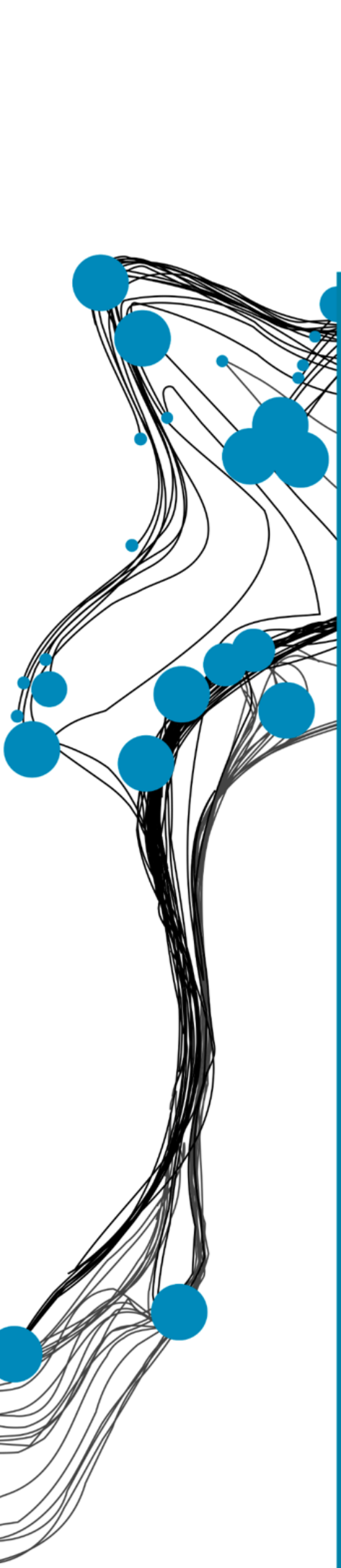

M.Sc. Thesis Applied Mathematics

Capacity Planning in Queueing Networks

An Iterative Method which Combines the Queueing Network Analyser and Simulation

Lotte C. Gerards, B.Sc.
s1851233

Graduation Committee:
Dr. Ir. A. Braaksma
Prof. Dr. R.J. Boucherie
Prof. Dr. J.L. Hurink

September 29, 2021

Stochastic Operations Research
Department of Applied Mathematics
Faculty of Electrical Engineering,
Mathematics and Computer Science

UNIVERSITY OF TWENTE.

Acknowledgements

Before you lies the master thesis *Capacity Planning in Queueing Networks; An Iterative Method which Combines the Queueing Network Analyser and Simulation*, written as the final project for the master Applied Mathematics at the University of Twente. Working on this thesis did not come without its difficulties and I have learned a lot during the process. Therefore, I would like to use this page to thank the people who helped and supported me.

Firstly, I would like to thank to my supervisor Dr. Ir. Aleida Braaksma. Thank you for your guidance and smart comments. After each of our weekly meetings I always had a lot of new ideas and knew how to proceed even if I got stuck before. I would also like to thank my second supervisor Prof. Dr. Richard Boucherie and the third member of my graduation committee, Prof. Dr. Johann Hurink.

This thesis was carried out in cooperation with the Dutch hospital ZGT. Therefore, I would like to thank my supervisors from ZGT, Barbara Koenis and Anja Meijnders, for the information they provided and their enthusiastic comments. Next, I would like to thank Laura Ooms for her help with the data and Dr. Irma Oving for giving me insight in the care pathways of cancer patients and for letting me join in the hospital for a day. Furthermore, I would also like to thank everyone from ZGT who helped me for the insightful and interesting conversations.

I would like to thank my friends, Annemarie, Femke, Jarco, Jente, Leander, Lucas, Sven, Tessa and Wisse, not only for their support during the process of writing this thesis, but also for the last 5 years of studying together. Lastly, I would like to thank my parents, my sister and my brother for supporting me throughout the years and always standing by my side.

I hope you enjoy reading this thesis.

Lotte Gerards
Stavanger, September 29th, 2021

Summary

Queueing networks are used in many different applications to analyse processes on performance measures, like expected waiting time and number of customers present in a system. When we consider a queueing network with general arrival and service processes, these performance measures cannot always be determined in closed form. If this is the case, approximation methods or simulation are often used to analyse the system.

In this thesis we develop a method to determine the capacity planning in a queueing network. This capacity planning should satisfy two conditions. Firstly, at least a certain percentage of the customers must finish certain parts of a route through the network within a given target time. Secondly, the servers must work efficiently, meaning that not more capacity is added than needed to satisfy condition one. In order to satisfy both conditions we combine an approximation method called the Queueing Network Analyser (QNA) and Discrete Event Simulation into an iterative method in order to determine the capacity planning.

This iterative method is first tested on a simple queueing network, for which the optimal capacity planning is known. We observe here that the capacity allocation determined by the iterative method is very close the theoretical optimum for this network. After that we conduct a case study for the cancer department of the Dutch hospital ZGT located in Hengelo and Almelo. This case study shows that the capacity given by the iterative method strongly depends on the initialisation of the method. Therefore, we propose a different initialisation method at the end of the thesis. Lastly, we conclude that the iterative method developed in this thesis combines QNA and optimisation well and is suitable for capacity planning problems with strict completion time requirements.

Contents

Chapter 1

Introduction

A queueing network is a collection of stations, each with one or multiple servers and a waiting room. Customers or jobs arrive to the network with a certain rate and follow a route through it, encountering multiple stations. When a customer arrives at a station and the server is busy, the customer waits until it is his turn to be served. By transforming a complex real-life process into a queueing network, one can analyse performance measures of the system like the expected waiting time and number of customers present.

Queueing networks have a wide range of applications and are often used to evaluate the performance of manufacturing and communication systems, see among others the reviews [1] and [2]. For some special cases, like Jackson networks with exponential interarrival and service times [3] [4], and Kelly Whittle networks [5] the performance measures can be determined in closed form. Furthermore, a technique called Mean Value Analysis can be used to obtain the expected value of performance measures for single queues with exponential interarrival times and general service time distributions [6]. For the vast majority of queueing networks, where exponential interarrival or service times are not natural anymore, information about the waiting time and number of customers in the system cannot be determined in closed form [6]. For these queueing networks simulations or approximation methods can be used [7].

Both simulation and approximation methods have advantages and disadvantages. A simulation can resemble the behaviour of a complex network very well, but this method is also known for its computational costs and becomes intractable when a network should be analysed multiple times [8]. An example of an approximation method is the Queueing Network Analyser, or for short QNA, presented in 1983 in the paper of Whitt [9]. QNA is a parametric decomposition method, which decomposes a queueing network into single queues. The only parameters required for this method are the first and second moment of both the interarrival times of customers entering the network and the service times of customers at stations. This enables us to approximate the performance measures of a queueing network with only a small amount of input variables and computational steps and hence in a limited amount of time. However, since it is an approximation method the results could differ from reality.

In this thesis we are interested in the capacity needed at every station in a queueing network to satisfy the demand, while satisfying strict requirements regarding completion times of parts of a customer's route. With strict requirements we mean that at least a certain percentage of the customers must finish their route part within a given time. One way to solve this would be to assign a very high capacity to every station. However, most application fields for queueing networks have constraints on the available capacity. Hence, the servers' time should be used efficiently and no unnecessary capacity should be added. For this problem we formulated the following research objective:

We aim to design a capacity allocation method for queueing networks, where we minimise the sum of capacities needed to meet the strict requirements regarding the completion times of customers' route parts.

The capacity allocation method developed in this report uses both QNA and simulation. We propose to use the advantage of one technique – closely resembling reality for simulation and low computational costs for QNA – to compensate for the disadvantage of the other – high computational costs for simulation and an approximation error for QNA. We have to take into account here that when QNA is used in combination with standard optimisation techniques, the objective function should be convex in order to be certain that an optimum can be found by these methods. Since the performance measures, like the waiting time and number of customers in the system, depend non-linearly on both the first and second moment of the service time in QNA, this is not the case for this problem [10]. Therefore, we design an iterative method that uses QNA to come to a capacity allocation fast and that uses simulation to check whether the given allocation actually satisfies the completion time requirements. We answer the following sub-questions:

1. *In what way is QNA used for optimisation in literature?*
2. *What problems do we face when we use QNA for optimisation and how can these problems be overcome?*
3. *How can we combine QNA and simulation in order to incorporate the strict completion time requirements?*
4. *How well does the designed method perform?*

To answer the last sub-question, we check the method with a simple test network, for which we can determine the performance measures in closed form. Afterwards, the iterative method is used in a case study for the Dutch hospital ZGT. At the beginning of 2022, this hospital will open an outpatient oncology clinic, where multiple oncology professionals work together and have outpatient appointments. For this clinic we want to know the needed capacity to handle the demand for outpatient appointments and satisfy the requirements set regarding completion times of stages in a patient's hospital journey.

This report is organised in the following way. In Chapter 2 we discuss the literature available about QNA in combination with optimisation problems and answer sub-

question 1. Thereafter, in Chapter 3 we adress sub-question 2 and give a description of the queueing network and the mathematical formulation of the problem. Additionally, Chapter 3 considers sub-question 3 and contains a description of the developed iterative method to find the optimal capacity allocation. We answer sub-question 4 by looking at a small test network and the case study for the Dutch hospital ZGT in Chapter 4. Lastly, in Chapter 5 we draw a conclusion using the results of our iterative method and end with recommendations for further research. An overview of all mathematical notation introduced in this report can be found in Appendix A.

Chapter 2

Literature review

In this chapter we look at the research done regarding the Queueing Network Analyser in combination with optimisation. QNA was developed by Bell Laboratories and was first introduced under this name in [9]. The method is an extension of the decomposition method introduced in [11] by Reiser and Kobayashi and improved by Kuehn [12]. Further developments of this method are described in for example [13], where they also include features if manufacturing systems like breakdowns and batch services, [14], where they include the effect of customers interfering with each other, and [15], where they remove some approximations done in QNA to obtain an approximation method for queues with phase-type interarrival and service time distributions.

QNA has been applied in different fields and is used for a wide range of applications, of which we give some examples here. QNA was used by Haverkort to analyse open queueing networks with finite buffer queues in [16]. Furthermore, Bhatia used QNA to analyse Mobile Ad-Hoc Networks, also known as MANET [17]. QNA is also used in the field of communication in the papers by Schneider et al. [18] and Heijink et al. [19]. Zonderland et al. [20] analysed a preanesthesia evaluation clinic on length of stay of patients and utilisation of the employees using QNA, where they also showed the performance of alternative designs of the clinic. Both Alenany and El-Baz [21] and Albin et al. [22] also applied QNA in a hospital setting. Alenany and El-Baz analysed the waiting times of patients in a hospital in Egypt, while Albin et al. use QNA to decrease the waiting time in a health care centre.

For our problem, we are interested in optimising the capacity in a queueing network. In order to do so, we use the approximation method QNA. Therefore, we now present literature where QNA is used in combination with optimisation. We focus on papers addressing the problem of resource allocation, but extend our view to include other optimisation problems that were investigated using QNA.

The paper of Bitran and Morabito [10] discusses capacity planning in manufacturing systems with open queueing networks. They look at the optimisation of two problem classes by changing the capacity: minimise the costs of a network under performance constraints of the system and minimise the performance costs by allocating the avail-

able capacity. This is formulated as a linear programme and done, among others, for networks with general arrival and service time distributions. Here QNA is used as an analysis technique. For general networks they assume the squared coefficient of the service and interarrival times does not change when the capacity changes, to ensure that the function which is minimised is convex. In the paper of Bitran and Tirupati [23] this assumption is substantiated for certain circumstances. An application of this method can be found in the paper of Silva et al. [24]. They apply it to the job-shop of a Brazilian metallurgical plant. The performance measures found with the machine allocation determined through the linear programmes are checked with a simulation. In total six different decomposition approximations were tested and the authors concluded that the best performing approximation came sufficiently close to the results of the simulation.

The paper of Hopp et al. [25] uses QNA to plan the capacity of new semiconductor fabrication facilities. Their aim is to minimise the facility costs but still meet the requirements regarding mean cycle time and volume. Unlike our problem, they deal with batch processes in their queueing network. An iterative method is used, which adds capacity to one station every iteration, to determine the optimal resource allocation. The results obtained with the iterative method are compared to results obtained via simulation. The authors note that there can be a big error between the two in some situations, but conclude that for their context and problem this is acceptable. Another paper about resource allocation with QNA in a semiconductor fabrication facility is written by Connors et al. [26]. Among other things, they take into account tool breakdown and rework. They minimise the costs of the resources used with a marginal allocation algorithm, that iteratively adds one tool to the station that causes the largest reduction in cycle-time until all requirements with respect to the mean cycle-time are met. For a short numerical example, most performance measures given by QNA were within or close to the confidence interval of these performance measures given by simulation. In this paper it is proposed to use simulation for fine-tuning by using it to explore the neighbourhood of the optimal capacity found by QNA, but this idea is not executed.

Takemoto and Arizono [27] look at the resource allocation in a manufacturing system with general interarrival and service time distributions and one type of job. Their aim is to optimise the tardiness costs, which are dependent on the distribution of the lead time. This distribution is hard to determine for a system with general interarrival and service time distributions. The system is analysed with QNA and a distribution free approach is used to find an upper bound for the expected time between the required lead time and the lead time given by QNA. The optimisation problem is formulated in a non-linear programme and solved using the Lagrange multiplier method.

QNA has also been used in other optimisation problems than resource allocation. The paper written by Morabito et al. [28] uses QNA in combination with a cycle cancelling algorithm to find routes between nodes in a graph that minimise the arc costs. The waiting time in a hospital department is minimised in the paper of Creemers and Lambrecht [29] by finding the optimal number of patients to be served during a service session. Another health care application of QNA and optimisation can be found in the paper of Van Brummelen et al. [30], where QNA is used to determine a lower bound

for the staff members needed in a blood collection organisation such that the expected waiting time is below a certain level. Furthermore, QNA is not the only approximation technique used in combination with optimisation in a queueing network. Both Cruz and Van Woensel [31] and Kerbache and Smith [32], for example, used the Generalised Expansion Method.

From these papers we can conclude that QNA has been used in combination with optimisation several times. For resource planning often requirements regarding the expected cycle time should be met. In many of the cases an iterative method is used to come to the optimal capacity allocation and which is subsequently tested with a simulation. However, we do not see strict completion time constraints over smaller parts of a customer's route in literature. Papers look at requirements for the expected cycle time, instead of requirements for a percentage of customers that must finish their route part within a given target time, as we are interested in. In some fields, like for example certain departments in health care, we have to deal with these strict requirements.

In this thesis, we present an iterative method which combines QNA and simulation in order to find the best resource allocation in a queueing network. This is done under strict completion time requirements for stages of a route and servers should use their time efficiently. To deal with these requirements, a method is designed which combines QNA and discrete event simulation. To the best of our knowledge, the combination of QNA and simulation into one optimisation method has not been made before. Often simulation is used to check the results afterwards, but it is not included in the optimisation technique.

Chapter 3

Capacity planning

This chapter describes the mathematical methods used in this thesis for capacity planning. First we describe the kind of queueing network we consider. Thereafter, we formulate the problem in a mathematical programme. Then, we introduce a new iterative method, which combines QNA and simulation. Each step of the iterative method is described extensively in this chapter. Lastly, we look at the convergence of the iterative method. An overview of all mathematical notation and variables introduced in this chapter, including a description, can be found in Appendix A.

3.1 The queueing network

In this thesis, we consider an open queueing network. Suppose that the customers need to visit a subset of J different stations, so we have as set of stations $\{1, 2, \ldots, J\}$. Each station is modelled as a single server queue, with general arrival processes and general service time distributions, $G/G/1$ queues in Kendall's notation. Customers arrive from outside of the queueing system and after a number of visits to some stations in the network they leave the system again. The outside will be referred to as station 0. At each station customers are served in order of arrival.

We consider multiple customer classes. The number of possible customer classes is denoted by K. The customers that arrive at the system will follow their own path through the different stations. For each of the customer classes different requirements hold regarding the time spent in certain stages of their path. We distinguish a number of types within a customer class that resemble the possible pathways of this class. Class $k \in \{1, 2, \ldots, K\}$ will have the set of types S_k. The total number of types is then given by $S = |S_1| + |S_2| + \cdots + |S_K|$ and therefore we number the types from 1 until S. A customer of class k is of type $s \in S_k$ with probability p_{ks}, with $\sum_{s \in S_k} p_{ks} = 1$ for all $k \in \{1, 2, \ldots, K\}$. Every type has a fixed route through the network. We denote this in the following way for type s:

$$r(s, 1), r(s, 2), \ldots, r(s, L(s)),$$

where $r(s,1)$ denotes the station visited first by the customer and $r(s,L(s))$ denotes the station visited last before leaving the network. Hence $L(s)$ denotes the total number of stations that a customer of type s visits. A customer may visit the same station multiple times. The mean service time of a customer at station $i = r(s,v)$ is given by $\mathbb{E}[G_i^{(s,v)}]$ for $i \in \{1,2,\ldots,J\}$, $s \in \{1,2,\ldots,S\}$ and $v \in \{1,2,\ldots,L(s)\}$.

Customers of type s arrive from the outside to station $r(s,1)$ with arrival rate $\gamma_0^{(s)}$. The arrival rate at station j of customers of type s in stage v of their route, for $s = 1,2,\ldots,S$ and $v = 1,2,\ldots,L(s)$, is given by:

$$\lambda_j^{(s,v)} = \begin{cases} \gamma_0^{(s)}, & \text{if } j = r(s,v) \\ 0, & \text{otherwise.} \end{cases}$$

The total arrival rate of a customer of type s to station j, $\lambda_j^{(s)}$, is then given by:

$$\lambda_j^{(s)} = \sum_{v=1}^{L(s)} \lambda_j^{(s,v)}, \qquad \text{for } j \in \{1,2,\ldots,J\} \text{ and } s \in \{1,2,\ldots,S\}. \tag{3.1}$$

Now that we set up the notation of the queueing network, we can move on to the optimisation problem.

3.2 Optimisation problem

The goal is to determine the capacity needed at every station to handle the demand and satisfy the completion time requirements. In order to change the capacity, we change the server's speed. Each station has a single server, but the servers can work with a different speed. We define the server's speed at station $i \in \{1,2,\ldots,J\}$ as β_i. This β_i can also seen as the fraction of time a server is working in the given time period. We assume this fraction can also be bigger than one. We can include the server speed in our queueing model in the following way. If the mean service length is expressed in the given time period, we obtain the following service rate for customers of type s in stage v at station i:

$$\mu_i^{(s,v)} = \frac{\beta_i}{\mathbb{E}[G_i^{(s,v)}]},$$

for $i \in \{1,2,\ldots,J\}$, $s \in \{1,2,\ldots,S\}$ and $v \in \{1,2,\ldots,L(s)\}$. This fraction β_i thus determines the rate with which the single server is working. We assume that the capacity is equally divided over the given time period, meaning that the server works full-time, but with a lower speed. Hence we need to adjust the service time with this fraction as well. If a server works with twice the rate, he works twice as fast, hence a customer will be finished in half the original service time. Therefore, we have as adjusted mean service time:

$$\mathbb{E}[B_i^{(s,v)}] = \frac{1}{\mu_i^{(s,v)}} = \frac{\mathbb{E}[G_i^{(s,v)}]}{\beta_i}.$$

This adjusted service time could be larger or smaller than the original service time, depending on the available capacity. For the variance of the service time we have the following:

$$\mathrm{Var}(B_i^{(s,v)}) = \frac{\mathrm{Var}(G_i^{(s,v)})}{\beta_i^2}.$$

Note that the squared coefficient of variation of the service time does not change.

Now we can formulate our problem as a mathematical optimisation problem. We are looking for the state of our system, a capacity allocation $\hat{\beta} = (\beta_1, \beta_2, \ldots, \beta_J)$, which optimises our queueing network. In order to do so, we need to take into account two main conditions; the servers' time must be used efficiently and the capacity allocation must satisfy strict time requirements regarding completion time of route parts.

The first condition means that we do not use more capacity than needed. This becomes the objective function of our problem. We aim to minimise the total capacity at all stations. Note here that the capacity at each station is different, meaning that we cannot move capacity from one station to the other. This is due to the fact that we assume that a server at a station requires a specific skill in order to serve there.

The second condition gives the constraints for the optimisation problem. For each requirement and customer class, we have a given completion time which must not be exceeded. The completion time requirements are very strict; all customers must complete certain phases of their route within a certain amount of time. However, since we are dealing with uncertainty in interarrival and service times this would result in a lot of capacity for the stations. Hence, for modelling purposes, we make the requirements slightly less strict by introducing a threshold, which tells us what fraction of the customers at least must be seen within the completion time requirements. We assume that the network occupation is high, such that the waiting times of customers are of much larger order compared to the service times. Therefore, we look at the sum of waiting times of a route part for determining whether the requirements are satisfied.

In order to formulate this mathematically, we first need to introduce some notation. Let $\mathcal{N}$ denote the set of completion time requirements. Then requirement $m \in \mathcal{N}$ and customer class $k \in \{1, 2, \ldots, K\}$ must be finished within an amount of time $\tau_m^{(k)}$, which we refer to as the target time. The threshold is denoted by $x_m^{(k)}$, which states that at most $x_m^{(k)} \cdot 100\%$ of the customers of class k may exceed the completion time requirement m. Lastly, the random variable which denotes the sum of the waiting times of a customer of class k over the stations in requirement m is given by $\mathcal{W}_m^{(k)}$. The optimisation problem is formulated as follows:

$$\begin{aligned}
\min \quad & \sum_{i=1}^{J} \beta_i && && \text{(3.2a)}\\
\text{s.t.} \quad & \mathbb{P}\left(\mathcal{W}_m^{(k)} \geq \tau_m^{(k)}\right) \leq x_m^{(k)}, && \forall m \in \mathcal{N}, k \in \{1, 2, \ldots, K\}, && \text{(3.2b)}\\
& \beta_i \geq 0, && \forall i \in \{1, 2, \ldots, J\}. && \text{(3.2c)}
\end{aligned}$$

The random variable of the waiting time over a requirement is the sum of the waiting times at the stations included in the route part of the requirement. For complex queueing networks, one can approximate the mean and variance of this variable, but the cumulative distribution function is less straightforward to approximate. Also, the approximation formulas depend on the coefficient of variation of the interarrival and service times, where the first is found via a system of equations. Hence, the left side of the constraints are not necessarily convex functions with respect to $\beta_1, \beta_2, \ldots, \beta_J$. Therefore, we are not sure whether this programme converges to the optimal solution with the standard optimisation techniques. Also, non-convex optimisation problems can have multiple local optima, but we need to find the global optimum. We hypothesise that for the complex queueing network considered here, the resulting programme is non-convex, due to the complex relation between the coefficients of variation of the interarrival and service times. Therefore, we design an iterative method to solve this problem. The problems addressed here are discussed while describing the procedure.

3.3 Iterative method to determine the capacity

In order to determine the capacity needed to satisfy the demand and the completion time requirements, we design an iterative method. This method combines the Queueing Network Analyser (QNA) and Discrete Event Simulation. We use QNA to find a capacity allocation fast and afterwards we check the requirements for this capacity allocation with the simulation. Therefore, we distinguish a QNA phase and a simulation phase. During the QNA phase we fix a capacity allocation and analyse the system using QNA. Afterwards we check the completion time requirements and update the capacity if the requirements are not yet satisfied. Since QNA is an approximation method and we deal with strict requirements regarding completion time, we also include a simulation phase in the iterative method to make sure the capacity allocation satisfies the requirements. If the requirements are not satisfied according to the simulation, we move back to the QNA phase. A general overview of the iterative method can be found in Figure 3.1. The parts of the QNA phase are highlighted in orange and the parts of the simulation are highlighted in blue.

We distinguish the following six phases of the iterative method, corresponding to the blocks in Figure 3.1: initialisation (Section 3.3.1), performance analysis using QNA (Section 3.3.2), stopping criterion of QNA (Section 3.3.3), update rule (Section 3.3.4), discrete event simulation (Section 3.3.5) and stopping criterion of the iterative method (Section 3.3.6). In Section 3.3.7 we summarise the steps in an algorithm written in pseudo-code. The performance analysis, update rule and stopping criterion of QNA need to be carried out for every iteration and the discrete event simulation and stopping criterion for the simulation for some of the iterations. However, we omit a counter for the number of iterations passed in the notation such that the variables are not overloaded with sub- and superscripts.

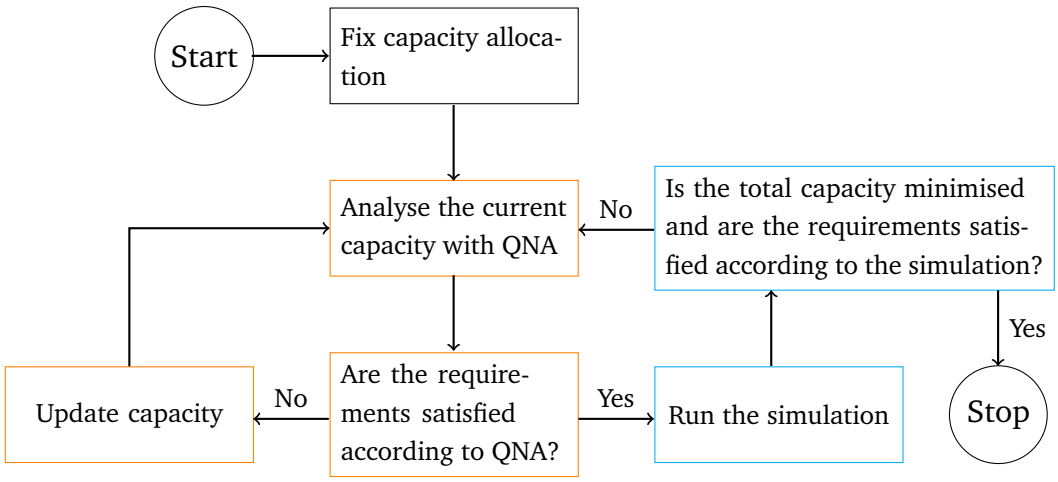

Figure 3.1: Overview of the iterative method.

3.3.1 Initialisation

We start the iterative method with the initialisation. In order to analyse the network, we combine the different types into one single type and initialise the capacity. We merge the different types by aggregating the arrival and service processes. For the arrival rates to the stations this is done by summing up the arrival rates of all types:

$$\lambda_j = \sum_{s=1}^{S} \lambda_j^{(s)}, \qquad \text{for } j \in \{1, 2, \ldots, J\}, \tag{3.3}$$

where the type specific arrival rates follow from Equation (3.1). Besides this, we should also aggregate the original service time. This is done by looking at the fraction of arrivals of a certain type. This yields the following:

$$\mathbb{E}[G_j] = \frac{1}{\lambda_j} \sum_{s=1}^{S} \sum_{v=1}^{L(s)} \gamma_0^{(s)} \mathbb{E}[G_j^{(s,v)}] \mathbb{1}\{r(s,v) = j\}, \qquad \text{for } j \in \{1, 2, \ldots, J\}. \tag{3.4}$$

From the original service times we can obtain the adjusted service time by dividing by β_j:

$$\mathbb{E}[B_j] = \frac{\mathbb{E}[G_j]}{\beta_j}, \qquad \text{for } j \in \{1, 2, \ldots, J\}.$$

The next step is to initialise this capacity $(\beta_1, \beta_2, \ldots, \beta_J)$. Since the mathematical programme we use is assumed to be non-convex, the objective function could have multiple local optima. Which optimum is found by the iterative method depends on the initialisation. In order to avoid ending up in a local minimum which is not the global optimum, we use a method also known as repeated local search or random restart [33]. Here we randomly generate M initial states $(\beta_1, \beta_2, \ldots, \beta_J)$ and these allocations are the starting points for the iterative method. Hence, the iterative method runs multiple times and

the capacity allocation with the smallest sum of capacities is then chosen as the optimal capacity allocation.

We randomly generate the initial capacities as follows. We have two conditions that the randomly generated initialisation state must satisfy: the capacity must be allocated such that the queueing system of Section 3.1 is stable and the allocation must not yet satisfy all completion time requirements. In order for a queueing system to be stable, the utilisation per server should be less than one. The utilisation at station j is given by:

$$\rho_j = \lambda_j \mathbb{E}[B_j] = \frac{\lambda_j \mathbb{E}[G_j]}{\beta_j}, \qquad \text{for } j \in \{1, 2, \ldots, J\}. \tag{3.5}$$

If we then set ρ_j to be less than one we obtain the lower bound for all stations:

$$\beta_j > \lambda_j \mathbb{E}[G_j], \qquad \text{for } j \in \{1, 2, \ldots, J\}.$$

We choose $\beta_{j,\text{stab}}$ for $j \in \{1, 2, \ldots, J\}$ such that this condition is just satisfied. From this state $\hat{\beta}_{\text{stab}}$, we generate an initial capacity by adding a number $\hat{\delta}$ to one of the stations' capacities h times in order to obtain $\hat{\beta}$. To which station this $\hat{\delta}$ is added is decided randomly following a uniform distribution. In total we generate M initial capacities in this way. By only taking a limited amount of h steps from the capacity that just satisfied the stability condition, we try to avoid starting the iterative method with an initialisation that already satisfies all completion time requirements.

3.3.2 Performance analysis using QNA

During the performance analysis phase we analyse the queueing network with the predetermined capacity. We are looking for the performance measure waiting time. In order to do so, we use the Queueing Network Analyser (QNA) described in [9]. This is a parametric decomposition method, which only requires the first and second moment of the arrival and service processes. Hence, it makes approximate analysis of more extensive queueing networks possible, where we have general arrival and service time distributions. Moreover, QNA can analyse a queueing network with multiple types of customers. The method described below is specifically for a network of single server queues.

Input

We start with the input for QNA. The first step is to aggregate the expected arrival rates and service times, which we already did in Equations (3.3) and (3.4). Besides this, we also require the aggregated internal flow rate, i.e. the flow from station i to station j:

$$\lambda_{ij} = \sum_{s=1}^{S} \sum_{v=1}^{L(s)-1} \gamma_0^{(s)} \mathbb{1}\{r(s,v) = i, r(s,v+1) = j\}, \qquad \text{for } i, j \in \{1, 2, \ldots, J\}.$$

The aggregated external arrival rate is the sum of external arrival rates per type:

$$\lambda_{0j} = \sum_{s=1}^{S} \gamma_0^{(s)} \mathbb{1}\{r(s,1) = j\}, \qquad \text{for } j \in \{1, 2, \ldots, J\}.$$

We also need the aggregated departure rate for each station, given by:

$$\lambda_{i0} = \sum_{s=1}^{S} \gamma_0^{(s)} \mathbb{1}\{r(s, L(s)) = i\}, \qquad \text{for } i \in \{1, 2, \ldots, J\}.$$

With these flow rates we can calculate the routing matrix Q, which contains the proportion of customers travelling from station i to station j:

$$Q_{ij} = \frac{\lambda_{ij}}{\sum_{l=0}^{M} \lambda_{il}}, \qquad \text{for } i, j \in \{1, 2, \ldots, J\}.$$

Furthermore, we need the squared coefficient of variation (SCV) of the external arrival process, which is given by:

$$C_{0j}^2 = (1 - w_j) + w_j \left(\sum_{s=1}^{S} (C_0^{(s)})^2 \frac{\gamma_0^{(s)} \mathbb{1}\{r(s,1) = j\}}{\lambda_{0j}} \right), \qquad \text{for } j \in \{1, 2, \ldots, J\},$$

with $C_0^{(s)}$ the SCV of the external arrival process of type s and

$$w_j = (1 + 4(1 - \rho_j)^2 (\hat{u}_j - 1)^2)^{-1},$$

with ρ_j the utilisation at station j calculated as in (3.5) and

$$\hat{u}_j = \left[\sum_{s=1}^{S} \left(\frac{\gamma_0^{(s)} \mathbb{1}\{r(s,1) = j\}}{\lambda_{0j}} \right)^2 \right]^{-1}.$$

Lastly, we need the aggregated SCV of the service time. This can be calculated by using the following formula:

$$C_{bj}^2 = \frac{\sum_{s=1}^{S} \sum_{v=1}^{L(s)} \gamma_0^{(s)} \mathbb{E}[B_{r(s,v)}^{(s,v)}]^2 ((C_{br(s,v)}^{(s,v)})^2 + 1) \mathbb{1}\{r(s,v) = j\}}{\mathbb{E}[B_j]^2 \lambda_j} - 1, \qquad \text{for } j \in \{1, 2, \ldots, J\}.$$

Pre-processing the queueing network: eliminating immediate feedback
In the queueing network, it could happen that a customer returns to the same station multiple times in a row. In [9] and [12] it is described that one could convert the network such that this immediate feedback is eliminated in order to obtain a better approximation. QNA obtains this elimination by letting the customer receive his/her total service, i.e. the service times of all the subsequent services together, at once when this customer can be seen by the server. In this way, the customer is put in front of the waiting line after finishing one service, instead of at the back. A customer routes from station j back to station j with probability Q_{jj}. Hence the number of visits to station j is geometrically distributed with success probability of leaving station j of $1 - Q_{jj}$. Hence

we can find the expected total service time by analysing the sum of service times with a geometrically distributed number of terms. This yields:

$$\begin{aligned}\mathbb{E}[\hat{B}_j] &= \frac{\mathbb{E}[B_j]}{1-Q_{jj}},\\ \hat{C}_{bj}^2 &= Q_{jj} + (1-Q_{jj})C_{bj}^2,\\ \hat{Q}_{ij} &= \begin{cases} 0, & \text{for } i=j,\\ \frac{Q_{ij}}{1-Q_{ii}}, & \text{for } i\neq j.\end{cases}\end{aligned}$$

Since we altered the routing matrix Q, we should recalculate the arrival rates at every station, using the following traffic equations:

$$\hat{\lambda}_j = \lambda_{0j} + \sum_{i=1}^{M} \hat{\lambda}_i \hat{Q}_{ij},$$

from which follows that $\hat{\lambda}_j = (1-Q_{jj})\lambda_j$. Hence, we do not need to update the utilisation, since $\rho_j = \hat{\lambda}_j \mathbb{E}[\hat{B}_j] = \lambda_j \mathbb{E}[B_j] = \rho_j$.

After calculating the expectation and variance of the waiting time, we compensate for the changes made to eliminate the immediate feedback to obtain the performance measures for the original system. This will be discussed at the end of this section.

Waiting time

Now we can start with the approximation of the performance measures. The first step is to obtain the variability of the internal arrival process, by calculating the SCV of the arrival process. These approximations are based on renewal theory. This yields the following system of equations:

$$C_{aj}^2 = a_j + \sum_{i=1}^{J} C_{ai}^2 b_{ij}, \qquad \text{for } j \in \{1,2,\ldots,J\},$$

where the constants a_j and b_{ij} are derived from the input data:

$$\begin{aligned}a_j &= 1 + w_j\left(P_{0j}C_{0j}^2 - 1 + \sum_{i=1}^{M} P_{ij}(1-\hat{Q}_{ij} + \hat{Q}_{ij}\rho_i^2 x_i)\right),\\ b_{ij} &= w_j P_{ij}\hat{Q}_{ij}(1-\rho_i^2),\\ x_i &= \max\{\hat{C}_{bi}^2, 0.2\},\end{aligned}$$

and

$$\begin{aligned}P_{ij} &= \frac{\hat{\lambda}_i \hat{Q}_{ij}}{\hat{\lambda}_j},\\ w_j &= (1+4(1-\rho_j)^2(u_j-1)^2)^{-1},\\ u_j &= \left[\sum_{i=0}^{M} P_{ij}^2\right]^{-1}.\end{aligned}$$

By doing these calculations, we can describe each station in the network with four parameters: the first and second moment of the interarrival time and the first and second moment of the service time. We decomposed the queueing network into individual stations which are analysed separately. An approximation of the expected waiting time can now be calculated using:

$$\mathbb{E}[\hat{W}_j] = \frac{C_{aj}^2 + \hat{C}_{bj}^2}{2} \frac{\rho_j}{1-\rho_j} \mathbb{E}[\hat{B}_j] g_j, \qquad \text{for } j \in \{1, 2, \ldots, J\}, \tag{3.6}$$

where we define g_j as:

$$g_j = \begin{cases} \exp\{-\frac{2(1-\rho_j)}{3\rho_j} \frac{(1-C_{aj}^2)^2}{C_{aj}^2 + \hat{C}_{bj}^2}\}, & \text{if } C_{aj}^2 < 1, \\ 1, & \text{if } C_{aj}^2 \geq 1. \end{cases}$$

For $C_{aj}^2 < 1$ this expression reduces to the Kraemer and Langenbach-Belz approximation [34], while for $C_{aj}^2 \geq 1$ this yields the Kingman approximation [35].

In order to approximate the variance of the waiting time, we first look at the probability of delay, denoted by $\Pi_{\hat{W}}$. This is given by:

$$\Pi_{\hat{W}_j} = \mathbb{P}(\hat{W}_j > 0) = \rho_j + (C_{aj}^2 - 1)\rho_j(1-\rho_j)h_j, \qquad \text{for } j \in \{1, 2, \ldots, J\},$$

with h_j depending on ρ_j, C_{aj}^2 and $\hat{C}_{bj}^2$:

$$h_j = \begin{cases} \frac{1 + C_{aj}^2 + \rho_j \hat{C}_{bj}^2}{1 + \rho_j(\hat{C}_{bj}^2 - 1) + \rho_j^2(4C_{aj}^2 + \hat{C}_{bj}^2)}, & \text{for } C_{aj}^2 \leq 1, \\ \frac{4\rho_j}{C_{aj}^2 + \rho_j^2(4C_{aj}^2 + \hat{C}_{bj}^2)}, & \text{for } C_{aj}^2 \geq 1. \end{cases}$$

For an $M/G/1$ queue, this reduces to $\Pi_{\hat{W}_j} = \rho_j$, the fraction of time a server is working. Besides this, we also need the squared coefficient of variation of the conditional delay variable, i.e., the waiting time given that the server is busy. This random variable is denoted by $\hat{D}_j$ for station $j \in \{1, 2, \ldots, J\}$ and its SCV can be calculated with:

$$C_{Dj}^2 = 2\rho_j - 1 + \frac{4(1-\rho_j)d_{bj}^3}{3(\hat{C}_{bj}^2 + 1)^2}, \qquad \text{for } j \in \{1, 2, \ldots, J\}$$

with

$$d_{bj}^3 = \frac{\mathbb{E}[\hat{B}_j^3]}{\mathbb{E}[\hat{B}_j]^3}.$$

If the third moment of the service time is unavailable, one can approximate it based on the Erlang or hyperexponential service distribution. This yields the following approximation:

$$d_{bj}^3 = \begin{cases} (2\hat{C}_{bj}^2 + 1)(\hat{C}_{bj}^2 + 1), & \text{if } \hat{C}_{bj}^2 < 1, \\ 3\hat{C}_{bj}^2(1 + \hat{C}_{bj}^2), & \text{if } \hat{C}_{bj}^2 \geq 1. \end{cases}$$

Now the variance of the waiting time is given by:

$$\mathrm{Var}(\hat{W}_j) = \mathbb{E}[\hat{W}_j]^2 C^2_{\hat{W}_j}, \qquad \text{for } j \in \{1, 2, \ldots, J\},$$

where

$$C^2_{\hat{W}_j} = \frac{C^2_{Dj} + 1 - \Pi_{\hat{W}_j}}{\Pi_{\hat{W}_j}}.$$

Post-processing the queueing network after elimination of immediate feedback
After calculating the performance measures, we should adjust these to take into account the elimination of immediate feedback. The post-processing expressions are based on heavy-traffic theorems for networks of queues. We are interested in the waiting time of an original visit to this station, which we obtain by multiplying the expected waiting time obtained from QNA by the factor $1 - Q_{jj}$:

$$\mathbb{E}[W_j] = (1 - Q_{jj})\mathbb{E}[\hat{W}_j], \qquad \text{for } j \in \{1, 2, \ldots, J\}.$$

The parameters, like arrival rate and service time, can be transformed to their original values as follows:

$$\begin{aligned} \lambda_j &= \frac{\hat{\lambda}_j}{1 - Q_{jj}}, \\ \mathbb{E}[B_j] &= (1 - Q_{jj})\mathbb{E}[\hat{B}_j], \\ C^2_{bj} &= \frac{\hat{C}^2_{bj} - Q_{jj}}{1 - Q_{jj}}. \end{aligned}$$

Furthermore, we want to know the variance of the waiting time. In order to find an expression, we go through several steps. We start with a pre-approximation for the variance of the waiting time of an individual visit to station j. We use here that a customer arriving at the queue needs to wait on average all service times of the customers before him. The number of customers in the queue is given by the random variable N_j. This results in:

$$\begin{aligned} \mathrm{Var}(W'_j) &= \mathrm{Var}\left(\sum_{\ell=1}^{N_j} B^\ell_j\right) \\ &= \mathbb{E}[N_j]\mathrm{Var}(B_j) + \mathbb{E}[B_j]^2\mathrm{Var}(N_j), \end{aligned}$$

where B^ℓ_j represents the service time of the ℓ^{th} customer in the system with immediate feedback, but since the aggregated service times of all customers are identically distributed we change this to B_j. The expected number of customers at station j can be determined using Little's law, where we take the expected waiting time given by QNA before post-processing it:

$$\mathbb{E}[N_j] = \hat{\lambda}_j(\mathbb{E}[\hat{B}_j] + \mathbb{E}[\hat{W}_j]).$$

Note that the number of customers at a station does not change due to the elimination of immediate feedback. For an $M/G/1$ queue, the variance of the number of customers is as follows:

$$\mathrm{Var}(N_j)_{M/G/1} = \hat{\lambda}_j \mathbb{E}[\hat{W}_j] + \rho_j + \rho_j^2 \hat{C}_{bj}^2 + \hat{\lambda}_j^2 \mathrm{Var}(\hat{W}_j).$$

For a general $G/G/1$ queue, we need to make some modifications and use the results obtained for $M/G/1$ in the factor:

$$c_{N_j}^2 = \frac{\mathrm{Var}(N_j)_{M/G/1} \cdot Z_1}{Z_2},$$

with

$$Z_1 = \frac{1 - \rho_j + \Pi_{\hat{W}_j}}{\max\{1 - \Pi_{\hat{W}_j} + \rho_j, 0.000001\}} \qquad \text{and} \qquad Z_2 = \max\{(\rho_j + \hat{\lambda}_j \mathbb{E}[\hat{W}_j])^2, 0.000001\}.$$

The variance for an $G/G/1$ queue is then given by:

$$\mathrm{Var}(N_j) = \mathbb{E}[\hat{W}_j]^2 c_{N_j}^2.$$

Next we calculate the SCV of the pre-approximation of the sojourn time of an individual visit to station j, by adding the waiting time and service time variable:

$$C_{T_j'}^2 = \frac{\mathrm{Var}(W_j') + \mathrm{Var}(B_j)}{(\mathbb{E}[W_j] + \mathbb{E}[B_j])^2}.$$

Through the above expression, we find an approximation for the total sojourn time of all subsequent visits to station j. Suppose Y_j is the random variable for the number of subsequent visits to station j, which is geometrically distributed with success probability $1 - Q_{jj}$. By heavy traffic limit theory, the total sojourn time $\hat{T}_j'$ is approximated by $T_j' Y_j$, the number of visits times the duration of a single visit. This results in:

$$\begin{aligned} C_{\hat{T}_j'}^2 &= \frac{\mathbb{E}[(T_j')^2]\mathbb{E}[Y_j^2]}{\mathbb{E}[T_j']^2 \mathbb{E}[Y_j]^2} - 1 \\ &= (C_{T_j'}^2 + 1)(C_{Y_j}^2 + 1) - 1 \\ &= C_{T_j'}^2 (1 + Q_{jj}) + Q_{jj}. \end{aligned}$$

Next, we obtain the variance for the total sojourn time by using the definition of the SCV:

$$\begin{aligned} \mathrm{Var}(\hat{T}_j') &= C_{\hat{T}_j'}^2 \mathbb{E}[\hat{T}_j']^2 \\ &= C_{\hat{T}_j'}^2 (\mathbb{E}[Y_i]\mathbb{E}[T_j'])^2 \\ &= C_{\hat{T}_j'}^2 \left(\frac{1}{1 - Q_{jj}} (\mathbb{E}[W_j] + \mathbb{E}[B_j]) \right)^2 \\ &= C_{\hat{T}_j'}^2 (\mathbb{E}[\hat{W}_j] + \mathbb{E}[\hat{B}_j])^2. \end{aligned}$$

With the last expression we are able to obtain the variance of the waiting time in a network with immediate feedback. We divide the variance of the total sojourn time by the mean number of subsequent visits to station j and subtract the variance of the service time to obtain the following approximation:

$$\mathrm{Var}(W_j) = (1 - Q_{jj})\mathrm{Var}(\hat{T}_j') - \mathrm{Var}(B_j).$$

3.3.3 Stopping criterion of QNA

We require enough capacity such that the strict completion time requirements can be satisfied. However, we also want that the time of a servers is used efficiently, since they could have multiple tasks to complete and are not employable in the network the whole week. Hence, we need to design a realistic stopping criterion for this problem.

For a feasible solution of the capacity, we must have that Equation (3.2b) is satisfied for all requirements and customer classes. However, determining the left side of these inequalities exactly is not possible for this queueing network. Hence, we use Chebyshev's inequality [36] to determine whether the completion time requirements are satisfied and approximate inequality (3.2b). Now we only need to determine the expectation and variance of the sum of the waiting times over the stations included in the requirement.

Some of the requirements include multiple stations and hence we need some notation to specify the stations included in a requirement. For each type $s \in \{1, 2, \ldots, S\}$, we need to define the phases of the route which are contained in a requirement $m \in \mathcal{N}$. We define the stage $b_m^{(s)}$ as the begin stage and $e_m^{(s)}$ as the end stage of requirement m for type s. The station of the beginning of the requirement and ending of the requirement will then be given by $r(s, b_m^{(s)})$ and $r(s, e_m^{(s)})$, respectively. When a type s does not take part in the requirement m, we let both $b_m^{(s)}$ and $e_m^{(s)}$ be zero.

With this notation, we can define the random variable for the waiting time over all stations contained in a requirement for a certain customer class. This is done by taking the weighted sum over all types and its phases contained in the requirement:

$$\mathcal{W}_m^{(k)} = \sum_{s \in S_k} p_{ks} \mathbb{1}\{b_m^{(s)} \neq 0, e_m^{(s)} \neq 0\} \sum_{v=b_m^{(s)}}^{e_m^{(s)}} W_{r(s,v)}, \qquad \text{for } m \in \mathcal{N},\ k \in \{1, 2, \ldots, K\},$$

where we take into account the probability that a customer is of a certain type. Note that the waiting time at a station is the same for all customer types.

With the information obtained from the performance analysis step, we can determine the expectation of this random variable by using the linearity of the expectation.

$$\mathbb{E}[\mathcal{W}_m^{(k)}] = \sum_{s \in S_k} p_{ks} \mathbb{1}\{b_m^{(s)} \neq 0, e_m^{(s)} \neq 0\} \sum_{v=b_m^{(s)}}^{e_m^{(s)}} \mathbb{E}[W_{r(s,v)}]. \tag{3.7}$$

In order to do the same for the variance of a weighted sum of random variables, we need the fact that all variables are independent. By using QNA, we decompose the network of queues into individual parts and analyse each queue separately, as if the performance measures are independent. Hence, we assume here that the waiting time random variables of each station are independent, such that we can sum the variances of the individual stations to obtain the total variance:

$$\mathrm{Var}(\mathcal{W}_m^{(k)}) = \sum_{s \in S_k} p_{ks} \mathbb{1}\{b_m^{(s)} \neq 0, e_m^{(s)} \neq 0\} \sum_{v=b_m^{(s)}}^{e_m^{(s)}} \mathrm{Var}(W_{r(s,v)}). \tag{3.8}$$

Note, however, that in reality there is a correlation between the waiting times at each station. Not much is known in literature about this correlation and hence we assume for practical purposes that the waiting times are independent.

With these two performance measures we need to determine whether the requirements are satisfied under the current resource allocation. For this we use Chebyshev's inequality, which states:

$$\mathbb{P}\left(\left| \mathcal{W}_m^{(k)} - \mathbb{E}[\mathcal{W}_m^{(k)}] \right| \geq \sqrt{\frac{\mathrm{Var}(\mathcal{W}_m^{(k)})}{x_m^{(k)}}} \right) \leq x_m^{(k)}. \tag{3.9}$$

Through this equation we know that values of the random variable $\mathcal{W}_m^{(k)}$ lie outside the interval:

$$\left(\mathbb{E}[\mathcal{W}_m^{(k)}] - \sqrt{\frac{\mathrm{Var}(\mathcal{W}_m^{(k)})}{x_m^{(k)}}}, \mathbb{E}[\mathcal{W}_m^{(k)}] + \sqrt{\frac{\mathrm{Var}(\mathcal{W}_m^{(k)})}{x_m^{(k)}}} \right),$$

with probability less than $x_m^{(k)}$. Hence $x_m^{(k)}$ part of the waiting times does not exceed the upper bound of this interval, so for our stopping criterion we must have that:

$$\mathbb{E}[\mathcal{W}_m^{(k)}] + \sqrt{\frac{\mathrm{Var}(\mathcal{W}_m^{(k)})}{x_m^{(k)}}} \leq \tau_m^{(k)}, \qquad \text{for all } m \in \mathcal{N}, k \in \{1, 2, \ldots, K\}. \tag{3.10}$$

When the current capacity allocation is not able to satisfy the stopping criterion, we need to update the capacity and return to the performance analysis for this new allocation. When the capacity does satisfy the stopping criterion, we have found the capacity needed in the system and we stop the QNA part of the iterative method. However, since QNA is an approximation method and we make the assumption that all stations can be analysed separately, we want to be sure that the completion time requirements are satisfied with the current capacity. In this case the next step of iterative method would be to use discrete event simulation to check the waiting times calculated by the iterative method and also find the percentage of customers which cannot be seen within the requirements. Hence, we have two options, which are described in the next sections.

3.3.4 Update rule

After the analysis step is performed, one outcome would be that the current capacity allocation does not yet satisfy the completion time requirements. In this case, we decide at which station the capacity needs to be updated and move back to the analysis step. Hence, we aim to design a smart update rule. However, there is not only one correct assignment of capacity to the stations. Since some of the requirements include multiple stations, these could be satisfied by increasing any of those stations. Therefore, we need to find a method to decide which station is updated.

We use a neighbourhood search procedure to find the station to be updated, meaning that we look at the neighbourhood of the current state and choose a new state within this neighbourhood. The current state is the capacity allocation $\hat{\beta}$. The neighbourhood of state $\hat{\beta}$ is given by the set $\Omega(\hat{\beta})$, which contains all neighbours of $\hat{\beta}$. For this problem, we take the parameter δ as the amount with which we could update the capacity by adding this amount to the capacity of one station. This means that we have as set of neighbours:

$$\Omega(\hat{\beta}) = \left\{ \hat{\beta} + \delta \cdot \hat{\mathbf{e}}_i : i \in \{1, 2, \ldots, J\} \right\},$$

where $\hat{\mathbf{e}}_i$ stands for the unit vector with a one at the i^{th} place in the vector, corresponding to station i, and a zero everywhere else. Note that this implies that we only update one station each iteration. Also, this step size δ could be different from the step size $\hat{\delta}$ used for the initialisation.

Neighbourhood search now prescribes that each iteration we choose one state in $\Omega(\hat{\beta})$ as our update. We still need to determine which of the states in the neighbourhood of $\hat{\beta}$ is the best choice as our next state when minimising the total capacity used. The constraints (3.10) used for the stopping criterion depend on the SCV of the arrival process and the service time, which change at a station when the capacity is changed. Hence, these constraints are not necessarily convex.

In literature, some propose the assumption that the SCV of the arrival process and the SCV of the service process are independent of changes in the server's speed [10][23]. The approximation function (3.6) for the expected waiting time becomes convex with respect to $\beta_1, \beta_2, \ldots, \beta_J$ under this assumption (for a proof see [23]). Also, for an increase in number of customer types, the influence of the server speed on the SCV seems to decrease, as the departure process of a customer class then approaches its arrival process [23]. However, this only holds for specific cases. Therefore, we assume during this step of the iterative method that the SCVs are constant, such that we can use convex optimisation techniques, but during the analysis step, we do update the SCV of the arrival process and the SCV of the service process at each station. By using multiple states as starting point of the iteration, as described in Section 3.3.1, we try to avoid ending in a local minimum.

Now we can decide which of the states in $\Omega(\hat{\beta})$ is going to be our next state. We need to satisfy a number of constraints while keeping the sum of capacity minimal. Hence, we are looking for an update direction which results in the biggest step towards satisfying

the requirements. We use Equation (3.10) as an approximation of the constraints (3.2b) in the mathematical programme, since those cannot be determined exactly. Hence, we look at Equation (3.10) to determine the update rule. The left side of this inequality depends on both the expectation and the variance of the waiting time. For each norm m and customer class k we have a constraint. When we assume the SCVs of the arrival and service process are constant, only the expectation is convex with respect to the capacity. The variance has a complicated dependency on the capacity. Hence, we update the capacity for the station that results in the biggest decrease in the expectation of the waiting time.

In order to substantiate this choice, we have the following reasoning. When the expectation of the waiting time decreases by increasing the capacity at one station, this results in a better distribution of the load over the stations in the network. If the load is lower in a network, then the difference in waiting time between customers becomes smaller, since the probability on high waiting time decreases. Therefore, the variance of the waiting time also decreases. So even though we only look at the influence of the capacity change on the expectation, we still may assume that the overall variance decreases as well. This does not yield that by only looking at the expectation we know for sure that we obtain the biggest decrease in the constraints of Equation (3.10), but it gives an indication that we made a good decision.

For the expected waiting time we can determine the influence of updating a station j by taking the gradient with respect to the server's speed β_j. This results in the following function $U_{m,j}^{(k)}(\beta_1, \ldots, \beta_J)$:

$$U_{m,j}^{(k)}(\beta_1, \ldots, \beta_J) = \frac{\partial \mathbb{E}[\mathcal{W}_m^{(k)}]}{\partial \beta_j} = \sum_{s \in S_k} p_{ks} \sum_{v=b_m^{(s)}}^{e_m^{(s)}} \frac{\partial \mathbb{E}[W_{r(s,v)}]}{\partial \beta_j}.$$

Note that each element in the sum only depends on the server speed of the station it corresponds to. This causes that only the elements corresponding to the station of which we take the partial derivative remain. We choose to not state the partial derivatives of $\mathbb{E}[\mathcal{W}_m^{(k)}]$ explicitly in this report, but it can be easily derived from Equation (3.6).

Now that we have the partial derivative for one requirement and station, we need to determine what the influence of updating this station is on all requirements. When doing this, we only need to consider the requirement and customer class combination that do not yet satisfy the completion time requirements. Let Ψ denote the set containing these combinations (m, k) of requirement m and customer class k for which the target time is not yet reached. We then have

$$U_j(\beta_1, \ldots, \beta_J) = \sum_{(m,k) \in \Psi} U_{m,j}^{(k)}(\beta_1, \ldots, \beta_M), \qquad \text{for } j \in \{1, 2, \ldots, J\}.$$

If we fill in the current capacity allocation, we can find the station to update, denoted by j^*, by looking at the station with the smallest sum of partial derivatives, and hence

the largest total decrease in the expected waiting time:

$$j^* = \arg \min_{j \in \{1,2,\ldots,J\}} U_j(\beta_1, \ldots, \beta_J).$$

This means that we choose neighbouring state $\hat{\beta} + \delta \cdot \hat{\mathbf{e}}_{j^*}$ as our next capacity to analyse using QNA.

3.3.5 Discrete event simulation

To check the results of an approximation method for queueing networks often simulation is used. For this network we develop a discrete event simulation, which models the arrival streams of customers and their routes through the network. The basic idea is that we jump in time from one event to next event, which is either an arrival or a departure of a customer at one of the stations. The times of events are randomly generated following the probability distributions of the interarrival and service rates of customers. If a server is already busy upon arrival of a customer, the customer is placed in a queue. The information we collect from an individual customer, like waiting time at stations, is called an observation.

The simulation starts with a warm-up period, such that the network is in a stable state when we start to gather observations regarding the performance measures we are interested in. For the warm-up period, we look at the waiting time at each station and wait until enough customers have departed such that the waiting time has stabilised at each station. We determine the stabilisation using a technique similar to the moving average method with window size q proposed by Welch [37]. Of a total of n_i observations at station i, we look at the last q observation and compare the average of those to the mean of the observations in the z periods of length q before the last observation. In this way we can determine whether the waiting time at a station has changed a lot during the last departures. If not, the network has stabilised and we end the warm-up period. This can be formulated in the following stopping criterion:

$$\left| \frac{\frac{1}{q} \sum_{\ell=n_i-q}^{n_i} W_i(\ell)}{\frac{1}{z} \sum_{j=1}^{z} \frac{1}{q} \sum_{\ell=n_i-q-j}^{n_i-j} W_i(\ell)} - 1 \right| \leq \omega, \qquad \text{for all } i \in \{1, 2, \ldots, J\},$$

where ω is the parameter chosen for this stopping criterion and must be close to zero. After every departure at some station in the network, we calculate this stopping criterion for this station. In this way, n_i is incremented by one for each calculation. After the warm-up period, we start to gather the observations to check the completion time requirements.

When enough arrivals are generated and performance measures like waiting time and number of customers in the system are stored for every customer, we are able to draw statistically meaningful conclusions from the data gathered by the simulation. In this case we are mainly interested in the number of customers who are not able to finish their route parts within the completion time requirements, so the value of $\mathbb{P}(\mathcal{W}_m^{(k)} \geq \tau_m^{(k)})$. We

start by defining a new observation $X_m^{(k)}(\ell)$, linked to the observation of the time spent on the completion time requirement m from a customer of class k, which is denoted by $\mathcal{W}_m^{(k)}(\ell)$. This observation takes the following value:

$$X_m^{(k)}(\ell) = \begin{cases} 1, & \text{if } \mathcal{W}_m^{(k)}(\ell) \geq \tau_m^{(k)}, \\ 0, & \text{otherwise}, \end{cases} \qquad \text{for } m \in \mathcal{N}, k \in \{1, 2, \ldots, K\},$$

where $\tau_m^{(k)}$ is the target time of requirement m and customer class k. We use the Batch Means Method [38] to analyse this performance measure.

For the Batch Means Method only one long simulation run is needed with one warm-up period, instead of multiple shorter runs each with its own warm-up period. The observations collected during the long simulation run are divided into r batches, such that each batch is approximately independent of the others. Often, a number between 25 and 50 is chosen as r. First we determine the batch size, if we have a total of $n_m^{(k)}$ observations. Since the number of observations does not need to be divisible by the number of batches, we have as batch size of batch y:

$$\nu_{m,y}^{(k)} = \begin{cases} \left\lfloor \frac{n_m^{(k)}}{r} \right\rfloor + 1, & \text{for } y = 1, 2, \ldots, F_m^{(k)} - 1, \\ \left\lfloor \frac{n_m^{(k)}}{r} \right\rfloor, & \text{for } y = F_m^{(k)}, F_m^{(k)} + 1, \ldots, r, \end{cases}$$

with

$$F_m^{(k)} = n_m^{(k)} - r \left\lfloor \frac{n_m^{(k)}}{r} \right\rfloor.$$

To calculate the mean of every batch, we sum over the elements in the batch and divide by the size of the batch. With these batches we are able to determine a confidence interval of the probability that the time spent on the completion time requirement is longer than the target time. For this we need the sample mean and variance of the batches. If the mean of batch y is denoted by $\overline{X}_{m,y}^{(k)}$, we get as sample mean:

$$\overline{X}_m^{(k)} = \frac{1}{r} \sum_{y=1}^{r} \overline{X}_{m,y}^{(k)}.$$

The sample variance can be calculated as follows:

$$(\overline{S}_m^{(k)})^2 = \frac{1}{r-1} \sum_{y=1}^{r} (\overline{X}_{m,y}^{(k)} - \overline{X}_m^{(k)})^2.$$

Now we determine a $(1-\alpha) \cdot 100\%$ confidence interval. For batches of a size large enough, we may assume that the batch means are approximately normally distributed. Hence we have as $(1-\alpha) \cdot 100\%$ confidence interval for $\mathbb{P}(\mathcal{W}_m^{(k)} \geq \tau_m^{(k)})$:

$$\left[\overline{X}_m^{(k)} - \frac{t_{r-1,1-\alpha/2} \sqrt{(\overline{S}_m^{(k)})^2}}{\sqrt{r}}, \overline{X}_m^{(k)} + \frac{t_{r-1,1-\alpha/2} \sqrt{(\overline{S}_m^{(k)})^2}}{\sqrt{r}} \right], \tag{3.11}$$

with $t_{r-1,1-\alpha/2}$ as the $(1-\alpha/2)$-percentile of the Student's t-distribution for $r-1$ degrees of freedom. This is the main performance measure we are interested in. We also determine the confidence interval of the mean sojourn time over a completion time requirement using the batch means method, using observations of $\mathcal{W}_m^{(k)}$.

3.3.6 Stopping criterion of the iterative method

When QNA has found a capacity allocation which satisfies the completion time requirements, we move to discrete event simulation and determine the performance measures found as described in the previous chapter. With these results we aim to decide whether QNA has found a reasonable capacity allocation or whether the results obtained there gave too big approximation errors and the capacity planning could be optimised more. In this section, we describe a stopping criterion for the simulation, and with that for the whole iterative method. We also describe the steps that need to be taken when the stopping criterion is not satisfied.

We run the simulation twice each simulation phase with different capacity allocations. Of course one of these capacity allocations is given by the QNA phase, which satisfies the completion time requirements according to the QNA stopping criterion. This allocation is called the current allocation and has variable $\hat{\beta}_{\text{cur}}$. The second capacity allocation for which we run the simulation is referred to as the previous allocation and is denoted by the variable $\hat{\beta}_{\text{prev}}$. This capacity is used to check whether we did not add too much capacity. We subtract the step size δ from the current capacity of each of the stations to obtain this allocation. If the station becomes unstable after subtracting δ, we set the capacity back to the minimum capacity for which $\rho < 1$. Hence,

$$\hat{\beta}_{i,\text{prev}} = \begin{cases} \hat{\beta}_{i,\text{cur}} - \delta, & \text{if } \hat{\beta}_{i,\text{cur}} - \delta > \hat{\beta}_{i,\text{stab}}, \\ \hat{\beta}_{i,\text{stab}}, & \text{otherwise}, \end{cases} \qquad \text{for } i \in \{1, 2, \ldots, J\}.$$

We look at the performance measures of both capacity allocations given by the simulation and compare the two.

With Equation (3.11), we are able to determine the probability that a customer does not meet the completion time requirements. The confidence interval tells us that with $(1-\alpha)\cdot 100\%$ confidence, we can say that the value of $\mathbb{P}(\mathcal{W}_m^{(k)} \geq \tau_m^{(k)})$ lies within the two bounds. This means that if we would look at a different set of n observations multiple times and determine the confidence interval in (3.11) for each, in $(1-\alpha)\cdot 100\%$ of the cases the value of $\mathbb{P}(\mathcal{W}_m^{(k)} \geq \tau_m^{(k)})$ will lie within the bounds. Note therefore that the $(1-\alpha)\cdot 100\%$ has nothing to do with the threshold $x_m^{(k)}$ we set earlier.

Now for the strict requirements, we must have that the upper bound of the confidence interval of $\mathbb{P}(\mathcal{W}_m^{(k)} \geq \tau_m^{(k)})$ must lie below the threshold $x_m^{(k)}$. If this is the case, we can say with $(1-\alpha)\cdot 100\%$ confidence that $\mathbb{P}(\mathcal{W}_m^{(k)} \geq \tau_m^{(k)})$ is smaller than the threshold. If $x_m^{(k)}$ lies within the confidence interval, $\mathbb{P}(\mathcal{W}_m^{(k)} \geq \tau_m^{(k)})$ could still be above or below the threshold. We always keep the $\alpha \cdot 100\%$ of the cases where $\mathbb{P}(\mathcal{W}_m^{(k)} \geq \tau_m^{(k)})$ is not

contained in the interval, but that is the consequence of analysing a complex queueing system which cannot be approached exactly. Now the completion time requirements are satisfied if:

$$\overline{X}_m^{(k)} + \frac{t_{r-1,1-\alpha/2}\sqrt{(\overline{S}_m^{(k)})^2}}{\sqrt{r}} \leq x_m^{(k)}, \qquad \text{for all } m \in \mathcal{N}, k \in \{1, 2, \ldots, K\}. \tag{3.12}$$

Since we run the simulation two times, with two different capacity allocations, we could have the following situations:

1. The current allocation satisfies the requirements and the previous does not.
2. The current and previous allocation both do not satisfy the requirements.
3. The current and previous allocation both satisfy the requirements.

In situation 1, we know we are close to an optimal allocation and QNA did a good job in approximating the performance measures. However, we do not know if the completion time requirements would still be satisfied for an allocation somewhere in between the current and previous allocation. Since we have a discrete step size δ for updating the capacity, it could be that we added too much capacity and could do with less. Therefore we decrease the step size, until we consider the step size sufficiently small. The step size is updated following:

$$\delta \leftarrow \delta/\eta. \tag{3.13}$$

Hence, we have the following rule for this situation:

When situation 1 occurs and the step size is still sufficiently large, move back to QNA. Decrease the update step size following Equation (3.13) *and start with the previous capacity allocation. If the update step size is smaller than δ_{min}, we stop the iterative method for this initial capacity and the optimal capacity allocation is the current allocation.*

The value of δ_{min} depends on the problem at hand and a suitable value should be carefully considered.

For the second situation, QNA has underestimated the time used for the requirements. We have not yet reached the capacity needed to satisfy the completion time requirements. Hence, we need to update the capacity again. Since using simulation for this would take too long if it needed to be run multiple time, we move back to QNA. However, since the current allocation already satisfied the completion time requirements, any capacity update will still do the same. Hence, we need to update our stopping criterion for QNA or rather the values used in this stopping criterion for the requirement-class combinations that are not yet satisfied. We denote the set containing these combinations again by Ψ.

First we calculate the error between the mean value of time passed in the completion time requirements determined by the simulation, $\overline{\mathcal{W}}_m^{(k)}$, and by QNA, $\mathbb{E}[\mathcal{W}_m^{(k)}]$. For this

we use the function:

$$\varepsilon_2 = \frac{\mathbb{E}[\mathcal{W}_m^{(k)}] - \overline{\mathcal{W}}_m^{(k)}}{\overline{\mathcal{W}}_m^{(k)}}, \qquad \text{for } (m,k) \in \Psi. \tag{3.14}$$

Since the waiting time given by the simulation is bigger than the one given by QNA for this situation, this error has a negative value. Now we use this value to update the target time for the completion time requirements, which yields:

$$\tau_m^{(k)} \leftarrow (1+\varepsilon_2)\tau_m^{(k)}, \qquad \text{for } (m,k) \in \Psi. \tag{3.15}$$

The new target time will be smaller than the old and hence we can use QNA and its corresponding stopping criterion again without immediately satisfying the requirements. Note that this updating is meant for the QNA part of this iterative method and that we never use this updated target time when checking the capacity allocation using simulation. Also, we can immediately update the capacity, since we know that the current capacity does not satisfy the requirements yet. Therefore, we apply the following rule in situation 2:

When situation 2 occurs, move back to QNA. Update the target time for the completion time requirements as stated in Equation (3.15) *with the results of the current simulation and start with the current allocation updated following Section 3.3.4.*

Lastly, we have as possible scenario situation 3. In this case QNA has used more capacity than needed according to the simulation. In this situation, we do not change the target times as in situation 2, since we are not interested in the difference in expectation between QNA and the simulation. This is due to the fact that this difference could either be positive or negative, depending on whether QNA stopped too late due to the generality of Chebyshev's inequality or the overestimation of the waiting time. Hence, in this case we look at the difference between the upper bound of the confidence interval of the probability that the target time is exceeded given in Equation (3.12) and the threshold for this probability.

Following QNA in combination with Chebyshev's inequality, the current allocation does satisfy all the requirements while the previous allocation does not. Therefore, we know for the classes that do not satisfy a requirement in the previous allocation that we overestimated the fraction of customers that has not finished their route part within the target time. By looking at the difference between the fraction given by the simulation in (3.12) and the threshold, we update the threshold for QNA, such that QNA stops for a lower capacity. This yields the following update factor:

$$\varepsilon_3 = \frac{x_m^{(k)} - \text{CI}_m^{(k)}}{\text{CI}_m^{(k)}}, \qquad \text{for} \quad \text{CI}_m^{(k)} = \overline{X}_m^{(k)} + \frac{t_{r-1,1-\alpha/2}\sqrt{(\overline{S}_m^{(k)})^2}}{\sqrt{r}}, \tag{3.16}$$

where we use the values of the previous capacity allocation. Note that for some of the norms which are already satisfied, ε_3 can take a very big value if $C_m^{(k)}$ is very small.

However, we observed that situation 3 is a very rare case and therefore we decided to solve this by only changing the threshold for the class-norm combinations that have $\text{CI}_m^{(k)} > x_m^{(k)}/2$. The updated threshold then becomes:

$$x_m^{(k)} \leftarrow (1+\varepsilon_3)x_m^{(k)}, \qquad \text{for all } (m,k) \text{ for which } \text{CI}_m^{(k)} > x_m^{(k)}/2. \tag{3.17}$$

Again, we do not use this updated threshold when checking a capacity allocation using simulation.

For the start allocation we cannot use the current or previous allocation, since they both use too much capacity. Hence, we need to go back to a capacity allocation which for sure did not satisfy the completion time requirements according to the simulation. This is exactly the allocation which was tested during the last time we switched from QNA to the simulation. If in the previous simulation we had situation 1, we take the previous allocation from this simulation. If, on the other hand, situation 2 had occurred, we take the current allocation from this simulation. This all results in the following rule for situation 3:

When situation 3 occurs, move back to QNA. Update the threshold for the completion time requirements as stated in Equation (3.17) with the results of the previous allocation. Start with the allocation of the previous simulation. If in this simulation we had situation 1, start with the previous allocation, if we had situation 2, start with the current allocation.

3.3.7 Algorithm iterative method

In Algorithm 1 a summary of the iterative method can be found. The algorithm continues on the next page.

Algorithm 1: The iterative method for capacity planning

Step 1a. Initialisation
Aggregate arrival rates and service times. Initialise the capacity as a list of M capacities. Each initialisation is created by giving each station minimal capacity for which $\rho < 1$ and randomly add $\hat{\delta}$ to h stations. Also, initialise step size δ.

For $n = 1, \ldots, M$:
Select capacity $\hat{\beta}^n$ from initial capacity list.

Step 2. Performance analysis using QNA
Find the expectation $\mathbb{E}[\mathcal{W}_m^{(k)}]$ and variance $\text{Var}(\mathcal{W}_m^{(k)})$ of the waiting time of norm m and class k for capacity $\hat{\beta}^n$ using QNA.

Step 3. Stopping criterion of QNA
If the following holds for $\hat{\beta}^n$ move to *Step 5*:

$$\mathbb{E}[\mathcal{W}_m^{(k)}] + \sqrt{\frac{\text{Var}(\mathcal{W}_m^{(k)})}{x_m^{(k)}}} \leq \tau_m^{(k)}, \qquad \text{for all } m \in \mathcal{N}, k \in \{1,2,\ldots,K\}.$$

Else move to *Step 4*.

Step 4. Update rule
Select station j^* by looking at the largest decrease in waiting time of the requirements for the smallest increase of capacity. Update capacity as follows:

$$\hat{\beta}^n \leftarrow \hat{\beta}^n + \delta \cdot \hat{\mathbf{e}}_{j^*}.$$

Continue with *Step 2*.

Step 5. Discrete event simulation
Execute discrete event simulation for capacity allocation $\hat{\beta}_{\text{cur}}$ and $\hat{\beta}_{\text{prev}}$, with $\hat{\beta}_{\text{cur}} = \hat{\beta}^n$ and

$$\hat{\beta}_{i,\text{prev}} = \begin{cases} \hat{\beta}_{i,\text{cur}} - \delta, & \text{if } \hat{\beta}_{i,\text{cur}} - \delta > \hat{\beta}_{i,\text{stab}}, \\ \hat{\beta}_{i,\text{stab}}, & \text{otherwise,} \end{cases} \qquad \text{for } i \in \{1, 2, \ldots, J\}.$$

Use the Batch Means Method to determine $\overline{X}_m^{(k)}$ and $(\overline{S}_m^{(k)})^2$, the sample mean and variance of the probability that the target time of norm m is exceeded by class k.

Step 6. Stopping criterion of the iterative method
Check the following constraints for both $\hat{\beta}_{\text{cur}}$ and $\hat{\beta}_{\text{prev}}$:

$$\overline{X}_m^{(k)} + \frac{t_{r-1,1-\alpha/2}\sqrt{(\overline{S}_m^{(k)})^2}}{\sqrt{r}} \leq x_m^{(k)}, \qquad \text{for all } m \in \mathcal{N}, k \in \{1, 2, \ldots, K\}. \quad (3.18)$$

If Inequality (3.18) holds for $\hat{\beta}_{\text{cur}}$, but not for $\hat{\beta}_{\text{prev}}$ (situation 1):
If step size $\delta < \delta_{\text{min}}$ then $n = n + 1$, if $n = M$ then move to *Step 7*.
Else let $\delta \leftarrow \delta/\eta$ and move to *Step 2*. Let $\hat{\beta}^n \leftarrow \hat{\beta}_{\text{prev}}$.

Elif Inequality (3.18) does not hold for both $\hat{\beta}_{\text{cur}}$ and $\hat{\beta}_{\text{prev}}$ (situation 2):
Decrease target time $\tau_m^{(k)}$ for $(m, k) \in \Psi$ and let $\hat{\beta}^n \leftarrow \hat{\beta}_{\text{cur}}$. Continue with *Step 4*.

Elif Inequality (3.18) holds for both $\hat{\beta}_{\text{cur}}$ and $\hat{\beta}_{\text{prev}}$ (situation 3):
Loosen thresholds $x_m^{(k)}$ and let $\hat{\beta}^n \leftarrow \hat{\beta}_{\text{cur}}$ if the last simulation ended in situation 2 or $\hat{\beta}^n \leftarrow \hat{\beta}_{\text{prev}}$ if last simulation ended in situation 1. Continue with *Step 2*.

Step 7. Optimal capacity
The optimal capacity is given by:

$$\hat{\beta}_{\text{opt}} = \min_{n \in \{1,\ldots,M\}} \hat{\beta}^n.$$

3.4 Convergence of the iterative method

For the usability of our iterative method, we are interested in whether the method actually stops after a limited amount of time and whether it the convergences to a solution close to the optimal capacity allocation. In Section 3.3.6, we described the interaction between QNA and the simulation and when we are satisfied with the capacity allocation given. In this section we first argue why the method stops and afterwards why this method converges.

When running the simulation, we see three different situations when looking at the current and previous capacity allocation, where for each situation we take a different action. Only situation 1 can lead to the end of the iterative method, since we decide to stop in this situation when the step size δ for updating is sufficiently small. This is also the only situation where we decrease the step size, such that it can become sufficiently small, since we know we are close to the optimal solution. Beforehand, we set a bound for the step size, which it must transcend to be sufficiently small. Together with the method for updating the step size, we can determine the number of times situation 1 must occur before the iterative method stops. Therefore, we know that if we have situation 1 for a limited amount of times, the iterative method will stop.

In the QNA phase of the method, we only add capacity each iteration. Hence, each iteration we come closer to a capacity that satisfies the completion time requirements, i.e. the constraints (3.2b), if these were not satisfied yet. Note that this does not necessarily mean that we come closer to the optimal capacity, since we could have already surpassed the optimal capacity when only adding capacity each step. Next, we run the simulation phase to check the capacity allocation, where we could end up in 3 different situations. These situations and the possibilities after each situation are described in Figure 3.2. We call a capacity allocation feasible if it satisfies the completion time requirements according to the simulation.

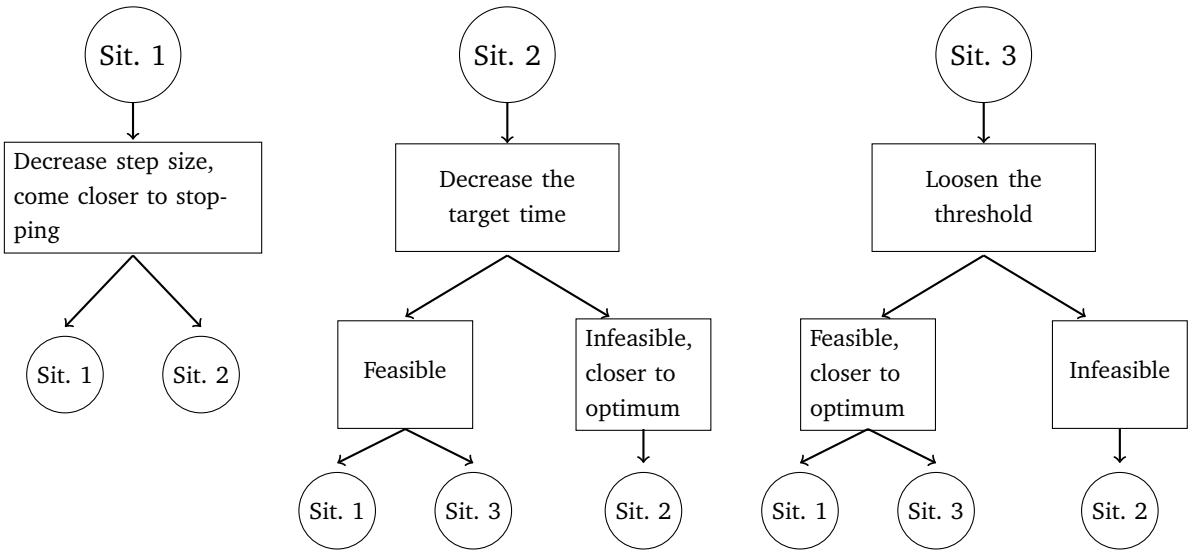

Figure 3.2: Possible progress in the iterative method.

In situation 1, we decrease the step size and come closer to the end of the method. In the next simulation phase we could end up in situation 1 again, or in situation 2, if not enough capacity was added to satisfy the requirements according to the simulation. Note that if we end up in situation 1, the new solution will be closer or just as close to the optimal solution.

When the simulation gives situation 2, we adjust the target time for the norms that are not satisfied yet and start a new QNA phase. This new QNA phase will therefore end with an allocation that has a bigger total capacity than the QNA phase before. Either the capacity allocation is closer to the optimum, but still does not satisfy the completion time requirements, or it is a feasible solution. These options correspond to situation 2 or situation 1 or 3.

Lastly, we have situation 3. In this case, we adjust the requirement thresholds, such that they become more strict, which causes that the QNA phase stops for a lower total capacity. We start the next QNA phase with a capacity allocation that did not yet satisfy the requirements. Consequently, in the next iteration the capacity given by the QNA phase will either be closer to the optimal capacity and satisfy the completion time requirements or will not be feasible yet. These options correspond to situation 1 or 3, or situation 2.

Because we are always striving to come closer to the optimum with the adjustments to the constraints, we come closer to ending up in situation 1 as well. However, the error of QNA with respect to the simulation is hard to predict and may not always be the same. Therefore, we cannot say with certainty how many times we end up in situation 1. However, the difference between QNA and the simulation does give an indication of the error for the current capacity allocation and hence can also give an indication of the error for capacity allocations in the neighbourhood. Therefore, it is very likely that we reach situation 1 after a limited number of QNA phases. Since we only need to have situation 1 a limited number of times, the iterative method will eventually stop.

For this method, we always stop in situation 1, which means that the current capacity allocation satisfies all the completion time requirements, while the previous did not. Therefore, we know that the current capacity must be close to the optimal capacity. Since the stopping criterion after the simulation is quite strict, we always end up in a feasible solution with respect to the constraints (3.2b). It still could happen that at one station we added too much capacity, but by using multiple initialisations, we make different decisions regarding the station to which we add capacity. Hence, we increase the probability that the method converges to a capacity allocation close the optimum.

Chapter 4

Results

In this chapter we discuss the results obtained with the iterative method, described in the previous chapter. First, we test the iterative method with a simple network, for which the performance measures can be determined in closed form. Furthermore, we carry out a case study for a Dutch hospital called ZGT in Section 4.2. We describe the problem in this hospital and the patients' care pathways through the hospital. Thereafter, we link the problem to a queueing network and our iterative method and explain the data and analysis done. Lastly, we discuss the results for this case study.

4.1 Testing the iterative method

We test the iterative method using a simple test network, for which we can determine the waiting time distribution in closed form. If a network has this characteristic, we are able to find the optimal capacity for the given constraints. First we introduce the chosen network and discuss the optimal capacity. Afterwards, the iterative method is run for this network and we compare the found capacity allocation with the optimal capacity allocation determined using the closed form results available for the network.

4.1.1 The test network

The test network consists of three $M/M/1$ stations and two customer classes, each with only one type. Customer class 1 visits stations 1, 2 and 3 on its route, which is given as the dashed arrows in Figure 4.1. Customer class 2 visits stations 1 and 2, which is given as the solid arrows in Figure 4.1. The completion time requirements are given in this figure as well. For class 1 we have the two completion time requirements in orange: station 1 must be finished within 8 time units and station 3 must be finished within 10 time units. For class 2 we have one completion time requirement in blue: station 1 and 2 must be finished within 15 time units.

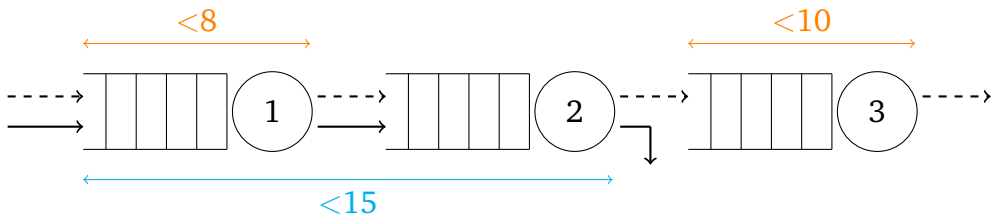

Figure 4.1: Test queueing network.

The expectation and SCV of the service and interarrival times can be found in Tables 4.1 and 4.2, respectively. We assume the service time at each station is the same for both customer classes.

Table 4.1: Service process at each station.

Station i	$\mathbb{E}[G_i]$	C_{bi}^2
1	$1/4$	1
2	$1/3$	1
3	$1/2$	1

Table 4.2: Arrival process for each class.

Class/Type s	$\gamma_0^{(s)}$	$(C_0^{(s)})^2$
1	6	1
2	4	1

The network shown in Figure 4.1 is a single server tandem network, which is a queueing network with nice properties. Namely, the sojourn time variables at the stations are independent [39], which enables us to determine the waiting time distribution at each station, but also over multiple stations. Therefore, we can find the optimal capacity at each station for which the completion time requirements are satisfied. The distribution of the waiting time for an $M/M/1$ queue is given by:

$$\mathbb{P}(W_i > t) = \rho_i e^{-\mu_i(1-\rho_i)t}, \qquad \text{for } i = 1, 2, 3 \text{ and } t \geq 0, \tag{4.1}$$

where $\mu_i = \beta_i/\mathbb{E}[G_i]$. This cumulative distribution function can be interpreted as a mixture of two distributions; the waiting time is 0 with probability $(1-\rho_i)$ and exponentially distributed with parameter $\mu_i(1-\rho_i)$ with probability ρ_i. Two of the requirements only include one station, so for these the above formula suffices. However, the requirement for customer class 2 contains two stations. Since the waiting times are independent, we can take the convolution of the density functions of the waiting time distribution to obtain the density function of the total waiting time:

$$f_{W_1+W_2}(z) = \int_0^z f_{W_1}(t) f_{W_2}(z-t) \mathrm{d}t.$$

Since the waiting time is a mixture of distributions, the density is not straightforward to determine by taking the derivative of the cumulative distribution. Therefore, this convolution is also not straightforward. However, we mentioned in Section 3.2 that we assume that the network occupation is high and we therefore can look at the sojourn

time instead of the waiting time. We use this same assumption here and say that the waiting time distribution is approximately the sojourn time distribution, which gives the following:

$$\mathbb{P}(W_i > t) \approx e^{-\mu_i(1-\rho_i)t}, \qquad \text{for } i = 1, 2, 3 \text{ and } t \geq 0,$$

Note that $\rho_i < 1$ and hence this probability is always larger than the one given in (4.1). Therefore, when taking this probability, we always have sufficient capacity to satisfy the requirements. This gives the following density function:

$$f_{W_i}(t) \approx (1 - \rho_i)\mu_i e^{-\mu_i(1-\rho_i)t}, \qquad \text{for } i = 1, 2 \text{ and } t \geq 0.$$

By integrating $f_{W_1+W_2}(z)$ we can obtain the cumulative distribution function of the waiting time over station 1 and station 2.

Taking 0.05 as threshold for all completion time requirements, we obtain the optimal capacities as given in Table 4.3.

Table 4.3: Optimal capacities in test network at each station.

Station i	Optimal capacity
1	2.593
2	3.426
3	3.148
Total	9.167

4.1.2 Numerical results

We run the iterative method for the test network and compare the results to the optimal capacity determined in the previous section. We first state some parameter values taken for this test network. These are given in Table 4.4.

Table 4.4: Parameters for the iterative method.

Description	Parameter	Value
Step size used for initialisation of random restart	$\hat{\delta}$	0.025
Number of (distinct) initialisations	M	8
Number of stations to which the initial step size is added	h	3
Starting step size for iterative method	δ	0.1
Update factor for the step size	η	2
Window size in the moving average method	q	10
Periods to look back at in moving average method	z	3
Stopping parameter for the warm-up period	ω	0.05
Number of batches	r	25
Confidence interval percentage	α	0.05
Sufficiently small step size	δ_{min}	0.025
Total number of departures in the simulation	-	$1,000,000$

Each time situation 1 occurs (see Section 3.3.6 for the explanation of this situation) in the simulation phase of the iterative method, the step size for updating the capacity is divided by 2. This means that situation 1 must be reached 3 times before the iterative method stops. The iterative method is executed twice for 5 initial capacities. Among the 10 initial capacities, 8 distinct initial capacities were considered as 2 initial capacities are considered twice. The results for the iterative method are given in Table 4.5, as well as the optimal capacity and the relative difference between the two. This difference gives the increase of the capacity given by the iterative method compared to the optimal capacity. In the table, the optimal capacity given by the iterative method is referred to as the optimal IM capacity. The sub-optimal capacity allocations from other initial capacities are referred to as IM capacities.

Table 4.5: Capacity given by the iterative method compared to the optimal capacity in test network.

Station i	Optimal IM capacity	Optimal capacity	Absolute difference	Relative difference
1	2.601	2.593	0.008	0.31%
2	3.435	3.426	0.009	0.26%
3	3.151	3.148	0.003	0.10%
Total	9.187	9.167	0.020	0.22%

We see that the optimal capacities given by the iterative method are close to the optimal values. The optimal IM capacities are always larger than the optimal values, which means that the iterative method has determined capacities which indeed satisfy the completion time requirements. The relative difference between the two is less than 1 percent for each station and in total only 0.22%.

The fact that there is a difference between the IM capacity and the optimal capacity can be explained by the decision to take a discrete step size when updating the capacity. The smallest step size taken is 0.025. We see that for this optimum, the absolute difference is always less than this step size. However, this is not the case for every initialisation. We note that different initialisations can lead to different optima, but the same initialisation can also lead to a different optimum in this iterative method. In Table 4.6 we see some initialisations with their capacity allocation according to the iterative method.

Table 4.6: Capacity given by the iterative method for different initialisations.

	Initialisation per station			IM capacity per station		
	1	2	3	1	2	3
Initialisation 1	2.551	3.360	3.001	2.601	3.435	3.151
Initialisation 2	2.551	3.360	3.001	2.626	3.435	3.151
Initialisation 3	2.551	3.335	3.026	2.626	3.460	3.176

In the table, initialisation 1 gives the optimal capacity given by the iterative method. Initialisation 2 has the same initial capacities, but a different IM capacity was obtained.

We note that at most one step size is added to the optimal capacity in the outcome of initialisation 2 and 3. For the other initialisations the IM capacity differed at most two step sizes from the optimal IM station. For this example, this results in a worst case relative difference between an IM capacity of a single station and the optimum obtained in Table 4.3 of 1.69%. The total relative difference is at most 1.31%.

The cause of this difference between multiple runs, even with the same initialisation, lies in the simulation phase. In Section 3.3.6 about the stopping criterion of the whole iterative method, we describe that the whole confidence interval of the probability that the target time is exceeded must lie below the threshold for each requirement. Otherwise, we do not view this requirement as satisfied. Due to the nature of a confidence interval we are very strict here. Each time we run the simulation the confidence interval will be slightly different, since we are dealing with uncertainty in interarrival and service times. Therefore, the number of requirements that is satisfied can be different even though we give the same capacity allocation as input.

Taking a small update step size also affects the simulation outcome, since it results in a higher chance of having simulation results for the current and previous allocation that are very similar. However, this small step size also provides the possibility to come closer to the optimum. Moreover, the initial stable capacity was given by the vector $\hat{\beta} = (2.501, 3.335, 3.001)$. If we compare this to the optimal capacity, only a small amount of capacity should be added to reach it. Therefore, it is important to choose the step size according to the problem, not only for the update step size δ, but also the initialisation step size $\hat{\delta}$.

Since we use multiple initialisations, we see that we can still come very close to the optimal capacity. Also, because we take a small step size, the difference between the optimal capacity and the optimal IM capacity is never big. If for this step size we would translate the capacity allocation into working hours in a 40-hour work week, this would results in an extra hour per added step size. Therefore, the capacity could differ 1 or 2 hours per station.

Lastly, we look at the course of the iterative method, with respect to the number of QNA and simulation phases. In Figure 4.2 the number of QNA phases and the situation given in the simulation after each QNA phase are shown for the optimal IM capacity. We see similar routes for the other initial capacities. The dark grey nodes give the number of QNA iterations done before moving to the simulation phase. The light grey nodes give the situation obtained in the simulation phase. The whole iterative method for 10 initial capacities takes around 4.5 hours, when executed on the laptop model Thinkpad T460s Signature Edition, with processor Intel(R) Core(TM) i5-6200U CPU @ 2.30GHz 2.40 GHz, RAM of 8.00 GB and Windows 10 version 21H1.

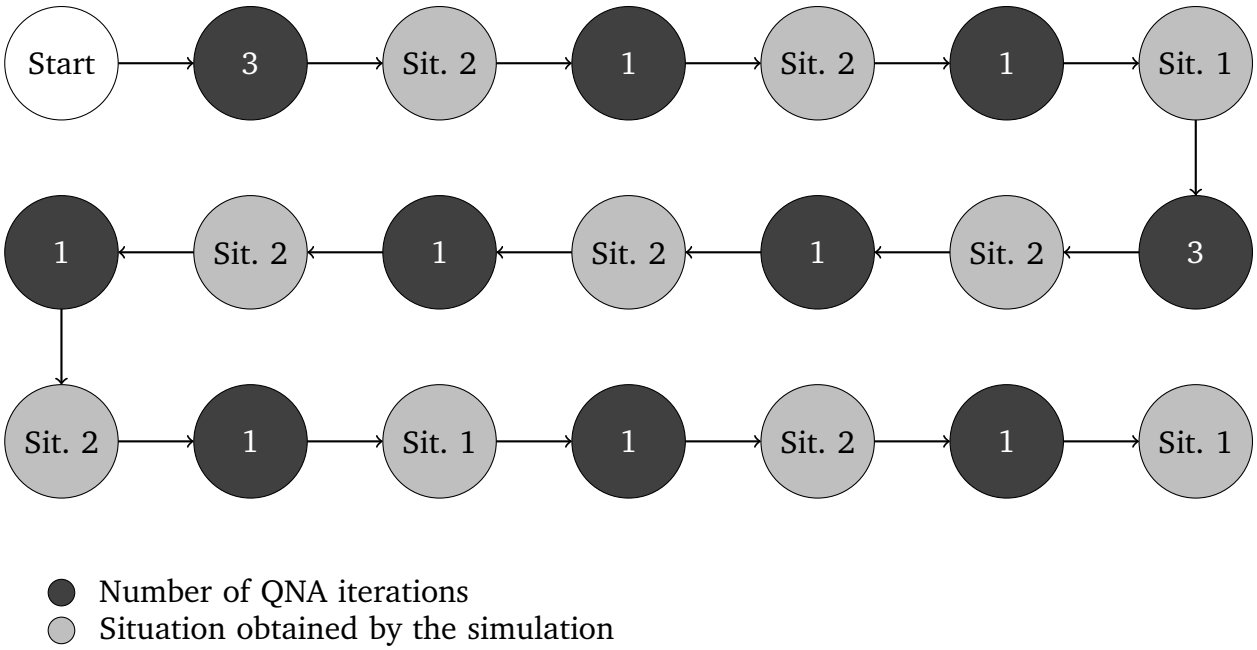

Figure 4.2: Course of iterative method for optimal IM capacity.

In the figure we can see that 10 simulation phases were needed and 14 QNA iterations. Note that the number of simulation phases is the same as the number of QNA phases, since a simulation phase is always preceded by a QNA phase. In general, we see that the first QNA phase is 3 or 4 iterations long, depending on the initial capacity. The first simulation phase then results in either situation 1 or 2. After situation 1 we often have a shorter QNA phase than right after the start. We see that if situation 1 occurs again after this, the length of the QNA phase decreases and is most of the times only one iteration long. After situation 2 the next QNA iteration results immediately in a next simulation phase, with only a few exceptions where the QNA phase is two iterations long. We execute the next simulation for a different capacity allocation when this happens, since we update the current capacity after situation 2 and begin the QNA phase with this new capacity.

The fact that we see situation 2 in multiple subsequent iterations with only one QNA phase in between implies that the update rule for the target times in situation 2 does not influence the stopping criterion of QNA a lot. Only a capacity update for one of the stations, which we execute directly after situation 2, is already enough to start a new simulation phase. However, we still see situation 2 occurring afterwards, which means that we could have set an even stricter stopping criterion for QNA. This implies that the QNA phase often underestimates the fraction of customers that cannot finish their route part within the given target time. Note that this can also be a consequence of taking the upper bound of the confidence interval of the fraction of customers that cannot finish within the target time in the simulation, which is a rather strict bound.

From this test network we can conclude that the iterative method is able to come very

close to the optimal capacity. The sum of the optimal capacity given by the iterative method only differs 0.22% from the actual optimum. We do note that per initialisation the IM capacity can be slightly different due to the sensitivity of the simulation. Also, for this test network the iterative method needs to execute quite a large number of simulation phases compared to the number of QNA iterations. For larger networks this could result in a long run time.

4.2 Case study: outpatient oncology clinic

Besides the simple test case in the previous section, we also conduct a case study for a hospital in the Netherlands. Therefore, we first give the problem description. Afterwards, we discuss the care pathways of patients and the completion time requirements which must be satisfied within these care pathways. The link between the hospital situation and a queueing network is described next and the used data is explained. Lastly, we show the capacity planning results obtained with the iterative method.

4.2.1 Problem description

Ziekenhuisgroep Twente (ZGT) is a hospital in the east of the Netherlands with two locations, one in Almelo and one in Hengelo. In 2017, the two hospitals together had over 3200 employees and around 300 volunteers [40]. In 2018 the hospital decided to make the distinction between the two locations more clear by distinguishing the health care offered at the locations [41]. Almelo was given the focus on the intensive care, high risk care and the clinical patients, while Hengelo developed into a service oriented treatment centre for short stays (at most one night). This includes an outpatient oncology clinic where the health care should become patient-centred by means of more intensive interdisciplinary collaboration.

In the current situation, different parts of the cancer patient's care pathway take place at both locations. A pathway consists of the referral to a specialist by a family doctor or colleague specialist, the first intake with the specialist, the diagnostics, a multidisciplinary meeting, the treatment and the follow up care. A patient visits multiple outpatient clinics, each with its own specialisation, but this is about to change.

At the end of 2021, the ZGT in Hengelo will become the meeting point for the majority of the oncology specialists, nurse practitioners, oncology nurses and supportive practitioners. At the outpatient oncology clinic a team of dedicated oncology professionals will work together interdisciplinary such that the health care can be centred around the patient. This will be done in close collaboration with the oncology professionals at the outpatient clinic in Almelo and at the clinical oncology departments. Most of the professionals will be working partly in Hengelo and partly in Almelo. For their outpatient appointments patients have to visit one outpatient clinic most of the time, in which the most of the care pathway will take place, from the intake until the follow up care. Moreover, this will also benefit the specialists as they can interact more easily during

the day.

To establish the situation described above a reevaluation of the current situation around outpatient appointment planning is recommended. We analyse the availability of specialists and medical staff needed to meet the requirements regarding the time horizon of an oncology care pathway. In the report of the organisation *Stichting Oncologische Samenwerking* (SONCOS, English: Foundation for oncological collaboration) norms for the maximum time between the stages of the care pathways is specified [42]. The specialists should use their time efficiently, so that they have enough time for their other tasks. All this asks for a reconsideration of the capacity planning.

With the SONCOS norms and completion time constraints and the limited availability of specialists, this problem is suitable for our iterative method. We investigate the outpatient appointment hours needed per medical professional in order to satisfy the completion time requirements. We focus on the care pathways of the department breast oncology. The outpatient appointments of this department will fully take place in Hengelo after the establishment of the outpatient oncology clinic.

4.2.2 Patients' care pathways and completion time requirements

An important part of the iterative method are the routes of customers used as the input for the model. In the hospital case, we call them care pathways. For each oncology patient we can roughly distinguish the same stages in the care pathway. These care pathways consist of the referral to a specialist in a hospital by a family doctor or colleague specialist, the first intake with the specialist and nurse practitioner or oncology nurse, the diagnostics phase, a multidisciplinary meeting (MDO, *Dutch: multidisciplinair overleg*), the treatment and the follow up care. This is of course a very generalised care pathway and the interpretation of the stages differs per patient and cancer type. In Figure 4.3, a graphical representation of the stages of the care pathway can be seen.

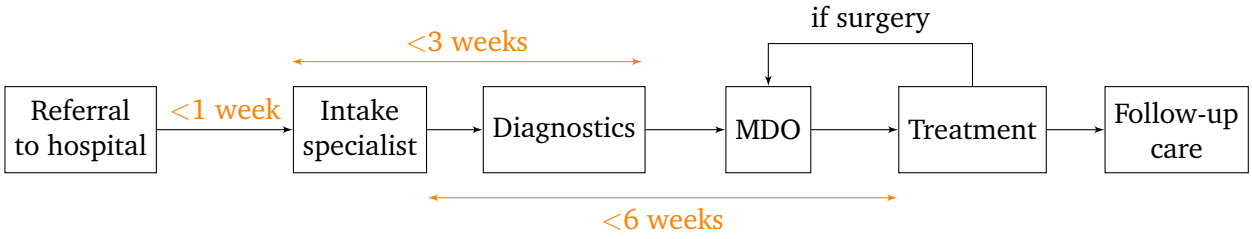

Figure 4.3: Care pathway stages with SONCOS time requirements.

The organisation SONCOS writes a yearly report about the expectations with regard to the multidisciplinary vision on the quality of the oncological health care in the Netherlands [42]. The first report came out in 2012 and every year since then a new report is written, where the previous one is updated and improved. Hospitals have one year to implement the requirements, unless stated otherwise. The report contains some general requirements of the oncological health care and some cancer-type-specific requirements.

This report includes, among other things, standards about the length of each stage of the care pathways in the oncological care. The time requirements set by SONCOS can also be found in Figure 4.3.

When the family doctor suspects someone is a possible cancer patient they will refer the patient to see a specialist in a hospital. An appointment will be made with the hospital. In the newest report of SONCOS the norm is set that in general the waiting time for the first intake must not exceed one week [43]. If the patient has had the first intake with the specialist and the nurse practitioner or oncology nurse the diagnostics trajectory can begin. We often already see the first appointment at the hospital as a part of the diagnostics. The complete diagnostics trajectory may not last more than 3 weeks [43]. This can include for example physical examination, a computerised tomography (CT) scan or biopsy depending on the cancer type.

The next step is an MDO with all concerned oncology professionals and the possibility of consultation with external parties, like an academic hospital. The medical staff that should be present for every cancer type are specified in [43]. In most departments, these meetings are held weekly at a fixed time, often during lunch hours or at the end of the day, unless the cancer type allows a less frequent meeting. SONCOS states that these meetings should discuss at least 90% of the patients [43]. During these meetings the treatment path is discussed. Many departments discuss a patient in an MDO before the treatment plan is discussed with the patient. Some care pathways also contain a postoperative MDO discussion. This is only the case when the first treatment is a surgical intervention. After each MDO the family doctor of the patient is updated about the topics discussed in the MDO.

When a patient starts their treatment, this must be within six weeks after the first intake at the hospital [43]. Treatments can consist of, but are not limited to: a surgical intervention, chemotherapy, radiation or some combination of those. The goal is to let the patient see the same specialist every time, if possible. Every patient will also be assigned to a case manager, which will be their point of contact throughout their hospital journey and in the follow-up phase. The low complexity treatments take place in Hengelo, but the more complex interventions, such as extensive surgery with ICU backup or robot surgery, will be executed in Almelo. Hengelo will only be the location for short stays (at most one night). Sometimes surgery is performed in other surrounding hospitals. Radiotherapy is conducted in Enschede and Deventer. The only exception is the breast cancer care pathway, this almost fully takes place in Hengelo, except for radiotherapy.

The last part of the care pathway is the follow-up care. Depending on the cancer type this is a meeting 2 weeks, 1 month, 3 months, 6 months or 1 year et cetera after the ending of the treatment phase. Every care pathway makes their own agreements about content and frequency of the follow up care following the national guidelines [43].

4.2.3 Queueing network design and data analysis

The problem of ZGT can easily be modelled as a queueing network. We are interested in the capacity needed for outpatient appointments per medical profession. Hence, we can see the different professions as the stations in the queueing network, where we make no distinction between individual specialists, but rather look at their role in the care pathway. The following professions are included in the breast cancer pathway and therefore represent a station in the queueing network:

- Breast surgeon
- Nurse practitioner medical oncology
- Medical oncologist
- Breast doctor
- Breast nurse
- Day treatment nurse
- Radiotherapist
- Nurse practitioner breast

With the server speed, we determine the fraction of a 40-hour work week that is needed for outpatient appointments. Since we have multiple staff members per profession, this server speed could also be bigger than one.

In the hospital we have different types of cancer. Per type different diagnostic phases and treatments are possible, which results in a different sequence of outpatient appointments. Also, for each type the SONCOS norms are specified. The cancer types represent the customer classes. In this case study we only look at one cancer type, namely breast cancer, so we have one customer class. The possible routes represent the types within a class. It could happen during a customer's path that a consultation is with multiple oncology professionals, for example a specialist and a nurse practitioner. These professionals also have some consultations on their own, so we cannot model it as one station. Hence, we model an appointment with for example a specialist and a nurse practitioner as first a visit to the specialist's station and afterwards a visit to the nurse practitioner's station. In this way we can also have appointments with a single practitioner, but still capture the actual demand arriving to each profession. We assume the arrival of patients happens according to a Poisson process. Moreover, the service times are assumed to be deterministic.

In order to determine the routes through the network of possible types, we look at the data gathered by the hospital and information gained from conversations with health care professionals. In this way we also determine the length of appointments. The ZGT provided the data of all outpatient appointments held in 2018 and 2019 which are linked to the oncology breast department. In this data we can see the appointments an oncology patient has had in these two years along with additional information, for instance on the length and date of the appointments. In Table 4.7 a few fictional patients are given together with the most important information about their appointments.

Table 4.7: Outpatient appointments in the data set.

Patient number	Start date	Appointment length	Medical staff	Appointment code
12345678	05-11-2018	15 min	Specialist 1	BELCONS
12345678	12-01-2019	30 min	Nurse 2	HCM
23456789	03-03-2019	10 min	Medical oncologist 1	NCONC
34567890	24-07-2019	20 min	Nurse practitioner 3	HCONC

We refer to a patient by their patient number, given by the first column. The second and third column give the date at which the appointment took place and the scheduled time for the appointment. The fourth column specifies the professional who held the appointment, which, in the original data set, is given as the name of the individual and not as the profession. In the last column we can find the appointment code which gives the nature of the appointment. BELCONS, for example, denotes a phone consultation, while HCM stands for a repeated consultation. Besides the columns given in the table, we also have information about the registration date of the appointment, the planned start and finish time and whether the appointment was cancelled or not. These cancelled appointments are deleted from the data. Since we have the patient number for each appointment, we can see the path of outpatient appointments a patient has followed.

After talking to a breast nurse, we learned that most appointments with both a surgeon and a nurse are only registered for the surgeon. Therefore, the data underestimates the number of appointments with a breast nurse. To take into account this extra demand, we assume that all appointments with a surgeon in the data were also attended by a nurse. This was confirmed by the breast nurse as a reasonable assumption.

From the data we need to extract care pathways. However, each patient is unique and therefore not many patients follow the exact same route through the hospital. In order to know whether the SONCOS norms are satisfied, it is important that we can distinguish a certain number of routes and define route parts which have a completion time requirement. Therefore, we talked to different professionals and used the information gained from them to establish a number of paths through the network. Based on the load for each station obtained from the data we selected a subset of these paths to use in the iterative method. The selected paths can be found in Table B.3 of Appendix B.1. The arrival rates of each path are also given in Table B.3. Since the paths are purely based on the loads per station and only a subset is taken from the paths established in consultation with health care professionals, the results are meant as an illustration of the practical applicability of our iterative method, not as a direct advise for ZGT.

Within each path we need to determine the stations included in a SONCOS norm. These norms include the waiting time until the first intake, the diagnostic phase and the time between the first intake and the start of the treatment. The intake and diagnostics norm can easily be determined from the paths in Table B.2 included in Appendix B.1. However, the start of the treatment is not so easily determined from only outpatient

appointments, since patients do not have an appointment just before their treatment. From health care professionals we learned it takes approximately 10 days until a treatment starts after the last appointment. Therefore, we take the last outpatient appointment before the surgery or start of the chemotherapy as the start of the treatment and subtract 10 working days from the original norm of 6 weeks. We also assume for the norms that people can only arrive during working hours and that 1 week contains 40 working hours. This yields the target times given in Table 4.8.

Table 4.8: Target times of the SONCOS norms for breast cancer.

Norm	Target time (hours)
Intake	40
Diagnostics	120
Start treatment	160

We note that some of the stations are not included in any of the norms. These stations are the radiotherapist, since radiotherapy is never the first treatment, the nurse practitioner breast, since this profession is only involved in the follow-up phase, and the nurse practitioner medical oncology, since this profession is only involved in the treatment itself. In this report we decided to only focus on the waiting time of stations in the norms, but if needed one could add an extra requirement for those single stations to reduce the waiting time there.

4.2.4 Numerical Results

We use the iterative method to find the number of outpatient appointment hours that need to be scheduled in order to meet the SONCOS norms in our case study. In the previous section we described the path and station inputs. All remaining input values for the simulation and step size used to generate the results can be found in Appendix B.2. For the ZGT case we decided to use a fractional step size instead of a discrete step size to generate the different initial capacities. This means that instead of adding a fixed amount to the capacity, we add a certain percentage of the stable capacity. This decision was made since we noted a big difference in needed capacity between the stations, such that for some stations this discrete step size had a much bigger influence on the waiting time than for other stations.

The iterative method is used for different thresholds, to see the behaviour of the capacity allocation under different thresholds. We used the thresholds 0.05, 0.025 and 0.01 and the threshold was the same for all norms in each run. This means that the results give the capacity such that at least 95%, 97.5% and 99% of the patients can be seen within the target times of the norms. The optimal capacity for each threshold together with the stable capacity can be found in Table 4.9. In this table the server speeds are translated into outpatient appointment hours needed per week. All capacities are rounded up to ensure enough capacity is used to satisfy the SONCOS norms. Again, we do not advise

the ZGT to use these results, since the input is based on the loads per station and not on the real data.

Table 4.9: Optimal number of outpatient appointment hours per 40-hour working week for different thresholds.

Station	Stable capacity	$x_m^{(k)} = 0.05$	$x_m^{(k)} = 0.025$	$x_m^{(k)} = 0.01$
Surgeon	37.28	38.16	38.29	38.79
Nurse practitioner medical oncology	10.62	10.62	10.62	10.62
Medical oncologist	19.04	19.54	19.54	19.54
Breast doctor	14.76	15.77	15.91	16.77
Breast nurse	110.39	111.39	111.39	111.89
Day treatment nurse	2.71	3.24	3.21	3.24
Radiotherapist	2.12	2.12	2.12	2.12
Nurse practitioner breast	11.06	11.06	11.06	11.06
Total	207.95	211.86	212.11	213.98

In the table we can see that in order to satisfy the SONCOS norms for the different thresholds a bit more capacity should be used than the capacity needed for a stable system. However, for individual stations this difference is never very big, meaning the target times required are not very strict for the arrival rates of patients used in this case study. Depending on their part in the care pathways and the length of their appointments, some health care professions need many outpatient appointment hours, like the breast nurse, while others need only a few, like the radiotherapist and the day treatment nurse. Moreover, as expected, we see that the capacity at the stations nurse practitioner medical oncology, radiotherapist and nurse practitioner breast do not change during the iterative method, since they are not included in any of the norms. Therefore, the stable capacity of these stations is the optimal capacity.

For the other stations we do see a change in capacity when we run the iterative method and decrease the threshold. When we look at the total hours of outpatient appointments needed to satisfy the norms, we see that the number of hours increases if the threshold increases. For the individual stations, this is most often the case as well. For a threshold of 0.05, however, the outpatient appointment hours for the day treatment nurse is higher than for a threshold of 0.025. Also, for the stations medical oncologist and breast nurse the optimal capacity does not change when we increase the threshold.

One way to explain the decrease in capacity of the day treatment nurse when increasing the threshold, is the increase of capacity of other stations. The day treatment nurse is only part of the start treatment norm, in which a lot of other stations are also involved. We see an increase in the stations surgeon and breast doctor, which results in less capacity needed for the day treatment nurse to still satisfy the stricter start treatment norm. A different initialisation has also played a role, since the difference is smaller than the minimal step size of the capacity. Therefore, the difference must be caused by the initialisation step, since there we increase the capacity fractionally instead of

discretely.

Similar as for the test case, we see that different initial capacities lead to different optimal capacities found by the iterative method. Since we now consider more stations and more complex patient routes, multiple capacity allocations can lead to a feasible system, with respect to the completion time requirements. We see that it really depends on the initialisation which capacity is decided to be the optimum according to the iterative method. In Table 4.10 the outcome of three different runs of the iterative method, each with 10 initial capacities, can be seen for a threshold of 0.05. The stations not included in any of the norms have been omitted, since the capacity for these stations did not change in any of the runs.

Table 4.10: Optimal number of outpatient appointment hours per 40-hour working week for different initial capacities with threshold 0.05.

Station	Run 1	Run 2	Run 3
Surgeon	38.16	38.29	38.16
Medical oncologist	19.54	19.54	19.54
Breast doctor	15.77	16.06	15.91
Breast nurse	111.39	111.89	111.89
Day treatment nurse	3.24	3.21	3.21
Total	211.86	212.77	212.49

Run 1 gives the minimal capacity out of the three iterative method runs, while run 2 and 3 are the capacity allocations found in the other runs. From the table we note that both run 2 and 3 gave a higher or equal capacity for all of the stations included in one of the norms compared to run 1, except for the station of the day treatment nurse. We can relate this difference to the minimum step size taken as input. This namely has the value of 0.0125, which is converted to a 40-hour work week equal to half an hour. We observe that for all stations, except the breast nurse, the outcomes of run 2 and 3 lie within this half an hour from the outcome of run 1. Therefore, we can conclude that it depends on the initialisation which minimal outcome is found by the iterative method. for the station of the breast nurse we see the effects of the careful decisions we make regarding the feasibility of a capacity allocation in the simulation phase. Moreover, we needed to take a small step size for this case study, which makes the difference between the current and previous capacity analysed with the simulation very small.

If we compare run 2 and run 3, we see they are very similar, except that run 2 has added more capacity to the stations of the surgeon and the breast doctor. Again, this is a consequence of a different initial capacity. This suggests that we could possibly use the information from the old initial capacities to choose new initial capacities. We note that the initial capacity of the stations surgeon and breast doctor in run 3 enable the iterative method to end up with a lower overall capacity. Therefore, we could decide to fix the initial capacities of run 3 for those stations and only change the other stations when generating an initial capacity. We should be careful though that we do not make

this decision too soon, since then we could miss out on other initial capacities which lead to a more optimal capacity allocation.

Lastly, we look at the behaviour of the iterative method. Situation 1 must have happened three times in order for the iterative method to stop. Similar as for the test case, we see that besides situation 1, situation 2 happens the most often. During the three different runs, each with 10 different initial capacities, situation 3 occurred only twice. The course of the iterative method for initial capacity resulting in the optimal capacity for a threshold of 0.05 can be found in Figure 4.4. The whole iterative method for 10 initial capacities takes around 2 hours, when executed on the computer HP Pavilion Desktop 595-p0xx, with processor Intel(R) Core(TM) i7-9700F CPU @ 3.00GHz 3.00 GHz, RAM 16.0 GB and Windows 10 version 21H1. Note that this is a different computer than the one used to run the test case.

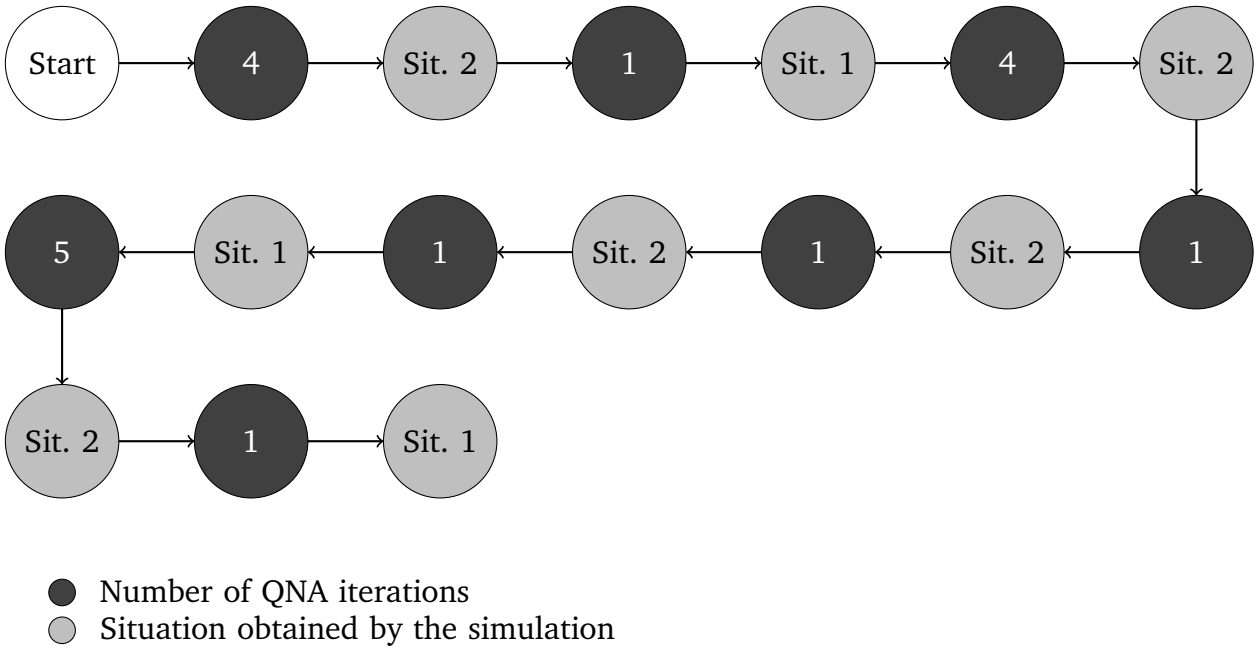

Figure 4.4: Course of iterative method for optimal capacity and threshold 0.05.

We see that after a simulation resulting in situation 2 only one QNA phase is executed, which we also saw in the test case. This means that the current capacity of this simulation only differs in one station compared to the last simulation, since after situation 2 capacity is added to only one station. Again, the target time update did not influence the QNA phase a lot. We note that whenever the iterative method comes closer to a feasible capacity, the simulation is executed more often to check if the current capacity already satisfies the completion time requirements. When situation 1 occurs, it takes multiple QNA phases until the next simulation. For the different initialisations the number of times situation 2 occurs ranges from 0 to 6.

Chapter 5

Conclusion and discussion

In this master thesis we have designed an iterative method for capacity planning in queueing networks. The iterative method combines an approximation method, called the Queueing Network Analyser (QNA), and Discrete event Simulation. The iterative method was tested on a small test network and afterwards a case study was conducted for the hospital ZGT. In this chapter, the main conclusions are presented and recommendations for further research are discussed.

5.1 Main conclusion

The aim of this research was to design a capacity allocation method for queueing networks, such that the total capacity used is minimised and the strict requirements for the completion time of customer's route parts are satisfied. We looked into two possible methods to analyse a general queueing network, namely the approximation method QNA and Discrete Event Simulation. Due to the advantages and disadvantages of both methods and the strict completion time requirements, we decided to combine them into one iterative method.

Based on the test case we have seen that the designed method is capable of finding a capacity very close to the theoretical optimum. Furthermore, we observed that the method uses QNA to take the first steps in the direction of the optimal solution. When approaching the optimum, simulation is used more often to check regularly if we have actually reached it. This shows that it is important to check the completion time requirements with a simulation after the QNA phase has found a feasible capacity allocation, since we observe that often they are not immediately satisfied. Therefore, the interaction between QNA and the simulation within the iterative method is as we intended it to be.

We have shown an application of the iterative method in the case study for the hospital ZGT. The cancer department has to deal with norms regarding the completion time of parts of a patient's care pathway, which makes the iterative method developed here a

suitable way to determine the capacity planning. We see the same interaction between QNA and the simulation as in the test case. Also, the initial capacity and the sensitivity of the simulation greatly influence the outcome of the iterative method. Therefore, it is important to look at multiple initial capacities, as we do with random restart.

Overall we can conclude that the iterative method developed in this thesis is suitable for capacity allocation and it introduces a new way to use QNA in combination with optimisation. Moreover it addresses strict time requirements, whereas most capacity allocation methods in literature only look at the requirements for the expectation of the completion time. Therefore, the method is able to find a capacity allocation in situations where we are interested in more than the expectation, for example in a hospital setting.

5.2 Discussion and further research

During the research, we have noted several ways to improve and extend the iterative method. In this section we discuss these points and give recommendations for future research.

As we observed earlier, we have seen that the simulation is used more often when we approach a feasible capacity allocation and each time only one QNA phase is needed. We use QNA such that we do not need to check a lot of different capacity allocations with the simulation, since a simulation can take a long time. However, in this method we still need to run multiple simulations, which can result in a long running time for more extensive queueing networks with many possible routes for customers. This should be taken into account when using this method in other situations, but on the other hand it should be noted that capacity allocation does not need to be determined regularly. Therefore, a long running time is not a huge disadvantage for a capacity planning method.

To reduce the running time, one could consider to change the target time update after situation 2 has occurred in the simulation phase, in order to decrease the number of simulations used. We look here at the difference in expected waiting time between QNA and the simulation, while we are actually interested in the fraction of customers that exceeds the target time. The mean waiting time should give an idea about the general difference in QNA and the simulation, but not directly about the difference in fraction of customers that is supposed to be within the target time. A possibility would be to update the threshold (similar as for situation 3) instead of the target time for all requirements that are not yet satisfied. However, in that case we do not have a direct link between QNA and the simulation. Therefore, we could also look at the variance of the waiting time, in combination with the mean. Perhaps one could use the right hand side of Chebyshev's inequality for the sample mean and variance from the simulation and compare the result with the one of QNA. This difference can then be used to update the target time. These are two possible changes for the update in situation 2, but more research should be done regarding the practical use of both of them.

Due to the strict completion time requirements, the method is very careful when deciding whether a capacity allocation is feasible or not. Since we look at the upper bound of the confidence interval constructed by the simulation of the probability that anyone exceeds the target time, we could decline an already feasible capacity allocation. Therefore, the optimal capacity allocation given by the iterative method could turn out higher than what is actually needed. We also saw this in the test case when running the iterative method twice for the same initial capacity. The same initial capacity can lead to different optimal capacities, but due to the use of random restart we still come close to the theoretical optimum. This shows that the outcome of the simulation is a very important factor in defining the course of the iterative method. However, in the case of very strict requirements, we decided that it is better to add a bit too much capacity than ending up having too little.

Next, we would like to expand on the topic of multiple initial capacities. For this master thesis we chose to use random restart, since it is an easy method to implement and it still enables the method to start from different points within the stable capacity region and therefore to avoid ending up in a local minimum. However, there are also sophisticated methods to decide on different initialisations, which use the information obtained from the initial capacities already analysed to generate new initial capacities. We discussed a possible restart method in Section 4.2.4, which follows this idea. This would be a way to smartly choose the next initial capacity to be analysed and therefore reduce the number of initial capacities needed.

In this research, we aim to minimise the total sum of capacities. However, in some situations the costs of adding extra capacity can differ per station or for some stations it may be harder to find staff. In the objective in Equation (3.2a) we consider all stations to be equal. Hence, we recommend for further research to add weights to the capacity of each station to resemble the possible difference between stations. One should also incorporate these weights in the update rule at the end of the QNA phase of the iterative method.

Lastly, it would be interesting to test the iterative method on a larger network with more customer classes and types. The network we have considered in the ZGT case study is still quite small with only 8 stations, 1 customer class and 11 possible routes. Running the iterative method for a bigger network can give new insights in the behaviour of the method and running time. Also, we would advise to increase the number of initial capacities analysed, due to the different results we have seen for different initial capacities.

Bibliography

[1] G. R. Bitran and S. Dasu, "A review of open queueing network models of manufacturing systems," *Queueing systems*, vol. 12, no. 1, pp. 95–133, 1992.

[2] H. Kobayashi and A. Konheim, "Queueing models for computer communications system analysis," *IEEE Transactions on Communications*, vol. 25, no. 1, pp. 2–29, 1977.

[3] J. R. Jackson, "Networks of waiting lines," *Operations Research*, vol. 5, no. 4, pp. 518–521, 1957.

[4] J. R. Jackson, "Jobshop-like queueing systems," *Management science*, vol. 10, no. 1, pp. 131–142, 1963.

[5] F. P. Kelly, "Networks of queues," *Advances in Applied Probability*, vol. 8, no. 2, pp. 416–432, 1976.

[6] R. J. Boucherie and N. M. Van Dijk, *Queueing networks: a fundamental approach*. Springer Science & Business Media, 2010, vol. 154.

[7] K. M. Chandy and C. H. Sauer, "Approximate methods for analyzing queueing network models of computing systems," *ACM Computing Surveys (CSUR)*, vol. 10, no. 3, pp. 281–317, 1978.

[8] J. Banks, "Introduction to simulation," vol. 1, Feb. 2000, 9–16 vol.1.

[9] W. Whitt, "The queueing network analyzer," *The bell system technical journal*, vol. 62, no. 9, pp. 2779–2815, 1983.

[10] G. Bitran and R. Morabito, "Open queueing networks: Optimization and performance evaluation models for discrete manufacturing systems," *Production and Operations Management*, vol. 5, Feb. 1994.

[11] M. Reiser and H. Kobayashi, "Accuracy of the diffusion approximation for some queuing systems," *IBM Journal of Research and Development*, vol. 18, no. 2, pp. 110–124, 1974.

[12] P. Kuehn, "Approximate analysis of general queuing networks by decomposition," *IEEE Transactions on communications*, vol. 27, no. 1, pp. 113–126, 1979.

[13] M. Segal and W. Whitt, "A queueing network analyzer for manufacturing," *Teletraffic science for new cost-effective systems, networks and services*, pp. 1146–1152, 1989.

[14] G. R. Bitran and D. Tirupati, "Multiproduct queueing networks with deterministic routing: Decomposition approach and the notion of interference," *Management Science*, vol. 34, no. 1, pp. 75–100, 1988.

[15] R. Sadre and B. R. Haverkort, "Decomposition-based queueing network analysis with FiFiqueues," in *Queueing networks*, Springer, 2011, pp. 643–699.

[16] B. R. Haverkort, "Approximate analysis of networks of PH/PH/1/K queues with customer losses: Test results," *Annals of Operations Research*, vol. 79, pp. 271–291, 1998.

[17] H. Bhatia, R. Lening, S. Srivastava, and V. Sunitha, "Application of QNA to analyze the 'queueing network mobility model' of MANET," Technical Report, Dhirubhai Ambani Institute of Information & Communication, Tech. Rep., 2007.

[18] G. Schneider, M. Schuba, and B. R. Haverkort, "Qna-mc: A performance evaluation tool for communication networks with multicast data streams," in *Computer Performance Evaluation*, R. Puigjaner, N. N. Savino, and B. Serra, Eds., Berlin, Heidelberg: Springer Berlin Heidelberg, 1998, pp. 63–74.

[19] G. J. Heijenk, M. El Zarki, and I. G. Niemegeers, "Modelling segmentation and reassembly processes in communication networks," in *The fundamental role of teletraffic in the evolution of telecommunications networks: proceedings of the 14th International Teletraffic Congress, ITC 14, Antibes Juan-les-Pins, France, 6-10 June, 1994*, Elsevier, 1994, pp. 513–524.

[20] M. Zonderland, F. Boer, R. Boucherie, A. de Roode, and J. van Kleef, "Redesign of a university hospital preanesthesia evaluation clinic using a queuing theory approach," English, *Anesthesia and analgesia*, vol. 109, no. 5, pp. 1612–1621, 2009.

[21] E. Alenany and M. A. El-Baz, "Modelling a hospital as a queueing network: Analysis for improving performance," *International Journal of Industrial and Manufacturing Engineering*, vol. 11, no. 5, pp. 1181–1187, 2017.

[22] S. L. Albin, J. Barrett, D. Ito, and J. E. Mueller, "A queueing network analysis of a health center," *Queueing Systems*, vol. 7, no. 1, pp. 51–61, 1990.

[23] G. R. Bitran and D. Tirupati, "Tradeoff curves, targeting and balancing in manufacturing queueing networks," *Operations Research*, vol. 37, no. 4, pp. 547–564, 1989.

[24] C. Rogério, N. Silva, and R. Morabito, "Performance evaluation and capacity planning in a metallurgical job-shop system using open queueing network models," *International Journal of Production Research*, vol. 47, Aug. 2009.

[25] W. J. Hopp, M. L. Spearman, S. Chayet, K. L. Donohue, and E. S. Gel, "Using an optimized queueing network model to support wafer fab design," *Iie Transactions*, vol. 34, no. 2, pp. 119–130, 2002.

[26] D. Connors, G. Feigin, and D. Yao, "A queueing network model for semiconductor manufacturing," *IEEE Transactions on Semiconductor Manufacturing*, vol. 9, no. 3, pp. 412–427, 1996.

[27] Y. Takemoto and I. Arizono, "Production allocation optimization by combining distribution free approach with open queueing network theory," *The International Journal of Advanced Manufacturing Technology*, vol. 63, no. 1-4, pp. 349–358, 2012.

[28] R. Morabito, M. C. de Souza, and M. Vazquez, "Approximate decomposition methods for the analysis of multicommodity flow routing in generalized queuing net-

works," *European Journal of Operational Research*, vol. 232, no. 3, pp. 618–629, 2014.
[29] S. Creemers and M. Lambrecht, "Modeling a hospital queueing network," in *Queueing Networks: A Fundamental Approach*. Boston, MA: Springer US, 2011, pp. 767–798.
[30] S. van Brummelen, N. van Dijk, K. van den Hurk, and W. de Kort, "Waiting time-based staff capacity and shift planning at blood collection sites," *Health systems*, vol. 7, no. 2, pp. 89–99, 2018.
[31] F. R. Cruz and T. van Woensel, "Finite queueing modeling and optimization: A selected review," *Journal of Applied Mathematics*, 2014.
[32] L. Kerbache and J. Smith, "Multi-objective routing within large scale facilities using open finite queueing networks," *European Journal of Operational Research*, vol. 121, no. 1, pp. 105–123, 2000.
[33] A. Törn and A. Zilinskas, *Global Optimization*. Berlin: Springer, 1989, vol. 350.
[34] W. Krämer and M. Langenbach-Belz, "Approximate formulae for the delay in the queueing system GI/G/1," *Congressbook, 8th ITC, Melbourne*, pp. 235–1, 1976.
[35] J. Kingman and M. Atiyah, "The single server queue in heavy traffic," *Oper. Manag. Crit. Perspect. Bus. Manag*, vol. 57, p. 40, 2003.
[36] M. Capiński and P. E. Kopp, *Measure, Integral and Probability*. London: Springer, 2004, vol. 14.
[37] P. Welch, "Chapter 6 the statistical analysis of simulation," *The Computer Performance Modeling Handbook, Notes and Reports in Computer Science and Applied Mathematics*, vol. 4, pp. 268–328, Jan. 1983.
[38] H. Tijms, *Operationele Analyse*, Dutch, ser. 54. Epsilon Uitgeverij, 2004, tweede herziene druk.
[39] J. Walrand and P. Varaiya, "Sojourn times and the overtaking condition in Jacksonian networks," *Advances in Applied Probability*, vol. 12, no. 4, pp. 1000–1018, 1980.
[40] ZGT, "ZGT richting 2022. Onze keuze voor de toekomst," 2017.
[41] ZGT, *Jaardocument 2018*, "https://www.zgt.nl/media/19549/jaardocument-2018.pdf", Accessed on 23-01-2021, 2019.
[42] Stichting Oncologische Samenwerking, *Normeringsrapport*, "https://www.soncos.org/kwaliteit/normeringsrapport/", Accessed on 20-07-2021.
[43] Stichting Oncologische Samenwerking, "Multidisciplinaire normering oncologische zorg in Nederland," SONCOS, Utrecht, Tech. Rep. 9, 2021.

Appendix A

List of notation

List of notation used in this paper in order of first appearance.

J	:	Number of stations in the network.
K	:	Number of customer classes.
S	:	Number of customer types.
p_{ks}	:	Probability that a customer of class k is of type s.
$r(s, v)$	:	Station visited in the v^{th} stage of the route of customer type s.
$L(s)$	:	Number of stages of the route of customer type s.
$G_i^{(s,v)}$	:	Service time at station i for a customer s in stage v of his route.
$\gamma_0^{(s)}$	:	Arrival rate of customer type s at the first station of their route.
$\lambda_j^{(s,v)}$	:	Arrival rate at station j of customer type s in stage v of their route.
$\lambda_j^{(s)}$	:	Arrival rate at station j of customer type s.
β_i	:	Capacity, expressed in server speed, at station i.
$\mu_i^{(s,v)}$	:	Service rate at station i of customers of type s and in stage v of their route.
$B_i^{(s,v)}$	:	Adjusted service time at station i for customer type s in stage v of their route.
$\hat{\beta}$	:	Vector containing the capacities at each station.
$\mathcal{N}$	:	Set containing the completion time requirements.
$\tau_m^{(k)}$	:	Target time for requirement m and customer class k.
$x_m^{(k)}$	:	Threshold for requirement m and customer class k.
$\mathcal{W}_m^{(k)}$	:	Waiting time over requirement m for customer class k.
λ_j	:	Aggregated arrival rate at station j.
G_j	:	Aggregated service time at station j.
B_j	:	Adjusted and aggregated service time at station j.
ρ_j	:	Utilisation at station j.

$\hat{\beta}_{\text{stab}}$	:	Minimal capacity for which $\rho_i < 1$ for all stations i.
$\hat{\delta}$	:	Step size used to generate multiple initialisations for the capacity.
M	:	Number of initial capacities.
h	:	Number of times the step size $\hat{\delta}$ is added to a station to obtain the initial capacities.
λ_{ij}	:	Internal flow from station i to j.
λ_{0j}	:	External arrival rate to station j.
λ_{i0}	:	Departure rate from station i.
Q_{ij}	:	Proportion of customers travelling from station i to station j.
C_{0j}^2	:	Coefficient of variation of the external arrival process at station j.
C_{bj}^2	:	Coefficient of variation of the service process at station j.
C_{aj}^2	:	Coefficient of variation of the arrival process at station j.
W_j	:	Waiting time at station j.
Π_{W_j}	:	Probability of delay at station j.
D_j	:	Waiting time at station j given that the server is busy.
N_j	:	Number of customers at station j.
T_j	:	Sojourn time at station j.
Y_j	:	Number of subsequent visits to station j.
$b_m^{(s)}$	:	Begin stage of requirement m and customer type s.
$e_m^{(s)}$	:	End stage of requirement m and customer type s.
$\Omega(\hat{\beta})$	:	Neighbourhood of capacity allocation $\hat{\beta}$.
$U_{m,j}^{(k)}(\hat{\beta})$	:	Function for the derivative with respect to β_j of the expected waiting time over requirement m and customer class k.
Ψ	:	Set with all combinations of requirements and customer classes that are not satisfied.
$U_j(\hat{\beta})$	:	Function of the sum of derivatives $U_{m,j}^{(k)}(\hat{\beta})$ over all unsatisfied combinations of requirements m and classes k.
q	:	Window size of moving average method.
n_i	:	Number of observations in the simulation at station i.
z	:	Number of periods to look back at for the moving average method, in order to determine the end of the warm-up period.
$W_i(\ell)$	:	Waiting time at station i of the ℓ^{th} customer in the simulation.
ω	:	Parameter for the stopping criterion of the warm-up period in the simulation.
$X_m^{(k)}(\ell)$	:	Observation with value 1 if customer ℓ of class k does not satisfy the target time of requirement m and 0 otherwise.
$n_m^{(k)}$	:	Number of observations in the simulation of customers of class k and requirement m after the warm-up period has ended.

r	:	Number of batches for the Batch Means Method.
$\nu_{m,y}^{(k)}$	:	Batch size of the y^{th} batch of observations of customer class k and norm m.
$\overline{X}_m^{(k)}$	:	Mean of the Batch Means Method for observations $X_m^{(k)}(\ell)$.
$\overline{S}_m^{(k)}$	:	Sample variance of the Batch Means Method for observation $X_m^{(k)}(\ell)$.
$t_{r-1,1-\alpha/2}$	:	The $(1-\alpha/2)$-percentile of the Student's t-distribution for $r-1$ degrees of freedom.
$\hat{\beta}_{\text{cur}}$	:	Vector containing the capacities after a QNA phase.
$\hat{\beta}_{\text{prev}}$	:	Vector containing the capacities after a QNA phases minus the step size.
η	:	Update factor for the step size when in situation 1.
δ_{min}	:	Minimum step size for which the iterative method continues.
ε_2	:	Update error for the target time in situation 2.
$\text{CI}_m^{(k)}$	:	Upper bound of the confidence interval of the fraction of customers that has exceeded the target time in the simulation.
ε_3	:	Update error for the threshold in situation 3.

Appendix B

Input for iterative method in ZGT case

This appendix contains tables with the input data of the iterative method for the ZGT case study. In Section B.1 information about the care pathways of breast cancer patients can be found. The other input used for the iterative method is given in Section B.2.

B.1 Care pathways for breast cancer

The queueing network constructed for the ZGT case study consists of 8 stations, which can be found in Table B.1 together with the number we assigned to each station. From the data and meetings with different health care professionals, we divided the care pathway in three main sub-paths: diagnostics (D), treatment (T) and follow up (F), and two optional sub-paths: additional tests (A) and chemo therapy (C). The paths for each of these phases can be found in Table B.2. Only the used sub-paths are stated here, but originally we constructed more of them, which is why the numbering of the paths is not consistent. The red number states the number of days until the start of the actual treatment. Lastly, Table B.3 gives the complete patient routes considered for the ZGT case together with their arrival rates. The patient routes which are represented by a health care profession only visit the station corresponding to the profession.

Table B.1: Number assigned to each profession for stations involved in breast cancer.

Number	Profession
1	Breast surgeon
2	Nurse practitioner medical oncology
3	Medical oncologist
4	Breast doctor
5	Breast nurse
6	Day treatment nurse
7	Radiotherapist
8	Nurse practitioner breast

Table B.2: Breast patients' sub-care pathways.

Path	Description	Appointments (length in minutes)
D1	Diagnostics, benign	4 (10) - 4 (15)
D3	Diagnostics, malicious	4 (10) - 4 (15) - 5 (15) - 4 (15) - 1 (20) - 5 (20 + 20)
D5	Diagnostics, benign	1 (10) - 1 (15)
D7	Diagnostics, benign	1 (10) - 1 (15) - 1 (20) - 5 (20 + 20)
A1	Additional tests	1 (10) - 5 (10)
T1	Surgery	1 (30) - 5 (30 + 45) - 10 days - surgery - 1 (30) - 5 (30 + 45)
T2	Surgery, radiotherapy	1 (30) - 5 (30 + 45) - 10 days - surgery - 7 (30) - 1 (30) - 5 (30 + 45)
T3	Chemotherapy, surgery	3 (30) - 6 (60) - 10 days - C1 - 1 (30) - 5 (30 + 45) - surgery - 1 (30) - 5 (30 +45)
T5	Chemotherapy, surgery, radiotherapy	3 (30) - 6 (60) - 10 days - C1 - 1 (30) - 5 (30 + 45) - 7 (30) - surgery - 1 (30) - 5 (30 + 45) - radiotherapy
C1	Chemo appointments	3 (15) - 2 (20) - 3 (15) - 2 (20) - 3 (15) - 2 (20) - 3 (15)
F3	Follow up without chemo	5 (30) - 8 (15) - 1 (10) - 5 (10+30)
F4	Follow up with chemo	5 (30) - 8 (15) - 1 (10) - 5 (10+30) - 3 (15)

Table B.3: Arrival rates per patient type.

Type number	Patient route	Arrival rate (per hour)
1	D1	0.390
2	D3A1T2F3	0.073
3	D3T3C1F4	0.035
4	D4A1T1F3	0.321
5	D5	0.212
6	D7A1T1F3	0.034
7	D7T5C1F4	0.032
8	Nurse practitioner breast	0.620
9	Medical oncologist	1.429
10	Nurse practitioner medical oncology	0.593
11	Breast nurse	1.528

B.2 Other input for the ZGT case

Table B.4: Parameters for iterative method.

Description	Parameter	Value
Fractional step size used for initialisation of random restart	$\hat{\delta}$	1.05
Number of (distinct) initialisations	M	10
Number of stations to which initial step size is added	h	4
Starting step size for iterative method	δ	0.05
Update factor for step size	ν	2
Window size in the moving average method	q	10
Periods to look back at in moving average method	z	3
Stopping parameter of warm-up period	ω	0.05
Number of batches	r	25
Confidence interval percentage	α	0.05
Sufficiently small step size	δ_{min}	0.0125
Minimum number of departures per type in simulation	-	$5,000$