EINDHOVEN UNIVERSITY OF TECHNOLOGY
DEPARTMENT OF MATHEMATICS AND COMPUTER SCIENCE

MASTER'S THESIS

Performance Analysis of Zone Picking Systems

R.F. Bakker

Supervisors:
Dr. ir. I.J.B.F. Adan (TU/e)
Dr. J.A.C. Resing (TU/e)
Ir. M.S. van den Broek (Vanderlande Industries)
B.M.L. Dumoulin, MSc. (Vanderlande Industries)

Eindhoven, December 2007

Acknowledgements

This thesis is the result of a nine-month internship at Vanderlande Industries, as well the conclusion of the master's program in applied mathematics at the Eindhoven University of Technology. Numerous people have helped me to bring this thesis to a good end, and I am greatly indebted to them.

First I would like to thank my supervisors from the Eindhoven University of Technology, Ivo Adan and Jacques Resing. I enjoyed discussing the mathematical problems we encountered. They gave me the freedom to shape my research and commented several versions of my thesis, for which I owe them much gratitude.

Furthermore I am grateful to my daily supervisor at Vanderlande Industries, Monique van den Broek, for coping with me for the past nine months. During my internship she helped where needed and her enthusiasm was infectious. Likewise I am grateful for Bas Dumoulin, also of Vanderlande Industries, for making this internship possible and his practical input. Moreover I would like to thank Stef van Eijndhoven for being a member of my graduation committee.

The project was carried out at the Simulation department of Vanderlande Industries and the group Stochastic Operations Research of the TU/e. I enjoyed my stay at both groups very much and I would like to thank all the people for their hospitality and kindness.

Finally I would like to thank my family for their support throughout my studies and my friends for the wonderful years.

Richard Bakker
Eindhoven, December 2007

Summary

Zone Picking Systems (ZPS) is an orderpicking concept within the company Vanderlande Industries. The concept of such systems is based on the assembly of an order by passing a tote from zone to zone. Such a zone is manned by an operator and so an order is fulfilled by multiple operators. The importance of such systems to Vanderlande Industries can be illustrated with the start of a special group called the 'ZPS A-team'. In this team all disciplines of the company are combined to work efficiently and reduce costs.

The modelling and analysis of these systems is not as straightforward because the sequence in which zones are visited is not pre-determined; if a tote arrives at a zone which is fully occupied the tote is 'blocked' and travels to his next destination and at the end there will be checked if a tote has visited all the zones. If not the tote will be recirculated, otherwise the tote will leave the system.

An iterative method is developed which is based on jump-over blocking [6], and especially the corresponding product form solution of the limiting probabilities. For small systems, or small number of zones, this method is accurate and also very fast. However for large systems, the normalizing constant becomes the limiting factor of the speed of the method. Therefore a mean value analysis is used in each iteration, to reduce calculation time. To compare the method, a simulation is made. The results for utilization, throughput and mean buffersize are in general accurate.

In reality an operator sometimes suffers from hindering of totes on the conveyor when he needs to place a tote back on the conveyor. This is included to make the model more realistic. Results are in general again accurate and the settings for which a specific ZPS design is optimized can be determined using this method. The model is also expanded for a so-called multi-segment ZPS, but for these systems more research is needed to determine the accuracy. Furthermore it seems that the model, with some minor adjustments, can be applied to other picking concepts within Vanderlande Industries, but also here more research is desirable.

Contents

Chapter 1

Introduction

The graduation project in this thesis has been carried out at Vanderlande Industries and focuses on Zone Picking Systems (ZPS). In this chapter we will first give an introduction to Vanderlande Industries, where this research was done for the past nine months. A brief history will be given of the company and something more will be told about the Simulation group where this research took place. Some terminology used throughout this theses is listed. In section 1.3 some background information about ZPS will be given, because more insight is desirable. Furthermore some assumptions will be stated that will be valid for all problems in this thesis. In the final section the problem will be described and also an overview of this thesis will be given.

1.1 Vanderlande Industries

According to [18], Vanderlande Industries is a prominent worldwide player in material handling systems and has been on this market since 1949. The company focuses on the implementation of automated material handling processes at companies. The solutions Vanderlande Idustries offers can range from small and medium-sized systems through the worlds largest. The areas of business are Baggage Handling, Warehousing and Distribution and Express Parcelling. The head office is located in Veghel and several Customer centres are located in Belgium, France, Germany, Spain, United Kingdom, China, South Africa and the USA. This section will shortly describe the history, strategy and an overview of the Systems Simulation where this research is carried out.

1.1.1 History

In 1949, Vanderlande Industries was established in Veghel. Eddy van der Lande established 'Machinefabriek E. van der Lande' as a general machinery and construction company. In the beginning, the company was dedicated to the service and repair of weaving looms and other machinery for the textile industry. After a few years the core business of the company moved to material handling systems like cranes, trolleys and conveyors. In 1962 the company joined forces with the American organization 'the Rapids Standard Company'. The partnership ensured acceleration of the growth in the field of material handling. The name of the company changed into 'Rapistan Lande'. In the late 1980's a management buy-out led to the separation of Rapistan and all the shares returned to Dutch ownership. The company's name changed into Vanderlande Industries as which it is still known today. By integrating the German software company

GamBit into the organization in 1997, the company is now able to design, develop and integrate warehouse management systems into their material handling solutions.

1.1.2 Strategy

Vanderlande is a project driven organization, which means that when one of the customer centres sells a project, the rest of the organization will start working to fulfill this order. Most of the preliminary work is done at the company's main location in Veghel. Also supporting departments as R&D, Engineering and Manufacturing are based in Veghel. The company focuses its activities on three market segments namely warehouse and distribution centres, express parcelling, baggage handling for airports, and their related maintenance services.

In the distribution segment, Vanderlande provides automated solutions for storage, order picking, consolidation and warehouse control in the distribution supply chain. Vanderlande is able to provide customers with solutions that improve logistic execution and that meet their demands and future growth, using the knowledge of distribution processes and warehouses available at Vanderlande.

For the express parcels industry the company offers a wide range of information and sorting systems for both parcels and letter post. Vanderlande is building systems in small local depots up to the worlds largest postal sorting centres.

Vanderlande also supplies baggage handling systems for airports around the globe. The systems are able to provide fast and safe storage, transportation and sorting of departing, transfer and arriving luggage. In this business segment, Vanderlande is among the world's top three suppliers. Vanderlande is currently, among other baggage handling projects, working on huge baggage handling systems at the new terminal (T5) at Heathrow Airport in London and working on a new pier and several upgrades on existing systems at Schiphol Airport in Amsterdam.

1.1.3 Systems Simulation

Beside the operational activities, also a number of other centralized departments can be found in Veghel. Among these departments is the Systems department.

It is the responsibility of the Systems department to technically and commercially assist customer centres and projects in developing system solutions, which meet the need and requirements of the customer. The Systems department consists of several smaller departments.

One of these departments is the Systems Simulation department. This department analyses customer's requirements and wishes, and builds digital models of the (possible) system(s).

Because of these digital models, it is possible:

1. to simulate the systems behavior, so that possible bottlenecks are discovered early,
2. to animate the system for showing the possible solution(s) to the customer, or for other commercial purposes,
3. to emulate the system for (automated) testing of controls software purposes.

Furthermore the department also provides analysis.

1.1.4 Distribution

As said earlier, one of the markets Vanderlande Industries focuses on is Distribution. Within the company five order picking solutions have been developed. These solutions can be classified in two main classes: operator-to-part systems and part-to-operator systems. In the first class of systems the man is going to the goods while in the latter class the goods are going to the man.

Operator-to-part solutions

There are three different operator-to-part solutions defined within the company:

1. Classical orderpicking: picking one order at a time, with an order having one operator who picks the whole order.
2. Zone picking: a picking area is divided in zones. Picking one part of an order in an assigned picking zone. Because zone picking systems are the subject of this final project, they will be discussed in more detail later.
3. Batch picking: picking multiple orders at a time, followed by a sorting action.

Part-to-operator solutions

Within the company there are two solutions defined:

1. Order distribution (ODS): picking of multiple orders at a time, from product totes passing the picking stations. This is like zone picking but the role of the orders and products are switched.
2. Compact picking: picking one or multiple orders at a time, from a product tote sent from a compact storage device (for example a miniload).

Vanderlande Industries uses a concept selection diagram as can be seen in Figure 1.1, which gives an indication of the most promising concept. Often it can be a mix of the several concepts. The selection of a picking concept is based on the expected number of order-lines per hour and the expected number of Storage Keeping Units (SKUs), or i.e. different products.

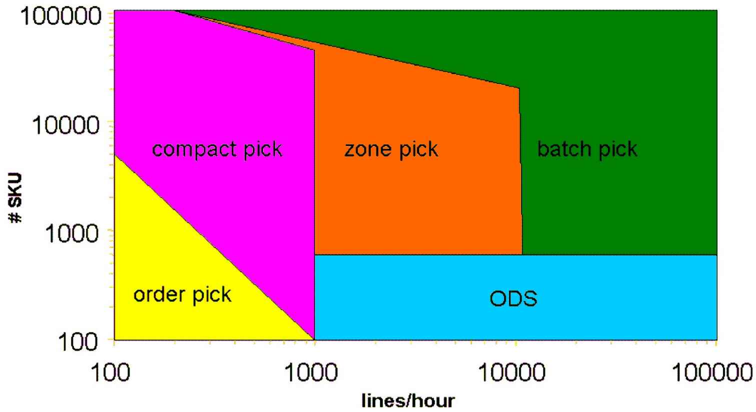


Figure 1.1: A item-picking concept selection diagram

1.2 Terminology

In this section the terminology used in this thesis is listed.

Buffer The region where temporarily totes cane be hold including the tote where the operator is working on

Deadlock Situation where the conveyor system completely blocks

Effective Utilization The percentage of time the operator is working on totes excluding additional waiting time

Gap The minimum space between two successive totes on a conveyor mostly denoted as a multiple of the length of a tote

Mean buffersize The mean number of totes in the buffer of a zone

Operator The person doing the operations on a tote in his/hers zone

Tote A carton or plastic box in which products are placed

Utilization The percentage of time the operator is working on totes including additional waiting time

Waiting positions The number of positions of the buffer minus one; the tote where the operator is working on is not included

Window An imaginary moving section of a conveyor in which a tote is or can be transported. The length of a window is defined as product length plus gap length

Zone The area where the picking of products takes place by the operator

1.3 Zone Picking Systems

One of the five solutions used within Vanderlande Industries as earlier described are the so-called Zone Picking Systems (ZPS). This research focuses on these systems. The concept of such systems is based on the assembly of an order by passing a tote from zone to zone. Such a zone is manned by an operator and so an order is fulfilled by multiple operators. Totes are released until a certain threshold is reached, then totes travel from zone to zone on a conveyor. An example of a ZPS is given in Figure 1.2, where above in the middle the tote release is positioned, on the left the picking zones and on the right the shipping area. A ZPS is a very flexible concept and it can be applied to many different processes. In [7] the principal of ZPS is described and also an overview is given of all design aspects that are relevant for ZPS. In this section we summarize those aspects as an introduction to ZPS and to gain some more insight in the complexity of ZPS.

1.3.1 Lay-out

The lay-out of a ZPS is (of course) very important when designing one. ZPS lay-outs can be classified in three different basic lay-outs or shapes:

1. Serial routing.
2. Circular routing.
3. Multi-segment routing.

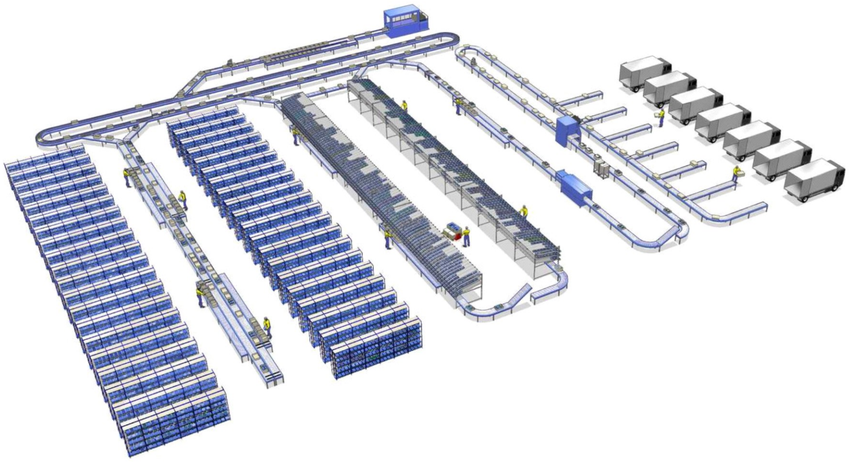


Figure 1.2: An example of a Zone Picking System

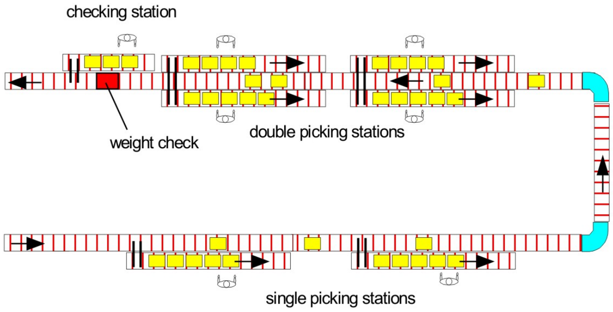


Figure 1.3: A serial routing lay-out

Serial routing

When using serial routing the sequence in which zones will be visited is pre-determined. When totes arrive at a zone but the buffer is fully occupied, the totes will have to wait on the conveyor. This is not wanted because production and money is lost. This serial routing lay-out is shown in Figure 1.3. In this figure we can see that there are different types of picking stations. These will be discussed in section 1.3.5.

Circular routing

With circular routing (Figure 1.4) the problem of stopping the conveyor, because a buffer is fully occupied, is solved. There is a possibility to convey a tote back to the beginning of the main conveyor. The main conveyor will act as a loop and with a recirculation conveyor a tote can be routed back to the beginning. In this situation the sequence in which zones are visited is not pre-determined; if a tote arrives at a zone which is fully occupied the tote is ’blocked’ and travels to his next destination and at the end there will be checked if a tote has visited all the zones. If not, the tote will be recirculated, otherwise the tote will leave the system. This recirculation is very useful, however a high amount of recirculation must be prevented to avoid saturation of

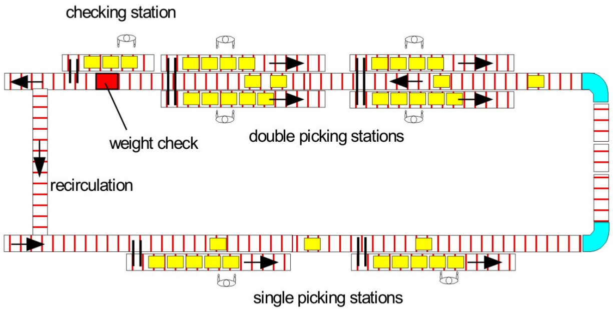


Figure 1.4: A circular routing lay-out

the main line and also a long sojourn time for the tote. Totes are released in this system until a maximum number of totes is in the system; on conveyors and in buffers. The system takes workload balancing between zones in account when releasing.

Multi-segment routing

The lay-out could be more complex by linking multiple circular segments with each other, by a (high speed) central loop, as can be seen in Figure 1.5. Here also totes are released in the system until a maximum number of totes is reached, this is done per segment and for the total system. Workload balancing is considered between zones and segments when releasing. The sequence in which a tote will visit the zones will again be different; the routing normally will be done as follows:

- A tote is put on the conveyor and will visit the first segment where it needs to be served and where the workload of the segment allows the tote to enter the segment.
- Within the segment the tote will visit all the zones, before it will leave the segment. This is the same as the circular routing lay-out described in the previous part.
- If weight checking is applied it is preferred to be done per segment to avoid mistakes and unnecessary long travel times.

1.3.2 Pick face type

There are several shapes and sizes of the racking in which the products are presented to the operators. The following criteria for choosing the right pick face have to be balanced: size of the pick face, walking distance, replenishment effort, repacking effort and the waste removal. If the pick faces are very big then replenishment would probably be easy, because you can move a large amount in little time. The negative effect is however that the walking distance will also get bigger, so an operator needs to walk a great distance to obtain certain products. There are 5 main different pick faces:

1. Pallet pick face.
2. Carton flowrack pick faces.
3. Tote flowrack pick faces.

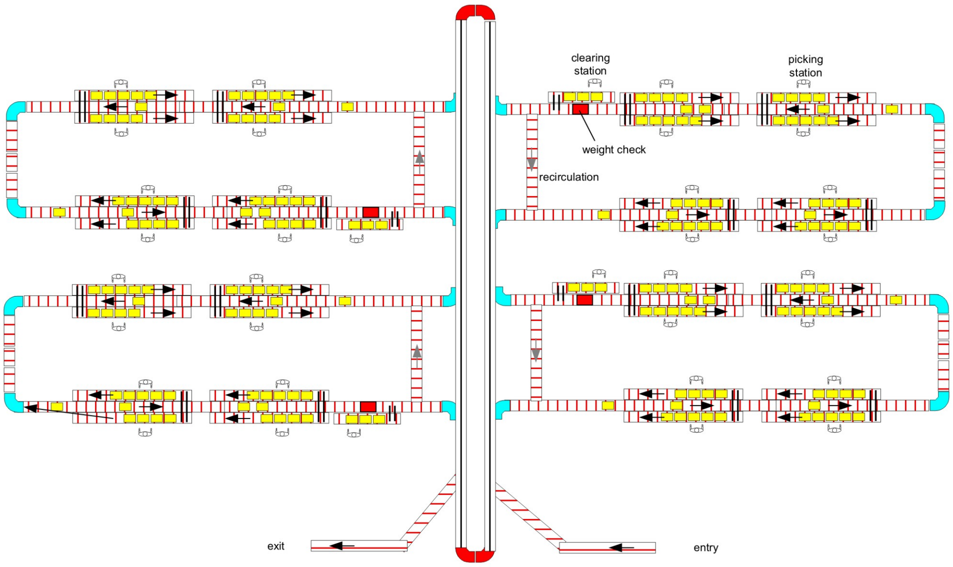


Figure 1.5: A multi-segment routing lay-out

4. Shelve racking.

5. Static or dynamic pick face.

Some of these pick faces are shown in Figure 1.6.

Pallet pick face

When using pallets the replenishment is very efficient, with a single forklift one can move a large amount of products in relatively little time and also no repacking is required. The biggest disadvantage is that the length of the pick face is very large, so this requires a lot of space for storage. Furthermore an operator has to remove packing material before he can begin to pick, this also means a lot of waste.

Carton flowrack pick face

Complete pallets are sometimes too big, so individual cartons can also be presented in a flowrack. The number of levels is normally 2 or 3 and one can be placed below the conveyor. Most of the levels will be placed over the conveyor, so fast picking is possible and also this can be done ergonomic. Furthermore a second flowrack can be situated behind the operator, the only disadvantage there is that an operator has to turn around to pick a product. Replenishment can be done from the back of the racks and during picking the operators have to remove some waste materials so a waste-conveyor could be required.

Tote flowrack pick face

The third option is to repack all the products in totes before they are presented to the operators. This will be done in a special area. The main advantages are that there is no waste in the picking area, the replenishment can be done automatically and there is less room for errors, because of the structured presentation. On the other hand there is the effort of repacking and there has to be space for a special area where the repacking is situated.

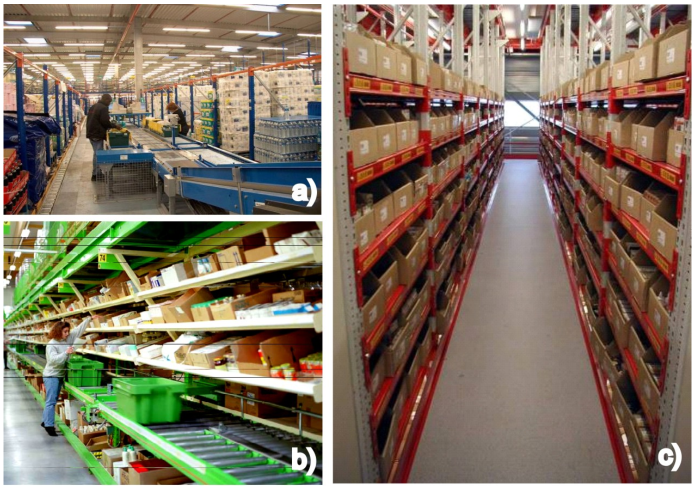


Figure 1.6: Three different pick faces: Pallet pick face (a), Carton pick face (b) and Shelve racking (c)

Shelve flowrack pick face

When large number of SKU's need to be presented to the operators, shelve racking is also an option. Aligning the shelves square to the direction of the conveyors is a commonly used layout.

Static or dynamic pick face

In the previous 4 pick faces, the products have a fixed position within the rack. This is stated as a static pick face.
Dynamic pick face is also possible when for example miniload cranes are used. The tote is returned to the storage location if the operator is finished picking. Dynamic pick face has the advantages that many SKUs can be presented on a small area and static and dynamic locations can be combined. The main disadvantage are the complex procedures.

1.3.3 Workstation type

The workstations are an important part of a ZPS; here the products are picked and when a tote is done it is placed back on the conveyor. In all configurations discussed in this section a tote will be automatically diverted into a zone. In this zone it will also automatically be transported into the buffer. When a tote eventually arrives at the operator, there are three types of workstations possible:

1. Manual handling.

2. Automatic handling.

3. Zone-lift arrangement.

Manual handling

Further handling of the tote will be strictly manual. The tote will be scanned with a handheld scanner and when the operator is finished picking he/she pushes the tote back in the main conveyor. The operator can drag the tote along the pickface, which can be helpful when picking many orderlines. The investment costs are, of course, very low because no complicated additional system needs to be bought.

Automatic handling

As the name suggests, here the system will do many things automatically; the system scans the barcode of the tote automatically and when all the orderlines are picked the tote will be transferred on the main conveyor (also) automatically. Here the costs are higher then with manual handling, because several things are required like an accumulating conveyor. This handling has as advantage that the tote will be identified automatically.

Zone-lift arrangement

This workstation type has much in common with automatic handling. Now the conveyors are situated underneath a platform on which the operator stands. The pick position is lifted to the working area by a lift. The conveyors are not obstructing potential picking locations. This zone-lift is patented by Vanderlande Industries.

1.3.4 Interface type

An operator has several possibilities to see which product needs to be picked and in which tote. In this section the following interface types will be discussed

1. Paper based picking.

2. RF-terminal.

3. Pick-to-light displays.

4. Pick-by-voice terminal.

In Figure 1.7 the interface types are shown.

Paper based picking

The interface type which is mostly used is a picking list. When a tote starts, this list will be inserted and an operator can see which and how many products he needs to put in the tote. Obviously this interface type is easy to work with and has low costs. On the other hand this way can not handle with exceptions like out of stock or when a tote is full.

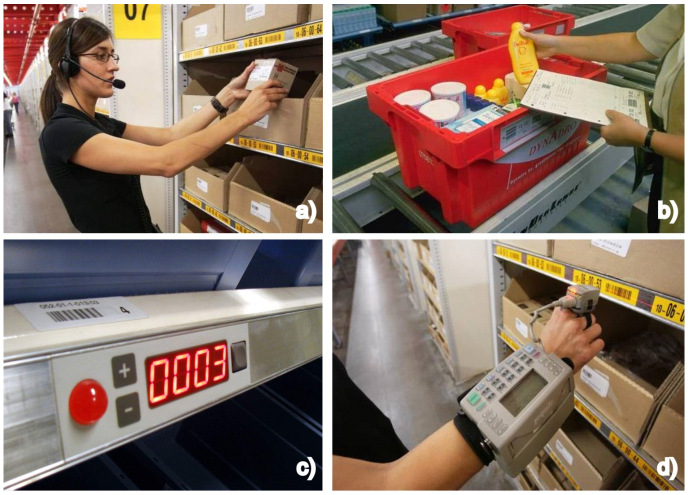


Figure 1.7: Four interface types: Pick-by-voice (a), Paper based picking (b), Pick-to-light (c) and a RF-terminal (d)

RF-terminal

A radio frequency wireless terminal (RF-terminal) has some additional possibilities; it can handle exceptions, the performance of the operator can be tracked and there is error prevention by scanning barcodes on rack or product. As can be seen in Figure 1.7(d) the main disadvantage is that the hands of the operator are not entirely free and the process is a little bit slow because of scanning and reading.

Pick-to-light displays

Using pick-to-light displays leads to faster picking, because the hands of the operator are free and the interface is very intuitive, see also Figure 1.7(c). The high investment per location can be reason to choose an other interface type.

Pick-by-voice terminal

A relatively new interface type is the use of a computer which generates speech and also can recognize voice. The pick instructions are given to the operators by verbal instructions and with a terminal the operator can confirm his/hers actions also with verbal commands. Fast picking is possible here, because hands are free and this system is also suited for any environment (frozen etc.). This system is only taught a limited number of words like number 0 to 9 and other short instructions. Also the operators voice needs to be teached to the system.

1.3.5 Zone Type

In Figure 1.3 two different picking stations (or zones) where shown. In fact there are 5 different zone concepts, all with their own specific lay-out. These can be classified in two main groups:

1. Single sided versus double sided zones.

2. Negative tote flow versus positive tote flow to the flow on the conveyor.

The concepts determine the location of the offline conveyor of the zone with its pickstation. The first one looks at the possibility to have pick locations at both sides of the main conveyor or at only one side. Negative flow means that the offline conveyor of the zone moves in opposite direction of the main conveyor and positive flow in the same direction of the main conveyor. Vanderlande Industries developed the 5 zone concepts and they will be discussed in this section. The single sided one station is the most familiar and we will assume this concept throughout this thesis.

1. Single sided 1 station: there is 1 station at one side of the main conveyor, totes travel in the buffer of the zone in the same direction as the main conveyor (positive flow).
2. Single sided 2 stations: the same as above but with 2 stations in the zone.
3. Double sided 1 station forwards: the same as number 1 but the station is situated at both sides of the conveyor.
4. Double sided 1 station backwards: the same as number 3 but the flow in the zone is negative to the flow of the main conveyor.
5. Double sided 2 station: 2 stations and at both sides of the main conveyor.

1.4 Assumptions

For clarity and simplicity, we make the following simplifying assumptions, valid for all the different problems handled in this thesis:

1. All totes are identical.
2. There is an infinite number of totes available.
3. No deadlock situation can occur.
4. Replenishment of new products is done infinitely fast.
5. The time operators work on a tote is exponentially distributed.

1.5 Problem Description

Zone Picking Systems are one of the distribution solutions which Vanderlande Industries provides to its customers. These systems are very complex, because there are a lot of variables which play a role in the performance of such systems. The importance of ZPS to Vanderlande Industries can be illustrated with the start of a special group called the 'ZPS A-team'. In this team, consisting of 5 people, all disciplines of the company are combined. The purpose is to reduce costs and to get more insight in the principle of ZPS.

Within Vanderlande Industries these systems are simulated to gain some insight in the performance of the chosen design. This process of simulating is time consuming and therefore also expensive. This is also the reason that simulation is done when the lay-out of a ZPS is determined for most of the part. For Vanderlande Industries however it is desirable that different lay-outs of ZPS can be compared with each other in an earlier phase of the selling-traject. This way some lay-outs can be discarded in a very early stage of the design phase.

Therefore the main goal of Vanderlande Industries and also the main research objective is

"Develop a tool/model which can calculate or approximate the performance measures of different ZPS lay-outs in terms of utilization of the operators and throughput of totes per hour"

In this research objective 'approximate' is also used, because exact calculation probably can not be done. If we want to compare 2 lay-outs an approximation will be sufficient, because we are only interested in which lay-out is better than the other. The ZPS lay-outs that will be considered are circular- and multi-segment routing. These are of most interest for Vanderlande Industries. In order to succeed in this objective we will have a hierarchical approach and the chapters of this thesis also follow this approach.

1.6 Thesis Outline

In this chapter we introduced the research problem and we have given some useful background information. The remainder of this thesis is structured as follows.

In chapter 2 the system is regarded which consists of one segment with only one zone. This is not a system which will be used in reality, but it does give us several models which will be useful for future purposes. These models are compared with simulation results.
In chapter 3 one of the models described in chapter 2 is expanded to a more general system. This general system is one segment with multiple zones. First the model for 2 zones is described and compared with simulation results. From this '2 zone model' it is a small step to a model with a general number of zones. A Java tool is developed which determines most important performance measures and which is easy to use. Again this model is compared with results obtained by simulation and also with known data of a ZPS analyzed by Vanderlande.

Subsequently, in chapter 4 the same model will be considered as in chapter 3. The difference is that in this chapter the conveyor will have limited capacity, in the previous chapters this capacity was assumed to be infinite. This will also be included in the Java tool. The same scenarios will be compared with simulation results and also the optimal utilization will be determined.

In chapter 5 the multi-segment model will be presented, which describes a much bigger and complicated ZPS. The picking concept called CPS will be discussed in the next chapter, where we see much similarities with our ZPS model. Finally, in the last chapter a conclusion is stated together with some recommendations for further research.

Chapter 2

One Segment, One Zone

In this chapter we will narrow our scope to a specific design, which will gain us some more insight in the concept of Zone Picking Systems. This design is one segment with only one zone. In other words this is a circular system with one zone and this zone has N waiting positions. A tote arrives on the conveyor and travels to the zone to get served. If however this zone is fully occupied, or in other words all waiting positions are occupied, the tote will continue his way on the conveyor and will make another attempt. If the tote again finds the zone fully occupied he will again continue his way on the conveyor and try to make another attempt. This process keeps repeating itself until the tote is served and then he will leave the system on the conveyor. This system can be illustrated as shown in Figure 2.1.

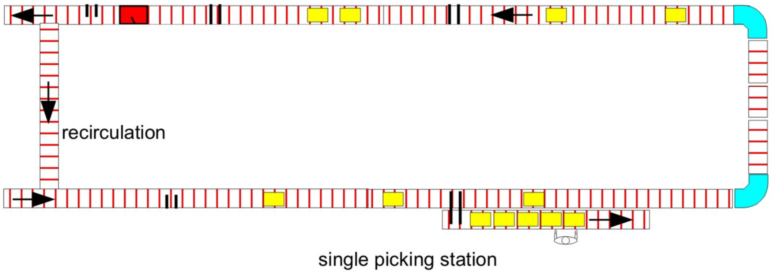


Figure 2.1: A circular routing system with one zone

2.1 Introduction

This system can be modelled as a so-called single server 'retrial queue'. This queue consists of a single server and an 'orbit'. Customers that find the server fully occupied will take place in the orbit to come back after a certain amount of time, called the 'retrial time'. This clearly corresponds with our circular system, where the retrial time is the time it takes for the tote to travel a full cycle. The orbit in a retrial queue can be seen as the conveyor in our real system and the server is the picking station. These kind of queues are extensively discussed in literature. The $M/M/1$ retrial queue in the steady state was studied long ago by Cohen [5] in 1957. He in fact considered the $M/M/c$ retrial queue with impatient customers and no waiting positions. The first results on $M/G/1$ retrial queues are due to Keilson, Cozzolino and Young [14] who used the method of supplementary variable to investigate the joint distribution of the number of customers. In later literature [8] also the single server model with no waiting positions and two types of customers, where one type has priority above the other, is considered.

We can model the system of Figure 2.1 in three ways which all have some advantages and/or disadvantages. The first model is the 'open model'. In this model it is assumed that there is infinite amount of work available at the beginning of the conveyor. The mean inter-arrival time between two totes is given and it does not depend on the workload of the system. In this model performance measures like utilization and throughput are more or less pre-defined by the mean service time and the mean inter-arrival time of the totes.
The second model we are considering is the 'finite source model'. As can be seen from the name a finite amount of work is assumed in the system, this is the maximum number of totes allowed in the system. There are two types of finite source models. The first one considers the amount of work in the system; if the system is highly utilized, the rate at which new totes arrive will be less then when the system has a very low utilization. The second one assumes that the rate of arrival is contant but stops if there is a certain amount of totes in the system. Performance measures like utilization and throughput are not pre-defined as in the open model.
The third and last model is the saturated model, here we assume that a fixed number of totes are either on the conveyor or are in the zone. The arrival of new totes is modelled as replacing a tote that has completed service by a new one. Or in other words we assume that there is constantly a certain amount in the system, which may be true if we are in a very busy period of time. This model has the same advantages as the finite source model and maybe better to work with in the future.

For all models we assume that the service time of the picking station is exponentially distributed with parameter μ and also the retrial time is exponentially distributed, with parameter γ. The retrial time is in reality deterministic, because a conveyor has fixed speed en fixed length. For the open model we assume furthermore that all totes arrive according to a Poisson process with parameter λ. For the finite source model we assume that each tote in the source generates an arrival according to a Poisson process with rate λ, for the infinite server source. For the single server source, the source generates arrivals with an exponential inter-arrival time with parameter λ, as long as there are totes present in the source. If there are no totes in the source, no totes will be generated. Finally for the saturated model we don't have to assume anything more. A very important assumption here is that the operator can always place a tote on the conveyor, so we assume that there is enough place on the conveyor or the time an operator he has to wait is negligible.

In this chapter we will first discuss the open model, where we will give several methods which can calculate the performance measures almost exactly.
The second section will discuss the finite source model, where we will give methods to calculate the performance measures of the regarded system.
In section 2.4 three different saturated models are discussed: the model where the retrial time is exponentially distributed, Erlang-2 distributed and hyper-exponentially distributed. At the end of this section we shall give a proof that for the saturated models it does not matter which distri-

bution the retrial time has, the mean of the retrial time is sufficient. The simulation is discussed and validated in the following section.
In section 2.6 we present the results of the simulation and compare them with the different models. Finally we summarize this chapter and sate a conclusion.

2.2 Open Model

The first model we are discussing is the open model of the system we just described. In Kendall's notation this is the $M/M/1/N$ retrial queue, where the retrial time is exponentially distributed. For stability we require $\rho := \frac{\lambda}{\mu} < 1$. The system looks schematically like Figure 2.2. Here the first

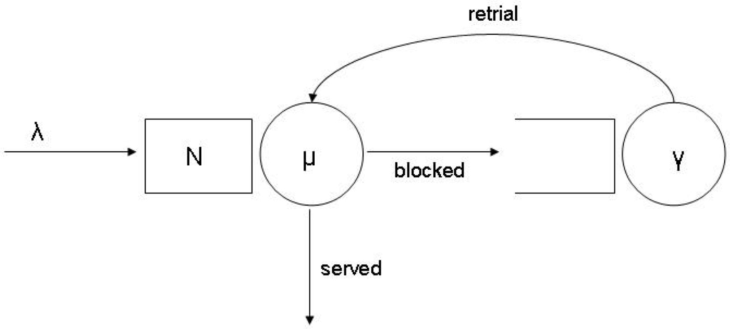


Figure 2.2: An open retrial queueing system

station is a single server station and the second an infinite server station. The first buffer has a limited capacity of N places, while the second buffer has infinitely many places. Because we are considering the open model, we already know the utilization (ρ) of the picking station and the throughput (δ). These performance measures are given by

$$\begin{aligned}\rho &= \frac{\lambda}{\mu}\\ \delta &= \lambda\end{aligned}$$

This model describes the situation when there is always supply of new totes. The reason that we do discuss this model is because we will obtain methods to calculate other performance measures like for example the mean number of re-circulations.

2.2.1 Performance Measures

Literature about some exact and approximation methods are given in the book of Falin and Templeton [8] and also Neuts and Rao [15] have investigated some numerical methods. Falin and Templeton however only look at the case when $N = 1$ and Neuts and Rao look at the multi server queue with no waiting positions. The method used in the latter can be extended to the one we are interested in. In this section we will first describe this model as a Markov process, then we will analyze the case when there is no waiting position, one waiting position and finally we shall describe a method for the case when the buffer has size N.

The system at time t can be described by means of a two-dimensional Markov process $X(t) = (L_c(t), L_z(t))$, where $L_c(t)$ is the number of totes in orbit (the number of totes on the conveyor) and $L_z(t)$ is the number of totes at the zone. For now, we will assume that the conveyor has no limitations. Note that $\{X(t); t \geq 0\}$ is an irreducible Markov process with the lattice semi-strip $S = \mathcal{Z}^+ \times \{0, \ldots, N\}$ as the state space.
Let us define the limiting distribution:

$$P(i,j) = P[(L_c, L_z) = (i,j)] = \lim_{t\to\infty} P[(L_c(t), L_z(t)) = (i,j)], \quad (i,j) \in S. \tag{2.1}$$

No waiting positions

For the system with no waiting positions, so we have a buffer of size $N = 1$, this is the same as described in [8]. We write out the balance equations:

$$\begin{aligned}(\lambda + j\gamma)P(j,0) &= \mu P(j,0),\\ (\lambda + \mu)P(j,1) &= \lambda P(j,0) + (j+1)\gamma P(j+1,0) + \lambda P(j-1,1).\end{aligned}$$

With generating functions we can solve these equations. Therefore define:

$$P_0(z) = \sum_{j=0}^{\infty} z^j P(j,0), \;\; P_1(z) = \sum_{j=0}^{\infty} z^j P(j,1).$$

The balance equations become

$$\begin{aligned}\lambda P_0(z) + \gamma z P_0'(z) &= \mu P_1(z),\\ (\mu + \lambda - \lambda z)P_1(z) &= \lambda P_0(z) + \gamma P_0'(z).\end{aligned}$$

Eliminating $P_1(z)$ we obtain the following differential equation

$$P_0'(z) = \frac{\lambda\rho}{\gamma(1-\rho z)} P_0(z), \tag{2.2}$$

with solution

$$P_0(z) = C\frac{1}{(1-\rho z)^{\frac{\lambda}{\gamma}}}. \tag{2.3}$$

We obtain from the balance equations

$$P_1(z) = C\frac{\rho}{(1-\rho z)^{(\frac{\lambda}{\gamma}+1)}}. \tag{2.4}$$

The constant C is found with the normalization equation:

$$\sum_{j=0}^{\infty}(P(j,0) + P(j,1)) = P_0(1) + P_1(1) = 1, \tag{2.5}$$

from which we obtain $C = (1-\rho)^{(\frac{\lambda}{\gamma}+1)}$.

We easily obtain various performance characteristics like the mean and the variance of the number of totes in the orbit (L_c) or in the system $(L := L_c + L_z)$. We can also determine the probability that the buffer is fully occupied, denoted as the blocking probability (P_b). Using generating functions we obtain

$$\begin{aligned}
\mathbb{E}[L_c] &= \frac{\rho(\lambda+\rho\gamma)}{\gamma(1-\rho)}, \\
Var[L_c] &= \frac{\rho(\lambda+\rho\gamma+\rho^2\gamma-\rho^3\gamma)}{\gamma(1-\rho)^2}, \\
\mathbb{E}[L] &= \frac{\rho(\lambda+\gamma)}{\gamma(1-\rho)}, \\
Var[L] &= \frac{\rho(\lambda+\mu)}{\gamma(1-\rho)^2}, \\
P_b &= P_1(1) = \rho.
\end{aligned}$$

With Little's theorem we can obtain the average time spent by the tote in the orbit $(\mathbb{E}[S_{L_c}])$ and also the average number of circulations (c) before a tote is served, as is presented in [8]:

$$\begin{aligned}
\mathbb{E}[S_{L_c}] &= \mathbb{E}[L_c]/\lambda, \\
c &= \gamma\mathbb{E}[S_{L_c}].
\end{aligned}$$

One waiting position

In available literature [8] there are no explicit formulas for the case we have one waiting position. However, in this book the case with 2 servers and no waiting positions is considered, which has many similarities. This approach however can therefore be extended to our case. For this define:

$$\begin{aligned}
P(j,0) &:= P(L_c = j, L_z = 0), \\
P(j,1) &:= P(L_c = j, L_z = 1), \\
P(j,2) &:= P(L_c = j, L_z = 2).
\end{aligned}$$

These probabilities satisfy the following set of balance equations

$$\begin{aligned}
(\lambda + j\gamma)P(j,0) &= \mu P(j,1), && (2.6)\\
(\lambda+\mu+j\gamma)P(j,1) &= \lambda P(j,0) + (j+1)\gamma P(j+1,0) + \mu P(j,2), && (2.7)\\
(\lambda+\mu)P(j,2) &= \lambda P(j,1) + (j+1)\gamma P(j+1,1) + \lambda P(j-1,2), \quad j \geq 1, && (2.8)
\end{aligned}$$

and of course the normalization condition

$$\sum_{j=0}^{\infty}(P(j,0)+P(j,1)+P(j,2)) = 1.$$

Because we can express $P(j,1)$ in $P(j,0)$ via 2.6. We can also express $P(j,2)$ in $P(j,0)$ and $P(j+1,0)$ via 2.7

$$\mu P(j,2) = [\frac{(\lambda+j\gamma)^2}{\mu} + j\gamma]P(j,1) - (j+1)\gamma P(j+1,0) \qquad (2.9)$$

Substituting this expression in (2.8) we obtain a recursive relation for probabilities $P(j,0)$

$$\begin{aligned} & \frac{\lambda}{\mu}[\frac{(\lambda+j\gamma)^2}{\mu}+j\gamma)]P(j,0)-(j+1)\frac{\gamma}{\mu}[\mu+2\lambda+(j+1)\gamma]P(j+1,0) \\ = \; & \frac{\lambda}{\mu}[\frac{(\lambda+(j-1)\gamma)^2}{\mu}+(j-1)\gamma)]P(j-1,0)-j\frac{\gamma}{\mu}[\mu+2\lambda+j\gamma]P(j,0). \end{aligned}$$

According to [8] this yields that for all j we have

$$\frac{\lambda}{\mu}[\frac{(\lambda+(j-1)\gamma)^2}{\mu}+(j-1)\gamma]P(j-1,0)-j\frac{\gamma}{\mu}[\mu+2\lambda+j\gamma]P(j,0)=0,$$

or equivalently,

$$P(j,0)=\frac{\lambda}{j\gamma}\frac{\frac{(\lambda+(j-1)\gamma)^2}{\mu}+(j-1)\gamma}{\mu+2\lambda+j\gamma}P(j-1,0). \tag{2.10}$$

So we obtain

$$P(j,0)=\frac{\lambda^j}{j!\gamma^j}\prod_{k=0}^{j-1}\frac{\frac{(\lambda+k\gamma)^2}{\mu}+k\gamma}{\mu+2\lambda+\gamma+k\gamma}P(0,0). \tag{2.11}$$

And from the balance equations we can find the probabilities $P(j,1)$ and $P(j,2)$

$$\begin{aligned} P(j,1) &= \frac{(\lambda+j\gamma)}{\mu}\frac{\lambda^j}{j!\gamma^j}\prod_{k=0}^{j-1}\frac{\frac{(\lambda+k\gamma)^2}{\mu}+k\gamma}{\mu+2\lambda+\gamma+k\gamma}P(0,0), \\ P(j,2) &= [(\frac{\lambda+j\gamma}{\mu})^2+\frac{j\gamma}{\mu}-\frac{\lambda}{\mu}\frac{\frac{(\lambda+j\gamma)^2}{\mu}+j\gamma}{\mu+2\lambda+(j+1)\gamma}]P(j,0), \end{aligned}$$

where $P(j,0)$ is given as in (2.11). The initial probability $P(0,0)$ can be obtained with the normalizing condition, by summing over j and let this sum up to 1. We have then obtained all the limiting probabilities from which we can calculate the performance measures in the same manner as done before.

More waiting positions

Now we will consider the system with more than one waiting positions. Then we can see this as a quasi-birth death process. The approach we will describe in this section is already done in [15] for the multi server case with no waiting positions. In this paper a matrix geometric solution is suggested.
The generator $\mathbf{Q}$ of this process has a tri-diagonal structure and is of the form:

$$\mathbf{Q}=\begin{pmatrix} \mathbf{A_{10}} & \mathbf{A_0} & & & \\ \mathbf{A_{21}} & \mathbf{A_{11}} & \mathbf{A_0} & & \\ & \mathbf{A_{22}} & \mathbf{A_{12}} & \mathbf{A_0} & \\ & & & \cdots & \\ & & & & \cdots \end{pmatrix},$$

where the submatrices are squared and have dimension $N+1$

$$\mathbf{A_0} = \begin{pmatrix} \mathbf{0} & & \\ & & \\ & & \lambda \end{pmatrix},$$

$$\mathbf{A_{1i}} = \begin{pmatrix} -i\gamma-\lambda & \lambda & & & \\ \mu & -i\gamma-\mu-\lambda & \lambda & & \\ & \dots & \dots & \dots & \\ & & \mu & -i\gamma-\mu-\lambda & \lambda \\ & & & \mu & -\mu-\lambda \end{pmatrix},$$

$$\mathbf{A_{2i}} = \begin{pmatrix} i\gamma & & \\ & \ddots & \\ & & i\gamma \\ & & \end{pmatrix}.$$

Let $P(i,j)$ denote the steady state probability of state (i,j) and let $\mathbf{p}$ be a row vector with elements $P(i,j)$ arranged in the same order as the Markov chain. The system of balance equations is given by

$$\mathbf{pQ} = 0, \quad \mathbf{pe} = 1, \tag{2.12}$$

where $\mathbf{e}$ is the column vector of all ones. This system could be approximated by restricting the number of totes in the orbit, say at a number M. When M is chosen appropriately the effect on the equilibrium probability vector is minimal.
This approximation modifies the generator $\mathbf{Q}$ to the following, where $\mathbf{A_1} = \mathbf{A_{1M}}$ and $\mathbf{A_2} = \mathbf{A_{2M}}$,

$$\mathbf{Q} = \left(\begin{array}{ccccc|cccc} \mathbf{A_{10}} & \mathbf{A_0} & & & & & & & \\ \mathbf{A_{21}} & \mathbf{A_{11}} & \mathbf{A_0} & & & & & & \\ & \dots & \dots & \dots & & & & & \\ & & & \mathbf{A_{2,M-1}} & \mathbf{A_{1,M-1}} & \mathbf{A_0} & & & \\ \hline & & & & \mathbf{A_2} & \mathbf{A_1} & \mathbf{A_0} & & \\ & & & & & \mathbf{A_2} & \mathbf{A_1} & \mathbf{A_0} & \\ & & & & & & \dots & \dots & \dots \end{array}\right),$$

Now we partition the vector $\mathbf{p}$ as follows $\mathbf{p} = [\mathbf{p_0}, \mathbf{p_1}, \ldots]$ where $\mathbf{p_i}$ is the vector corresponding to the states $(i,j); 0 \le j \le N$. This vector has a matrix geometric form and satisfies

$$\mathbf{p}_{M+i-1} = \mathbf{p}_{M-1}\mathbf{R}^i, \ \ i > 0, \tag{2.13}$$

where the rate matrix R is the unique non negative solution of

$$\mathbf{A_0} + \mathbf{RA_1} + \mathbf{R}^2\mathbf{A_2} = 0.$$

Because of the special structure of $\mathbf{A_0}$ we can solve $\mathbf{R}$ explicitly.

We can write $\mathbf{A_0}$ as

$$\mathbf{A_0} = \omega \cdot \beta$$

where

$$\omega = \begin{pmatrix} 0 \\ \vdots \\ 0 \\ \lambda \end{pmatrix}, \quad \beta = \begin{pmatrix} 0 & \cdots & 0 & 1 \end{pmatrix}, \quad \beta\mathbf{e} = 1.$$

For the computation of the matrix $\mathbf{R}$ we may rewrite (2.2.1) in the form

$$\mathbf{R} = -(\mathbf{A_0} + \mathbf{R}^2\mathbf{A_2})\mathbf{A_1}^{-1};$$

note that $\mathbf{A_1}$ is indeed invertable, since $\mathbf{A_1}$ is a transient generator. The above (fixed point) equation may be solved by successive substitutions, so

$$\mathbf{R_{k+1}} = -(\mathbf{A_0} + \mathbf{R_k}^2\mathbf{A_2})\mathbf{A_1}^{-1}, \;\; k = 0, 1, 2, \ldots \tag{2.14}$$

starting with $\mathbf{R_0} = \mathbf{0}$. It can be shown that, as k tends to infinity, that $\mathbf{R_k} \uparrow \mathbf{R}$. With this recursive scheme for this matrix $\mathbf{R}$ we obtain

$$\mathbf{R_0} = 0, \;\; \mathbf{R_1} = -\mathbf{A_0}\mathbf{A_1}^{-1} = -\omega\beta\mathbf{A_1}^{-1} = \omega \cdot \mathbf{a_1},$$

where $a_1 = -\beta\mathbf{A_1}^{-1}$. Repeating this iteration shows that all $\mathbf{R_k}$'s are of the form

$$\mathbf{R_k} = \omega \cdot \mathbf{a_k}$$

where a_k is a row vector. Since $\mathbf{R_k} \uparrow \mathbf{R}$ as $k \to \infty$, we can conclude that also $\mathbf{R}$ is of the form

$$\mathbf{R} = \omega \cdot \alpha$$

for some row vector α. Hence

$$\mathbf{p}_{M+i-1} = \mathbf{p}_{M-1}\mathbf{R^i} = \eta^{i-1}\mathbf{p}_{M-1}\mathbf{R} = \eta^{i-1}\mathbf{p}_M, \quad i \geq 1,$$

with $\eta = \alpha\omega = \alpha_1\lambda$ and where $\alpha = (\alpha_1, \alpha_2, \ldots)$. So we can substitute this solution in equation (2.2.1) and we get the following expression:

$$\mathbf{p}_M[\mathbf{A_0} + \eta\mathbf{A_1} + \eta^2\mathbf{A_2}] = 0$$

Note that η can be characterized as the unique root in $(0, 1)$ of the determinant equation

$$det(\mathbf{A_0} + \eta\mathbf{A_1} + \eta^2\mathbf{A_2}) = 0, \tag{2.15}$$

which may be computed by straightforward bisection.

The remaining probability vectors $[\mathbf{p}_0, \mathbf{p}_1, \ldots, \mathbf{p}_{M-1}]$ can be determined by looking at the generator. The generator is given by

$$\mathbf{T} = \begin{pmatrix} \mathbf{A_{10}} & \mathbf{A_0} & & & \\ \mathbf{A_{21}} & \mathbf{A_{11}} & \mathbf{A_0} & & \\ & \mathbf{A_{22}} & \mathbf{A_{12}} & \mathbf{A_0} & \\ & & & \cdots & \\ & & & \mathbf{A_{2,M-1}} & \mathbf{A_{1,M-1}} + \mathbf{R}\mathbf{A_2} \end{pmatrix},$$

and the remaining vectors are normalized by

$$\sum_{i=0}^{\infty}\mathbf{p}_i\mathbf{e} = [\sum_{i=0}^{M-2}\mathbf{p}_i + \mathbf{p}_{M-1}(\mathbf{I}-\mathbf{R})^{-1}]\mathbf{e} = 1 \tag{2.16}$$

There are several schemes to obtain these remaining vectors. One of them is the block Gauss-Seidel scheme, which is quite useful for these calculations.

Other system characteristics are now easily obtained, in [15] several quantities are summarized. Here we give the two of most interest, namely the blocking probability (P_b) or in other words the probability that all servers are busy and the mean number of totes at the conveyor ($\mathbb{E}[L_c]$):

$$\begin{aligned} P_b &= \sum_{i=0}^{\infty} p_{i,N}, \\ \mathbb{E}[L_c] &= [\sum_{i=0}^{\infty} i\mathbf{p}_i]\mathbf{e}. \end{aligned}$$

With Little's theorem we can obtain the average time spent by a tote in the orbit $\mathbb{E}[S_{L_c}]$ and also the average number of circulations (c) before a tote is served:

$$\begin{aligned} \mathbb{E}[S_{L_c}] &= \mathbb{E}[L_c]/\lambda, \\ c &= \gamma\mathbb{E}[S_{L_c}]. \end{aligned}$$

2.3 Finite Source

In this finite source network we have to introduce an extra variable, namely the total number of totes in the system. This is denoted as C, where we assume $N < C < \infty$. This variable C could be based on known data, like a proportion of the capacity of the main conveyor and N is again the size of the buffer. We here say a proportion because if we choose the full capacity then deadlocks are certain to occur. Because $C < \infty$ this model is also known as a retrial queue with finite source. If C would be infinite we would have the open model again, when the source is a single server station.

In [8] and also in [9] the finite source model is described shortly for the multi server case without waiting positions. The methods used there can also be applied to our single server model with waiting positions. This section will have the same structure as the previous. So first we will describe this model as a Markov process, then we will analyze the case when there is no waiting position and finally we shall describe a method for the case when there are waiting positions. The system looks schematically as in Figure 2.3. The source can be modelled as an infinite server or as a single server. For our analysis we shall assume an infinite server and in the end we will adjust the analysis to obtain results for the single server.

This system can be modelled as a two-dimensional Markov process $X(t) = (L_z(t), L_c(t))$, where $L_z(t)$ is the number of totes at the service station and $L_c(t)$ is the number of totes at the conveyor. Here $L_z(t) = 0, \ldots, N$ and $L_c(t) = 0, \ldots, C-N$. We have the transition rates between states as given in Figure 2.4. Let us define the joint distribution of the number of totes in the service queue

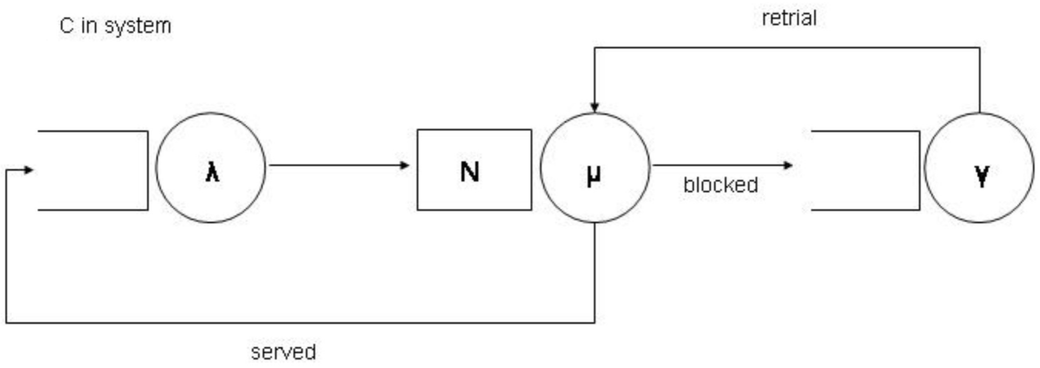


Figure 2.3: A finite source retrial queueing system

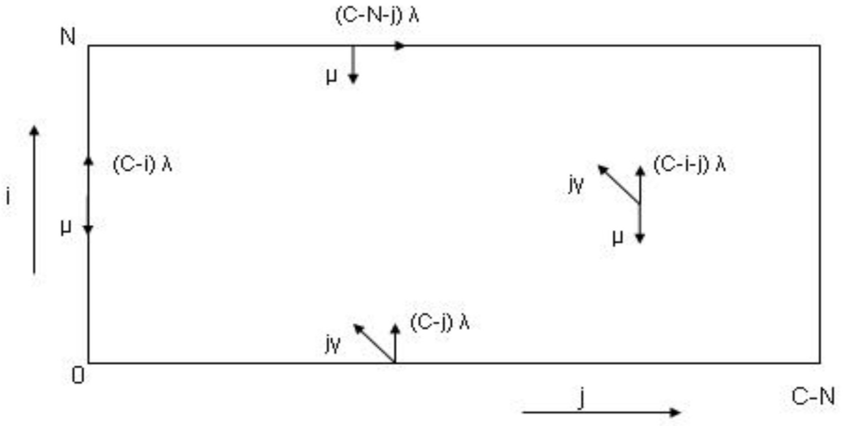


Figure 2.4: The transition rates for the Markov chain of finite source retrial queue

and the number of totes in the orbit in steady state, as follows:

$$P(i,j) = \lim_{t\to\infty} P[L_z(t) = i, L_c(t) = j] \tag{2.17}$$

These probabilities satisfy the following balance equations:

$$\begin{aligned}
((C-j)\lambda + j\gamma)P(0,j) &= \mu P(1,j), & (2.18)\\
((C-i-j)\lambda + j\gamma + \mu)P(i,j) &= P(i-1,j+1)(j+1)\gamma + P(i-1,j)(C-i+1-j)\lambda & (2.19)\\
&+ P(i+1,j)\mu, \quad 1 \le i \le N-1, &\\
((C-N-j)\lambda + \mu)P(N,j) &= P(N-1,j)((C-N+1-j)\lambda) & (2.20)\\
&+ P(N-1,j+1)(j+1)\gamma + P(N,j-1)((C-N-j+1)\lambda. &
\end{aligned}$$

The limiting probabilities that are not in the state space are here defined as 0.

2.3.1 No waiting positions

For the case we have no waiting positions, so the buffer has size $N = 1$, these balance equations simplify to

$$\begin{aligned} ((C-j)\lambda + j\gamma)P(0,j) &= \mu P(1,j), & (2.21)\\ ((C-1-j)\lambda + \mu)P(1,j) &= P(0,j)((C-j)\lambda) + P(0,j+1)(j+1)\gamma & (2.22)\\ &+ P(1,j-1)(C-j)\lambda. \end{aligned}$$

An analytical solution is described in [9] for a more general retrial queue:

Summing up these two equations and rewriting gives:

$$(C-1-j)\lambda P(1,j) - (j+1)\gamma P(0,j+1) = (C-j)\lambda P(1,j-1) - j\gamma P(0,j) \qquad (2.23)$$

this implies that

$$(C-j)\lambda P(1,j-1) - j\gamma P(0,j) = 0. \qquad (2.24)$$

Eliminating $P(1,j-1)$ gives

$$j\gamma\mu P(0,j) = (C-j)\lambda((C-j+1)\lambda + (j-1)\gamma)P(0,j-1)$$

Or in other words

$$\begin{aligned} P(0,j) &= \frac{(C-j)\lambda[(C-j+1)\lambda + (j-1)\gamma]}{j\gamma\mu} P(0,j-1)\\ &= \frac{\rho}{\gamma}\frac{[(C-j+1)\lambda + (j-1)\gamma](C-j)}{j} P(0,j-1)\\ &= \frac{(\frac{\rho}{\gamma})^2}{j(j-1)}(C-j)(C-j+1)[(C-j+1)\lambda + (j-1)\gamma][(C-j+2)\lambda + (j-2)\gamma]P(0,j-2)\\ &= \frac{(\frac{\rho}{\gamma})^j}{j!}\prod_{i=0}^{j-1}(C-j+i)[(C-i)\lambda + i\gamma]P(0,0) \end{aligned}$$

And we have from (2.21) that

$$\begin{aligned} P(1,j) &= (1/\mu)((C-j)\lambda + j\gamma)P(0,j)\\ &= \frac{1}{C\mu}\frac{(\frac{\rho}{\gamma})^j}{j!}\prod_{i=0}^{j}(C-j+i)[(C-i)\lambda + i\gamma]P(0,0) \end{aligned}$$

We get formulas for generating functions $P_0(z)$ and $P_1(z)$ and using the normalization condition $P_0(1) + P_1(1) = 1$ we can find the probability $P(0,0)$.

Several important performance characteristics are given in [8]. Here we will present the most important measures like the utilization of the server and the throughput of the system

$$\begin{aligned} \rho &= 1-\sum_{j=0}^{C-N} P(0,j), \\ \delta &= \mu\rho, \end{aligned}$$

Also we can obtain the mean number of totes in the orbit, or the mean number of repeated totes ($\mathbb{E}[L_c]$), and also the mean number of totes at the queue of the picking station ($\mathbb{E}[L_z]$)

$$\begin{aligned} \mathbb{E}[L_c] &= \sum_{i=0}^{N}\sum_{j=0}^{C-N} jP(i,j), \\ \mathbb{E}[L_z] &= \sum_{i=0}^{N}\sum_{j=0}^{C-N} iP(i,j). \end{aligned}$$

2.3.2 Waiting positions

For the case when there are waiting positions we can use the algorithm given in [9] for the multi server case. First we sum up the local balance equations to obtain

$$(C-N-j)\lambda P(N,j) = (j+1)\gamma \sum_{i=0}^{N-1} P(i,j+1), \;\; 0 \le j \le C-N. \tag{2.25}$$

Before we start the algorithm we have to introduce the variables

$$r_{ij} = \frac{P(i,j)}{P(0,C-N)},$$

so that

$$P(i,j) = \frac{r_{ij}}{\sum_{i=0}^{N}\sum_{j=0}^{C-N} r_{ij}}.$$

Variables r_{ij} satisfy the set of balance equations and also satisfy

$$r_{0,C-N} = 1. \tag{2.26}$$

The variables r_{ij} are calculated by group of size $N+1$. First $r_{0,C-N}, \ldots, r_{N,C-N}$ will be calculated, then $r_{0,C-N-1}, \ldots, r_{N,C-N-1}$ and so on. The algorithm works as stated in Algorithm 1. And finally we obtain

$$P(0,C-N) = \frac{1}{\sum_{i=0}^{N}\sum_{j=0}^{C-N} r_{ij}}.$$

Now we can calculate all the $P(i,j)$ using that $P(i,j) = r_{ij}P(0,C-N)$. The performance characteristics like utilization and mean throughput are given by

$$\begin{aligned} \rho &= 1-\sum_{j=0}^{C-N} P(0,j), \\ \delta &= \mu\rho. \end{aligned}$$

Algorithm 1 Algorithm finite (infinite) source

```
1: Set j=C-N and calculate
     $r_{1,C-N} = \frac{N\lambda + (C-N)\gamma}{\mu},$
     $r_{i,M} = \frac{(N-i+1)\lambda + \mu + (C-N)\gamma}{\mu} r_{i-1,C-N} - \frac{(N-i+2)\lambda}{\mu} r_{i-2,C-N}, \quad i = 2, \dots, N.$
2: Set j=j-1 and calculate
     $r_{Nj} = \frac{(j+1)\gamma}{(C-N-j)\lambda} \left( \sum_{i=0}^{N-1} r_{i,j+1} \right). \qquad (2.27)$
   let us introduce the variables $B_i, D_i, 0 \leq i \leq N-1$, see [10], defined by the equations
     $B_0 = (C-j)\lambda + j\gamma,$
     $B_i = (C-i-j)\lambda + \mu + j\gamma + \frac{(C-i-j+1)\lambda\mu}{B_{i-1}}, \quad 1 \leq i \leq N-1,$
     $D_0 = 0,$
     $D_i = (j+1)\gamma r_{i-1,j+1} + \frac{(C-i+1-j)\lambda D_{i-1}}{B_{i-1}}, \quad 1 \leq i \leq N-1.$
   Then recursively calculate $r_{ij}, 0 \leq i \leq N-1$ with
     $r_{ij} = \frac{D_i + \mu r_{i+1,j}}{B_i}, \quad i = N-1, N-2, \dots 1, 0. \qquad (2.28)$
3: Repeat step 2 while $j \geq 0$.
```

Other performance characteristics can be calculated using the same formulas as given in previous parts. To calculate the limiting probabilities, we have written a program which uses this algorithm to solve these probabilities. The outcome of the program is checked with the exact answers when there are no waiting positions at the picking station, which are given in the previous part. In all the three cases checked, the relative error to the exact answers was 0%. As should be, the program gives exact answers and relative fast.

2.3.3 Single server

To obtain results for the single server case we can adjust the algorithm by simply replacing all $i\lambda$ by λ. The analysis still holds and for completeness we present the algorithm for the single server case in Algorithm 2. The performance measures are calculated in the same manner as done in the case when the source is an infinite server. This algorithm is also checked for the case when there are no waiting positions and in all three cases the relative error was again 0%, so the algorithm gives exact answers.

Algorithm 2 Algorithm finite (single) source

```
1: Set j=C-N and calculate
     $r_{1,C-N} = \frac{\lambda + (C-N)\gamma}{\mu},$
     $r_{i,M} = \frac{\lambda + \mu + (C-N)\gamma}{\mu} r_{i-1,C-N} - \frac{\lambda}{\mu} r_{i-2,C-N}, \quad i = 2, \ldots, N.$
2: Set j=j-1 and calculate
     $r_{Nj} = \frac{(j+1)\gamma}{\lambda} \left( \sum_{i=0}^{N-1} r_{i,j+1} \right).$   (2.29)
   let us introduce the variables $B_i, D_i, 0 \le i \le N-1$ defined by the equations
     $B_0 = \lambda + j\gamma,$
     $B_i = \lambda + \mu + j\gamma + \frac{\lambda\mu}{B_{i-1}}, \quad 1 \le i \le N-1,$
     $D_0 = 0,$
     $D_i = (j+1)\gamma r_{i-1,j+1} + \frac{\lambda D_{i-1}}{B_{i-1}}, \quad 1 \le i \le N-1.$
   Then recursively calculate $r_{ij}$, $0 \le i \le N-1$ with
     $r_{ij} = \frac{D_i + \mu r_{i+1,j}}{B_i}, \quad i = N-1, N-2, \ldots 1, 0.$   (2.30)
3: Repeat step 2 while $j \ge 0$.
```

2.4 Saturated Models

In this section we will discuss saturated network models. In these models we have a maximum number of C totes in the system and we will discuss three different saturated models. We assume for all systems that the distributions used are exponential and to avoid triviality $C \ge N$.

The first one is the saturated model with two stations, one station with a buffer of size N and parameter μ and the other is an infinite sever station with parameter γ. The second model discusses the saturated model with three stations, here the second station is replaced by two infinite server stations in serial with parameter 2γ. The third and last model is the same as the first model only the second station is replaced by two infinite server stations in parallel with parameter γ_2 and γ_3 respectively. These two stations are visited from the first station with probability q, respectively $1-q$. In general when all waiting positions at the first station are occupied and a new tote arrives at the first station it immediately transfers to the second or third station. If a tote has been served we 'replace' this one with a new tote. So the model describes the real system reasonably well.

The first model describes the operator with the first station and the conveyor with the second station. The other models do the same, only they model the traveltime on the conveyor as Erlang-2 distributed or hyper exponential distributed, respectively. In the end of this section we will see that it does not matter which distribution is used and only the mean of the retrial time is important. This saturated jump-over model should give exact results, as we will see in section 2.6.

2.4.1 Exponential traveltime

As said earlier we will discuss the model with a finite station with parameter μ and N waiting positions and an infinite queue with parameter γ. This model can schematically be presented as Figure 2.5. Recall that the total number of totes in the system is C. We can describe this model as a Markov process. Therefore define, p_i as the limiting probability that there are i totes at the first station. The flow diagram of this Markov process can presented as follows

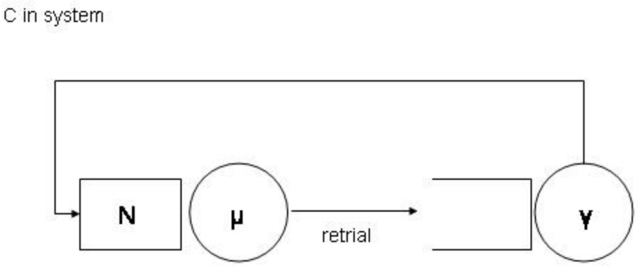


Figure 2.5: A saturated retrial queueing system

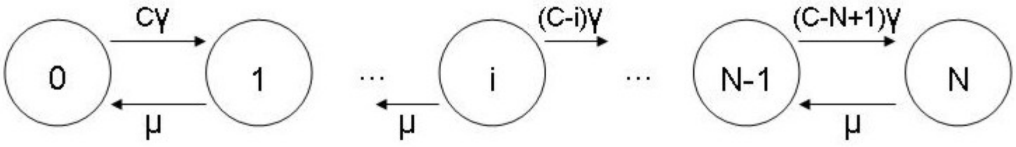


Figure 2.6: The transition rates for the Markov chain of the saturated retrial queue

The balance equations for this process are given by

$$(C-i)\gamma p_i = \mu p_{i+1}, \;\; i = 0, \dots N-1 \tag{2.31}$$

and equivalently

$$\frac{(C-i)\gamma}{\mu} p_i = p_{i+1}, \;\; i = 0, \dots N-1. \tag{2.32}$$

We have the following expression for p_k

$$p_k = (\frac{\gamma}{\mu})^k \prod_{i=1}^{k} (C+1-i) p_0 = k!(\frac{\gamma}{\mu})^k \binom{C}{k} p_0, \;\; \text{for} \;\; k > 0. \tag{2.33}$$

With the normalizing equation

$$\sum_{k=0}^{N} p_k = [\sum_{k=0}^{N} k!(\frac{\gamma}{\mu})^k \binom{C}{k}]p_0 = 1$$

we obtain an expression for p_0

$$p_0 = [\sum_{k=0}^{N} k!(\frac{\gamma}{\mu})^k \binom{C}{k}]^{-1}. \tag{2.34}$$

The other probabilities p_k are given by (2.33).

The most important performance measures; utilization and throughput are given by

$$\begin{aligned} \rho &= 1 - p_0, & (2.35)\\ \delta &= \rho\mu. & (2.36) \end{aligned}$$

We can also calculate the mean number of totes at the zone ($\mathbb{E}[L_z]$) and on the conveyor ($\mathbb{E}[L_c]$)

$$\begin{aligned} \mathbb{E}[L_z] &= \sum_{k=0}^{N} kp_k, \\ \mathbb{E}[L_c] &= C - \mathbb{E}[L_z]. \end{aligned}$$

We can calculate the utilization as a function of the several variables. First (in Figure 2.7) we will variate the total number of totes that are allowed in the system (C), where we have chosen $\gamma = 1/100, \mu = 1/30$ and $N = 4$. We can see that the utilization converges rapidly to 1 as C increases. This is what we would expect, because when C is large a new tote will arrive very fast to the picking station.

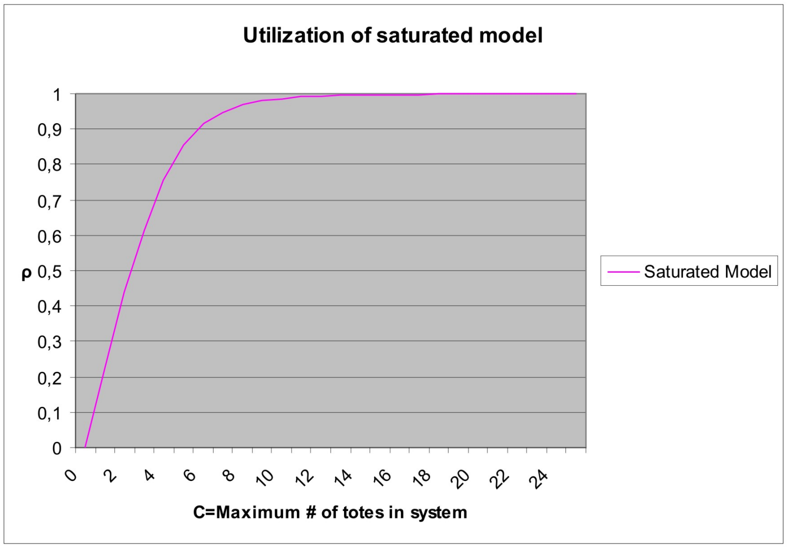


Figure 2.7: The utilization as a function of C, where $\gamma = 1/100, \mu = 1/30$ and $N = 4$

We also calculate the utilization as a function of the buffer size. In Figure 2.8 we have chosen γ and μ as before and $C = 10$. Here we can see again that the utilization increases rapidly to 1.

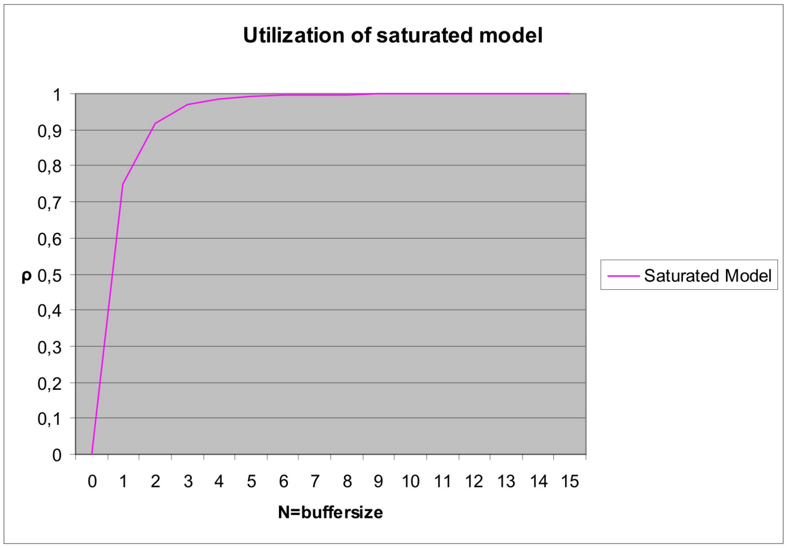


Figure 2.8: The utilization as a function of N, where $\gamma = 1/100$, $\mu = 1/30$ and $C = 10$

We can also see that the utilization is relative high for low values of N, so the buffer size has less influence on the utilization than the maximum number of totes allowed in the system.

The ratio between the service time and the retrial time $r = \gamma/\mu$ is also an interesting variable that influences the performance of the system. In Figure 2.9 we can see the results of these settings on the system performance. We can see that when $r > 1$, the utilization is close to 1. This is what we would expect because when γ is larger than μ, the mean retrial time is smaller than the mean service time. This implies that totes are coming back faster then totes are served, so the operator will be busy all the time.

2.4.2 Erlang-2 traveltime

The second model we are discussing is the model with three serial stations, the first one is the finite capacity queue we have seen earlier and the other two are infinite server stations with parameter $\gamma' = 2\gamma$. This system can schematically be illustrated as Figure 2.10. We can describe this model as a two-dimensional Markov process. Therefore, define $P(n_1, n_2)$ as the limiting probability that there are n_1 totes at the first station and n_2 at the second station. The flow diagram of this Markov process as well as the balance equations are given in appendix A.1.

These balance equations have a nice solution, because this is a special case of the queueing system described in [6]. In this article, queueing networks with finite capacity are considered and if jobs encounter a fully occupied node, they jump over this node as they were being served at an infinite speed. All the service times are assumed to be exponential and the jobs are routed from node i to j with probability p_{ij}. Van Dijk [6] gives a proof that the steady state distribution is

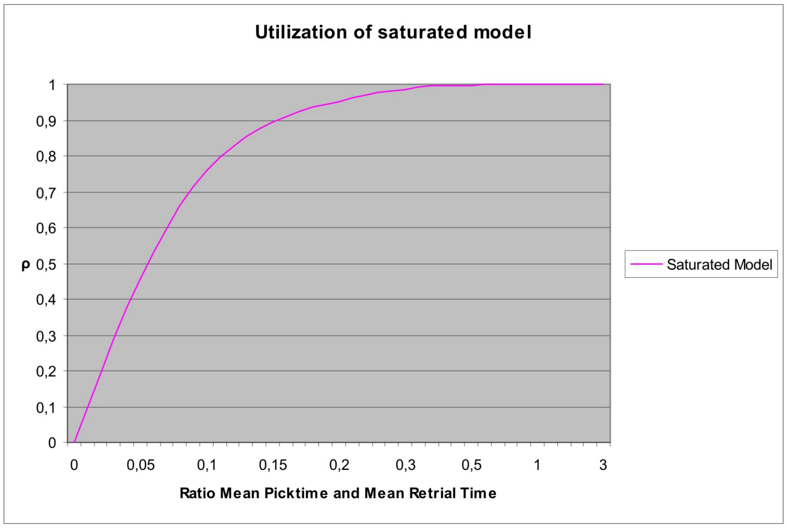


Figure 2.9: The utilization as a function of $p = \gamma/\mu$, where $C = 10$ and $N = 4$

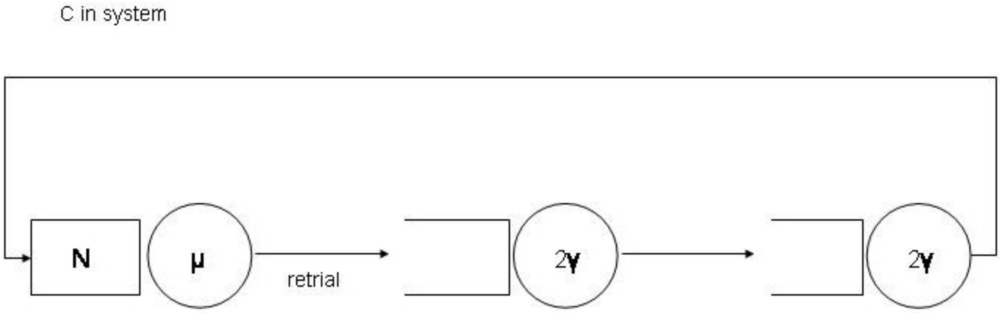


Figure 2.10: A saturated, with Erlang-2 retrial times, retrial queueing system

given by a product form. This can be verified by substituting the following product form into the balance equations:

$$P(n_1, n_2) := P(n_1, n_2, n_3) \quad = \quad d(\frac{\lambda}{\mu})^{n_1} \frac{1}{n_2!}(\frac{\lambda}{\gamma'})^{n_2} \frac{1}{n_3!}(\frac{\lambda}{\gamma'})^{n_3}, \quad \text{for } n_1 + n_2 + n_3 = C, \quad (2.37)$$

where d is the normalizing constant. In appendix A.1 the first balance equation is written out as an illustration, the other equations follow in the same manner by substituting and straightforward algebra.

The limiting probabilities are known explicitly, so we can determine the performance measures in terms of these probabilities. The main performance measures are given by

$$\begin{aligned} \rho &= 1 - \sum_{n_2=0}^{C} P(0, n_2, C - n_2), \\ \delta &= \mu\rho. \end{aligned}$$

Also other performance measures can be determined, like the mean number of totes at the picking station and on the conveyor. These follow directly from the steady state distribution.

2.4.3 Hyper-exponential traveltime

The third and final model we are discussing is the model with three stations, the first one is the finite capacity queue we have seen earlier and the other two are two parallel infinite server stations with parameter γ_2 and γ_3 respectively. The probability to go to the first infinite station is q and for the other queue $1 - q$. This system can schematically be illustrated as Figure 2.11. We can describe this model as a two-dimensional Markov process. Therefore define the limiting probabilities $P(n_1, n_2)$ as the limiting probability that there are n_1 totes at the first station and n_2 at the first infinite station. Notice that when $q = 0$ or $q = 1$, this system is the same as the exponential model and this system can be modelled as a one dimensional Markov process, as is done in previous sections. The flow diagram of this Markov process as well as the balance equations are given in appendix A.2.

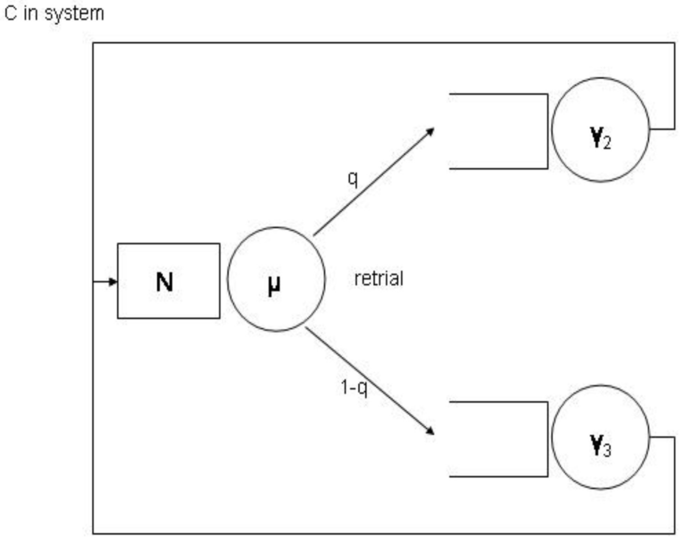


Figure 2.11: A saturated, with hyper-exponential retrial times, retrial queueing system

Again the state space has the same structure as we have seen in the previous section only the balance equations are different. The approach of van Dijk [6] can be applied to this system as well. We will verify the product form by substituting the following solution and check for the balance equations:

$$P(n_1, n_2) := P(n_1, n_2, n_3) = d(\frac{\lambda}{\mu})^{n_1} \frac{1}{n_2!} (\frac{q\lambda}{\gamma_2})^{n_2} \frac{1}{n_3!} (\frac{(1-q)\lambda}{\gamma_3})^{n_3}, \text{ for } n_1 + n_2 + n_3 = C, \quad (2.38)$$

Substituting this solution in the first balance equation is done in appendix A.2. The other balance equations also have this product form solution which can be proven with straightforward algebra. The normalizing constant is given by summing over all possible states, which are in the state space.

The limiting probabilities are known explicitly, so we can determine the performance measures as is done in the previous section.

2.4.4 Insensitivity of distribution

We have discussed three saturated models and we have found expressions for the limiting probabilities. In this section we will show that the distribution of the retrial time does not influence the system performance; the mean of the retrial time is enough to work with. We will show this by calculating the limiting probabilities that there are k totes at the zone and the rest of the $C-k$ totes are on the conveyor. We will show that these are the same for the Erlang-2 model and the hyper-exponential model, which are for their part the same as the limiting probabilities for the exponential model. Here we will take the mean of the whole retrial part the same, so in the Erlang-2 case we have as parameter 2γ and for the hyper exponential case we have $\frac{1}{\gamma} = q\frac{1}{\gamma_2} + (1+q)\frac{1}{\gamma_3}$. We state the following theorem.

Theorem 1 (Insensitivity of distribution). *The system performance is insensitive of the distribution of the retrial time.*

Proof. As sketched just before we will look at the limiting probabilities that there are k totes at the zone for all the three models.

Erlang model

$$\begin{aligned}
P(n_1 = k, n_2 + n_3 = C - k) &= d(\frac{\lambda}{\mu})^k \sum_{i=0}^{C-k} \frac{1}{i!} (\frac{\lambda}{2\gamma})^i \frac{1}{(C-k-i)!} (\frac{\lambda}{2\gamma})^{C-k-i} \\
&= d(\frac{\lambda}{\mu})^k (\frac{\lambda}{2\gamma})^{C-k} \sum_{i=0}^{C-k} \frac{1}{i!} \frac{1}{(C-k-i)!} \\
&= d(\frac{\lambda}{\mu})^k (\frac{\lambda}{2\gamma})^{C-k} \sum_{i=0}^{C-k} \binom{C-k}{i} \frac{1}{(C-k)!} \\
&= d(\frac{\lambda}{\mu})^k (\frac{\lambda}{2\gamma})^{C-k} \frac{2^{C-k}}{(C-k)!} \\
&= d(\frac{\lambda}{\mu})^k (\frac{\lambda}{\gamma})^{C-k} \frac{1}{(C-k)!}.
\end{aligned}$$

Hyper exponential model

$$\begin{aligned}
P(n_1 = k, n_2 + n_3 = C - k) &= d(\frac{\lambda}{\mu})^k \sum_{i=0}^{C-k} \frac{1}{i!}(\frac{q\lambda}{\gamma_2})^i \frac{1}{(C-k-i)!}(\frac{(1-q)\lambda}{\gamma_3})^{C-k-i} \\
&= d(\frac{\lambda}{\mu})^k \lambda^{C-k} \sum_{i=0}^{C-k} \binom{C-k}{i} \frac{1}{(C-k)!}(\frac{q}{\gamma_2})^i(\frac{1-q}{\gamma_3})^{C-k-i} \\
&= d(\frac{\lambda}{\mu})^k \lambda^{C-k} \frac{1}{(C-k)!}(\frac{q}{\gamma_2} + \frac{1-q}{\gamma_3})^{C-k} \\
&= d(\frac{\lambda}{\mu})^k \lambda^{C-k} \frac{1}{(C-k)!}(\frac{1}{\gamma})^{C-k} \\
&= d(\frac{\lambda}{\mu})^k (\frac{\lambda}{\gamma})^{C-k} \frac{1}{(C-k)!}.
\end{aligned}$$

We have already shown that these limiting probabilities are the same for these two models. In fact these probabilities have the same solution as for the exponential model. Therefore we only have to calculate the normalizing constant, which is given by summing over all possible states. So

$$\begin{aligned}
d^{-1} &= \sum_{k=0}^{N} (\frac{\lambda}{\mu})^k (\frac{\lambda}{\gamma})^{C-k} \frac{1}{(C-k)!} \\
&= (\frac{\lambda}{\gamma})^C \sum_{k=0}^{N} (\frac{\gamma}{\mu})^k \frac{1}{(C-k)!}
\end{aligned}$$

So we have

$$\begin{aligned}
P(n_1 = k, n_2 + n_3 = C - k) &= d(\frac{\lambda}{\mu})^k (\frac{\lambda}{\gamma})^{C-k} \frac{1}{(C-k)!} \\
&= \frac{(\frac{\lambda}{\mu})^k (\frac{\lambda}{\gamma})^{C-k} \frac{1}{(C-k)!}}{(\frac{\lambda}{\gamma})^C \sum_{k=0}^{N} (\frac{\gamma}{\mu})^k \frac{1}{(C-k)!}} \\
&= \frac{(\frac{\gamma}{\mu})^k \frac{1}{(C-k)!}}{\sum_{i=0}^{N} (\frac{\gamma}{\mu})^i \frac{1}{(C-i)!}} \\
&= \frac{k!(\frac{\gamma}{\mu})^k \binom{C}{k}}{\sum_{i=0}^{N} i!(\frac{\gamma}{\mu})^i \binom{C}{i}},
\end{aligned}$$

which is the same expression for p_k in the case of the exponential saturated model.□

We just proved the fact that it doesn't matter which distribution the retrial time has, it holds for the exponential, Erlang-2 and hyper-exponential distribution. With last two distributions every general distribution can be fitted, using the moments. This means that with a given buffer, maximum number of totes in the system and the retrial- and picking time we can calculate the limiting distribution, from which we obtain performance measures.

2.4.5 Iterative Method

For certain closed models van Dijk [6] gives an explicit formula for the steady state distribution. In our case we could represent the real life situation as closed model with a source, a conveyer and a service station. The source can be modelled as a single server or as an infinite server, according to the most appropriate situation. The conveyor piece is represented as an infinite server with rate γ and the service station of course as a single server with rate μ. This model also includes jump-over blocking; if a tote finds the buffer fully occupied it jumps over that station, or it is served at that station with infinite speed. In reality if a tote is blocked the tote travels to the conveyor for another attempt, if the tote is served it will leave the system and enters the source. Our model however approximates the fraction of the totes that are (not) blocked. This means that served or not served; with probability p a tote will make a retrial and with probability $1-p$ the tote will enter the source. This is where our model and reality differ, but on average this should give same performance. Schematically this model looks like Figure 2.12. For this system we know the limiting probabilities as stated in [6]:

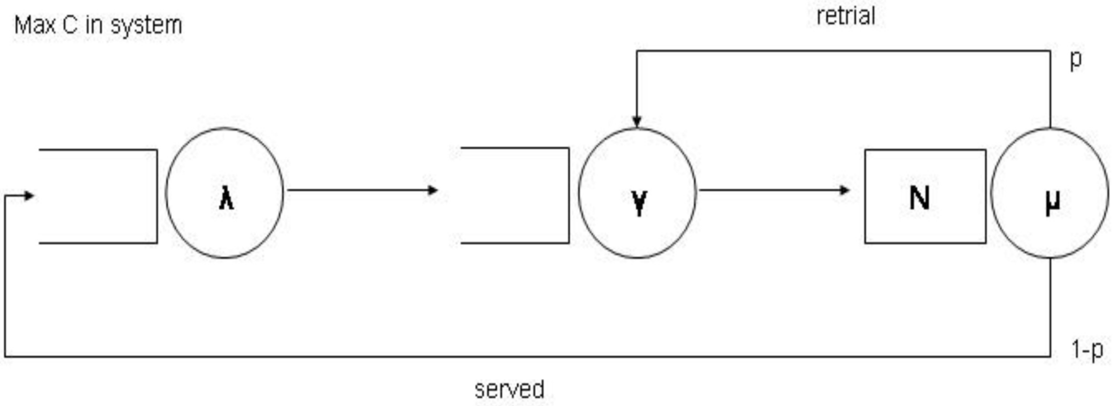


Figure 2.12: A closed retrial queueing system, with a finite source

$$P(n_1,n_2,n_3) = c(\frac{1-p}{\lambda})^{n_1}\frac{1}{n_2!}(\frac{1}{\gamma})^{n_2}(\frac{1}{\mu})^{n_3}, \text{ for single server source,} \quad (2.39)$$

$$P(n_1,n_2,n_3) = c\frac{1}{n_1!}(\frac{1-p}{\lambda})^{n_1}\frac{1}{n_2!}(\frac{1}{\gamma})^{n_2}(\frac{1}{\mu})^{n_3}, \text{ for infinite server source,} \quad (2.40)$$

where c is the normalizing constant and n_1, n_2, n_3 denotes the number of totes at the source, the conveyor and the zone respectively. Furthermore we demand $n_1+n_2+n_3 = C$ and $n_3 \leq N$. Just like the saturated models we can prove that it does not matter which distribution the retrial time has, the system performance will be independent of the choice of this distribution.

The probability that a tote is blocked at the zone is unknown but we can determine this iteratively. Let us first assume that no blocking occurs, then $p = 0$ and we have the explicit formula for the limiting probabilities. With this formula we can determine the blocking probability; this is the probability when $n_3 = N$, but here we have to be careful because of the arrival theorem. This theorem states that an arriving tote at the zone sees the system in equilibrium but with one tote (himself) less, this shall be proven in Theorem 2. This new blocking probability can be used and we have a new formula for the limiting probabilities and we can determine the blocking probability again. This can be repeated until this blocking probability does not change anymore.

Theorem 2 (Arrival Theorem). *An arriving tote at the zone sees the system in equilibrium, but with one tote less.*

Proof. We shall proof this for the single server case, the proof for the infinite server case works in a similar manner. To prove that this theorem holds we need to look at the quotient of the arrival rate into the state that there are n_1, n_2, n_3 respectively at the different stations (with $n_1 + n_2 + n_3 = C - 1$) with the total arrival rate into that state:

$$\frac{(n_2+1)\gamma P_C(n_1, n_2+1, n_3)}{\sum_{n_1}\sum_{n_2}\sum_{n_3}(n_2+1)\gamma P_C(n_1, n_2+1, n_3)} \tag{2.41}$$

where the subscript C represents that there are exactly C totes allowed in the system. We now substitute the expression we have for the limiting probabilities and simplify immediately to obtain:

$$\begin{aligned}\frac{d(\frac{1-p}{\lambda})^{n_1}\frac{1}{n_2!}(\frac{1}{\gamma})^{n_2}(\frac{1}{\mu})^{n_3}}{\sum_{n_1}\sum_{n_2}\sum_{n_3} d(\frac{1-p}{\lambda})^{n_1}\frac{1}{n_2!}(\frac{1}{\gamma})^{n_2}(\frac{1}{\mu})^{n_3}} &= c'(\frac{1-p}{\lambda})^{n_1}\frac{1}{n_2!}(\frac{1}{\gamma})^{n_2}(\frac{1}{\mu})^{n_3} \\ &= P_{C-1}(n_1, n_2, n_3),\end{aligned}$$

because $n_1 + n_2 + n_3 = C - 1$ by assumption and so c' must be the normalizing constant for the limiting probabilities when there are at most $C - 1$ totes allowed in the system. This ends the proof and gives us indeed that an arriving tote sees the system in equilibrium, but with one tote less in this system. □

The algorithm eventually works as follows:

Algorithm 3 Iterative Method

```
1. Set $i = 0$ and $p_0 = 0$.
2. Set $i = i + 1$ and calculate new blocking probability:
       $p_i = \sum_{n_1=0}^{C-1-N} P(n_1, C-1-N-n_1, N),$
   where $P(n_1, n_2, n_3)$ is given by (2.39) for the single server source and by (2.40) for the infinite server source, with $p = p_{i-1}$.
3. Repeat step 2 until $|p_i - p_{i-1}| < \epsilon$.
```

The utilization of the operator is eventually given by

$$\rho = 1 - \sum_{n_1=0}^{C} P(n_1, C-n_1, 0), \tag{2.42}$$

with p the last calculated p_i of Algorithm 3. In Mathematica 5.2 this algorithm is easily implemented and for each case the utilization can be determined. We can prove that Algorithm 3 will converge for the single server source with induction, this proof is stated in Theorem 3. The proof for the infinite server source works in the same manner.

Theorem 3 (Convergence). *Algorithm 3 converges.*

Proof. We will show that the blocking probabilities p_i form a monotonically increasing sequence. This will be done with mathematical induction, the induction statement will be

$$p_i \leq p_{i+1}, \quad i \geq 0.$$

Basis: If $i = 0$ then we clearly have

$$p_1 := d \sum_{n_2=0}^{C-1-N} (\frac{1}{\lambda})^{C-1-N-n_2} \frac{1}{n_2!} (\frac{1}{\gamma})^{n_2} (\frac{1}{\mu})^N \geq 0 =: p_0$$

Step: Let us assume $p_n \leq p_{n+1}$. Then we know

$$\begin{aligned} & p_{n+2} - p_{n+1} \\ := \quad & \frac{\sum_{n_2=0}^{C-1-N} (\frac{1-p_{n+1}}{\lambda})^{C-1-N-n_2} \frac{1}{n_2!} (\frac{1}{\gamma})^{n_2} (\frac{1}{\mu})^N}{D(p_{n+1})} - \frac{\sum_{n_2=0}^{C-1-N} (\frac{1-p_n}{\lambda})^{C-1-N-n_2} \frac{1}{n_2!} (\frac{1}{\gamma})^{n_2} (\frac{1}{\mu})^N}{D(p_n)} \\ = \quad & \frac{D(p_n) \sum_{n_2=0}^{C-1-N} (\frac{1-p_{n+1}}{\lambda})^{C-1-N-n_2} \frac{1}{n_2!} (\frac{1}{\gamma})^{n_2} (\frac{1}{\mu})^N - D(p_{n+1}) \sum_{n_2=0}^{C-1-N} (\frac{1-p_n}{\lambda})^{C-1-N-n_2} \frac{1}{n_2!} (\frac{1}{\gamma})^{n_2} (\frac{1}{\mu})^N}{D(p_n) D(p_{n+1})} \end{aligned}$$

here $D(p_{n+1})$ and $D(p_n)$ are summations over the state space which depend on p_{n+1}, p_n respectively. To prove the statement we now only need to show that $p_{n+2} - p_{n+1} \geq 0$, or equivalently:

$$\begin{aligned} C(p_n) \sum_{n_2=0}^{C-1-N} (\frac{1-p_{n+1}}{\lambda})^{C-1-N-n_2} \frac{1}{n_2!} (\frac{1}{\gamma})^{n_2} (\frac{1}{\mu})^N \quad & \geq \quad C(p_{n+1}) \sum_{n_2=0}^{C-1-N} (\frac{1-p_n}{\lambda})^{C-1-N-n_2} \frac{1}{n_2!} (\frac{1}{\gamma})^{n_2} (\frac{1}{\mu})^N \\ & \geq \quad C(p_{n+1}) \sum_{n_2=0}^{C-1-N} (\frac{1-p_{n+1}}{\lambda})^{C-1-N-n_2} \frac{1}{n_2!} (\frac{1}{\gamma})^{n_2} (\frac{1}{\mu})^N \\ & \Rightarrow \quad C(p_n) \geq C(p_{n+1}) \end{aligned}$$

which is true because of the induction hypotheses; $p_n \leq p_{n+1}$ or $1 - p_n \geq 1 - p_{n+1}$ and hence we know $C(p_n) \geq C(p_{n+1})$. This ends the proof and gives us that the blocking probabilities form a monotonically increasing sequence, so the algorithm converges, because the probability is limited by 0 and 1. □

For this relative simple model with one zone this iterative method can also be avoided. The blocking probability can be determined exactly by solving the polynomial equation:

$$\begin{aligned} p &= \sum_{n_1=0}^{C-1-N} P(n_1, C-1-N-n_1, N) \\ &= c \sum_{n_1=0}^{C-1-N} (\frac{1-p}{\lambda})^{n_1} \frac{1}{(C-1-N-n_1)!} (\frac{1}{\gamma})^{C-1-N-n_1} (\frac{1}{\mu})^{N}, \\ &\quad \text{for single server source and} \\ &= c \sum_{n_1=0}^{C-1-N} \frac{1}{n_1!} (\frac{1-p}{\lambda})^{n_1} \frac{1}{(C-1-N-n_1)!} (\frac{1}{\gamma})^{C-1-N-n_1} (\frac{1}{\mu})^{N}, \\ &\quad \text{for infinite server source.} \end{aligned}$$

We can use Mathematica 5.2 to do this and this generates the same answers as the iterative method. It does not matter which we will use, but we prefer the iterative method because we think this will work faster for more complex models.

It is also interesting to say something about calculation time and the order of the algorithm. In each iteration some calculations must be done, which depend on the maximum number of totes C and the buffersize N. These two parameters determine the size of the state space. It is easy to see that the size of the state space and also order of this algorithm is given by

$$\text{Order algorithm} = \mathcal{O}(CN), \tag{2.43}$$

because we have the state space of $n_2 + n_3 \leq C$ and $n_3 \leq N$. The calculation time is small for this algorithm, but for more complex systems this will get much longer if we increase C. Then also the order will be different, because the state space will be bigger.

2.5 Simulation

To determine whether or not the models are applicable to the real life situation at Vanderlande we have to verify the accuracy by simulation. Within the Simulation Group of Vanderlande Industries the simulation program AutoMod is commonly used. This package has a good reputation in 3D graphical simulations of material handling systems and has been used many times for the simulation of material handling systems. AutoMod utilizes a 3 dimensional color graphics environment for developing material handling system models, which can be viewed in real time execution. The package provides extensive tools and facilities for accurately modelling mechanical systems, activity generation, processing and routing logic, as well as very detailed control algorithms when necessary. This is why we also will use this program to verify the different models.

In this program we will model the circular conveyor by a conveyor piece that is a straight line. At the end of this line is the zone with his buffer. New totes arrive with exponential inter-arrival times and only if the total number of totes in the system does not exceed the maximum number of C. If totes at the end of the conveyor find the buffer full they immediately are returned to the beginning of the conveyor. The simulation does not take a gap between totes into account. The program can take these gaps into account if wanted. A view of the model in AutoMod when running is given in Figure 2.13. The simulation both can simulate the source as a single- or infinite server, furthermore traveltimes are deterministic as is the case in reality. The separate simulation runs last 100 hours each and the total number of runs is 10 with a warm up period of 10 hours. After this warm up period approximately 1000 totes have been served. Each run the different performance measures are being calculated and saved to a file.

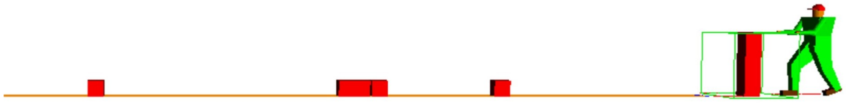


Figure 2.13: View of the AutoMod model when running.

2.5.1 Validation

Although AutoMod is a program which is used extensively within Vanderlande Industries, we will do a validation with special cases of this system. These special cases are the two well-known queueing models: the $M/M/1$-queue and the $M/M/1/1$-queue. For the first case we take N and C sufficiently large so no blocking will occur at the queue and at the beginning of conveyor. In the latter case $N = 1$ is chosen and C is chosen large enough so there will be enough supply of new totes. In the latter system the totes will not circulate when they find the buffer fully occupied, but they will be lost. We will represent the outcomes of the calculation, simulation and the relative errors for both models.

$M/M/1$-queue

We have calculated the utilization and throughput per hour for several values of λ, where we have chosen $\mu = \frac{1}{30}$. The utilization and throughput per hour are given according to the formulas from [2]:

$$\begin{aligned} \rho &= \frac{\lambda}{\mu}, \\ \delta &= 3600 \cdot \lambda. \end{aligned}$$

The outcomes of this simulation are for the utilization of the operator given in Table 2.1. Here, and throughout this thesis, the absolute relative errors are denoted as $\Delta\%$.

The width of the 95% confidence interval of this simulation is approximately $0,04$.

The same can be done for the results of the throughput per hour, in this case the results look like Table 2.2.

$M/M/1/1$-queue

Again we do the same calculations but now for the $M/M/1/1$-queue, the results are shown here. For the utilization we have the results presented in Table 2.3. The width of the 95% confidence

$1/\lambda$	Simulation	Theory	Δ%
32	0,927 (±0,016)	0,938	1,1
34	0,882 (±0,018)	0,882	0,0
36	0,830 (±0,019)	0,833	0,4
38	0,792 (±0,021)	0,790	0,3
40	0,768 (±0,023)	0,750	2,4

Table 2.1: Simulation results for utilization compared with results from queueing theory

$1/\lambda$	Simulation	Theory	Δ%
32	112,4 (±2,0)	112,5	0,1
34	107,4 (±2,0)	106,0	1,4
36	100,7 (±2,0)	100,0	0,7
38	94,9 (±2,0)	94,7	0,2
40	91,0 (±1,9)	90,0	1,1

Table 2.2: $M/M/1$-queue simulation results for throughput per hour compared with results from queueing theory

$1/\lambda$	Simulation	Theory	Δ%
32	0,486 (±0,009)	0,484	0,4
34	0,466 (±0,008)	0,469	0,6
36	0,455 (±0,010)	0,455	0,1
38	0,437 (±0,008)	0,441	1,0
40	0,424 (±0,009)	0,429	1,0

Table 2.3: $M/M/1/1$-queue simulation results for utilization compared with results from queueing theory

interval of this simulation is approximately $0,02$. The same can be done for the results of the throughput per hour, in this case the results are given in Table 2.4. Here the width of the confidence interval is approximately 2.

$1/\lambda$	Simulation	Theory	Δ%
32	58,0 (±1,1)	58,1	0,2
34	57,6 (±1,2)	56,3	2,4
36	55,8 (±1,0)	54,6	2,2
38	53,0 (±1,1)	52,9	0,2
40	51,4 (±1,2)	51,4	0,0

Table 2.4: $M/M/1/1$-queue simulation results for throughput per hour compared with results from queueing theory

Special case: $N = C$

For the special case when $N = C$, no blocking can occur at the zone. In this case the finite source model simplifies to a simple closed network with 2 single servers (the source and the zone) and one infinite server (the conveyor). Schematically this can be represented as Figure 2.14.

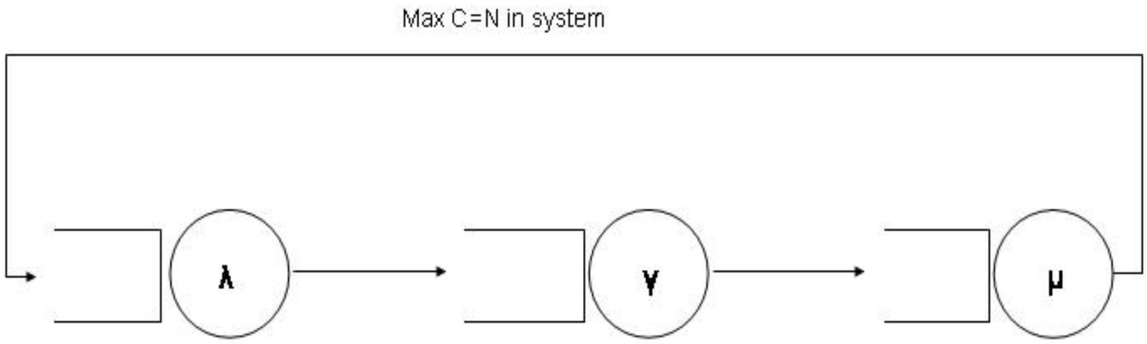


Figure 2.14: Finite source model for the special case when $N = C$

In [1], a mean value analysis for exponential closed networks is given. An arbitrary tote moving from one station to another sees the system in equilibrium corresponding to a population with one tote less. We number the workstations in logical order according to Figure 2.14. First we have to introduce some notation:

$$\begin{aligned}
\mathbb{E}[S_i(k)] &= \text{mean production lead time at station } i;\\
\mathbb{E}[L_i(k)] &= \text{mean number of jobs at station } i;\\
\Lambda_i(k) &= \text{throughput of station } i, \text{in a network with population } k;
\end{aligned}$$

For each station we write the mean production lead time, which is based on the lead time of an $M/M/c$-system:

$$\begin{aligned}
\mathbb{E}[S_1(k)] &= \Pi_1(k-1)\frac{1}{\lambda} + (\mathbb{E}[L_1(k-1)] - \frac{\Lambda_1(k-1)}{\lambda})\frac{1}{\lambda} + \frac{1}{\lambda},\\
\mathbb{E}[S_2(k)] &= \frac{1}{\gamma},\\
\mathbb{E}[S_3(k)] &= \Pi_3(k-1)\frac{1}{\mu} + (\mathbb{E}[L_3(k-1)] - \frac{\Lambda_3(k-1)}{\mu})\frac{1}{\mu} + \frac{1}{\mu},
\end{aligned}$$

where $\Pi_i(k-1)$ is the probability that all machines in workstation i are busy, in a network with population $k-1$. Instead of trying to compute $\Pi_i(k-1)$, we approximate this probability by the probability that all servers are busy in an $M/M/c_i$ system with arrival rate $\Lambda_i(k-1)$ and service rate μ_i. Where we have in our case $c_1 = c_3 = 1$ and $c_2 = \infty$, furthermore it is clear that $\mu_1 = \lambda$, $\mu_2 = \gamma$ and finally $\mu_3 = \mu$. Hence for workstation i,

$$\Pi_i(k-1) \approx \frac{\frac{1}{c_i}(\frac{\Lambda_i(k-1)}{\mu_i})^{c_i}}{(1-\frac{\Lambda_i(k-1)}{c_i\mu_i})\sum_{j=0}^{c_i-1}(\frac{\Lambda_i(k-1)}{\mu_i})^j + \frac{1}{c_i!}(\frac{\Lambda_i(k-1)}{\mu_i})^{c_i}}.$$

The throughput of each station is the same and is given by

$$\Lambda_i(k) = \frac{k}{\mathbb{E}[S_1(k)] + \mathbb{E}[S_2(k)] + \mathbb{E}[S_3(k)]}, \quad i = 1, 2, 3. \tag{2.44}$$

By Little's law

$$\mathbb{E}[L_i(k)] = \Lambda_i(k)\mathbb{E}[S_i(k)], \quad i = 1, 2, 3.$$

These relations can be used to recursively compute $\Lambda_i(k), \Pi_i(k), \mathbb{E}[S_i(k)]$ and $\mathbb{E}[L_i(k)]$ for arbitrary populations. To start the recursion we have to initialize with

$$\mathbb{E}[L_i(0)] = 0, \quad i = 1, 2, 3.$$

The utilization of the zone is given using equation (2.44) for the throughput

$$\rho_3(k) = \frac{\Lambda_3(k)}{\mu} \tag{2.45}$$

This algorithm can be used to compute the utilization of the zone and to compare it with the simulation results. The outcome of both computations and the relative error in percentages are given in Table 2.5. As we can see in Table 2.5 the simulation generates values which are close to the values we would expect according to the theory.

$N = C$	Simulation	Theory	$\Delta\%$
1	0,184 (±0,007)	0,181	2,0
2	0,339 (±0,011)	0,335	1,2
3	0,447 (±0,010)	0,461	2,9
4	0,552 (±0,012)	0,558	1,2
5	0,635 (±0,013)	0,632	0,5

Table 2.5: Simulation results for utilization compared with results from queueing theory

When we have an infinite server as a source then the algorithm has to be adjusted. The only two things that are changed are: $c_1 = \infty$ and $\mathbb{E}[S_1(k)] = \frac{1}{\lambda}$. The rest of the algorithm stays the same. We also checked whether or not there is a big difference between the use of deterministic travel times or exponential travel times. This difference is negligible and so we will simulate with deterministic travel times, which coincided with reality.

These three special cases suggest that the simulation is reliable so we can generate results and compare them with our models.

2.6 Results

In this section we will present results obtained from the simulation and analysis, n=by adjusting some of the system parameters. The outcomes of the simulation and analysis will be compared with each other, so we can see if the analysis is accurate enough. We will look how well the models we presented will match with a real situation (the simulation). The open model can be simulated by choosing the value of C large enough so that there is a supply of new totes with constant rate, the saturated model can be simulated by choosing the time at the source very small so that there are always C totes in the system. This situation can occur during a period when the system is very busy. If we keep the arrival rate constant the single server is simulated and if the arrival rate is depending on the number of totes in the system, the infinite server finite source model can be simulated. When generating results the several variables are chosen to standardly take on the following values:

1. Number of waiting positions: $N = 4$,
2. Mean service time: $1/\mu = 30$ seconds,
3. Mean inter-arrival time: $1/\lambda = 36$ seconds,
4. Mean retrial time: $1/\gamma = 100$ seconds,
5. Maximum number of totes in system: $C = 10$.

2.6.1 Finite Source Model (single server)

The results for the finite source model, with a single server source, as well as the open model will be presented in this section. This because when C is large enough these two models will act the same and the open model will be a limiting situation of the finite source model. In this section several variables will be changed. This model represents the situation in reality that the zone is at the beginning of the conveyor and totes will leave the system directly if served.

Influence of total numbers of totes in system

The total number of totes in the system is an important variable so therefore we shall begin with changing this variable and keeping the rest fixed. The utilization is an important performance measure that we will consider here first. The results are given in Table 2.6 en Figure 2.15. As we can see the results of the simulation are very close compared with analytical results when C is relatively large. For small C the calculation is very bad compared with the simulation results. In our model, for small C, almost no retrial takes place because this only will happen if a tote is blocked. In the simulation this will not be the case, there always retrial takes place, resulting in bad results for small C. We will also see this later on when regarding other variables. For this case, when C is larger than 10 the relative error is pretty reasonable. We can also see that the values of the finite source model and of the simulation converge to the open model, as predicted. We have checked this also by doing one simulation for $C = 100$ and we obtained an utilization of $\rho = 0,8355$.

In the same manner we can make a table of the results for the throughput of the system. These results can be found in Table 2.7. Here we see the same trend as we saw by the results of utilization. Also the same conclusions can be stated concerning the accuracy of the results.

C	Simulation	Finite Source Model	$\Delta\%$
5	0,633 (±0,013)	0,747	18,1
6	0,673 (±0,013)	0,766	13,7
7	0,705 (±0,015)	0,780	10,5
8	0,743 (±0,016)	0,791	6,4
9	0,764 (±0,014)	0,799	4,6
10	0,780 (±0,017)	0,805	3,3
11	0,794 (±0,018)	0,811	2,1
12	0,799 (±0,018)	0,815	2,0
13	0,796 (±0,016)	0,818	2,7
14	0,813 (±0,018)	0,821	1,0
15	0,813 (±0,017)	0,823	1,3
16	0,813 (±0,020)	0,825	1,5

Table 2.6: Simulation results for utilization compared with results of the finite source model, when C is varied

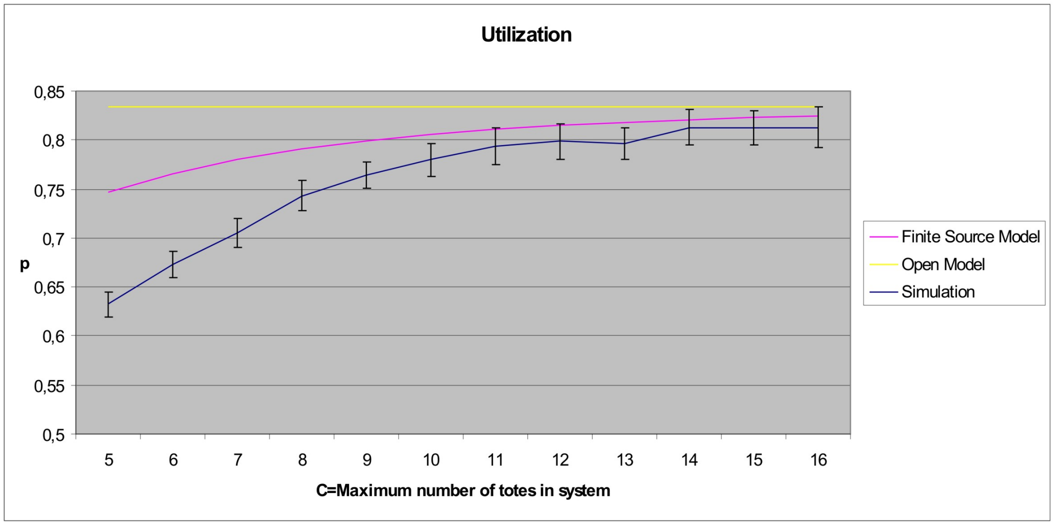


Figure 2.15: Simulation results and finite source model results for the utilization, when C is varied.

Influence of buffersize

The buffersize is the second important variable which will be changed to generate the results for the simulation and the analysis of the finite source model. The utilization results are given in Table 2.8. Here we can see that for all values of N the relative error is approximately 4%, which is acceptable but not very good. The buffersize does not have as much influence as the maximum number of totes in the system. If we choose C larger, then the results will become better, as can be seen in the previous section where C was varied.
The throughput results show the same accuracy and also the same conclusions can be applied to these results.

C	Simulation	Finite Source Model	$\Delta\%$
5	74,9 (±1,0)	89,6	19,6
6	82,3 (±1,2)	91,9	11,7
7	86,7 (±1,3)	93,6	7,9
8	91,0 (±1,5)	94,9	4,3
9	91,8 (±1,5)	95,9	4,4
10	93,6 (±1,5)	96,7	3,3
11	94,8 (±1,4)	97,3	2,1
12	96,2 (±1,7)	97,8	2,0
13	96,9 (±1,6)	98,2	2,7
14	99,1 (±2,0)	98,5	1,0
15	97,9 (±2,0)	98,8	1,3
16	97,5 (±1,9)	99,0	1,5

Table 2.7: Simulation results for throughput per hour compared with results of the finite source model, when C is varied

N	Simulation	Finite Source Model	$\Delta\%$
4	0,776 (±0,019)	0,805	3,8
5	0,7815 (±0,015)	0,8076	3,4
6	0,780 (±0,016)	0,807	3,5
7	0,774 (±0,016)	0,807	4,3
8	0,783 (±0,015)	0,807	3,2
9	0,764 (±0,016)	0,807	5,6

Table 2.8: Simulation results for utilization compared with results of the finite source model, when N is varied

Influence of mean pick time

The time the operator is working on one tote, the pick time, is an interesting variable to variate and to see how this influences the system performance, especially the utilization. In Table 2.9 the results can be found including relative errors and also some results for the open model. We can see that when the operator is working very fast the utilization will be low, this is what we would expect because then he has to wait relatively a long time to work on the next tote. Also we can see

$1/\mu$	Simulation	Finite Source Model	$\Delta\%$	Open Model	$\Delta\%$
5	0,137 (±0,004)	0,139	1,8	0,139	1,8
10	0,269 (±0,008)	0,278	3,2	0,278	3,2
20	0,541 (±0,015)	0,555	2,5	0,556	2,7
30	0,779 (±0,017)	0,805	3,4	0,833	6,9
40	0,923 (±0,011)	0,944	2,4	-	-
50	0,974 (±0,007)	0,985	1,2	-	-

Table 2.9: Simulation results for utilization compared with results of the finite source model, when μ is varied

that the results are good for the finite source model. The open model is also very good for small mean pick times. This is because the maximum number of totes in the system (C) is unlikely to be reached, because the operator works too fast.

Influence of mean retrial time

The time a tote has to travel over the conveyor, also called retrial time, is the fourth variable which we shall discuss here. The results of the utilization and throughput per hour will be more or less the same, so we shall present only the results for the utilization of the operator (Table 2.10). We can see immediately that when the retrial time is large the relative error is getting bigger and the analysis is not accurate anymore.

$1/\gamma$	Simulation	Finite Source Model	$\Delta\%$
30	0,802 (±0,019)	0,807	0,7
45	0,794 (±0,016)	0,807	1,6
60	0,785 (±0,016)	0,806	2,8
75	0,779 (±0,016)	0,806	3,6
90	0,777 (±0,017)	0,806	3,7
120	0,747 (±0,016)	0,805	7,7

Table 2.10: Simulation results for utilization compared with results of the finite source model, when γ is varied

Influence of quotient of mean retrial time and mean pick time

In this section we shall investigate the influence of the quotient of the mean retrial time and the mean pick time. The results are presented graphically in Figure 2.16 and the most interesting results (when $1/\mu = 45$) are given in Table 2.11. We can clearly see again that the faster the operator works, the lower the utilization. Also we see a very low influence of the mean retrial time on the utilization, this influence is bigger when the mean pick time is bigger. The accuracy of the model is then also getting worse, but for most of the cases the model is pretty accurate. The biggest relative error is $7,9\%$ when the quotient and the mean retrial time take on their maximum value.

$\frac{\mu}{\gamma}$	Simulation ($\frac{1}{\mu} = 45$)	Finite Source Model	$\Delta\%$
1	0,970 (±0,006)	0,976	0,5
2	0,953 (±0,009)	0,973	2,0
3	0,941 (±0,009)	0,969	3,0
4	0,917 (±0,011)	0,965	5,2
5	0,890 (±0,010)	0,960	7,9

Table 2.11: Simulation results for utilization compared with results of the finite source model, when $\frac{\mu}{\gamma}$ and $1/\mu = 45$

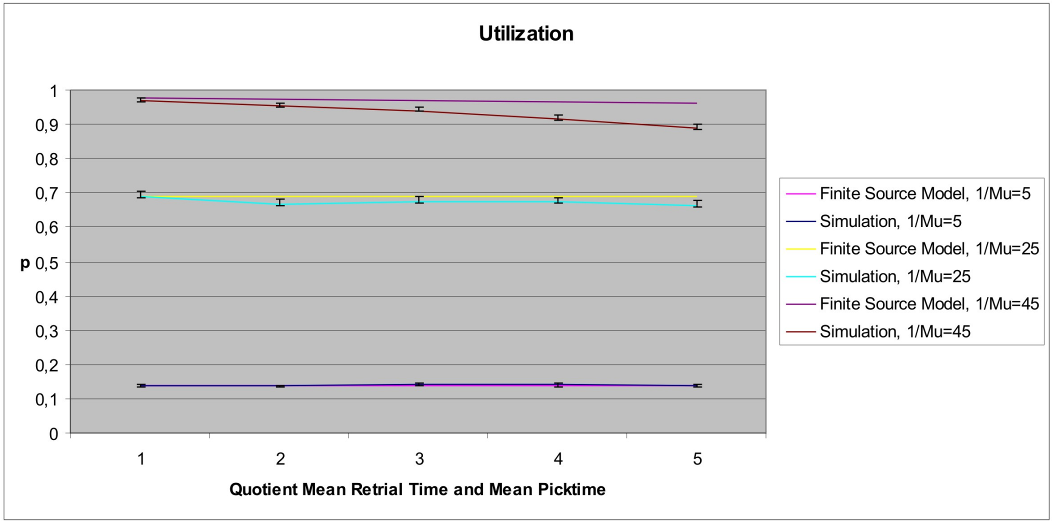


Figure 2.16: Simulation results and finite source model results for the utilization, when $\frac{\mu}{\gamma}$ is varied.

Influence of mean inter-arrival time

The mean inter-arrival time is also a variable that we can alter and investigate its influence. For the utilization of the operator we have the following results, see Table 2.12 and Figure 2.17.

$1/\lambda$	Simulation	Finite Source Model	$\Delta\%$
24	0,922 (±0,010)	0,967	4,9
26	0,913 (±0,009)	0,951	4,2
28	0,885 (±0,012)	0,930	5,1
30	0,864 (±0,013)	0,903	4,5
32	0,836 (±0,014)	0,873	4,4
34	0,817 (±0,017)	0,840	2,7

Table 2.12: Simulation results for utilization compared with results of the finite source model, when λ is varied

The relative error is for all values almost the same, so the inter-arrival time does not not have a positive or a negative influence on the accuracy of the model to the simulation. When C is taken larger the relative error will be better, as we have seen before. We can also see in Figure 2.17 that when the mean of the inter-arrival time is getting bigger the value of the utilization converges to the one of the open model.

2.6.2 Finite Source Model (infinite server)

In this section the finite source model with an infinite server as source will be investigated. The simulation has to be adjusted to make a good comparison; the arrival rate is not constant anymore but depends on the number of totes in the system. This is a relatively easy adjustment; we divide

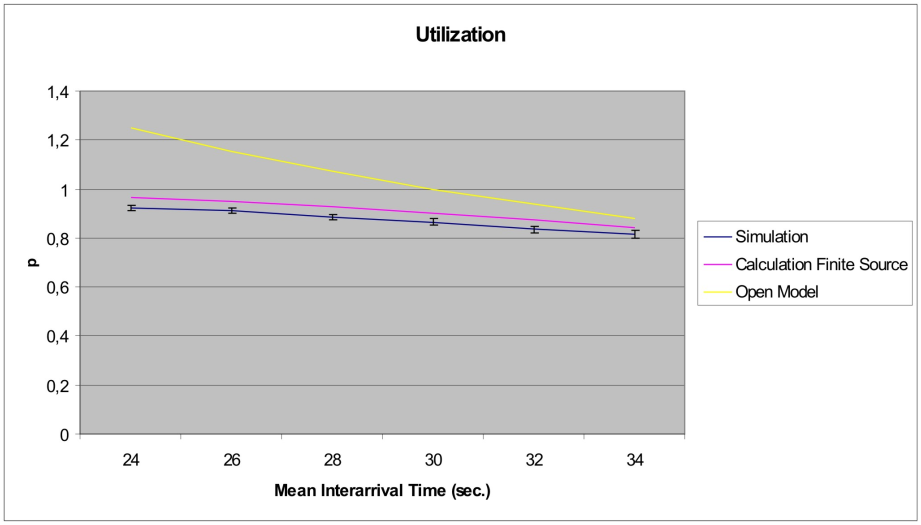


Figure 2.17: Simulation results and finite source model results for the utilization, when λ is varied.

the inter-arrival time by the number of totes which could be added to the system. If no tote can be added, because the system is full, the supply will stop until the system is not full anymore and then the inter-arrival time will be again as just stated. We will vary the same variables as done in the previous section, with the same values. This model also represents the situation in reality that the zone is at the beginning of the conveyor and totes will leave the system directly if served.

Influence of total numbers of totes in system

In the previous section, when we considered the finite source model with single server, we saw that the maximum number of totes allowed in the system (C) had a great influence on the system performance. This parameter shall be of interest in this part of this section. In Table 2.13 the results are given. The same relation can be seen as with the other finite source model. For this model however the relative errors are bigger and again we see pretty good results when C is bigger than 10. The large differences occur because of the difference between the simulation and the model; in the simulation the zone is at the end of the conveyor and not at the beginning. This will also explain the errors in the rest of this section.

Influence of buffersize

The buffersize did not have great influence when we simulated the finite source model with single server. In Table 2.31 we can see the results for the finite source model but now with an infinite server. We see the same trend as we saw with the other finite source model; the buffersize does not have great influence on the utilization and the relative error is approximately the same as for all buffersizes. For bigger values of C the results will be better, as we have seen in the previous part. The throughput results are similar.

C	Simulation	Finite Source Model (inf. server)	$\Delta\%$
5	0,750 (±0,007)	0,986	31,5
6	0,832 (±0,007)	0,991	19,2
7	0,899 (±0,006)	0,994	10,5
8	0,928 (±0,005)	0,996	7,3
9	0,958 (±0,004)	0,997	4,1
10	0,978 (±0,003)	0,998	2,0
11	0,983 (±0,002)	0,998	2,1
12	0,988 (±0,002)	0,999	2,0
13	0,990 (±0,001)	0,999	2,7
14	0,993 (±0,001)	0,999	1,0
15	0,995 (±0,001)	0,999	1,3
16	0,995 (±0,001)	0,999	1,4

Table 2.13: Simulation results for utilization compared with results of the finite source (infinite server) model, when C is varied

N	Simulation	Finite Source Model (inf. server)	$\Delta\%$
4	0,975 (±0,003)	0,9978	2,3
5	0,983 (±0,002)	1,000	1,7
6	0,987 (±0,002)	1,000	1,3
7	0,988 (±0,002)	1,000	1,2
8	0,989 (±0,002)	1,000	1,1
9	0,990 (±0,002)	1,000	1,0

Table 2.14: Simulation results for utilization compared with results of the finite source (infinite server) model, when N is varied

Influence of mean pick time

The time the operator is working on one tote, the pick time, is the next in line to vary. In Table 2.15 the results can be found including relative errors. We can see that when the operator is working very fast the utilization will be low, this is what we would expect because then he has to wait relatively a long time for a new tote. Also we can see that the results are bad for the finite source

$1/\mu$	Simulation	Finite Source Model (inf. server)	$\Delta\%$
5	0,348 (±0,005)	0,821	136,1
10	0,620 (±0,010)	0,952	53,5
20	0,904 (±0,009)	0,992	9,7
30	0,974 (±0,003)	0,998	2,4
40	0,991 (±0,002)	0,999	0,8
50	0,997 (±0,001)	1,000	0,3

Table 2.15: Simulation results for utilization compared with results of the finite source (infinite server) model, when μ is varied

model, when the operator works very fast. The model gives a high utilization, because the totes will return very quickly to the source, in the simulation this will not be the case. We do see good results when the mean pick time is greater then 30 seconds.

Influence of mean retrial time

The travel time over the conveyor is the variable which tells something about the supply of totes and therefore it is a parameter which shall be varied and discussed here. The results can be found in Table 2.16. Here we see the same trend as we saw in the single server case; when the retrial time increases the model will be less accurate. But for this model the relative errors are still reasonable for the cases which are considered.

$1/\gamma$	Simulation	Finite Source Model (inf. server)	$\Delta\%$
30	0,999 (±0,000)	1,000	0,0
45	0,998 (±0,000)	1,000	0,1
60	0,994 (±0,001)	0,999	0,5
75	0,991 (±0,001)	0,999	0,7
90	0,985 (±0,002)	0,998	1,3
120	0,960 (±0,004)	0,997	3,9

Table 2.16: Simulation results for utilization compared with results of the finite source (infinite server) model, when γ is varied

Influence of quotient of mean retrial time and mean pick time

As said for the previous model, the interaction between the retrial time and pick time is of interest for the performance of the system. The same values are used as is done earlier. So we choose a mean pick time of $5, 25$ and 45 seconds and the quotient of the mean retrial time and mean pick time is varied between $1, 2, 3, 4$ and 5. In Table 2.17 the results are given for the case when $1/\mu = 5$ and in Figure 2.18 all the results are presented. The results are as we would expect, for small pick times and long retrial times the results are bad as we have seen earlier. This can be seen from the results from Table 2.17 and also in the graph of Figure 2.18. In this graph the other results are also presented and here we can see that the model is more accurate for these values, but still not very good.

$\frac{\mu}{\gamma}$	Simulation ($\frac{1}{\mu} = 5$)	Finite Source Model (inf. server)	$\Delta\%$
1	0,879 (±0,004)	0,911	3,7
2	0,830 (±0,005)	0,906	9,1
3	0,781 (±0,005)	0,900	15,3
4	0,737 (±0,006)	0,893	21,2
5	0,693 (±0,006)	0,887	27,9

Table 2.17: Simulation results for utilization compared with results of the finite source model, when $\frac{\mu}{\gamma}$ and $1/\mu = 5$

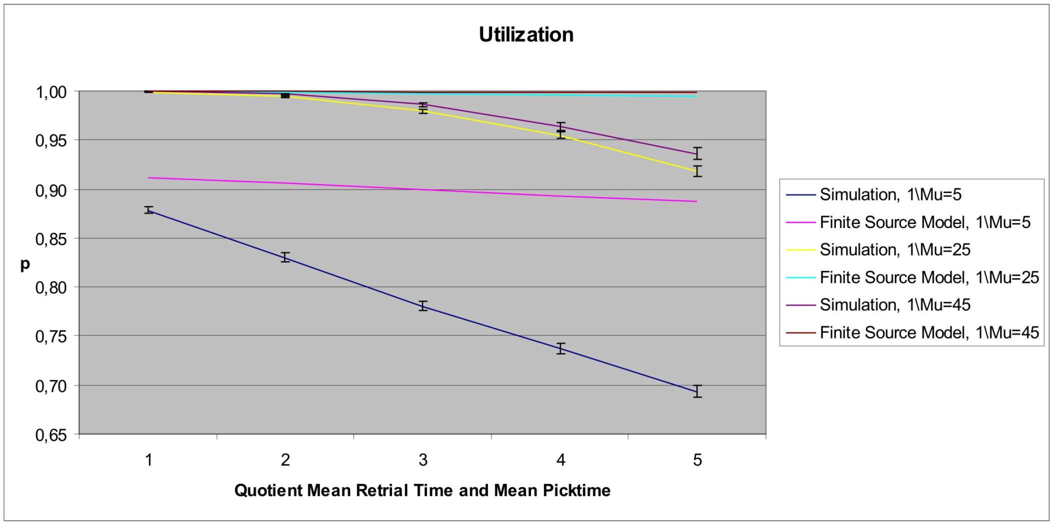


Figure 2.18: Simulation results and finite source (infinite server) model results for the utilization, when $\frac{\mu}{\gamma}$ is varied.

Influence of mean inter-arrival time

The retrial time tells something about the supply of totes but the mean inter-arrival time also is a great factor for the supply. In the single server case we saw that the relative error was approximately the same and that the inter-arrival time had the influence we would expect. In Table 2.18 the results when we change this variable can be found. Here we see that the inter-arrival time does not have great influence on the utilization of the operator. This can be explained by the fact that we look at the number of totes that are in the system to define the mean arrival time. When the mean inter-arrival is relative small the influence of the number of totes in the system is also small, because then there is not much difference in the arrival rate if there is one more or one less tote. The model gives relative errors of approximately 2% so this is very reasonable.

$1/\lambda$	Simulation	Finite Source Model (inf. server)	$\Delta\%$
24	0,980 (±0,003)	0,999	2,0
26	0,979 (±0,003)	0,999	2,0
28	0,978 (±0,002)	0,999	2,1
30	0,978 (±0,002)	0,998	2,1
32	0,980 (±0,002)	0,998	1,9
34	0,978 (±0,002)	0,998	2,1

Table 2.18: Simulation results for utilization compared with results of the finite source (infinite server) model, when λ is varied

2.6.3 Saturated Model

As said earlier we will choose the inter-arrival time very small, we have chosen for 1 second, to simulate the saturated model. All other variables will be chosen as before and we will change one parameter at the time. We will study the influence on the system performance of the total number of totes in the system (C), the buffersize (N), the mean pick time ($1/\mu$) and also the mean retrial time ($1/\gamma$). At the end we also look at the influence of the quotient of these last two variables. This is interesting because the pick time says something about the rate at which totes leave the zone and the retrial time about the rate when they arrive at the zone. The interaction (or quotient) between these two variables is very useful to investigate. For this we choose three mean pick times, one small ($1/\mu = 5$), one medium ($1/\mu = 25$) and one large ($1/\mu = 45$). For each pick time we evaluate 5 different values for the mean retrial time, such that the quotient of the mean retrial time and mean pick time takes on the values: $1, 2, 3, 4$ and 5.

Influence of total number of totes in system

The utilization of the operator will be regarded, as we vary the total number of totes in the system. The results can be found in Table 2.19. Note that the relative errors are given in percentages,

C	Simulation	Saturated Model	$\Delta\%$
5	0,853 (±0,007)	0,855	0,2
6	0,914 (±0,006)	0,914	0,0
7	0,945 (±0,005)	0,948	0,4
8	0,968 (±0,003)	0,968	0,0
9	0,978 (±0,002)	0,979	0,1
10	0,985 (±0,002)	0,986	0,1

Table 2.19: Simulation results for utilization compared with results of the saturated model, when C is varied

so these calculation results of our analysis are very close to the results obtained by simulation. We can see that when more totes are allowed in the system, the higher the utilization. This is what we would expect, because more totes (so more work) will travel around in the system and hindering at the conveyor is not considered.

The same can be seen for the results of the throughput per hour, here we will just present the results in table form (Table 2.20). Again the results of the closed model calculation is very accu-

C	Simulation	Saturated Model	$\Delta\%$
5	101,0 (±1,4)	102,6	1,5
6	110,3 (±1,6)	109,7	0,6
7	114,8 (±1,7)	113,8	0,9
8	114,5 (±2,0)	116,1	1,4
9	118,4 (±1,8)	117,5	0,7
10	119,3 (±1,8)	118,3	0,9

Table 2.20: Simulation results for throughput per hour compared with results of the saturated model, when C is varied

rate and the same influence of C can be seen.

Influence of buffersize

In this section we will keep the total number of totes in the system constant and we will vary the buffersize of the zone. The results of the simulation, analysis and the relative error are presented in Table 2.21. We can see that when the buffersize is increasing then also the utilization increases, but the buffersize has clearly less influence on the utilization than the total number of totes allowed in the system. The results of the analysis are very close to the results of the simulation, again note here that the relative error is given in percentages!

N	Simulation	Saturated Model	$\Delta\%$
4	0,986 (±0,002)	0,986	0,0
5	0,992 (±0,001)	0,993	0,1
6	0,995 (±0,001)	0,996	0,1
7	0,998 (±0,001)	0,997	0,0
8	0,998 (±0,001)	0,998	0,0
9	0,998 (±0,001)	0,998	0,0

Table 2.21: Simulation results for utilization compared with results of the saturated model, when N is varied

The results for the throughput are comparable and are given in Table 2.22.

N	Simulation	Saturated Model	$\Delta\%$
4	118,7 (±2,3)	118,3	0,3
5	120,9 (±2,2)	119,2	1,4
6	119,9 (±2,2)	119,5	0,3
7	120,0 (±2,2)	119,7	0,2
8	121,0 (±1,9)	119,8	1,0
9	118,6 (±2,3)	119,8	1,0

Table 2.22: Simulation results for throughput per hour compared with results of the saturated model, when N is varied

Influence of mean pick time

The time an operator needs to serve a tote is an important variable which can be changed. The utilization results are given in Table 2.23. The results are again good, but we can also see that for a small service time the model deviates more from the simulation than for longer times. But even when the service time is small the model is accurate.

Influence of mean retrial time

Another interesting quantity which can be varied is the mean retrial time or the time the tote travels on the conveyor to reach the zone. In Table 2.24 the results are presented, here also the

$1/\mu$	Simulation	Saturated Model	$\Delta\%$
5	0,449 (±0,004)	0,458	2,1
10	0,740 (±0,010)	0,758	2,4
20	0,942 (±0,007)	0,951	1,0
30	0,986 (±0,002)	0,986	0,1
40	0,996 (±0,001)	0,995	0,1
50	0,998 (±0,001)	0,998	0,0

Table 2.23: Simulation results for utilization compared with results of the saturated model, when μ is varied

$1/\gamma$	Simulation	Saturated Model	$\Delta\%$
30	1,000 (±0,000)	1,000	0,0
45	0,999 (±0,000)	0,999	0,0
60	0,998 (±0,001)	0,998	0,0
75	0,995 (±0,001)	0,995	0,0
90	0,990 (±0,002)	0,990	0,0
120	0,974 (±0,003)	0,975	0,0

Table 2.24: Simulation results for utilization compared with results of the saturated model, when γ is varied

relative errors are given. As we can see the utilization is converging to 1 when the retrial time is getting smaller. This is clearly the case, because a blocked tote will be back very soon and the operator will be busy all of the time. The results are again very good and the relative errors are very small. Approximately the same results are obtained when looking at the throughput per hour of the system.

Influence of quotient of mean retrial time and mean pick time

Instead of only looking at the time a tote travels over the conveyor or only at the time the tote is being served, the quotient of these two is also very interesting. To investigate the influence of this quotient we will use the values for μ and the quotient stated at the beginning of this section. Note that the outcomes for the saturated model do only depend on the quotient of the mean retrial time and mean pick time. In other words; for each value of the quotient the same outcome will be generated, independent of the choice of the mean pick time.

In Table 2.25 and Figure 2.19 the results are summarized and we can see that the relative errors are very low and acceptable. We can also see that when the mean pick time and the quotient of the mean retrial time and mean pick time increase then the relative error is also increasing, but still very reasonable.

In Table 2.25 the results for $1/\mu = 25$ are omitted, these results and also the relative errors are in between by the results of $1/\mu = 5$ and $1/\mu = 45$. This can also graphically be seen in Figure 2.19. The width of the 95% confidence interval is here at most $0,015$. The results for the utilization per hour show the same effect when the mean pick time and the quotient are increased, also the relative errors are good.

$\frac{\mu}{\gamma}$	Saturated Model	Simulation ($\frac{1}{\mu} = 5$)	$\Delta\%$	Simulation ($\frac{1}{\mu} = 45$)	$\Delta\%$
1	1,000	1,000 (±0,000)	0,0	1,000 (±0,000)	0,0
2	0,998	0,998 (±0,000)	0,0	0,997 (±0,001)	0,0
3	0,990	0,991 (±0,001)	0,0	0,989 (±0,003)	0,1
4	0,975	0,974 (±0,001)	0,1	0,974 (±0,004)	0,2
5	0,951	0,950 (±0,002)	0,1	0,936 (±0,007)	1,6

Table 2.25: Simulation results for utilization compared with results of the saturated model, when $\frac{\mu}{\gamma}$ is varied and $1/\mu = 5$ and $1/\mu = 45$

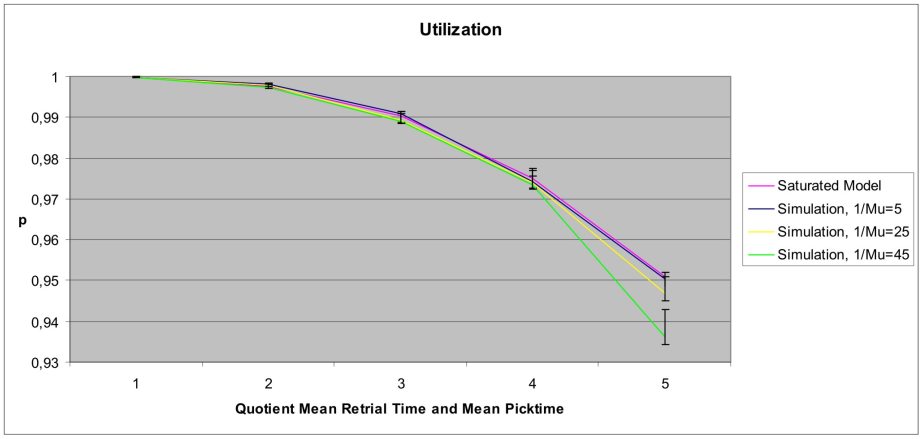


Figure 2.19: Simulation results and saturated model results for the utilization, when $\frac{\mu}{\gamma}$ is varied.

2.6.4 Iterative Method (single server)

In this section we will investigate the accuracy of the algorithm proposed in section 2.4.5. We shall vary the different parameters to see if the algorithm works well for all special cases. We have chosen $\epsilon = 10^{-8}$ as accuracy for the blocking probability. In this section we will assume the single server case for the source and in the next section we will investigate the infinite server case.

Influence of total number of totes in system

First we will vary the total number of totes that are allowed in the system. The utilization results can be found in Table 2.26 and for the throughput results the same results are obtained. We can see that the iterative method give reasonable results, the relative error is almost for any case under 4%. For bigger C this error will become smaller, because the answer of the iterative method will change barely and the simulation will converge to that answer.

Influence of buffersize

The buffersize is also one of the main parameters which will be varied. The results for the utilization can be found in Table 2.27, where in the last column the relative errors are stated. The results

C	Simulation	Iterative Method	Δ%
5	0,633 (±0,013)	0,633	0,0
6	0,673 (±0,013)	0,689	2,3
7	0,705 (±0,015)	0,733	3,9
8	0,743 (±0,016)	0,766	3,1
9	0,764 (±0,014)	0,792	3,6
10	0,780 (±0,017)	0,809	3,8
11	0,794 (±0,018)	0,821	3,4
12	0,799 (±0,018)	0,828	3,6
13	0,796 (±0,016)	0,831	4,4
14	0,813 (±0,018)	0,833	2,4
15	0,813 (±0,017)	0,833	2,5
16	0,813 (±0,020)	0,833	2,4

Table 2.26: Simulation results for utilization compared with results of the iterative method, when C is varied

N	Simulation	Iterative Method	Δ%
4	0,776 (±0,019)	0,809	4,3
5	0,781 (±0,015)	0,806	3,2
6	0,780 (±0,016)	0,800	2,5
7	0,774 (±0,016)	0,793	2,4
8	0,783 (±0,015)	0,787	0,5
9	0,764 (±0,016)	0,783	2,5

Table 2.27: Simulation results for utilization compared with results of the iterative method, when N is varied

are again reasonable, for almost all cases the relative error is smaller then 4%. We also see that the buffersize has almost no influence on the utilization, we can see this in the simulation results but also in the results of the iterative method. When the buffersize increases we see an increase in the utilization, which is remarkable. This is for both the simulation as the iterative method the case. It seems that this counter intuitive result is a property of the system we are considering.

Influence of quotient of mean retrial time and mean pick time

In this section we shall change two variables which are important for the system; the mean retrial time and the mean pick time. For this we choose the same values as done for the other models, so $1/\mu = 5, 25$ and 45 and for the quotient we have the values 1 to 5. Here we will only present the results for $1/\mu = 45$ in Table 2.28, because these are the most interesting results. The relative errors for $1/\mu = 5$ are around 1% and for $1/\mu = 25$ the errors are approximately 2%. The most interesting results (when $1/\mu = 45$) are given in Table 2.28. We can see that when the mean retrial time increases, the utilization decreases. This makes sense because when a tote is coming back very fast the operator will have more work per time unit. We also saw in the other results that when the mean pick time is small the utilization is also small, which is a logical consequence. Overall the influence of the quotient is there but this influence is very limited as also can be seen in Figure 2.20.

$\frac{\mu}{\gamma}$	Simulation ($\frac{1}{\mu} = 45$)	Iterative Method	$\Delta\%$
1	0,970 (±0,006)	0,999	2,9
2	0,953 (±0,009)	0,987	3,6
3	0,941 (±0,009)	0,963	2,3
4	0,917 (±0,011)	0,930	1,4
5	0,890 (±0,010)	0,893	0,3

Table 2.28: Simulation results for utilization compared with results of the iterative method, when $\frac{\mu}{\gamma}$ is varied and $1/\mu = 45$

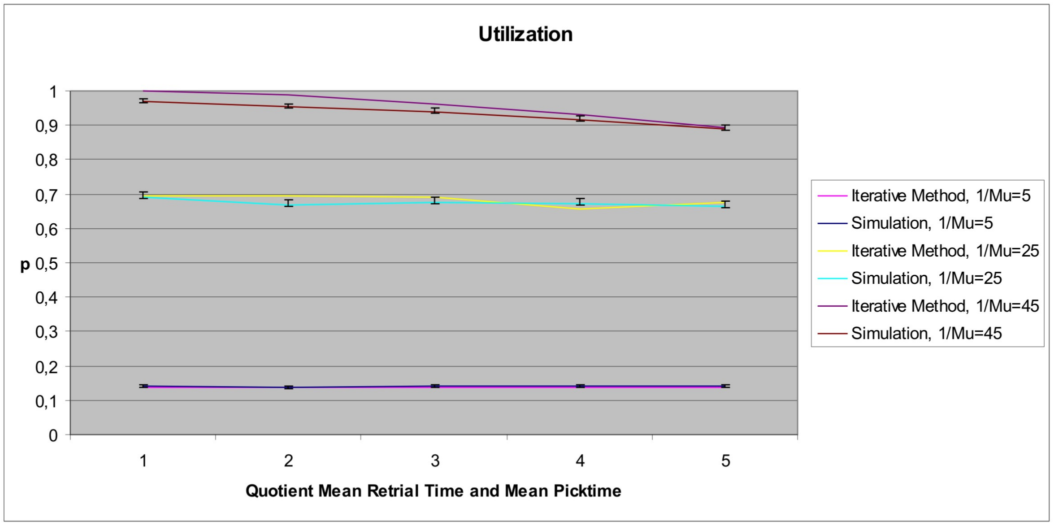


Figure 2.20: Simulation results and iterative method results for the utilization, when $\frac{\mu}{\gamma}$ is varied.

In Table 2.28 we can see that the results for large mean pick times are worse than for small. The relative error however is around 3% which is pretty good. It also seems that the accuracy of the model is getting better when the mean retrial time is getting bigger, because then the tote will have a longer time to get back and it will have more chance to see the system in equilibrium.

Influence of mean inter-arrival time

The mean inter-arrival time of new totes is the last parameter which we will change. In Table 2.29 the results for the utilization of the zone are presented for the simulation and the iterative method. In the final column the relative errors are stated. As we would expect the utilization increases when the mean inter-arrival time decreases, because more work will arrive to the operator. The accuracy of the iterative method is approximately $3 - 4\%$, so this is a little bit worse than what we have seen earlier, but this will also depend on the choice of the other parameters.

$1/\lambda$	Simulation	Iterative Method	$\Delta\%$
24	0,922 (±0,010)	0,952	3,2
26	0,913 (±0,009)	0,940	3,9
28	0,885 (±0,012)	0,924	4,4
30	0,864 (±0,013)	0,902	4,4
32	0,836 (±0,014)	0,875	4,7
34	0,817 (±0,017)	0,844	3,2

Table 2.29: Simulation results for utilization compared with results of the iterative method, when λ is varied

2.6.5 Iterative Method (infinite server)

Influence of total number of totes in system

For the single server case we saw that the total number of totes in the system had a great influence on the performance of the system. In Table 2.30 the results for the infinite server case are presented and compared with the iterative method. These results are very good, as we can see the total number of totes still has great influence on the utilization. Furthermore we can see that the iterative method is very accurate, all relative errors are smaller then 1% and the largest relative error is (only) $0,7\%$.

C	Simulation	Iterative Method	$\Delta\%$
5	0,750 (±0,007)	0,752	0,4
6	0,832 (±0,007)	0,838	0,7
7	0,899 (±0,006)	0,898	0,2
8	0,928 (±0,005)	0,936	0,9
9	0,958 (±0,004)	0,960	0,3
10	0,978 (±0,003)	0,975	0,4
11	0,983 (±0,002)	0,983	0,0
12	0,988 (±0,002)	0,989	0,1
13	0,990 (±0,001)	0,992	0,2
14	0,993 (±0,001)	0,994	0,1
15	0,995 (±0,001)	0,996	0,1
16	0,995 (±0,001)	0,997	0,2

Table 2.30: Simulation results for utilization compared with results of the iterative method, when C is varied

Influence of buffersize

For the size of the buffer we see the same trend as we saw earlier; this buffersize does have influence on the utilization but much less then the total number of totes in the system. In Table 2.31 the results can be found for both the simulation and the iterative method. These results are even better then we saw in the previous part, here all relative errors are far beneath 1% and the iterative method almost gives the exact answer.

N	Simulation	Iterative Method	$\Delta\%$
4	0,975 ($\pm$0,003)	0,975	0,1
5	0,983 ($\pm$0,002)	0,984	0,1
6	0,987 ($\pm$0,002)	0,988	0,1
7	0,988 ($\pm$0,002)	0,989	0,1
8	0,989 ($\pm$0,002)	0,989	0,0
9	0,990 ($\pm$0,002)	0,989	0,0

Table 2.31: Simulation results for utilization compared with results of the iterative method, when N is varied

Influence of quotient of mean retrial time and mean pick time

As said earlier in this section the quotient of the mean retrial time and mean pick time is an important variable. The same values will be chosen as was done before and we shall only present the results which are most remarkable, which can be found in Table 2.32. In Figure 2.21 all the results are presented graphically. The results are again for most of the part very good, which can be seen in Figure 2.21. The worst results are given when the mean pick time and the mean retrial time are very small, then the relative error is approximately 2%, these most interesting results can be found in Table 2.32. The results are getting better when the mean retrial time and/of the mean pick time increases.

$\frac{\mu}{\gamma}$	Simulation ($\frac{1}{\mu} = 5$)	Iterative Method	$\Delta\%$
1	0,879 ($\pm$0,004)	0,895	1,8
2	0,830 ($\pm$0,005)	0,8345	0,4
3	0,780 ($\pm$0,005)	0,779	0,2
4	0,737 ($\pm$0,006)	0,731	0,9
5	0,693 ($\pm$0,006)	0,687	0,9

Table 2.32: Simulation results for utilization compared with results of the iterative method, when $\frac{\mu}{\gamma}$ and $1/\mu = 5$

Influence of mean inter-arrival time

The rate at which each new tote arrives is the last parameter we will vary and the results of the simulation, the iterative method and the relative errors are stated in Table 2.33. The iterative method is very accurate; the relative errors are far beneath 1% and the biggest relative error is only $-0,3460\%$. For the single server case we saw a great influence of the inter-arrival time on the utilization. Here we can see that for the infinite server case the influence is much less. Also note the utilization is overall very high.

2.6.6 Iterative Method: Area of Accuracy

In this section we will take a closer look at the iterative method, because this method gives good answers and seems useful in the future. In the previous sections we investigated only the utilization (and throughput) of the system. Also other performance measures can be of interest, like

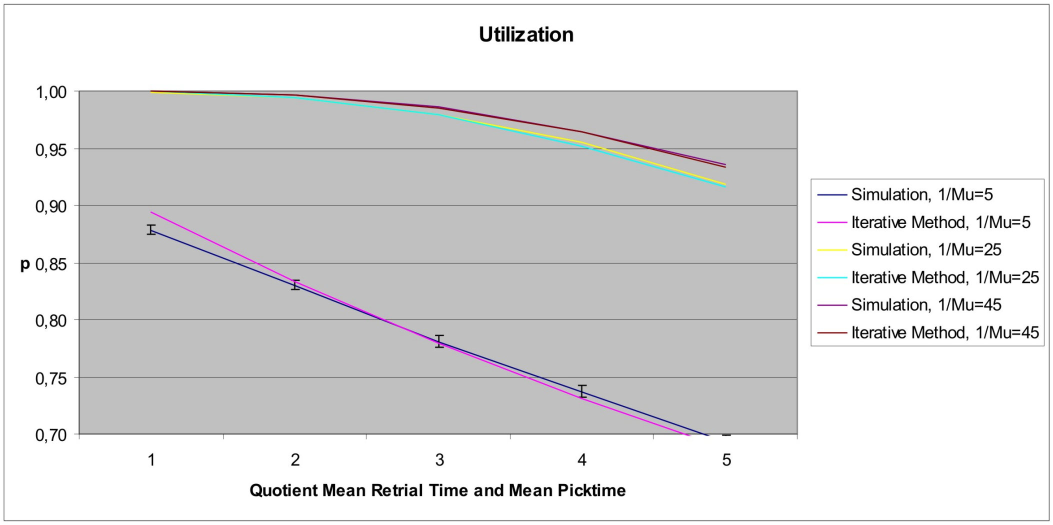


Figure 2.21: Simulation results and iterative method results for the utilization, when $\frac{\mu}{\gamma}$ is varied.

$1/\lambda$	Simulation	Iterative Method	$\Delta\%$
24	0,980 ($\pm$0,003)	0,979	0,0
26	0,979 ($\pm$0,003)	0,979	0,0
28	0,978 ($\pm$0,002)	0,978	0,0
30	0,978 ($\pm$0,002)	0,977	0,1
32	0,980 ($\pm$0,002)	0,976	0,3
34	0,978 ($\pm$0,002)	0,976	0,2

Table 2.33: Simulation results for utilization compared with results of the iterative method, when λ is varied

the mean buffersize. The blocking probability is also important because the iterative method is based on this probability and this probability is estimated every iteration step. For this we run 36 different situations of which we will not give all the results. The standard settings are the threshold of $C = 10$ and the mean pick time of $1/\mu = 30$.

The mean retrial time and the mean inter arrival time will be chosen to be smaller, equal and bigger than the mean pick time:

1. $1/\gamma = 10$, $1/\gamma = 30$ and $1/\gamma = 90$,
2. $1/\lambda = 10$, $1/\lambda = 30$ and $1/\lambda = 40$.

We will take every combination of these two variables and we have done this for buffersize $N = 1, 2, 5$ and 8, hence the number of 36 scenarios. For every scenario the results for utilization and throughput are good, except for two situations where the relative error is approximately 9%. These situations are the ones that the mean arrival time is the same as the mean pick time and the time for a retrial is equal or smaller than the mean pick time. As said earlier in this section we are more interested in the other performance measures, especially the probability of blocking (p).

We expect that the results for blocking will be bad when the mean time of a retrial is relatively small to the mean pick time. The main assumption in the iterative method is that an arriving tote sees the system in equilibrium but with one tote less in the system. If it takes a long time before a tote will come back after being blocked this seems to hold, but we do not know the accuracy when the time to retrial is relatively small.
The effects of the mean interarrival time are not quite clear beforehand, so it is interesting to see what will happen if we alter this variable. Now let us focus on the mean buffersize and the blocking probability.

The mean buffersizes (L_z) are for most of the cases very accurate, except for the two situations where the approximation of the utilization was also bad. The mean number of totes in the source (L_s) is calculated very accurate for all situations and for most situations the mean number of totes on the conveyor (L_c) is also approximated accurately.

The results for the blocking probability are much more interesting, because our method is based on on this probability. Here we will present some results to gain a clear insight into the accuracy. We will give the results of the simulation (with 95% confidence interval), denoted with subscript sim and of the iterative method, denoted with subscript it, when the buffer has size 5 for the 9 different scenarios. In Table 2.34 we state the results for the utilization and the blocking probability, we have omitted the results for the throughput, because these are very similar to the results for the utilization. In Table 2.35 we also state the mean number of totes in the source (L_s), the mean number of totes on the conveyor (L_c) and finally the mean number of totes in the zone or mean buffersize (L_z). Here the simulation error is for all three different means maximally $0,1$. In both tables the first two columns represent the variables that are altered; a - sign represents the choice of the smallest value of the variable, a 'o' the same value as $1/\mu$ and a + sign the highest value.
In both tables we can see that results are very good when the mean inter arrival time is smaller than the mean pick time. When the mean inter-arrival time is the same as the mean pick time we only see good results if the time to retrial is big. The results for the other values are not good but also not dramatic. If $1/\lambda$ is at its highest level all the results are bad, even a relative error of more

$1/\lambda$	$1/\gamma$	ρ_{it}	ρ_{sim}	$\Delta\%$	b_{it}	b_{sim}	$\Delta\%$
-	-	1,000	1,000 ±(0,000)	0,0	0,928	0,927 ±(0,000)	0,1
-	o	1,000	1,000 ±(0,000)	0,0	0,790	0,788 ±(0,001)	0,3
-	+	0,993	0,994 ±(0,000)	0,0	0,443	0,437 ±(0,003)	1,4
o	-	0,998	0,906 ±(0,004)	10,2	0,675	0,810 ±(0,006)	16,6
o	o	0,981	0,894 ±(0,004)	9,7	0,480	0,555 ±(0,008)	13,5
o	+	0,912	0,872 ±(0,004)	4,6	0,226	0,234 ±(0,006)	3,2
+	-	0,750	0,730 ±(0,006)	2,7	0,101	0,572 ±(0,016)	82,3
+	o	0,750	0,735 ±(0,007)	2,0	0,101	0,321 ±(0,019)	68,5
+	+	0,741	0,723 ±(0,007)	2,5	0,088	0,121 ±(0,006)	26,9

Table 2.34: Utilization and blocking probability results for simulation and iterative method, when $C = 10, N = 5$ and $1/\mu = 30$.

$1/\lambda$	$1/\gamma$	L_s^{it}	L_s^{sim}	$\Delta\%$	L_c^{it}	L_c^{sim}	$\Delta\%$	L_z^{it}	L_z^{sim}	$\Delta\%$
-	-	0,44	0,44	0,0	4,62	4,63	0,2	4,94	4,93	0,2
-	o	0,45	0,44	2,3	4,76	4,78	0,4	4,79	4,78	0,3
-	+	0,45	0,43	4,7	5,36	5,37	0,2	4,19	4,20	0,3
o	-	4,45	4,81	7,5	1,02	1,64	37,8	4,53	3,55	27,6
o	o	4,08	4,58	10,9	1,89	2,09	9,6	4,03	3,33	21,0
o	+	3,36	3,63	7,4	3,54	3,49	1,4	3,10	2,88	7,6
+	-	7,74	7,30	6,0	0,28	0,60	53,3	1,98	2,10	5,7
+	o	7,18	6,79	5,7	0,83	1,11	25,2	1,98	2,10	5,5
+	+	5,63	5,59	0,7	2,44	2,50	2,4	1,93	1,91	0,8

Table 2.35: Mean source size, mean conveyor size and mean buffersize for simulation and iterative method, when $C = 10$, $N = 5$ and $1/\mu = 30$.

than 80% occurs. If the retrial time increases the relative error decreases. If we would choose $1/\gamma$ even bigger the results will get better and better. This can be explained by the fact that our assumption of the Arrival Theorem will only be good if the time to retrial is big, as we already expected.

Furthermore for large inter-arrival times we see that many totes are situated in the source, this means that the source is working all the time. This is not an interesting situation for our model and also for Vanderlande, because if the source is working all the time the threshold does not serve a useful purpose. The open model would already give good estimations for the utilization and throughput. In the first three rows of Table 2.35 we see that there are almost no totes in the source, so all of the totes are on the conveyor or in the zone. This situation is much more interesting for both the model and Vanderlande. A welcome bonus is that the results for these situations are good for all investigated performance measures.

Conclusion

We have investigated several scenarios to see for which values the iterative method is most accurate and also we have explained why the model is accurate (or not) for each specific situation. The accuracy of the model also depends on the choice of the other values that occur in the model, like the threshold and the buffersize. In general we can say that results will be good if the retrial time is big and/or the mean inter arrival time is small. For both variables we mean with big, relatively big to the mean pick time. The accuracy of the model is roughly sketched in Figure 2.22 for several buffersizes. If the area is lightest of color than the iterative method works well for all performance measures that are investigated, the next most lightest area is the area where the method works rather bad for the blocking probability and in some cases for the mean buffersize and finally the darkest area is where the method gives very bad results for the blocking probability. Also remember that the method in general gives good results for the utilization and throughput. We have also investigated the accuracy for several other values of the threshold C and the mean pick time $1/\mu$. These results give the same conclusions regarding the accuracy of the method and the same figures can be sketched as in Figure 2.22.

The overal summarizing conclusion we can state is that the method works very good if the mean inter arrival time is small *and* the retrial time is rather big. This is also the situation which is most

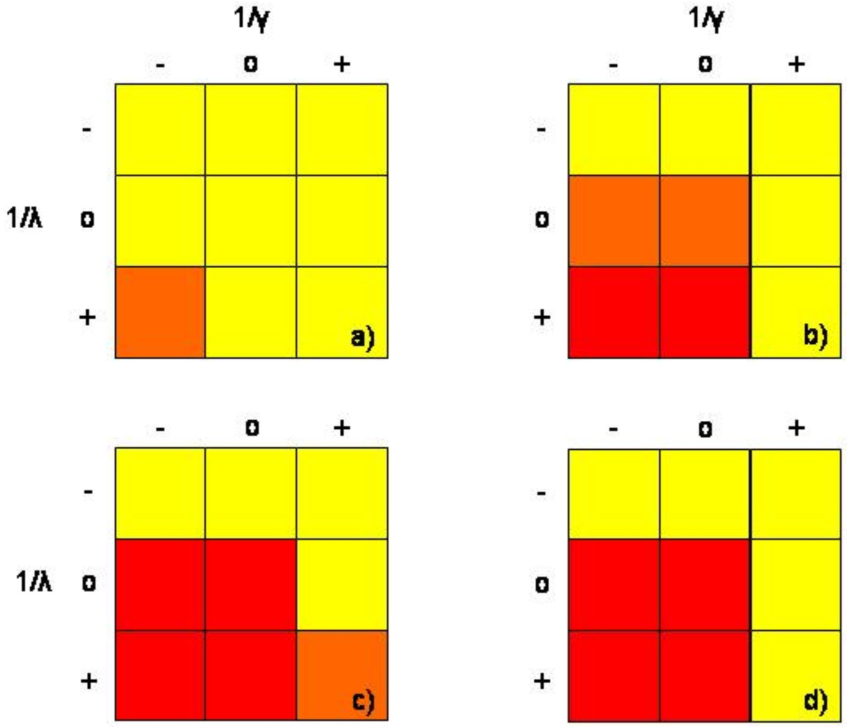


Figure 2.22: Sketch of the area of accuracy for the iterative method for buffersizes 1 (a), 2 (b), 5 (c) and 8 (d). When $C = 10$ and $1/\mu = 30$.

applicable to ZPS of Vanderlande, because for larger systems the retrial time will be much bigger than the mean pick time and the time until a new tote is generated will be small.

2.7 Summary and Conclusion

In this section we state a short summary and conclusion concerning the models and results presented in this chapter. We investigated a ZPS with one segment and only one zone. This is not very interesting for Vanderlande but gives us insight in the principle of ZPS and also several models to work with. We consider here all the models again, starting with the open model.

The open model is very accurate when the maximum number of totes in the system will not be reached. This is a situation which will almost never occur in real life, so this model is not very applicable to the real concept of a Zone Picking System. It can however give some nice results for some periods when the maximum is not reached and the supply is at a constant rate.

The finite source model is divided in two models; the single server source and the infinite server source model. The single server model describes the system when there is a constant supply of totes and this supply stops when a certain level is reached. The infinite server model also stops when the same level is reached but the supply of totes depends on the number of totes which

are in the segment. When there are many totes in the segment the supply will be low and vice versa. This last model seems to describe the situation when the segment is part of a larger (multi-segment) ZPS while the single server case describes the one segment ZPS.
Both models give reasonable results, except when C is very small, then the model does not even come close to the value of the simulation. Also we see that for very long pick times and long retrial times the model is not as accurate as one would wish.

The saturated model describes the system well when it is very busy. The total number of totes is always at its maximum, when a tote leaves the system a new tote immediately arrives in the system. This model is very accurate and gives good results for all parameters. We have also proved that it does not matter which distribution is assumed for the retrial time; the mean retrial time completely determines the performance of the system.

The iterative method is a method which works very good for the infinite server case and good for the single server case. For all different situations this method gives the same order of relative error. This method is most promising according to the results and probably can be extended to a segment with more zones. We have investigated more scenarios to gain more insight in the practicality of this method. For small retrial times and large inter-arrival times this method gives bad approximations. However when the time of a retrial is big and inter-arrival times are small, results are good also for the mean buffersize of the zone and probability of blocking at the zone. This is exactly the situation most applicable to ZPS of Vanderlande, especially when systems become bigger. This method will be the basis of our further research.

Chapter 3

One Segment, Multiple Zones

In this chapter we will broaden our scope to a system that consists of one segment with multiple zones. This is of much more interest, because within Vanderlande Industries a segment with just one zone will not be presented to a customer as a solution. In such a segment a tote travels on the conveyor towards the first zone it has to visit, if it can enter the zone it will do that otherwise the tote will continue his way on the conveyor. The tote travels to his second destination, if it has one, and there the same principle will hold. At the end of the segment the weight check will determine if a tote has visited all its zones and if not it will be recirculated and it will try to visit all the zones he missed in the first tour. Then the principle repeats itself until the tote has visited all the zones and it will leave the segment. In Figure 3.1 this system is presented schematically with four zones and a weight check at the end of the segment. Here several zone types are included. As said earlier we will only regard the single sided 1 stations, stated at the bottom of the figure.

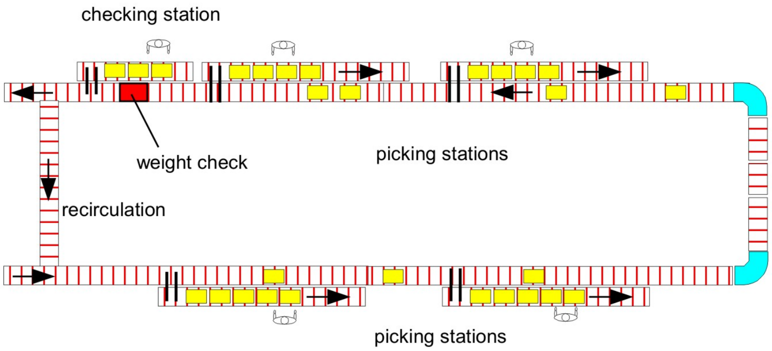


Figure 3.1: A segment with 4 zones, a weight check and recirculation

3.1 Introduction

This system is much more complex than the lay-out of one segment with one zone, because there are several extra things we have to take into account compared to the one zone case. One of these things is that totes can have more and different destinations. In the previous model (with one zone) this could not be the case. Each tote had to travel to the same and only zone. The order in which a tote visits all the zones is not pre-determined, for example the first zone a tote really visits could be the third one he needs to visit. Also a recirculation could occur because a tote has missed one zone, but also because he missed several zones. These situations did not occur when we had only one zone, there the order was pre-determined and also a recirculation only occurred when the only zone was fully occupied at arrival.

In this chapter first the model will be extended to a model for a system of one segment and two zones. Different types of totes will be distinguished and these types are the states of a Markov chain, where the transition probabilities depend on the blocking probabilities. The model is compared with results of the simulation for both single server source as infinite server source in section 3.2.4 and 3.2.5.
In the second part of this chapter the model is extended to a system of one segment and multiple zones. The Markov chain of different tote types is extended to the more general case. A mean value analysis is presented to cope with long calculation times we encounter. In section 3.3.4 the simulation is presented for multiple zones in one segment. The newly introduced analysis is implemented in a Java application, which will be discussed in section 3.3.5. Furthermore results are generated and especially a real simulation project is considered. Finally this chapter is summarized and some conclusions are stated.

3.2 One Segment, Two Zones

The first logical step to extend the previous system, is looking at a system that consists of one segment with two zones. This is already a much more complex system as we discussed earlier. There are more variables which we can vary, like the buffersizes of the two zones, the working speed of the two workers and the speed of the conveyor pieces. Furthermore totes may have to travel only to zone 1, only to zone 2 or to both zones. This is an important aspect which also has an influence on the principle of workload balancing and so has to be taken into account. In the previous chapter several models where suggested for the one zone case. One of the best methods used there was the iterative method, for both single server and infinite server source. This method will in this section be extended to a model for one segment with two zones.

3.2.1 Iterative Method

In this section we will describe the iterative method for the general two zones case. This method will work good if the probability of blocking at the zones is small, we will come back to this later. First we have to make several assumptions and introduce some notation:

1. The maximum amount of totes in the segment is denoted as C.

2. The station of the weight check will be omitted.

3. Totes arrive according to a Poisson process with rate λ if totes are allowed in the system, for the single server source. For the infinite server source; each tote in the source stays an exponential amount of time with rate λ.

4. The operators at zone i work on a tote an exponential amount of time with mean $1/\mu_i$, $i = 1, 2$.

5. The buffersize of zone i is denoted as $N_i, i = 1, 2$.

6. We will model the conveyor in two pieces; one piece from the first zone to the second zone and the other piece is describes the rest of the conveyor. Each conveyor piece is described as an infinite server with exponential rate γ.

These are the main assumptions which are more or less the same as we have seen for the one zone case. There are several variables which need to be introduced to fully describe the system. These variables depend on the type of system we are concerning. First we will consider the case that a tote either travels to only zone 1 or only zone 2; we can decompose this system. After that we will consider the system where also totes can visit both zones.

Either zone 1 or zone 2

In this part we will discuss the system when totes visit either zone 1 or zone 2 and if the buffer is fully occupied they will return for another attempt, otherwise they will leave the system and enter the source. In this model the system can be represented as can be seen in Figure 3.2. In general we shall assume that a fraction p of the totes will visit zone 1 and $1-p$ zone 2. In reality workload balancing is applied to this system, so that we have

$$\frac{p}{\mu_1} = \frac{1-p}{\mu_2} \text{ for } p \in (0,1). \tag{3.1}$$

We will consider here the general case, because the workload can not exactly be balanced and little fluctuations can occur. The model can be schematically described as Figure 3.2, where we have also stated the probability of blocking of a zone. In the same manner as for the 1 zone case, we apply jump-over blocking to this system. This means that when a tote arrives at one of the zones, it either is blocked because the buffer is full and it jumps over this zone or it is served in the zone. After 'visiting' a zone the tote is routed to the source or again to the zone, with a certain probability. Note that this probability is independent of whether a tote jumped over the zone or not, i.e. even if a tote is served at a zone it can be recirculated to that zone. In reality this

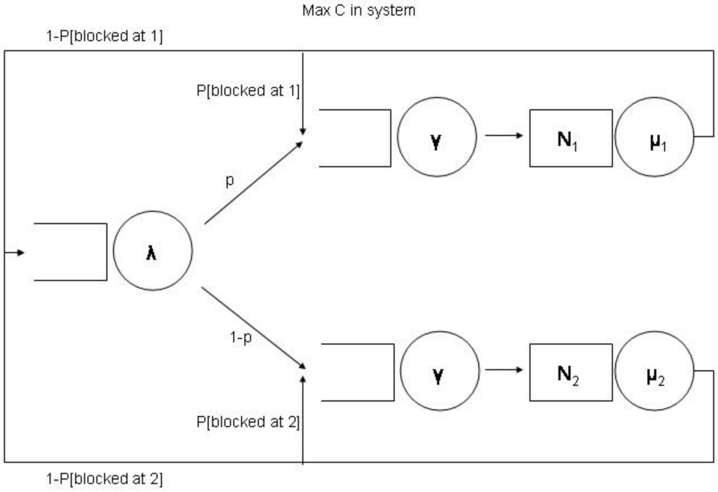


Figure 3.2: A closed queueing model of a segment with 2 zones, with source and totes visit either zone 1 or zone 2

will not be the case, but in this model we approximate the blocking behavior by these routing probabilities.

Before we will state the limiting probabilities of this system, which have a product form according to the jump-over blocking [6], we will introduce some notation for abbreviation purposes:

$$b^{(1)} = P(\text{blocked at } 1), \tag{3.2}$$
$$b^{(2)} = P(\text{blocked at } 2). \tag{3.3}$$

Because of the jump-over blocking this system has the following product form solution:

$$P(n_1,n_2,n_3,n_4,n_5) = d(\frac{1}{\lambda})^{n_1}\frac{1}{n_2!}(\frac{p}{(1-b^{(1)})\gamma})^{n_2}(\frac{p}{(1-b^{(1)})\mu_1})^{n_3}\frac{1}{n_4!}(\frac{1-p}{(1-b^{(2)})\gamma})^{n_4}(\frac{1-p}{(1-b^{(2)})\mu_2})^{n_5}, \text{ for single server source,} \tag{3.4}$$

$$P(n_1,n_2,n_3,n_4,n_5) = d\frac{1}{n_1!}(\frac{1}{\lambda})^{n_1}\frac{1}{n_2!}(\frac{p}{(1-b^{(1)})\gamma})^{n_2}(\frac{p}{(1-b^{(1)})\mu_1})^{n_3}\frac{1}{n_4!}(\frac{1-p}{(1-b^{(2)})\gamma})^{n_4}(\frac{1-p}{(1-b^{(2)})\mu_2})^{n_5}, \text{ for infinite server source,} \tag{3.5}$$

where d is the normalizing constant and n_1, n_2, n_3, n_4, n_5 denotes the number of totes at the source, the conveyor in front of the first zone, the first zone, the conveyor in front of the second zone and the second zone respectively. With these limiting probabilities we can determine every probability we want to know, including the blocking probability in each zone. These are the ones we need to know and we will again do this with an iterative method.

The principle will be the same as for the one zone case, so at first we will start with the case that there is no blocking. Then we can determine the new blocking probabilities for both zones with the known expressions for the limiting probabilities. This blocking probability for zone 1 is the probability that $n_3 = N_1$ and the system has $C-1$ totes in the system, because an arriving totes does not see itself. This can also be proven by the same reasoning as in the Arrival Theorem (Theorem 2) for the one zone case. The new blocking probability for zone 2 equals the probability that $n_5 = N_2$ and there are $C-1$ totes in the system. This can be repeated until the blocking probabilities do not change anymore. The algorithm is given in Algorithm 4.
Eventually we have the limiting probabilities of the system so we can calculate any performance measure. The most interesting, the utilization, will be the same for both of the zones when the workload is balanced. We need to calculate this only once, at the end of the algorithm, to obtain the utilization of both zones, by using the formula:

$$\rho_1 = 1 - \sum_{n_5=0}^{N_2}\sum_{n_4=0}^{C-n_5}\sum_{n_2=0}^{C-n_4-n_5} P(C-n_2-n_4-n_5, n_2, 0, n_4, n_5), \tag{3.6}$$
$$\rho_2 = 1 - \sum_{n_3=0}^{N_1}\sum_{n_4=0}^{C-n_3}\sum_{n_2=0}^{C-n_3-n_4} P(C-n_2-n_3-n_4, n_2, n_3, n_4, 0). \tag{3.7}$$

Where $b^{(1)}$ and $b^{(2)}$ are the last calculated $b_i^{(1)}$ and $b_i^{(2)}$ of Algorithm 4. This algorithm converges because of the same argument as is given in Theorem 3, a proof will not be given here.

Algorithm 4 Iterative Method for 2 Zones (1 visit)

```
1. Set $i = 0$ and $b_0^{(1)} = 0$, $b_0^{(2)} = 0$.
2. Set $i = i + 1$ and calculate new blocking probabilities for zone 1 and zone 2:
    $$b_i^{(1)} = \sum_{n_5=0}^{\min[C-1-N_1,N_2]} \sum_{n_4=0}^{C-1-n_5} \sum_{n_2=0}^{C-1-n_4-n_5} P(C-1-n_2-N_1-n_4-n_5, n_2, N_1, n_4, n_5),$$
    $$b_i^{(2)} = \sum_{n_3=0}^{\min[C-1-N_2,N_1]} \sum_{n_4=0}^{C-1-n_3} \sum_{n_2=0}^{C-1-n_3-n_4} P(C-1-n_2-n_3-n_4-N_2, n_2, n_3, n_4, N_2),$$
    where $P(n_1, n_2, n_3, n_4, n_5)$ is given by one of the product forms ((3.4) or (3.5)), depending on the use of a single or infinite server source respectively and where we use $b^{(1)} = b_{i-1}^{(1)}$ and $b^{(2)} = b_{i-1}^{(2)}$.
3. Repeat step 2 until $|b_i^{(1)} - b_{i-1}^{(1)}| < \epsilon$ and $|b_i^{(2)} - b_{i-1}^{(2)}| < \epsilon$.
```

Both zones

Next, we will consider the case that totes can visit both zones. In this situation the model is not as easy as before, because different types of totes have to be taken into account. These types depend on which zones the tote needs to visit, we distinguish three types: a tote has to visit only zone 1, only zone 2 or both zones. The assumptions that are stated at the beginning of this section also hold for this model. There are several variables which need to be introduced to fully describe the system:

P(no service at zone 1): the probability that an arbitrary tote does not need to visit zone.

P(no service at zone 2): the probability that an arbitrary tote does not need to visit zone.

P(retrial): the probability that an arbitrary tote needs to make a recirculation due to the fact that one of the zones was fully occupied.

The first two variables are clear from definition and could be based on known data from the customer. The last variable of the three introduced variables is the one of main interest and therefore let us take a closer look at this probability

$$\begin{aligned} P(\text{retrial}) &= P(\text{visit zone 1}) \cdot P(\text{blocked at zone 1}) \\ &+ P(\text{visit zone 2}) \cdot P(\text{blocked at zone 2}) \\ &+ P(\text{visit both zones}) \cdot P(\text{blocked at either one of the zones (or both)}) \\ \\ &= P(\text{visit zone 1}) \cdot P(\text{blocked at zone 1}) \\ &+ P(\text{visit zone 2}) \cdot P(\text{blocked at zone 2}) \qquad (3.8) \\ &+ P(\text{visit both zones}) \cdot [P(\text{blocked at zone 1}) + P(\text{blocked at zone 2}) \\ &- P(\text{blocked at both zones})]. \end{aligned}$$

If there are no totes which will visit both zones this model should give (approximately) the same answers as the model we discussed just before. We use here approximately, because the conveyor is modelled in different manners in the two methods. If a tote visits either one of the zones the conveyor is modelled as one piece, while if a tote can visit both zones the conveyor is modelled as two pieces.

The probability that a tote is blocked at either one of the zones (or both) is given by the expression in equation 3.8. This is because of the 'sum rule' for probabilities; for this let us define A as the event that a tote is blocked at zone 1 and B as the event that a zone is blocked at zone 2. The probability that a tote is blocked by either one of the zones (or both) is then given by:

$$P(A \cup B) = P(A) + P(B) - P(A \cap B), \tag{3.9}$$

which in words is the same as given before.

The model for the system of two zones can schematically be described as Figure 3.3, where also the newly introduced probabilities are included. The stations with μ_i's are modelled as a single server and the stations with γ_i's are infinite servers. The source, the station with λ, can be either single server or infinite server, depending on the choice of the model. Here you can see that with a probability of $P(\text{no service at zone } 1)$ a tote will not go in zone 1 and will stay on the conveyor. Also the probability of a retrial (or recirculation) can be seen. The blocking probabilities are not included, but if a zone is fully occupied the tote will 'jump' over the zone and will travel on the conveyor piece behind the zone, the so-called jump-over blocking. The routing probability is again independent of whether a tote is served or blocked at the zone.

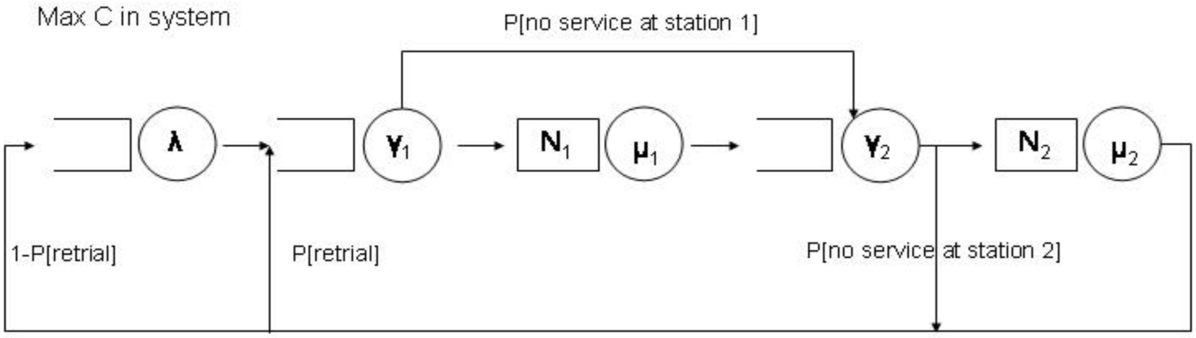


Figure 3.3: A closed queueing model of a segment with 2 zones

These blocking probabilities are the ones we need to find and with these probabilities also the probability of retrial is given by (3.8). The calculation of the blocking probabilities will again be done with an iterative method. The principle will be the same as is done for the one zone case, at first the blocking probabilities will be set to 0, so also the probability of a retrial will be 0. We know the limiting probabilities, because these follow a product form according to the approach of van Dijk [6]:

$$\begin{aligned} P(n_1,n_2,n_3,n_4,n_5) &= d(\frac{1-p^{(r)}}{\lambda})^{n_1}\frac{1}{n_2!}(\frac{1}{\gamma_1})^{n_2}(\frac{p_1+p_3}{\mu_1})^{n_3}\frac{1}{n_4!}(\frac{1}{\gamma_2})^{n_4}(\frac{p_2+p_3}{\mu_2})^{n_5}, \\ & \quad \text{for single server source,} \qquad (3.10) \\ P(n_1,n_2,n_3,n_4,n_5) &= d\frac{1}{n_1!}(\frac{1-p^{(r)}}{\lambda})^{n_1}\frac{1}{n_2!}(\frac{1}{\gamma_1})^{n_2}(\frac{p_1+p_3}{\mu_1})^{n_3}\frac{1}{n_4!}(\frac{1}{\gamma_2})^{n_4}(\frac{p_2+p_3}{\mu_2})^{n_5}, \\ & \quad \text{for infinite server source,} \qquad (3.11) \end{aligned}$$

where d is the normalizing constant and n_1, n_2, n_3, n_4, n_5 denotes the number of totes at the source, the first piece of the conveyor, the first zone, the second conveyor piece and the second zone respectively. Here we also, for abbreviation purposes, introduced the notations:

$$\begin{aligned} p_1 &= P(\text{service at only zone 1}), \\ p_2 &= P(\text{service at only zone 2}), \\ p_3 &= P(\text{service at both zones}), \\ p^{(r)} &= P(\text{retrial}), \end{aligned}$$

where the first three are given by the data of the order of the customer. With the limiting probabilities we now can determine the blocking probabilities for each zone, for zone 1 this is the probability that $n_3 = N_1$ and for zone 2 this is the probability that $n_5 = N_2$. Note here that again the limiting probabilities with one tote less in the system should be chosen, as a consequence of Theorem 2 which in the same manner can be applied to a system with two zones.

Once we have obtained the new blocking probabilities, we can recalculate the probability that a tote will have to make a recirculation using (3.8). This can be repeated until the blocking probabilities and so also the retrial probability does not change anymore. We assume here that the blocking probabilities are independent of each other and we use the same notation for the blocking probabilities as in the previous model. This assumption can be explained by the fact that in each iteration step or circulation we will look at two places, in front of each zone, if a tote is blocked or not. Here we forget what is happened to this tote until he arrived at this point. This implies that the probability of being blocked at both zones is the product of the two blocking probabilities, this will be used in the algorithm. The algorithm is stated in Algorithm 5.

Algorithm 5 Iterative Method for 2 Zones (general)

```
1. Set $i = 0$ and $b_0^{(1)} = 0$, $b_0^{(2)} = 0$ so $p_0^{(r)} = 0$.
2. Set $i = i + 1$ and calculate new blocking probabilities for zone 1 and zone 2:
   $$b_i^{(1)} = \sum_{n_5=0}^{N_2} \sum_{n_4=0}^{C-1-N_1-n_5} \sum_{n_2=0}^{C-1-N_1-n_5-n_4} P(C-1-n_2-N_1-n_4-n_5, n_2, N_1, n_4, n_5),$$
   $$b_i^{(2)} = \sum_{n_3=0}^{N_1} \sum_{n_4=0}^{C-1-N_2-n_3} \sum_{n_2=0}^{C-1-N_2-n_3-n_4} P(C-1-n_2-N_2-n_3-n_4, n_2, n_3, n_4, N_2),$$
   where $P(n_1, n_2, n_3, n_4, n_5)$ is given by one of the product forms ((3.10) and (3.11)), depending on the use of a single or infinite server source respectively and where we use $p^{(r)} = p_{i-1}^{(r)}$.
3. Calculate the new probability of a retrial:
   $$p_i^{(r)} = p_1 \cdot b_i^{(1)} + p_2 \cdot b_i^{(2)} + p_3 \cdot (b_i^{(1)} + b_i^{(2)} - b_i^{(1)} \cdot b_i^{(2)})$$
4. Repeat step 2 until $|b_i^{(1)} - b_{i-1}^{(1)}| < \epsilon$ and $|b_i^{(2)} - b_{i-1}^{(2)}| < \epsilon$.
```

Eventually we have the limiting probabilities of the system so we can calculate any performance measure. The most interesting, the utilization, will be the same for both of the zones when the workload is balanced. To calculate the utilization of both zones, we can use (3.6) and (3.7). Algorithm 5 is implemented in Mathematica 5.2 and eventually the utilization is calculated, other performance measures are also possible. This algorithm will converge, but a short sketch of the proof will be given. The proof for the one zone case was already given in Theorem 3, this proof will work in a similar way using induction.

Let us consider the blocking probabilities at both zones and let us investigate if $(b_i^{(1)}, b_i^{(2)})$ form a monotonically increasing sequence. The induction basis is again obvious. The induction step works exactly the same as for the one zone model, the key observation is here that if $b_{i+1}^{(1)} \geq b_i^{(1)}$ and $b_{i+1}^{(1)} \geq b_i^{(1)}$ then we also have:

$$\begin{aligned} p_{i+1}^{(r)} &= p_1 b_{i+1}^{(1)} + p_2 b_{i+1}^{(2)} + p_3 (b_{i+1}^{(1)} + b_{i+1}^{(2)} - b_{i+1}^{(1)} b_{i+1}^{(2)}) \\ &\geq p_1 b_i^{(1)} + p_2 b_i^{(2)} + p_3 (b_i^{(1)} + b_i^{(2)} - b_i^{(1)} b_i^{(2)}) \\ &= p_i^{(r)}. \end{aligned}$$

Where we used that $f(x, y) := x + y - xy$, is an increasing function for $x, y \in [0, 1]$. This implies that $1 - p_{i+1}^{(r)} \leq 1 - p_i^{(r)}$ and hence we know by the same reasoning as in Theorem 3 that the blocking probabilities form a monotonically increasing sequence. Because the probabilities are limited, Algorithm 5 converges.

The order of the algorithm is just like the 1 zone model given by the state space over which we have to sum all the limiting probabilities, because in each iteration the normalizing constant needs to be calculated:

$$\text{Order algorithm} = \mathcal{O}(N_1 N_2 C^2). \tag{3.12}$$

As can be seen we assumed here that the fraction of types of totes stays constant, which is the case if blocking probabilities are small. In fact this fraction will differ depending on the probabilities that totes are blocked and the fraction of types of totes that are given by the specific order. This observation will be taken into account in the next section.

3.2.2 Iterative Method with Markov Chain

In this section we will discuss an improvement of the iterative method, as mentioned before. When using the iterative method of the previous section we assumed that the proportion of types of totes is constant in time and does only depend on the proportion of totes which is released at the beginning of the main conveyor. This proportion determines the probability to jump over a certain zone as can be seen in Figure 3.3. The model of Figure 3.3 is also applicable to this section only with a small adjustment. Here we will also consider three types of totes.

A type of a tote can change into another type because the tote has been served. With a certain probability this occurs and these probabilities depend on the blocking probabilities. The blocking probabilities were calculated iteratively so it would be logical to also do this with the transition probabilities between tote types. It is easy to see that if the blocking probabilities are very small the normal iterative method must perform approximately as good as this new method, but if the blocking probabilities are big this method should be better because it takes blocking into account when considering the proportion of tote types. Let us first define the different types of totes:

Type 1: The tote has to go to zone 1.

Type 2: The tote has to go to zone 2.

Type 3: The tote has to go to both zones.

Earlier we said that the transition probabilities depend on the blocking probabilities, which can be explained with a simple example. Say a tote is of type 1 then it will still be of this type if this tote is blocked at zone 1. This happens with the blocking probability which is calculated during the iteration. A type 1 tote can change into a any type of tote if the tote is served at zone 1, which happens with 1 minus the probability that the tote is blocked.

A special case of the transitions occurs when a tote is served in one circulation. If this is the case a tote will change in a type with a probability which depends on the proportion of tote-types of the order. If the proportion is equally distributed over the three different types then this proportion is $\frac{1}{3}$ for each type. For example consider a type 1 tote, this will change in a type 2 tote with the probability that it is served times $\frac{1}{3}$. The transitions between tote types describe a Markov chain, with the transition probabilities described as above. The general Markov model is given in Figure 3.4, to understand this proces we first have to introduce some notation:

$$\begin{aligned} b^{(1)} &:= P(\text{The tote has been blocked at zone } 1). \\ b^{(2)} &:= P(\text{The tote has been blocked at zone } 2). \\ p_i &:= \text{The proportion of totes of type } i = 1, 2, 3 \text{ released, which is constant in time.} \end{aligned}$$

Let us define π_i as the limiting probability that the Markov chain is in state i, we can state the

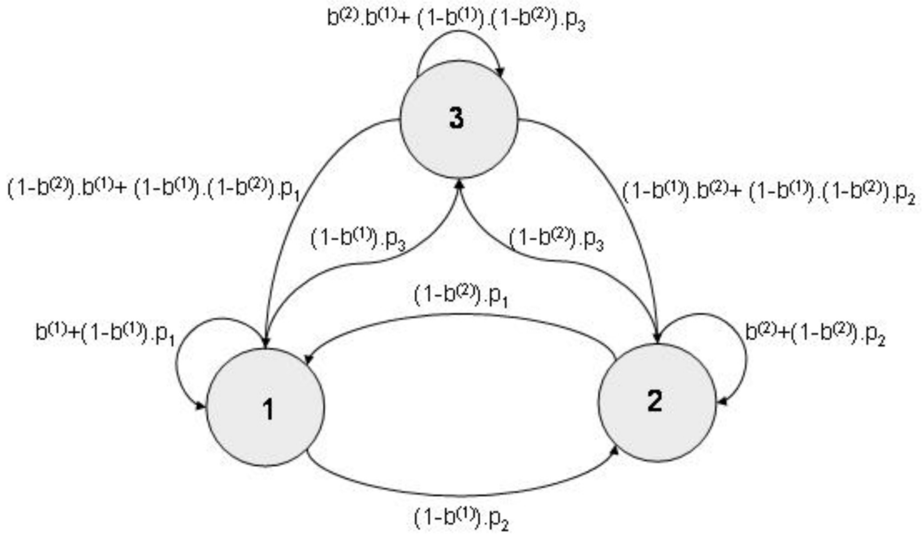

$b^{(2)}.b^{(1)}+ (1-b^{(1)}).(1-b^{(2)}).p_3$
3
$(1-b^{(2)}).b^{(1)}+ (1-b^{(1)}).(1-b^{(2)}).p_1$
$(1-b^{(1)}).b^{(2)}+ (1-b^{(1)}).(1-b^{(2)}).p_2$
$(1-b^{(1)}).p_3$
$(1-b^{(2)}).p_3$
$(1-b^{(2)}).p_1$
$b^{(1)}+(1-b^{(1)}).p_1$
$b^{(2)}+(1-b^{(2)}).p_2$
1
2
$(1-b^{(1)}).p_2$

Figure 3.4: The transition probabilities of the Markov chain

following balance equations using Figure 3.4:

$$\begin{aligned}
\pi_1 &= \big(b^{(1)} + (1-b^{(1)})p_1\big)\pi_1 + \big((1-b^{(2)})p_1\big)\pi_2 + \big((1-b^{(2)})b^{(1)} + (1-b^{(1)})(1-b^{(2)})p_1\big)\pi_3, \\
\pi_2 &= \big((1-b^{(1)})p_2\big)\pi_1 + \big(b^{(2)} + (1-b^{(2)})p_2\big)\pi_2 + \big((1-b^{(1)})b^{(2)} + (1-b^{(1)})(1-b^{(2)})p_2\big)\pi_3, \\
\pi_3 &= \big((1-b^{(1)})p_3\big)\pi_1 + \big((1-b^{(2)})p_3\big)\pi_2 + \big(b^{(1)}b^{(2)} + (1-b^{(1)})(1-b^{(2)})p_3\big)\pi_3.
\end{aligned}$$

We can solve this set of equations using the normalizing equation; the probabilities sum up to 1. We can express all probabilities in terms of π_3 and with the normalizing constant we have an expression for π_3 and hence for the other probabilities. This can be done algebraically but we can also use Mathematica 5.2. In the general case the algebraic expressions are horrible, but for given proportions of tote types at the release point the expressions look more friendly. We shall not do this here but with simple algebra this can be done.

As said earlier these transition probabilities will be calculated each iteration just like the blocking probabilities. With these transition probabilities we can determine the probability that a tote does not have to visit a zone, for example for zone 1 this probability is given by the limiting probability π_2 and for zone 2 by π_1.

The iterative method with the use of a Markov chain works in the same way as the normal iterative method, but in each iteration the transition probabilities also need to be determined as can be seen in Algorithm 6.

The utilization of both zones can be calculated in the same manner as was done before, also other performance measures can be determined because we have an expression for the limiting probabilities. The proof of convergence of Algorithm 6 will not be given here, but it will work in a similar way as before. The order of this algorithm is also given by (3.12).

Algorithm 6 Iterative Method with Markov Chain for 2 Zones

1. Set $i = 0$ and $b_0^{(1)} = 0$, $b_0^{(2)} = 0$ so $p_0^{(r)} = 0$.
2. Set $i = i + 1$ and calculate new blocking probabilities for zone 1 and zone 2:

$$\begin{aligned} b_i^{(1)} &= \sum_{n_5=0}^{N_2} \sum_{n_4=0}^{C-1-N_1-n_5} \sum_{n_2=0}^{C-1-N_1-n_5-n_4} P(C-1-n_2-N_1-n_4-n_5, n_2, N_1, n_4, n_5), \\ b_i^{(2)} &= \sum_{n_3=0}^{N_1} \sum_{n_4=0}^{C-1-N_2-n_3} \sum_{n_2=0}^{C-1-N_3-n_3-n_4} P(C-1-n_2-N_2-n_4-n_3, n_2, n_3, n_4, N_2), \end{aligned}$$

   where $P(n_1, n_2, n_3, n_4, n_5)$ is given by one of the product forms ((3.10) and (3.11)), depending on the use of a single or infinite server source respectively and where we use $p^{(r)} = p_{i-1}^{(r)}$ and the new values for the proportion of totes. These are given in the following steps of the iteration.

3. Calculate the new transition probabilities using the new blocking probabilities
4. Calculate the new proportion of each type of totes:

$$\begin{aligned} p_1 &= \pi_1, \\ p_2 &= \pi_2, \\ p_3 &= \pi_3. \end{aligned}$$

5. Calculate the new probability of a retrial:

$$p_i^{(r)} = p_1 \cdot b_i^{(1)} + p_2 \cdot b_i^{(2)} + p_3 \cdot (b_i^{(1)} + b_i^{(2)} - b_i^{(1)} \cdot b_i^{(2)})$$

6. Repeat step 2 until $|b_i^{(1)} - b_{i-1}^{(1)}| < \epsilon$ and $|b_i^{(2)} - b_{i-1}^{(2)}| < \epsilon$.

Iterative Methods Revisited

In the last two sections we have stated 3 different iterative methods; the iterative method when a tote visits only one of the zones, the iterative method when a tote can also visit both zones and the iterative method with a Markov chain. These methods have a lot in common and are different in subtle things. In certain special cases the different methods should give the same answer and in this part we shall discuss these similarities. Also we will say something about what we expect for the accuracy of the methods.

First we shall compare the two iterative methods with no Markov chain; these models are the same if in the second method there are no totes that can travel to both zones. The only difference here is that in the second model the conveyor is modelled as two parts, this can have some influence on the performance of the system. These two methods will be good when the blocking probabilities are low for each zone, because these methods do not take into account that the proportion of types of totes change when a tote is blocked.

The iterative method with Markov chain will give better results compared to the other two methods, when there is a lot of blocking at the zones. On the other hand if the probability of blocking is very small, this method will give same results as the iterative method for visiting both zones.

We come back to these predictions in sections 3.2.4 and 3.2.5.

3.2.3 Simulation

To verify if this algorithm works well for the two zone case, we have to compare it with a simulation. This simulation is based on the simulation written for the one zone model and is also written in Automod. We will model the segment as two individual conveyor pieces with at each end a zone with a finite buffer. If a tote comes from zone 1 it will be directed to the second conveyor piece, served or not served. At the second and last zone a tote will be send back to the beginning of conveyor piece 1 in case of a retrial, otherwise the tote will leave the system. The simulation again does not take gaps between totes into account.

The hardest part is to keep track which type of tote is on the conveyor, we have divided them in four classes and these numbers are given to the tote as an attribute in Automod:

0) The tote does not have to go to any zone anymore. This will be the case when a tote is on the second piece of the conveyor and was already served by zone 1.

1) The tote has to go to zone 1.

2) The tote has to go to zone 2.

3) The tote has to go to both zones.

During the total trip a tote will change from class when he has visited a zone. For example a tote that has to visit both zones will change to 'class 1' if its first zone visit is zone 2. After a certain amount of time, (with or without circulations), this tote will be served at zone 1 and the tote will change to 'class 0'. If now the tote arrives at zone 2, this tote will be sent to die (or out of the system), because he has visited all zones.

In the simulation we will generate the last three types of totes and the proportion of these types can be adjusted to every desired value. The release of new totes will be done according to a single or infinite server, with a threshold of C totes in the total segment (so the two conveyor parts and the two buffers).

Validation simulation

This simulation will also be validated with two special cases (for the single source case), first we consider the case that N_i and C are chosen very big. Then we have a tandem queue. In this simulation we generate an equal proportion of the three types of totes, so that means that $\frac{2}{3}$ of the generated totes needs to go to zone 1, this also holds for the second zone. The other parameters are chosen: $1/\lambda = 36$, $1/\mu_1 = 1/\mu_2 = 30$ and finally $1/\gamma_1 = 1/\gamma_2 = 100$. According to the theory [2] the utilization should be $\rho = \frac{2}{3} \cdot \frac{30}{36} \approx 0,5556$ and for the throughput per hour at the end of the conveyor $\delta = 120 \cdot \frac{30}{36} = 100$. The results for utilization and throughput per hour are given in Table 3.1 and as we can see the simulation gives results as should be according to the theory.

	Simulation	Theory	Relative Error $\Delta\%$
ρ	0,555	0,556	0,01
δ	100,6	100	0,6

Table 3.1: Validation of simulation results for utilization (ρ) and throughput per hour (δ)

Secondly we will consider the case that the buffersizes are infinite, so no blocking can occur. We can also see this system as the case that $N = C$. Then this system is a closed network and has also a product form solution. In [1] a mean value analysis for exponential closed networks is given, this mean value analysis was also done for the 1 zone case and only the results will be used to check the simulation results. When N and C are chosen very big, we will get the same results as the case just considered. The results are given in Table 3.2 and we can see that they are very accurate. These special cases are enough to believe that the simulation works well and we can generate results to compare them with the iterative method.

$N = C$	Simulation	Theory	$\Delta\%$
1	0,072 (±0,006)	0,072	0,61
2	0,136 (±0,008)	0,141	3,65
5	0,316 (±0,011)	0,318	0,77
10	0,487 (±0,014)	0,495	1,73
20	0,559 (±0,018)	0,555	0,66

Table 3.2: Simulation results for utilization compared with results from queueing theory for zone 1

Scenarios

In the next section we will discuss the results of the iterative methods. In the previous chapter we changed all parameters to see what happened to the utilization of the operator. This was done to get some feeling for the problem and also to verify the model.

That is why for the two zone case we will run the scenarios which will give a good impression of the accuracy of the model. We will vary 4 most important parameters; the buffersize, the maximum number of totes allowed in the system, the mean inter-arrival time and the quotient of the mean time on the conveyor and the mean pick time of an operator. These 4 variables will take on several values leading to in total 24 scenarios we need to run. First we will state the standard values of the other parameters:

1. Mean pick time: $1/\mu_1 = 1/\mu_2 = 30$ seconds
2. Proportion of types of totes will be equally distributed, i.e. $p_i = \frac{1}{3}$ for $i = 1, 2, 3$.

The other parameters will take on several values, these are given in Table 3.3. Here $1/\gamma = 1/\gamma_1 + 1/\gamma_2$, this is the total mean time on the conveyor. However for the infinite source model we have to take other values for λ to obtain interesting results, we have chosen $1/\lambda = 100, 250$ respectively.

Variables	Values
$1/\lambda$	$4, 10$
C	$10, 25$
μ_i/γ	$4, 6$
$N_1 = N_2$	$1, 2, 5$

Table 3.3: The values of the variables for the 24 different scenarios

We will also consider some other scenarios where the workload is unbalanced and the proportion of types of totes is not equally distributed. This will be applied to the most interesting scenarios. To obtain an unbalanced workload we just alter the mean pick time of the first zone to $1/\mu_1 = 32$, the proportion is equally distributed so this will generate an unbalanced system.
Furthermore we will consider two extra scenarios where the proportion is not equally distributed. These 2 cases will be generated by taking on the values:

Case 1: $p_1 = p_2 = 0.25$ and $p_3 = 0.5$, and

Case 2: $p_1 = p_2 = 0.45$ and $p_3 = 0.1$.

First we will generate results for the single server source, followed by the infinite server source. In the next two sections we will state the results for these two cases respectively.

3.2.4 Results Single Server

In this section we present the results when the source is a single server. This represents the situation when in a segment the supply of new totes occurs with a constant rate, until of course the maximum number of totes is reached in which case the supply stops. We will run the scenarios and present the results in this section only for the iterative method with Markov chain, because the iterative method without Markov chain will always give worse results.

We will show this using one example where we vary the buffersize, the other variables are chosen to be $1/\lambda = 4$, $C = 10$ and $\mu/\gamma = 4$. The results we will present for this small example will be the utilization of the first zone and the total throughput, which are the most important performance measures, and these are presented in Table 3.4. The superscript sim stands for simulation results, it for iterative method results *with* Markov chain and itw for iterative method *without* Markov chain.

N	ρ_1^{sim}	ρ_1^{it}	ρ_1^{itw}	δ^{sim}	δ^{it}	δ^{itw}
1	0,576	0,580	0,600	103,3	104,4	107,9
2	0,726	0,741	0,758	130,9	133,3	136,5
5	0,802	0,818	0,822	144,8	147,2	148,0
10	0,803	0,803	0,803	144,4	144,5	144,5

Table 3.4: Iterative method with and without Markov chain, compared with simulation

We can see that the iterative method with Markov chain is always better than the method without, which is what we expect. Also we can see that the methods give more similar results when the buffersize increases, i.e. the blocking probability decrease. When there is no blocking ($N = C = 10$) the results of the methods are the same and give very good results when compared with the simulation.

For the 24 scenarios we will present the results for both zones and here the results also include the mean buffersize (L_{zi}) and the blocking probability ($b^{(i)}$) at zone i. Again the superscript sim denotes the results for the simulation and it for the iterative method with Markov chain. The absolute relative errors ($\Delta\%$) will be given in percentages.

The results for the 24 scenarios will be presented first, followed by the scenarios when the workload is unbalanced and finally we state the cases where the proportion of totes is not equally distributed. When considering the last 2 cases, workload unbalanced and different proportions, we only investigated the results when $C = 10$ and $1/\lambda = 4$. Finally we will discuss the results.

In the simulations we have run, the maximal error of the simulations for the utilization is 0.008, for the throughput per hour this error is 0.1, the mean buffersize has an error of maximum 0.1 and finally for the blocking probability this is 0.01.

Buffersize: $N = 1$

$1/\lambda$	C	μ/γ	$\rho_1^{it} = \rho_2^{it}$	ρ_1^{sim}	$\Delta\%$	ρ_2^{sim}	$\Delta\%$	δ^{it}	δ^{sim}	$\Delta\%$
4	10	4	0,580	0,576	0,8	0,570	1,7	104,4	103,3	1,0
4	25	4	0,778	0,753	3,3	0,753	3,2	140,0	135,9	3,0
4	10	6	0,486	0,484	0,3	0,486	0,0	87,4	87,3	0,2
4	25	6	0,702	0,685	2,5	0,683	2,8	126,4	123,2	2,6
10	10	4	0,573	0,564	1,6	0,563	1,8	103,1	101,8	1,4
10	25	4	0,774	0,743	4,2	0,739	4,8	139,4	133,3	4,6
10	10	6	0,480	0,480	0,0	0,479	0,3	86,5	86,4	0,0
10	25	6	0,699	0,670	4,3	0,674	3,7	125,8	121,7	3,4

$1/\lambda$	C	μ/γ	$L_{z1}^{it} = L_{z2}^{it}$	L_{z1}^{sim}	$\Delta\%$	L_{z2}^{sim}	$\Delta\%$	$b_{it}^{(1)} = b_{it}^{(2)}$	$b_{sim}^{(1)}$	$\Delta\%$	$b_{sim}^{(2)}$	$\Delta\%$
4	10	4	0,58	0,58	0,8	0,57	1,7	0,553	0,554	0,1	0,552	0,2
4	25	4	0,78	0,75	3,3	0,75	3,2	0,770	0,771	0,1	0,774	0,4
4	10	6	0,49	0,48	0,3	0,49	0,0	0,459	0,450	1,9	0,460	0,3
4	25	6	0,70	0,69	2,5	0,68	2,8	0,693	0,696	0,4	0,696	0,4
10	10	4	0,57	0,56	1,6	0,56	1,8	0,546	0,547	0,0	0,552	1,0
10	25	4	0,77	0,74	4,2	0,74	4,8	0,767	0,773	0,8	0,775	0,9
10	10	6	0,48	0,48	0,0	0,48	0,3	0,454	0,449	1,0	0,452	0,4
10	25	6	0,70	0,67	4,3	0,67	3,7	0,690	0,691	0,1	0,702	1,6

Buffersize: $N = 2$

$1/\lambda$	C	μ/γ	$\rho_1^{it} = \rho_2^{it}$	ρ_1^{sim}	$\Delta\%$	ρ_2^{sim}	$\Delta\%$	δ^{it}	δ^{sim}	$\Delta\%$
4	10	4	0,741	0,726	2,0	0,724	2,3	133,3	130,9	1,9
4	25	4	0,935	0,907	3,1	0,901	3,8	168,3	162,9	3,3
4	10	6	0,628	0,622	1,0	0,627	0,2	113,0	112,4	0,5
4	25	6	0,880	0,857	2,7	0,851	3,5	158,5	154,3	2,7
10	10	4	0,728	0,715	1,8	0,713	2,0	131,0	128,6	1,9
10	25	4	0,932	0,900	3,5	0,889	4,7	167,7	161,2	4,1
10	10	6	0,617	0,615	0,4	0,615	0,4	111,1	111,0	0,1
10	25	6	0,876	0,846	3,6	0,844	3,8	157,7	152,2	3,6

$1/\lambda$	C	μ/γ	$L_{z1}^{it} = L_{z2}^{it}$	L_{z1}^{sim}	$\Delta\%$	L_{z2}^{sim}	$\Delta\%$	$b_{it}^{(1)} = b_{it}^{(2)}$	$b_{sim}^{(1)}$	$\Delta\%$	$b_{sim}^{(2)}$	$\Delta\%$
4	10	4	1,14	1,12	2,2	1,11	2,5	0,359	0,368	2,4	0,370	2,9
4	25	4	1,65	1,58	4,4	1,56	5,5	0,703	0,708	0,7	0,704	0,2
4	10	6	0,91	0,90	1,1	0,91	0,0	0,250	0,247	1,0	0,255	1,9
4	25	6	1,49	1,44	3,4	1,42	4,6	0,592	0,597	0,9	0,597	0,9
10	10	4	1,11	1,09	1,9	1,09	2,1	0,346	0,356	2,8	0,355	2,6
10	25	4	1,64	1,56	5,0	1,53	6,9	0,696	0,708	1,7	0,697	0,2
10	10	6	0,89	0,89	0,4	0,89	0,6	0,241	0,241	0,1	0,240	0,5
10	25	6	1,47	1,41	4,6	1,40	4,9	0,585	0,591	1,1	0,597	2,1

Buffersize: $N = 5$

$1/\lambda$	C	μ/γ	$\rho_1^{it} = \rho_2^{it}$	ρ_1^{sim}	$\Delta\%$	ρ_2^{sim}	$\Delta\%$	δ^{it}	δ^{sim}	$\Delta\%$
4	10	4	0,818	0,802	1,9	0,806	1,5	147,2	144,8	1,7
4	25	4	0,996	0,965	3,2	0,953	4,5	179,2	172,4	4,0
4	10	6	0,707	0,708	0,1	0,701	0,8	127,3	127,1	0,1
4	25	6	0,981	0,944	3,9	0,941	4,2	176,5	169,9	3,9
10	10	4	0,798	0,788	1,2	0,787	1,4	143,6	141,9	1,3
10	25	4	0,995	0,959	3,8	0,955	4,2	179,1	171,7	4,3
10	10	6	0,690	0,685	0,8	0,692	0,2	124,3	123,8	0,4
10	25	6	0,978	0,946	3,3	0,941	3,9	176,0	170,7	3,1

$1/\lambda$	C	μ/γ	$L_{z1}^{it} = L_{z2}^{it}$	L_{z1}^{sim}	$\Delta\%$	L_{z2}^{sim}	$\Delta\%$	$b_{it}^{(1)} = b_{it}^{(2)}$	$b_{sim}^{(1)}$	$\Delta\%$	$b_{sim}^{(2)}$	$\Delta\%$
4	10	4	2,20	2,20	0,3	2,19	0,5	0,071	0,086	17,0	0,083	14,3
4	25	4	4,38	4,06	8,0	3,87	13,3	0,581	0,621	6,5	0,579	0,3
4	10	6	1,61	1,63	1,0	1,59	1,4	0,029	0,031	6,4	0,029	0,1
4	25	6	3,98	3,66	8,9	3,60	10,6	0,425	0,457	7,2	0,447	4,9
10	10	4	2,09	2,10	0,6	2,07	1,0	0,063	0,074	15,1	0,070	10,8
10	25	4	4,34	3,95	9,8	3,90	11,4	0,566	0,593	4,6	0,582	2,8
10	10	6	1,53	1,53	0,4	1,54	0,7	0,026	0,027	3,1	0,028	7,2
10	25	6	3,89	3,66	6,3	3,55	9,5	0,409	0,444	7,8	0,421	2,7

Unbalanced workload

N	μ_2/γ	ρ_1^{it}	ρ_1^{sim}	$\Delta\%$	ρ_2^{it}	ρ_2^{sim}	$\Delta\%$	δ^{it}	δ^{sim}	$\Delta\%$
1	4	0,606	0,598	1,4	0,568	0,558	1,8	102,2	100,9	1,3
2	4	0,772	0,754	2,4	0,724	0,709	2,1	130,3	127,6	2,2
5	4	0,851	0,834	2,0	0,798	0,786	1,5	143,6	140,9	1,9
1	6	0,509	0,506	0,6	0,477	0,475	0,6	85,9	85,2	0,9
2	6	0,658	0,649	1,4	0,617	0,615	0,3	111,0	109,9	1,0
5	6	0,740	0,736	0,6	0,694	0,690	0,5	124,9	124,7	0,2

N	μ_2/γ	L_{z1}^{it}	L_{z1}^{sim}	$\Delta\%$	L_{z2}^{it}	L_{z2}^{sim}	$\Delta\%$	$b_{it}^{(1)}$	$b_{sim}^{(1)}$	$\Delta\%$	$b_{it}^{(2)}$	$b_{sim}^{(2)}$	$\Delta\%$
1	4	0,61	0,60	1,4	0,57	0,56	1,8	0,580	0,579	0,2	0,541	0,546	0,9
2	4	1,21	1,18	2,7	1,11	1,08	2,2	0,397	0,404	1,8	0,340	0,355	4,3
5	4	2,43	2,40	1,0	2,08	2,08	0,1	0,092	0,106	13,5	0,061	0,078	21,5
1	6	0,51	0,51	0,6	0,48	0,48	0,6	0,482	0,480	0,4	0,450	0,452	0,4
2	6	0,97	0,96	1,6	0,89	0,89	0,1	0,276	0,275	0,5	0,240	0,251	4,4
5	6	1,77	1,76	0,5	1,55	1,55	0,2	0,038	0,041	7,6	0,027	0,028	4,5

Proportion: Case 1

N	μ/γ	$\rho_1^{it} = \rho_2^{it}$	ρ_1^{sim}	$\Delta\%$	ρ_2^{sim}	$\Delta\%$	δ^{it}	δ^{sim}	$\Delta\%$
1	4	0,595	0,586	1,6	0,593	0,5	95,2	94,0	1,3
2	4	0,761	0,745	2,1	0,748	1,8	121,7	119,3	2,0
5	4	0,842	0,825	2,1	0,823	2,4	134,8	132,2	1,9
1	6	0,502	0,503	0,0	0,503	0,1	80,4	80,7	0,3
2	6	0,653	0,646	1,1	0,653	0,0	104,6	104,0	0,6
5	6	0,742	0,745	0,3	0,736	0,8	118,7	118,5	0,2

N	μ/γ	$L_{z1}^{it} = L_{z2}^{it}$	L_{z1}^{sim}	$\Delta\%$	L_{z2}^{sim}	$\Delta\%$	$b_{it}^{(1)} = b_{it}^{(2)}$	$b_{sim}^{(1)}$	$\Delta\%$	$b_{sim}^{(2)}$	$\Delta\%$
1	4	0,60	0,59	1,6	0,59	0,5	0,569	0,562	1,2	0,574	0,9
2	4	1,19	1,16	2,3	1,16	1,9	0,382	0,386	0,8	0,393	2,7
5	4	2,36	2,37	0,4	2,31	2,4	0,086	0,105	18,2	0,098	12,5
1	6	0,50	0,50	0,0	0,50	0,1	0,475	0,462	2,9	0,475	0,1
2	6	0,96	0,95	1,5	0,96	0,1	0,272	0,265	2,4	0,278	2,3
5	6	1,78	1,81	1,7	1,75	1,9	0,039	0,042	6,4	0,038	2,3

Proportion: Case 2

N	μ/γ	$\rho_1^{it} = \rho_2^{it}$	ρ_1^{sim}	$\Delta\%$	ρ_2^{sim}	$\Delta\%$	δ^{it}	δ^{sim}	$\Delta\%$
1	4	0,552	0,539	2,4	0,545	1,2	120,4	118,4	1,7
2	4	0,702	0,691	1,7	0,687	2,3	153,3	150,3	2,0
5	4	0,770	0,758	1,6	0,762	1,1	168,1	165,6	1,5
1	6	0,456	0,450	1,3	0,448	1,7	99,4	98,8	0,6
2	6	0,581	0,577	0,7	0,571	1,8	126,8	126,1	0,5
5	6	0,643	0,641	0,3	0,644	0,2	140,3	139,7	0,5

N	μ/γ	$L_{z1}^{it} = L_{z2}^{it}$	L_{z1}^{sim}	$\Delta\%$	L_{z2}^{sim}	$\Delta\%$	$b_{it}^{(1)} = b_{it}^{(2)}$	$b_{sim}^{(1)}$	$\Delta\%$	$b_{sim}^{(2)}$	$\Delta\%$
1	4	0,55	0,54	2,4	0,55	1,2	0,525	0,528	0,5	0,537	2,2
2	4	1,06	1,04	1,5	1,04	2,2	0,319	0,334	4,7	0,332	4,1
5	4	1,93	1,94	0,4	1,93	0,2	0,049	0,060	18,3	0,060	18,1
1	6	0,46	0,45	1,3	0,45	1,7	0,429	0,430	0,2	0,434	1,2
2	6	0,83	0,82	0,4	0,81	1,9	0,212	0,220	3,4	0,214	1,0
5	6	1,33	1,34	0,2	1,35	0,9	0,017	0,018	7,8	0,019	11,1

Discussion Results Single Server

The first thing we notice is that the results are in general very good. We will discuss the results in more detail for the different cases separately.

For a buffer of size 1 the results for the utilization and mean buffersize are of course the same. The method generates good results for the utilization, so the results for throughput and mean buffersize are therefore also accurate. The blocking probability is a little bit smaller than the mean buffersize which is a logical consequence of the Arrival Theorem and the method is here very accurate. When the buffer is twice as big the errors for the throughput, utilization and blocking probability are the same as for the previous situation. The iterative method is for the mean buffersize not as good as before but it is still very accurate. We can see for both situations ($N = 1$ and $N = 2$) that the relative errors are in general a little bit bigger when the threshold is increasing.

When the buffersize is equal to 5 we again see good results for the utilization and throughput. The buffersize is again accurate, but for the case when the threshold is chosen to be 25 we can see some peaks in the relative errors. In the calculation of the blocking probabilities we can see some larger relative errors. These errors occur when the traveltime and threshold are chosen te be the smallest of the two. This can be explained by the fact that the main assumption of arriving totes seeing equilibrium does not necessarily hold.
In general we can state the following about the results of the 24 scenarios; the method is very good for the calculation of the utilization and throughput. If the buffersize is chosen rather big (in comparison with the threshold) the results for the mean buffersize and the probability of blocking can be bad, depending on the choice of the other parameters. For the blocking probability this error is not very important, because sometimes these probabilities are very small so a big error is made very quickly and also this is not the measure of our main interest. Our method gives for example a probability of blocking of $7,4\%$ while the simulation gives $\pm 6,3\%$, this is not even bad but in terms of relative errors this means an error of 15%. If the traveltime is big enough the method will give good results for all performance measures and also for both zones.

When the workload is slightly unbalanced the method still gives good results for all variables, especially for our main measures of interest: utilization and throughput. Also the blocking probability is estimated very good, except for one situation. A relative error of $21,5\%$ occurs there, but there we are also considering a rather small probability.

Let us now consider the results when altering the proportions to case 1: $p_1 = p_2 = 0.25$ and $p_3 = 0.5$. In this system more totes of type 3 are present so there are more totes that need to visit both zones. We should see this in the results; we would expect an utilization that is higher than the 'standard' situation. This is clearly the case for all investigated situations, but we do see a throughput that is less then the 'standard' situation, which can be explained with the chosen proportion of type of totes. The results for all performance measures of the simulation are approximated very accurate by the iterative method, except for one situation where the blocking probability is not approximated very well. Here the blocking probability is also rather small.
If the proportion of totes is altered so there are almost no type 3 totes then we would expect, of course, an utilization that is less than the 'standard' situations and this is the case. Also there is more throughput per hour, which is a logical consequence. The results are again overall very good, except for 2 peaks in the relative error of the blocking probability. Here again the probabilities are small, so an error of 18% can occur.

The method works very well in every situation for the most important performance measures: utilization and throughput. The mean buffersize is approximated also accurate for most cases and also the limiting distribution of the number of totes in the buffer is approximated nicely. This observation we have seen in a simulation, but here no results are presented. For small blocking probabilities the results are less accurate than when these probabilities are bigger. If we increase the mean traveltime then the calculation of the blocking probability will be better. This method always gives a higher value than simulated (except for the blocking probability), so we could say that our method gives an upper bound.

3.2.5 Results Infinite Server

In this section we present the results when the source is an infinite server. Here the arrival rate depends on the number of totes in the segment; if there are many totes in the segment then the supply will be less compared to the case when there are few totes in the segment. Again we will only present the results for the iterative method with Markov chain.

This section will follow the same structure as for the single server case, so the results for the 24 scenarios will be presented first, followed by the scenarios when the workload is unbalanced and finally we state the cases where the proportion of totes is not equally distributed. When considering the last 2 cases, workload unbalanced and different proportions, we only investigated the results when $C = 25$ and $1/\lambda = 100$.

In the simulations we have run, the maximal error of the simulations for the utilization is roughly everywhere beneath 0.02, for the throughput per hour this error is 1, the mean buffersize has an error of maximum 0.02 and finally for the blocking probability this is 0.02.

Buffersize: $N = 1$

$1/\lambda$	C	μ/γ	$\rho_1^{it} = \rho_2^{it}$	ρ_1^{sim}	$\Delta\%$	ρ_2^{sim}	$\Delta\%$	δ^{it}	δ^{sim}	$\Delta\%$
100	10	4	0,506	0,493	2,6	0,481	5,3	91,1	88,1	3,4
100	25	4	0,748	0,704	6,4	0,711	5,2	134,7	125,7	7,2
100	10	6	0,427	0,410	4,0	0,423	0,8	76,8	75,1	2,2
100	25	6	0,672	0,631	6,4	0,635	5,8	120,9	113,3	6,7
250	10	4	0,400	0,384	4,3	0,402	0,3	72,1	71,3	1,0
250	25	4	0,693	0,667	3,9	0,646	7,3	124,8	118,3	5,5
250	10	6	0,346	0,342	1,2	0,345	0,3	62,4	62,7	0,5
250	25	6	0,619	0,590	4,9	0,575	7,5	111,4	105,9	5,2

$1/\lambda$	C	μ/γ	$L_{z1}^{it} = L_{z2}^{it}$	L_{z1}^{sim}	$\Delta\%$	L_{z2}^{sim}	$\Delta\%$	$b_{it}^{(1)} = b_{it}^{(2)}$	$b_{sim}^{(1)}$	$\Delta\%$	$b_{sim}^{(2)}$	$\Delta\%$
100	10	4	0,51	0,49	2,5	0,48	5,2	0,479	0,498	3,7	0,474	1,2
100	25	4	0,75	0,70	6,2	0,71	5,2	0,741	0,738	0,4	0,751	1,4
100	10	6	0,43	0,41	3,9	0,42	0,8	0,401	0,397	1,0	0,422	5,2
100	25	6	0,67	0,63	6,2	0,64	5,7	0,663	0,669	1,0	0,675	1,9
250	10	4	0,40	0,38	4,3	0,40	0,3	0,375	0,373	0,4	0,393	4,6
250	25	4	0,69	0,67	3,8	0,65	7,3	0,684	0,696	1,7	0,684	0,0
250	10	6	0,35	0,34	1,2	0,35	0,3	0,323	0,329	1,9	0,318	1,4
250	25	6	0,62	0,59	4,7	0,58	7,5	0,609	0,623	2,3	0,605	0,6

Buffersize: $N = 2$

$1/\lambda$	C	μ/γ	$\rho_1^{it} = \rho_2^{it}$	ρ_1^{sim}	$\Delta\%$	ρ_2^{sim}	$\Delta\%$	δ^{it}	δ^{sim}	$\Delta\%$
100	10	4	0,615	0,605	1,6	0,608	1,1	110,6	109,6	0,9
100	25	4	0,908	0,851	6,7	0,872	4,2	163,5	156,8	4,2
100	10	6	0,523	0,518	0,8	0,517	1,1	94,1	94,0	0,1
100	25	6	0,843	0,805	4,8	0,798	5,7	151,8	144,2	5,2
250	10	4	0,449	0,451	0,4	0,442	1,5	80,8	80,3	0,6
250	25	4	0,839	0,798	5,2	0,797	5,2	151,0	144,5	4,5
250	10	6	0,394	0,398	0,8	0,379	4,1	71,0	71,8	1,1
250	25	6	0,764	0,746	2,4	0,733	4,2	137,5	133,7	2,8

$1/\lambda$	C	μ/γ	$L_{z1}^{it} = L_{z2}^{it}$	L_{z1}^{sim}	$\Delta\%$	L_{z2}^{sim}	$\Delta\%$	$b_{it}^{(1)} = b_{it}^{(2)}$	$b_{sim}^{(1)}$	$\Delta\%$	$b_{sim}^{(2)}$	$\Delta\%$
100	10	4	0,89	0,87	1,8	0,88	1,4	0,238	0,243	1,8	0,247	3,5
100	25	4	1,57	1,43	9,8	1,48	6,0	0,644	0,612	5,3	0,652	1,2
100	10	6	0,72	0,71	1,3	0,71	1,0	0,171	0,171	0,2	0,173	1,7
100	25	6	1,39	1,30	6,5	1,29	7,2	0,530	0,539	1,8	0,532	0,4
250	10	4	0,60	0,60	0,6	0,59	1,9	0,126	0,135	6,5	0,129	2,6
250	25	4	1,38	1,29	7,1	1,30	6,2	0,523	0,514	1,9	0,543	3,6
250	10	6	0,51	0,51	0,7	0,48	5,2	0,097	0,093	5,1	0,095	2,6
250	25	6	1,20	1,17	2,2	1,14	5,5	0,421	0,447	5,7	0,418	0,8

Buffersize: $N = 5$

$1/\lambda$	C	μ/γ	$\rho_1^{it} = \rho_2^{it}$	ρ_1^{sim}	$\Delta\%$	ρ_2^{sim}	$\Delta\%$	δ^{it}	δ^{sim}	$\Delta\%$
100	10	4	0,652	0,651	0,1	0,647	0,7	117,3	117,3	0,0
100	25	4	0,985	0,938	5,0	0,939	5,0	177,3	169,2	4,8
100	10	6	0,561	0,567	1,0	0,553	1,5	101,1	101,3	0,2
100	25	6	0,951	0,916	3,9	0,900	5,7	171,2	165,0	3,8
250	10	4	0,459	0,458	0,3	0,458	0,2	82,6	82,6	0,1
250	25	4	0,913	0,883	3,3	0,869	5,1	164,3	158,8	3,5
250	10	6	0,407	0,416	2,2	0,415	2,0	73,2	73,0	0,3
250	25	6	0,850	0,835	1,8	0,827	2,8	153,0	150,0	2,0

$1/\lambda$	C	μ/γ	$L_{z1}^{it} = L_{z2}^{it}$	L_{z1}^{sim}	$\Delta\%$	L_{z2}^{sim}	$\Delta\%$	$b_{it}^{(1)} = b_{it}^{(2)}$	$b_{sim}^{(1)}$	$\Delta\%$	$b_{sim}^{(2)}$	$\Delta\%$
100	10	4	1,37	1,35	1,0	1,35	0,9	0,018	0,024	25,5	0,025	27,1
100	25	4	4,05	3,64	11,2	3,61	12,2	0,455	0,459	0,9	0,453	0,6
100	10	6	1,04	1,06	1,8	1,03	1,7	0,008	0,013	37,9	0,014	43,5
100	25	6	3,48	3,31	5,3	3,10	12,5	0,305	0,336	9,3	0,297	2,6
250	10	4	0,75	0,73	2,9	0,73	2,0	0,003	0,004	19,8	0,005	38,9
250	25	4	3,06	2,94	4,1	2,84	7,6	0,222	0,266	16,3	0,258	14,0
250	10	6	0,62	2,52	3,6	0,63	1,2	0,002	0,007	77,9	0,003	45,1
250	25	6	2,54	3,64	0,8	2,45	3,6	0,143	0,170	15,7	0,170	16,0

Unbalanced workload

N	μ_2/γ	ρ_1^{it}	ρ_1^{sim}	$\Delta\%$	ρ_2^{it}	ρ_2^{sim}	$\Delta\%$	δ^{it}	δ^{sim}	$\Delta\%$
1	4	0,778	0,733	6,1	0,729	0,689	5,8	131,2	125,2	4,8
2	4	0,936	0,893	4,8	0,878	0,829	5,9	158,0	150,3	5,1
5	4	0,997	0,967	3,1	0,935	0,899	4,0	168,3	163,1	3,2
1	6	0,700	0,653	7,3	0,656	0,628	4,5	118,2	111,6	5,9
2	6	0,875	0,844	3,7	0,820	0,779	5,3	147,7	141,2	4,6
5	6	0,978	0,936	4,5	0,917	0,897	2,3	165,0	162,5	1,6

N	μ_2/γ	L_{z1}^{it}	L_{z1}^{sim}	$\Delta\%$	L_{z2}^{it}	L_{z2}^{sim}	$\Delta\%$	$b_{it}^{(1)}$	$b_{sim}^{(1)}$	$\Delta\%$	$b_{it}^{(2)}$	$b_{sim}^{(2)}$	$\Delta\%$
1	4	0,78	0,73	5,9	0,73	0,69	5,7	0,770	0,767	0,5	0,721	0,725	0,6
2	4	1,65	1,54	7,6	1,48	1,37	8,3	0,707	0,696	1,5	0,587	0,574	2,3
5	4	4,45	4,02	10,8	3,27	3,17	3,2	0,618	0,559	10,6	0,255	0,341	25,0
1	6	0,70	0,65	7,0	0,66	0,63	4,4	0,691	0,687	0,7	0,647	0,668	3,2
2	6	1,47	1,40	5,4	1,33	1,25	6,8	0,583	0,589	1,1	0,496	0,493	0,7
5	6	3,90	3,48	12,0	3,09	3,04	1,5	0,413	0,373	10,7	0,224	0,275	18,4

Proportion: Case 1

N	μ/γ	$\rho_1^{it} = \rho_2^{it}$	ρ_1^{sim}	$\Delta\%$	ρ_2^{sim}	$\Delta\%$	δ^{it}	δ^{sim}	$\Delta\%$
1	4	0,762	0,719	5,9	0,725	5,0	121,9	114,1	6,8
2	4	0,919	0,869	5,8	0,883	4,1	147,1	141,8	3,7
5	4	0,990	0,945	4,8	0,963	2,8	158,4	154,1	2,8
1	6	0,688	0,650	5,8	0,649	5,9	110,0	103,6	6,2
2	6	0,860	0,817	5,3	0,826	4,1	137,6	129,9	6,0
5	6	0,965	0,939	2,8	0,915	5,5	154,4	148,3	4,1

N	μ/γ	$L_{z1}^{it} = L_{z2}^{it}$	L_{z1}^{sim}	$\Delta\%$	L_{z2}^{sim}	$\Delta\%$	$b_{it}^{(1)} = b_{it}^{(2)}$	$b_{sim}^{(1)}$	$\Delta\%$	$b_{sim}^{(2)}$	$\Delta\%$
1	4	0,76	0,72	5,7	0,73	5,0	0,754	0,747	0,9	0,764	1,3
2	4	1,60	1,47	9,0	1,51	6,2	0,668	0,637	4,8	0,673	0,7
5	4	4,17	3,69	13,0	3,82	9,2	0,498	0,465	7,0	0,500	0,4
1	6	0,69	0,65	5,6	0,65	5,8	0,679	0,685	0,9	0,687	1,2
2	6	1,43	1,34	7,1	1,35	5,7	0,557	0,543	2,6	0,582	4,2
5	6	3,67	3,53	4,0	3,24	13,3	0,348	0,386	9,8	0,318	9,4

Proportion: Case 2

N	μ/γ	$\rho_1^{it} = \rho_2^{it}$	ρ_1^{sim}	$\Delta\%$	ρ_2^{sim}	$\Delta\%$	δ^{it}	δ^{sim}	$\Delta\%$
1	4	0,723	0,672	7,6	0,688	5,0	157,7	147,6	6,9
2	4	0,885	0,834	6,0	0,828	6,8	193,0	183,1	5,4
5	4	0,971	0,920	5,6	0,924	5,1	211,8	201,3	5,2
1	6	0,642	0,596	7,7	0,604	6,4	140,2	132,4	5,9
2	6	0,809	0,778	3,9	0,769	5,2	176,4	169,4	4,1
5	6	0,917	0,859	6,8	0,899	2,0	200,1	193,7	3,3

N	μ/γ	$L_{z1}^{it} = L_{z2}^{it}$	L_{z1}^{sim}	$\Delta\%$	L_{z2}^{sim}	$\Delta\%$	$b_{it}^{(1)} = b_{it}^{(2)}$	$b_{sim}^{(1)}$	$\Delta\%$	$b_{sim}^{(2)}$	$\Delta\%$
1	4	0,72	0,67	7,5	0,69	5,0	0,714	0,706	1,2	0,731	2,3
2	4	1,50	1,38	8,8	1,37	9,1	0,599	0,588	1,9	0,601	0,4
5	4	3,77	3,39	11,1	3,42	10,1	0,374	0,393	5,0	0,406	7,9
1	6	0,64	0,60	7,5	0,60	6,4	0,633	0,637	0,6	0,646	2,0
2	6	1,30	1,25	4,3	1,22	7,2	0,479	0,497	3,5	0,477	0,5
5	6	3,10	2,76	12,5	3,12	0,6	0,230	0,228	0,8	0,287	19,9

Discussion Results Infinite Server

In general the iterative method gives good approximations for all performance measures and it is also clear that these results are a little bit worse than for the single server case. It is not quite clear why this is the case. What we also have seen for the single server case is that the performance measures for both zones in the simulation do not give the same values, while for the iterative method this is the case. This can be explained by the simulation error, if we would simulate much longer those values will be approximately the same. In the remaining part of this section we will discuss the method and its results in more detail for the different cases separately.

If there is only 1 place in the buffer the results are acceptable for all performance measures and also for both zones. Again the iterative method gives always a higher value for most of the performance measures, except for the blocking probabilities. For a buffer of size 2 the results are comparable and therefore also acceptable.
For buffersize 5 the results for utilization and throughput are very good and also the mean buffersize is calculated accurate. However for large values for the threshold and λ there are some peaks of 12% for relative errors. The measure for which the method is most off is again the blocking probability. Here we again see some major peaks, even of $77,9\%$, but this error occurs when this blocking probability is very small (approximately 1%). While for all other measures the method provides an upper bound, for the blocking probability this is a lower bound.

If we change one mean pick time, in this case of the operator at zone 1, we generate an unbalanced workload and this is the following situation we will discuss. The results are again acceptable despite the change in workload, especially the utilization and throughput are approximated well by the iterative method. Again the calculation of the mean buffersize is acceptable, with some moderate peaks, this also holds for the blocking probabilities.

If we let half of the totes go to both zones and an equal part go to only zone 1 or zone 2, we would expect the utilization to be bigger than in the 'standard' scenarios. This is the case and just like the single server source the utilization is less then before, which is a clear consequence of the chosen parameters. The relative errors for all performance measures, even for the blocking probabilities, are acceptable.
For the other choice of different proportions; almost no type 3 totes, the iterative method gives approximately the same results as the previous case. We see what we would expect beforehand; the utilization is lower than the 'standard' scenarios. Here also the blocking probabilities are calculated accurately.

In general we can say that the iterative method works good for the infinite server source, but in general worse than for the single server case. The blocking probability however is calculated more precise than for the single server case.

3.3 One Segment, Multiple Zones

In this section we will model a whole segment with any number of zones, say $z \in \mathbb{N}^+$. This model will be an extension of the model described in the previous section and also uses an iterative algorithm (with Markov chain) to calculate the performance measures of this system. We will only present the algorithm with Markov chain here, because this is the more accurate algorithm and when considering more zones more blocking will occur.

The main idea stays the same: the blocking probability of each zone will be calculated in each step of the iteration and with these probabilities the probability of a retrial can be calculated. The iteration stops if the difference between all the blocking probabilities is small enough. The segment will be modelled in the same manner as in the previous section and schematically this looks like Figure 3.5. Here again the stations with the μ_i's are modelled as single server stations and the stations with the γ_i's are infinite server stations. The source, the station with λ, can be single server of infinite server. We will define the variables used here for completeness sake

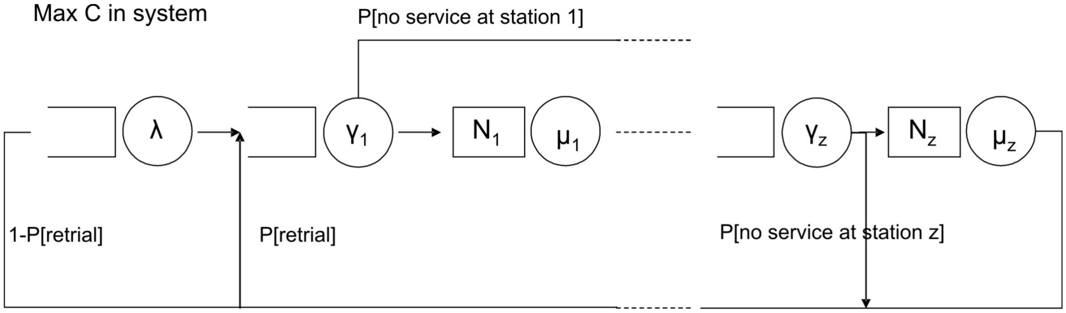


Figure 3.5: A closed queueing model of a segment with z zones

P(no service at zone i): the probability that an arbitrary tote does not need to visit zone i (=1 $-$ P(service at zone i)), with $i = 1, \ldots, z$.

P(retrial): the probability that an arbitrary tote needs to make a recirculation because he did not visited all his zones, because these zones where fully occupied.

The last variable is again of our main interest and this is the probability that a tote is blocked by at least one of the zones he needs to visit. This probability can be calculated in the same manner as done in the previous chapter.

Let us define B_i as the event that a tote is blocked at zone i. When a tote does not need to visit a zone this event does not occur. Using the probability axiom

$$P(A \cup B) = P(A) + P(B) - P(A \cap B), \tag{3.13}$$

and extend this for a more general case we obtain

$$\begin{aligned} P(\text{retrial}) := P(\cup_{i=1}^{z} B_i) &= \sum_{i=1}^{z} P(B_i) - \sum_{i \neq j} P(B_i \cap B_j) + \sum_{i \neq j \neq l} P(B_i \cap B_j \cap B_l) - \ldots \\ &+ (-1)^{z+1} \sum_{i \neq j \neq \ldots \neq r} P(B_i \cap B_j \cap \ldots \cap B_r). \end{aligned} \tag{3.14}$$

If we obtain the blocking probabilities for every zone this will also give us the probability of a retrial, because the probabilities of visiting a certain zone are given from the order-data. We will determine these blocking probabilities in the same manner as is done in the previous chapter for the one zone case and in the previous section for the 2 zone case. The system again considers jump-over blocking and the routing probabilities will be independent of whether a tote is served or not at the previous zone, this is again the difference with reality. This however will describe the system very well as we will see later.

The blocking probabilities will be set to 0 for all the zones and therefore also the probability of a retrial will be 0. The limiting probabilities have a product form as we have seen earlier and also is stated in [6]:

$$\begin{aligned} P(n_1, n_2, \ldots, n_{2z+1}) &= d(\frac{1-p^{(r)}}{\lambda})^{n_1} \prod_{i=1}^{z} \frac{1}{n_{2i}!}(\frac{1}{\gamma_i})^{n_{2i}}(\frac{q_i}{\mu_i})^{n_{2i+1}} \\ &\quad \text{for single server source,} \qquad (3.15) \\ P(n_1, n_2, \ldots, n_{2z+1}) &= d\frac{1}{n_1!}(\frac{1-p^{(r)}}{\lambda})^{n_1} \prod_{i=1}^{z} \frac{1}{n_{2i}!}(\frac{1}{\gamma_i})^{n_{2i}}(\frac{q_i}{\mu_i})^{n_{2i+1}}, \\ &\quad \text{for infinite server source,} \qquad (3.16) \end{aligned}$$

where d is the normalizing constant and n_i denotes the number of totes at station i of Figure 3.5. Here we also, for abbreviation purposes, introduced the notations:

$$\begin{aligned} q_i &:= P(\text{service at workzone } i), \\ p^{(r)} &:= P(\text{retrial}). \end{aligned}$$

With the limiting probabilities as given in (3.15) and (3.16) we can determine the blocking probabilities for every zone, by calculating the probability that the specific zone is fully occupied (so N_i totes are present at zone i) in the system with $C-1$ totes. This last remark can again be proven with the Arrival Theorem, but this will not be done here.

As said before we will only consider the iterative method with Markov Chain, because it is a more accurate method. This method will be described in the next section.

3.3.1 Iterative Method with Markov Chain

In this section we will describe the iterative method with Markov Chain for the one segment case with a general number of zones. The first thing we want to determine is the proportion of totes that jumps certainly over a zone, because they do not have to visit that zone. Therefore we have to consider the different types of totes. In general there can be $2^z - 1$ different types of totes, because a tote must visit at least one zone. In reality however it is not likely that a tote needs to visit all the zones or a major part of all the zones. The number of types will therefore be much less than $2^z - 1$, but here we will consider this general case.

First we will consider the different types of totes that can occur when considering this model. We will number the types in logical ordening where we start with type 1 which represents the tote that only needs to visit zone 1 until type z which is the tote that only needs to visit zone z. Type $z+1$ is a tote that needs to visit zone 1 and zone 2, type $z+2$ a tote that needs to visit zone 1 and zone 3 etc. We will do this until we have type $2^z - 1$ which is the tote that needs to visit all the zones.

These types are the states of the Markov Chain and the transition probabilities depend on the blocking probabilities at the different zones and the proportion of types released, which works in the same manner as for the 2 zone case. If a tote is served in at least one zone it changes into another type and if a tote has visited all its zones it becomes a type i tote with probability p_i. The notation will be similar as before, but for completeness we introduce

$$\begin{aligned} b^{(i)} &:= P(\text{The tote has been blocked at zone } i), \quad i = 1, \ldots, z, \\ p_i &:= \text{The proportion of new totes of type } i = 1, \ldots, 2^z - 1 \text{ released such that } \sum_{i=0}^{2^z-1} p_i = 1. \end{aligned}$$

With these variables we can describe the transition probabilities between the different tote types. For $z = 2$ the transitions probabilities of the Markov chain are given in Figure 3.4, for more zones this will work in the same manner.
The transitions are known and we can solve the balance equations rather easily, because we will have 2^z equations (for each type we have 1 equation plus the normalizing equation) and we have $2^z - 1$ unknown (the limiting probabilities π_i of the Markov chain). In reality some of these equations will drop, because some types will not occur in the system.

Now we have introduced the notation for the blocking probabilities and we have the limiting probabilities of the Markov chain we can get an expression for the probability of a retrial using (3.14) and the assumption that these blocking events are independent of each other. This assumption can again be grounded by the fact that in each iteration we will look at the arrival at each zone if a tote is blocked. Here we forget what happened to this tote before so therefore we assume independency. The probability of a retrial simplifies to

$$P(\text{retrial}) := \sum_{i=1}^{z} \pi_i b^{(i)} - \pi_{z+1} b^{(1)} b^{(2)} - \pi_{z+2} b^{(1)} b^{(3)} - \ldots + \ (-1)^{z+1} \prod_{i=1}^{z} b^{(i)} \pi_{2^z-1}. \tag{3.17}$$

In the end we will have an expression for the limiting probabilities and now we can determine the utilization and other performance measures, like mean number of totes in the several zones, probability of blocking at all zones and many more. The utilization of zone i is formally given by

$$\begin{aligned} \rho_1 &= 1 - \sum_{n_{2z+1}=0}^{N_z} \ldots \sum_{n_2=0}^{C-1-\sum_{j=3}^{2z+1} n_j} P(C - 1 - \sum_{j=2}^{2z+1} n_j, n_2, 0, \ldots, n_{2z+1}), && (3.18) \\ &\vdots && \vdots \\ \rho_z &= 1 - \sum_{n_{2z}=0}^{C-1} \ldots \sum_{n_2=0}^{C-1-\sum_{j=3}^{2z+1} n_j} P(C - 1 - \sum_{j=2}^{2z+1} n_j, n_2, \ldots, n_{2z}, 0). && (3.19) \end{aligned}$$

The order of the algorithm is just like the other models before given by the state space on which we have to sum up all the limiting probabilities, because in each iteration the normalizing constant needs to be calculated.

$$\text{Order algorithm} = \mathcal{O}(N_1 \ldots N_z C^z), \tag{3.20}$$

We can see that the order grows rapidly and when the number of zones is large, the value of C will also be chosen large. In the next section we will present a new method to calculate the same performance measures.

Algorithm 7 Iterative Method with Markov Chain for z Zones

1. Set $i = 0$ and $b_0^{(j)} = 0$, for $j = 1, \ldots, z$ so $p_0^{(r)} = 0$.

2. Set $i = i + 1$ and calculate new blocking probabilities for all zones:

$$\begin{aligned} b_i^{(1)} &= \sum_{n_{2z+1}=0}^{N_z} \cdots \sum_{n_2=0}^{C-1-\sum_{j=3}^{2z+1} n_j} P(C-1-\sum_{j=2}^{2z+1} n_j, n_2, N_1, \ldots, n_{2z+1}), \\ &\vdots \qquad\qquad\qquad\qquad\qquad\qquad \vdots \\ b_i^{(z)} &= \sum_{n_{2z}=0}^{C-1} \cdots \sum_{n_2=0}^{C-1-\sum_{j=3}^{2z+1} n_j} P(C-1-\sum_{j=2}^{2z+1} n_j, n_2, \ldots, n_{2z}, N_z), \end{aligned}$$

   where $P(n_1, n_2, \ldots, n_{2z+1})$ is given by one of the product forms ((3.15) and (3.16)), depending on the use of a single or infinite server source respectively and where we use $p^{(r)} = p_{i-1}^{(r)}$ and the new values for the proportion of totes. These are given in the following steps of the iteration.

3. Calculate the new transition probabilities $\pi_j, j = 1, \ldots, 2^z - 1$ using the new blocking probabilities

4. Calculate the new proportion of each type of totes:

$$p_j = \pi_j, \quad j = 1, \ldots, 2^z - 1.$$

5. Calculate visit frequencies at each zone, by summing over types that visit the zone.

6. Calculate the new probability of a retrial using (3.17).

7. Repeat step 2 until $|b_i^{(j)} - b_{i-1}^{(j)}| < \epsilon$ for $j = 1, \ldots, z$.

3.3.2 Iterative MVA

To avoid long calculation times we will describe a Mean Value Analysis (MVA). We already have an algorithm which gives us the limiting probabilities of the total system, so why this new analysis which only will determine means. First we mention that this analysis will not be completely different, we will still use jump over blocking and especially the Arrival Theorem for these systems. The answers therefore will also be the same but only will give mean performance measures instead of the entire limiting distribution. The MVA described here will be called the Iterative MVA, because it is used in each iteration step. Later on we will also describe a MVA that will not be used iteratively, this will be called the Direct MVA.

This analysis will be important to determine the performance measures for large systems very quickly. Algorithm 7 will be very slow for large systems and therefore not useful, for small systems or small values of the threshold this algorithm will work and can be of interest if we want to know more than means. This new method has an order which will be linear in the number of zones. This is a very big reduction of the order and therefore also a big reduction in the time to calculate the different performance measures. This can be done for every system. First we wil describe the MVA for the one zone model, to sketch and understand this analysis. Followed by the MVA for a general number of zones.

One Zone Model

Let us recall the one zone model, this model is schematically given in Figure 3.6. The source can be a single server or an infinite server, depending on the choice of the model. The zones will be numbered in logical order and the MVA will work as follows for single server source, when the population has size k. First we will introduce some notation; where all quantities depend on the population size k:

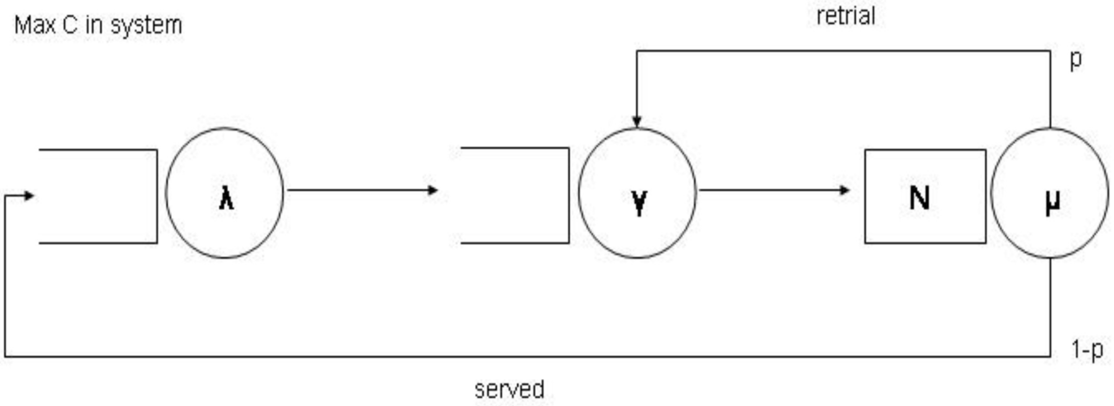


Figure 3.6: A closed retrial queueing system, with a finite source

$$\begin{aligned} S_i(k) &= \text{mean production lead time at work station } i \\ L_i(k) &= \text{mean number of jobs at work station } i \\ \Lambda_i(k) &= \text{throughput of work station } i \end{aligned}$$

For the mean production lead time at the three stations we will have the following equations:

$$S_1(k) = L_1(k-1)\frac{1}{\lambda} + \frac{1}{\lambda}, \tag{3.21}$$

$$S_2(k) = \frac{1}{\gamma}, \tag{3.22}$$

$$S_3(k) = \sum_{i=0}^{N-1} P_i(k-1)[\frac{i}{\mu} + \frac{1}{\mu}] + P_N(k-1) \cdot 0, \tag{3.23}$$

if we would consider the infinite server source then we would have $S_1(k) = \frac{1}{\lambda}$. Here $P_i(k-1)$ represents the probability that there are i totes in the zone when the population has size $k-1$. We know these probabilities exactly because we have the limiting probabilities, but we will not use this here and we will determine these later on. The mean cycle time is defined as the time between two visits at the second station, and is given by

$$C(k) = \sum_{i=1}^{3} \nu_i S_i(k) = (1-p) \cdot S_1(k) + S_2(k) + S_3(k), \tag{3.24}$$

where p is the routing probability as sketched in Figure 3.6 and ν_i is the relative visiting frequency to station i. The throughput is then given for each zone by

$$\begin{aligned} \Lambda_1(k) &= \frac{(1-p)k}{C(k)}, &(3.25)\\ \Lambda_2(k) &= \frac{k}{C(k)}, &(3.26)\\ \Lambda_3(k) &= \frac{k}{C(k)}, &(3.27) \end{aligned}$$

and finally by Little's Law we have the mean number of totes at zone $1, 2$ and 3:

$$\begin{aligned} L_1(k) &= S_1(k) \cdot \Lambda_1(k),\\ L_2(k) &= S_2(k) \cdot \Lambda_2(k),\\ L_3(k) &= S_3(k) \cdot \Lambda_3(k). \end{aligned}$$

We can recursively compute $S_i(k),\ \Lambda_i(k)$ and $L_i(k)$ for populations $k = 0$ up to $k = C$. Initially we have

$$\begin{aligned} L_i(0) &= 0, \quad i = 1, 2, 3 \text{ and} &(3.28)\\ P_0(0) &= 1, \quad \text{so } P_i(0) = 0 \;\; i = 1, \dots, N. &(3.29) \end{aligned}$$

The only thing we have to determine are the probabilities $P_i(k-1)$ in each iteration step. This can be done by looking at the transition rates between the states of the zone. The states of the zone are the number of totes that are present in this zone and these states with transition rates are schematically described in Figure 3.7. The rate from state i to state $i-1$ is given by the service rate μ multiplied by the probability we are in state i. Transition rates from state i to state $i+1$ are given by the rate at which new totes arrive $\Lambda_3(k)$, and here we also multiply by the probability we are in state i. However here this probability is given that there is one tote less in the system, because of the Arrival Theorem.

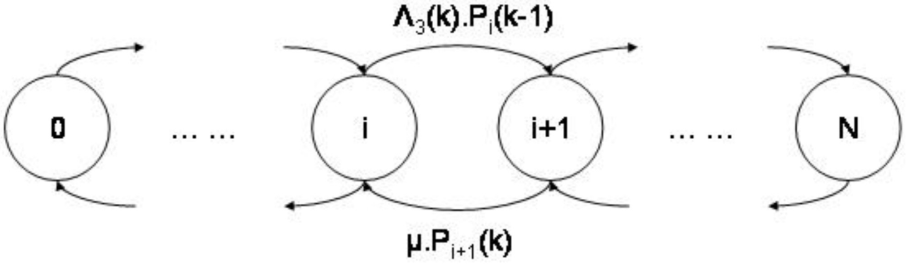


Figure 3.7: Transition rates between the states of the zone

We can use the local balance equations to obtain the following expression for $P_{i+1}(k)$:

$$P_{i+1}(k) = \frac{\Lambda_3(k)P_i(k-1)}{\mu}, \quad i = 0, \ldots, N-1, \tag{3.30}$$

together with the normalizing equation

$$\sum_{i=0}^{N} P_i(k) = 1, \tag{3.31}$$

we obtain an expression for all probabilities, which we need to determine the mean lead time of the zone in the next iteration. The algorithm for this one zone model looks very similar as we have seen before, the only thing we have to alter is the calculation of the blocking probability and at the end of the iteration the calculation of the performance measures. The calculation of the blocking probability is replaced by the MVA described above and where eventually the blocking probability is given by $P_N(C-1)$. The throughput and mean buffersize are calculated each step of the MVA and eventually the utilization of the zone is given by:

$$\rho = 1 - P_0(C). \tag{3.32}$$

In theory this method should give the same results and this is successfully checked for several cases. The algorithm will be given in the next part, where we give the MVA for a general number of zones.

General Number of Zones

The MVA analysis for an arbitrary number of z zones works in a similar manner. Here again the different type of totes play an important role. Just like the old analysis, with limiting probabilities, we will see the influence of these types only in the visiting frequencies at the different zones. In general we can state the following scheme for the MVA when the population is k, for single source:

$$\begin{aligned} S_s(k) &= L_1(k-1)\frac{1}{\lambda} + \frac{1}{\lambda}, & (3.33)\\ S_{cj}(k) &= \frac{1}{\gamma_j}, \; j = 0, \ldots, z, & (3.34)\\ S_{zj}(k) &= \sum_{i=0}^{N-1} P_i^j(k-1)[\frac{i}{\mu_j} + \frac{1}{\mu_j}] + P_N^j(k-1)\cdot 0, \; j = 0, \ldots, z. & (3.35)\end{aligned}$$

Here the subscript s stands for the source, cj for conveyor piece j and zj for zone j. If we consider the infinite source we would have $S_s(k) = \frac{1}{\lambda}$. Here $P_i^j(k-1)$ represents the probability that there are i totes in zone j when the population has size $k-1$. The mean cycle time of a tote is given by

$$C(k) = \sum_{i=1}^{2z+1} \nu_i S_i(k), \tag{3.36}$$

where ν_i is the visiting frequency of station i. If this station represents a conveyor piece we know that $\nu_{cj} = 1$ and otherwise these frequencies are determined in the same manner as is done in step 5 of Algorithm 7.

The throughput for each zone is given by

$$\begin{aligned}\Lambda_s(k) &= \frac{k \cdot \nu_s}{C(k)},\\ \Lambda_{cj}(k) &= \frac{k}{C(k)},\ \ j = 0, \ldots, z\\ \Lambda_{zj}(k) &= \frac{k \cdot \nu_{zj}}{C(k)}\ \ j = 0, \ldots, z,\end{aligned}$$

and finally by Little's Law we have the mean number of totes at zone i:

$$L_i(k) = S_i(k) \cdot \Lambda_i(k),\ \ i = 0, \ldots, 2z + 1.$$

We can recursively compute $S_i(k), \Lambda_i(k)$ and $L_i(k)$ for populations $k = 0$ up to $k = C$. Initially we have

$$\begin{aligned}L_i(0) &= 0,\ \ i = 1 \ldots, 2z + 1\ \ \text{and} && (3.37)\\ P_0^j(0) &= 1,\ \ \text{so } P_i^j(0) = 0\ \ i = 1, \ldots, N \text{ and } j = 1, \ldots, z. && (3.38)\end{aligned}$$

In the same manner as for the one zone model we can determine $P_{i+1}^j(k)$ by using:

$$P_{i+1}^j(k) = \frac{\Lambda_{zj}(k) P_i^j(k-1)}{\mu_j},\ \ \ i = 0, \ldots, N-1 \text{ and } j = 1, \ldots, z, \tag{3.39}$$

together with the normalizing equation

$$\sum_{i=0}^{N} P_i^j(k) = 1,\ \ \ j = 1, \ldots, z. \tag{3.40}$$

For the 2 zone model we have checked the results this method provides with those we already obtained using the limiting probabilities, for both single server as infinite server source. The results obtained are the same and with this MVA method the results are obtained significantly faster than before. For larger systems this will be more of an issue as said earlier. We can see this also in the order of the algorithm, which depends only on the choice of the number of zones and the maximum number of totes allowed in the system

$$\text{Order algorithm} = \mathcal{O}(zC), \tag{3.41}$$

which is much less than the order we saw in (3.20) for the old iterative method. A very short outline of the new algorithm in general is given in Algorithm 8.

3.3.3 Direct MVA

Instead of iterating the blocking probability as is done with iterative MVA, we can also determine this probability without iteration. This is an direct MVA method and for the greater part the approach is the same as for the iterative MVA in the previous section. The direct MVA will be explained according to the one zone model.

Algorithm 8 Algorithm with MVA

```
initialize Set blocking probabilities to 0
repeat
    Mean Value Analysis: determine new blocking probabilities by $b_i^{(j)} = P_N^j(C-1)$
    Markov Chain: determine new visit frequencies and retrial probability
until Blocking probabilities do not alter
return Performance measures
```

Recall that the iteration for the iterative MVA started by first setting the routing probability p of equations (3.24) and (3.25) to 0. For the direct MVA this routing probability p of (3.24) and (3.25) will be changed into the blocking probability $P_N(k-1)$ which is determined in each step of the regular MVA approach. Note that for populations of size $k \leq N$ this probability will be 0 and no circulation will occur. Eventually we can determine the performance measures in the same manner as is done for the iterative MVA. When we consider more zones this blocking probability will be determined for each zone and the rest of the calculation is similar as before.

Two main advantages for this method are that no iteration is needed and that the performance measures can easily be plotted as a function of the threshold. In this way the best value for the threshold can be chosen when the other variables are known. A disadvantage can be that for big thresholds the fractions of visit must be determined many times, this can be more than the number of iterations of the iterative MVA. The order of the direct MVA is the same as for the iterative MVA. We can not tell on forehand which of the two MVA's will work faster or better, but this will be determined later on.

3.3.4 General Simulation

The algorithms described in the previous section must be compared with a simulation, to check the accuracy of our model. Therefore we have to expand the simulation to an arbitrary number of zones. In our simulation we have chosen that the number of zones can not be greater than 26. In reality a segment will in most cases not contain more than 26, in those situations a solution with multiple segments will often be chosen. In this section we will describe how the simulation has been set up and how the simulation is validated. An overview in Automod where 10 zones of the 26 are selected is given in Figure 3.8.

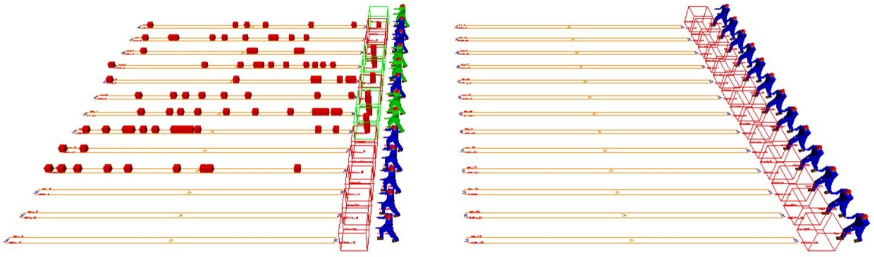


Figure 3.8: The Automod simulation model for any zone, here 10 zones are activated.

In the simulation for 2 zones we made a distinction between the different types, but for a large

number of zones this can not be done in an orderly manner and therefore will not be useful. Instead of making this distinction we determine at the creation of each tote which zones it has to visit and put this in an array. This is a 0 on position j if the tote does not need to visit the zone j and a 1 if it does need to visit that zone. The tote travels to each zone and if a tote arrives at a zone there are three possible situations: the tote needs to be there and the buffer is not fully occupied, in this case the tote enters the zone for service and the simulation sets the variable of visit to 0, because this tote does not need service there anymore.
Secondly, a tote needs to be at this zone but the buffer is fully occupied, then the tote enters the next conveyor and the variable of visit stays 1. Finally, the last situation is when the tote does not have to be there, then the zone just enters the next conveyor and the variable of visit stays 0. This is just the principle of ZPS.

The only thing that is tricky is at the last zone. There we have to check if a tote has visited all its zones; we do this by summing over the array of visit variables. If this sum is 0 the tote has visited all zones otherwise he still has to visit 1 or more zones. In the first case the tote is sent to die and leaves the system. If the tote still needs to visit some zones the tote enters the first conveyor and continues its journey.

If a tote has been served at a zone the throughput for that zone is increased by 1 and at the final zone also the total throughput is increased by 1. Also we determine the number of totes which want to visit the zone but can not be served because of the capacity of the buffer, this is useful for the calculation of the blocking probability. At the end of each run these performance measures, including utilization and mean buffersize are written to a file.

Let us take a closer look at the zones a tote has to visit. We introduce parameter v as the parameter that gives the mean number of zones to visit and recall that z is the total number of zones in the segment. We will distinguish two ways to generate these visits:

1. For each zone with probability $\frac{v}{z}$ the zone is visited and with probability $1 - \frac{v}{z}$ the zone is not visited. In this way 0 to z zones can be visited.

2. The parameter v is the number of zones that will be visited exactly.

This only holds when v is integer. If this is not the case a linear combination of two integers can be taken, but we will assume that v is integer. First let us see why these two ways will generate same results and why this would help in our analysis:

On average the number of zones that are visited in the 2 methods are the same, so therefore one would say also the utilization, throughput and mean buffersize will be the same. For the blocking probability this is also the case, because this probability is calculated by dividing the total number of totes that could not get in the zone with the total number of totes that would like to enter the zone. These two quantities will on average also be the same, regardless of the choice which way to generate visits. We would expect that this will give the same results but why would we do that much effort.
This observation would help in our analysis, because it will reduce the total number of types. If we generate visits using method 1 the total number of types of types will be 2^z, while using method 2 this will be reduced very fast. The types we have to distinguish are the types that will visit 1 to v zones; during a circulation a tote can become a type that still has to visit less than v

zones. The total number of types (T) will therefore become:

$$T = \sum_{i=0}^{v} \binom{z}{i}$$

To see the impact of this simplification: for a total number of zones of 20 and a mean number of 5 visits method 1 would have a total number of types of more than a million, while the second method only has 21.699 different types. This means a reduction of a factor of almost 50 in the number of variables.

Mean number of visits

When considering the first way of generating visits to the zones, we have to be careful. When calculating the mean of the number of visits we also included that new totes can visit 0 zones. In reality this can never occur, because these totes are useless (they will only travel and stay empty). In the simulation we will generate a new tote until it visites more than 0 zones. In theory this can happen infinitely many times, but with v and z large enough the probability that a tote has to visit 0 zones is very small. These types of totes should therefore not be included, but the mean number of visits should be v to make comparison between the two ways useful and also the number of zones is still z. To obtain this we must alter the probability that a zone is included, this can but certainly not always be the same as $\frac{v}{z}$. We can determine the new probability p that a zone is visited in terms of parameters v and z:

$$\begin{aligned} v &= \mathbb{E}[V|V>0] \\ &= \frac{\mathbb{E}[V.I[V>0]]}{\mathbb{P}[V>0]} \\ &= \frac{\mathbb{E}[V]}{\mathbb{P}[V>0]} \\ &= \frac{pz}{1-(1-p)^z}. \end{aligned}$$

In general we can not get a closed expression for p in terms of p and v. If we have exact values for z and v we can determine the value of p. When $v = z$ we can clearly see that $p = 1$, which agrees with the case that all zones needs to be visited, and when $v = 1$ we get $p = 0$. This last situation gives a contradiction, because when $p = 0$ no zone will be visited while the mean number of zones to be visited $v = 1$. In reality this situation will also not occur so the case that one zone is visited will be omitted here. To see the difference for small v between this new probability p and the old $\frac{v}{z}$ we show some values when $z = 10$ in Table 3.5. In this table we can see that when the mean number of zones to visit is small the probability $\frac{v}{z}$ is not accurate enough to get an average number of visited zones of v. For larger v this probability is accurate enough. We will use these newly obtained probabilities in the next part to see if the two different methods will generate the same result, so we can use that fact to simplify our analysis, because less types will be used.

Comparison of the two methods

In this part we will investigate if the two methods described in the previous part will generate approximately similar results for the main performance measures. If this is the case we will for further purposes use the second method in which we visit exactly v zones. First we will show

v	p	$\frac{v}{z}$
2	0,168	0,2
3	0,290	0,3
4	0,397	0,4
5	0,500	0,5
6	0,600	0,6
$\vdots$	$\vdots$	$\vdots$
10	1,000	1,0

Table 3.5: Probabilities to visit a zone, when mean number of zones to visit is v

that the two methods give similar results.

For this we first introduce some variables which we used in the simulation:

1. Number of zones: $z = 10$,
2. Buffersize: $N_i = 5$, for $i = 1, \ldots, z$,
3. Threshold: $C = 100$,
4. Mean pick time: $1/\mu_i = 30$ seconds, for $i = 1, \ldots, z$,
5. Mean inter arrival time: $1/\lambda = 4$ seconds,
6. Mean time per conveyor: $1/\gamma_i = 90$ seconds, for $i = 1, \ldots, z$

The only parameter we vary is the mean number of visits: $v = 3$ and $v = 7$. If the results of the 2 methods coincide with each other for these 2 specific cases we believe that this also will be the case for any other case. First we will show the results for $v = 3$ followed by the results for $v = 7$. Here the script ex represents the results when the number of zones visited is exact v and the script $mean$ when only the mean number of zones visited is v. We will only present the results for zone 1 and zone 10, for the other zones these results are similar. The simulation errors are not bigger than $0,003$, $0,008$ and $0,009$ for the utilization, mean buffersize and blocking probabilities respectively.

$Mean\ zones\ visited:\ v = 3$
We will discuss the results when the mean number of zones visited is $v = 3$. In Table 3.6 we state the results for utilization, mean buffersize and blocking probability. We can see from the presented results in Table 3.6 that the two different methods give similar results. Only for the blocking probability we see relative errors that are a bit larger than wanted. These probabilities fluctuate when looking at the different zones, but when looking at the average blocking probability over the 10 zones we see; $b_{ex} = 0,069$ and $b_{mean} = 0,070$ with a relative error of $0,5\%$, which is very accurate.

$Mean\ zones\ visited:\ v = 7$
Here the mean number of zones to be visited is $v = 7$ and we can find the results in Table 3.7. We see similar results between the different methods, the blocking probability is even better than we saw for $v = 3$. The average blocking probability is also similar; $b_{ex} = 0,205$ and $b_{mean} = 0,201$

ρ_1^{ex}	ρ_1^{mean}	$\Delta\%$	ρ_{10}^{ex}	ρ_{10}^{mean}	$\Delta\%$
0,687	0,683	0,7	0,690	0,685	0,8

L_{z1}^{ex}	L_{z1}^{mean}	$\Delta\%$	L_{z10}^{ex}	L_{z10}^{mean}	$\Delta\%$
1,68	1,66	1,2	1,68	1,66	1,3

$b_{ex}^{(1)}$	$b_{mean}^{(1)}$	$\Delta\%$	$b_{ex}^{(10)}$	$b_{mean}^{(10)}$	$\Delta\%$
0,073	0,070	4,8	0,071	0,079	10,8

Table 3.6: Simulation results when mean number of visits is $v = 3$

ρ_1^{ex}	ρ_1^{mean}	$\Delta\%$	ρ_{10}^{ex}	ρ_{10}^{mean}	$\Delta\%$
0,827	0,836	1,2	0,827	0,826	0,1

L_{z1}^{ex}	L_{z1}^{mean}	$\Delta\%$	L_{z10}^{ex}	L_{z10}^{mean}	$\Delta\%$
2,55	2,59	1,7	2,56	2,54	0,9

$b_{ex}^{(1)}$	$b_{mean}^{(1)}$	$\Delta\%$	$b_{ex}^{(10)}$	$b_{mean}^{(10)}$	$\Delta\%$
0,205	0,206	0,4	0,204	0,214	4,8

Table 3.7: Simulation results when mean number of visits is $v = 7$

with a relative error of $1,9\%$.

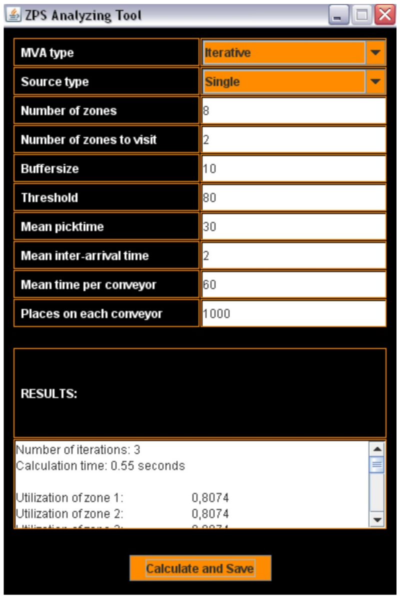
After seeing these results we can assume that there is a negligible difference between the two methods. This observation will result in the use of the second method: each tote will visit exactly v zones. As mentioned earlier this will help us in our analysis, especially for large systems.

3.3.5 Java Application: ZPS Analyzing Tool

The two methods described in previous sections, the iterative and direct MVA, are implemented in Java $6.0.3$. In this section we will shortly explain how this program, called 'ZPS Analyzing Tool', is implemented and what its limitations and possible extensions are. This section can be regarded as a short user manual.

The program uses a user interface as can be seen in Figure 3.9. Here we can see an input area for the variables in the top of the interface, a white output area is just below and at the bottom of the interface we can find the calculate button.
When the program is started this interface is shown and the user can give input values for the variables, these are respectively the following

1. The type of MVA: Iterative or Direct.
2. The type of source: Single or Infinite.
3. Number of zones which are in total in the segment.
4. The (exact) number of zones each tote needs to visit.



MVA type	Iterative
Source type	Single
Number of zones	8
Number of zones to visit	2
Buffersize	10
Threshold	80
Mean picktime	30
Mean inter-arrival time	2
Mean time per conveyor	60
Places on each conveyor	1000

RESULTS:

```
Number of iterations: 3
Calculation time: 0.55 seconds

Utilization of zone 1:          0,8074
Utilization of zone 2:          0,8074
```

Figure 3.9: The user-interface of the Java application, with output

5. The buffersize of each zone.
6. The threshold or maximum number of totes allowed in the system.
7. The mean pick time of each operator of a zone.
8. The mean inter-arrival time.
9. The mean time per each conveyor piece.

There is also a possibility to give in the number of places on each conveyor, this is not yet of interest here because we do no take this in account. If all variables are correctly given the user can press the 'Calculate and Save' button and than the algorithm will be carried out with these settings and we use an accuracy of $\epsilon = 10^{-3}$. This means that the algorithm will stop if successive outcomes of the blocking probabilities differ less than ϵ. If the method is running the button will be gray and if it does not do anything the button will be orange. After some calculation time the results will appear in the white output area and also in an Excel-file which is in the same directory as the class file. The output area makes use of a scroll option, so if the data is too much for the current size the user can automatically scroll down and up to see all the output. If a new calculation is done the results will be overwritten with the new output. The output standard will contain the following results:

1. The number of iterations needed to calculate these values.
2. The calculation time of the method.
3. The utilization of each zone.
4. The mean buffersize of each zone.

5. The probability that a tote is blocked when visiting a zone.

6. The probability of a circulation.

7. The throughput per hour of each zone.

8. The total throughput per hour of the total segment.

Of course other performance measures, if wanted, can be printed to the output area, this is also one of the possible extensions we will describe in the next part. This is a short description of how to use and interpret this program. We will also describe how the most important parts of the method are implemented.

The implementation of the MVA is divided up in several Java-methods, which all calculate a separate performance measure, like mean lead time, mean number of totes etc. For the direct MVA this also includes solving the Markov chain and calculating the new visiting fractions at each zone, for iterative MVA this is done at the end of the iteration of the MVA. Solving the Markov chain is not as straightforward as we would like to think, we will describe how this is done more specifically because this will turn out to be the step which will take most of the time, especially setting up the matrix.

We showed in the previous part that it does not matter if a totes needs to visit exact v zones or an average of v zones. However, to implement this in Java is rather hard. The most difficult task is to determine which type needs to visit which zones and also the transition probability from this type to another type. We could not find a way to logically order the types when exact v zones are visited, recall that we used this to limit the total number of types so we would not have to work with $2^z - 1$ types. We did find a way to logically number the types when we consider all the $2^z - 1$ types; we used a binary representation of the zones to visit. Or in other words when there are 6 zones, tote type 7 has binary representation 111000. This means that this tote needs to visit zone 1, 2 and 3. This way we can also easily determine the transition probability between different types; the transition from type 7 to type 2 is the probability that the tote is blocked at zone 2 multiplied by the probability that it is not blocked (or served) at zone 1 and 3.

It seems that we did not use the observation (exactly vs mean) here of the previous section, but we take this into account when making up the matrix of transition probabilities. This matrix will contain a lot of zeros and therefore will be a sparse matrix. This will be of use when solving this system of linear equations, because solvers will have less trouble solving a system with many zeros than a system with all non-zero entries. To solve this system we used the Matrix Toolkit for Java (MTJ) package [13], which has included several direct and iterative solvers. The speed of the tool is also determined by the choice of the solver, so we tried every solver of the MTJ package and we have chosen to use the QMR solver. QMR is the Quasi-Minimal Residual method, which is a Krylov subspace method and solves asymmetric linear system, see also [17].

During the development of the tool we improved the speed of the tool constantly, by smart programming. We declared variables more logically and also we improved the method which fills the matrix for the Markov chain several times. The first time we ran an 11 zones example the calculation time was 26 minutes, after several improvements this is just 26 seconds.

Limitations and extensions

We will give some limitations of the Java application just described and also some extensions for the future. First we wil give some limitations.

Although the program uses the algorithm with MVA, there are still some problems with the virtual memory of Java. This because the data the program needs to remember grows rapidly when the number of zones is increasing. The default value of this memory is 64Mb, but it can be increased if necessary. This can be done by typing on the command line

```
java -Xmx...m <ClassName>
```

where on the dots the new value can be given. We have seen while running several examples that this will happen when the number of zones will be bigger than 11. In Eclipse, a program used by the Simulations Group, this virtual memory can easily be upgraded.
Depending on the computer the user is using and the choice of the other parameters we can say that when the number of zones is equal to 10 the calculation time of one iteration is approximately 4 seconds. When we use one zone more the calculation time grows with approximately a factor 4, so this gives us insight in the speed of the tool for bigger systems.
In the application we have identical zones, i.e. buffersizes and mean pick times are the same. We also have identical conveyors, which in reality also can be different. These are limitations for this application but this can be extended easily. We have chosen not to do this here, because most systems are symmetric and choices had to be made.

These extensions can be implemented by for instance using an input field for each zone where these variables can be changed for each zone, this can also be done for each conveyor part. As stated earlier we can also give more output if wanted.
Because we also implemented the direct MVA we can determine the performance measures for several thresholds in one run. We could automatically make a graphical representation of several performance measures as a function of this threshold. This way we can determine the best value for this parameter when other parameters are kept the same. The output is written to an Excel-sheet, so this can easily be done. However if one uses several times the iterative MVA, one can get same results in approximately the same time.
The most important extension is to make the application faster by smart programming and deleting variables (types of totes) we are not interested in. If one can find a logical ordening of the different types, this would help a lot. An interesting extension, especially for Vanderlande, is the use of a code generator which is currently developed at the Simulation Department. This means that when the layout of a zone is known, this can be presented in Java code. One can draw the layout of a system and the code generator switches this layout to the different arrays (like array of mean lead time etc.) and with these arrays the algorithm can be carried out. This is something which can be done on (very) long-term notice.

3.3.6 Results Single Server: 8 Zones

In this section we will present and discuss the results for the 8 zones case, which is a 'normal' sized segment. First we will vary the number of zones a tote needs to visit and for one specific case we will present more results. In the second part of this section we will take this specific case and we will vary some parameters to investigate the accuracy of the model. We will present only the results of the iterative MVA method, because the results of the iterative MVA and direct MVA differ only on the fourth or fifth decimal. This means that the results are roughly the same and because the iterative MVA is much faster we will only present the results of this method.

The performance measures we will give are the utilization, throughput per hour, mean buffersize and blocking probability. For all the results presented in this section the simulation errors are at most 0.005, 1, 0.1 and 0.3 for utilization, throughput, mean buffersize and blocking probability respectively. The mean number of totes in the source in all the scenarios is small, so the source will not be working all the time and we know that we are investigating interesting examples.

Variable number of zones to visit

In this part we will vary the (exact) number of zones a tote needs to visit. The other parameters are chosen to be

1. The buffers of all zones have size $N = 10$.
2. The maximum number of totes allowed in the segment is $C = 80$.
3. The mean pick time of each operator is $1/\mu = 30$ seconds.
4. The mean inter-arrival time is $1/\lambda = 2$ seconds.
5. The mean time on each conveyor piece is $1/\gamma = 60$ seconds.

The (exact) number of zones a tote needs to visit, denoted as v is the parameter we will alter. We will present the results for a number of visits of $1, 2, 4$ and 6 respectively. For the case that 2 zones are visited we will present the results for each zone.

For the other presented tables, results will be the average of the 8 zones, so we will not compare our results with each zone but with the average of these zones. We will give the maximum relative error we have seen when our method is compared with each zone separately, these are given in between brackets behind the relative errors.

In the first table, Table 3.8, we will present the results for the utilization, the throughput per zone and the total throughput of both the simulation and the iterative method. Overall we can see that results are very good. Obviously most of the output values increase when the number of zones that will be visited also increase, this because the fraction of totes that will visit a specific zone will also increase. The maximum relative error which we have seen is also acceptable for all performance measures that are studied here. The method gives good answers for the average of all zones but also for each zone separately. In the simulation results there are some fluctuations in the results as we will see in the next part. We do see a little increase in the error when the

v	ρ_{it}	ρ_{sim}	$\Delta\%$	δ_{it}	δ_{sim}	$\Delta\%$	δ_{it}^{total}	δ_{sim}^{total}	$\Delta\%$
1	0,548	0,549	0,0 (2,3)	65,8	65,9	0,2 (0,7)	526,2	527,0	0,2
2	0,807	0,799	1,1 (2,4)	96,9	95,6	1,4 (2,0)	387,6	382,2	1,4
4	0,919	0,891	3,1 (3,7)	110,3	106,6	3,4 (3,9)	220,5	213,2	3,4
6	0,948	0,916	3,4 (4,1)	113,7	110,0	3,4 (3,7)	151,6	146,6	3,4

Table 3.8: Simulation and iterative method results for utilization and throughput and the total throughput

number of visits is increasing. This effect can also be seen for other performance measures.

In Table 3.9 the results are given for the mean buffersize, the blocking probabilities and the time it takes in seconds for the method to generate these results, denoted as τ. Here we see for the mean buffersize very good results when the number of visits is 1 or 2, also the maximum relative error is acceptable. However when the number of visits is 4 or 6 we again see acceptable results and the maximum relative error is on the borderline of acceptable or not. The blocking probability was not calculated very accurate for segments with few zones as we have seen earlier. This is also the case here except when 1 zone is visited by each tote. Then the results are good however the maximum relative error is big again. For the other blocking probabilities we can see that relative errors are roughly 20% and the maximum relative error is 30%. These errors are big but such errors are made easily with these kind of values. An approximation of 3.2% by 2.4% seems good but gives a big relative error. The method takes little time to generate these results, just a couple of seconds and output is given. We can see that when more zones are visited the calculation time increase, this can be explained because also more iterations are needed to get the desired accuracy.

v	L_z^{it}	L_z^{sim}	$\Delta\%$	b_{it}	b_{sim}	$\Delta\%$	τ
1	1,17	1,18	0,7 (6,1)	0,001	0,001	4,1 (26,5)	0,3
2	3,19	3,19	0,1 (4,6)	0,024	0,032	25,2 (35,7)	0,5
4	5,09	4,84	5,3 (8,0)	0,085	0,110	22,3 (31,3)	1,4
6	5,84	5,50	6,3 (9,7)	0,128	0,153	15,8 (23,7)	1,6

Table 3.9: Simulation and iterative method results for mean buffersize and blocking probability

In Table 3.10 all the results are stated when the number of visits is 2 and we can see that the results for the first three performance measures are very good. The blocking probability is again not very accurate and the relative errors are roughly 20%. We can see that results can differ for each zone in the simulation, if we would simulate an infinite amount of time we will expect that these values will coincide with each other. In the next part we vary more parameters for the case that totes need to visit 2 zones to obtain more insights in the accuracy of the model.

Two zones to visit

We will discuss and present the results when the number of zones that are visited is 2. The parameters will be the same as in the previous part when we varied the number of zones that need to be visited, except for 3 parameters which are varied:

1. The buffersize (N): 5, 10.
2. The threshold (C): 80, 100.
3. The mean time on each conveyor ($1/\gamma$): 60, 90.

This leads to 8 scenarios which are discussed here.

In Table 3.11 the results for the utilization and throughput are given. Note that the simulation results are the average of the 8 zones, so the results for each zone separately can differ from the results given below. We can clearly see an obvious increase in performance measures when the

$i \rightarrow$	1	2	3	4	5	6	7	8
ρ_i^{it}	0,807							
ρ_i^{sim}	0,798	0,790	0,802	0,810	0,802	0,802	0,788	0,798
$\Delta_i\%$	1,2	2,1	0,6	0,3	0,7	0,6	2,4	1,1

δ_i^{it}	96,9							
δ_i^{sim}	95,6	95,0	95,9	95,8	95,6	95,8	95,0	95,8
$\Delta_i\%$	1,4	2,0	1,0	1,1	1,4	1,2	2,0	1,1

L_{zi}^{it}	3,19							
L_{zi}^{sim}	3,18	3,08	3,26	3,31	3,19	3,23	3,05	3,20
$\Delta_i\%$	0,4	3,6	2,2	3,6	0,0	1,1	4,6	0,4

$b_{it}^{(i)}$	0,024							
$b_{sim}^{(i)}$	0,030	0,027	0,036	0,033	0,031	0,034	0,030	0,037
$\Delta_i\%$	19,9	11,3	32,2	25,9	21,8	29,5	19,3	35,7

Table 3.10: Simulation and iterative method results per zone when number of visits is 2

buffersize increases, the threshold increases and the time on each conveyor decreases. The results for the utilization and throughput are very accurate, the maximum relative error that is given is only 4.0%. The same conclusion can be drawn when we take a look at the results for the mean buffersize, which can be found in Table 3.12 together with the results for the blocking probability. The maximum relative error for the buffersize is 5.1% which is still accurate. The results for the blocking probability are again worse than the other performance measures. When the buffer has a size of 5 totes the relative error is approximately 10%, which for the blocking probability is acceptable. For a buffer of size 10 the results are a bit worse; there relative errors of approximately 30% occur. The time to calculate these results with our method is again very small. We can see that when the probability of blocking increases the method needs more time (and iterations) to generate these results. In the next section we will do the same when the source is modelled as an infinite server, but first we will discuss an example of a ZPS for a customer of Vanderlande Industries.

N	C	$1/\gamma$	ρ_{it}	ρ_{sim}	$\Delta\%$	δ_{it}	δ_{sim}	$\Delta\%$	δ_{it}^{total}	δ_{sim}^{total}	$\Delta\%$
5	100	60	0,861	0,831	3,6 (4,0)	103,3	99,8	3,5	413,1	399,3	3,5 (3,8)
5	100	90	0,741	0,730	1,5 (1,6)	88,9	87,6	1,5	355,6	350,5	1,5 (0,2)
5	80	60	0,784	0,768	2,0 (2,7)	94,1	92,3	1,9	376,3	369,2	1,9 (2,5)
5	80	90	0,647	0,643	0,6 (1,4)	77,6	77,2	0,6	310,5	308,6	0,6 (1,4)
10	100	60	0,888	0,864	2,8 (3,4)	106,5	103,7	2,8	426,1	414,7	2,8 (3,3)
10	100	90	0,776	0,771	0,6 (1,3)	93,1	92,7	0,5	372,4	370,7	0,5 (0,7)
10	80	60	0,807	0,799	1,1 (2,4)	96,9	95,6	1,4	387,6	382,2	1,4 (2,0)
10	80	90	0,670	0,669	0,0 (1,6)	80,4	80,5	0,1	321,5	321,9	0,1 (0,5)

Table 3.11: Simulation and iterative method results for utilization and throughput per zone and the total throughput when number of visits is 2

N	C	$1/\gamma$	L_z^{it}	L_z^{sim}	$\Delta\%$	b_{it}	b_{sim}	$\Delta\%$	τ
5	100	60	2,68	2,55	5,1 (6,6)	0,186	0,207	10,2 (13,9)	1,0
5	100	90	1,92	1,90	1,1 (1,9)	0,089	0,098	8,8 (14,1)	0,8
5	80	60	2,16	2,11	2,1 (0,4)	0,114	0,127	9,9 (15,1)	0,8
5	80	90	1,46	1,46	0,2 (1,3)	0,047	0,052	8,5 (13,6)	0,7
10	100	60	4,46	4,29	3,9 (8,3)	0,061	0,085	28,2 (33,9)	0,7
10	100	90	2,83	2,83	0,1 (4,4)	0,017	0,021	16,1 (23,1)	0,6
10	80	60	3,19	3,19	0,1 (4,6)	0,024	0,032	25,2 (35,7)	0,5
10	80	90	1,86	1,88	1,0 (5,3)	0,004	0,005	3,4 (19,8)	0,5

Table 3.12: Simulation and iterative method results for mean buffersize and blocking probability when number of visits is 2

Simulation Project: Flex Logistics

Within Vanderlande Industries an example can be found of a ZPS which consists of one segment with 8 zones and 2 zones to visit for each tote. This is the system of Flex Logistics and some (but not much) data is available which we can use to compare with the results of our model. We have taken a closer look to this simulation and we can compare it with our model. The parameters for this system are extracted from the available data:

1. The buffersize of each zone is 10.
2. The threshold of the system is 100.
3. The mean inter-arrival time is 2 seconds.
4. The total circulation time on the conveyor is 4 minutes, so time per conveyor is $240/8 = 30$ seconds.

The available data mentions 2 choices for the mean pick time: $14,5$ seconds and 18 seconds. The output that was given is the utilization of each zone, the throughput per zone and the total throughput. We will compare the available data with our method to investigate the accuracy of the model.

Mean pick time: $14,5$ *seconds*
The first example we will investigate is when the mean pick time is $14,5$ seconds. In Table 3.13 the results for the performance measures can be found of the model and the available data for each zone. The total throughput of the model is 872 totes per hour and from the data we obtained a throughput of 871 totes per hour, the relative error of this results is $0,2\%$. We can see that the model is very accurate and that all relative errors are beneath 1%.

Mean pick time: 18 *seconds*
Here we will investigate the system when the mean pick time is 18 seconds. In Table 3.14 the results for the performance measures can be found of the model and the available data for each zone. The total throughput of the model is 736 totes per hour and from the data we obtained a throughput of 720 totes per hour, the relative error of this results is $2,2\%$. The relative errors of all the performance measures are roughly $2-3\%$ which is accurate.

$i \rightarrow$	1	2	3	4	5	6	7	8
ρ_i^{it}	0,878							
ρ_i	0,874	0,881	0,874	0,879	0,874	0,880	0,874	0,872
$\Delta_i\%$	0,5	0,3	0,5	0,1	0,5	0,2	0,5	0,7

δ_i^{it}	218,1							
δ_i	216,9	218,7	216,9	218,1	217,0	218,6	217,1	216,5
$\Delta_i\%$	0,5	0,3	0,5	0,1	0,5	0,2	0,5	0,7

Table 3.13: Comparison results for Flex Logistics, when mean pick time is $14,5$ seconds

$i \rightarrow$	1	2	3	4	5	6	7	8
ρ_i^{it}	0,920							
ρ_i	0,906	0,893	0,899	0,900	0,898	0,899	0,901	0,897
$\Delta_i\%$	1,5	3,0	2,3	2,2	2,4	2,3	2,1	2,6

δ_i^{it}	184,0							
δ_i	181,3	178,5	179,8	180,1	179,6	179,8	180,1	179,4
$\Delta_i\%$	1,5	3,0	2,3	2,2	2,4	2,3	2,1	2,6

Table 3.14: Comparison results for Flex Logistics, when mean pick time is 18 seconds

We have seen that the results of our model coincided with the (little) available data of a real system. If there would be more data we could investigate the accuracy for more values of the given parameters. Together with the simulation results we generated on our own, we think our model describes reality accurate and therefore is a useful model for Vanderlande Industries.

3.3.7 Results Infinite Server: 8 Zones

Here we consider the source as an infinite server, recall that this is the case when we see the segment as a part of a larger (multi-segment) ZPS. For this comparison between the method and the simulation we keep the same parameters as for the single server case, also for the number of zones to be visited. The only variable we need to alter is the 'mean inter-arrival time'. Here this is the mean time a tote is spending in the rest of the system, before returning to this segment. For this problem we will set $1/\lambda = 100$ seconds. The results will be presented in the same order as in the previous section; first we will give all the results when the number of zones visited is variable and finally we will present the results when 2 zones are visited. By the same reasoning as before we will only present the results for the iterative MVA. The simulation error for all the results will not be bigger than 0.006 for the utilization, 1 for the throughput, 0.2 for the buffersize and finally 0.3 for the probability of blocking.

Variable number of zones to visit

The utilization and throughput results can be found in Table 3.15, where we can see that the iterative method gives very good results for both of the performance measures and again our method gives most of the time an upper bound. The maximum relative error we see here is only $4,3\%$

which is also very low and acceptable. The same can be said about the mean buffersize results,

v	ρ_{it}	ρ_{sim}	$\Delta\%$	δ_{it}	δ_{sim}	$\Delta\%$	δ_{it}^{total}	δ_{sim}^{total}	$\Delta\%$
1	0,472	0,473	0,2 (1,4)	56,6	56,7	0,2	453,0	453,7	0,2 (1,1)
2	0,753	0,749	0,5 (1,3)	90,3	89,9	0,4	361,5	359,7	0,5 (1,2)
4	0,900	0,878	2,5 (3,0)	108,0	105,4	2,5	216,1	210,7	2,5 (2,8)
6	0,938	0,907	3,4 (4,3)	112,5	109,1	3,2	150,0	145,4	3,2 (3,9)

Table 3.15: Simulation and iterative method (infinite source) results for utilization and throughput and the total throughput

here the maximum relative error is however $9,1\%$. The blocking probability results are, just like the single server case, not as accurate as wanted. Here the relative errors are approximately 20% but again it is not that bad that a probability of 9% is approximated by 7%. Just like for the single server source, we will give all the results when 2 zones are visited. These results can be found in Table 3.17. We can see that the simulation results for the zones separately differ, while the iterative method gives one answer for all the zones. We can see in Table 3.17 that the maximum relative error for the blocking probability is $30,7\%$, while the relative error for the mean results is 'only' $19,3\%$. When the blocking probability increases we also see (Table 3.16) an increase in the calculation time. In the next part we vary more parameters for the case that totes need to visit 2 zones to obtain more insights in the accuracy of the model when we consider the infinite source case.

v	L_z^{it}	L_z^{sim}	$\Delta\%$	b_{it}	b_{sim}	$\Delta\%$	τ
1	0,88	0,88	0,6 (3,9)	0,000	0,000	16,5 (40,8)	0,3
2	2,57	2,59	0,7 (3,6)	0,013	0,016	19,3 (30,7)	0,6
4	4,69	4,52	3,7 (7,5)	0,069	0,087	20,8 (29,8)	1,0
6	5,56	5,27	5,5 (9,1)	0,107	0,134	20,3 (31,4)	1,5

Table 3.16: Simulation and iterative method (infinite source) results for mean buffersize and blocking probability

Two zones to visit

When we consider totes that will visit 2 zones we will vary the same parameters with the same values as was done for the single server source. The results of the 8 different scenarios can be found in Table 3.18 and Table 3.19, where utilization and throughput results can be found and mean buffersizes and blocking probabilities, respectively. The results for utilization and throughput are very good; most of the relative errors are approximately 1% while the maximum relative error is $2,9\%$. The same errors can be seen for the mean buffersize.
As we have seen before, the blocking probabilities are approximated worse than the other performance measures. For a buffer of size 5 the relative errors are roughly 10%, while for a buffer of size 10 these errors are 20%. Overall we can say that the method works well for both the single as the infinite server source and also it generates these results very fast.

$i \rightarrow$	1	2	3	4	5	6	7	8
ρ_i^{it}	0,753							
ρ_i^{sim}	0,748	0,744	0,743	0,753	0,746	0,750	0,753	0,755
$\Delta_i\%$	0,6	1,1	1,3	0,1	0,9	0,4	0,1	0,3
δ_i^{it}	90,3							
δ_i^{sim}	90,2	89,8	89,8	90,0	89,6	89,5	90,4	90,1
$\Delta_i\%$	0,1	0,6	0,6	0,3	0,8	0,9	0,1	0,3
$L_{z,i}^{it}$	2,57							
$L_{z,i}^{sim}$	2,58	2,55	2,53	2,61	2,54	2,59	2,67	2,65
$\Delta_i\%$	0,4	0,9	1,6	1,6	1,1	0,6	3,6	3,0
$b_{it}^{(i)}$	0,013							
$b_{sim}^{(i)}$	0,015	0,016	0,014	0,016	0,015	0,016	0,018	0,016
$\Delta_i\%$	13,2	19,6	12,7	19,4	13,6	20,1	30,7	22,1

Table 3.17: Simulation and iterative method (infinite source) results per zone when number of visits is 2

N	C	$1/\gamma$	ρ_{it}	ρ_{sim}	$\Delta\%$	δ_{it}	δ_{sim}	$\Delta\%$	δ_{it}^{total}	δ_{sim}^{total}	$\Delta\%$
5	100	60	0,823	0,801	2,8 (3,2)	98,8	96,0	2,9	395,0	383,9	2,9 (3,1)
5	100	90	0,701	0,693	1,2 (1,6)	84,1	83,2	1,1	336,5	332,8	1,1 (1,5)
5	80	60	0,734	0,723	1,5 (2,2)	88,1	86,8	1,4	352,2	347,4	1,4 (1,9)
5	80	90	0,602	0,599	0,5 (1,1)	72,2	72,0	0,3	288,8	288,0	0,3 (1,0)
10	100	60	0,847	0,832	1,8 (2,6)	101,7	99,8	1,9	406,8	399,2	1,9 (2,3)
10	100	90	0,729	0,728	0,2 (1,3)	87,5	87,4	0,2	350,0	349,4	0,2 (1,1)
10	80	60	0,753	0,749	0,5 (1,3)	90,3	89,9	0,4	361,3	359,7	0,5 (1,2)
10	80	90	0,618	0,618	0,0 (1,0)	74,1	74,2	0,2	296,5	297,0	0,2 (0,8)

Table 3.18: Simulation and iterative method (infinite source) results for utilization and throughput per zone and the total throughput when number of visits is 2

3.4 Summary and Conclusion

In this chapter we have extended the model of the previous chapter, by first considering 2 zones followed by multiple zones. The final model stated here describes a segment with multiple zones, this segment can be part of a larger ZPS (infinite source) or it can be a ZPS of just one segment (single source). It is not very different from the one zone model; we model the ZPS as a closed queueing network where totes are routed to the several zones with Markovian routing. We again use the jump-over blocking model, which gives us the limiting probabilities.

The most important extension is the use of different types of totes, because totes can visit different zones and this needs to be included in the model to obtain the visit frequencies at each zone. These types describe the states of a Markov chain where the transition probabilities are determined by the probability that a tote is blocked at a certain zone. We have to notice that the number of types is growing exponentially and this can play an important role in the calculation

N	C	$1/\gamma$	L_z^{it}	L_z^{sim}	$\Delta\%$	b_{it}	b_{sim}	$\Delta\%$	τ
5	100	60	2,41	2,33	3,3 (4,3)	0,148	0,168	12,2 (15,1)	0,7
5	100	90	1,71	1,70	0,5 (1,5)	0,069	0,076	9,3 (11,6)	0,8
5	80	60	1,87	1,85	1,2 (3,6)	0,084	0,094	10,7 (16,9)	0,8
5	80	90	1,27	1,28	0,0 (2,2)	0,034	0,036	5,5 (13,0)	0,7
10	100	60	3,77	3,72	1,3 (4,8)	0,039	0,055	28,6 (35,5)	0,7
10	100	90	2,36	2,38	1,0 (3,5)	0,010	0,011	10,8 (13,7)	0,5
10	80	60	2,57	2,59	0,7 (1,6)	0,013	0,016	19,3 (22,1)	0,6
10	80	90	1,53	1,54	0,7 (2,3)	0,002	0,002	0,3 (22,7)	0,6

Table 3.19: Simulation and iterative method (infinite source) results for mean buffersize and blocking probability when number of visits is 2

time of the model.

We have the limiting probabilities of a system with multiple zones, which is of course very nice. We can use these limiting probabilities to calculate the performance measures, however we also have to calculate the normalizing constant. For systems with more than 2 zones this will take a very long time. Therefore we will use Mean Value Analysis (MVA) to calculate performance measures, this is also sufficient because we are only interested in mean performance measures, like utilization. With this new way of determining performance measures we can generate the same results in several seconds. However not for all systems; when the number of zones is bigger than 10 the calculation time is rather big.

As said earlier the number of types of totes is growing rapidly when the number of zones is increasing, this will also have its effect on the calculation time of the method. We have to reduce the number of types we are considering. With simulation we have determined that visiting exactly v zones gives the same results as visiting on average v zones. This will decrease the number of types we need, while the results will be the same. This will help in our analysis and also to reduce the calculation time of our model. We have implemented the model with MVA in Java to generate results for the main performance measures; utilization, throughput, mean buffersize and blocking probability at each zone. The application is rather slow when more than 10 zones are considered and this is due to the fact that we still have to use many types of totes. In the application we use all $2^z - 1$ types, when z is the number of zones. It seems now that we do not use the observation (exact vs. mean visits) we have seen earlier, but this is included in the transition probabilities between types of totes. We have seen that solving the Markov chain and especially generating the transition matrix, is the limiting factor of the application. Most of the calculation time can be won there. This can be done if we can find a better way to order the types. The Java application (uses an user interface) and gives the most important performance measures of each system. These results are directly presented in this interface, but are also saved to an Excel-file. There are still several extensions possible, like for example more input/output possibilities.

We also made a simulation to compare simulation results with the results of our application. The segment with 2 zones is thoroughly investigated and gives good results. This system is much more of interest than the segment with one zone, but still not entirely applicable to systems of Vanderlande. Therefore we investigated a system with one segment and 8 zones, which is a 'normal' sized segment. First we varied the (exact) number of zones a tote needs to visit. For the case that 2 zones are visited, we have compared 8 scenarios where the buffersize, threshold and mean time on conveyor are varied. This is done for both the single source and the infinite source.

In general the results for the single source are very accurate for utilization and (total) throughput. The mean buffersize is approximated well and here we see a maximum relative error of $8,3\%$. The blocking probabilities are not as good approximated as desired; the relative error is roughly 10% with peaks of 30%. This single server case is compared with an example of a ZPS for a customer of Vanderlande Industries. The little available data is compared with the results of our method and relative errors. The maximum relative error we observe is only 3%, which is of course very good.
The same is done for the infinite server, only here no example from the reality is given. Here the method gives again very good results for the utilization and (total) throughput. The same results can be seen for the mean buffersize, where relative errors are all beneath 5%. The relative errors for the blocking probabilities are of the same order as for the single server case.

When no gaps are considered we have developed a method that generates good results for almost all performance measures, very fast. In the next chapter we will include these gaps in our model, to make it more realistic.

Chapter 4

Hindering at Conveyor

An important influence on the utilization of an operator is, together with (not enough) supply of totes, that the operator can not place the totes back on the conveyor because it is occupied by other totes. We have neglected this until now for simplicity. In this chapter we will take this hindering into account and we will try to adjust the performance measures, especially the utilization. In the method presented so far one could increase the total number of totes allowed in the system and the utilization will increase to 1. In reality this will not be the case, there will be a optimum value for this threshold value. If the threshold is very large, the operator has little space to place the totes back and his effective utilization will be low. We here say effective because he will be 'busy' waiting to place the tote on the conveyor, but this is not part of an operators utilization.

We have seen that our models give results that exceed the results of the simulation, so we have to find an adjustment that decreases our utilization results. First we will present the adjustment and we will give examples why this intuitively corresponds with the real situation. We will do this here for the single source, 1 zone model, to make it more understandable. The approach will work in the same way for more zones and for the infinite source model. For more zones the effect of hindering will probably be bigger. We present the results for the iterative method for the single source and the infinite source case. Finally we consider the Flex Logistics ZPS project done within Vanderlande.

4.1 Introduction

The most important thing we use in this adjustment is that we know the limiting probabilities of the system. Recall that this is for the 1 zone model:

$$P(n_1, n_2, n_3) = d(\frac{1-p}{\lambda})^{n_1} \frac{1}{n_2!} (\frac{1}{\gamma})^{n_2} (\frac{1}{\mu})^{n_3}, \tag{4.1}$$

where p was the blocking probability. This also implies that we know the probability of the number of totes on the conveyor. Let us assume that the conveyor has capacity K, which may be smaller than C. How we define this capacity will be discussed later.

Let us consider the operator in the zone and let us look at the moment when a tote is hindered and especially at the probability a tote is hindered. This is the variable we need to find so that we can adjust our utilization.

If we know this hindering probability (calculated in section 4.3), denoted as

$$P(\text{hindered at conveyor}),$$

we can use this probability to determine the mean delay. Here the mean delay is defined as the time an operator on average has to wait before he can place the tote back on the conveyor. Now assume we know the mean delay which will be added to the time the tote was already in the zone for service, which we denote as

$$T := \mathbb{E}[\text{delay}].$$

This delay will be determined in section 4.4 and will depend on the hindering probability. The average total time an operator is occupied by a tote is now given by

$$\mathbb{E}[\text{working time}] = \frac{1}{\mu} + T,$$

while the effective working time is just given by the mean pick time. This means that the fraction the operator is working effectively on a tote is

$$f_a := \frac{1/\mu}{1/\mu + T} = \frac{1}{1 + \mu T}. \tag{4.2}$$

Note that $0 < f_a \leq 1$ so this fraction will be our factor which will decrease our utilization. We will define our adjusted utilization (ρ_a) as

$$\rho_a := f_a \rho, \tag{4.3}$$

and in the next part we will give some intuitive arguments why this factor will work and after that we will say some more about the capacity of a conveyor.

4.1.1 Intuitive arguments

To intuitively see that this adjustment will work we will look at some special cases, from which we know what the adjustment should be. First we will consider the case when we adjust the maximum number of totes on the conveyor, followed by the mean pick time and finally the speed of the conveyor.

Maximum number of totes

First we assume that $K = 0$, so no tote is allowed on the conveyor. Then we know that the probability of hindering will be 1 by definition and we would expect a utilization which is approximately 0. If the hindering probability is indeed 1 the mean delay will be very big. Now let us take a look at our adjusted utilization, this becomes by definition

$$\rho_a = \frac{1}{1 + \mu_1 T}\rho,$$

which for very big T will be close to 0. So for $K = 0$ the adjustment seems to hold.

In the other extreme case of $Kk = \infty$, the conveyor has an infinite length or infinite capacity. In this situation no hindering can occur and so this probability will be 0 and also the mean delay wil be 0. The utilization should be the same as before and it is easy to see from (4.3) that this will be the case.

Mean pick time

When the mean pick time is close to 0, i.e. μ is close to infinity, then the hindering probability will be high. When the number of totes in the system is big, we expect the real utilization to be low. The operator has to wait relatively long to put the tote on the conveyor. From (4.3) we can see that

$$\lim_{\mu \to \infty} \rho_a = 0, \tag{4.4}$$

which seems to hold with what we would expect.

The other extreme case is when the operator is working a long time on one tote, so $\mu = 0$. Than there will be almost no hindering and the operator will be working all of the time, because the mean pick time is also infinite. Again using (4.3) we see

$$\lim_{\mu \to 0} \rho_a = \rho = 1, \tag{4.5}$$

which is a similar result to what we expect.

Mean time on conveyor

If an operator is working very fast, i.e. μ is very big, and the conveyor is very slow then an operator has to wait relatively a long time before he can push the tote on the conveyor. In this case we would have a low utilization because the operator is not actually working most of the time but waiting before totes on the conveyor passed. We can also see this because f_a is a decreasing function in T, the mean time until a new free space.

4.2 Number of Places

One would say that the number of places which we must consider is the total places available at the conveyor, K. This is not completely true, because we now consider gaps. The standard gapsize is a gap of 1 time the length of a tote, which will be the value chosen throughout this chapter. The total number of places for this choice now seems given by $K/2$. However the situation can occur when there are two places between two totes, than no tote can be placed in between and the number of places will be $K/3$. The number of places which one tote 'uses' will be somewhere in between. We have to determine the average number of totes on the conveyor in order to be able to calculate the hindering probability and mean delay correctly. We will do this by looking at the placing on the conveyor as a discrete and as a continuous problem. In this section we wil first describe the discrete approach followed by the continuous approach.

4.2.1 Discrete case

Let us consider a conveyor with K places where we will uniformly pick a spot to place a tote. The places next to this place may not contain a tote anymore because of the gap of length 1, so these places are also taken. Again we will uniformly pick a spot and if this spot was already taken we will forget this and try again. Otherwise we will do the same as before and regard this place and places around it as taken. The question now is how many times we have to try before all places are taken, i.e. how many totes are on average on a conveyor of capacity K.

This will be done recursively and the scheme is as follows:

$$\begin{aligned} x_0 &= 0, && (4.6) \\ x_1 &= 1, && (4.7) \\ x_k &= 1 + \frac{2}{k} \sum_{i=0}^{k-2} x_i, \quad k \geq 2, && (4.8) \end{aligned}$$

where x_k is the average number of totes on a conveyor of length k.

The first two values are clear by definition and we will explain the recursive part with an example. First we will consider a conveyor of an odd length of 7:

With probability $\frac{1}{7}$ the first (or last) place will be picked first and this will leave us a conveyor of length $7 - 2 = 5$, because we must also consider the gap. We can do the same with the second (or 6th) place and this will leave us a conveyor of length $7 - 3 = 4$. When the third (or 5th) place is picked, again with probability $\frac{1}{7}$, this will give us 2 conveyors; one of length 1 and one of length 3. Finally with probability $\frac{1}{7}$ we will pick the place in the middle and this will leave 2 conveyors of equal length 2. Eventually we will have the following relation:

$$\begin{aligned} x_7 &= 1 + \frac{2}{7}x_5 + \frac{2}{7}x_4 + \frac{2}{7}(x_1 + x_3) + \frac{1}{7}(2 \cdot x_2) \\ &= 1 + \frac{2}{7}\sum_{i=1}^{5} x_i \\ &= 1 + \frac{2}{7}\sum_{i=0}^{7-2} x_i. \end{aligned}$$

The same can be done for a conveyor of even length 8. Only here there is no middle place and we will eventually have with probability $\frac{1}{8}$ that the fourth (or fifth) place was picked and we are left with two conveyors; one of length 2 and one of length 3. This will give us

$$\begin{aligned} x_8 &= 1 + \frac{2}{8}x_6 + \frac{2}{8}x_5 + \frac{2}{8}(x_1 + x_4) + \frac{2}{8}(x_2 + x_3) \\ &= 1 + \frac{2}{8}\sum_{i=1}^{6} x_i \\ &= 1 + \frac{2}{8}\sum_{i=0}^{8-2} x_i. \end{aligned}$$

In general we can state equation (4.8), which gives an expression for the average number of totes on a conveyor of length k.

Here we considered the conveyor with two open endings but we could also consider the conveyor as a closed loop. Than we can determine the mean number of totes on a closed conveyor (y_k) easily with

$$y_k = 1 + x_{k-3}, \tag{4.9}$$

because the first tote uses 3 places and then we are left with a conveyor with two open endings with 3 places less.

We are interested in the limit $c := \lim_{k\to\infty} x_k/k$, which is the percentage of total capacity that will be used. This limit will have the same value as the limit of y_k/k as can be seen easily. We will determine this value c in the following theorem.

Theorem 4. *The factor x_k/k converges for $k \to \infty$ to $c = \frac{1}{2}(1 - \mathrm{e}^{-2})$*

Proof. Here we use equation (4.8) and multiply this by k to obtain

$$kx_k = k + 2\sum_{i=0}^{k-2} x_i, \quad k \geq 2 \tag{4.10}$$

Define the generating function of x_k as

$$G(z) = \sum_{k=0}^{\infty} x_k z^k,$$

and use this in our equation to simplify

$$\begin{aligned}
\sum_{k=2}^{\infty} kx_k z^k &= \sum_{k=2}^{\infty} kz^k + 2\sum_{k=2}^{\infty}\sum_{i=0}^{k-2} x_i z^k \\
z\frac{\mathrm{d}}{\mathrm{d}z}\Big(\sum_{k=2}^{\infty} x_k z^k\Big) &= \sum_{k=2}^{\infty} kz^k + 2\sum_{i=0}^{\infty} x_i \sum_{k=i+2}^{\infty} z^k \\
z\Big(\frac{\mathrm{d}G(z)}{\mathrm{d}z} - \frac{\mathrm{d}}{\mathrm{d}z}z\Big) &= \sum_{k=2}^{\infty} kz^k + 2\sum_{i=0}^{\infty} x_i \frac{z^{i+2}}{1-z} \\
zG'(z) - z &= z\frac{\mathrm{d}}{\mathrm{d}z}\Big(\frac{1}{1-z}\Big) - z + \frac{2z^2}{1-z}G(z) \\
G'(z) &= \frac{2z}{1-z}G(z) + \frac{1}{(1-z)^2}.
\end{aligned}$$

This final equation is an inhomogeneous difference equation, where the solution is given by

$$G(z) = C(z)\frac{\mathrm{e}^{-2z}}{(1-z)^2}, \tag{4.11}$$

where $C(z)$ is a function which depends on z and we wil determine this function using expression (4.11) for $G(z)$. We obtain

$$\begin{aligned}
G'(z) &= C'(z)\frac{\mathrm{e}^{-2z}}{(1-z)^2} + C(z)\frac{\mathrm{e}^{-2z}}{(1-z)^2}\Big(-2 + \frac{2}{1-z}\Big) \\
&= C(z)\frac{\mathrm{e}^{-2z}}{(1-z)^2}\frac{2z}{1-z} + C'(z)\frac{\mathrm{e}^{-2z}}{(1-z)^2} \\
&= \frac{2z}{1-z}G(z) + C'(z)\frac{\mathrm{e}^{-2z}}{(1-z)^2} \\
&\Leftrightarrow \quad C'(z) = \mathrm{e}^{2z} \\
&\Leftrightarrow \quad C(z) = \frac{1}{2}\mathrm{e}^{2z} + A
\end{aligned}$$

Finally we have to determine the constant A, which will be given by solving $G(0) = 0$, this gives us $A = -\frac{1}{2}$. Eventually we have an expression for the generating function

$$G(z) = \frac{1 - \mathrm{e}^{-2z}}{2(1-z)^2}. \tag{4.12}$$

We want to determine the limit of x_k/k when k is big enough, which is defined as c.

$$c = \lim_{k\to\infty} x_k/k = \lim_{z\uparrow 1}(1-z)^2 G(z) = \frac{1}{2}(1 - \mathrm{e}^{-2}) \approx 0,43. \tag{4.13}$$

This ends the proof □.

We now know the factor which we should use to determine the number of places to calculate with. If the total capacity of a conveyor is K we will use in our calculations a number of places of $k = \frac{1}{2}(1 - \mathrm{e}^{-2})K$. We could also determine the factor for each value of K, by using Taylor expansion for the generating function. Which can be seen as the product of $1 - \mathrm{e}^{-2z}$ and $\frac{1}{(1-z)^2}$ and we can determine the coefficient x_K of z^K by polynomial multiplication. This will not be done here. As we can see in Figure 4.1, x_n/n converges rapidly to the calculated factor $c = \frac{1}{2}(1 - \mathrm{e}^{-2})$, so we will not determine each coefficient but just use c to calculate with.

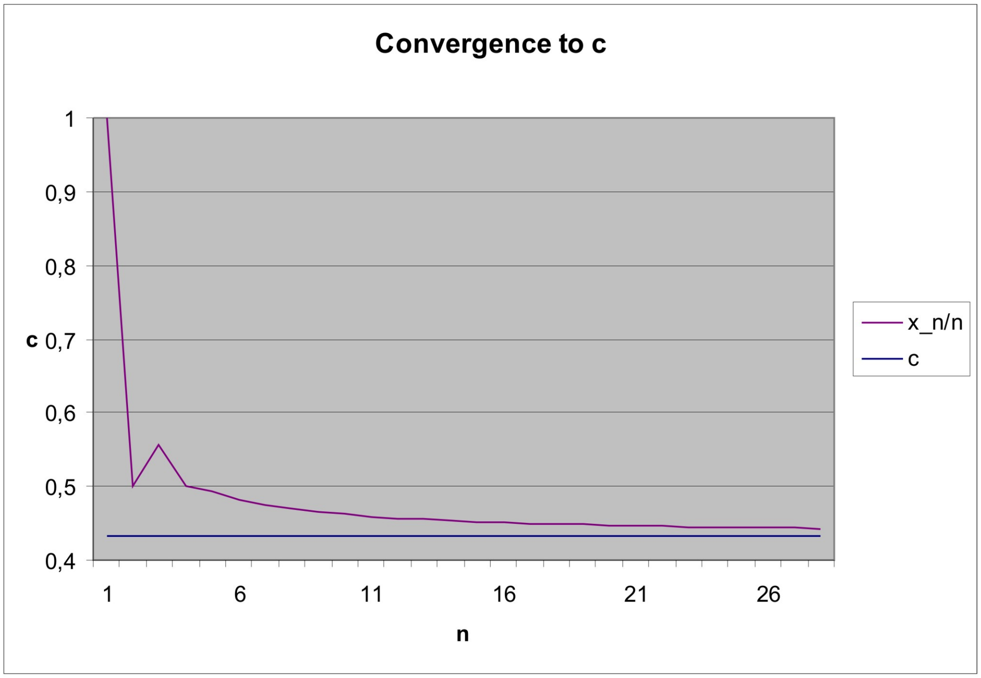


Figure 4.1: The convergence of x_n/n to the calculated factor c.

For other gap lengths this factor can be determined by using the same recursive technique. The following recursive scheme needs to be solved, when gap size is w times the length of a tote:

$$\begin{aligned} x_0 &= 0, & (4.14)\\ x_i &= 1, \;\; i = 1, \ldots, w, & (4.15)\\ x_k &= 1 + \frac{2}{k} \sum_{i=0}^{k-w-1} x_i, \;\; k \geq w+1, & (4.16) \end{aligned}$$

As before, introducing generating functions we can derive the following inhomogeneous differential equation:

$$G'(z) = \frac{2z^w}{1-z} G(z) + \frac{1}{(1-z)^2}. \tag{4.17}$$

This will not be done here. Once we found the solution of this differential equation we can determine the limit of x_n/n when $n \to \infty$.

4.2.2 Continuous case

When regarding the continuous case, there is much resemblance with the parking problem [16]. This parking problem regards the parking of cars on a one dimensional line and the mean occupation of this line is determined. Cars may be as close to each other as is possible, in this case there is a occupation rate of 100%. In the worst case scenario there are gaps between all cars and these gaps are just not wide enough to fit in a car. The occupation rate for this worst case scenario is approximately 50%. One could guess the mean occupation rate should be in the middle of these two values, because both scenarios are equally likely. We would have a occupation of 75%.
Renyi considered this problem and he determined, what is nowadays known as Renyi's parking constant. This constant is close to the number we just guessed, but it is a bit smaller than we determined: $\rho_r = 74,759\ldots\% \approx 75\%$.

The situation we are considering has many similarities with this problem, but we require a minimum distance between the totes. The parking problem can be used here, because we can now regard a car as a tote with its gap. In the old problem we have an occupation of 75%, this implies for our problem an occupation of $75/2 = 37,5\%$. This can also be done by looking at the best and worst case scenario. Here the worst case scenario is different than in the discrete case and this will also cause a difference in the factor. The best case scenario is that all cars (totes with gap) are placed next to each other, contributing to a occupation rate for the totes of 50%. The worst case scenario is the case that the gaps between totes are just too small to place a tote. This means for a gap of size 1 that the number of free places between two totes is a little bit smaller than 3 places, because the tote must have enough distance to both totes. This will cause a occupation of approximately 25%. By same reasoning as before we therefore get an occupation of $\frac{1}{2}(50+25) = 37,5\%$, which is the same as calculated with Renyi's parking number. The factor c, denoted in the previous section is therefore $c = \frac{3}{8}$.

We can also consider this for different gap lengths, which is rather straightforward when regarding the Renyi number. We will consider a 'car' as a tote with its gap length of w times the tote length. Therefore the factor we will use for different gapsizes will be

$$c = \frac{3}{4} \frac{1}{w+1}. \tag{4.18}$$

The continuous case is the case which describes the situation of the simulation best and therefore will probably be better than the discrete case, therefore the factor c of (4.18) will be used in future calculations.

4.3 Hindering Probability

The calculation of the hindering probability is the next important part in the calculation of the adjustment factor and this will be done here. We will do this in parts which depend on the number of totes that are on the conveyor. An operator has to wait if there is one tote in front of the operator.

Let us therefore take a closer look at the zone and see when the operator is hindered, this can be seen in Figure 4.2. In the calculations that will follow we assume that totes are equally distributed on the conveyor. In Figure 4.2 we can see when a tote is hindered, namely when the first place is occupied with a tote. Also we can see that when a tote is not hindered the rest of the conveyor can be occupied with a lot of totes. Crucial here is that a tote is hindered if the first place is occupied and that it can happen that many places on the conveyor are occupied but no hindering occurs. Here a black square represents the mean number of places one tote uses ($= 1/c$) and a white square is an empty spot, so a tote could be placed there. The number of places is therefore defined as $k = cK$. We will calculate the probability of hindering when there is a given number of totes on the conveyor.

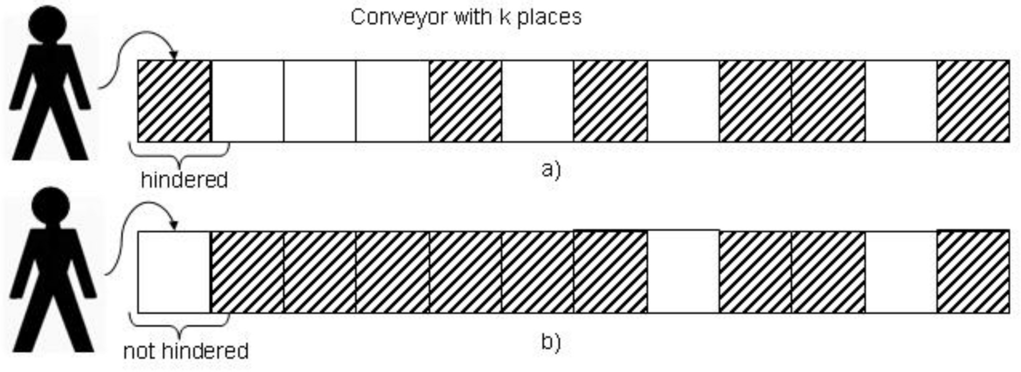


Figure 4.2: The hindering principle when a tote is served and needs to pushed back on the conveyor: tote is hindered (a), tote is not hindered (b)

4.3.1 Zero totes

If 0 totes are on the conveyor, no hindering can occur so this will not contribute to the hindering probability by definition.

4.3.2 One tote

The probability that there is one tote on the conveyor equals

$$P(n_2 = 1) := \sum_{n_1} \sum_{n_3} P(n_1, 1, n_3), \text{ with } \sum_{i=1}^{3} n_i = C - 1.$$

In total there are k places for this single tote where it can be, but only one will cause a hindering of the tote which just has service. We therefore have to divide this probability by k to obtain the contribution to the probability the tote is hindered.

4.3.3 Two totes

The probability that there are 2 totes on the conveyor we are considering is given by

$$P(n_2 = 2) := \sum_{n_1} \sum_{n_3} P(n_1, 2, n_3), \text{ with } \sum_{i=1}^{3} n_i = C - 1.$$

In total there are $\binom{k}{2}$ possibilities to allocate the two totes on a place of the conveyor. We are only interested in the case that there is one tote at the beginning and it does not matter where the second one is located.
Let us place the first tote on the first place, then there are $k-1$ places left for the second tote, i.e. $\binom{k-1}{1}$. The contribution to the hindering probability will therefore be

$$\frac{P(n_2 = 2)}{\binom{k}{2}/k-1} = \frac{(k-1) \cdot P(n_2 = 2)}{\binom{k}{2}} = \frac{2}{k} P(n_2 = 2). \tag{4.19}$$

4.3.4 i totes

We can get a general expression for allocating i totes on the conveyor, so that the first place is occupied. This will be done in the same manner as we have done for 2 totes.
In total there are $\binom{k}{i}$ possibilities to allocate all i totes on a place of the conveyor. We pick one tote and place this one on the first place of the conveyor, than there are $k-1$ places left to allocate the other $i-1$ totes. This can be done in $\binom{k-1}{i-1}$, so the contribution to the hindering probability will be in the same manner as in (4.19).

$$\frac{\binom{k-1}{i-1} \cdot P(n_2 = i)}{\binom{k}{i}} = \frac{i}{k} P(n_2 = i) \quad i = 0, \ldots, C-1. \tag{4.20}$$

We have now calculated the contribution to the hindering probability when there are i totes on the conveyor. Summing these gives us the probability of hindering which is given as

$$P(\text{hindered at conveyor}) = \sum_{i=0}^{C-1} \frac{i}{k} P(n_2 = i) = \frac{1}{k} \sum_{i=0}^{C-1} i P(n_2 = i) = \frac{L_c}{k}, \tag{4.21}$$

where L_c is the mean number of totes on the conveyor when there are in total $C-1$ totes in the system. It could happen for systems that $k < L_c$, so to be complete we have

$$P(\text{hindered at conveyor}) = \min[\frac{L_c}{k}, 1], \tag{4.22}$$

We can see here that when the number of places on the conveyor goes to infinity the hindering probability will go to 0, or i.e.

$$\lim_{k \to \infty} P(\text{hindered at conveyor}) = 0,$$

which agrees with reality.
Also consider the system where many totes are travelling in, this means that the mean number of totes on the conveyor will be close to k, i.e.

$$\lim_{L_c \uparrow k} P(\text{hindered at conveyor}) = 1,$$

which also agrees with what we would expect in reality.

The expression we obtained is intuitively also clear, because we assume that totes are equally distributed over the conveyor we should expect that a place is occupied with a probability of L_c/k. We are only interested in the first place of the conveyor, the other places may or may not be occupied. This means that the hindering probability is equal to the probability that an arbitrary place is occupied.

We now have an expression for the hindering probability at the conveyor in terms of the capacity of the conveyor (k) and the mean number of totes on the conveyor. In the following section we will calculate the mean delay (T) which depends on this hindering probability. This is the only unknown factor in f_a.

4.4 Mean Delay

The mean delay depends on the speed of the conveyor, the capacity of the conveyor and the number of totes for which the operator has to wait. Let us assume that the number of successive occupied places, the operator has to wait for, have a geometric distribution. If an operator has to wait for i totes than we know this will happen with probability

$$(1 - P(\text{hindered at conveyor})) \cdot P(\text{hindered at conveyor})^i, \quad i \geq 0.$$

The conveyor has capacity k and the time it takes to travel over the conveyor is $\frac{1}{\gamma}$, this means that the time it takes for i gaps to pass by is defined by

$$\frac{i(1/\gamma)}{k}, \quad i \geq 0.$$

Note that the number of places can be infinite, because totes can recirculate and therefore an infinite number of places can pass before a tote can be placed on the conveyor.

The mean delay is now determined by summing these terms to obtain

$$
\begin{aligned}
T &:= \sum_{i=0}^{\infty}(1-P(\text{hindered at conveyor}))\cdot P(\text{hindered at conveyor})^i\frac{i(1/\gamma)}{k}\\
&= \frac{(1-P(\text{hindered at conveyor}))}{k\gamma}\sum_{i=0}^{\infty} iP(\text{hindered at conveyor})^i\\
&= \frac{(1-P(\text{hindered at conveyor}))}{k\gamma}\cdot P(\text{hindered at conveyor})\frac{1}{(1-P(\text{hindered at conveyor}))^2}\\
&= \frac{1}{k\gamma}\cdot\frac{P(\text{hindered at conveyor})}{1-P(\text{hindered at conveyor})},\\
&= \frac{1}{k\gamma}\frac{L_c}{k-L_c}.
\end{aligned}
$$

We demand here that if $k < L_c$ we assume $k = L_c$, which coincided with the definition of the hindering probability in 4.22. The expression for the adjusted utilization now simplifies to (4.23).

$$
\rho_a = (1+\frac{\mu}{k\gamma}(\frac{L_c}{k-L_c}))^{-1}\cdot\rho. \tag{4.23}
$$

4.5 Hindering in the Iterative Method

The principle of hindering must be included in the iterative method. When changing the time an operator is working on the tote, the limiting probabilities change and therefore also the mean performance measures. On first thought we would expect that the utilization will grow when increasing the time an operator is working an a tote. This is true but because the limiting probabilities change the mean number of totes on the conveyor will change and therefore also the hindering probability on the conveyor and the mean delay. This will again cause a difference in the time the operator works which changes the limiting probabilities etc.

In each step of the iterative method we should therefore do some extra calculations; we must calculate the hindering probability on the conveyor, the mean delay and the new mean service time. These three calculation steps should be added in each iteration

1. Hindering probability on conveyor: $P(\text{hindered at conveyor}) = \min[\frac{L_c}{k}, 1]$
2. Mean delay: $T = \frac{1}{k\gamma}\frac{L_c}{k-L_c}$, where $T = \infty$ if $P(\text{hindered at conveyor}) = 1$
3. New mean service time: $1/\mu' = 1/\mu + T$

The iterative method will work in the same manner as before but with these three extra steps. In the end we have to adjust the utilization to the effective utilization using (4.23). The complexity of the algorithm will not change, in each step there are calculations added with the same or lower complexity.

4.5.1 Java Application: ZPS Analyzing Tool

The method just described needs to be implemented in our Java application, which is a simple extension to the program already made when no hindering was considered. An extra input parameter is needed which the user can give in. The interface will get an extra input field which we already have seen in Figure 3.9, where this parameter can be initialized

10. The number of places on the conveyor

Here the number of places on the conveyor is the total number of totes which can stand on the conveyor, when the gapsize is 0. This value is obviously given by the total length of the conveyor divided by the length of one tote. The output field and output results are the same as before.

The program behind the interface does change, but not much, because in every iteration step we have to add three calculation steps as just given and in the end we will also adjust the utilization to its effective utilization. When every performance measure is calculated the results are printed to the output field and also saved to an Excel-file.

4.6 Simulation

Until now we simulated the system in such a way that totes could get as close to each other as we wanted, they could even be placed on each other. However, in real life a gap is used which is normally the length of one tote. This can also be simulated in Automod, where we used the old simulation and only adjusted this gap length. The simulation can also take other gap lengths into account or other capacities of the conveyor.
In our simulation a conveyor has a length of 50 meter and a tote has a standard length of 1 meter. We can type in the number of places K the conveyor can maximally handle by increasing or decreasing the length of a tote. If for example the capacity of the conveyor is set to $K = 20$ than the length of one tote is set to $2,5$ meter, because $50/2,5 = 20$.

4.6.1 Validation

The simulation should be validated, but there are no standard queueing systems known which use these gaps. If we however set the buffersize and the threshold very high, than we would expect to see almost an $M/M/1$ queue. In Table 4.1 we can see the results for utilization and throughput for the 1 zone model when N and C are infinite. Here we used $1/\mu = 30$, $1/\lambda = 34$ and $1/\gamma = 60$ seconds. As we would expect the results for the simulation are very similar to the theory. This does not imply that our simulation works well for the use of gaps, but for this extreme case the simulation still works as well as before.

	Simulation	Theory	$\Delta\%$
ρ	0,877	0,882	0,6
δ	105,7	105,9	0,2

Table 4.1: Validation of simulation results for utilization (ρ) and throughput per hour (δ)

We will inspect the influence of the gapsize to the utilization, as part of our validation. We expect that the utilization will drop when the gapsize increases, because the operator has to wait more on average when the gap is bigger. To do this we have chosen the following scenario

1. The number of zones is 8.

2. The number of zones to visit is 2.
3. The buffers of all zones have size 10.
4. The maximum number of totes allowed in the segment is 100.
5. The mean pick time of each operator is 30 seconds.
6. The mean inter-arrival time is 2 seconds.
7. The mean time on each conveyor piece is 60 seconds.

In Table 4.2 we can see the results when changing the gapsize from 0 to 5 for the mean utilization of the 8 zones. Here gapsize 0 is the situation we investigated in the previous chapters, so totes can be placed on each other. In the rightmost column we can see the effective utilization results as percentages of the utilization when we have a gapsize of 0, denoted as relative utilization. From Table 4.2 we can see that indeed the utilization will drop when the gapsize increases, so it seems that the simulation works correctly. The utilization suddenly drops very fast when increasing the gapsize. Apparently when the gap has size 3 a deadlock situation will occur and the operator can not put totes back on the conveyor, causing an utilization of 0. In the following sections, where the results are presented, we will also check whether the utilization is smaller than calculated before as part of our validation. This will also be a check for our simulation.

Gapsize	ρ_{sim}	Relative utilization
0	0,861	100,0
1	0,831	96,5
2	0,744	86,4
3	0	0
4	0	0
5	0	0

Table 4.2: Simulation results for utilization when gapsize is varied

4.7 Results Single Server: 8 Zones

In this section we will present the results of our method compared with results from the simulation, for a 8 zone case. The errors presented ($\Delta\%$) are again the absolute relative errors. This case will be the same as in the previous chapter so we can see that indeed the utilization will be lower than before, we present only the effective utilization because this is the utilization we are interested in. This section will have the same structure as the result section of the previous chapter.

The performance measures we will give are again the utilization, throughput per hour, mean buffersize and blocking probability. For all the results presented in this section the simulation errors are maximally 0.007, 1.1, 0.1 and 0.2 for utilization, throughput, mean buffersize and blocking probability respectively. The mean number of totes in the source in all the scenarios is small, so the source will not be working all the time and we know that we are investigating interesting examples. The gap length will be equal to the length of the tote, which is a value close to reality.

The conveyor has a length of 50 meter and totes are 1 meter long so the total number of places on a conveyor is $K = 50$.

4.7.1 Variable number of zones to visit

In this part we will vary the (exact) number of zones a tote needs to visit. These parameters will take in the same values as in the previous chapter. To be complete, the other variables are chosen to be:

1. The buffers of all zones have size 10.
2. The maximum number of totes allowed in the segment is 80.
3. The mean pick time of each operator is 30 seconds.
4. The mean inter-arrival time is 2 seconds.
5. The mean time on each conveyor piece is 60 seconds.

The (exact) number of zones a tote needs to visit, denoted as v is the parameter we will vary. We will present the results for a number of visits of $1, 2, 4$ and 6 respectively. For the case that 2 zones are visited we will present the results for each zone. For the other presented tables, results will be the average of the 8 zones, so we will not compare our results with each zone but with the average of these zones. We will give the maximum relative error we have seen when our method is compared with each zone separately, these are given in between brackets behind the relative errors.

The results for utilization and throughput are stated in Table 4.3, where we can see that the simulation results are lower than before (Table 3.8) as we would expect. Also the results generated by our method give lower results than before. The relative errors are small and also the maximum relative errors are acceptable. Notice that all the results are again bigger than that of the simulation, except when 1 zone is visited. There the method corrects the utilization too much and our method gives a lower value than that of the simulation. The utilization is rather low here and blocking does not often occur, because simulation results of the previous section and this section are almost the same.

v	ρ_{it}	ρ_{sim}	$\Delta\%$	δ_{it}	δ_{sim}	$\Delta\%$	δ_{it}^{total}	δ_{sim}^{total}	$\Delta\%$
1	0,501	0,522	4,0 (4,3)	60,1	63,7	5,6 (5,9)	481,0	507,8	5,3
2	0,773	0,767	1,4 (2,0)	93,3	95,1	1,9 (2,2)	373,1	380,3	1,9
4	0,890	0,865	2,9 (3,3)	106,8	107,4	0,5 (0,8)	213,7	214,8	0,5
6	0,923	0,897	2,9 (3,3)	110,7	111,3	0,5 (0,6)	147,6	148,4	2,2

Table 4.3: Simulation and iterative method results for utilization and throughput and the total throughput

The remaining performance measures including the calculation time (τ) of our tool (in seconds) are given in Table 4.4. We see rather good results for the blocking probabilities and the mean buffersizes. Notice that when totes visit 1 zone the results are bad for the blocking probability, we saw this already when regarding utilization and throughput. Here this probability is also very low and this is overestimated by the model (a relative error of more than 100%). This scenario is therefore not very interesting. The other situations are interesting and we can see that buffersizes are estimated fairly accurate. The results for blocking probabilities are slightly better as before; we see rather small relative errors but big maximum relative errors. The mean buffersizes and blocking probabilities are bigger than we have seen before (Table 3.9), as we would expect because an operator will work longer on one tote. The calculation time of the tool is small and results are generated within 2 seconds.

v	L_z^{it}	L_z^{sim}	$\Delta\%$	b_{it}	b_{sim}	$\Delta\%$	τ
1	1,17	1,20	2,1 (3,7)	0,009	0,001	>100 (>100)	0,3
2	3,39	3,17	6,9 (9,7)	0,029	0,031	7,0 (14,3)	0,6
4	5,18	4,81	7,9 (11,3)	0,089	0,107	16,6 (20,0)	1,4
6	5,91	5,47	7,9 (11,7)	0,126	0,149	15,5 (23,1)	1,5

Table 4.4: Simulation and iterative method results for mean buffersize and blocking probability

The results per zone are stated in Table 4.5, where the results for utilization and throughput are very good and a maximum relative error of only 2% can be found. The results for the mean buffersize are also acceptable and for the blocking probability results are roughly 10%. Again we can see the fluctuations in the results for the different zones. In the next part we vary more parameters for the case that totes need to visit 2 zones to obtain more insights in the accuracy of the model.

$i \rightarrow$	1	2	3	4	5	6	7	8
ρ_i^{it}	0,773							
ρ_i^{sim}	0,769	0,769	0,765	0,765	0,762	0,768	0,771	0,764
$\Delta_i\%$	1,1	1,1	1,6	1,6	2,0	1,2	0,9	1,7
δ_i^{it}	93,3							
δ_i^{sim}	95,0	95,3	95,0	95,1	94,9	95,2	95,3	94,9
$\Delta_i\%$	1,8	2,1	1,8	1,9	1,7	2,0	2,2	1,7
L_{zi}^{it}	3,39							
L_{zi}^{sim}	3,18	3,18	3,16	3,09	3,18	3,14	3,21	3,20
$\Delta_i\%$	6,4	6,5	7,0	9,7	6,4	7,9	5,4	5,8
$b_{it}^{(i)}$	0,029							
$b_{sim}^{(i)}$	0,030	0,030	0,028	0,033	0,029	0,031	0,032	0,033
$\Delta_i\%$	3,7	4,6	2,5	12,4	1,2	8,4	11,5	14,3

Table 4.5: Simulation and iterative method results per zone when number of visits is 2

4.7.2 Two zones to visit

Here we will discuss results when exactly 2 zones are visited. The same 3 parameters will be varied as in the previous chapter, recall that this will be

1. The buffersize (N): 5, 10.
2. The threshold (C): 80, 100.
3. The mean time on each conveyor ($1/\gamma$): 60, 90.

The 8 scenarios, generated by these values, will be presented and discussed here.

The model is very accurate for the most important performance measures; utilization, throughput per zone and total throughput. In Table 4.6 we can see that all relative errors are beneath 4% for every scenario. The maximum relative errors are also beneath this value. Compared with the results of the previous section (with no hindering in Table 3.11) we can see that the simulation results are indeed lower, as we would expect. The results for mean buffersize and blocking probability are given in Table 4.7. The mean buffersizes, and therefore also the blocking probabilities, are bigger compared to the results when hindering at the conveyor is not considered (Table 3.12). This is in accordance with what we would expect. The results for the mean buffersize are in general good; relative errors are roughly 6%. The maximum relative errors are roughly 10%. The

N	C	$1/\gamma$	ρ_{it}	ρ_{sim}	$\Delta\%$	δ_{it}	δ_{sim}	$\Delta\%$	δ^{total}_{it}	δ^{total}_{sim}	$\Delta\%$
5	100	60	0,796	0,784	1,6 (2,2)	95,5	94,0	1,6 (2,4)	375,3	375,9	0,2
5	100	90	0,669	0,686	2,4 (3,0)	80,3	82,3	2,4 (2,9)	321,1	329,2	2,4
5	80	60	0,749	0,743	0,8 (1,2)	89,9	89,1	0,8 (1,0)	359,6	356,5	0,8
5	80	90	0,613	0,623	1,7 (2,6)	73,5	74,8	1,7 (2,5)	294,1	299,1	1,7
10	100	60	0,839	0,826	1,6 (2,2)	100,7	99,3	1,6 (2,0)	402,9	397,2	1,6
10	100	90	0,714	0,733	2,6 (3,1)	85,7	88,1	2,6 (3,6)	342,7	352,2	2,6
10	80	60	0,777	0,777	0,0 (0,5)	93,3	93,2	0,0 (0,4)	373,1	372,7	0,1
10	80	90	0,639	0,653	2,3 (2,8)	76,6	78,3	2,2 (2,6)	306,5	313,2	2,3

Table 4.6: Simulation and iterative method results for utilization and throughput per zone and the total throughput when number of visits is 2

N	C	$1/\gamma$	L_z^{it}	L_z^{sim}	$\Delta\%$	b_{it}	b_{sim}	$\Delta\%$	τ
5	100	60	2,88	2,66	8,3 (9,8)	0,217	0,229	5,0 (12,0)	0,8
5	100	90	2,25	2,08	7,9 (10,3)	0,127	0,118	6,8 (13,0)	0,8
5	80	60	2,29	2,19	4,4 (7,2)	0,130	0,138	5,7 (10,5)	0,8
5	80	90	1,66	1,57	5,4 (7,1)	0,063	0,059	7,9 (14,0)	0,8
10	100	60	4,74	4,45	6,4 (10,5)	0,072	0,096	24,7 (33,5)	0,6
10	100	90	3,39	3,16	7,1 (9,5)	0,030	0,028	6,0 (11,1)	0,9
10	80	60	3,39	3,30	2,6 (6,2)	0,029	0,035	19,1 (35,2)	0,6
10	80	90	2,19	2,06	6,1 (8,6)	0,008	0,006	46,3 (80,8)	0,6

Table 4.7: Simulation and iterative method results for mean buffersize and blocking probability when number of visits is 2

approximation of the blocking probabilities are better than before. When the buffer is of size 5 the approximations are good and if the buffer 10 we see the same (maximum) errors as before. For small buffersizes we see that also the maximum relative errors are quite good. The biggest error we have seen is of 80%, this is because a very small probability is approximated so large errors are made very easily. The calculation time of our model is again very good, within a second the results are generated.
In the next section we will present and discuss the results for the infinite server source.

4.8 Results Infinite Server: 8 Zones

Here we consider the source as an infinite server, where the errors presented ($\Delta\%$) are the absolute relative errors. For this comparison between the method and the simulation we keep the same values of the parameters as for the single server case except for $1/\lambda = 100$ seconds. The results will be presented in the same order as in the previous section, first we will give all the results when the number of zones visited is varying and finally we will present the results when 2 zones are visited. The simulation error for all the results will not be bigger than $0,008$ for the utilization, 1 for the throughput, $0,2$ for the buffersize and finally $0,2$ for the probability of blocking. The gap length will again be equal to the length of the tote. The conveyor has a length of 50 meter and totes are 1 meter long so the number of places on a conveyor is $K = 50$.

4.8.1 Variable number of zones to visit

The first table (Table 4.8) presents the results for the utilization and throughput of a zone and the total throughput of a segment. When more than 1 zone is visited the model is very accurate; all relative errors are less than 3%. When 1 zone is visited we again see that the model corrects the utilization too much and results are a little bit worse. This will also not be an interesting situation for Vanderlande, because utilization and throughput are very low. The same conclusion can be drawn when looking at the results of mean buffersize and blocking probability (Table 4.9); when the number of zones that is visited is bigger than 1 the results are good. The relative errors for the blocking probability are $\pm 20\%$ and for maximum relative errors we again see errors of approximately the same size. When 1 zone is visited we see almost no blocking and therefore the relative error is here very big; much more than 100%. Within 2 seconds the model can generate these results, as can be seen in the last column. For the situation that 2 zones are visited by each tote, we have presented the results for each zone in Table 4.10. The results of the simulation fluctuate and therefore we see some difference in the relative errors per zone. Note that relative errors of utilization and throughput are almost the same, which also should be the case.

v	ρ_{it}	ρ_{sim}	$\Delta\%$	δ_{it}	δ_{sim}	$\Delta\%$	δ_{it}^{total}	δ_{sim}^{total}	$\Delta\%$
1	0,440	0,469	6,2 (6,5)	52,8	56,3	6,2 (6,3)	422,6	450,6	6,2
2	0,732	0,736	0,5 (1,0)	87,8	88,1	0,5 (0,7)	351,3	352,5	0,5
4	0,876	0,860	1,8 (2,2)	105,9	103,1	1,8 (1,8)	210,1	206,3	1,8
6	0,915	0,893	2,4 (2,8)	109,8	107,3	2,3 (2,6)	146,4	143,1	2,3

Table 4.8: Simulation and iterative method results for utilization and throughput and the total throughput

v	L_z^{it}	L_z^{sim}	$\Delta\%$	b_{it}	b_{sim}	$\Delta\%$	τ
1	0,88	0,95	8,0 (9,0)	0,007	0,000	>100 (>100)	0,3
2	2,74	2,70	1,4 (2,8)	0,016	0,018	14,9 (22,4)	0,6
4	4,78	4,57	4,7 (7,0)	0,072	0,092	21,3 (26,0)	1,1
6	5,62	5,33	5,6 (7,8)	0,111	0,138	20,1 (23,5)	1,6

Table 4.9: Simulation and iterative method results for mean buffersize and blocking probability

We can clearly see the differences in accuracy of the model when regarding the blocking probability at each zone. In the next part we vary more parameters for the case that totes need to visit 2 zones to obtain more insights in the accuracy of the model when we consider the infinite source case.

$i \rightarrow$	1	2	3	4	5	6	7	8
ρ_i^{it}	0,732							
ρ_i^{sim}	0,735	0,735	0,736	0,733	0,734	0,737	0,740	0,737
$\Delta_i\%$	0,4	0,4	0,5	0,1	0,3	0,7	1,0	0,7

δ_i^{it}	87,8							
δ_i^{sim}	88,1	88,1	88,0	87,9	88,1	88,2	88,5	88,2
$\Delta_i\%$	0,3	0,3	0,2	0,1	0,4	0,5	0,7	0,4

L_{zi}^{it}	2,74							
L_{zi}^{sim}	2,69	2,71	2,67	2,71	2,70	2,67	2,71	2,75
$\Delta_i\%$	1,7	1,1	2,8	1,0	1,7	2,5	1,1	0,4

$b_{it}^{(i)}$	0,016							
$b_{sim}^{(i)}$	0,019	0,016	0,017	0,016	0,020	0,020	0,018	0,019
$\Delta_i\%$	18,5	5,0	10,7	0,8	21,2	22,4	15,9	19,8

Table 4.10: Simulation and iterative method results per zone when number of visits is 2

4.8.2 Two zones to visit

In this section the results of the 8 different scenarios, as described before, will be stated and discussed. The results can be found in Table 4.11 and Table 4.12.

The utilization and throughput results of the model approximate the results of the simulation very good; (maximum) relative errors are all below 3%. Again we can see that these performance measures are lower than presented in the previous chapter (Table 3.18), where hindering at the conveyor was not considered. The model also gives good approximations for the mean buffersize, which can be seen in Table 4.12. When the buffersize is 5 the blocking probability is approximated better than before; all relative errors are below 9% and maximum relative errors are also not very big. However if the buffer is 10 we see the same results as before, namely rather

N	C	$1/\gamma$	ρ_{it}	ρ_{sim}	$\Delta\%$	δ_{it}	δ_{sim}	$\Delta\%$	δ_{it}^{total}	δ_{sim}^{total}	$\Delta\%$
5	100	60	0,778	0,766	1,5 (2,1)	93,3	91,8	1,5 (2,2)	373,2	367,3	1,5
5	100	90	0,650	0,664	2,1 (2,7)	78,0	79,7	2,1 (2,4)	312,2	318,6	2,9
5	80	60	0,711	0,708	0,5 (1,3)	85,3	84,9	0,5 (0,9)	341,2	339,7	0,5
5	80	90	0,579	0,587	1,4 (2,1)	69,5	70,5	1,4 (2,8)	277,9	281,9	1,4
10	100	60	0,811	0,805	0,7 (1,1)	97,3	96,6	0,7 (0,9)	389,2	386,5	0,7
10	100	90	0,686	0,702	2,4 (2,7)	82,3	84,3	2,4 (2,5)	329,1	337,1	2,4
10	80	60	0,732	0,736	0,5 (0,7)	87,8	88,1	0,5 (0,6)	351,3	352,5	0,5
10	80	90	0,598	0,609	1,9 (2,4)	71,7	72,9	1,9 (2,5)	286,8	291,7	1,9

Table 4.11: Simulation and iterative method results for utilization and throughput per zone and the total throughput when number of visits is 2

big (maximum) relative errors. The time until results are generated using our tool is again very small, within a second these results are generated.

N	C	$1/\gamma$	L_z^{it}	L_z^{sim}	$\Delta\%$	b_{it}	b_{sim}	$\Delta\%$	τ
5	100	60	2,59	2,44	5,9 (7,9)	0,172	0,188	8,5 (15,0)	0,8
5	100	90	2,00	1,88	6,2 (8,4)	0,098	0,093	4,8 (16,4)	0,8
5	80	60	1,99	1,93	2,8 (6,4)	0,095	0,103	7,6 (14,7)	0,6
5	80	90	1,44	1,38	4,3 (6,1)	0,046	0,044	4,6 (10,3)	0,9
10	100	60	4,03	3,90	3,2 (5,2)	0,047	0,062	24,1 (26,3)	0,6
10	100	90	2,84	2,68	6,2 (12,4)	0,017	0,016	8,2 (10,7)	0,7
10	80	60	2,74	2,70	1,4 (2,8)	0,016	0,018	14,9 (19,8)	0,7
10	80	90	1,78	1,70	4,7 (8,8)	0,004	0,003	40,6 (61,2)	0,7

Table 4.12: Simulation and iterative method results for mean buffersize and blocking probability when number of visits is 2

4.9 Simulation Project: Flex Logistics

In the previous chapter we used known data from a project within Vanderlande, this project was Flex Logistics. In this section we will use the settings of this project and use our simulation and model to compare results. The model we are using is the single server source and the recall that variables will take on the following values:

1. The total number of zones is 8.
2. The number of zones (exactly) visited is 2.
3. The buffersize of each zone is 10.
4. The mean pick time is $14, 5$ seconds.
5. The mean inter-arrival time is 2 seconds.

6. The total circulation time on the conveyor is 4 minutes, so the time per conveyor is $240/8 = 30$ seconds.

Note that we did not mention the value of the threshold. This is because the goal of this section will be to determine the optimum value of this threshold. With optimum value we mean the value that generates an utilization, and therefore also throughput, as large as possible. That this optimum exists is obvious, there must be an optimal value where the operator has maximum profit of the total number of totes in the system. If the threshold is bigger than this optimal value the operator has difficulty with the many number of totes on the conveyor and if the threshold is smaller the operator could have done more if there was supply of new totes.
We will also see a deadlock situation for certain values and our second goal will be to determine how our method can warn for this unwanted situation.

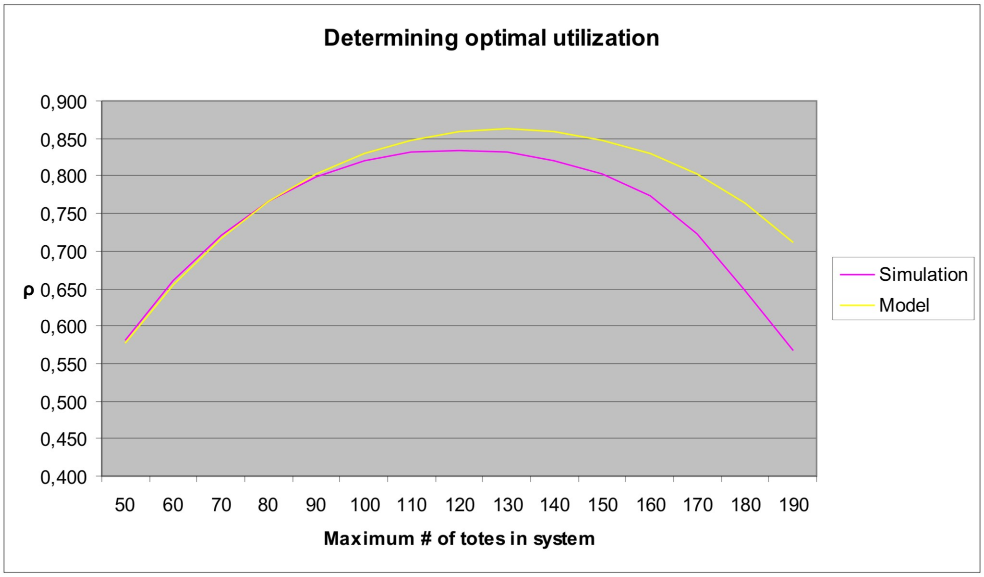


Figure 4.3: Utilization results for simulation and model, when varying threshold.

4.9.1 Determine optimal utilization

In this section we will determine the optimal value for the system of Flex Logistics. Results are generated by varying the threshold from 50 to 200 totes and we plotted the mean utilization of the zones, Figure 4.3. The results for a threshold of 200 totes is not presented here, because a deadlock situation will occur for these settings in the simulation. These answers are generated with our model within 2 seconds.
We can see that for small values of the threshold our model follows the simulation results very well. The simulation reaches its optimal value when the threshold is 120 totes and after that the utilization starts to drop again. Our method reaches its optimum at a threshold of 130 and starts to drop from thereon. The model does not determine the optimal value exactly, but comes very close to the actual value determined by the simulation. For values close to the optimal value of the threshold we also see little difference in utilization. We can also see that our method drops much slower than the simulation, resulting in growing relative errors as can be seen in Figure

4.3 and also Table 4.13. In Table 4.13 we can see a deadlock situation (DL) for a threshold value of 200, this is denoted by the simulation *and* the model. The model does determine a deadlock when the simulation does, but we do not know if this happens exactly for the same value. In section 4.9.3 we will take a closer look to this deadlock situation.

C	ρ_{it}	ρ_{sim}	$\Delta\%$
50	0,577	0,580	0,5
60	0,655	0,660	0,9
70	0,717	0,720	0,4
80	0,765	0,766	0,1
90	0,802	0,799	0,5
100	0,829	0,820	1,2
110	0,848	0,831	2,0
120	0,858	**0,834**	2,9
130	**0,862**	0,832	3,6
140	0,859	0,821	4,6
150	0,848	0,802	5,8
160	0,830	0,773	7,3
170	0,803	0,723	11,0
180	0,764	0,647	18,1
190	0,711	0,567	25,4
200	DL	DL	-

Table 4.13: Simulation and iterative method results for determining optimal utilization

4.9.2 Increasing gapsize / decreasing number of places

In the simulation it is time consuming to change the number of places on a conveyor by increasing or decreasing the length of the conveyor. We can however easily change the gapsize. By changing the gapsize the number of places is also changed and this will have the same effect. From (4.18) we can see that changing the gap size from 1 to 3 leads to halving the factor c and therefore also the number of available places. We will only increase the gapsize (decrease number of places) and choose the other parameters as before. We would expect an utilization which is lower than before and also we expect to see a deadlock situation for smaller thresholds. In our simulation and model we changed our gapsize to 2. This means that between successive totes there must be room for at least 2 totes. In Figure 4.4 the results of the simulation and model are presented schematically, while in Table 4.14 the results are given exactly with their relative errors. Our model generates each single results within 4 seconds. We can see in Figure 4.4 that our model follows the simulation very well for small values of the threshold. The maximum of the simulation does not coincide with the maximum of the model, as we can see in the figure and also in Table 4.14 where the maxima are denoted in bold. The value of C where this maximum is reached is the same. In this table we can also see that relative errors are small, except for thresholds bigger than 120 totes.

The simulation drops again very fast after it have reached its maximum, while the model slowly drops and nicely follows a parabola. The results are only presented up to a threshold of 150, after this value the model also returns a deadlock exception. We now see that our model does

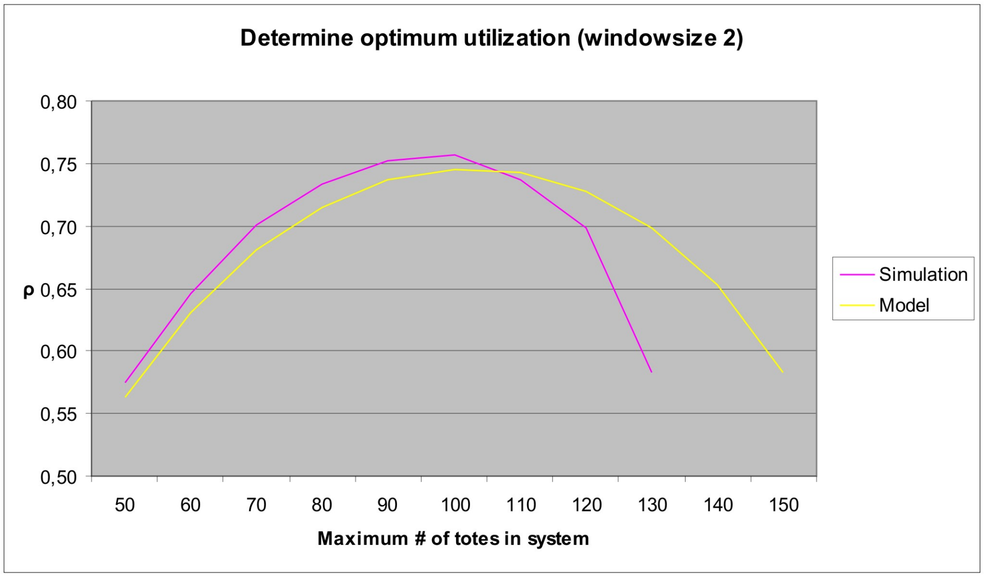


Figure 4.4: Utilization results, simulation and model, when varying threshold and gapsize is 2.

C	ρ_{it}	ρ_{sim}	$\Delta\%$
50	0,563	0,574	1,9
60	0,631	0,646	2,5
70	0,681	0,701	2,8
80	0,715	0,734	2,6
90	0,737	0,752	2,1
100	**0,745**	**0,757**	1,5
110	0,742	0,737	0,7
120	0,727	0,699	4,0
130	0,699	0,583	20,0
140	0,653	DL	-
150	0,583	DL	-

Table 4.14: Simulation and iterative method results for determining optimal utilization, with gapsize 2

not denote a deadlock situation while the simulation does. Again we can see that the utilizations around the maximum do not differ very much from the optimal utilization. One could also choose a value near the optimal threshold and the utilization will still be high. In the next part we will determine how our model can be slightly adjusted so it will determine a deadlock situation earlier.

4.9.3 Deadlock situation

A deadlock situation is something that we would like to avoid to happen and therefore it would be nice if our method denoted a deadlock situation when this is the case. Our model now determines a deadlock situation if the mean number of totes is greater or equal to the total number of places, i.e. the probability of hindering at the conveyor is 1. However a deadlock situation is likely to occur if the conveyor is occupied for approximately 95%. Recovery is for those situations still possible but not likely to happen. In our model we will therefore denote a deadlock situation if the conveyor occupancy is greater than 95%, this will cause the model to determine such a situation earlier as before.

In the previous part we have seen that the simulation will have a deadlock much earlier than the model. With the just mentioned adjustment this will not drastically change; if the threshold is 150 the model has an occupation of the conveyor of $81,2\%$, which is not even close to the 95% we just suggested.

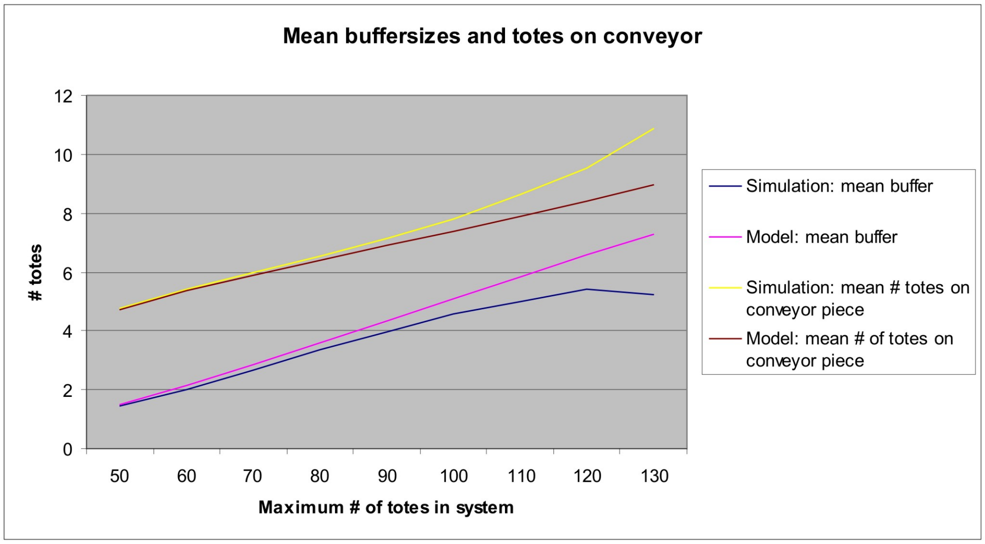


Figure 4.5: Mean buffersizes and mean number of totes on conveyor, for the simulation and model.

We can clearly see that our model does not work well for very busy systems or for determining deadlocks. This can be explained by the approximation of the number of totes on the conveyor. We have seen earlier in the results that the mean buffersize is overestimated by our model, causing an underestimation of the mean number of totes on the conveyor. This can be seen in Figure 4.5. For very busy systems this approximation error will increase and will cause that the utilization will be adjusted with a factor smaller than in reality. In Table 4.15 the results for mean number of totes on the conveyor are given with its relative errors. When the threshold is bigger than 110 relative errors are becoming large and the method will give bad results for the utilization, as we have seen in Table 4.14. Another explanation can be our assumption that the number of occupied places on the conveyor is geometrically distributed. This can be different especially for very busy systems. If a place on the conveyor is occupied it is more likely that the next place is also occupied, so here the assumption of a geometric distribution does not hold. A simulation study could be done to fit this distribution and eventually this can be implemented in the Java application.

Fortunately the values of C of interest for Vanderlande are those before the optimum and maybe just after this optimum, where our method works well.

C	L_c^{it}	L_c^{sim}	$\Delta\%$
50	4,73	4,76	0,4
60	5,36	5,41	0,8
70	5,90	5,99	1,5
80	6,41	6,55	2,1
90	6,90	7,16	3,7
100	7,39	7,81	5,4
110	7,90	8,63	8,5
120	8,42	9,52	11,6
130	8,97	10,89	17,6

Table 4.15: Simulation and iterative method results for mean number of totes on conveyor

4.10 Summary and Conclusion

In a ZPS an the utilization of an operator is determined by the supply of new totes and the number of totes an operator has to wait before he can put its tote back on the conveyor. The operator will also be busy when he can not place a tote on the conveyor, but this does not mean that he is busy doing something effective. In the previous chapters we assumed an operator can always place a tote back on the conveyor. However this does not agree with reality and if we would increase the maximum number of totes in the system the utilization will increase until it is 100%. In this chapter we have developed a method that will take the hindering at the conveyor into account.

Using the one zone model we sketched a method that will adjust the utilization and we have given some intuitive arguments that this method will work. The main part of the method is to determine the probability that a place at the conveyor is occupied, this turns out to be the mean number of totes on the conveyor divided by the total number of places on the conveyor. The number of places on the conveyor depends on the chosen gap length of a tote and the total length of the conveyor. By assuming a geometric distribution of occupied places we determine the mean time an operator is waiting to place the tote on the conveyor. When this extra time is determined we can determine the effective utilization.

The adjusting of the utilization should also be implemented in the iterative method of the previous chapters, because changing the total time an operator is working on a tote will have its impact on the limiting distribution. The mean number of totes on the conveyor and buffer will change. This will also change the adjusting factor, because this depends on the mean number of totes on the conveyor. Eventually we determine the effective utilization by multiplying with the adjusting factor. This is also implemented in our Java application, where now the user can also give in the number of places on the conveyor. This is defined as the length of the conveyor divided by the length of the tote.

We have simulated the same scenarios as in the previous chapter, but now we considered the effect of hindering when placing a tote on the conveyor. The simulation indeed gives utilization and throughput results that are smaller than before when comparing the same situation. The mean buffersize and blocking probability at the zone are bigger than before, as we would also expect.

When considering the single server source we see very good results for utilization and throughput, except when the number of zones to visit is very small. The method overcorrects which causes a higher relative error than before. The same results can be seen for mean buffersize and hindering probability; when only 1 zone is visited the results are bad while for other scenarios especially the results for the mean buffersize are accurate. Overall the relative errors for the blocking probability are roughly 15%. For the case that totes visit one zone the utilization and blocking probability at the zone are small, which means that this is not a very interesting case for Vanderlande Industries and it also may explain the overcorrection.

If we see the segment as part of a multi-segment ZPS, we are considering the source as an infinite server. Here we see same results as for the single server; when one zone is visited results are worse than for other scenarios. When regarding these results we did not see a deadlock situation. Also we did not focus on the best scenario possible for some settings. These problems are tackled when looking at an example of a ZPS of one of the customers of Vanderlande Industries.

This system is a ZPS of Flex Logistics and is the same example as chosen in the previous chapter. This specific ZPS has 8 zones and totes visit (exactly) 2 zones. First we focus on finding the optimal threshold value so that utilization is at its maximum. We have chosen two gapsizes to investigate; a gap of size 1 and of size 2. For both scenarios we see that our method works very well for small thresholds and also it approximates the optimal value very closely. After the simulation reaches its maximum it drops very fast, while the model also drops but not as fast.

The model therefore does not determine a deadlock situation while the simulation has already determined such a situation much earlier. This can be explained by the approximation of the model; for very busy systems the mean buffersize is overestimated causing an underestimation in the number of totes on the conveyor, or i.e. an underestimation of the extra time before a tote can be placed on the conveyor. Fortunately the model works well for values of interest for Vanderlande.

Chapter 5

Multi-Segment ZPS

In this chapter we will broaden our scope to a system that consists of multiple segments with multiple zones, the so called multi-segment ZPS. In these kind of systems segments are linked by a (high speed) central loop. In such a system a tote travels on the conveyor towards the first segment he has to visit, if he can enter the segment he will do that otherwise the tote will continue his way on the conveyor. The tote travels to his second destination, if it has one, and there the same principle will apply. In each segment the following holds; at the end of each segment the weight check will determine if a tote has visited all his zones and if not he will be recirculated and he will try to visit all the zones he missed in the first tour. Then the principle repeats itself until the tote has visited all the zones and he will leave the segment. There is a maximum number of totes allowed in the total system and in each segment as well. First we will not consider hindering at the conveyor here, because first the accuracy of the model without hindering must be checked. In Figure 5.1 this system is presented schematically with four segments, with each four (double) zones and a weight check at the end of each segment.

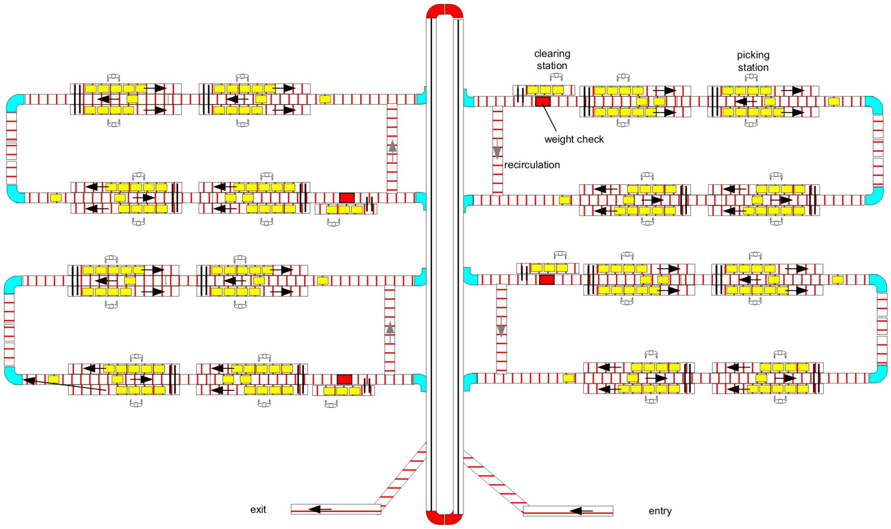


Figure 5.1: A multi-segment routing lay-out with four segments, each with four zones and a weight check

5.1 Introduction

This system is more complex than the models we have seen before, but there are also a lot of similarities. These similarities are based on the structure we also see in Figure 5.1; a segment with multiple zones is sort of a smaller version of the multi-segment ZPS. We could observe this structure as the structure that we also observe when considering fractals. This structure is going to help us to again obtain an iterative method. In this chapter we will present this method together with some results.

First we present Norton's Theorem, which will be used when describing a multi-segment ZPS. Therefore we apply this theorem to the two models used throughout this thesis to describe the multi-segment ZPS. These models will be combined to fully describe these systems and a new algorithm will be presented. This algorithm will be implemented and two scenarios will be compared with simulation results. Furthermore the tool will be tested and hindering will also be included. Finally we will summarize this chapter and state a conclusion.

5.1.1 Norton's Theorem

The theorem of Norton is originally from electrical circuit theory. However there is also a form for (closed) Jackson networks ([4]), which states that:

Theorem 5 (Norton's Theorem). *It is possible to replace a subset of the queueing network by a single queue such that the equilibrium distribution of the rest of the networks remains unchanged.*

Such a subset is often denoted as a cluster. Norton's theorem will help us to describe a multi-segment ZPS in two ways, both replacing another subset of the queueing network corresponding to the multi-segment ZPS.

We can regard each segment as a cluster of several queueing stations, as is used in our analysis of one segment ZPS. The first model will use that principle and replaces each segment by a single queue. Therefore we will get a model similar to our single server source model. Some adjustments are necessary to describe the system properly. This model with its adjustments will be discussed in the next section.

The second model will look at one segment and the rest of the ZPS will be observed as one giant cluster. This cluster will also be modelled as a single queue. If we look more closely to this system we see it has much in common with our infinite server source model. Together with some adjustments this model will be described in section 5.3.

Much research has been done concerning Norton's theorem and its application to queueing networks. In [4] it is shown that Norton's theorem can be generalized to include state-dependent routing and servicing in exponential networks. Especially the state dependent servicing will help us to model a segment as well as the rest of the ZPS as a single queue. This will be described in the following two sections respectively.

5.2 Single Server: Segments as Zones

Keeping the special structure in mind as just mentioned, we can use the single source model to describe the multi-segment ZPS. The zones in the model now describe the segments of the total system. Therefore we have to change some variables to describe this system properly. The first variable is the number of totes that may enter the system, this will for the total system be denoted as D. This total is defined as the number of totes in the segments and on the mainline of the system. As before we allow in a segment maximally C totes, so the buffersize of these stations in the model becomes this threshold of C. Furthermore we have to change the type of stations that describe the segments. These stations are not single server stations, as was the case when considering zones. The rate at which totes will leave a segment depends on the number of totes in the segment; if there are many totes in the segment this rate will be bigger than when there are few totes present. Therefore we could model these stations as infinite servers, but then we would not use information we do have.

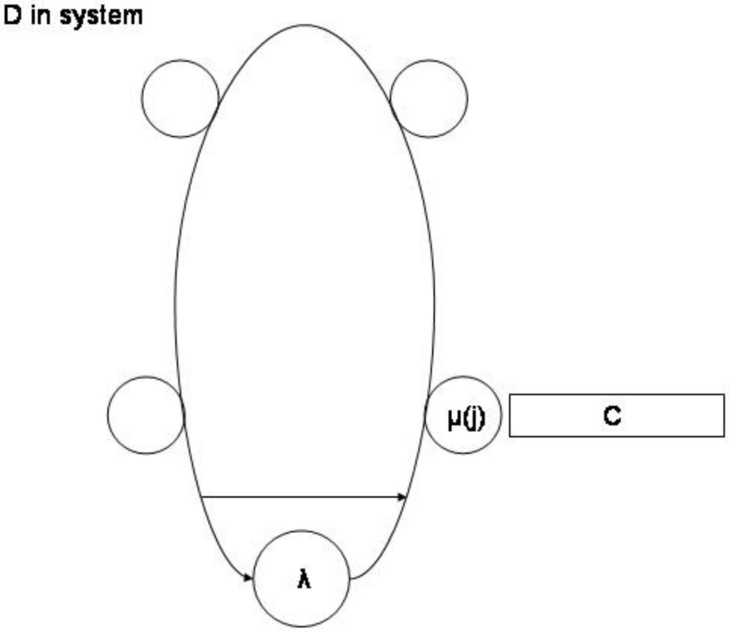


Figure 5.2: Single source model: segments as zones

The stations will be modelled as stations with queue dependent service rates. The rate at which totes leave the station when there are j totes in the segment, will be denoted as $\mu(j)$. The problem here is that we do not know these rates. If we would, we could calculate the performance measures of each segment in the same manner as we did for the zones in previous chapters. Also we can calculate the supply of new totes at each segment. The model described in this section is schematically presented in Figure 5.2.

5.3 Infinite Server: Segment as Part of ZPS

The second model we will use, is the infinite source model. We already said earlier that this model describes the segment as part of a multi-segment ZPS. The model will be the same as before, except for some small changes for the source.

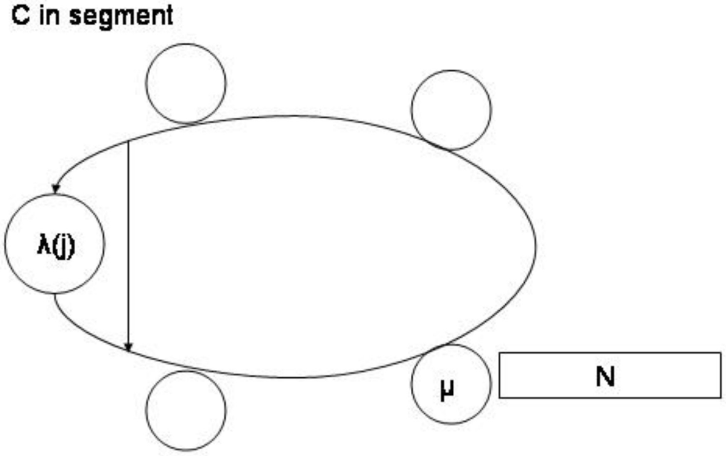


Figure 5.3: Infinite source model: segment as part of ZPS

This source was modelled as an infinite server, but this does not describe the real situation completely. When there are many totes on the main conveyor, totes will enter the segment more quickly than described by an infinite server. Therefore we will model this station as a station with queue dependent service rates. The rate at which totes enter the segment when j totes are in the segment, will be denoted as $\lambda(j)$. This principle is presented in Figure 5.3.

The only problem here is that we do not know these queue dependent rates. If we would know these, we could calculate the performance measures of the zones in the same manner as before. Furthermore we can determine the rate at which totes leave the segment when there are j customers, which is the parameter missing in the former model.

5.4 Combining the Models

We have described the two models we used throughout this thesis and their application to a multi-segment ZPS. In the single server model we do know the parameter of supply of new totes but we do not know the rate at which totes leave the segments. However in the infinite server model we do know the mean picktime of the zones and we do not know the rate at which totes enter the segment. This observation gives to believe that we could combine these two models and use each others output as input for the other model. In [4] it is also shown that global processes (between segments) and local processes (in segments) can be analyzed separately, which is an implication of Norton's theorem. Therefore segments will be regarded as well as zones as part of the bigger ZPS. This principle is denoted in Figure 5.4.

5.4.1 The Method

To calculate performance measures we will iteratively switch between the two models. First we will describe how to extract the parameter values from one model, which we need as input for the other model.

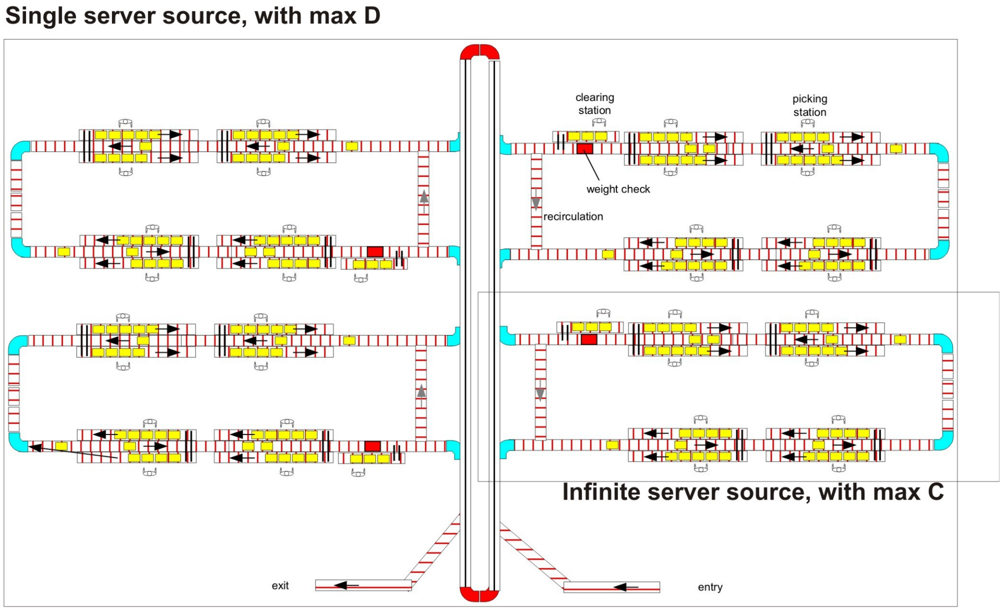

Single server source, with max D
clearing station
picking station
weight check
recirculation
Infinite server source, with max C
exit
entry

Figure 5.4: Combining the single and infinite model for multi-segment ZPS

Let us first start with the single server source, which describes the entire system and we see segments as stations with queue dependent service rates. Let us assume that we know the rates $\mu(j)$, recall that these are the rates of the stations when there are j totes in the queue. With this model we can determine the arrival rates at each of the queue dependent stations, given that there are j totes in its queue. This rate we already defined as $\lambda(j)$ and is the input we need for our 'segment' model. We can determine the queue length distribution of each station and together with the Arrival Theorem this gives us an expression for $\lambda(j)$:

$$\lambda(j) = \frac{\Lambda(D).P_j(D-1)}{P_j(D)}, \tag{5.1}$$

where $P_j(D)$ is the probability that there are j totes in the queue given a total number of D in the network. Recall that D was the maximum number of totes allowed in the system and $\Lambda(D)$ is the throughput of the observed station when there are D totes in the network. We have all

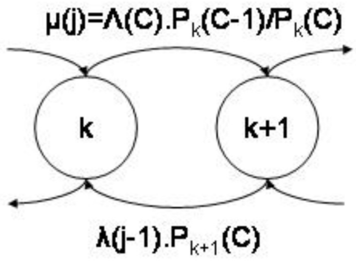


Figure 5.5: Transition rates between the states of the segment

variables we need for our next model. Using this model we must determine the rate at which totes leave the segment when there are j customers in this segment. We can determine this rate $\mu(j)$ by looking at the transition rates between the states of the source. These states are defined as the number of totes in the source. The transition rates between states $k+1$ and k are given in Figure 5.5, where $k := C - j$.

Here $P_k(C)$ represent the limiting probability that there are k totes in the source, given that there are C totes allowed in the segment. Furthermore $\Lambda(C)$ is the total throughput of the segment. Note that the rate from k to $k+1$ is defined as $\mu(j)$ and is given by the Arrival Theorem. This principle we also have seen earlier when regarding zones. Also we conditioned on the event that there are k totes in the source, or i.e. conditioning on the event that there are j totes in the segment. This way the total throughput of the segment will also balance out, as should be, by summing up:

$$\sum_{k=0}^{C-1} P_k(C) \frac{\Lambda(C) P_k(C-1)}{P_k(C)} = \sum_{k=0}^{C-1} \Lambda(C) P_k(C-1) = \Lambda(C).$$

We have determined the rates $\mu(j)$ and this is the new input for our other model, where we observe the total system. This process can be repeated until the rates do not alter anymore. With the model for one segment, we than can determine the performance measures of the zones we are interested in. This will be done in the same manner as before. The algorithm is schematically presented in Algorithm 9. The output of the algorithm can be the performance measures of the segments and also of the zones. Since we are mostly interested in the performance measures of a single operator, the tool we develop will only give performance measures of one segment.

Algorithm 9 Algorithm for Multi-Segment ZPS

```
initialize Set $\mu(j) = \infty$, ∀ j
repeat
    Use single server source model to obtain $\lambda(j)$'s
    Use infinite server source model to obtain $\mu(j)$'s
until Rates do not alter
return Performance measures of segments and especially of zones
```

The convergence of Algorithm 9 is not obvious, but we do know that each step of the algorithm will converge. This is already shown in the previous chapters. The convergence of Algorithm 9 will not be proven here, but we have seen for several scenarios that this will be the case.

5.4.2 Number of zones/segments to visit

In our model a tote visits exactly v zones per segment and exactly s segments. In reality this will however not be the case; totes can visit any number of segments as well zones per segment. Therefore the total number of zones a tote visits can differ much per tote. However in chapter 3 we have shown that in the segment it does not matter if we regard a mean number of v zone visits of a tote or that a tote visit exactly v zones. Because we have the same model when regarding segments, the same conclusion can be drawn for the number of segments to visit.

Therefore we only have to determine the mean number of zones per segment that are visited and the mean number of segments that are visited, using data from the orderlist. As said earlier, in our model we however visit exactly this number of zones per segment or segments. Therefore the total number of zones visited by each tote is given by the product of these two. If the mean number of visits is not an integer value, we can take a linear combination of two integers.

5.5 Simulation

The simulation model we used here, is a simulation developed within the Simulation Department of Vanderlande Industries. This simulation model is useful for ZPS of up to 6 segments, each up to 20 zones. We use this simulation because it has many similarities with the simulation made before and we can use most of it, especially the conveyor system which is already drawn. Furthermore some adjustments must be done, like for example the number of totes in a segment. In our simulation we always considered the number of totes in the system as the sum of the number of totes in the buffers and conveyors. The simulation made by the department does not take the buffers into account when determining the numbers in a segment, so this is adjusted.

5.5.1 Validation

We have to check whether this simulation works well. We do this by checking the results for the one segment case with multiple zones, here the time between segments is taken 0 to make comparison possible. The scenarios of the previous chapter are checked and the simulation gives similar results. We will only present the case of one segment where totes visit only 2 zone. The variables are chosen the same as before:

1. The buffers of all zones have size 10.
2. The maximum number of totes allowed in the segment is 80.
3. The mean picktime of each operator is 30 seconds.
4. The mean inter-arrival time of the single server source is 2 seconds.
5. The mean time on each conveyor piece is 60 seconds.

The results of the new simulation and old simulation are given in Table 5.1. Here the subscripts old stands for the old simulation and new for the simulation introduced in this section. Again the results presented here are the means of all the zones. We can see in this table that results are similar, except for the blocking probability. This can be explained by the fact that this time we simulated for a very short period of only 25 hours, because of time issues. The other results give to believe that the simulation works the same and we therefore will use this simulation when generating results.

ρ_{new}	ρ_{old}	$\Delta\%$	δ_{new}	δ_{old}	$\Delta\%$
0,799	0,799	0,1	96,0	95,6	0,3

L_z^{new}	L_z^{old}	$\Delta\%$	b_{new}	b_{old}	$\Delta\%$
3,17	3,19	0,6	0,025	0,032	21,8

Table 5.1: New and old simulation results for utilization and throughput and the total throughput

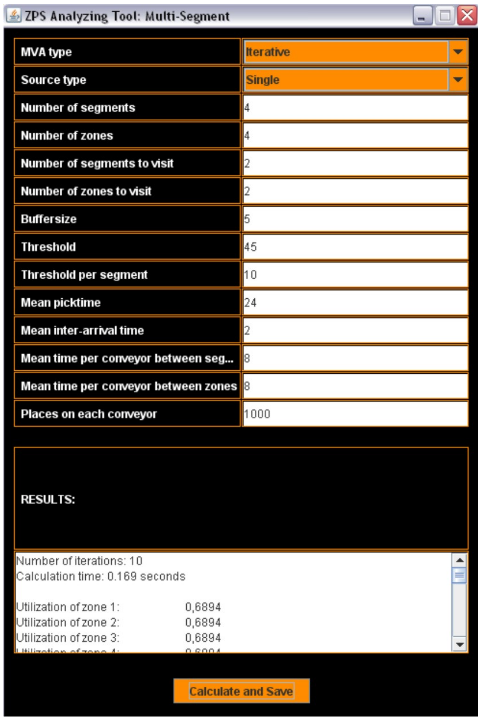
5.6 Java Application: ZPS Analyzing Tool, Multi-Segment

The method described in this chapter is implemented in Java 6.0.3. In this section we will shortly explain how this program, called 'ZPS Analyzing Tool, Multi-Segment', is implemented.

We can use the tool made until now to extend this to a multi-segment tool. All methods for the two different models, single and infinite, are already available and these can be used in this new program. The only difference is the main part of the tool, which is the algorithm as given in Algorithm 9. We have to switch between the single model and infinite model and the necessary variables need to take in the right value for the model. The algorithm stops if the mean cycle time in segment does not alter anymore, with a accuracy of $\epsilon = 10^{-3}$. This is the same as chosen for the MVA in each model.

The user interface will be the same as before only there will be no choice in the source- or the MVA type. There are also more variables that must be given in by the user and from others names are altered. The variables that are new to give in by the user are all on 'segment-level':

1. Number of segments.
2. Number of segments to visit.
3. Threshold per segment.
4. Mean time per conveyor between segments.



MVA type	Iterative
Source type	Single
Number of segments	4
Number of zones	4
Number of segments to visit	2
Number of zones to visit	2
Buffersize	5
Threshold	45
Threshold per segment	10
Mean picktime	24
Mean inter-arrival time	2
Mean time per conveyor between seg...	8
Mean time per conveyor between zones	8
Places on each conveyor	1000

```
Number of iterations: 10
Calculation time: 0.169 seconds

Utilization of zone 1:          0,6894
Utilization of zone 2:          0,6894
Utilization of zone 3:          0,6894
```

Figure 5.6: User interface of the ZPS Analyzing Tool for multi-segment ZPS

These variables, together with the old variables, can be initialized using the user interface. A representation of this interface can be found in Figure 5.6. Note that here the number of places on each conveyor is still an input variable. When chosen big enough we are considering the situation with no hindering. Later on we will include this variable and shortly explain the implementation of this hindering in the tool.

The output area stays the same and also the same input will be given, namely the performance measures of the zones in a single segment.

We have checked our application for the specific lay-out of a ZPS with one segment. When the traveltime to the first (and only) segment is 0 this application should give the same answers as before (chapter 3). The different scenarios for the one segment case have been run and similar results where obtained. At the third decimal we see differences, which probably occur due to successive approximation errors. This gives to believe that also our tool works well and we can now generate results.

5.7 Results: Beeline

In this section we will present the results of our tool compared with the simulation, described in the previous two sections. We will only consider two scenarios for this system, which is based on a project for a company called Beeline. For this company a ZPS with 6 segments, each with 20 zones was developed. However the multi-segment ZPS we will consider here is the same as in Figure 5.1; a system with 4 segments all with 4 zones. Our model can not handle 20 zones and therefore we have chosen for this example. The several variables are chosen to take on the values corresponding with the lay-out of the Beeline project.

1. Number of zones and segments to visit is both (exactly) 2, so in total 4 zones will be visited by each tote.
2. The buffer of each zone has size 5.
3. The mean pick time of the operator is 24 seconds.
4. The inter-arrival time of the single server source is 2 seconds.

To generate two scenarios we will define the settings for these two cases

Case 1: 1. The threshold per segment is 10 totes.
2. The threshold for the total system is 45 totes.
3. The mean time on a conveyor between segments and zones is 8 seconds.

Case 2: 1. The threshold per segment is 20 totes.
2. The threshold for the total system is 85 totes.
3. The mean time on a conveyor between segments and zones is 16 seconds.

The threshold of the total system is a consequence of the choice of the threshold for each segment separately; we allow 5 extra totes in the system if all segments are fully occupied. Furthermore the time on the conveyor between segments and between zones are also a consequence of the choice of the segment threshold. If we allow twice as many totes in a segment, the conveyors between the zones also should be longer. Therefore the mean time on these conveyors will also be longer and here we have chosen for twice that value.

We will state the results for both cases in the same manner as before. In Table 5.2 the results for utilization and throughput are given and in Table 5.3 the results for the buffersize and blocking probability. Here also the calculation time in seconds (τ) is given. Note that results are again performance measures of the zones; we have taken the average of each performance measure to compare with our iterative method. In brackets the maximum relative errors that occurred are given.

The simulation errors are bigger than before, because we did not simulate as long as desirable. For utilization the simulation error is not bigger than 0.013, for throughput not bigger than 2. Furthermore for buffersize this error will not exceed 0.1 and finally for the blocking probability this will be 0.01.

Case	ρ_{it}	ρ_{sim}	$\Delta\%$	δ_{it}	δ_{sim}	$\Delta\%$	δ_{it}^{total}	δ_{sim}^{total}	$\Delta\%$
1	0,670	0,669	0,1 (1,6)	100,4	100,0	0,4 (1,7)	200,9	200,8	0,1
2	0,830	0,801	3,5 (5,3)	124,5	120,0	3,6 (5,2)	249,0	240,8	3,3

Table 5.2: Simulation and iterative method results for utilization and throughput and the total throughput

Case	L_z^{it}	L_z^{sim}	$\Delta\%$	b_{it}	b_{sim}	$\Delta\%$	τ
1	1,46	1,56	5,7 (8,6)	0,026	0,033	20,1 (28,7)	0,2
2	2,38	2,41	1,1 (4,4)	0,120	0,095	25,2 (31,2)	0,1

Table 5.3: Simulation and iterative method results for mean buffersize and blocking probability

5.7.1 Discussion of Results

Here we will discuss the results for the two cases presented in Table 5.2 and Table 5.3.

When considering ’Case 1’ we see that results for utilization and throughput are very accurate; the maximum relative error is also very small with $1,7\%$. The results for mean buffersize are a little bit worse but still acceptable. The errors for the blocking probabilities have the same order as before and results are generated very fast. This speed is due to the fact that per segment the threshold is very small and the number of segments (zones respectively) is not too big.

Similar results can be seen for the so-called ’Case 2’. We clearly see the increase in all performance measures, compared with the first case, which we expect because of the increase of the thresholds. The results for utilization and throughput are good; relative errors are approximately 4%. Here the results for mean buffersize are very good and the relative errors for the blocking probabilities are approximately the same as for the previous case. The calculation time is again very small.

Overall we can see that results seem to be of the same accuracy as before and the speed of the tool is impressive. This is because there are little number of zones/segments and the thresholds are small.

5.7.2 Testing the Tool

We tested the tool for only two scenarios, which is too little to make a good comparison. We can test the tool in a different manner; when varying several variables we can see if results are as expected. We already tested it for the one segment situation, there the results where roughly the same. We will now also look at more general situations.

When variables are altered that only appear within a segment, the results are as expected. If we, for example, only increase the number of zones, than we see a decrease in the performance measures. Also when we decrease the time between two zones, utilization and throughput drop. There is however one variable that gives somewhat strange results when varying. This variable is the buffersize of each zone. When increasing the size of the buffer, sometimes the utilization slightly drops. The opposite also occurs, which is also counter intuitive. However because of the complex closed system we are considering, this can happen and also it can occur because of successive approximations.

The variables that are of more interest are variables that describe the system outside of the segments. When the total threshold increases, we also see an increase in utilization of the zones. More totes will travel in the main conveyor, therefore there will be more supply of new totes in a segment. The increase is not that big, because the segments are limited by their own threshold. We also see this when using our tool. Furthermore there is a decrease in performance measures when the traveltime between segments is increased. This is the same consequence when increasing the traveltime between zones. All performance measures decrease when the number of segments to visit becomes bigger. Totes have to travel a long time before they will visit a zone, therefore no new totes may enter the system and utilization of the zones will be smaller.

It seems that the output of our tool coincides with what we expect and we also note that the algorithm converges for every scenario tested.

5.8 Hindering at Conveyor

Until now we did not take hindering at the conveyor into account. We will only take hindering into account within a segment; when an operator places a tote back on the conveyor. Hindering when a tote leaves a segment and returns on the main line will not be included, because this is negligible. In reality there will also be several possibilities to slow down or speed up a tote, so this problem can be evaded.

Therefore hindering is already implemented in the tool, because it will work in a similar manner as before. This was already done when we considered only one segment. For this special situation we can check whether our tool gives similar answers as before. This is the case and again we see some differences at the third decimal. When applying hindering to the two scenarios of section 5.7 we see results we would expect; when the system is busy the number of places has more influence on the performance than for less busy systems. The output of the tool, when number of places is 10, is given in Table 5.4. More research is desirable when considering hindering at the conveyor, but we do see results one would expect.

Case	ρ_{it}	δ_{it}	δ_{it}^{total}	L_z^{it}	b_{it}
1	0,663	99,5	198,9	1,47	0,027
2	0,773	115,9	231,8	2,51	0,135

Table 5.4: Iterative method results when hindering is considered

5.9 Summary and Conclusion

In this chapter we considered a multi-segment ZPS. We note that the multi-segment ZPS has many similarities with the single segment with its zones. This structure of a ZPS is very useful and this will be used to model this complex system. The two main models used throughout this thesis are used to describe the system separately and eventually these are combined.

Using Norton's theorem we developed an iterative method that switches between the two models. Therefore some adjustments need to be done to describe these systems properly. The output of one model is the input of the other model. This is repeated until several variables do not alter anymore. This algorithm is also implemented in a Java tool. Furthermore a simulation is made to generate results we can compare with our tool.

Only two scenarios, that are based on a project for a company called Beeline, are investigated. Results are the same as before; results for utilization, throughput and mean buffersize are accurate. While the blocking probability is approximated with an error of roughly 20%. Furthermore the tool generates results one would expect. When, for example, the total threshold increases, we see a slight increase in the performance measures. Hindering is also included, but only within a segment. Hindering when leaving a segment is negligible and within a segment we already took hindering into account, so this is a simple adjustment in our tool.

When considering one segment, the tool gives similar results as before. The results differ on the third decimal, both when hindering is included or excluded from the model. It seems that our model and tool work properly, but little research is done in this chapter and therefore more research is certainly recommended and desirable.

Chapter 6

A Step Aside to CPS

Besides ZPS, a Compact Pick System (CPS) is also a picking concept developed within Vanderlande Industries. Only here the parts are going to the operator, these parts are going to the operator from a compact storage device (for example a HDS, see also Figure 6.1). A HDS is a automatic storage system where the parts are grabbed by cranes. The parts are stored in totes which are stored in the HDS. If a certain tote is requested the HDS finds the tote automatically and then grabs it and brings it to the 'exit' of the HDS. With a sorter and conveyors the tote travels to the operator which has a buffer, the operator grabs the part which he needs from this tote. The sorter can be regarded as a conveyor loop. The tote then travels back to the storage device via the sorter and gets back in its place. If the operator has all his parts the order is done and he pushes the order on a conveyor next to him. An example of such a system is given in Figure 6.1. The main difference with ZPS is that when using CPS, parts are going to the operator and not the other way around. However note that in both systems totes are travelling to an operator which need to work on that tote for a certain time. This last similarity is the reason why we will investigate if we can also model a CPS using the models we already introduced in previous chapters.

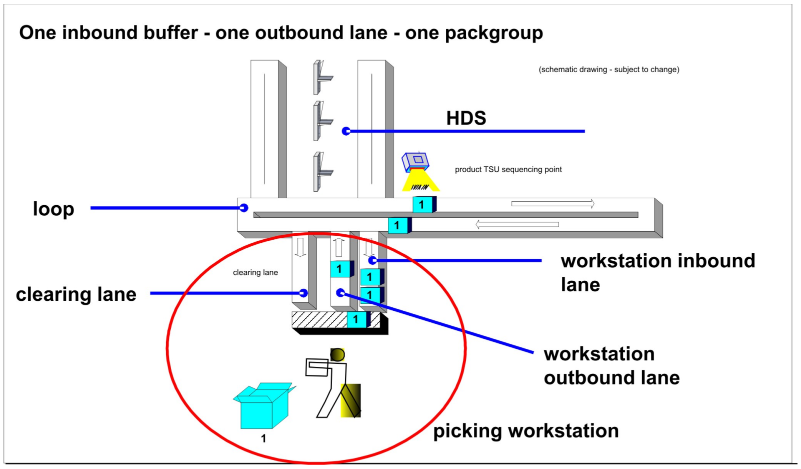

One inbound buffer - one outbound lane - one packgroup
(schematic drawing - subject to change)
HDS
product TSU sequencing point
loop
1
1
clearing lane
clearing lane
1
1
1
1
workstation inbound lane
workstation outbound lane
1
picking workstation

Figure 6.1: An example of a Compact Pick System

A big advantage of CPS is that totes are released if there is space available in the buffer in front of the operator. Totes are released if there are less totes present at the buffer than a certain threshold (C), which is smaller than the buffersize (N). This means that no blocking can occur, which made the modelling of ZPS complicated. The operator indicates to the storage device if he is done with a certain tote, then a new tote will be supplied by the storage device and it will travel to the operator. This is done for every pickstation, so each system has its own threshold and this threshold can also depend on the speed of the operator, but not necessarily. The workload in a CPS is, just like ZPS, balanced between the several stations.

6.1 Introduction

In this section we will discuss the model for CPS, we will do this for a general number of pickstations. First we will do this for one pickstation and we wil expand this for more pickstations.

6.1.1 One pickstation

We will first model this system for one pickstation and we will model this as a closed queueing network with three stations; the source, a delay (travel) part and the pickstation. In real life the source is the storage device and the travel part is of course the conveyor. The three stations will all have an infinite buffer, so also the pickstation. The pickstation however in real life has finite buffer, but because totes will only arrive if there is space available we can also model it as an infinite buffer. The pickstation is a single server, because an operator works with a certain speed on one tote at a certain moment. Whereas the other stations are infinite servers, including the source. This because the HDS has several outbound lanes for the example we will consider here and can work on more totes at once. All stations work under the FIFO policy and will follow an exponential distribution. The pickstation will have rate μ, the conveyor rate γ and the source will have rate λ. We assume that requested totes are always available and otherwise this (average) waiting time is included in the arrival parameter.

This system has almost the same representation as the model of the iterative method, only the blocking probability is always 0 by definition and the buffersize at the pickstation is infinite. This network can be found in Figure 6.2.

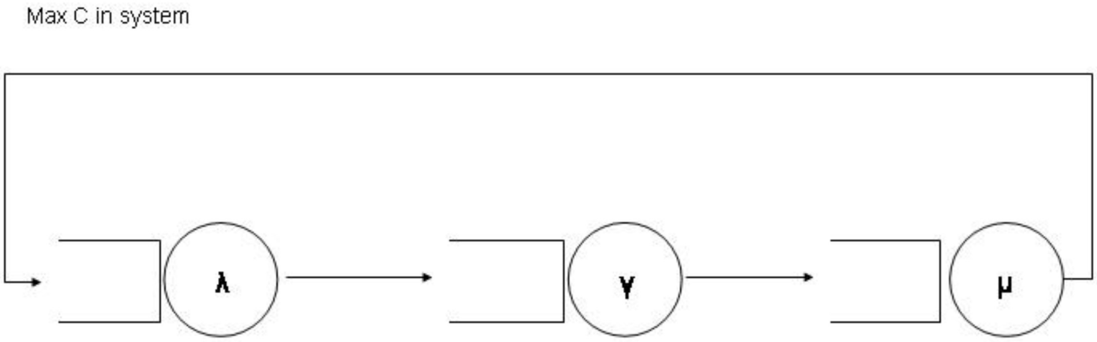


Figure 6.2: The CPS model for 1 pickstation

Hence no iteration needs to be done and we do know the limiting probabilities of this system, because we have a closed product-form network, see for example [3]:

$$P(n_1, n_2, n_3) := d\frac{1}{n_1!}(\frac{1}{\lambda})^{n_1}\frac{1}{n_2!}(\frac{1}{\gamma})^{n_2}(\frac{1}{\mu})^{n_3}, \tag{6.1}$$

where d is the normalizing constant and n_1, n_2, n_3 are the number of totes at the storage device, conveyor and station respectively. Here we also have $n_1 + n_2 + n_3 = C$, because of the threshold. These limiting probabilities can also easily be checked by substituting in the balance equations.

We now can easily determine several performance measures, like the utilization and the throughput per hour:

$$\begin{aligned}\rho &= 1 - \sum_{i=0}^{C} P(i, C-i, 0),\\ \delta &= 3600 \cdot \rho \cdot \mu.\end{aligned}$$

These performance measures can also be determined using mean value analysis. An other interesting parameter is the buffersize (N) of the workstation. We modelled it as an infinite buffer, but in real life this will not be the case. In the design fase the buffer should be chosen big enough so (almost) no blocking will occur but not too big because of money- and space issues. We also can determine the probability of a certain amount of totes at the conveyor, so this can help us to determine the capacity of that part of the system.

6.1.2 General number of pickstations

In this part we will discuss the CPS with a general number of stations. Let us first assume the symmetric case with two stations; both stations will receive half of the totes which are supplied by the HDS. The threshold will be the same for both stations, denoted as C. We could model this the same way as with one station and with probability of one half the totes travel from the conveyor to the first station and also with probability of one half to the second station. This model again has a product form and we can calculate anything we want to know.

However, this will not work because in this model we allow $2C$ totes on the conveyor, but only C totes in the buffers of the stations. In this model it is possible that $2C$ totes are present at the stations, so this model will not work. Now note that we are only interested in the situation at the station; we want to know the utilization, throughput and mean buffersize. This means that the performance of the conveyor and/or source is of secondary interest and could be investigated separately.

We will now model the system with general number of stations the same way as the one station model, with a closed product-form network with three stations. This will have the same solution as given in (6.1), with only one difference: the parameter of inter-arrival time. In this case this means for the rate of the source of station i, when there are z stations:

$$\lambda \longmapsto \frac{\lambda}{z} \;\; i = 1, \dots z.$$

We can now determine the performance measures in the same way as before for every station.

6.2 Simulation Project: Plus Retail

Within the Simulation group a simulation is made for Plus Retail with three workstations, see also [11]. Plus is a supermarket in the Netherlands. The simulation gives an accurate impression of what happens in this specific CPS. This simulation is adjusted and used to compare with our model. The adjustments include the use of an arbitrary number of workstations (with a maximum of 3), an infinite speed for the conveyor in front of the operator and only one family group. In the simulation the operator works according to an exponential distribution, so we do not have to adjust that. The HDS supplies new totes after a minimum time plus a Gamma distributed time, this will not be adjusted in this simulation. The Automod model is given in Figure 6.3, where we can see the three pickstations with buffer on the right, the sorter in the middle (conveyor-loop) and the HDS on the left (represented as blocks). Behind the three stations we can see the conveyor for finished orders.

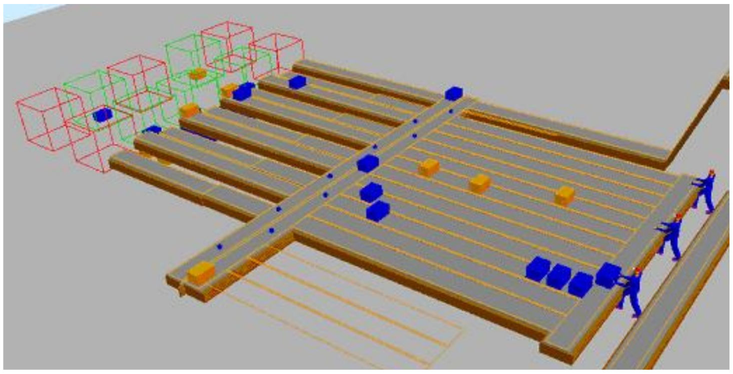


Figure 6.3: The Automod model for the CPS of Plus Retail

First we have to record what is the mean time that new totes are supplied from moment of asking and the mean time on the conveyor. We have done this by letting just a single tote travelling through the system. The mean pick time is already given, and eventually the standard parameters (as where chosen for Plus Retail) for this simulation and calculation will be:

1. Mean service time: $1/\mu = 12,5$ seconds.
2. Mean inter-arrival time: $1/\lambda = 38$ seconds.
3. Mean travel time: $1/\gamma = 60$ seconds.
4. Threshold: $C = 26$ totes.

In the following part we will vary one parameter of interest, namely the threshold and we will investigate the influence on the utilization, throughput per hour and the mean buffersize (L_z).

6.2.1 One pickstation

In this part we will discuss the results when there is only one pickstation. The first situation is the standard situation, with the standard parameters. We will present the results for the three performance measures mentioned earlier (Table 6.1). We can see that the results are very good, but also not very interesting because the utilization is very high. Then it is easy to get small

	Simulation	CPS Model	Δ%
ρ	0,998 ±0,001	1	0,2
δ	290,1 ±15,2	288,0	0,7
$\overline{N}$	17,7 ±0,3	18,2	2,8

Table 6.1: Simulation results compared with the CPS model, using the standard values

differences. Hence we will vary the threshold, because this will have the greatest impact on the performance of the system. We have chosen several values to see how well the model works: $C = 1, 3, 5$ and 10. Our main focus will be the utilization results and these are stated in Table 6.2. We can see that most of the results are acceptable, but the simulation results are not as reliable as we want. This because the runtime of the simulation is too small and the values fluctuate with a great range as can be seen from the output of the utilization when $C = 5$ in Figure 6.4. Here the utilization of the operator is given in time. The runtime is small because the simulation picks its data from a database and the simulation stops when there is no more data left. We could extend the database, but then the simulation will take much longer and it already takes a pretty long time. This is also a reason why analysis is a useful method. The results for throughput per hour and mean buffersize give similar acceptable results and will therefore not be presented here. The mean buffersize and the probability of a certain number of totes in the buffer will be discussed later.

C	Simulation	CPS Model	Δ%
1	0,117 ±0,013	0,113	3,3
3	0,327 ±0,023	0,330	0,8
5	0,510 ±0,026	0,529	3,7
10	0,833 ±0,024	0,886	6,3

Table 6.2: Simulation results compared with the CPS model of the utilization, when C is varied

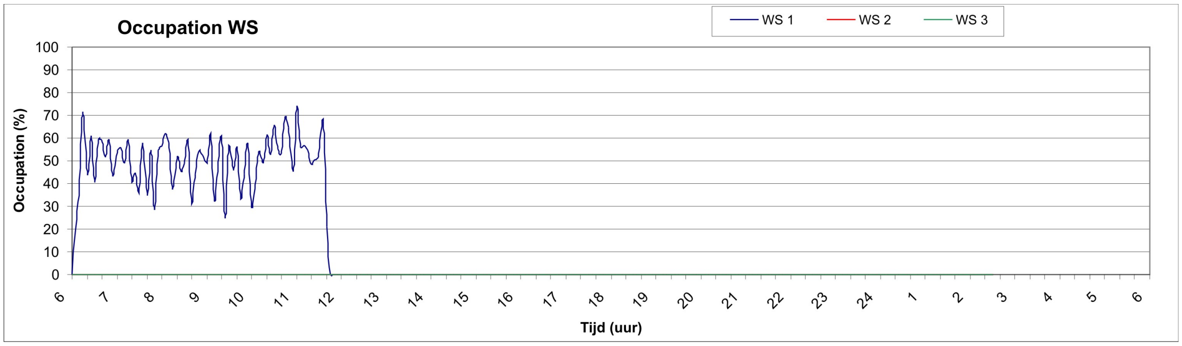


Figure 6.4: The utilization of the simulation against time, when $C = 5$

We have also changed the mean pick time of an operator, we have chosen two values higher and two lower than the standard value: $1/\mu = 5, 1/\mu = 10$ and $1/\mu = 15, 1/\mu = 20$. Here we also assume the threshold to be 10, because this will generate more interesting results than the standard value for the threshold. The most results are acceptable (see Table 6.3), only when the mean pick time is small the results are bad. This can be explained by the capacity of the cranes and sorter; if an operator works fast the time between the demand of new totes is small. In the model we can generate and supply these new totes, but in reality the cranes and sorters are limited by its capacities. This is also the reason why the value of the model is bigger than of the simulation. We can see this also happening when the threshold is getting bigger. Advantage is here that utilization will be big and relative errors therefore small. Also the simulation error is again big. For the situation when $1/\mu = 5$ we have recorded the mean time that a tote is travelling on the conveyor, this is not 60 seconds anymore because of the limited capacity of the sorter but approximately 72 seconds. Totes have to wait before they can get on the sorter. If we do the analysis with this parameter we obtain a utilization of $\rho = 0,424$, with a relative error of $2,2\%$ which is much better then the 13% we had earlier. In the next part we will compare the results of the simulation with the model when 2 respectively 3 workstations are used and they get an equal amount of work, followed by a non-symmetric situation with 2 stations.

	ρ			δ		
$1/\mu$	Simulation	CPS Model	$\Delta\%$	Simulation	CPS Model	$\Delta\%$
5	0,415 ±0,015	0,470	13,3	296,7 ±5,0	338,7	14,1
10	0,760 ±0,041	0,795	4,6	268,8±8,9	286,1	6,0
15	0,931 ±0,019	0,939	0,9	222,1 ±9,0	225,4	1,5
20	0,972 ±0,012	0,983	1,2	174,9 ± 10,2	177,0	1,2

Table 6.3: Simulation results compared with the CPS model of the utilization and throughput, when μ is varied

6.2.2 General number of pickstations

Here we will discuss the results when more stations are used. First we will do this for the 2 station case and we will investigate the two most important performance measures (utilization and throughput) just for the case when $C = 10$ and with the other standard parameters. After that we will do this for the system with 3 stations and in the end we will consider a non-symmetric case with 2 stations with different working speeds and so also different thresholds. The performance measures wil be approximately the same, but differ because of the travel time to the zone. We already have the travel time to station 1, and we have recorded the other travel times. The travel time to station 2 is 63 seconds and to station 3 approximately 66 seconds.

2 **pickstations**

We now consider the case when there are 2 pickstations and they get the same amount of work. This means that they have the same threshold $C = 10$. The results are given in Table 6.4 for both stations. We can see that the results are acceptable for the utilization for both stations and very good for the throughput per hour. The error of the simulation is big so it may happen that the results are worse, but there is also a probability that they are even getting better. In the next part we will see if this will be also the case for the system with 3 stations.

	Station 1			Station 2		
	Simulation	CPS Model	$\Delta\%$	Simulation	CPS Model	$\Delta\%$
ρ	0,726 ±0,049	0,746	2,7	0,710±0,045	0,735	3,5
δ	218,2 ±10,6	214,8	1,6	211,4 ±9,5	211,7	0,1

Table 6.4: Simulation results for 2 stations compared with the CPS model, when $C = 10$

3 **pickstations**

As said earlier we will present in this part the results for the 3 station case and threshold $C = 10$. We present only the results for station 1 and station 3, because these are most interesting and the results for station 2 are similar as the other two stations. Again we can see that the results are acceptable and this time the throughput results have approximately the same errors as the utilization results. We also see that the relative errors are bigger than in the 2 station case, again this can be explained by the fact that in reality we have limited capacity of sorter and cranes. Note that the simulation error is again very big.

	Station 1			Station 3		
	Simulation	CPS Model	$\Delta\%$	Simulation	CPS Model	$\Delta\%$
ρ	0,657 ±0,045	0,626	4,8	0,584 ±0,052	0,609	4,2
δ	187,4 ±11,5	180,1	3,9	177,1 ±13,4	175,4	1,0

Table 6.5: Simulation results for 3 stations compared with the CPS model, when $C = 10$

2 **pickstations: non-symmetric**

For this example we let the operator at the second zone work twice as slow as the first operator, so he has a mean pick time of 25 seconds. We now consider the case when $C = 10$ for the first station and the second operator has a threshold of $C = 5$. The results can be found in Table 6.6. Here we see that the utilization of the two stations are not equal to each other but the results of the simulation and model do give almost the same results. The relative errors are acceptable and this also holds for the throughput per hour. The only disadvantage is (again) the simulation error.

	Station 1			Station 2		
	Simulation	CPS Model	$\Delta\%$	Simulation	CPS Model	$\Delta\%$
ρ	0,772 ±0,067	0,746	3,4	0,683 ±0,070	0,671	1,7
δ	225,4 ±16,3	214,8	4,7	98,6 ±8,2	96,7	1,9

Table 6.6: Simulation results for 2 stations compared with the CPS model, non-symmetric case

6.3 Summary and Conclusion

In this chapter we made a step aside to another picking concept of Vanderlande Industries, called Compact Pick System (CPS). We observed that CPS is very similar to ZPS and the model of ZPS is adjusted so that an example of a CPS can be regarded. The main difference with ZPS is that when using CPS, parts are going to the operator and not the other way around. However note that in both systems totes are travelling to an operator which need to work on that tote for a certain time. This last similarity is the reason why the analysis is comparable.

This implies that we can adjust the Java application for ZPS a little bit, to also make it work for CPS. Overall the results are acceptable. There are however some small points which need to be noticed here, which influence the accuracy of the model.

The simulation error is big for every situation, therefore the accuracy of the results is rather hard to determine.
In the simulation the cranes and sorter have finite capacity, in the model this is neglected. This can be the cause for differences in results, especially when the operator is working very fast or the threshold is relatively big. This can be taken into account when this model extended.
For more than one pickstation we do not have a model which describes the system completely, but it does describe the situation at the station. There is also the possibility to include more types of totes into the model, this way the model can also include more pickstations.

We could use our simulation of the ZPS with one zone to compare it with our CPS model. We have already seen that when the buffersize is getting close to the threshold, the results for the iterative method with infinite server gives results close to the simulation. This gives enough reason to believe, together with the results of the simulation of Plus Retail, that the model for CPS is pretty accurate. More research is desirable, so the model can be further tested and improved.

Chapter 7

Conclusion and Recommendations

In this chapter we first state some conclusions concerning the developed model and tool, followed by several recommendations for further research.

7.1 Conclusion

In the beginning of this thesis we stated a goal for this research, namely:

"Develop a tool/model which can calculate or approximate the performance measures of different ZPS lay-outs in terms of utilization of the operators and throughput of totes per hour"

We can conclude that this goal is mostly reached.

The model for a ZPS that consists of one segment and multiple zones, describes the system very well and also the corresponding tool works very well. This model and tool can be used when these kind of small ZPS systems are considered. Besides utilization and throughput, the mean buffersize is also approximated which can also be useful when regarding this one segment system. The tool however only works well for a total number of zones up to 10.

The multi-segment ZPS model is also stated and only checked for two scenarios. The model and corresponding tool seems to describe these systems well, when varying several variables we see expected output. Because only two scenarios are considered, more research is needed and desirable.

The model is not accurate when regarding small traveltimes relative to the mean pick times. We have not determined when this will be the case, but we can roughly say that the total traveltime in a segment must be bigger than the mean pick time.

Furthermore there are some recommendations which are stated in the next section.

7.2 Recommendations

Although lot of effort is put in this thesis, there are some recommendations for future research. These recommendations are divided into two categories, namely general recommendations and the modelling of other picking concepts. First the general recommendations will be presented.

7.2.1 General recommendations

There are many general recommendations one can make, but here we will state the most important ones.

The most important recommendation is to validate the multi-segment model. Therefore more scenarios need to be run with the simulation and these results need to be compared with the output-results of the tool. In these scenarios the most important variables need to be altered, so we can say something more about the accuracy of the model.

We stated here a general multi-segment ZPS lay-out, but there are some variations known within Vanderlande. These variations include a shortcut from segment to segment and also a shortcut within a segment. This way a tote can reduce travelling time, because it does not have to complete the circulation within a segment. In our model this means an extra routing probability between stations and of course some adjustments to these routing probabilities.

Also in some layouts totes can leave the system earlier when they are done, this can be taken into account in our model by approximating these routing probabilities.

The tool can also be adjusted where wanted. More input variables can be provided as input; for each zone and conveyor the mean time can be given in separately. The same can be done for the buffersize of each zone and the threshold per segment. This can be done with a simple input field and in the program the variables should be changed to arrays. Costs can also be included in the tool, so a global sketch of the costs will be given for the chosen lay-out.

More output variables can also be written to the output file, like mean cycle time per segment or the total time in the system. The most important ones are already included in the program. These are extensions that are nice, but more interesting ones are reduction of calculation time and using the code generator:

Decreasing calculation time can be obtained by filling the transition matrix in smarter way, but especially by using the little number of types we observe when using exactly v number of zones to visit. If there is a nice way to number these types, the matrix will decrease rapidly in size and this step will last take less time. Also the virtual memory needed, will be less.

Using the code generator is something of the distant future. Within Vanderlande a code generator is being developed, which can alter Autocad drawings into Java code. This could also be applied to our model and tool, where conveyors will be represented as infinite servers with their mean time and zones as single servers with their mean time. This observation is somewhat premature, but here are possibilities.

7.2.2 Other picking concepts

Within Vanderlande Industries people think that the concepts of goods-to-man and man-to-goods are completely different from each other. In reality this is true, but for modelling of these systems this does not really matter. Totes travel to an operator and this operator works on this tote for a certain amount of time. This can be putting a product in this tote or grabbing a product from this tote, but this does not matter for our model.

We have seen this when studying the picking concept of CPS. With minor adjustments the model of ZPS could be converted to a model for CPS. In this thesis this model and system is briefly investigated and discussed. Therefore more research is necessary to refine and expand the model.

Also other concepts can be modelled using closed queueing networks. The concept of order distribution (ODS) is like ZPS but the role of the orders and products are switched. Totes with products are passing to the zones and there an order is put together. However the difference is that when a product tote is empty, this tote is being removed from the system and after a while a full tote will be inserted. This will cause loss of throughput, our ZPS model therefore can give an upper bound on the utilization and throughput. A more thorough look at these systems is necessary to see if our model is really applicable.

Appendix A

Saturated Models

A.1 Erlang-2 model

Flow diagram

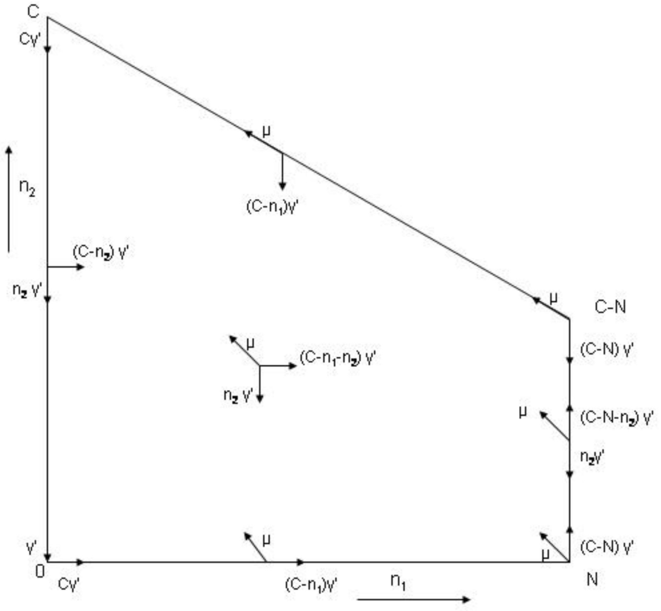


Figure A.1: The transition rates of the saturated model with Erlang-2 traveltimes

Balance equations

$$\begin{aligned}
(\mu+(C-n_1)\gamma')P(n_1,n_2) &= \mu P(n_1+1,n_2-1)+(n_2+1)\gamma' P(n_1,n_2+1)\\
&+ (C-n_1-n_2+1)\gamma' P(n_1-1,n_2),\\
&\quad 1\leq n_1\leq N-1, 1\leq n_2\leq C-1, n_1+n_2<C\\
C\gamma' P(0,n_2) &= \mu P(1,n_2-1)+(n_2+1)\gamma' P(0,n_2+1),\ \ 1\leq n_2\leq C-1,\\
(\mu+(C-N)\gamma')P(N,n_2) &= (n_2+1)\gamma' P(N,n_2+1)+(C-N-n_2+1)\gamma' P(N-1,n_2)\\
&+ (C-N-n_2+1)\gamma' P(N,n_2-1),\ \ 1\leq n_2\leq C-N-1,\\
(\mu+(C-n_1)\gamma')P(n_1,0) &= \gamma' P(n_1,1)+(C-n_1+1)P(n_1-1,0),\ \ 1\leq n_1\leq N,\\
(\mu+(C-n_1)\gamma')P(n_1,C-n_1) &= \mu P(n_1+1,C-n_1-1)+\gamma' P(n_1-1,C-n_1),\ \ 1\leq n_1\leq N-1,\\
C\gamma' P(0,C) &= \mu P(1,C-1),\\
(\mu+(C-N)\gamma')P(N,C-N) &= P(N,C-N-1)\gamma'+\gamma' P(N-1,C-N),\\
C\gamma' P(0,0) &= \gamma' P(0,1).
\end{aligned}$$

Substitution of producform in first balance equation

$$\begin{aligned}
(q\mu+(C-n_1)\gamma')\left\{(\frac{\lambda}{\mu})^{n_1}\frac{1}{n_2!}\frac{1}{n_3!}(\frac{\lambda}{\gamma'})^{n_2+n_3}\right\} &= \mu\left\{(\frac{\lambda}{\mu})^{n_1+1}\frac{1}{(n_2-1)!}\frac{1}{n_3!}(\frac{\lambda}{\gamma'})^{n_2+n_3-1}\right\}\\
&+ (n_2+1)\gamma'\left\{(\frac{\lambda}{\mu})^{n_1}\frac{1}{(n_2+1)!}\frac{1}{(n_3-1)!}(\frac{\lambda}{\gamma'})^{n_2+n_3}\right\}\\
&+ (n_3+1)\gamma'\left\{(\frac{\lambda}{\mu})^{n_1-1}\frac{1}{n_2!}\frac{1}{(n_3+1)!}(\frac{\lambda}{\gamma'})^{n_2+n_3+1}\right\}.
\end{aligned}$$

Cancelling the common terms $(\frac{\lambda}{\mu})^{n_1-1}$ and $(\frac{\lambda}{\gamma'})^{n_2+n_3-1}$ gives

$$\begin{aligned}
(q\mu+(C-n_1)\gamma')(\frac{\lambda}{\mu})\frac{1}{n_2!}\frac{1}{n_3!}(\frac{\lambda}{\gamma'}) &= \mu\left\{(\frac{\lambda}{\mu})^2\frac{1}{(n_2-1)!}\frac{1}{n_3!}\right\}\\
&+ (n_2+1)\gamma'\left\{(\frac{\lambda}{\mu})\frac{1}{(n_2+1)!}\frac{1}{(n_3-1)!}(\frac{\lambda}{\gamma'})\right\}\\
&+ (n_3+1)\gamma'\left\{\frac{1}{n_2!}\frac{1}{(n_3+1)!}(\frac{\lambda}{\gamma'})^2\right\}.
\end{aligned}$$

Simplifying and cancelling the common factorial terms

$$\begin{aligned}
(\mu+(C-n_1)\gamma')(\frac{\lambda}{\mu})\frac{1}{n_2}\frac{1}{n_3}(\frac{\lambda}{\gamma'}) &= \mu\left\{(\frac{\lambda}{\mu})^2\frac{1}{n_3}\right\}+\gamma'\left\{(\frac{\lambda}{\mu})\frac{1}{n_2}(\frac{\lambda}{\gamma'})\right\}\\
&+ \gamma'\left\{\frac{1}{n_2}\frac{1}{n_3}(\frac{\lambda}{\gamma'})^2\right\}.
\end{aligned}$$

Again with simple algebra cancelling the common terms we obtain

$$(C-n_1)\frac{1}{n_2}\frac{1}{n_3}=\frac{1}{n_3}+\frac{1}{n_2},$$

which balances out, because $n_1+n_2+n_3=C$. The normalizing constant is given by summing over all possible states, which are in the state space.

A.2 Hyper-exponential model

Flow diagram

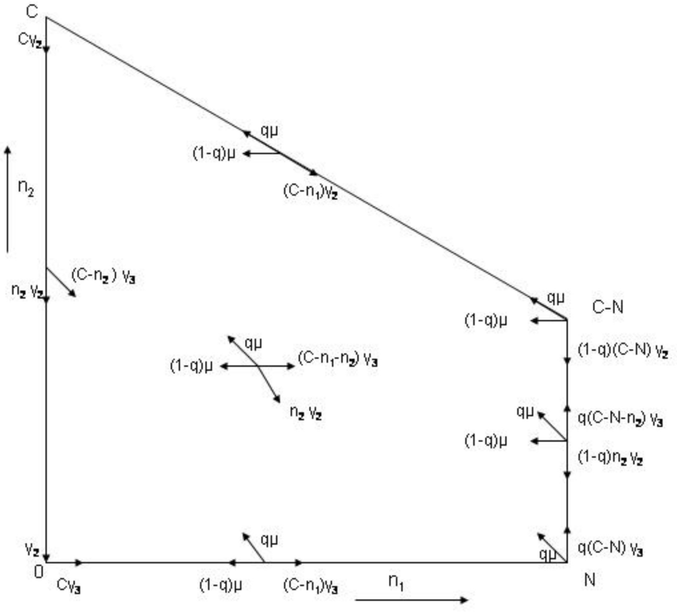


Figure A.2: The transitions rates for the Markov chain of the saturated hyper-exponential like retrial queue

Balance equations

$$
\begin{aligned}
(\mu + n_2\gamma_2 + (C - n_1 - n_2)\gamma_3)P(n_1, n_2) &= q\mu P(n_1 + 1, n_2 - 1) + (1 - q)\mu P(n_1 + 1, n_2) \\
&+ (n_2 + 1)\gamma_2 P(n_1 - 1, n_2 + 1) + (C - n_1 - n_2 + 1)\gamma_3 P(n_1 - 1, n_2), \\
&\quad 1 \leq n_1 \leq N - 1, 1 \leq n_2 \leq C - 1, n_1 + n_2 < C \\
(n_2\gamma_2 + (C - n_2)\gamma_3)P(0, n_2) &= q\mu P(1, n_2 - 1) + (1 - q)\mu P(1, n_2), \;\; 1 \leq n_2 \leq C \\
(\mu + (1 - q)n_2\gamma_2 + q(C - N - n_2)\gamma_3)P(N, n_2) &= (n_2 + 1)\gamma_2 P(N - 1, n_2 + 1) + (C - N - n_2 + 1)\gamma_3 P(N - 1, n_2) \\
&+ (1 - q)(n_2 + 1)\gamma_2 P(N, n_2 + 1) + q(C - N - n_2 + 1)\gamma_3 P(N, n_2 - 1), \\
&\quad 1 \leq n_2 \leq C - N - 1 \\
(\mu + (C - n_1)\gamma_3)P(n_1, 0) &= (C - n_1 + 1)\gamma_3 P(n_1 - 1, 0) + \gamma_2 P(n_1 - 1, 1), \\
&+ (1 - q)\mu P(n_1, +1, 0), \;\; 1 \leq n_1 \leq N, \\
(\mu + (C - n_1)\gamma_2)P(n_1, C - n_1) &= q\mu P(n_1 + 1, C - n_1 - 1) + (C - n_1 + 1)\gamma_2 P(n_1 - 1, C - n_1 + 1) \\
&+ \gamma_3 P(n_1 - 1, C - n_1), \;\; 1 \leq n_1 \leq N - 1,
\end{aligned}
$$

$$\begin{aligned} C\gamma_2 P(0,C) &= q\mu P(1,C-1),\\ (\mu+(1-q)(C-N)\gamma_2)P(N,C-N) &= (C-N+1)\gamma_3 P(N-1,C-N+1)+\gamma_2 P(N-1,C-N)\\ &= q\gamma_3 P(N,C-N-1),\\ C\gamma_3 P(0,0) &= (1-q)\mu P(1,0).\end{aligned}$$

Substitution of producform in first balance equation

$$\begin{aligned} & (\mu+n_2\gamma_2+n_3\gamma_3)\left\{(\frac{\lambda}{\mu})^{n_1}\frac{1}{n_2!}(\frac{q\lambda}{\gamma_2})^{n_2}\frac{1}{n_3!}(\frac{(1-q)\lambda}{\gamma_3})^{n_3}\right\}\\ = & \quad q\mu\left\{(\frac{\lambda}{\mu})^{n_1+1}\frac{1}{(n_2-1)!}(\frac{q\lambda}{\gamma_2})^{n_2-1}\frac{1}{n_3!}(\frac{(1-q)\lambda}{\gamma_3})^{n_3}\right\}\\ + & \quad (1-q)\mu\left\{(\frac{\lambda}{\mu})^{n_1+1}\frac{1}{n_2!}(\frac{q\lambda}{\gamma_2})^{n_2}\frac{1}{(n_3-1)!}(\frac{(1-q)\lambda}{\gamma_3})^{n_3-1}\right\}\\ + & \quad (n_2+1)\gamma_2\left\{(\frac{\lambda}{\mu})^{n_1-1}\frac{1}{(n_2+1)!}(\frac{q\lambda}{\gamma_2})^{n_2+1}\frac{1}{n_3!}(\frac{(1-q)\lambda}{\gamma_3})^{n_3}\right\}\\ + & \quad (n_3+1)\gamma_3\left\{(\frac{\lambda}{\mu})^{n_1-1}\frac{1}{n_2!}(\frac{q\lambda}{\gamma_2})^{n_2}\frac{1}{(n_3+1)!}(\frac{(1-q)\lambda}{\gamma_3})^{n_3+1}\right\}\end{aligned}$$

Deleting the common terms $(\frac{\lambda}{\mu})^{n_1-1}$, $(\frac{q\lambda}{\gamma_2})^{n_2-1}$ and $(\frac{(1-q)\lambda}{\gamma_3})^{n_3-1}$ to obtain the following equation

$$\begin{aligned} (\mu+n_2\gamma_2+n_3\gamma_3)\left\{(\frac{\lambda}{\mu})\frac{1}{n_2!}(\frac{q\lambda}{\gamma_2})\frac{1}{n_3!}(\frac{(1-q)\lambda}{\gamma_3})\right\} &= q\mu\left\{(\frac{\lambda}{\mu})^2\frac{1}{(n_2-1)!}\frac{1}{n_3!}(\frac{(1-q)\lambda}{\gamma_3})\right\}\\ &+ (1-q)\mu\left\{(\frac{\lambda}{\mu})^2\frac{1}{n_2!}(\frac{q\lambda}{\gamma_2})\frac{1}{(n_3-1)!}\right\}\\ &+ (n_2+1)\gamma_2\left\{\frac{1}{(n_2+1)!}(\frac{q\lambda}{\gamma_2})^2\frac{1}{n_3!}(\frac{(1-q)\lambda}{\gamma_3})\right\}\\ &+ (n_3+1)\gamma_3\left\{\frac{1}{n_2!}(\frac{q\lambda}{\gamma_2})\frac{1}{(n_3+1)!}(\frac{(1-q)\lambda}{\gamma_3})^2\right\}\end{aligned}$$

Simplifying the factorial terms and deleting the common factorial terms

$$\begin{aligned} (\mu+n_2\gamma_2+n_3\gamma_3)\left\{(\frac{\lambda}{\mu})\frac{1}{n_2}(\frac{q\lambda}{\gamma_2})\frac{1}{n_3}(\frac{(1-q)\lambda}{\gamma_3})\right\} &= q\mu\left\{(\frac{\lambda}{\mu})^2\frac{1}{n_3}(\frac{(1-q)\lambda}{\gamma_3})\right\}+(1-q)\mu\left\{(\frac{\lambda}{\mu})^2\frac{1}{n_2}(\frac{q\lambda}{\gamma_2})\right\}\\ &+ \gamma_2\left\{\frac{1}{n_2}(\frac{q\lambda}{\gamma_2})^2\frac{1}{n_3}(\frac{(1-q)\lambda}{\gamma_3})\right\}+\left\{\gamma_3\frac{1}{n_2}(\frac{q\lambda}{\gamma_2})\frac{1}{n_3}(\frac{(1-q)\lambda}{\gamma_3})^2\right\}\end{aligned}$$

Expanding these expressions gives us

$$\begin{aligned} \frac{(1-q)q\lambda^3}{\gamma_2\gamma_3}\frac{1}{n_2n_3}+\frac{(1-q)q\lambda^3}{\mu\gamma_3}\frac{1}{n_3}+\frac{(1-q)q\lambda^3}{\mu\gamma_2}\frac{1}{n_2} &= \frac{(1-q)q\lambda^3}{\mu\gamma_3}\frac{1}{n_3}+\frac{(1-q)q\lambda^3}{\mu\gamma_2}\frac{1}{n_2}\\ &+ \frac{(1-q)q^2\lambda^3}{\gamma_2\gamma_3}\frac{1}{n_2n_3}+\frac{(1-q)^2q\lambda^3}{\gamma_2\gamma_3}\frac{1}{n_2n_3}\end{aligned}$$

Equality between expressions follow from the fact that for all q

$$(1-q)q=(1-q)q^2+(1-q)^2q,$$

References

[1] I. Adan (2005).
Lecture Notes: Block Queueing Theory and Simulation
http://www.win.tue.nl/ iadan/blockq.

[2] I. Adan & J. Resing (2001).
Lecture Notes: Queueing Theory.
Department of Mathematics and Computing Science, Eindhoven University of Technology.
http://www.win.tue.nl/ iadan/que.

[3] G. Bolch, S. Griener, H. de Meer, K.S. Trivedi (1998).
Queueing Networks and Markov Chains.
John Whiley & Sons.

[4] R.J. Boucherie & N.M. van Dijk (1992).
A Generalization of Norton's Theorem for Queueing Networks.
Queueing Systems **13**, pp 251-289.

[5] J.W. Cohen (1957).
Basic Problems of Telephone Traffic Theory and the Influence of Repeated Calls.
Phillips Telecommunication Review **18**, No. 2, pp 49-100.

[6] N.M. van Dijk (1988).
On Jackson's Product Form with 'Jump-Over' Blocking.
Operations Research Letters **7**, No. 5, pp 233-235.

[7] B. Dumoulin (2004).
Zone Picking Systems concept application document.
Vanderlande Industries, intern report.

[8] G.I. Falin & J.G.C. Templeton (1997).
Retrial Queues.
Chapman and Hall.

[9] G.I. Falin (1999).
A Multiserver Retrial Queue with a Finite Number of Sources of Primary Calls.
Mathematical Modeling and Computer Modeling **30**, pp 33-49.

[10] S.K. Godunov & V.S. Ryabenkii (1987).
Difference Schemes.
North Holland, pp 47-50.

[11] J. Goes (2006).
Simulatie Rapport: NR 52202-086-12101-NL-B.
Vanderlande Industries, internal document.

[12] B.S. Greenberg & R.W. Wolff (1987).
An Upper Bound on the Performance of Queues with Returning Customers.
Journal of Applied Probability **24**, No. 2, pp 466-475.

[13] B.O. Heimsund (2006).
Matrix Toolkit for Java
http://ressim.berlios.de/

[14] J. Keilson, J. Cozzolino & H. Young (1968).
A Service System with Unfilled Requests Repeated.
Operations Research **16**, pp 1126-1137.

[15] M.F. Neuts & B.M. Rao (1990).
Numerical Investigations of a Multiserver Retrial Model.
Queueing Systems **7**, pp 169-190.

[16] A. Renyi (1958).
On a One Dimensional Problem Concerning Random Space Filling.
Publ. Math. Inst. Hung. Acad. Sci. **7**, pp 109-127.

[17] Y. Saad (1996).
Iterative Methods for Sparse Linear Systems
www.win.tue.nl/maubach/university/education/lectures/2n070/inf/index.html

[18] *Vanderlande Industries Annual report 2006: In a world of technology a belief in people.*
Vanderlande Industries, intern report.