VRIJE UNIVERSITEIT AMSTERDAM

Master Thesis
February-July 2021

Simulation models to evaluate long term effects in the elderly care system in the Amsterdam region

---

Author:
Tim Rens de Boer

Supervisors:
First supervisor: Dr. R. Bekker (VU Amsterdam)
Second reader: Dr. A. Zocca (VU Amsterdam)
External supervisor: Prof.dr. R.D. van der Mei (CWI)
External supervisor: Drs. R. Arntzen (CWI)

Preface

This thesis is written as part of the master Business Analytics at the Vrije universiteit Amsterdam. The research is done during a six-month internship at Centrum Wiskunde & Informatica (CWI) in the stochastics group, starting the first of February until the first of August. CWI is a national research institute for mathematics and computer science in the Netherlands. Research topics at CWI include, but are not limited to, health care logistics, machine learning and transportation problems.

In the thesis the methods, models and results are discussed from research done on how to model and simulate the elderly care system in the Amsterdam region. Two models are constructed, the first model to simulate the system on a macro-level, without stochastic elements and the second model where stochasticity is introduced. The models can then be used to determine where bottlenecks occur in the system, how the system behaves under changing conditions and what the capacity should be.

I would like to thank dr. René Bekker, prof.dr. Rob van der Mei and drs. Rebekka Arntzen for all the feedback and ideas, which helped me with all the aspects of this thesis. Ideas and feedback given used for modeling during our weekly meetings or feedback on the different versions of my thesis. I also would like to thank the rest of the Dolce Vita group and CWI for their help and interest in the research project.

Abstract

The research done in this thesis will help the health care sector by modeling the elderly care system in the Amsterdam region. The development of a macro model is crucial in order to understand the dynamics of the system, the effect of policy changes, the behaviour of the system on the long-run and to gain insight into the occurrence of bottlenecks.

The macro view is used to construct two different models: (1) a model based on a system dynamics approach and (2) a discrete event simulation model. The systems dynamics model uses a deterministic recursion relation to determine the state of the system and is not dependent on stochastic elements. The second model is based on discrete event simulation, where the state of the simulation depends on stochastic elements. The following are randomly drawn from a distribution in this simulation: service times, arrival times and transfers. These models are then used to observe and study the behaviour of the system given various parameters.

A source for inefficiency in the elderly care system is the presence of deadlocks. Deadlocks occur when people waiting at one location, block other patients from entering service. This results in patients waiting unnecessarily, capacity that is not fully used and possibly a system that gets stuck, requiring manual interference to recover. This paper proposes a deadlock recovery and avoidance method based on a linear program and integer linear program, where patients are swapped simultaneously between locations. Such a swap may for instance resolve a situation where patients block each other, resulting in a recovery from the deadlock or avoidance of the deadlock. This method would require various locations to co-operate and share information about possible transfers.

As can be seen the contribution of this thesis is twofold: (1) by development of macro models that capture the dynamics of the elderly care system, insight is gained into the current system and possible future alternatives, and (2) an innovative patient relocating technique is proposed to increase bed efficiency in the elderly care system, of which the effects are evaluated using the macro models.

Keywords: System Dynamics, Queueing Theory, Discrete Event Simulation, Deadlocks, Health Care, (Integer) Linear Programming

Contents

1 Introduction **5**

2 Model description **10**
2.1 Arrival and service process 11
2.2 Priority between patient flows 12
2.3 Deadlocks 13

3 Preliminaries **14**
3.1 System Dynamics 14
3.2 Queueing theory 15
3.2.1 Network of Queues 15
3.3 Discrete Event Simulation 19
3.4 Comparison between System Dynamics and Discrete Event Simulation 19
3.5 Linear programming 20

4 System Dynamics: Methodology **23**
4.1 System Dynamics model 23
4.2 Priority 27
4.3 Determining flow levels 28
4.3.1 Determining flow with Linear programming (with coordination) 29
4.4 Multiple patients/customer types 32

5 System Dynamics: Model in equilibrium **34**
5.1 Equilibrium of the system of a stationary system 34
5.1.1 Inflow equals outflow 36
5.1.2 Queueing theory approach 36
5.1.3 Minimum capacities 39
5.2 Effect of time dependent parameters 40
5.3 The effect of starting stock on the system 43
5.3.1 Configuration: Current practice 43
5.3.2 Configuration: optimization model 51

6 Discrete event simulation: Methodology **53**
6.1 Deadlocks in DES 53
6.2 Model and pseudo-code 57
6.3 Validation 60
6.3.1 Validation of M/M/1 and M/M/C queues 60
6.3.2 Validation of a Jackson network 61

7 Results **63**
7.1 Practical case 63
7.2 Results: SD model 65
7.3 Results: DES model 70

8 Conclusion **76**

9 Discussion **77**

Appendices **79**

A Possible paths in the system **79**

B Queueing theory basics **79**

C Example of a system dynamics model **80**

D Tips & tricks for linear programming **81**

E Graphs of oscillating patient levels **83**

1 Introduction

The population of the Netherlands is aging [15]; the percentage of the population that has an age of 65 or older is increasing. Figure 1 shows the (expected) growth of the elderly population in the Netherlands, between 2000 and 2060. The grey area in figure 1 shows the growth of the population with an age between 65 and 79, while the darker area is the population older than 80. Moreover figure 2 visualizes the increase in life expectancy. The dotted line in this figure corresponds to the life expectancy of women and the solid line for men. The increasing life expectancy and growing elderly population lead to a bigger burden on health care in the Netherlands as older persons require more care. The problem is twofold as the problem requires the healthcare to scale up, as an older population requires more care, while the budget hardly grows. The acute geriatric care of Amsterdam is not functioning optimally due to these reasons.

In 2017 a report was published that showed bottlenecks occurring in the system of acute geriatric care [27]. Patients have to wait before they can move to a nursing home or can be admitted by the hospital. The health of these patients can then decline further while they need to wait, which results in patients that cannot return home or need more rehabilitation than was expected. This is one example of the many cases in which inefficiency in the system causes harm to patients.

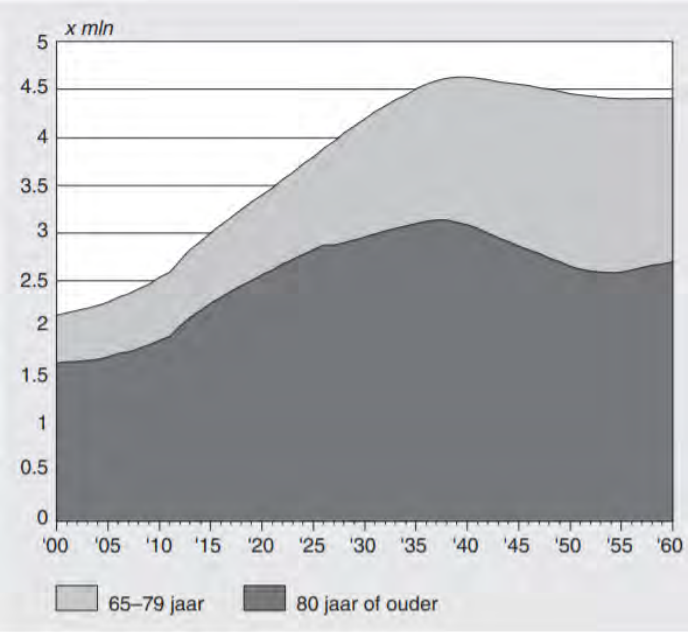


Figure 1: Expected growth in population of 65 years or older, taken from [15]

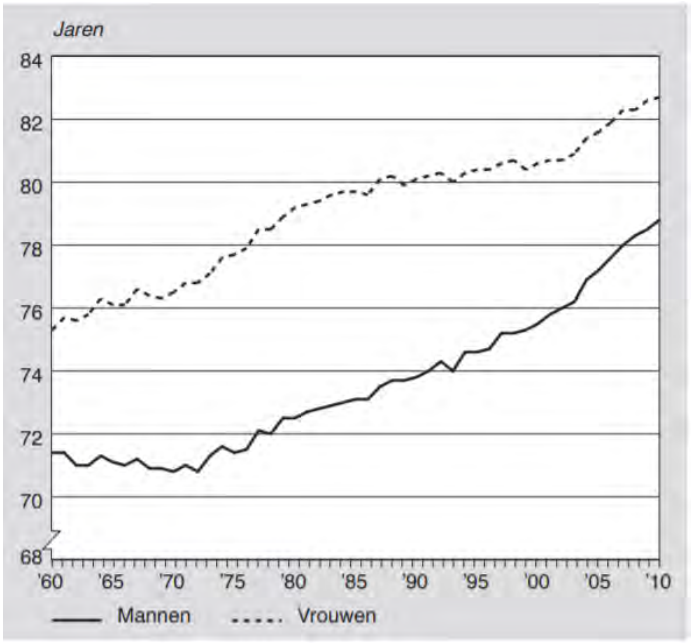


Figure 2: The increase in expected life expectancy, taken from [15]

The system described in the report 'Krakende Ketens' [27] can be found in figure 3. The complete system is quite complex as can be seen in the figure, since patients can go from almost any location to any other location. Most locations have a limited capacity concerning the number of patients that can be simultaneously present. Patients can only transfer to another location if there is capacity available at the destination location. The red lines show where, at time of publication of the report, the bottlenecks occur in the system. Red dotted lines show where no real bottlenecks were found, but here risks and areas of concern were noted. Green lines show where no bottlenecks were found and were not noted as areas of concern or risk. As can be seen most problems arise at transfer points between different care providers. Examples for different paths that patients can take in the system can be found in appendix A.

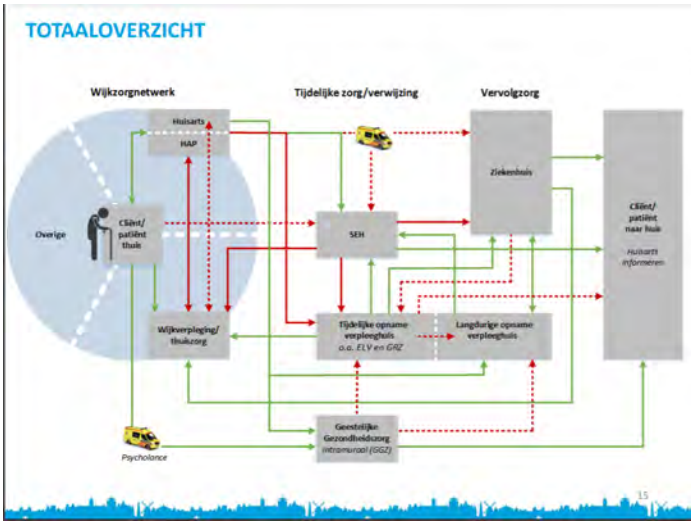


Figure 3: System sketch of care system of elderly patients in the Netherlands, taken from [27]

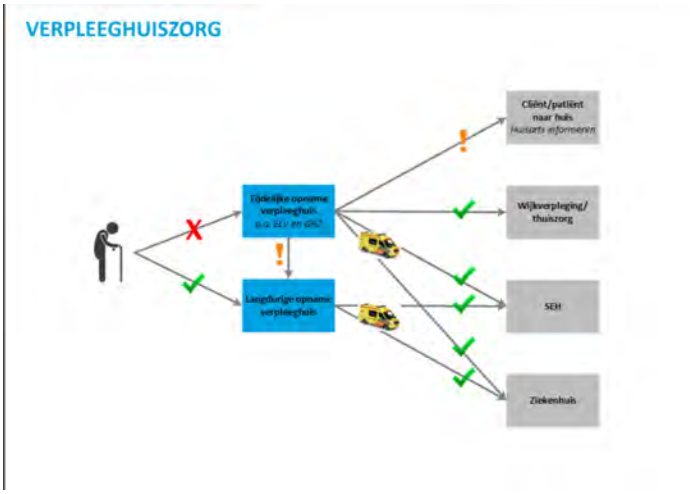


Figure 4: Example of a path an elderly patients can follow in the system [27]

Figure 4 shows an example of a path an elderly person can take in the system. As can be seen, bottlenecks occur when a patient transitions to temporary nursing home admission (in Dutch: Tijdelijke opname verpleeghuis). A temporary stay at a nursing home can be necessary if the patient requires only temporary care and does not require a hospital admission. Furthermore, there are two areas of concern: after a temporary stay at a nursing home back to home and to a prolonged nursing home stay. Some causes for these bottlenecks and areas

of concern are [27]:

- General practitioners (GP) do not have complete and/or recent information about the paths a patient can take, leading to unneeded or incorrect referrals.
- GP's not knowing the available capacity at the nursing home, while this is known at other locations such as the hospital.
- GP's not knowing all the options for referrals, leading to more than necessary hospital referrals.
- It is not clear if there is enough capacity for all the needed care at the nursing home.
- When a patient is admitted to the nursing home, it is often not known what kind of care the patient exactly needs. This makes it difficult to make sure that a patients is sent to the right location and receives the correct care.

The causes above show that in this case patients arriving at the nursing home are not always receiving the correct care. This is due to incorrect referrals or missing knowledge about the patient. Certain locations benefit from information about available beds at other locations, e.g. hospitals know the available capacity at the nursing home while GP's (who can send patients from home directly to the nursing home) cannot directly see the available capacity.

The main cause of these issues is an information mismatch, also resulting in a capacity mismatch, between care providers, such as different hospitals and nursing homes. This mismatch is mainly caused by the shortage of complete system views. A systematic view could provide every care provider with an insight into the causes of their bottlenecks, which may be dependent on other locations. This research aims to provide a system view to study the effect of policy and capacity changes on bottlenecks/waiting lists at transition points between care providers.

First, a systems dynamics approach is chosen. System dynamics distinguishes itself by focusing on a higher view of the system, where groups of patients are followed, instead of following each patient specifically. This makes it possible to simulate a larger system over a long time span, making it perfect to determine the effect of policy changes on a continually aging population. Secondly, another model is built using a discrete event simulation approach. In discrete event simulation each patient is followed as an individual, and each patient has a different path and time in the system based on random elements. In this model, the effect of stochastic elements is visible, since service and arrival times are randomly drawn. Both models are used to research the effect of possible deadlocks in the system, and how these can be detected, avoided or recovered

from.

Previous research of queueing and simulation models shows how effective simulation can be in optimizing various health care systems [24][20]. A paper written by Dangerfield et al. [13] has also shown how system dynamics can be used in a health care setting and how it helped hospital management by obtaining a clear overview of the system. Koizumi et al. [21] have introduced the possibility of blocking in a open queueing network to study the congestion and bottlenecks found in a system of mental health institutions. The inclusion of blocking meant that stream congestion could take place, where facilities with limited capacity could block a transfer, at which point this blocked transfer could block another transfer at their current node. The detection and recovery of deadlocks in a discrete event simulation model of a health care setting has been previously studied by Palmer et al. [25], the researchers used a motivating example of a system with a hospital and community care, where patients can block others from entering service. This research provided a deadlock detection and recovery method based on wait-for-graphs that can be constructed for a given state in the discrete event simulation model.

The paper contributes by providing macro models that can be used to understand the system of elderly health care system and what effects the future might have on the system. This research differs from previously done research by building these models from scratch to simulate the different aspects of the elderly health care system more precisely. The second contribution of this paper is a proposed method for allocating patients used to increase efficiency of capacity. This proposed method is based on the concept of deadlock detection and recovery, where deadlocks are recovered by using a linear program to determine the optimal transfers at specific times.

This thesis is structured as follows: section 2 introduces the model used in the rest of the paper, section 3 gives an overview about literature related to topics discussed in this paper. Section 4 and 5 are about the system dynamics model. In section 4 the model is explained, while in section 5 the various situations are evaluated in the model to obtain understanding about the macro-model. Section 6 described the discrete event simulation model used, followed by section 7 that gives a sketch and parameters used, similar to reality and the results that follow out of these parameters. The thesis concludes with section 8: the conclusion and section 9: the discussion.

2 Model description

The model used for this paper can be seen as a network of queues. In this network each node represents a location or a group of locations, where elderly patients stay to receive care. A network of queues is chosen, given the importance of the interaction of different locations. A visual representation can be seen in figure 5. This figure shows a network of two nodes (A and B), but in general the network could include any number of nodes. Each node has a queue without a maximum queue size (seen at 1. in the figure), followed by a queue with a maximum size (see 2. in the figure). Note that for this paper the queue in front is assumed to have a maximum size of zero, unless stated otherwise, to more closely resemble the elderly health care system. A patient waiting in one of these queues waits at home and his place in the queue is saved digitally. Each node has a maximum number of servers, after service each customer can either leave the system (see 4.) or request service from another node. A server a bed where the patient receives care, or a room in a nursing home or a hour of home care of a professional, it depends on the location of the server. Patients that require care from another node can immediately move to this other location if capacity is available at the destination node. Patients join the queue in front of the node (see 2.), if no capacity is available at the destination node. If the queue in front is also full the customer has to wait at his current node (see 5.). A customer still occupies a server, if he/she has to wait at his current node after service. This resembles a patient that still requires care after a hospital visit, but cannot yet move to the nursing home, he then needs to stay longer at the hospital.

Next, some terminology and symbols will be explained that will be used in the remainder of the paper. Each location is numbered $i \in I$, where I is the set of all locations, note that $|I| = n$, where n is the number of locations. The capacity C_i is the maximum number of patients that can receive care or wait at location i simultaneously. P and p capture the transfer percentages for which the definition is similar between the two models of this paper, but differs slightly. $P_{i,j}$ is used in the system dynamics model and denotes the percentage of patients want to leave i for care at j every timestep. The probability of a patient requiring care at location j after their service is completed at location i, $p_{i,j}$, is used in the discrete event simulation model, p can also be calculated using P. Note that $\sum_{j \in I} p_{i,j} \leq 1$, where $1 - \sum_{j \in I} p_{i,j}$ is the probability of a customer leaving the system after service at location i. The stock, S_i, is the number of patients in service at location i. The size of the waiting lists are denoted by $W_{i,j}$, which is the number of patients waiting at location i to go to location j. The number of patients waiting outside to enter service at node j can then be captured in $W_{0,j}$. The rest of this section will include assumptions made for this model and a definition of deadlocks.

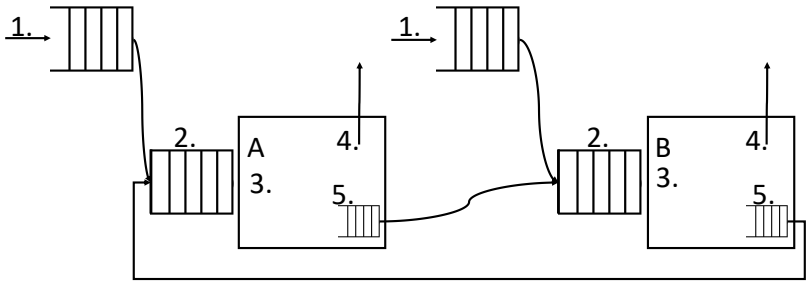


Figure 5: Illustration of the model with two nodes

2.1 Arrival and service process

The described model includes arrivals and departures to and from out of the system. These arrivals to and from outside the system can modeled in various ways:

- The outside population can be modeled as an extra node (node 0), but without a capacity. This is useful if outside arrivals and departures are heavily influenced by customers that are present in the network.
- The outside population can also be seen as a constant or determined by a formula, which is useful if the population mostly remains around the same size, or if the inflow is given by a parameter-dependent function. This function can be a sine function to represent a seasonal effect. In the rest of this paper the outside population will often be modeled by a constant or a given formula.

For queues it is often assumed that the interarrival and service times are exponentially distributed. In reality not all interarrival and service times are exponentially distributed. There is a possibility that some of these times more closely resemble another known distribution or should be drawn from a specific list of values. Service times and arrival times can be drawn using a specific distribution, list of values or a deterministic value. However, for the rest of this paper it is assumed that inter arrival and service times follow an exponential distribution, note that in case of system dynamics these arrivals and departures are set at the expectations of their distribution, instead of drawn from the distribution.

2.2 Priority between patient flows

Priority is an important aspect when capacity plays a role, as a patient can arrive at institute j from $n-1+1$ different flows. Where n is the number of institutes, 1 is subtracted, as patients cannot go from i to i directly and 1 is added to the possibility of arrivals from outside the system. Two different options are considered for priority setting. Absolute priorities, this means one source is prioritized above the other, this can be defined for each institute separately. The other option is wait list dependent priorities, longer wait list are prioritized. These priority settings are chosen since absolute priorities are frequently discussed in literature, but also flexible enough to approximate a real-life practical situation. Wait list dependent priority is chosen to provide a possibly more fair capacity allocation.

Absolute priorities

Absolute priorities means that patients from i are always prioritized over patients from j, such that patient from j can only transfer if there is no patient waiting from i. This can differ per institute, so the priorities will be presented in a matrix X. Where each row consist of integers from 0 to $n-1$ (or n if outside the system is included), where a lower integer means a higher priority. For example if $X = \begin{bmatrix} 0 & 1 & 2 \\ x_{2,1} & x_{2,2} & x_{2,3} \\ x_{3,1} & x_{3,2} & x_{3,3} \end{bmatrix}$, for this priority setting patients from location 1 to 0 has priority over patients from 2 to 0, see the first row, second and third column $x_{1,2} = 1 < 2 = x_{1,3}$, therefore a transfer from location 3 to 1 can only happen if the waiting list from location 2 to 1 is empty.

Wait list dependent priorities

This type of priority gives priority to patients from a long waiting list. Patients from one source could first be prioritized over all the others, while the next time all the others could be prioritized, depending on the size of the waitinglists. This can be seen as a fairer option than absolute priorities, since it is less likely that a patient has to wait indefinitely.

No priority

It is also possible that no priority setting is given. This would mean that no patient type is prioritized over the other. It might be helpful in optimization scenarios, where priorities might interfere with the optimal goal, for example keeping the waiting lists as small as possible. Note that this option is only possible when using optimization form with an objective to determine the best priority values.

2.3 Deadlocks

Deadlocks, also known as grid locks, is what happens when a subset of customer (or patients in this case) is blocked directly or indirectly by customers in that subset. The definition of a deadlock given by [25]:

Definition 2.1. When there is a subset of blocked customers who are blocked directly or indirectly by customers in that subset only, then the system is said to be in *deadlock*.

An example of a deadlock can be seen in figure 6. Patients in the hospital wait for community care and patients in the community care wait for a bed in the hospital, resulting in a deadlock if the capacities are fully used [25]. This results in a suboptimal state, since patients that are waiting at the wrong node, receive the wrong care, at perhaps a higher cost. The following can be concluded from the example: deadlocks are a result of blocking. If a patient is blocked due to no available capacity, he/she then remains at the node and blocks possibly other patients from entering this node. This problem is inherent to simulation, as in real-life situations ad hoc decision making is common for solving these kind of problems. This creates the following strategy for handling deadlocks in a simulation model:

1. Identify/detect deadlocks in a system
2. Recover from a deadlock such that the simulation can move on as intended.

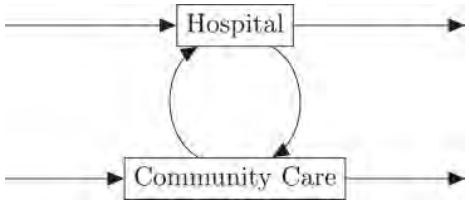


Figure 6: Example of a situation where a deadlock might arise

3 Preliminaries

Many studies have been conducted on how to model different systems in health care. Examples are queueing theory in staff scheduling [24], discrete event simulation for more complex systems and system dynamics to test how different health care systems behave when policies change. This paper aims to use system dynamics in a system setting where capacities play an important role in how the system behaves. First, system dynamics is discussed, which is a simulation technique where the focus is on large scale system and long term effects, which can be useful to see the effect of an aging population. Next, queueing theory is discussed to understand the theory behind this system, for the basics of systems where queues play an important role. Discrete event simulation is also discussed, as this provides a simulation possibility for systems too complex to analyze using queueing theory. Discrete event simulation can also be used as comparison with system dynamics.

3.1 System Dynamics

System Dynamics is a mathematical modeling technique developed during the 1950's to help managers obtain more understanding on industrial processes. Nowadays it is used in many fields to obtain understanding from a strategic level. Primary elements of a system's design are feedback, flows into stocks and delays. Feedback is the effect a change in one flow or stock has on another flow or stock. Flows is the flow from one stock to another or from outside of the system to inside the system or the other way around. Delay is the time it takes for an effect to take place. Finally stock is the amount or level of a specific entity.

The building process of a system dynamics model can be dividend in two different parts, since there are two different kind of models: causal loop diagrams and stock and flow models. Causal loop diagrams captures the interaction between different variables, an increase in one variable could mean a direct or indirect increase/decrease of another variable. This is done by drawing arrows between different variables, not these variables and effects are not quantified and the arrows only show the relation between the variables. Causal loop diagrams are often used to gain insight into the structure of the system, however it lacks precision. The size of the different effects in this causal loop diagram are often hard to see, and therefore it is hard to determine the behaviour of the system. The stock and flow model provides more precision by quantifying the different variables and effects. This starts by identifying different stocks and flows, what objects are important to keep track of in the system. These stocks could for example be the number of people in a hospital or the gallons of water in a bucket. Flows are the interaction between different stocks, where a decrease in one stock, means a increase in another stock, this interaction could be direct or with a delay. The size of these stock and flow can also be dependent on variables that can neither be defined as stock or flow. A stock and flow model for the amount of water in a bucket could have the position of the tap as important

variable. The strength of system dynamics is utilised by simulating the stock and flow model, this is done in discrete time or continuous time, where stocks, flows and variables are calculated using a list of equations. Note that this differs from other simulation techniques such as discrete event simulation, since running the same model twice will result in the same outcome (stochasticity does not play a part in the simulation of system dynamics) [28].

Various researchers have shown the use of System Dynamics in health care issues. Dangerfield et al. [13] have shown that research has been done on the use of system dynamics on various different health care scenarios, such as dental care in the Netherlands, where a large model was built for the system, which was then used to help different clients understand the model. Another situation used in the paper was one where hospital management was researching a new beds management, but had the tendency to "fight fires", solve the symptoms and not the cause. Again, System Dynamics helped by providing a model with a larger scope. For the model they built see appendix C [30]. This paper aims to expand on the previous done research by including the possibilities of deadlocks in the SD model and a direct comparison between the SD model and a simulation model with stochasticity.

3.2 Queueing theory

Queueing theory covers the study of queueing models, often used to describe real life stochastic situations. Examples include but are not limited to health care [24], traffic [19] and airport security [17].

Queueing theory is often applied in health care to help decision making resulting in lower waiting times and lower overall costs. Queueing theory can be used in reducing medical costs that are on the rise, due to a increasing life expectancy and aging population [24]. Applications of queueing theory in health care are abundant, examples include but are not limited to:

- Resource scheduling, such as: staff, bed and rooms managment
- System design of various elements, for example accident and emergency design.
- Analysis of abandonment, variable arrival rates, queue disciplines and limited capacities, to name a few.

This paper uses the basics of queuing theory to gain insight into the working of the elderly health care system. For more theory about the basics of queueing theory see appendix B.

3.2.1 Network of Queues

Many stochastic situations can be modeled using queueing theory with one queue, whereas multiple real-life situations cannot be modeled correctly and

accurately by a single queue and require a system of more than one queue, such as the system of elderly health care. This is where a network of queues is often used, ranging from tandem queues to more complicated network of queues. A tandem queue is perhaps the simplest non-trivial system of queues [23], see figure 7 (in this figure the nodes can be seen as queue and servers in one, so the node of B contains a queue and servers).

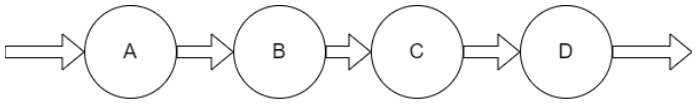


Figure 7: Example of a tandem of queues

Jackson networks are often seen as more complex, since a customer is not limited to moving forward in the system. For an example of a visual representation see figure 8. A network of queues is called a Jackson Network if it satisfies all of the following [23]:

- The network has N single-station queues.
- The ith station has s_i servers.
- The waiting room at each station has infinite capacity.
- Customers outside the system arrive at station i according to PP(λ_i).
- Service time at station i for all customers are i.i.d. Exp(μ_i).
- Customers finishing service at station i join the queue at station j with probability p_{ij} or leave the network altogether with probability r_i, independently of each other. Note that $r_i + \sum_j p_{ij} = 1$, meaning that after service all customers either leave the system or move to a station to receive service.

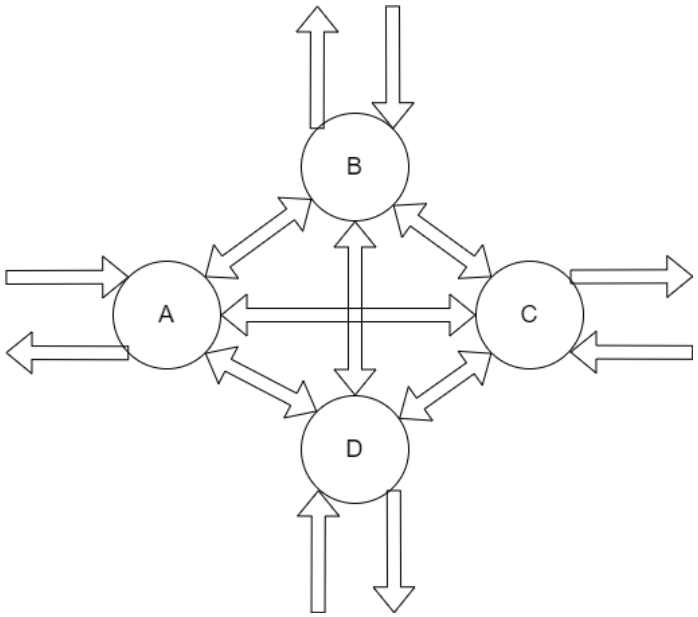


Figure 8: Example of a network of queues, where each node is a queue with servers

Jackson networks have infinite waiting room capacity, therefore the system may become unstable. If all service stations are stable, then traffic equations can be used:

$$a_j = \lambda_j + \sum_{i=1}^{N} a_i p_{ij}, \quad 1 \leq j \leq N \tag{1}$$

This can be rewritten in matrix form as:

$$a(I - P) = \lambda \tag{2}$$

The system is then stable if $(I - P)$ is invertible and

$$a_i < s_i \mu_i, \tag{3}$$

where s_i is the number of servers at station i, μ_i is the service rate at station i and a_j is the total arrival rate at station j.

Assuming a stable system the limiting behaviour of the network, X_i, can be determined, where X_i is the number of customers at station i. This can be combined to X to describe the system, where X is a vector consisting of X_i for all stations. So $X = [X_1, X_2, ...X_N]$, where N is the number of service stations in the network. Calculations and proofs described in [23] show that eventually the limiting distribution of a stable network can be written in product form and is given by:

$$p(n_1, n_2, ..., n_N) = p_1(n_1)p_2(n_2)...p_N(n_N), \tag{4}$$

where:

$$p_i(n) = p_i(0)\rho_i(n), i \geq 0, \tag{5}$$

$$\tag{6}$$

where $p_i(0)$ is given by the following:

$$p_i(0) = [\sum_{m=0}^{s_i-1} \frac{1}{m!}(\frac{a_i}{\mu_i})^m + \frac{(a_i/\mu_i)^{s_i}}{s_i!} \cdot \frac{1}{1 - a_i/(s_i\mu_i)}]^{-1}, \tag{7}$$

and $\rho_i(n)$ is given by:

$$\rho_i(n) = \begin{cases} \frac{1}{n!}(\frac{a_i}{\mu_i})^n, & \text{if } 0 \leq b \leq s_i - 1 \\ \frac{s_i^{s_i}}{s_i!}(\frac{a_i}{s_i\mu_i})^n, & \text{if } n \geq s_i \end{cases}$$

In this section only a Jackson network is described with the mentioned requirements. However, this can be extended to various other networks for which a product-form equilibrium distribution exists. These networks are called BCMP networks. A network is a BCMP network if the N queues are each one of the following four types [4]:

- The queue has a first-come-first-served discipline. All customers types in the system have service times drawn from the same exponential service time distribution. The service rate can be state dependent, so the service rate is $\mu(l)$ when the queue has a length of l.
- Processor sharing queues.
- Infinite server queues.
- The queue has a last-come-first-served discipline with pre-emptive resume.

Next to that the following conditions must hold:

- External arrivals form a Poisson process.
- After service completion the customer will go from i to j with probability p_{ij} or leave the system with $1 - \sum_{j=1}^{N} p_{ij}$

As can be seen for many different networks of queues a product-form equilibrium distribution can be found. There are also some situations for which the network is not a BCMP network and therefore does not have a product-form equilibrium distribution, for example, when a queue has a maximum capacity. In this case DES (discrete event simulation) provides a solution, see section 3.3

3.3 Discrete Event Simulation

As mentioned in section 3.2.1 discrete event simulation provides a solution for queues or network of queues that are hard or impossible to solve analytically. Applications range from traffic [8] to health care [20], and many more.

Discrete event simulation is a popular modeling technique [26], as it is flexible compared to queueing theory and is faster than its counterpart continuous simulation. Discrete Event Simulation differs from continuous simulation in how the simulation is ran. In continuous simulation, changes to the state of the simulation are made continuously by using differential equations. In DES, the system stays the same until the next event. For example: if congestion at an emergency department (ED) is modeled using DES, then the state would be the number of patients present, whereas events are newly arriving patients, abandoments, and patients that completed treatment at the ED.

For the same reason that queueing theory was used in health care, DES is used. The cost of health care was rising and some aspects of health care needed to be optimized to reduce costs [20]. DES could be used to focus more on complicated systems than queueing theory. Examples are similar to the examples given in section 3.2, for example the scheduling of staff and other resources, but also admission rules, i.e. what patients to admit when and where.

3.4 Comparison between System Dynamics and Discrete Event Simulation

Simulation is often used to model and analyze various complex and dynamical systems. Discrete event simulation (DES from now on) is often preferred, as one can follow the path of the entity in detail. Possible systems are for example: health care, modeling the journey of the patients, call centers, pandemics and many more. System dynamics (SD from now on) is less used, however, system dynamics can provide a macro-view of the system, showing the effect of parameter changes on the entire system. The differences between DES and SD are summarized in table 1, and while other uses for DES and SD are also possible, the table shows what each method is most used for [7].

	DES	**SD**
Scope	Operational, tactical	Strategic
Importance of variability	High	Low
Importance of tracking individuals	High	Low
Number of entities	Small	Large
Control	Holding(queues)	Rates(flows)
Relative timescale	Short	Long
Purpose	Decisions: optimisation, prediction and comparison	Policy making: gaining understanding

Table 1: Comparison of DES against SD [7]

As can be concluded from table 1, DES is best used for operational decisions, such as day-to-day management of departments, while SD can be best used to provide insight on the effect strategic decisions have on the state of the system.

3.5 Linear programming

Linear programming is included in the preliminaries, since the method for solving deadlocks in the system is based on a LP. The definition of a linear programming problem is either minimizing or maximizing a linear function adhering to given linear constraints (equality's or inequalities)[14]. This would result in the following format:

$$min/max \quad c^t x \tag{8}$$

$$s.t. \quad Ax \geq or \leq B \tag{9}$$

Where x are the decision variables, these are the variables to be changed to obtain the optimal solution. A and B are constants and A a matrix and B a column vector. Finally $c^t x$ describes the objective function which to maximize or minimize, in scalar form: $c_1x_1 + c_2x_2 + ... + c_nx_n$, where c_i's are constants. $Ax \geq or \leq B$ describes the constraints, in expanded form:

$$\begin{aligned} a_{11}x_1 + a_{12}x_2 + ... + a_{1n}x_n \geq or \leq b_1 \\ \vdots \\ a_{n1}x_1 + a_{n2}x_2 + ... + a_{nn}x_n \geq or \leq b_n \end{aligned}$$

There are various strategies to solve a LP(ranging in speed) and also various commercial solvers available to solve LP's. One of the most well known is perhaps simplex, this boils down to searching for the optimal solution in all the corner points of the feasible region.

Integer Linear Programming

Integer linear programming is a variation of linear programming where additional constraints are added, namely that the decision variables need to be integer. It is also possible that some of the decision variables need to be integer, which is known as mixed integer linear programming. The reason that integer linear programming is different from linear programming becomes clear when looking at an example and its graphical counterpart. The following LP problem is used:

$$\begin{aligned} max \quad & 2x_1 + x_2 \\ s.t. \quad & x_1 + x_2 \leq 10 \\ & 2x_1 \leq 15 \\ & x_2 \leq 8 \\ & x_1, x_2 \geq 0 \end{aligned}$$

The visual counterpart can be seen in figure 9, where each line represents the border of a constraint, and the blue area represents the feasible region of this problem. The optimal solution will be found at one of the following points, the corner points: (0,0), (0,8), (2,8), (7.5,2.5) and (7.5,0), with objective values 0, 8, 12,17.5 and 15. It is easy to see that the optimal solution is $(x_1, x_2) = (7.5,2.5)$. However, when the following constraint is added: $x_1, x_2 \in \mathbb{Z}$, the feasible region is reduced to the following points visible in figure 10. As can be seen, the previous optimal solution is unobtainable. If all the corner points of the feasible region would be integer, then problem can be solved similar to that of its LP counterpart. In many cases the corner points are not integer and therefore these (mixed) integer linear programming problems require another solving strategy. One of these strategies is called branch & bound [22], here first the bounds are set. The upper bound is equal to the optimal solution of the LP without integer constraints (also called the LP-relaxation), and the lower bound is set at $-\infty$, in a minimization problem the lower bound is set according to the LP relaxation and the upper bound is set at ∞. Next, branching takes place, which means that the feasible region is split, and each branch is solved independently. This branching process stops when the upper bound is reached or all the branches result in integer optimal solution or infeasible solutions. Using the branch & bound method the optimal solution of $(x_1, x_2) = (7,3)$ can be found.

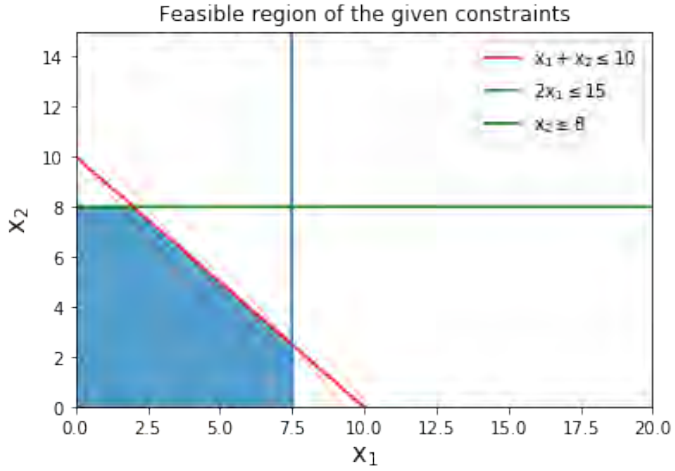


Figure 9: Feasible region of the given constraints

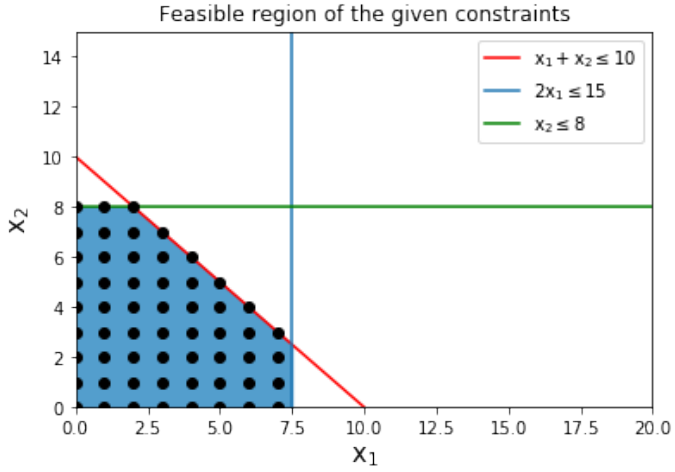


Figure 10: Feasible region of the given constraints and added integer constraint

Tips & tricks

There are various trick to model non-linear features in a linear way, such that it can be used in LP's and ILP's. Some of these useful tricks will be explained in section D, namely how to model: absolute values, min-max objective, either-or constraints and conditional constraints [6].

4 System Dynamics: Methodology

To describe the system of elderly health care in the Netherlands various options were considered, such as a model based on a system dynamics approach. This section will describe how this model is built and how it functions. This system will then be used in section 5, in which the behaviour will be analyzed using the following system configurations:

1. Flow is determined using the determination method where the previous timestep is used. Absolute priorities are used, and flows within the system are given priority over flows from outside the system. This configuration will from now on be called configuration current practice, as it most resembles the current practice. If needed the configuration where priorities are given for flows oustide the system are also given.

2. Flow is determined using the basic LP, without any priorities given. This configuration will from now on be called configuration optimization model.

4.1 System Dynamics model

The first model built for this project is a model based on the SD approach. This started by identifying the flows and stocks. The stocks in this case are the number of patients at different locations and the waiting lists to go from one location to the other. Flows are the number of patients going from one stock to another stock. The system is simulated in discrete time, meaning that at each timestep the flow is determined, based on what is already known at the current timestep and possibly what can be known in coordination with other locations. These determined flows are then used to determine the stock levels for the next timestep.

Parameters, variables and formulas

Important variables in the system include stock level, waiting lists, flow level, fresh flow level and available capacity. Stock level is the number of patients that are receiving service at a location, so $S_i(t)$ is the number of patients that are in service at location i at time t. Waiting lists are the patients that are waiting to go from one location to another, so $W_{i,j}(t)$ is the number of patients waiting to go from location i to location j at time t. Flow level, $F_{i,j}(t)$, denotes the number of patients that transfer from i to j between time t and $t+\Delta t$. Fresh flow level is similar to flow level, but instead of the number of patients that actually transfer from i to j, it denotes the patients that just completed service and want to go to another location. Thus $FW_{i,j}(t)$ is the number of patients that just completed service at i and want to go to j. Using the definitions of $W_{i,j}(t)$,$F_{i,j}(t)$ and $FW_{i,j}(t)$, it is possible to say that $F_{i,j}(t) \leq W_{i,j}(t) + FW_{i,j}(t)$, since it is not permitted to move patients that have not finished service yet. Finally, available capacity is denoted by $A_{i,j}(t)$ and is the available capacity at location j for patients from i at time t. Therefore, the following must hold: $F_{i,j}(t) \leq A_{i,j}(t)$.

The following formulas are used to represent the system. The formulas are determined with the assumption of absolute priorities and limited coordination between locations. Absolute priorities is when flow with a higher priority needs to have no waiting patients before a flow with a lower priority can happen. Limited coordination means that the flow is determined purely on the state of the system at the previous timestep (locations do not share information about transfers).

$$FW_{i,j}(t) = P_{i,j} * S_i(t) \tag{10}$$

This formula defines how many fresh patients want to go from i to j, where fresh can be seen as patients that just finished service and are not already waiting. This is calculated by the multiplying the percentage of patients that go from i to j, $P_{i,j}$, and the number of patients that are not waiting at i at time t, $S_i(t)$.

$$A_{i,j}(t) = C_j - S_j(t) - \sum_{k \in I} W_{j,k}(t) - \sum_{k \in I : x_k < x_j} F_{k,i}(t) \tag{11}$$

This formula can be seen as a placeholder for later, since this will be used only in determining the flow. This flow determines how much flow can go from i to j. This is done by subtracting the capacity of j, C_j, with the stock at j at time t, $S_j(t)$, which is then subtracted by the patients that are waiting at j to go to any other location at time t, $\sum_{i \in I} W_{j,i}(t)$. Finally, this is subtracted by the flows that have a higher priority, which are using capacity that cannot be used by the flow from i to j, $\sum_{j \in I : x_k < x_j} F_{j,i}(t)$. As can be seen for this formula absolute priorities are used, meaning if the flow from i to j has priority over k to j, then the waiting list of i to j needs to emptied before any flow from k to j can be sent. The priority setting can differ per location, so the priorities will be presented in a matrix X.

$$F_{i,j}(t) = min(A_{i,j}, FW_{i,j}(t) + W_{i,j}(t)) \tag{12}$$

In this formula the flow is determined. The flow from i to j is restricted by two values, namely the available capacity for patients from i at j, $A_{i,j}$, and how many patients want to go from i to j, $FW_{i,j}(t) + W_{i,j}(t)$. As can be seen, the flow is determined based on the state of the system at time t, therefore it is possible that some capacity is unused, as the actual availability may increase if patients also leave the destination location. Important to note is that for equation (11), the values of equation (12) with a lower priority, need to be known. Therefore, the order in which these formulas are used is important and will be later elaborated in the form of pseudo-code.

$$W_{i,j}(t + \Delta t) = W_{i,j}(t) + FW_{i,j}(t) - F_{i,j}(t) \tag{13}$$

$$S_i(t + \Delta t) = S_i(t) + \sum_{j \in I} F_{j,i}(t) - \sum_{j \in I} FW_{i,j}(t) \tag{14}$$

These formulas are used to update the levels of stock, $S_i(t+\Delta t)$, and the waiting lists, $W_{i,j}(t+\Delta t)$, for the next timestep. The new value for the waiting list is the level of the waiting list at the previous time, $W_{i,j}(t)$, plus the new patients that want to go from i to j, $FW_{i,j}(t)$, minus the patients that actually go from i to j, $F_{i,j}(t)$. The new value of stock is the old level, $S_i(t)$, plus all the incoming flows, $\sum_{j\in I} F_{j,i}$, and minus all the patients that want to leave, $\sum_{j\in I} FW_{i,j}$. These patients might actually still be at i, but at one of the waiting lists. It is also possible to find out how many patients are at i, which can be done by summing all the waiting lists at i and adding the stock level of i.

The parameters, variables and formulas are summarized in the following table. In the table it is also noted if the value of the variable is determined exogenous (given outside the model) or endogenous (determined within the model):

Name	**Symbol**	**Endogenous or exogenous**
Capacity	C_i	Exogenous
Percentage that wants to go from i to j	$P_{i,j}$	Exogenous
begin time, end time and Δt	-	Exogenous
Stock level	$S_i(t)$	Endogenous (only starting levels are given)
Waiting list	$W_{i,j}(t)$	Endogenous (only starting levels are given)
Flow level	$F_{i,j}(t)$	Endogenous
Fresh flow level	$FW_{i,j}(t)$	Endogenous
Available capacity	$A_{i,j}(t)$	Endogenous

Table 2: Summary of all the variables and parameters in the system

Note that in some situations parameters might depend on the current time of the system or on some other stock levels. This can be added to the system by seeing the parameters as functions, for example the capacity C_i. The value of C_i might increase per timestep, then $C_i = a$, where a was a constant can be changed to $C_i = a + b * t$, where b is the increment of the capacity per timestep. This can be done for all the other parameters as well. The order in which these formulas and variables are used will be elaborated in the following pseudo-code:

Initialize the values of C,P,begin time, end time, Δt, $S(0)$, $W(0)$ and t=begin time;
while $t + \Delta t \leq$ *end time* **do**
  for $i \in I$ **do**
    sort I based on the priorities of location i (found in row $x_{i,\cdot}$),this sorted list can be saved in J;
    for $j \in J$ **do**
      $FW_{i,j}(t) = P_{i,j} * S_i(t)$;
      $A_{i,j}(t) = C_j - S_j(t)$
      $- \sum_{i \in I} W_{j,i}(t)$
      $- \sum_{k \in I : x_k < x_j} F_{k,i}(t)$;
      $F_{i,j}(t) = min(A_{i,j}, FW_{i,j}(t) + W_{i,j}(t))$;
      $W_{i,j}(t + \Delta t) = W_{i,j}(t) + FW_{i,j}(t) - F_{i,j}(t)$;
    $S_i(t) + \sum_{j \in I} F_{j,i}(t) - \sum_{j \in I} FW_{i,j}(t)$
  $t = t + \Delta t$

In section 4.2 and 4.3 other options are discussed for priorities and flow determination.

A visual representation of a simple 2-location model as can be seen in figure 11. There are two options for patients that want to move from *Home* to *Hospital*. These patients can either move directly to the target location (*Hospital* in this case) or move to the waiting list. The capacity determines how many patients can move from *Home* to *Hospital*; if there is no capacity left then the patients have to go to the waiting list. Patients in the waiting list still use capacity of their location, so patients waiting on the waiting list from *hospital* to *home* still use the capacity of the *hospital*. This model can easily be expanded to n locations to represent a more complex and realistic system. Later visual representation of the model will often be abbreviated to the representation seen in figure 12, which is the same model as figure 11, the waiting lists are only included in the box of the location. These waiting lists however are still present and function the same as in figure 11.

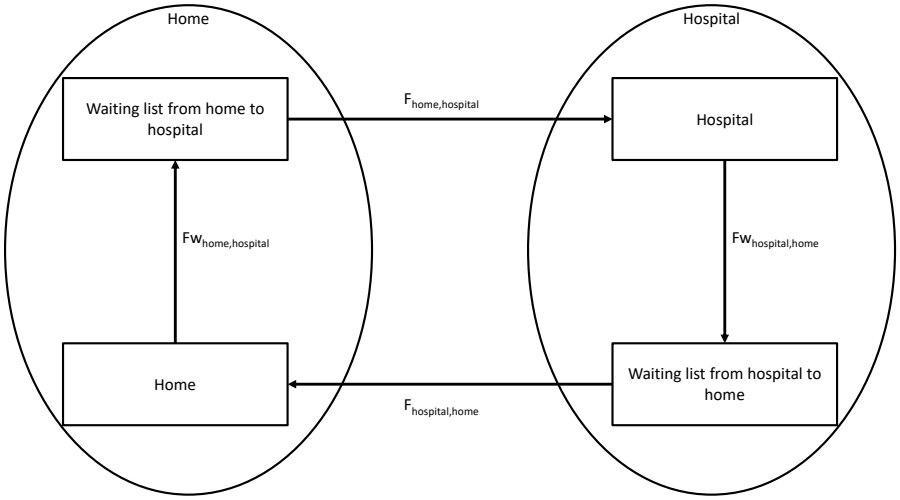


Figure 11: Visual representation of 2-location model

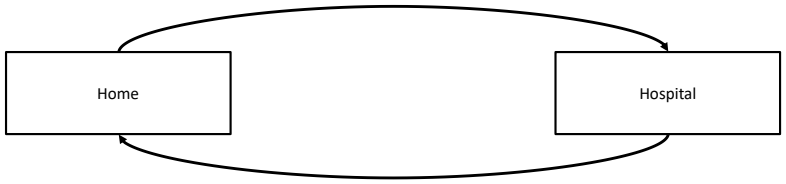


Figure 12: Abbreviation of the model shown in figure 11

4.2 Priority

This section will discuss what changes happen to the formulas if another priority setting is chosen. Absolute priorities are already discussed in section 4.1, so this section will only discuss waitlist dependent priorities.

Wait list dependent priorities

This type of priority gives priorities to flows with a long waiting list. For example, if an institute has 3 possible sources of flow, and each of the sources has a waiting list, then each flow is the maximum available flow times the fraction of the waiting list of that institute compared to the total waiting list. This can be seen as a fairer option than absolute priorities, since every source locations can use some capacity of the target location, therefore it will not happen that patients from one location have to wait for multiple timesteps, while patients from other locations use all the capacity. First, the total number of patients that want to go to location i have to be calculated, which is done by summing the waiting lists and the fresh flows from every other location to i. The formula can be seen in equation 15, where TW_i is the total number of patients that want to go to location i.

$$TW_i = \sum_{j \in I} (FW_{j,i}(t) + W_{j,i}(t)) \tag{15}$$

Next, the fraction of patients wanting to go from j to i of the total wanting to go to i have to be determined, see equation 16.

$$FR_{j,i} = \frac{FW_{j,i}(t) + W_{j,i}(t)}{TW_i} \tag{16}$$

Then, equations 11 and 12 are altered from section 4.1 to equations 17 and 18

$$A_j = C_j - S_j(t) - \sum_{i \in I} W_{j,i}(t) \tag{17}$$

$$F_{i,j}(t) = min(A_j * FR_{i,j}, FW_{i,j}(t) + W_{i,j}(t)) \tag{18}$$

As can be seen, $A_{i,j}$ is changed to A_j, as it now represents the available capacity at j, and instead of $A_{i,j}$ in equation 12, $A_j * FR_{i,j}$ is used in equation 18, as the flow from i to j can now only use a fraction of the total availability A_j. If $F_{i,j}(t) = FW_{i,j}(t) + W_{i,j}(t)$ (the second argument of the min function), then capacity is unused at j. It could be argued that this capacity should then be used by another flow. However, this is not necessary. Assume there are two source locations i and j and one target location k, assume that $F_{i,k}(t) = FW_{i,k}(t) + W_{i,k}(t)$, thus $A_k * FR_{i,k} > FW_{i,k} + W_{i,k} = TW_k * FR_{i,k}$, resulting in the observation that $A_k > TW_k$ and the fact that $A_k * FR_{j,k} > TW_k * FR_{j,k} = FW_{j,k} + W_{j,k}$. It can be concluded that if $F_{i,j}(t) = min(A_j * FR_{i,j}, FW_{i,j}(t) + W_{i,j}(t)) = FW_{i,j}(t) + W_{i,j}(t)$, then all the flows to j are determined by $F_{i,j}(t) = FW_{i,j}(t) + W_{i,j}(t)$, for $\forall i \in I$.

4.3 Determining flow levels

Two different options, without coordination and with coordination, are considered for the determination of flow from an arbitrary i to an arbitrary j, in the formulas noted as $F_{i,j}(t)$. The flow level can be determined solely by the state of the system at the previous timestep, also known as determining the flow

without coordination. For example, if location i has a capacity of 100, and at time t has 90 patients, then at most 10 new patients can arrive at i from other locations. Priority settings needs to be given either absolute priorities or waiting list dependent priorities to determine which 10 patients can be transferred. This method assumes the worst case that no patient leaves the target location, which is possible. One can immediately see that, while this option makes sure that there will never be more than 100 patients at location i, it is not optimal, since it is also possible that patients leave i, making total arrivals higher than 10 possible.

When looking at figure 12, it can be seen that these flows are linked. A flow from home to hospital makes a larger flow from hospital to home possible. Therefore, the flow determination can also be seen as an optimization problem, where constraints are given by the capacities and more constraints can be added to specify different policies, also known as determining the flows with coordination. These two different methods can be seen as situations where there is no/ limited coordination between locations. Other locations may only look at how many beds are free at a specific moment, versus a situation where there is coordination between location, where locations coordinate with each other on how to distribute patients among the locations. The method with limited coordination was used to construct the formulas above, in section 4.3.1 the option with coordination will be explained.

4.3.1 Determining flow with Linear programming (with coordination)

Determining flow based solely on the state of the system at the previous timestep might be inefficient, since a flow is only possible if the location has capacity available that is not used by patients on the previous timestep or by incoming patients of other locations with a higher priority. This method, however, does not consider that patients also leave the location at the specific time making room for more patients. Therefore, another option of determining the flow level from i to j for every pair of i, j is by solving an optimization problem, where we have to consider several constraints. This optimization will have the following

form:

$$\min \quad \sum_{(i,j)\in I} W_{i,j}(t+\Delta t) \tag{19}$$

$$\text{s.t.} \quad F_{i,j}(t) \leq FW_{i,j}(t) + W_{i,j}(t) \qquad \forall i,j \in I \tag{20}$$

$$C_i \geq S_i(t+\Delta t) + \sum_{j\in I} W_{i,j}(t+\Delta t) \qquad \forall i \in I \tag{21}$$

$$W_{i,j}(t+\Delta t) = W_{i,j}(t) + FW_{i,j}(t) - F_{i,j}(t) \qquad \forall i,j \in I \tag{22}$$

$$S_i(t+\Delta t) = S_i(t) + \sum_{j\in I} F_{j,i}(t) - \sum_{k\in I} FW_{i,k}(t) \qquad \forall i \in I \tag{23}$$

$$F_{i,j}(t), W_{i,j}(t+\Delta t), S_i(t+\Delta t) \geq 0 \qquad \forall i,j \in I \tag{24}$$

Here line (19) is the objective function, in this case the objective function is to minimize the sum of all waiting lists. Other possible examples for the objective function are: $min \sum_{(i,j)\in I} w_{i,j} * W_{i,j}(t+\Delta t)$, where each waiting list has a different weight, $w_{i,j}$ or to maximizing the flows: $max \sum_{(i,j)\in I} F_{i,j}(t)$, or a combination or something else desired by the user/policy maker.

Lines (20)-(24) show the constraints. Constraints (20) makes sure that flow from i to j can never be bigger than the patients actually waiting/wanting to go from i to j. Constraint (21) is there to limit the number of patients at location i to the capacity of i. Constraints (23) and (22) are similar to equations 14 and 13, as they are used to update the stock levels and waiting lists. Finally, the constraint on line (24) are non negative constraints, since patients cannot be negative. Note that in this LP the following values are given: $W_{i,j}(t), FW_{i,j}(t), C_i$ $\forall i,j \in I$, the following variables are decision variables: $F_{i,j}(t)$ $\forall i,j \in I$ and two variables are calculated using the LP: $W_{i,j}(t+\Delta t), S_i(t+\Delta t)$ $\forall i,j \in I$.

The effect of coordination is show by means of an example, see figure 13 and below.

Example 4.1. The example is a system with two nodes, both with a capacity of 100. Each location has a stock value of 90 and 10 waiting to go to the other location. This means that each location is fully using their capacity. The method of flow determination without coordination will result in a deadlock, since patient cannot go from A to B, since there is no capacity left and patients can also not go from B to A. If this system uses flow determination with coordination the patients waiting at each location could swap, resulting in a system that does not get stuck in a deadlock

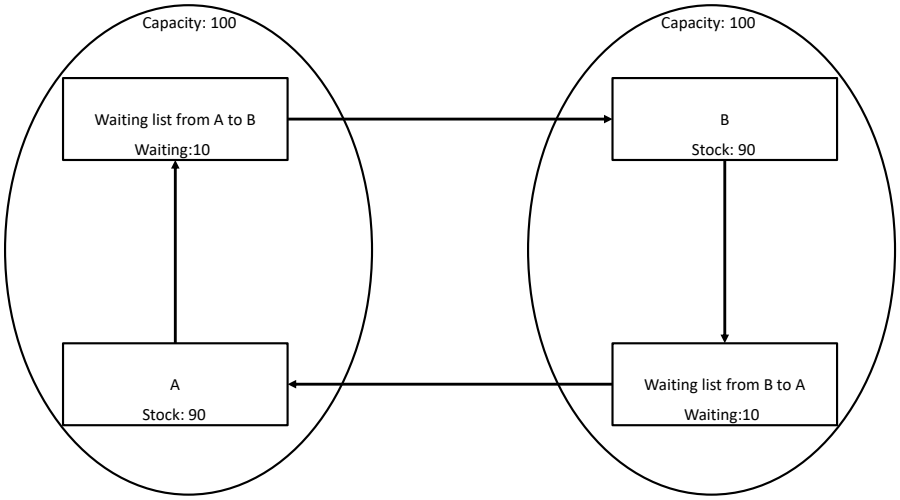


Figure 13: Example of a system, where the effect of coordination is visible

Possible additions to the LP

Extra constraints or other objective function could also be added to this LP. For example, it is possible to use absolute priorities, which can be done by adding constraints. Normally this could be done by an if-then statement, namely if the flow with the highest priority has no waiting list with the determined flow, then flow with second-highest priority may be larger than 0, otherwise this flow has to be one. This would result in the following statement: if $W_{i,k}(t + \Delta t) > 0$ then $F_{j,k} = 0$, else $F_{j,k} \geq 0$ and this should hold for every pair of (i, j) where i has a higher priority then j. This could also be translated to an or-statement, namely: $W_{i,k}(t + \Delta t) \leq 0$ or $F_{j,k} \leq 0$. This can then be modeled to an ILP constraint as follows[6]:

$$\begin{aligned} W_{i,k}(t + \Delta t) &\leq 0 + My_{i,j,k} \\ F_{j,k} &\leq 0 + M(1 - y_{i,j,k}) \\ y_{i,j} &\in \{0, 1\} \end{aligned}$$

Where M is a large constant and $y_{i,j,k}$ is introduced as binary variables to make sure that at least one of the constraints hold.

It is also possible to limit the number of patients that are part of a swap. The definition of a swap is the following.

Definition 4.1. A swap is the simultaneous transferring of patients to recover or avoid a deadlock. After the swap, patients could occupy capacity that was occupied before the swap, but became available due to the simultaneous transfers.

To limit the number of patients part in a swap the following variables and constraints are added:

$$\sum_{i \in I} OF_i \leq max\_patients\_in\_swap \tag{25}$$

$$OF_i \geq \sum_{j \in I}(F_{j,i}(t)) + \sum_{j \in I}(W_{i,j}(t)) + S_i(t) - C_i \tag{26}$$

$$OF_i \geq 0 \tag{27}$$

$$OF_i \leq max\_patients\_in\_swap\_of\_this\_location \tag{28}$$

A new variable OF is added to record the number of patients that are part of the swap, arriving at location i and could not have transferred without swaps, since these patients use capacity that was occupied before the swap. The name OF is chosen as short for overflow. In context, this could be done to simulate a maximum number of beds/rooms that can be cleaned by a location. Constraint (26) makes sure that OF_i is at least the same value as patients arriving at location i occupying beds that were also occupied by patients before the swaps. The right hand side of constraint sums the inflows, stock and waiting lists at location i and subtracting the capacity, resulting in the number of patients that after the swap use beds, that were previously occupied before the swap. If no swap was necessary for transfers to this location, then the right hand side would become 0 or negative, constraint (27) then makes sure that OF_i will not become negative. Constraints (25) and (28) can limit the total number of patients in the swaps or limit the number of incoming patients due to the swap per location.

These two additions can be used to determine the flow based on the current time as described in section 4.1, $max\_patients\_in\_swap$ in constraint (25) should then be set to 0.

4.4 Multiple patients/customer types

In many realistic situations, multiple types of customers or patients use the same capacity. Examples can be found in health care [3], call centers[5] and many other industries. In the system dynamics model, different types of patients may use the same capacity, but have different service rates or transition percentages. This has various effects depending on the flow determination and priority setting. The calculation of the fresh flow (FW) of equation (10) is changed to the following:

$$FW_{i,j,l}(t) = P_{i,j,l} * S_{i,l}(t), \tag{29}$$

where $P_{i,j,l}$ is the patient type dependent transition rate between location i and j, and $S_{i,l}(t)$ is the stock value of patient type l at location i, where L is the set of all the patient types. If flow is determined based on the state of the system on the previous time step and absolute priorities are given, then equation 11 is changed to the following, (assuming that all patient types use the same type of

capacity) :

$$A_{i,j,k} = C_j - \sum_{l \in L} S_{j,l}(t) - \sum_{j \in I} \sum_{l \in L} W_{j,i,l}(t) - F^1, \tag{30}$$

where $A_{i,j,k}$ is the available capacity at j that patients from i of type k may use. The stock is calculated by summing all stock values of different patient types($\sum_{l \in L} S_{j,l}(t)$) and the total number of waiting patients at location j is calculated by summing all the waiting lists from j to other locations of all patient types($\sum_{j \in I} \sum_{l \in L} W_{j,i,l}(t)$), where L is the set of all the different patient types. F^1 is the sum of all the flows to the same capacity, but where the combination of patient type and source has a higher priority, and L is the set of all the patient types.

If the priority setting is wait list dependent, then equations 15,16 and 17 change to the following:

$$TW_i = \sum_{j \in I} \sum_{l \in L} (FW_{j,i,l}(t) + W_{j,i,l}(t)$$

$$FR_{j,i,l} = \frac{FW_{j,i,l}(t) + W_{j,i,l}(t)}{TW_i}$$

$$A_j = C_j - \sum_{l \in L} S_{j,l}(t) - \sum_{j \in I} \sum_{l \in L} W_{j,i,l}(t)$$

Lastly if the flow is determined based on the LP, most parameters change to (i, j, l) to incorporate the patient type. This will results in the following basic model:

$$\begin{aligned}
min \quad & \sum_{(i,j) \in I} \sum_{l \in L} W_{i,j,l}(t + \Delta t) \\
s.t. \quad & F_{i,j,l}(t) \leq FW_{i,j,l}(t) + W_{i,j,l}(t) && \forall i, j \in I, \forall l \in L \\
& \sum_{l \in L} S_{i,l}(t + \Delta t) + \sum_{j \in I} \sum_{l \in L} W_{i,j,l}(t + \Delta t) \leq C_i && \forall i \in I \\
& W_{i,j,l}(t + \Delta t) = W_{i,j,l}(t) + FW_{i,j,l}(t) - F_{i,j,l}(t) && \forall i, j \in I, \forall l \in L \\
& S_{i,l}(t + \Delta t) = S_{i,l}(t) + \sum_{j \in I} F_{j,i,l}(t) - \sum_{k \in I} FW_{i,j,k}(t) && \forall i \in I, \forall l \in L \\
& F_{i,j,l}(t), W_{i,j,l}(t + \Delta t), S_{i,l}(t + \Delta t) \geq 0
\end{aligned}$$

5 System Dynamics: Model in equilibrium

To gain more understanding of the model, various aspects of the model are zoomed into and explained. These aspects are the equilibrium state, the effect of time dependent parameters and the effect the starting stock has on the behaviour of the system. Understanding the behaviour can help by determining whether the system can reach equilibrium without problem or needed interfering.

5.1 Equilibrium of the system of a stationary system

Two-location model

The equilibrium state is the state the system converges to, over time and given enough capacity. This is determined for a stationary system, meaning that the parameters are not time dependent. Below, we illustrate the stationary behavior using a simple two-node example. However, the principles are evidently valid for a network consisting of $n \geq 2$ nodes. These techniques will then be used as well for systems with time-dependent parameters. For this, the following scenario is used: see figure 14. Firstly, the equilibrium state is found by running the model without capacities, with two different starting values of the number of patients at each location. This can be seen in figures 15 and 16.

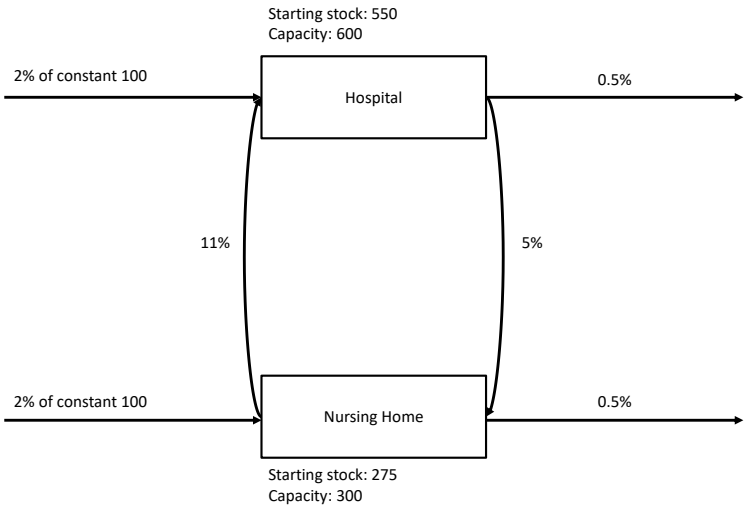


Figure 14: Example used for determining the equilibrium state of a two-location model with stationary parameters

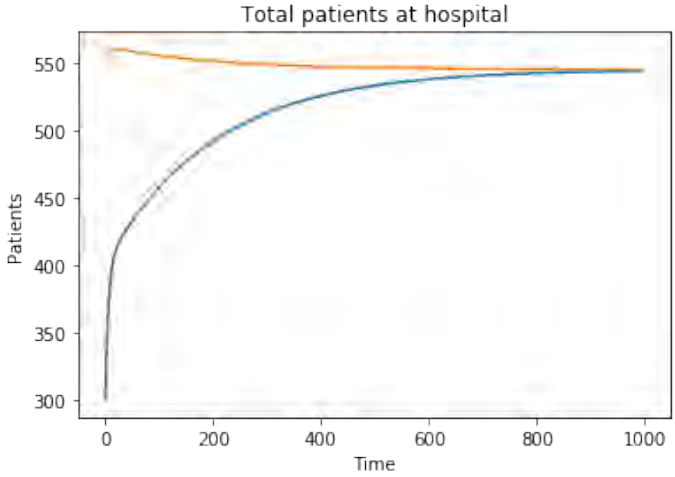


Figure 15: The number of patients at the hospital over time, for two different starting levels (blue and orange)

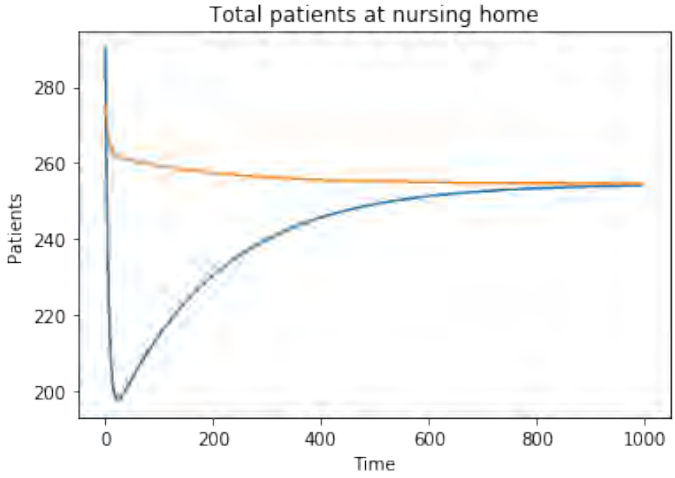


Figure 16: The number of patients at the nursing home over time, for two different starting levels (blue and orange)

Both graphs show that the number of patients at the hospital and at the nursing home will move to a equilibrium. This is independent of the starting state of system, as two different starting states are used, and both move to the equilibrium. These runs are done without taking the capacity into account, as capacity may stop the system from going into equilibrium. The number of patients at equilibrium are 545.455 and 254.545 for hospital and nursing home respectively.

While the equilibrium state can be determined by running the system long enough, it is not known whether the system will go to equilibrium within the given time and given capacity. Therefore, knowing the equilibrium without running the system, provides insight into the demand of each location. Two different approaches are considered and elaborated in the sections 5.1.1 and 5.1.2. Afterwards the equilibrium state will be used to analyze various starting situations.

5.1.1 Inflow equals outflow

For the first approach it is considered that in equilibrium the inflow at one location should be the same as the outflow. The number of patients at equilibrium at a location does not change, meaning that: $S_i(t+\Delta t) = S_i(t)+Flow_{in}-Flow_{out}$ and in equilibrium $S_i(t+\Delta t) = S_i(t)$, then $Flow_{in} = Flow_{out}$. This means that the $Flow_{in} = Flow_{out}$ has the following structure for each location:

$$\sum_{j\in I} F_{j,i}(t) = \sum_{j\in I} F_{i,j}(t) \tag{31}$$

Which can be rewritten to, (given that $FW_{i,j} = F_{i,j}$, which is true if no waiting lists are created):

$$\sum_{j\in I} P_{j,i} * S_j = \sum_{j\in I} P_{i,j} * S_i = (\sum_{j\in I} P_{i,j}) * S_i \tag{32}$$

For example in figure 14, the $Flow_{in} = Flow_{out}$ equation for hospital is the following:

$$0.02 * 100 + 0.11 * S_n = (0.05 + 0.005) * S_h \tag{33}$$

and for the nursing home it would be:

$$0.02 * 100 + 0.05 * S_h = (0.11 + 0.005) * S_n \tag{34}$$

This system of equations can be solved by hand to obtain: $S_n = \frac{2800}{11} \approx 254.545$ and $S_h = \frac{6000}{11} \approx 545.455$. Doing this by hand, when the system includes more locations will take too much time, therefore we will rewrite these equations in matrix form $AX = B$, with $A = \begin{bmatrix} 0.055 & -0.11 \\ -0.05 & 0.115 \end{bmatrix}$, $X = \begin{bmatrix} S_h \\ S_n \end{bmatrix}$ and $B = \begin{bmatrix} 2 \\ 2 \end{bmatrix}$. The solution can then be found by $X = (A^{-1})B$

5.1.2 Queueing theory approach

The second approach is inspired by queuing theory and networks of queues especially. Here, the number of patients at the equilibrium state is determined by the following formula:

$$S_i = \gamma_i/\mu_i \tag{35}$$

γ_i is the effective arrival rate of location i and μ_i is the service rate of location i. γ_i can be calculated by summing all possible flows to i. All possible flows to i include the external arrivals, α, to the system, multiplied by the probability that an external arrival moves to location i, $p_{0,i}$. The effective arrival rate of arrivals from within the system are calculated by multiplying the probability to go to i with the effective arrival rate of the source, see equation 36. For each location there is such an equation, resulting in $|I|$ equations, with $|I|$ unknown variables, the γ_i's.

$$\gamma_i = \alpha * p_{0,i} \sum_{j \in I} \frac{p_{j,i}}{\sum_{k \in I} p_{j,k}} * \gamma_j \tag{36}$$

These equations translate to a system of equations. Next, the service rate can be determined using the following formula:

$$\mu_i = \frac{1}{\sum_{k \in I} p_{i,k}} \tag{37}$$

The equations below are examples for the model seen in figure 14.

$$\gamma_h = 2 + \frac{0.11}{0.115} * \gamma_n \tag{38}$$

$$\gamma_h = 2 + \frac{0.05}{0.055} * \gamma_n \tag{39}$$

These can be solved in the same way as described in section 5.1.1, to obtain the following: $\gamma_h = 30$ and $\gamma_n = \frac{322}{11}$. Then the service rates are: $\mu_h = \frac{200}{11}$ and $\mu_n = \frac{200}{23}$, resulting in $S_n = \frac{2800}{11} \approx 254.545$ and $S_h = \frac{6000}{11} \approx 545.455$ at equilibrium. These values are equal to the ones found using the approach of section 5.1.1.

n-locations

When there are more than two locations these formulas for determining the equilibrium state of the system do not change, for an example see figure 17. The results can be seen in figure 18, and correspond to the equilibrium state of the system you can calculate by using the approach of section 5.1.1, which is approximately 19, 21, 23, 26, 30 patients at locations a,b,c,d,e respectively.

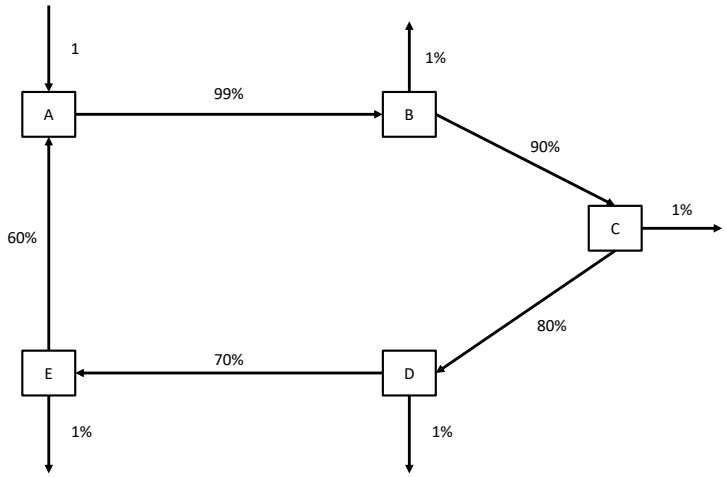


Figure 17: A system of 5 locations

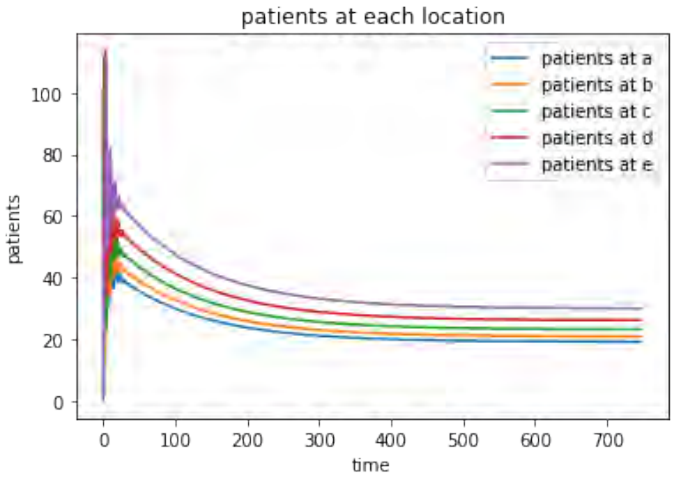


Figure 18: Results of the system seen in figure 17

Special cases

Special cases where the equilibrium state cannot be determined by the proposed formulas are closed systems and systems with a flow of 100% per timestep. A closed system is here defined by the following:

Definition 5.1. A system is closed if there is no possibility to enter or leave the system, therefore the total number of entities/persons in the system remains

constant.

The proposed methods do not work in case of closed systems, since the equilibrium state is mostly determined by the number of patients the system starts with. These patients will then be divided among the locations. However, running the SD model with a closed system, infinite capacities and a long enough run time, the equilibrium state can be determined. The equilibrium state of systems with a flow of 100% per timestep can not be determined using the proposed approaches, as can be seen by the example of figure 19. In this case at every timestep the patients at location A is fully sent to B, and patients at B are fully sent to C, C to D and finally D to A . The starting parameters of this system can be found in table 3 and as can be seen in appendix E, the equilibrium state of this system is not stationary but oscillates.

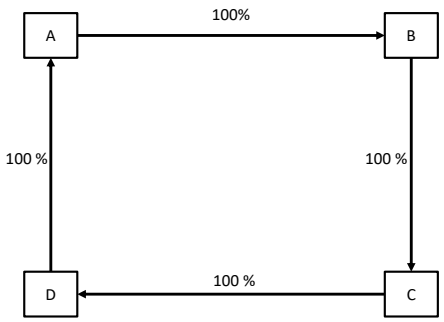


Figure 19: Special case of a closed system with flows of 100% per timestep

Location	Starting stock level	Capacity
A	100	1000
B	100	1000
C	100	1000
D	0	1000

Table 3: Starting parameters of the special case visualized in figure 19

5.1.3 Minimum capacities

Given the fact that a system has an equilibrium state, minimum capacities can be determined. The definition of minimum capacities is given by the following:

Definition 5.2. In the SD model, minimum capacities are the capacities needed for the system to maintain its equilibrium state.

The minimum capacity depends on how the flow is determined; if flow is purely dependent on the state of the system at the previous timestep, see section 4.3, then more capacity is required than if flow is determined using LP which makes swaps possible. Note that these minimum capacities are needed in the context of the SD model, in reality stochasticity might require the system to have more capacity at some or all locations.

First, the minimum capacities are determined, if flows are determined based on the previous timestep. To determine the minimum capacity level, it has to be taken into account how the flows are determined in the system and especially how waiting lists are created. It can be seen from equations 11, 12 and 13, that waiting lists are created when not all patients that want to go from i to j can actually go. This is due to the fact that $S_i(t) + \sum_{j \in I} F_{j,i}(t) \leq C_i^{min}$. If we denote the equilibrium values of stock and flow by S_i^* and $F_{i,j}^*$ respectively, we can say that in equilibrium: $S_i^* + \sum_{j \in I} F_{j,i}^* \leq C_i^{min}$. This is the first constraint for a minimum capacity. The last constraint for a capacity is that the capacity should be bigger than the starting stock level, $S_i(0) \leq C_i^{min}$. Therefore, for the following scenarios the capacity is set at $C_i^{min} = S_i^* + \sum_{j \in I} F_{j,i}^*$.

However, flow can also be determined based on the LP proposed in section 4.3.1. This method requires less capacity as it provides the opportunity to swap patients. Therefore, the minimum capacity has to be bigger than the equilibrium state of the system plus the inflow and minus the outflow per timestep, which is equal to the equilibrium of the stock or: $C_i^{min} \geq S_i^* + \sum_{j \in I} F_{j,i}^* - \sum_{j \in I} F_{i,j}^* = S_i^*$.

5.2 Effect of time dependent parameters

Many realistic situations do not have stationary parameters, but parameters that are time dependent or even dependent on other parameters, e.g. the increasing elderly population. This might result in increasing flows from outside the system. There might even be a seasonal effect, like a sine or cosine's effect. The equilibrium state of the system with a continually increasing or decreasing in/outflow will not result in any stationary equilibrium state, since for a continually increasing inflow the equilibrium state would also increase. However, given a seasonal, inflow it is possible to determine an average equilibrium state, with which it is possible to determine a minimum capacity. For an example, another look is taken at the example seen in figure 14. This example is altered to the following figure 20, where the inflow from outside the system is not from a population of a constant 100, but from a population determined by a sine function: $100 + sin(t/100) * 50$. The effect the sine function has on demand is clear by looking only at demand, this can be seen in figure 21. This is done by running the model without capacities. The demand/ the number of patients at

each location oscillates as expected, given the sine function.

If the capacities are set on the level needed to serve everyone on average equilibrium, there will not be enough capacity during the peaks of the function and more than enough in the troughs. This shortage and surplus in the peaks and troughs of the function would be the same, and the shortage at one moment will result in waitinglist, but these will be completely solved in the troughs when there is capacity left after helping fresh arrivals. When looking at figure 22, it can be seen that even though the demand oscillates, when the capacity is set at the level needed to serve the average of the demand, this capacity is completely used. The reason why this capacity is completely used becomes clear in figure 23; the system uses this capacity to eliminate the waitinglist that are created when there is not enough capacity during the peaks of the demand function. Another option would be to let the capacity move together with the demand. To summarize: if capacities are set on the level needed to serve everyone on average equilibrium, then waiting lists will grow during peak season, but will diminish during slow season and one can say that the capacity is enough.

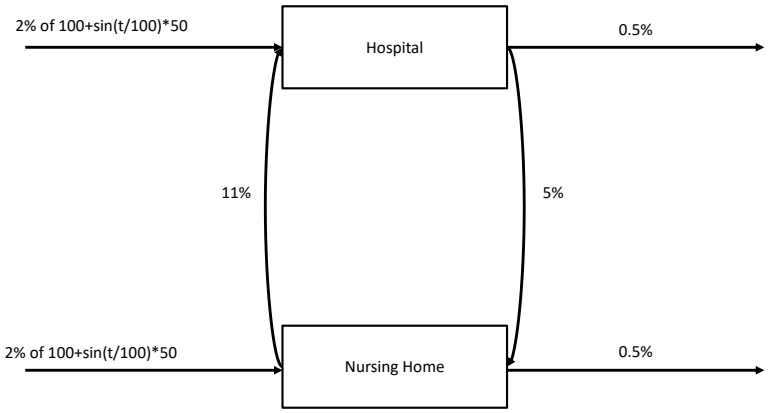


Figure 20: Example of a two-location system with time dependant arrivals from outside the system

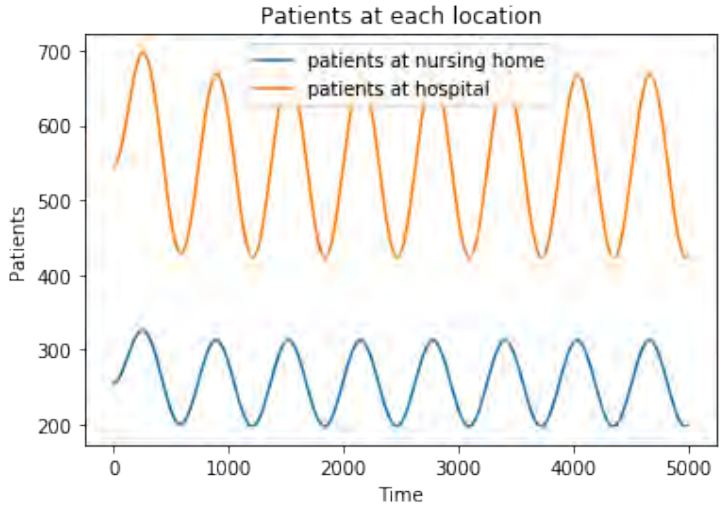


Figure 21: The demand per location of the example given in figure 20, without capacities

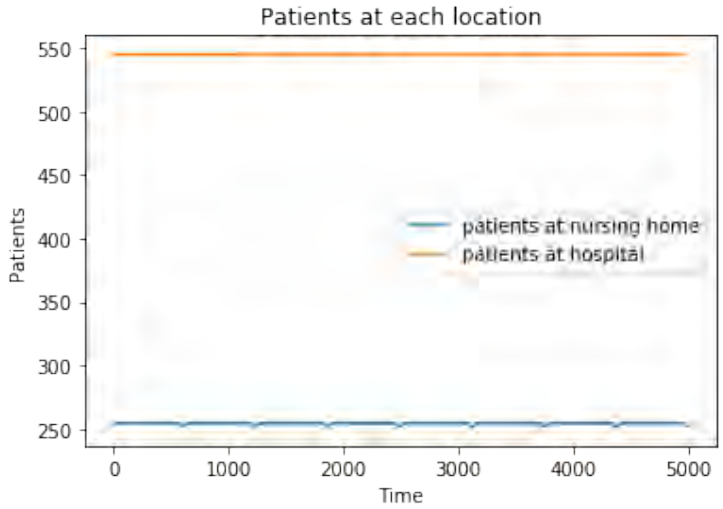


Figure 22: The number of patients at each location, when capacities are set at the minimum level required to serve all patients in the average equilibrium

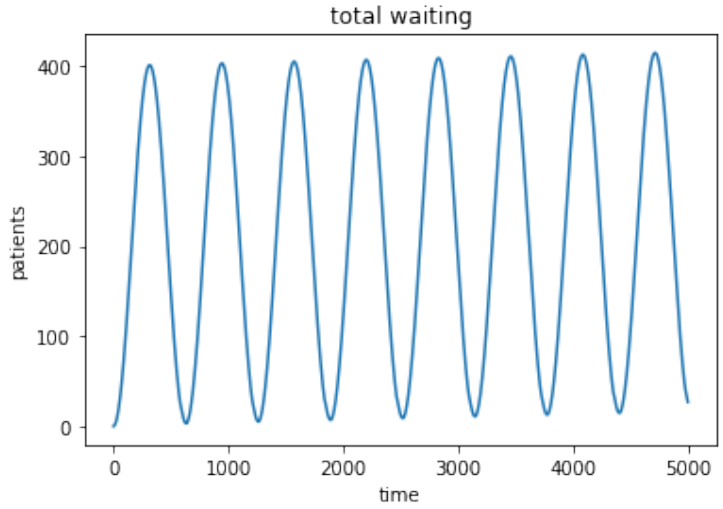


Figure 23: The number of patients waiting, when capacities are set at the minimum level required to serve all patients in the average equilibrium

5.3 The effect of starting stock on the system

Now various starting situations of the system are analyzed to gain insight in how the system behaves given a minimum capacity to serve the equilibrium state of the system and the starting levels of the stock. The configurations used for these analyzes can be found in section 4.

5.3.1 Configuration: Current practice

Every stocklevel starting below equilibrium

For the first scenario, every stock starts below or equal to equilibrium, $S_i(0) \leq S_i^*$ for every $i \in I$, where I are all possible locations and the capacity is set at the minimum capacity described in section 5.1.3, namely capacity is set at the equilibrium level of the stock plus the inflow at equilibrium. The capacity is set at this level since a capacity lower than this will not have enough capacity to sustain the equilibrium state and will therefore eventually fill itself with waiting lists. First, this system (see figure 14) is ran with absolute priorities that prioritize the flow within the system. The results of these experiments can be seen in figure 24 and 25. As can be seen in the graphs, if the stock levels start below or equal to the the equilibrium level, then the stock will move to the equilibrium state without issues, such as wait lists or deadlocks.

This can also be proven mathematically and be used for other priorities such as prioritizing outside flows or wait list dependent priorities.

Lemma 1. Given that all stock values start below the equilibrium, ($S_i(0) \leq S_i^*, \forall i \in I$), then the stock will move to the equilibrium state without issues (wait lists or deadlocks)

Proof. Issues can arise when at one point a wait list is created ($S_i(t)+\sum_{j\in I}(FW_{j,i}(t)+W_{j,i(t)}) > C_i$), we are certain no issues arise when no wait lists are created ($S_i(t) + \sum_{j\in I}(FW_{j,i}(t) + W_{j,i(t)}) \leq C_i$). Therefore, to prove that the system will go from the starting position to the equilibrium position, the following needs to hold:

$$S_i(t) + \sum_{j\in I}(FW_{j,i}(t) + W_{j,i(t)}) \leq C_i = S_i^* + \sum_{j\in I}(FW_{j,i}^* + W_{j,i}^*) \tag{40}$$

at any time t and for all locations $i \in I$. For ease of reading $FW_{j,i}(t)+W_{j,i(t)} = F_{j,i}^*(t)$. Firstly equation 40 is rewritten to, assuming $W_{i,j}(t) = 0$ for all t and i,j,:

$$S_i(t) + \sum_{j\in I} F_{j,i}^*(t) = \tag{41}$$

$$S_i(t) + \sum_{j\in I} S_j(t) * P_{j,i} \leq S_i^* + \sum_{j\in I} S_j^* * P_{j,i} \tag{42}$$

Therefore proving $S_i(t) \leq S_i^*$ for all $i \in I$, is enough to prove equation 40. To start we know that the start levels of the system are $S_i(0) \leq S_i^*$, and given equations 14,12,10, we can write:

$$S_i(0 + \Delta t) = S_i(0) + \sum_{j\in I}(F_{j,i}(0)) - \sum_{j\in I}(F_{i,j}(0)) \tag{43}$$

$$= S_i(0) + \sum_{j\in I}(S_j(0) * P_{j,i}) - \sum_{j\in I}(S_i(0) * P_{i,j}) \tag{44}$$

$$= S_i(0) + \sum_{j\in I}(S_j(0) * P_{j,i}) - S_i(0) * \sum_{j\in I}(P_{i,j}) \tag{45}$$

$$= (1 - \sum_{j\in I}(P_{i,j}))S_i(0) + \sum_{j\in I}(S_j(0) * P_{j,i}) \tag{46}$$

$$\tag{47}$$

It is assumed that the outflow of i can arrive at any other location j without waiting. This assumption can be proven by looking again at the state of the system in equilibrium, in which it is known that there is enough room at j to receive the flow from i, since $C_i = S_i^* + \sum_{j\in I} F_{j,i}^*$. Given that $S_i(0) \leq S_i^*$ for all locations, it can also be determined that $F_{j,i}(0) \leq F_{j,i}^*$, therefore $C_i \geq S_i(0)+\sum_{j\in I} F_{j,i}(0)$ and flow from i to j can be received without any waiting times.

Given the following facts:

- $p_{i,j} \geq 0$

- $\sum_{j \in I} p_{i,j} \leq 1$
- $S_i^* = (1 - \sum_{j \in I}(p_{i,j}))S_i^* + \sum_{j \in I}(S_j^* * P_{j,i})$

We can conclude that:

- $(1 - \sum_{j \in I}(p_{i,j}))S_i(0) \leq (1 - \sum_{j \in I}(p_{i,j}))S_i^*$, since $S_i(0) \leq S_i^*$ and $(1 - \sum_{j \in I}(p_{i,j})) \geq 0$
- $\sum_{j \in I}(S_j(0) * P_{j,i}) \leq \sum_{j \in I}(S_j^* * P_{j,i})$, since $S_j^* \geq S_j(0)$

And therefore

$$S_i(0 + \Delta t) \leq S_i^* \tag{48}$$

This process can then be repeated and therefore generalized to :

$$S_i(t) \leq S_i^* \tag{49}$$

This proof works with any priority setting. Since priority only plays a role when a wait list is created, meaning that the demand is bigger than supply at a moment, or in the setting of this project: the patients want to move to j, but there are not enough beds at j for all the patients. □

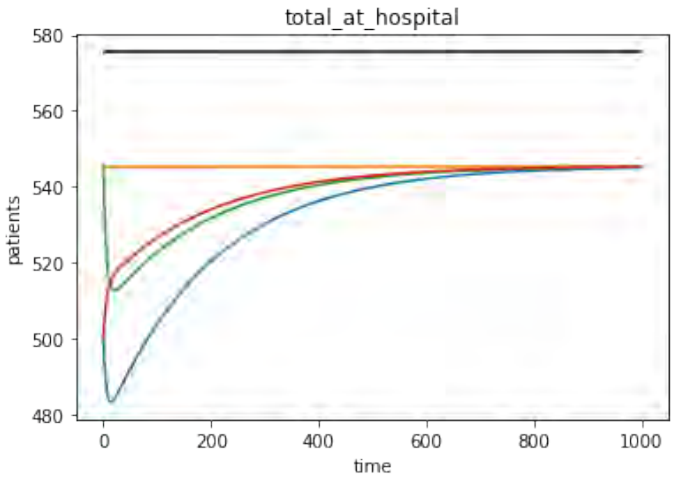


Figure 24: Patients at the hospital when starting below equilibrium for different scenarios

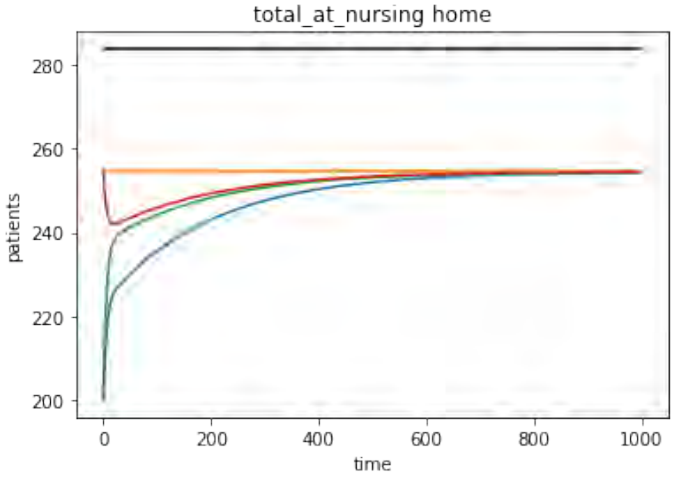


Figure 25: Patients at the nursing home when starting below equilibrium for different scenario

Every stocklevel starting above or equal to equilibrium

In this situation, every starting stocklevel in the system has a level equal or larger than the stocklevels at equilibrium. The capacity is again set at the level described in section 5.1.3.

Two outcomes are possible when every stock level starts at or above the equilibrium level, and inside flows are prioritized. This becomes clear when looking at the following example:

- Four locations (A,B,C,D) and the possibility of the in and out flow of the system.
- Flow described by the following matrix:

	A	B	C	D	Outside
A	$P_{A,A}$	$P_{A,B}$	$P_{A,C}$	$P_{A,D}$	$P_{A,O}$
B	$P_{B,A}$	$P_{B,B}$	$P_{B,C}$	$P_{B,D}$	$P_{B,O}$
C	$P_{C,A}$	$P_{C,B}$	$P_{C,C}$	$P_{C,D}$	$P_{C,O}$
D	$P_{D,A}$	$P_{D,B}$	$P_{D,C}$	$P_{D,D}$	$P_{D,O}$
Outside	$P_{O,A}$	$P_{O,B}$	$P_{O,C}$	$P_{O,D}$	$P_{O,O}$

Where $P_{A,B}$ is the percentage of patients wanting to go from A to B per time step. Filled in:

	A	B	C	D	Outside
A	0	0.1	0.05	0.05	0.02
B	0.05	0	0.1	0.05	0.02
C	0.05	0.05	0	0.1	0.02
D	0.1	0.05	0.05	0	0.02
Outside	0.01	0.01	0.01	0.01	0

Where the outside population is a constant 100.

- Given the in and outflow percentages the equilibrium state of the system can be determined. This is 50 patients for each location. This would mean that the minimum capacity (given that the flow is determined based on the state of system the previous timestep) is approximately 61 for each location.
- The capacities are each set at approximately 61.
- Priority is described by a matrix with same form, where $P_{A,B}$ is the priority level that patients from B have when going to A. This matrix filled in:

	A	B	C	D	Outside
A	0	1	2	3	4
B	0	1	2	3	4
C	0	1	2	3	4
D	0	1	2	3	4
Outside	0	1	2	3	4

As can be seen, flows within the system are prioritized compared to flows from outside the system.

The first outcome is that the system goes to the equilibrium and the extra population at the start is translated to a waitinglist in equilibrium that is pushed outside the system. Two of the stocks are set at a starting level of 55, so the total population at the start is 10 higher than the population in equilibrium. What can be seen in figures 26 and 27, is that the stocks all move the equilibrium state, where wait lists are created at the start of the system, but after a period the wait lists are reduced to 10, which is equal to the surplus at the start of the system.

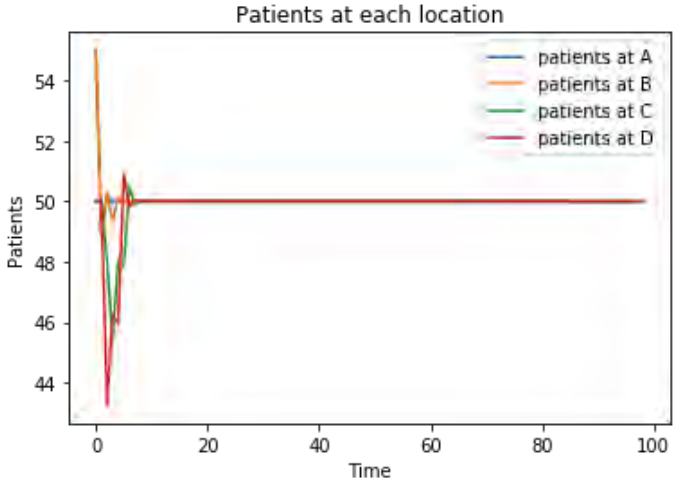


Figure 26: Number of patients at each location, when two of the stocks starts each with 5 over the equilibrium

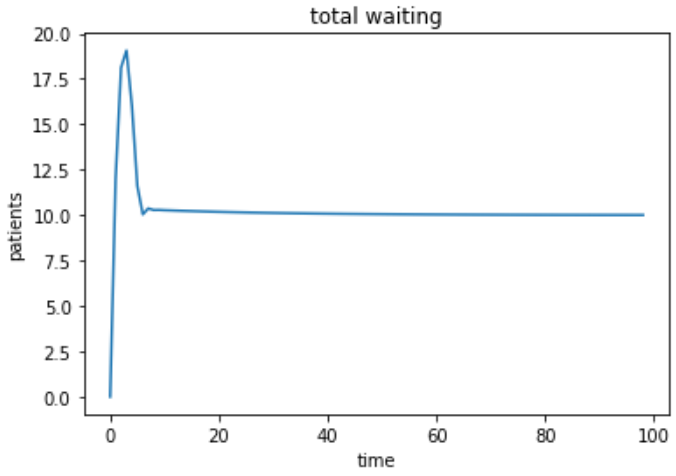


Figure 27: Number of patients waiting, when two of the stocks starts each with 5 over the equilibrium

The second option is that the system gets stuck in a deadlock. This becomes visible in the following example: each stock is set to start at a level of 55, 5 higher than the stock in equilibrium. The result is visible in figures 28 and 29, where the surplus is so big at the start that a deadlock arises, and therefore the wait lists continually increases due to new arrivals that cannot enter system.

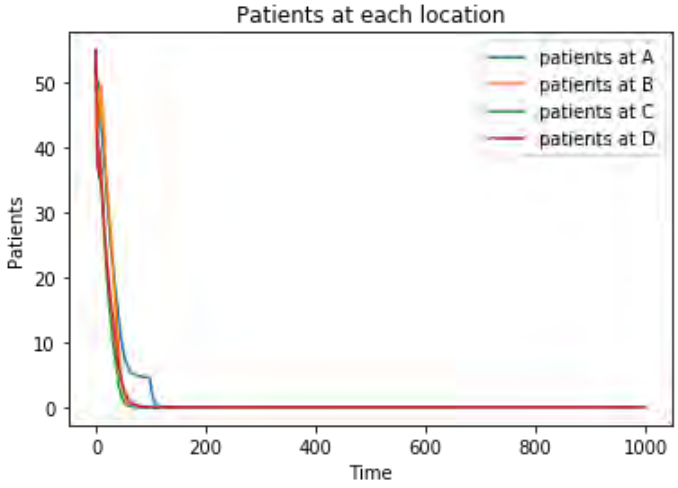


Figure 28: Number of patients at each location, when all of the stocks starts each with 5 over the equilibrium

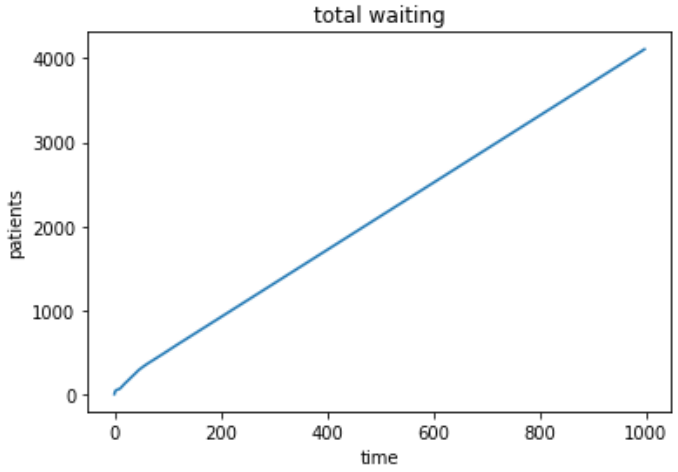


Figure 29: Number of patients waiting, when two of the stocks starts each with 5 over the equilibrium

While it is possible that there is a boundary to decide which option the situation will go, in this paper the flow determination using the LP will be used to avoid this problem.

When outside flows are given priority, the priority matrix changes to look like this:

	A	B	C	D	Outside
A	4	1	2	3	0
B	4	1	2	3	0
C	4	1	2	3	0
D	4	1	2	3	0
Outside	0	1	2	3	4

It is clear how the system will behave then. Since the flows from outside are given priority, at least one of the flows within do not have enough capacity to fully flow from one location to another creating a waitinglist. This waitinglist will not recover and only become larger as it is not given priority and will therefore not have enough capacity, eventually creating deadlocks. This can be seen in the results from the same system but now with only one location starting above the equilibrium level. For this experiment, the starting levels of stocks are 51,50,50 and 50 for locations A,B,C and D respectively, see figures 30 and 31.

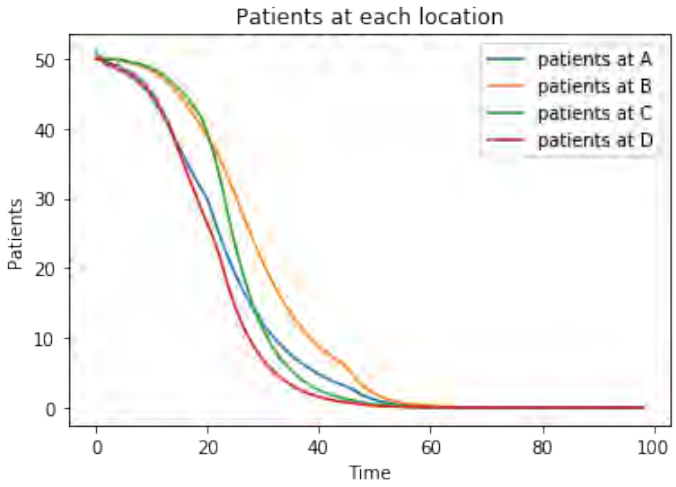


Figure 30: The number of patients receiving service at each of the location in the situation where flows from outside are given priority and at least one of the location starts with more patients than in equilibrium.

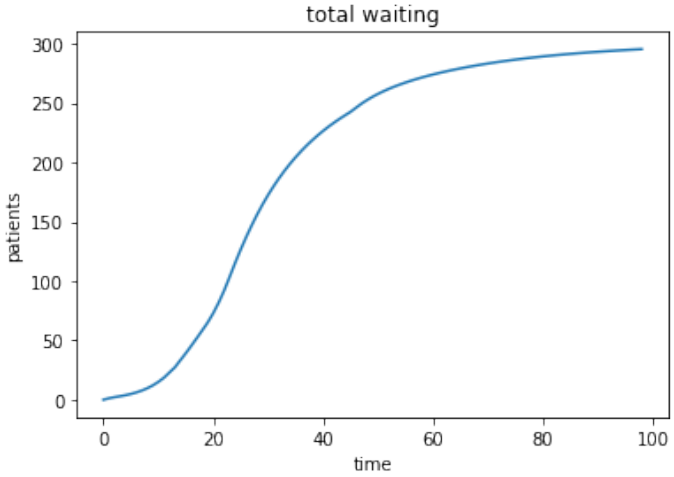


Figure 31: The number of patients waiting for transfer in the situation where flows from outside are given priority and at least one of the location starts with more patients than in equilibrium.

5.3.2 Configuration: optimization model

Every stocklevel starting below equilibrium

For the last configuration, the proof of section 5.3.1 still holds for when all stock start below equilibrium. The system will still go to the equilibrium state of the system without any issues.

Every stocklevel starting above or equal to equilibrium

In this situation every starting stocklevel in the system has a level equal or larger than the stocklevels at equilibrium, the capacity is again set at the level described in section 5.1.3.

Since the capacity for configuration 3 can be set to exactly the equilibrium state, starting higher than this equilibrium would mean starting higher than the capacity. Therefore the capacity is set at the starting level of the stocks. In the following each stock is set at 55, and therefore the capacity is also set at 55, as can be seen in figures 32, each location moves to the equilibrium state without creating any wait lists.

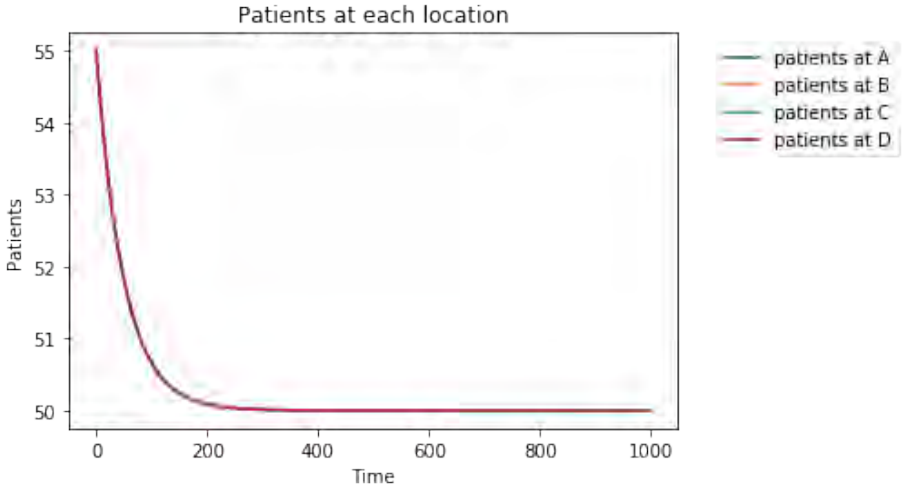


Figure 32: The number of patients at each location, when all starting positions are above equilibrium and each of the capacities are set at the minimum between starting position and the equilibrium state

6 Discrete event simulation: Methodology

The next model used is a discrete event simulation, to replicate the system of elderly care in the Amsterdam region. This section will start with an explanation of how deadlock recovery functions in this setting, then the pseudo code will be elaborated and finally the validation results of simulation of M/M/1 and M/M/C queues.

6.1 Deadlocks in DES

In networks where it is possible to directly or indirectly transfer between two locations, there is a possibility that deadlocks occur. To detect whether the system is in a deadlock, first a wait-for-graph needs to be constructed of the state of the system. A wait-for-graph is dependent on the state of the system. For example, a two location (a,b) system can be used, with each two servers. When the system is empty or no patients are waiting, the system will resemble figure 33.

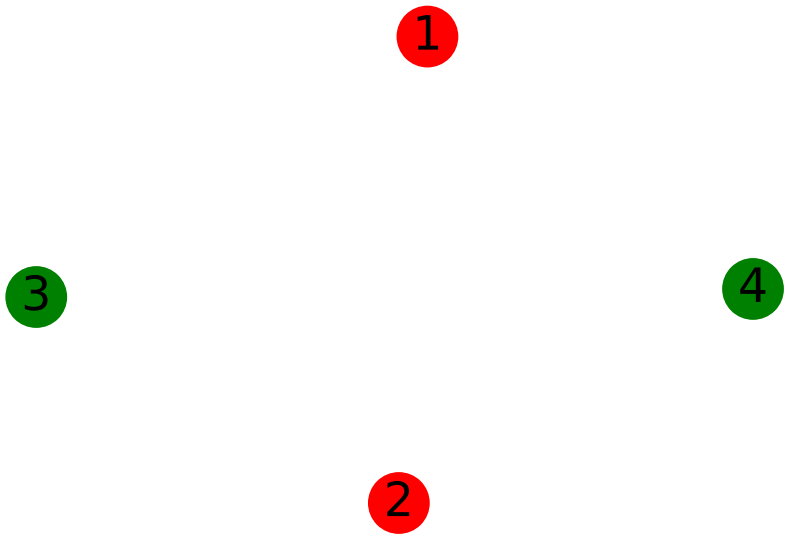


Figure 33: Wait-for-graph of an empty two locations system, red nodes correspond to location a and green to location b

Edges are added when there are patients waiting for capacity at another location. For example, if the patient at server 1 waits for available capacity at b, then the wait-for-graph will shows directed edges from server 1 to all servers of b, since this patient waits for one of those servers to become available, see figure 34.

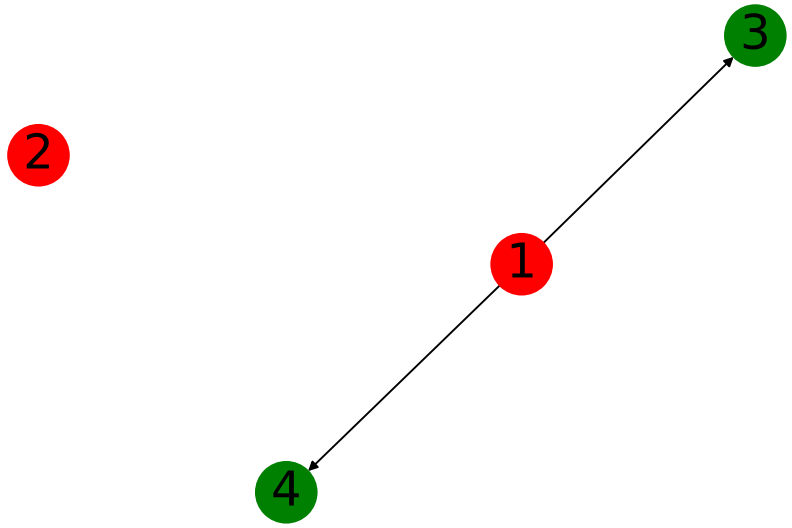


Figure 34: Wait-for-graph of a two locations system, red nodes correspond to location a and green to location b, with one patient waiting

The system is said to be in deadlock if the wait-for-graph contains a knot, which is a strongly connected component containing no nodes with an edge to vertices outside the strongly connected component. Two examples can be seen in figures 35 and 36. Figure 35 contains a knot and therefore the system is in deadlock. Figure 36 does not contain a knot since the strongly connected part contains nodes 1,2,4 and there are edges out of this strongly connected component to node 3, therefore this system is not in deadlock.

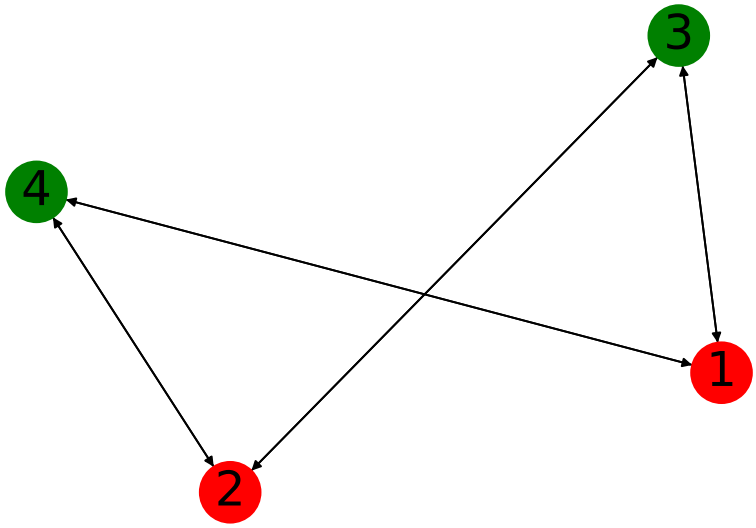


Figure 35: Wait-for-graph of a two locations system,in deadlock

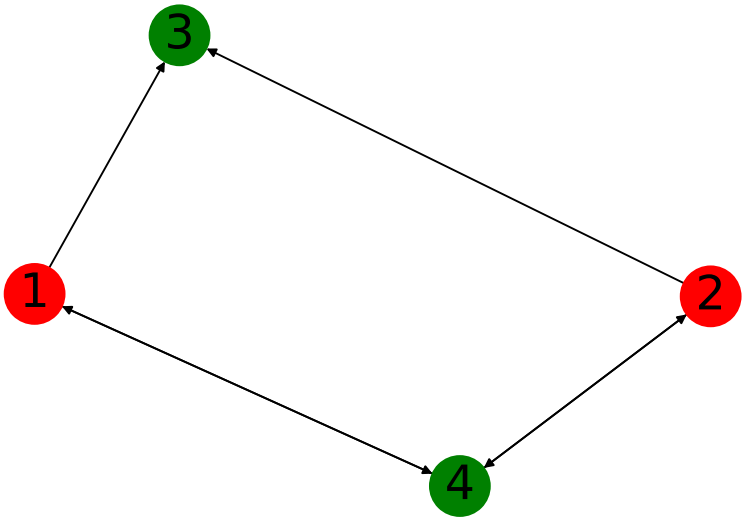


Figure 36: Wait-for-graph of a two locations system, not in deadlock

When a deadlock is detected, the system can be recovered to a system with-

out deadlock with the help of the LP described in section 4.3.1. The only change to this LP is that for the DES the LP is changed to an ILP, since only an integer number of patients can be send. Therefore the LP changes to the following ILP:

$$\min \quad \sum_{(i,j)\in I} W_{i,j}(t+\Delta t) \tag{50}$$

$$\text{s.t.} \quad F_{i,j}(t) \leq FW_{i,j}(t) + W_{i,j}(t) \qquad \forall (i,j) \in I \tag{51}$$

$$C_i \geq S_i(t+\Delta t) + \sum_{j\in I} W_{i,j}(t+\Delta t) \qquad \forall i \in I \tag{52}$$

$$F_{i,j}(t) \in \mathbb{Z}^+ \qquad \forall (i,j) \in I \tag{53}$$

The values of F are then used to determine which patients to swap, for example if $F_{i,j}(t) = y$, then the first y patients on the waiting list from i to j are sent from i to j.

While the deadlock detection and recovery helps the system after it gets stuck in a deadlock, it might be better to avoid a deadlock at all. This deadlock avoidance could be done by running the LP after each service completion to determine the optimal way of transferring patients. However, this might result in running the LP unnecessary. The LP should only be ran when there are swaps necessary to either avoid or recover a deadlock, or when a swap can help reduce waiting times. A swap can reduce the size of the waiting lists and therefore waiting times, by transferring patients that either block each other directly or indirectly if the wait-for-graph has a cycle. Therefore, in the DES-model, the LP will only be used when there are at least two patients waiting and when the wait-for-graph has a cycle. The complete wait-for-graph is then unnecessary, since a cycle would also be visible in a reduced wait-for-graph, where each node represents a location instead of a server. Figure 35 would then be reduced to the following: figure 37. For cycle detection depth-first-traversal is used, with time complexity $O(|V|+|E|)$ [12], where V is the number of vertices/nodes and E is the number of edges. For the wait-for-graph in this simulation V is the number of locations and E is lesser or equal than V^2, so time complexity is $O(|V|+|V^2|)$.

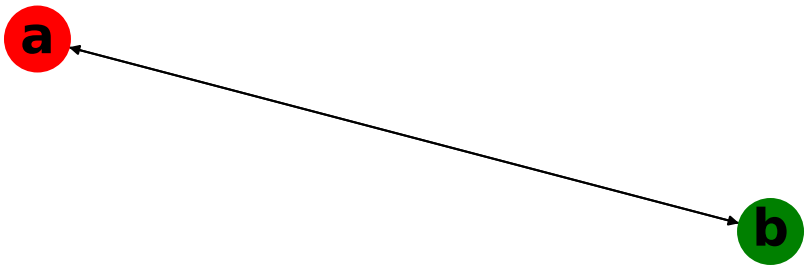


Figure 37: Reduced wait-for-graph of a two locations system, in deadlock

6.2 Model and pseudo-code

The simulation is explained by pseudo-code as well as in text.

Start of the simulation

The simulation is started by drawing an interarrival time from an exponential distribution and entering this as an arrival in the event list, as can be seen in the pseudo-code.

Running the simulation

The simulation is ran as long as the time t inside the simulation is smaller or equal to the given end time. One event is processed at a time, after which the next event is retrieved. The time t is set at the time of the event. The type of event determines how this event is process, an arrival requires other actions than a service completion.

Arrival process

If this next event is an arrival, it is determined based on the external arrival rates of each node to which node this patient goes. The patient tries to enter service at his target node, this is possible if there are no waiting patients for this node and there is a server free ($S_i < C_i$). The patient enters the waiting list for

this node, if the patient cannot enter service at his target node. When a patient enters service a service time is drawn from an exponential distribution and the service completion event is added to the event list. After the arrival is processed, a next arrival event is added to the event list, for which an interarrival time is drawn from an exponential distribution.

Service completion

If the next event is a service completion, it is first determined whether this patient requires care at another node, if this is the case, it is checked whether this service can start. If the service can start or if the patient leaves the system it checked what other waiting patients can transfer or start their service. If a patient cannot start service or leave his current node, he waits at the current node occupying a service spot. If there are two or more patients waiting in the system the deadlock detection method is ran and if necessary possible deadlocks are recovered or avoided.

Algorithm 1: Pseudo-code of the DES-model

Initialize the starting values of the DES: C,p,begin time, end time, $S = 0$, $W = 0$ and t=begin time.
Create an empty event list in which events and their times are saved and add the first arrival.;
while $t \leq$ *end time* **do**
  Retrieve the next event.
  t is updated to the time of the event.
  if *The next event is an arrival* **then**
    Add new arrival event to the event list with an interarrival time drawn from an exponential distribution.
    Determine the target node (i) of the arriving patient based on the external arrival rates.
    if $S_i(t) + \sum_{j \in I} W_{i,j}(t) < C_i$ **then**
      The patient enters service at node i.
      S_i is increased by one.
      A service time is drawn from an exponential distribution, the service completion event is added to the event list.
    else
      The patient waits in the queue outside of the system.
      $W_{0,i}$ is increased by one.
  else if *The next event is a service completion* **then**
    S_i is decreased by one.
    Determine next target j of the patient t, based on p, the transition probabilities.
    if *the next target location j is within the system* **then**
      if $S_j(t) + \sum_{k \in I} W_{j,k}(t) < C_j$ **then**
        The patient enters service at node j.
        S_j is increased by one.
        Service time is drawn from an exponential distribution, the service completion event is added to the event list.
        It is checked whether other patients can start service as well since there is now free capacity at i. All these service completions are as well added to the event list and S and W are updated if necessary
      else
        Patient cannot enter service and has to wait at the current node.
        Increase $W_{i,j}$ by one.
        Deadlock detection and recovery.
    else
      The patient departs from the system.
      It is checked whether other patients can start service as well since there is now free capacity at i. All these service completions are as well added to the event list and S and W are updated if necessary

6.3 Validation

Validation of the simulation model is based on a few different theoretical models, first M/M/1 and M/M/C. This is done to verify if the simulation works and behaves as expected. Lastly, the model is also validated using a Jackson network.

6.3.1 Validation of M/M/1 and M/M/C queues

The single queue models M/M/1 and M/M/C only differ in the number of server/agents are used, apart from that both models assume arrivals are drawn from a Poisson process, service times are drawn from an exponential distribution and the queue capacity is infinite. These two models are chosen, since for both models some statistics can be calculated without the need of simulation. Validation will be done by comparing results from the simulation and the theoretical models of 5 different randomly chosen parameter settings. The results used for comparison will be the expected number of customers in the system ($E(L)$), expected time spent in the system($E(S)$) and two service levels measures ($P(W > t), P(W > t|W > 0)$).

For validation purposes, the expected number of customers in the system ($E(L)$) or expected queue length($E(L_q)$), expected time spent in the system($E(S)$) and a service levels measure ($P(W > t)$) will be calculated using theoretical calculations and the simulations. The arrival rate (λ) and the service rate (μ) will be chosen such that $\rho = \frac{\lambda}{\mu} < 1$ for M/M/1 queues and $\rho = \frac{\lambda}{c\mu} < 1$ for M/M/C queues. Since expected queue length can tend to infinity for arrival and service rates for which $\rho \geq 1$. Calculations are done following the formulas given in [1]. For the random parameter settings and the validation results see tables 4 and 5. As can be seen in the table, the results from the simulation and the theoretical calculations differ slightly in some cases, but in general are very close. The simulation results could become more precise by running the simulation for longer time periods if needed.

Situation	**Type of Queue**	**c**	λ	μ	ρ
1	M/M/1	1	10	11	0.91
2	M/M/1	1	1	2	0.50
3	M/M/1	1	5	8	0.63
4	M/M/c	6	15	18	0.14
5	M/M/c	11	20	2	0.91
6	M/M/c	3	10	13	0.26
7	M/M/c	5	8	3	0.53
8	M/M/c	2	13	10	0.65

Table 4: Configuration of the queues used for validation.

Situation	Simulation results			Theoretical results		
	$E(L)$	$E(S)$	$P(W \leq 0.5)$	$E(L)$	$E(S)$	$P(W \leq 0.5)$
1	10.05	1.05	43.46	10.00	1.00	44.86
2	1.00	0.99	68.83	1.00	1.00	69.67
3	1.62	0.32	86.70	1.67	0.33	86.05
4	0.83	0.06	100.0	0.83	0.06	100.0
5	16.90	0.84	74.51	16.82	0.84	74.91
6	0.80	0.08	100.0	0.79	0.08	100.0
7	2.87	0.36	99.48	2.85	0.36	99.51
8	2.23	0.17	98.77	2.25	0.17	98.45

Table 5: Validation results: the results of the simulation compared to that of calculation based on theoretical knowledge of M/M/C queues.

6.3.2 Validation of a Jackson network

A Jackson network is a network of queues. For a detailed description see section 3.2.1. To quickly summarize a network of queues is a Jackson network if the following conditions hold:

- If external arrivals are possible to a node, then these external arrivals are formed by a Poisson process.
- All service times are drawn from an exponential distribution and customers are served following a first-come-first-served policy.
- A served customer at node i will go to another node j with probability P_{ij} or leave the system with probability $r_i = 1 - \sum_{j=1} P_{i,j}$.
- The utilization of every queue, ρ_i, is less than one, so $\rho_i < 1$.
- All queues have unlimited capacity.

For validation, the expected number of customers per node and in total between simulation model and the theoretical model are compared. An example is used with six locations $(1, 2, 3, 4, 5, 6)$, in total 600 customers arrive per time unit for all six locations. These external arrivals are evenly distributed among the location, therefore each location will have 100 external arrivals per time unit. The following parameters are known:

Node	**c**	λ	μ	r
1	24	100	30	1/6
2	35	100	20	1/6
3	20	100	40	1/6
4	60	100	12	1/6
5	16	100	40	1/6
6	20	100	36	1/6

Table 6: Parameters used for the validation of Jackson network

The transition probabilities are given by the following matrix. Note that the probability of a departure out the system is not in the matrix, but is 1 minus the sum of the row. Each location has the following probability of a patient leaving the system after service at the location: $1 - 5 * \frac{1}{6} = \frac{1}{6}$.

$$P = \begin{bmatrix} 0 & 1/6 & 1/6 & 1/6 & 1/6 & 1/6 \\ 1/6 & 0 & 1/6 & 1/6 & 1/6 & 1/6 \\ 1/6 & 1/6 & 0 & 1/6 & 1/6 & 1/6 \\ 1/6 & 1/6 & 1/6 & 0 & 1/6 & 1/6 \\ 1/6 & 1/6 & 1/6 & 1/6 & 0 & 1/6 \\ 1/6 & 1/6 & 1/6 & 1/6 & 1/6 & 0 \end{bmatrix}$$

The validation results can be found in the following table:

Node	**E(L) - simulation**	**E(L) - theory**
1	21.08	21.49
2	31.73	31.71
3	15.54	15.48
4	49.78	50.59
5	24.52	25.95
6	18.28	18.36
Total	160.93	163.58

Table 7: Validation results from the simulation of Jackson network

As can be seen in the table, the expected customers per node only differs slightly between the results from the simulation and from theory, validating the simulation as a correct simulation of the Jackson network.

7 Results

The results section is divided in two main parts. First the case is explained on which the models are used. Secondly, the results obtained from the SD model and results obtained from the DES model are discussed.

7.1 Practical case

In this section, a case will be described that is similar to the elderly care system in the Amsterdam region. This case will then be evaluated using the models in section 4 and 6, based on a systems dynamics approach and a discrete event simulation approach. The system used for this case is based on the figures seen in the report of "Krakende ketens"[27] and in figure 3. The system seen in figure 3 is reduced to a simpler system with only the relevant locations. These are locations where patients stay long term (more than a few days) and where capacity is a crucial aspect, either time, material or personnel wise. The reduced system can be seen in figure 38, with the following locations: hospital, nursing home and home care, arrivals from and departures to outside the system are elderly persons that do not require long term care, however these people can still use their GP's.

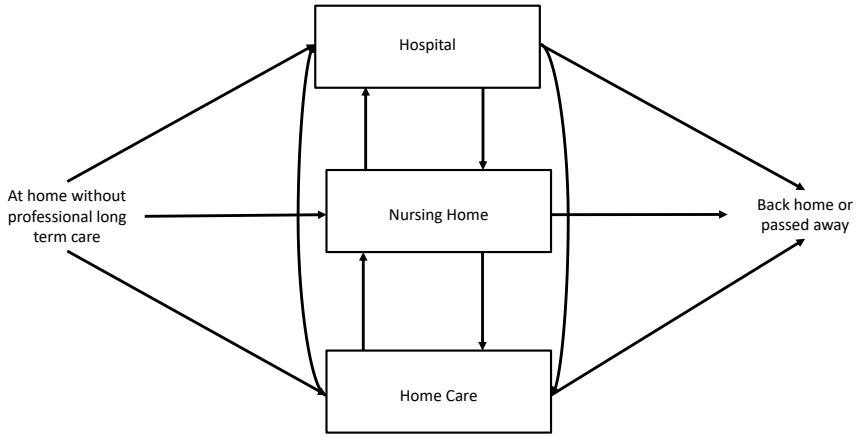


Figure 38: System of elderly care system in the Netherlands

To determine the parameters of this system, public and respectable online sources are used, such as open CBS data and news articles. First, the average population of each location is estimated and after that how many patients on average transfer from each location to another. For the SD-model these val-

ues are then translated to the following: P, the percentages of patients that move to another location every time step, and the population sizes. For the DES-model these values are translated to arrival and service rates, as well as transfer percentages.

Public data shows that there are around 45000 hospital admissions from elderly patients in Amsterdam[11]. It is estimated that on average elderly patients stay around one week. Therefore, it is estimated that each week around 750 patients then move back home or pass away, around 80 patients leave the hospital to require home care and around 10 move to the nursing home. Next, in the nursing home it was estimated that there are around 4000 patients in nursing homes in Amsterdam[16] and of these patients each week around 25 pass away[9][10]. Since it is also possible that elderly patients from the nursing home require hospital care, the number of patients that transfer each week from nursing home to the hospital is set to 5. Using public data from Vektis[29], the average population of home care is estimated to be around 9000 elderly persons, and since patients use home care long term, it is assumed that each week around 275 patients transfer from home care to the hospital, 5 from home care to the nursing home and 275 pass away or transfer from home care back to home without professional long-term care. Lastly, around 95000 elderly persons are on average at home without any professional long-term care.

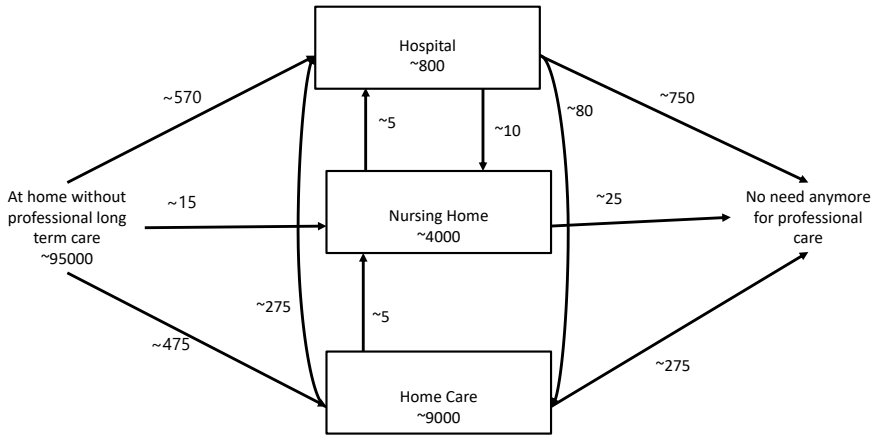


Figure 39: System of elderly care system in the Netherlands with estimated values for the average population size and average flow size per week

After trial and error the following values are used for the SD-model: the following transfer sizes per week,P:

From\To	Hospital	Home care	Nursing home	At home without home care
Hospital	0%	10%	1%	89%
Home care	3%	0%	0.05%	3%
Nursing home	0.1%	0%	0%	0.6%
At home without home care	0.6%	0.5%	0.015%	0%

Table 8: Parameters settings for the SD model of the practical case

The starting values of the stock are all set below the found equilibrium levels and the priorities are set such that at each location patients from the hospital are prioritized the highest followed by patients from home care, nursing home and last home without professional care. The capacities and time parameters are chosen based on the equilibrium state, which will be discussed in the results section.

Using the SD-model parameters, the following parameters are used for the DES-model, where each location has zero capacity for their queue in front:

Locations	arrival rate	service rate	Output percentages		
			Hospital	Home care	Nursing Home
Hospital	570	1	0	0.1	0.01
Home care	475	0.0605	0.496	0	0.008
Nursing home	14.25	0.007	0.143	0	0

Table 9: Parameters of the DES model based on the practical case

7.2 Results: SD model

Using the SD model the following results are obtained:

- The equilibrium levels given the system described in the practical case section.
- The needed capacity according to the SD model in case the system works with or without coordination.
- An example of how the SD model can be used to determine the effect of policy changes.
- What an aging population might do to the system if there are no policy changes.

Equilibrium levels

The equilibrium levels are found by running the SD model with the given parameters found in the practical case section, with $\Delta t = \frac{1}{7}$, where each time unit is one week. The begin time of 0 and an end time of 100 is chosen, since it was observed that the system stabilizes before 100 time units have passed. As can be seen in the table below, the found equilibrium levels are as expected.

Location	**Equilibrium level**
Hospital	852
Home care	9259
Nursing home	4134

Table 10: Found equilibrium levels of the SD model

Minimum capacities

Minimum capacities are calculated using the method described in section 5.1.3 and tested out by running the system with the described capacities in either a system with or without coordination. As can be seen in the tables and figure 40, the relative difference is the biggest at the hospital. The hospital needs 14% more capacity when no coordination is used compared to the situation where locations coordinate together how to transfer patients, according to the SD model. This effect is likely the largest at the hospital since the length of stay is here the shortest.

Locations	**With coordination**	**Without coordination**
Hospital	852	973
Home care	9259	9339
Nursing home	3914	3918

Table 11: Needed capacities according to the SD model for the system with and without coordination

Locations	**With coordination**	**Without coordination**
Hospital	100%	114%
Home care	100%	101%
Nursing home	100%	100%

Table 12: Needed capacities in percentages of the needed capacity with coordination according to the SD model

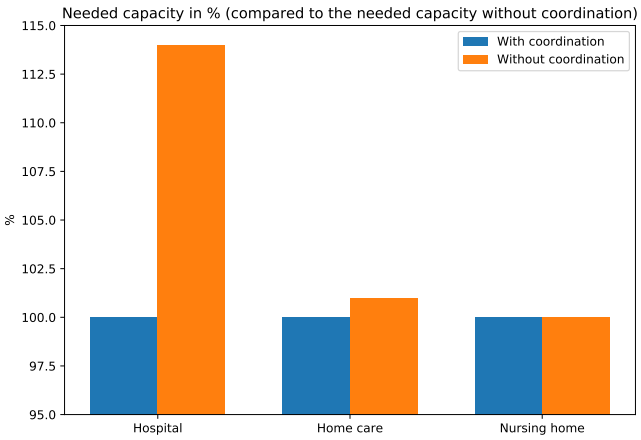


Figure 40: Bar chart of the needed capacity compared to the system with coordination

Aging population

According to a news article by AT5[2], the elderly population of Amsterdam will increase by approximately 45000 in the next 10 years. If linear growth is assumed, then each week the elderly population will increase with approximately 85. After 10 years this growth will stop and the population will stabilize. This can be modeled in the system by replacing the population of elderly persons at home without long-term professional care, which was a constant 95000 by the following formula $95000 + 85t$, where t is time elapsed in the system in weeks. If $t > 520$ then the population will be determined by $95000 + 85 * 520 = 139200$ and remain constant. Figure 41 shows how the average population grows during the increase in elderly persons and how it stabilizes after the growth stops. The growth is mostly visible in the average population of home care, but all three locations show an increase as can be seen in figure 41.

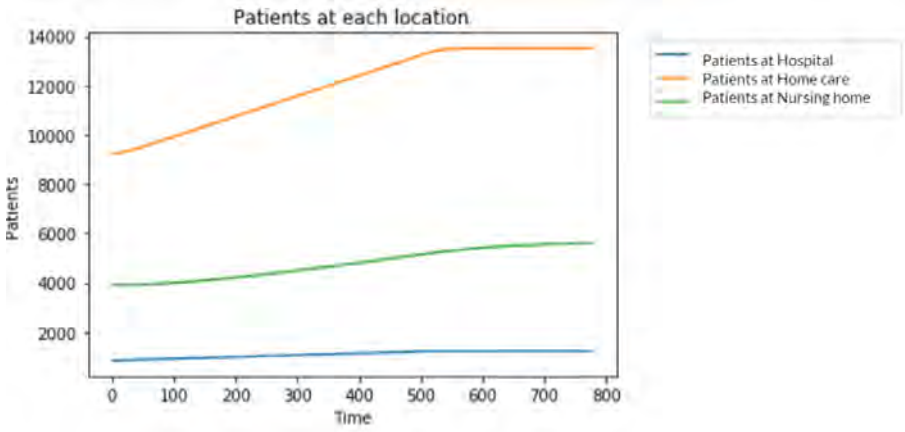


Figure 41: Line chart of average population at each location

Policy changes

Policy changes range from small to large, where either only one transfer size changes or multiple. In discussion with the research group it was difficult to quantify what the direct effect of some policy changes is on the parameters. To show what the SD model can do to quantify the full effect of policy changes, the following example is used: fall prevention. Fall prevention can help elderly people by making their homes safer such that falls become less frequent or less severe. This policy change has effect on the population that lives at home either with or without home care. It is assumed that due to fall prevention elderly persons at home are 40% less likely to go the hospital. This decrease is visible in the percentage that transfers from home care to the hospital (changes from 3% to $3 * 0.6 = 1.8\%$) and at home without home care to the hospital (changes from 0.6% to $0.6 * 0.6 = 0.36\%$). Assuming that the average length of home care use stays the same, the patients that finish home care and would previously go to the hospital, now stop using home care and stay at home without home care, therefore this percentage changes as well (changes from 3% to $3+3*0.4 = 4.2\%$). The parameters then become:

From\To	Hospital	Home care	Nursing home	At home without home care
Hospital	0%	10%	1%	89%
Home care	1.8%	0%	0.05%	4.2%
Nursing home	0.1%	0%	0%	0.6%
At home without home care	0.36%	0.5%	0.015%	0%

Table 13: The parameters of the SD model after the policy change

The following figures 42 and 43 show the effect of this policy change. Figure 42 shows the absolute differences between the situation before and after the policy change, it is difficult to tell using this figure what the actual effect is. The other figure, 43, shows the relative differences and tells us that the the average hospital population decreased by approximately 40%, which is as expected since most hospital admissions come from elderly persons living at home, but apart from that the average populations from home care and nursing home also decreases by around 10%. This shows that a change at one place in the system can have effect at other places.

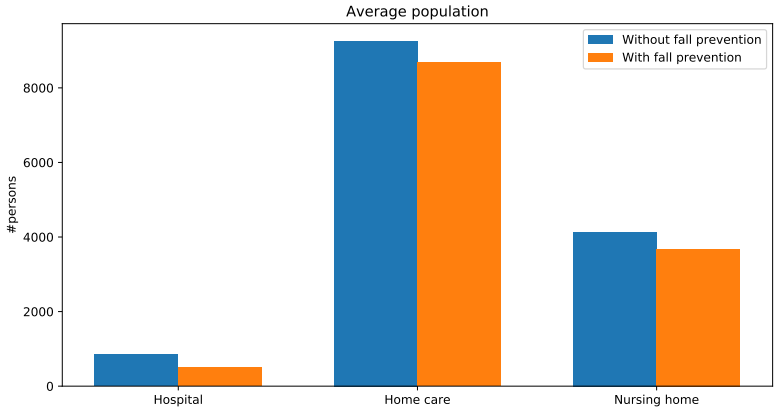


Figure 42: Average size of population at each location

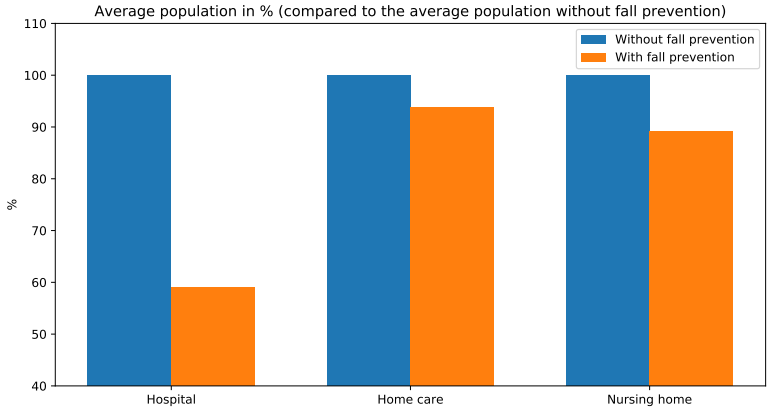


Figure 43: Average size of population at each location

7.3 Results: DES model

The DES model is used on the same system as the SD model, but now the stochastic elements are included. The effect of the stochasticity will become visible by doing the following runs with the DES model: runs where swaps are not possible and runs where swaps are executed as soon as found useful. Both these settings are ran with the capacities found in section 7.2 using coordination. These capacities are then again used but now increased by one times the square root of the capacity and two times the square root of the capacity. These increments are chosen based on research done by Green et al.[18], to see what effect increased capacities have on quality of service. Each run had a run time of 14 hours, in the simulation corresponding to around 150 weeks. The run time of the simulation is long due to the different time scales of the locations. The hospital has arrivals and departures every simulation hour, while the nursing home has a few departures every week in simulation time. The warm up period is shortened by starting the system full. This section will contain two parts: (1) to see the effect of stochasticity and (2) the effect of swaps.

The effect of stochasticity

By running the system with the needed capacity of the SD model it immediately becomes clear how big the effect of stochasticty can be on the system. Figures 44 and 45 show that, with or without swaps, the system does not have enough capacity and the waiting list blows up, resulting in ever growing waiting lists. These two graphs already show the effect of stochasticity, where the SD model had no waiting times and lists with this capacity, and the DES model shows that in reality this would not hold.

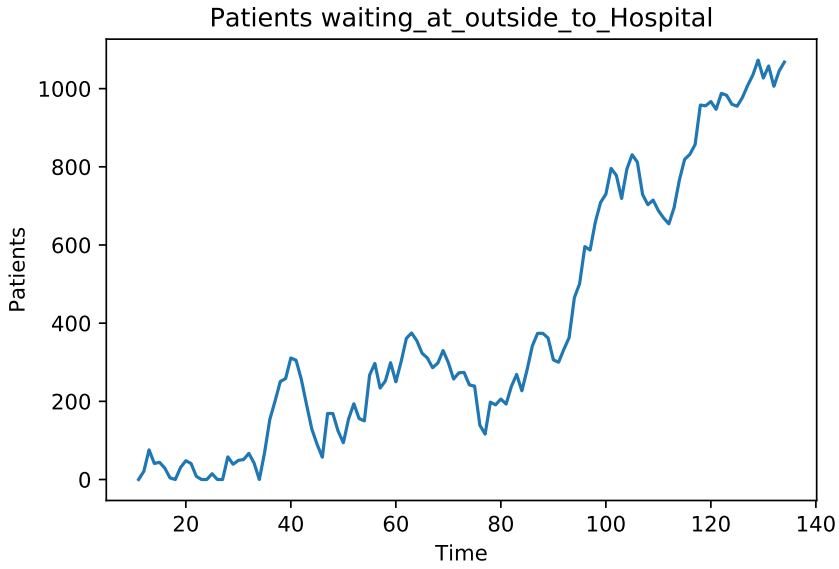


Figure 44: Number of patients waiting outside the system to go to the hospital when the DES model is ran using the capacity of the SD model and swaps are done when needed.

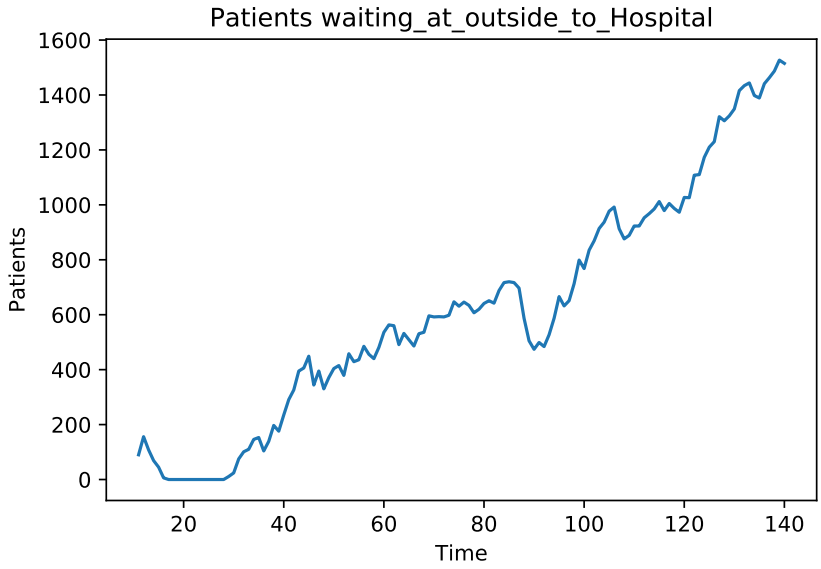


Figure 45: Number of patients waiting outside the system to go to the hospital when the DES model is ran using the capacity of the SD model and no swaps are done.

By increasing the capacities with one or two times the square root of the capacity, the system becomes stable as can be seen in figures 46 and 47. In these figures it is visible that there are still patients waiting at some times, so the result of no waiting patients at any time from the SD model could in this case only be obtained by using infinity capacity, otherwise there will always be a chance (however small) that all capacity will be used at one time.

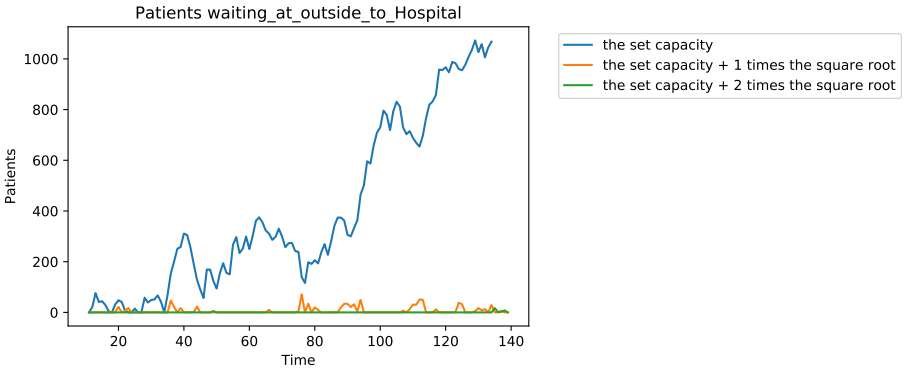


Figure 46: Number of patients waiting outside the system to go to the hospital when the DES model is ran with swaps done when needed.

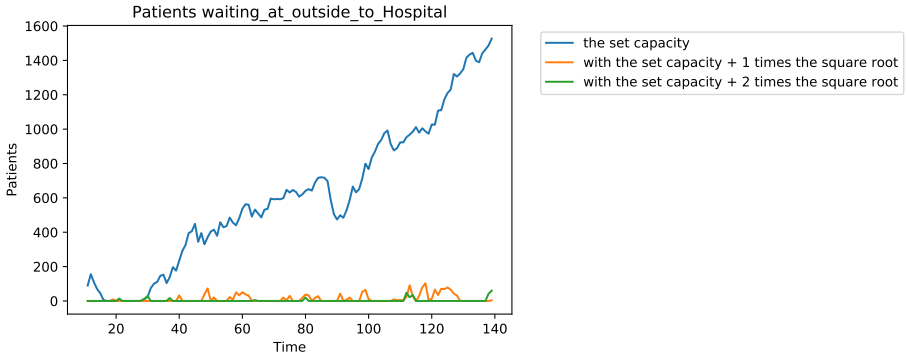


Figure 47: Number of patients waiting outside the system to go to the hospital when the DES model is ran and no swaps are done.

It is also visible that, due to stochasticity. there is always a probability that a person has to wait. However, by increasing the capacity, the average waiting time decreases and the probability that one has to wait also decreases. This can be seen in the following graph 48, where more patients can be served faster or start service without waiting.

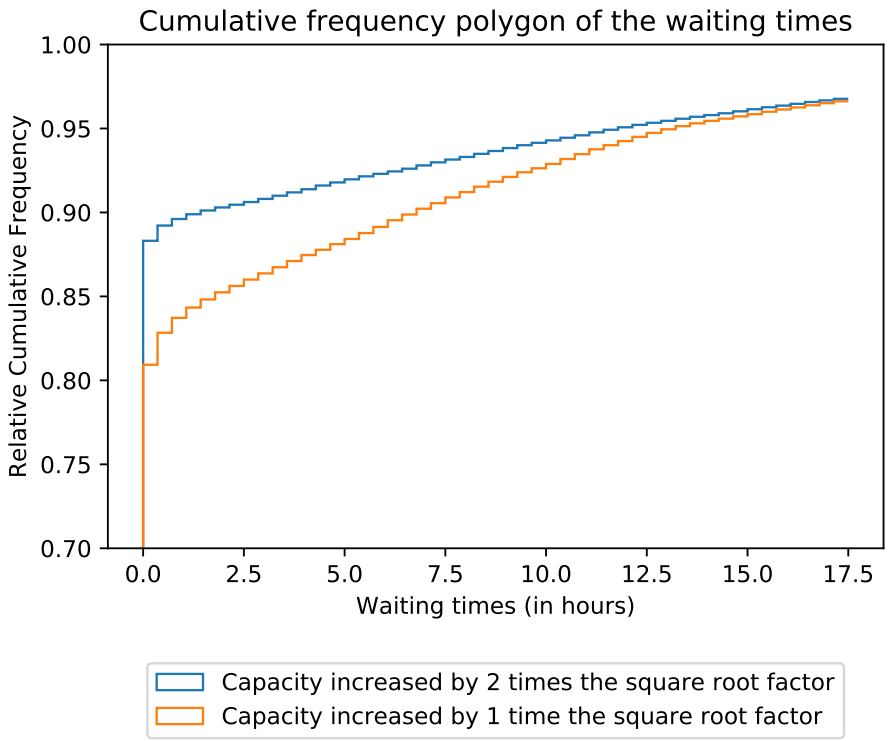


Figure 48: Cumulative frequency polygon of the waiting times of two capacity settings. The other capacity setting is not used as it resulted in a unstable system.

The effect of swaps

The effect of swaps on the system is measured by comparing the results when swaps are allowed and when swaps are not allowed, with every capacity setting used above. The effect is visible in figure 49, where with swaps the average waiting time is noticeably lower. However, note that these waiting times are heavily influenced by the waiting times for service at the hospital, since most patients arrive and are served at that location. The simulation runs proved to be too short to obtain reliable results for the average waiting times of nursing homes. A longer run time is therefore needed or another technique, such as estimation or running the locations separately. Some general observations were also made during these runs namely:

- More swaps were used in settings with less capacity, than in settings with more capacity
- Swaps were mostly done between two locations, namely the hospital and home care and consisted of 2 patients, one of each locations.

- A short test with only allowing swaps at a given time interval showed that swaps could still be useful, but more research is necessary to determine the full effect.

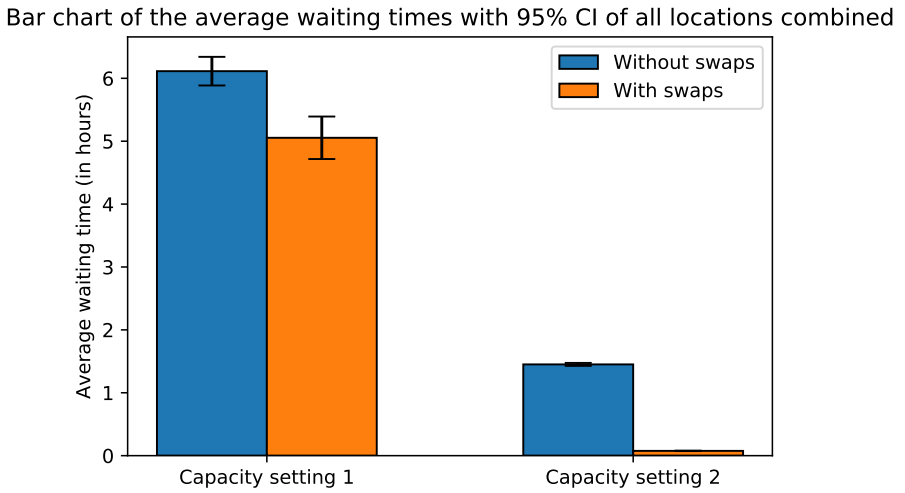


Figure 49: Bar graph of the average waiting times, where capacity setting 1 is where the capacity is increased by the square root factor and setting 2 is where the capacity is increased by two times the square root factor.

8 Conclusion

This paper shows the usefulness of SD model as well as DES model, together showing the importance of simulation models for elderly health care.

Results show that the SD model can be used to sketch a system view of the elderly health care in Amsterdam, showing that locations are dependent and affected by each other. The results in section 7 show how the SD model can be used to determine the average population size of locations, the possible effect of coordination and importantly the effect of an aging population and policy changes on the long term. The example used in the results section of fall prevention shows how a change of parameter at one location has effects on other locations as well. The DES model results however, show that, setting the capacity purely based on the SD model, ignores the effect of stochasticity and therefore underestimating the needed capacity and effect on waiting times. The conclusion therefore is that the SD model can best be used to determine what the system does on the long-term, such as large population changes or changes in health care policies by either local or national governments. The SD model cannot be used as a tool for all purposes as stochasticty is not included.

The DES model has a much longer run time than the SD model and it is therefore not advised to use for long-term simulations. This model can, however, be used to gain insight into the effect of stochasticity and shows how the swaps can work in this given setting. In the results section it was clear that ignoring the effect of stochasticity results in capacity set too low, letting the waiting lists grow. The DES model also shows what the effect of swaps can be. However, it is difficult to estimate how important these swaps are in the realistic setting. These swaps are necessary when there is chance on a full deadlock that locks the whole system, but swaps can also be beneficial in other cases. The effect of swaps seems to be largest in systems where there is barely enough capacity for the given demand.

To summarize, the SD model shows that macro models can be used to gain insight into the system and determine the effect of an elderly population growing. The SD model can also be used to determine indirect effects policy changes have on the system as a whole. The macro model should however not be used to determine the capacities as, due to the absence of stochastic element, the model underestimates waiting times and lists, resulting in unstable systems. The DES model can then be used to simulate the day-to-day operations and therefore determine actual capacities. However, this model can better not be used to determine long term effects, given the long run time and slow convergence.

9 Discussion

This paper aimed to show the use of simulation models in the elderly health care in Amsterdam. The modelling part required some assumptions and could be expanded in future use to closely resemble the reality, but due to time constraints these additions have to wait for possible future research. In this section some of these assumptions will be discussed and what additions could be made or what research could be done in the future.

The system described in section 7.1 is based on public news articles or public aggregated data from CBS. However, not all needed data was readily available and some of these values were estimated. For future research it will be advised to base these parameters on real and more complete data, such that this system will closely resemble the realistic situation. In this way the results and conclusions drawn from the system can be of real value for stakeholders, such as care providers or insurance companies. Right now, the results and conclusion from this paper can still be seen as valuable insights into the system.

For the SD model, it is chosen to only include the direct parameters, in reality the length of stay of a elderly person can be influenced by many factors, such as but not limited to the budget, the availability of informal care, the neighborhood the patient lives in, the season and many more. It might be difficult to identify and include more factors, but the most important of the factors could be useful to provide a more dynamic system to mirror policy changes or an aging population which might be more dependent on family and friends.

Both the SD and DES model do not take into account the possibility of elderly people worsening due to waiting for the right care. This worsening is also known in the queueing theory as abandonment, where a person can abandon the queue to departure out of the system or to join another queue. An example might be an elderly person waits for home care, he/she worsens in state and now requires hospital care. This could be crucial in a system where people have to wait too long for the right care.

The run time of the DES model is significantly longer than the SD model due to multiple reasons. First, the DES model follows each patient individually and therefore has to draw and keep track of more service, waiting and arrival times. Another reason is that, due to stochasticity, the results from the DES model need to converge, otherwise these are not reliable. Lastly and perhaps most interestingly, currently the DES model simulates three different locations where care is provided, namely hospital, home care and nursing home. These locations operate on different time scales; in the hospital patients stay days or perhaps weeks, in the nursing home patients stay years. The hospital has multiple arrivals and departures every day, while the nursing home changes less frequently. This results in the nursing home converging and warming up very slowly, while in the hospital this is not the case. Future research could be done

in reducing this needed run time, by either an approximation method, such as a machine learning model trained on a DES generated dataset or a method based on the known Jackson network, or splitting the locations in short and long term locations as the nursing home seems static for the hospital.

The use of swaps, transferring two patients occupying each other intended capacity, is also discussed. Swaps are only necessary in the case of deadlocks. In reality it is hard to estimate how often these deadlocks occur and how they are resolved manually by personnel of different locations. There are also limitations on the swaps: swapping 20 patients every day might prove to be impossible in reality, so there is probably a limitation on how many patients can be swapped, but there might as well be a limitation on when these patients can be swapped. Both these limitations can be added easily in the two models as the preparations for these additions are already made when the models were constructed. To summarize, the swaps are useful for resolving deadlocks, however, at the moment it is unknown whether deadlocks occur frequently and what kind of limitations are connected to these swaps.

This paper was solely focused on models for the elderly care system in Amsterdam. However, the described models could also be used for other purposes. The notion of deadlocks and swaps can be generalized to be used in various settings where capacity is important and waiting persons or objects can block other waiting persons or objects.

Appendices

A Possible paths in the system

Possibles paths an elderly person can take as visualized in 'Krakende Ketens'[[27]].

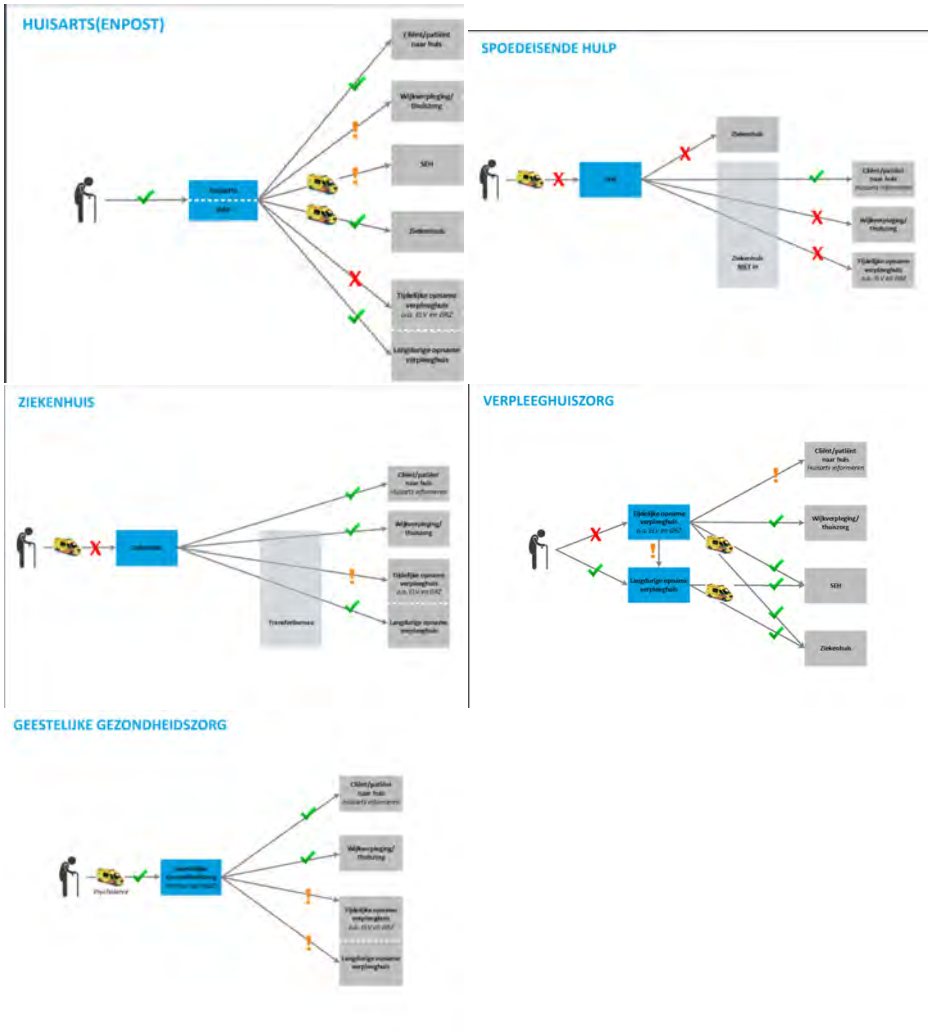


Figure 50: Possible paths in the system of elderly care

B Queueing theory basics

Figure 51: Visual representation of a simple one queue model

Four elements are of importance in a queueing model, namely: the arrival process, service process, number of servers and space in the queue. These four elements are often noted using Kendall's notation as A/B/C/D. A denotes the arrival process often M for Poisson arrivals. B is the service process, often M (Markovian or memoryless) for exponential service times. C is the number of servers and D is the number of spaces for customers. These models are often visually represented as in figure 51, as can be seen $C = 1$ and $D = \infty$ in this figure. If $D = \infty$ often D will not be given in the notation then.

The number of customers in a queueing system can often be described using (embedded) Markov chains. Using these Markov chains the steady-state distribution can be determined. These steady-state distributions tells us the probability of the system to have a number of customers, often noted as $\pi(x)$, where x is the number of customers. This can then be used to evaluate, predict or optimize the queue, for example to make sure that the probability of having a number of customers in the queue is minimal.

C Example of a system dynamics model

An example taken from available literature[30].

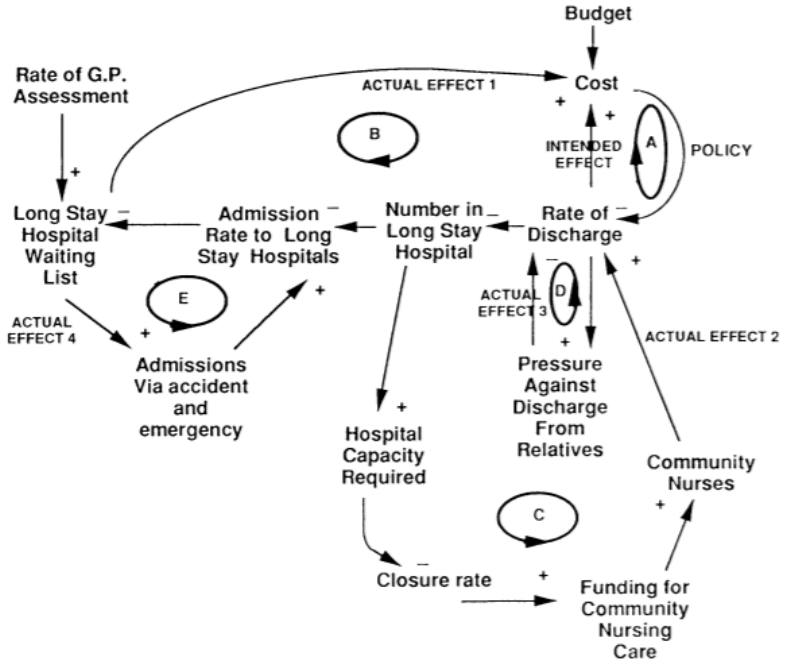


Figure 52: Example of a system dynamics model[[30]]

D Tips & tricks for linear programming

Absolute values

Ever decision variable of which the absolute value is used in either the constraints or objective can be replaced. This is done using the following replacement:

$$x_j = x_j^+ - x_j^-$$
$$|x_j| = x_j^+ + x_j^-$$
$$x_j^+, x_j^- \geq 0$$

This ensures that the problem after this replacement is linear even when absolute values where used.

Min-max objective

A minimax objective means that the objective of the LP is to minimize the maximum of function, which could look something like:

$$minimize \quad max_{k \in K} \sum_{j \in J} c_{kj} x_j$$

This can be transformed to a linear problem by the following transformation, introduce an extra decision variable z, alter the the objective and add an extra constraints as such:

$$\begin{aligned} min \quad & z \\ s.t. & \sum_{j \in J} c_{kj} x_j \leq z \qquad \forall k \in K \end{aligned}$$

Either-or constraints

Either-or constraints means that given two constraints at least one of the constraints must hold. If for example the following constraints are given:

$$\sum_{j \in J} a_{1j} x_j \leq b_1$$

$$or$$

$$\sum_{j \in J} a_{2j} x_j \leq b_2$$

Normally this cannot be done linearly since in a LP all constraints must. To model the either-or constraints, first a binary variable y is introduced, this is done to activate one of the two constraints. Next the constraints are rewritten with the help of a big value M.

$$\sum_{j \in J} a_{1j} x_j \leq b_1 + My$$

$$\sum_{j \in J} a_{2j} x_j \leq b_2 + M(1 - y)$$

If $y = 0$ then the first constraints is activated and the second weakened and if $y = 1$, then the first is weakened and the second one activated.

Conditional constraints

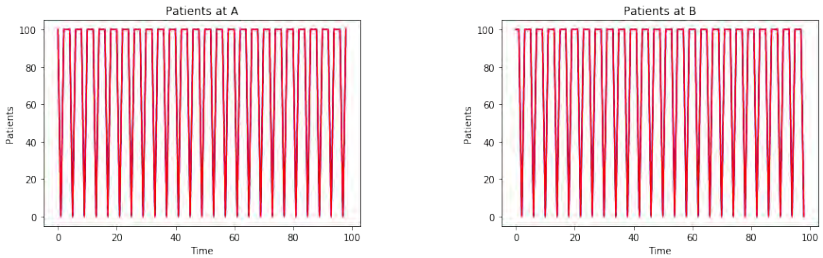
Conditional constraints are when if constraints a holds, then also constraints b must hold. For example:

$$\begin{aligned} & if: \\ & a) \quad \sum_{j \in J} a_{1j} x_j \leq b_1 \\ & then: \\ & b) \quad \sum_{j \in J} a_{2j} x_j \leq b_2 \end{aligned}$$

This can be rewritten to either-or constraints, since there are two possibilities if a holds then b holds, or a does not hold and then b does not have to hold, results in b or not a needs to hold. Which can be solved in the same way as either-or constraints.

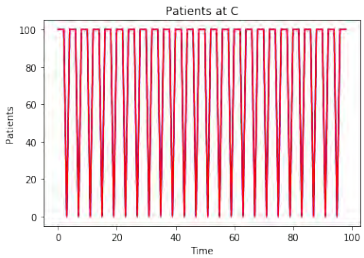
E Graphs of oscillating patient levels

Graphs of patient levels in a system with transition rates of 100%. The graphs show that the patient levels at the locations is not at a constant equilibrium, but oscillates due to the transition rates of 100%.

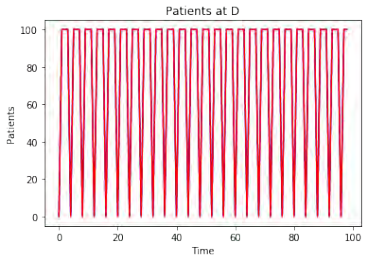


(a) Patients at A

(b) Patients at B



(c) Patients at C



(d) Patients at D

Figure 53: Plots of patients at different locations corresponding to the model seen in fig.19

References

[1] Adan, I. and Resing, J. (2002). Queueing theory.

[2] AT5 (2019). Amsterdam wordt steeds grijzer. aantal ouderen in de stad stijgt komende tien jaar hard.

[3] Ayvaz, N. and Huh, W. T. (2010). Allocation of hospital capacity to multiple types of patients. *Journal of Revenue and Pricing Management*, 9(5):386–398.

[4] Baskett, F., Chandy, K. M., Muntz, R. R., and Palacios, F. G. (1975). Open, closed, and mixed networks of queues with different classes of customers. *Journal of the ACM (JACM)*, 22(2):248–260.

[5] Bhulai, S., Koole, G., and Pot, A. (2008). Simple methods for shift scheduling in multiskill call centers. *Manufacturing & Service Operations Management*, 10(3):411–420.

[6] Bisschop, J. (2006). *AIMMS optimization modeling*. Lulu. com.

[7] Brailsford, S. and Hilton, N. (2001). A comparison of discrete event simulation and system dynamics for modelling health care systems.

[8] Burghout, W., Koutsopoulos, H. N., and Andreasson, I. (2006). A discrete-event mesoscopic traffic simulation model for hybrid traffic simulation. In *2006 IEEE Intelligent Transportation Systems Conference*, pages 1102–1107. IEEE.

[9] CBS (2020a). Aantal bewoners van verzorgings- en verpleeghuizen 2019.

[10] CBS (2020b). Aantal bewoners van verzorgings- en verpleeghuizen 2019.

[11] CBS (2020c). Ziekenhuisopnamen; diagnose-indeling ishmt, regio.

[12] Cormen, T. H., Leiserson, C. E., Rivest, R. L., and Stein, C. (2009). *Introduction to algorithms*. MIT press.

[13] Dangerfield, B. (1999). System dynamics applications to european health care issues. *Journal of the Operational Research Society*, 50(4):345–353.

[14] Ferguson, R. O. and Sargent, L. F. (1958). *Linear programming*, volume 19. McGraw-Hill.

[15] Garssen, J. (2011). *Demografie van de vergrijzing*. Centraal Bureau voor de Statistiek Den Haag.

[16] gemeente Amsterdam (2020). Data en informatie.

[17] Gilliam, R. R. (1979). An application of queueing theory to airport passenger security screening. *Interfaces*, 9(4):117–123.

[18] Green, L. V., Kolesar, P. J., and Whitt, W. (2007). Coping with time-varying demand when setting staffing requirements for a service system. *Production and Operations Management*, 16(1):13–39.

[19] Heidemann, D. (2001). A queueing theory model of nonstationary traffic flow. *Transportation Science*, 35(4):405–412.

[20] Jun, J., Jacobson, S. H., and Swisher, J. R. (1999). Application of discrete-event simulation in health care clinics: A survey. *Journal of the operational research society*, 50(2):109–123.

[21] Koizumi, N., Kuno, E., and Smith, T. E. (2005). Modeling patient flows using a queuing network with blocking. *Health care management science*, 8(1):49–60.

[22] Krumke, S. O. (2006). Integer programming: Polyhedra and algorithms. *Course notes*, page 23.

[23] Kulkarni, V. G. (2010). *Introduction to modeling and analysis of stochastic systems*. Springer.

[24] Lakshmi, C. and Iyer, S. A. (2013). Application of queueing theory in health care: A literature review. *Operations research for health care*, 2(1-2):25–39.

[25] Palmer, G. I., Harper, P. R., and Knight, V. A. (2018). Modelling deadlock in open restricted queueing networks. *European Journal of Operational Research*, 266(2):609–621.

[26] Robinson, S. (2005). Discrete-event simulation: from the pioneers to the present, what next? *Journal of the Operational Research Society*, 56(6):619–629.

[27] SIGRA (2017). 'krakende ketens in de zorg voor kwetsbare ouderen. verbeter de zorg, begin bij jezelf!'. stedelijk advies amsterdam. knelpunten en oplossingsmogelijkheden (in dutch).!

[28] Sterman, J. D. (2001). System dynamics modeling: tools for learning in a complex world. *California management review*, 43(4):8–25.

[29] Vektis (2021). Feiten en cijfers over ouderenzorg.

[30] Wolstenholme, E. F. (1993). A case study in community care using systems thinking. *Journal of the Operational Research Society*, 44(9):925–934.