub/22155

Rubbaniy, G., *Investment Behaviour of Institutional Investors*, Promotor(s): Prof.dr. W.F.C. Verschoor, EPS-2013-284-F&A, http://repub.eur.nl/pub/40068

Schoonees, P. *Methods for Modelling Response Styles*, Promotor: Prof.dr P.J.F. Groenen, EPS-2015-348-MKT, http://repub.eur.nl/pub/79327

Schouten, M.E., *The Ups and Downs of Hierarchy: the causes and consequences of hierarchy struggles and positional loss*, Promotors; Prof. D.L. van Knippenberg & Dr L.L. Greer, EPS-2016-386-ORG, http://repub.eur.nl/pub/80059

Shahzad, K., *Credit Rating Agencies, Financial Regulations and the Capital Markets*, Promotor(s): Prof.dr. G.M.H. Mertens, EPS-2013-283-F&A, http://repub.eur.nl/pub/39655

Sousa, M.J.C. de, *Servant Leadership to the Test: New Perspectives and Insights*, Promotor(s): Prof.dr. D.L. van Knippenberg & Dr. D. van Dierendonck, EPS-2014-313-ORG, http://repub.eur.nl/pub/51537

Spliet, R., *Vehicle Routing with Uncertain Demand*, Promotor(s): Prof.dr.ir. R. Dekker, EPS-2013-293-LIS, http://repub.eur.nl/pub/41513

Staadt, J.L., *Leading Public Housing Organisation in a Problematic Situation: A Critical Soft Systems Methodology Approach*, Promotor(s): Prof.dr. S.J. Magala, EPS-2014-308- ORG, http://repub.eur.nl/pub/50712

Stallen, M., *Social Context Effects on Decision-Making: A Neurobiological Approach*, Promotor(s): Prof.dr.ir. A. Smidts, EPS-2013-285-MKT, http://repub.eur.nl/pub/39931

Tarakci, M., *Behavioral Strategy: Strategic Consensus, Power and Networks*, Promotor(s): Prof.dr. D.L. van Knippenberg & Prof.dr. P.J.F. Groenen, EPS-2013-280-ORG, http://repub.eur.nl/pub/39130

Teixeira de Vasconcelos, M., *Agency Costs, Firm Value, and Corporate Investment*, Promotor(s): Prof.dr. P.G.J. Roosenboom, EPS-2012-265-F&A, http://repub.eur.nl/pub/37265

Tröster, C., *Nationality Heterogeneity and Interpersonal Relationships at Work*, Promotor(s): Prof.dr. D.L. van Knippenberg, EPS-2011-233-ORG, http://repub.eur.nl/pub/23298

Tsekouras, D., *No Pain No Gain: The Beneficial Role of Consumer Effort in Decision-Making*, Promotor(s): Prof.dr.ir. B.G.C. Dellaert, EPS-2012-268-MKT, http://repub.eur.nl/pub/37542

Tuijl, E. van, *Upgrading across Organisational and Geographical Configurations*, Promotor(s): Prof.dr. L. van den Berg, EPS-2015-349-S&E, http://repub.eur.nl/pub/78224

Tunçdoğan, A., *Decision Making and Behavioral Strategy: The Role of Regulatory Focus in Corporate Innovation Processes*, Promotor(s): Prof.dr.ing. F.A.J. van den Bosch, Prof.dr. H.W. Volberda, & Prof.dr. T.J.M. Mom, EPS-2014-334-S&E, http://repub.eur.nl/pub/76978

Uijl, S. den, *The Emergence of De-facto Standards*, Promotor(s): Prof.dr. K. Blind, EPS-2014-328-LIS, http://repub.eur.nl/pub/77382

Vagias, D., *Liquidity, Investors and International Capital Markets*, Promotor(s): Prof.dr. M.A. van Dijk, EPS-2013-294-F&A, http://repub.eur.nl/pub/41511

Veelenturf, L.P., *Disruption Management in Passenger Railways: Models for Timetable, Rolling Stock and Crew Rescheduling*, Promotor(s): Prof.dr. L.G. Kroon, EPS-2014-327-LIS, http://repub.eur.nl/pub/77155

Venus, M., *Demystifying Visionary Leadership: In search of the essence of effective vision communication*, Promotor(s): Prof.dr. D.L. van Knippenberg, EPS-2013-289-ORG, http://repub.eur.nl/pub/40079

Vermeer, W., *Propagation in Networks: The impact of information processing at the actor level on system-wide propagation dynamics*, Promotor: Prof.mr.dr. P.H.M.Vervest, EPS-2015-373-LIS, http://repub.eur.nl/pub/79325

Versluis, I., *Prevention of the Portion Size Effect*, Promotors: Prof. Ph.H.B.F. Franses & Dr E.K. Papies, EPS-2016-382-MKT, http://repub.eur.nl/pub/79880

Visser, V.A., *Leader Affect and Leadership Effectiveness: How leader affective displays influence follower outcomes*, Promotor(s): Prof.dr. D.L. van Knippenberg, EPS-2013-286-ORG, http://repub.eur.nl/pub/40076

Vlam, A.J., *Customer First? The Relationship between Advisors and Consumers of Financial Products*, Promotor(s): Prof.dr. Ph.H.B.F. Franses, EPS-2011-250-MKT, http://repub.eur.nl/pub/30585

Vries, J. de, *Behavioral Operations in Logistics*, Promotor(s): Prof.dr M.B.M de Koster & Prof.dr. D.A. Stam, EPS-2015-374-LIS, http://repub.eur.nl/pub/79705

Waltman, L., *Computational and Game-Theoretic Approaches for Modeling Bounded Rationality*, Promotor(s): Prof.dr.ir. R. Dekker & Prof.dr.ir. U. Kaymak, EPS-2011-248- LIS, http://repub.eur.nl/pub/26564

Wang, T., *Essays in Banking and Corporate Finance*, Promotor(s): Prof.dr. L. Norden & Prof.dr. P.G.J. Roosenboom, EPS-2015-352-F&A, http://repub.eur.nl/pub/78301

Wang, Y., *Information Content of Mutual Fund Portfolio Disclosure*, Promotor(s): Prof.dr. M.J.C.M. Verbeek, EPS-2011-242-F&A, http://repub.eur.nl/pub/26066

Wang, Y., *Corporate Reputation Management: Reaching Out to Financial Stakeholders*, Promotor(s): Prof.dr. C.B.M. van Riel, EPS-2013-271-ORG, http://repub.eur.nl/pub/38675

Weenen, T.C., *On the Origin and Development of the Medical Nutrition Industry*, Promotor(s): Prof.dr. H.R. Commandeur & Prof.dr. H.J.H.M. Claassen, EPS-2014-309-S&E, http://repub.eur.nl/pub/51134

Wolfswinkel, M., *Corporate Governance, Firm Risk and Shareholder Value*, Promotor(s): Prof.dr. A. de Jong, EPS-2013-277-F&A, http://repub.eur.nl/pub/39127

Yang, S., *Information Aggregation Efficiency of Prediction Markets*, Promotor(s): Prof.dr.ir. H.W.G.M. van Heck, EPS-2014-323-LIS, http://repub.eur.nl/pub/77184

Zaerpour, N., *Efficient Management of Compact Storage Systems*, Promotor(s): Prof.dr.ir. M.B.M. de Koster, EPS-2013-276-LIS, http://repub.eur.nl/pub/38766

Zhang, D., *Essays in Executive Compensation*, Promotor(s): Prof.dr. I. Dittmann, EPS- 2012-261-F&A, http://repub.eur.nl/pub/32344

Zwan, P.W. van der, *The Entrepreneurial Process: An International Analysis of Entry and Exit*, Promotor(s): Prof.dr. A.R. Thurik & Prof.dr. P.J.F. Groenen, EPS-2011-234-ORG, http://repub.eur.nl/pub/23422

Order picking, the process of retrieving customer orders from their storage locations, is the most critical operation in a warehouse. Any under performance in order picking can lead to unsatisfactory service and high operational cost for the warehouse, and consequently for the whole supply chain in which the company operates. This thesis develops new stochastic models for the performance evaluation of two state-of-the art order picking systems: zone picking and polling-based milkrun picking. These models adequately describe and predict the consequences of variability on the performance of these warehousing systems.

The first part of the thesis zone picking systems are studied, one of the most popular conveyor-based picker-to-parts order picking methods used in practice. We model the various elements of the system including conveyor merges as a network of queues with multiple order classes, with capacity constraints on subnetworks, and with the dynamic block-and-recirculate protocol. The resulting model is most suitable to support rapid and optimal design of complex zone picking systems. In the second part of the thesis, milkrun picking systems are investigated. In this system an order picker picks multiple orders that arrive in real-time and integrates them in the current picking cycle. This subsequently changes dynamically the stops on the order picker's picking route. Using polling models, we study order throughput times for various picking policies, and the effect of product allocation. The results of the model show that when the order arrival rate is high milkrun order picking significantly improves system performance compared to conventional batch picking. In addition, the best product allocation improves the order throughput time considerably.

ERiM

The Erasmus Research Institute of Management (ERIM) is the Research School (Onderzoekschool) in the field of management of the Erasmus University Rotterdam. The founding participants of ERIM are the Rotterdam School of Management (RSM), and the Erasmus School of Economics (ESE). ERIM was founded in 1999 and is officially accredited by the Royal Netherlands Academy of Arts and Sciences (KNAW). The research undertaken by ERIM is focused on the management of the firm in its environment, its intra- and interfirm relations, and its business processes in their interdependent connections.

The objective of ERIM is to carry out first rate research in management, and to offer an advanced doctoral programme in Research in Management. Within ERIM, over three hundred senior researchers and PhD candidates are active in the different research programmes. From a variety of academic backgrounds and expertises, the ERIM community is united in striving for excellence and working at the forefront of creating new business knowledge.

ERiM

ERIM PhD Series
Research in Management

Erasmus University Rotterdam (EUR)
Erasmus Research Institute of Management
Mandeville (T) Building
Burgemeester Oudlaan 50
3062 PA Rotterdam, The Netherlands

P.O. Box 1738
3000 DR Rotterdam, The Netherlands
T +31 10 408 1182
E info@erim.eur.nl
W www.erim.eur.nl