Beyond Static Parameters: Adaptive Simulated Annealing in Practice

A case study on environmentally-aware lockage scheduling for the North Sea Locks

Jasper Heijne

June 3, 2026

Beyond Static Parameters: Adaptive Simulated Annealing in Practice

THESIS

submitted in partial fulfilment of the
requirements for the degree of

MASTER OF SCIENCE

in

COMPUTER SCIENCE

by

Jasper Heijne
born in Haarlem, the Netherlands

Algorithmics Research Group
Department of Optimisation
Faculty EEMCS, Delft University of Technology
Delft, the Netherlands
www.ewi.tudelft.nl

Macomi BV
Westblaak 33
Rotterdam, the Netherlands
https://macomi.nl/

Cover picture: The sealocks in Terneuzen [1].

Beyond Static Parameters: Adaptive Simulated Annealing in Practice

Author: Jasper Heijne
Student id: 5247721

Abstract

The North Sea Locks connect the saline Western Scheldt to the Terneuzen–Gent canal, requiring efficient scheduling of vessels. While current scheduling models minimise vessel delays, lockages also cause freshwater loss and salt intrusion into the canal, negatively affecting drinking water, agriculture, and the surrounding ecosystem. This thesis extends an existing simulated annealing scheduling model with water loss and salt intrusion objectives, and evaluates adaptive hyperparameter mechanisms to reduce the burden of empirical tuning. The results confirm a direct tradeoff between the environmental impact and vessel delays; incorporating environmental objectives yields significant reductions in freshwater loss and salt intrusion at the cost of increased vessel delays. Among the adaptive approaches, those guided by domain-specific information, such as tidal conditions and salinity levels, show performance improvements. General approaches from the literature, specifically the Modified Lam Annealing schedule and memory-based neighbourhood selection, do not transfer effectively to the North Sea Lock scheduling problem. These findings suggest that for domain-specific optimisation problems, problem-specific adaptive mechanisms outperform general-purpose ones, and that meaningful environmental gains in lock scheduling are achievable through optimisation alone.

Thesis Committee:

Chair:	Dr. N. Yorke-Smith, Faculty EEMCS, TU Delft
University supervisor:	Dr. N. Yorke-Smith, Faculty EEMCS, TU Delft
Company supervisor:	Ir. P. Kołodziejczyk, Macomi
Committee Member:	Dr. V. Reppa, Faculty of Maritime and Transport Technology, TU Delft

Preface

Throughout my studies in computer science, I discovered my passion for solving puzzles. This thesis provided an opportunity to pursue that passion in a professional setting, at the intersection of algorithmic research and a real-world societal challenge. I am grateful to have conducted this work together with Macomi and TU Delft's Algorithmics research group. I would like to preface this thesis by thanking everyone who supported this project.

Specifically, I would like to thank Neil Yorke-Smith and Paweł Kołodziejczyk, my supervisors during this thesis. Thank you for your guidance, constructive feedback, and advice throughout this thesis. While I enjoy working independently, in moments of insecurity, your support and guidance greatly improved my confidence and the quality of this thesis.

Furthermore, I would like to thank all colleagues at Macomi for thinking along with me on my thesis, for answering my questions when they arose, and, maybe most importantly, for the fun game nights with excellent craft beers and pizza. I believe the alternation between writing my thesis and having fun outside working hours has greatly improved my experience of writing a thesis and, with that, the quality of my thesis.

On that note, I would like to thank my amazing family and friends for their support and for the needed distractions during the past months. You helped me keep this period light-hearted and fun, even during the more challenging stages of writing this thesis.

Lastly, I would like to thank the professionals willing to assist me in learning the non-computer science background necessary for this thesis. Specifically, Bouke Biemond, David Wüthrich, Floor Bakker, and Mark Voorendt from TU Delft for answering my questions on lock functionality and for pointing me toward valuable sources outside my familiar research domain. From *Rijkswaterstaat* and *Departement Mobiliteit en Openbare Werken*, I would like to thank Inge van Tongeren and Maarten Deschamps for meeting with me and helping me understand the specific locks in Terneuzen.

I started this thesis motivated by a passion for solving puzzles; I conclude it with an even greater appreciation for the people and disciplines that made solving this particular puzzle possible.

Jasper Heijne
Delft, the Netherlands
June 3, 2026

Contents

List of Figures

Chapter 1

Introduction

This chapter introduces the motivation and context for this thesis. We describe the problem of water loss and salt intrusion of the North Sea Locks and provide an outline for solving this issue. We also give the structure of this thesis by providing a brief overview of each chapter.

The Dutch and Belgian economies are experiencing growing demand in cargo transport, accompanied by a modal shift towards inland navigation, which leads to more and larger vessels arriving in Dutch and Belgian ports [3, 4]. In Terneuzen, one of the largest locks in the world was opened in 2024 [1]. The lock facilities in Terneuzen allow vessels from the North Sea to go to the port of Gent. Given the economic importance of minimising vessel travel times, efficient scheduling of these cargo vessels is essential. A bottleneck in scheduling vessels through the Terneuzen–Gent canal is planning these vessels through the North Sea Locks. To make this schedule efficient, Macomi, a Dutch software company specialised in maritime simulation, has developed a model which minimises vessel delays in the lockage planning. However, vessel delays are not the only variable at stake in lockage planning. Lockages also have large consequences for the surrounding water. As vessels travel through the locks, water at the side with a higher water level is lost to the lower level, and the waterbodies on either side of the locks mix. The loss of water and mixing of waterbodies have negative impacts on available drinking water, agriculture, industry, and the nature surrounding the North Sea Locks.

Algorithmic approaches exist to model vessels moving through the locks, often referred to as the *lock scheduling problem*. Macomi's simulation model uses simulated annealing (SA) to minimise the vessel delays. SA is dependent on hyper-parameters. The algorithm's performance is highly dependent on selecting the correct hyper-parameters [5]. Previous research has shown that empirically selecting hyper-parameters can lead to weaker-performing models [6]. Therefore, it is crucial to improve Macomi's current parameters.

Lockage scheduling is understudied; research focuses on more general scheduling problems. The environmental impact, although important [7, 8, 9], of lockage scheduling is often ignored. There is a direct tradeoff between the freshwater loss and salt intrusion, and the vessel delays in lockage scheduling [10]. This tradeoff can be optimised. Optimising the efficiency of lock scheduling is referred to as the *lock scheduling problem*. Since lockage scheduling research is sparse, research uses similarities to more general scheduling problems, which are studied thoroughly [11, 12, 13]. This research shows that the lockage scheduling problem can be solved to optimality, but this requires excessively long runtimes. Therefore, the necessity for metaheuristic approaches, such as SA, is highlighted.

SA is widely studied. Research proves the importance of using correctly tuned hyper-parameters [5]. To alleviate this prerequisite, there are adaptive SA approaches which modify the hyper-parameters based on feedback during the search. This feedback can be based on problem-specific data. This is rarely studied, as such approaches are not generalisable. However, the sparse research using problem-specific information to guide SA finds performance increases [14, 15].

The key hyper-parameters of SA are the evaluation function, cooling schedule, and neighbourhood control. The **evaluation function** is the goal to optimise. This function cannot be optimised, as there is no way to evaluate its quality. However, the adaptive approach can make the weights in the multi-objective evaluation function reactive to the dynamic lockage environment [15]. The **cooling schedule** can be made adaptive using the well-established Modified Lam annealing algorithm [16]. This algorithm shows promising results over a wide range of problems [17, 18, 19]. Therefore, we evaluate whether its strong performance transfers to the lockage scheduling problem. The selection of new mutations of a current solution is called **neighbourhood control**. Literature proposes the idea of a memory which stores what mutation operators work well historically [20, 21]. We adopt this method to Macomi's model.

Although much research into the general scheduling problem has been done, the lockage scheduling problem is overlooked. The sparsely available research is aimed at general algorithms, relying on minimal domain-specific data. There is an especially small amount of research considering the environmental effects of lockage planning. Research in different domains shows that using problem-specific information can cause better performance. We aim to extend research into lock scheduling by using domain-specific information to guide the SA.

To alleviate the importance of selecting the correct empirical hyper-parameters for SA, we explore the adaptive SA field. Literature in this field provides general approaches for SA [16, 20, 22]. These approaches show strong performance on generic test classes, but their adaptability to lock scheduling is unknown. We aim to evaluate these general algorithms on the North Sea Lock scheduling problem.

The aim of extending Macomi's algorithm is to provide environmental awareness in lock scheduling with environmental variables. We will then evaluate adaptive approaches to the SA algorithms, which aim to reduce the significance of the empirically selected SA parameters. This leads us to the following research question investigated in this thesis:

What is the effect of adaptive Simulated Annealing parameters on the minimisation of the environmental impact of the lockage scheduling problem of the North Sea Locks?

To answer this question, the following sub-questions are answered:

1. What environmental variables are crucial in the cost function of a lockage?
2. What is the effect of extending a Simulated Annealing lock-scheduling algorithm with environmental variables?

3. What is the effect of making the hyper-parameters of the Simulated Annealing lock scheduling algorithm adaptive?

We answer these research questions by exploring related work to find the important variables in lockage scheduling. We add these variables to the model and compare the effects on the water loss and salt intrusion between the extended model and the original model. Then, for each of the SA parameters, we explore an adaptive approach, from literature or based on problem-specific data, to see if the adaptive SA approaches perform well on the North Sea Lock scheduling problem and to see if using lock scheduling-specific data can improve the SA algorithm.

The remaining structure of this thesis is as follows. Initially, Chapter 2 provides the background of the North Sea Lock situation. We discuss the functionality of locks and how this impacts the water surrounding the locks. This chapter also gives the necessary background in SA, which is the algorithm used by Macomi to plan lockages. This thesis extends this algorithm.

Chapter 3 describes the problem of the North Sea Locks in more detail. We discuss the environmental impact that lockages have, especially under drought conditions. Furthermore, we elaborate on Macomi's algorithm to schedule lockages and highlight possible improvements of the empirically selected SA hyper-parameters.

In Chapter 4, we discuss related work on lockage scheduling. We show that research on the environmental impact of lockage scheduling is sparse, and we draw similarities to similar scheduling problems. We go over research regarding adaptive SA, which alleviates the difficult task of empirically selecting strong parameters for SA. We discuss two general approaches from the literature, which we adopt for lockage scheduling, and discuss adaptive parameters based on problem-specific data.

Chapter 5 describes our approach for adding water loss and salt intrusion variables to the SA model, forming the baseline for further experiments. Then, we describe the adaptive weights, temperature, and neighbourhood algorithms, leading to the final *fully adaptive* SA approach.

Then, in Chapter 6, we give the outline for the experiments to evaluate each algorithm. We perform twenty runs of each adaptive configuration and compare them against an equal number of runs of the baseline. We test if the new approaches significantly outperform or underperform the baseline.

The experiment results are provided and discussed in Chapter 7. We observe that adding water loss and salt intrusion variables to the SA algorithm reduces their negative effects at the cost of increased vessel delays. The adaptive approaches guided by problem-specific data show the clearest improvements. The adaptive SA algorithms directly from the literature do not show improvements.

Chapter 8 gives the conclusion of this thesis and provides guidance for future research. We show that the environmental impact, in the form of water loss and salt intrusion, of a lockage schedule can be reduced by reasoning over the environmental parameters and efficient scheduling, at a cost of increased vessel delays. We find that general all-purpose adaptive SA solutions are not always the best solution for practical problems. Algorithmic approaches using the problem-specific data of the problem under optimisation can offer better-performing solutions.

Chapter 2

Background

This chapter gives the required background to understand the main body of this work. We discuss the basic functionality of a lock to give some background to the domain we are working in. Lockages influence, and are influenced by, the water surrounding the lock, so we elaborate on the effects of lockages. Then, we give the necessary background in SA and Macomi's simulation model, which uses SA.

2.1 North Sea Locks

The North Sea Locks consists of 3 locks: the New Lock, for the largest sea vessels; the West Lock, for sea-going ships; and the East Lock, for inland shipping [1]. The locks differ in size and specific purpose, but they serve the same general purpose: to transfer vessels between different water levels on either side of the locks.

Although lockages serve differing general purposes, vessels can be scheduled in any lock that fits them. If space allows it, multiple vessels can be scheduled in a single lockage. This constitutes the lockage scheduling problem, which describes the problem of planning vessels through the locks efficiently.

Besides the vessels, the locks subsequently influence the transfer of water. Generally, the water level at the Western Scheldt side of the lock is lower than the level of the canal water. The water level of the canal is maintained at a fixed level of $+2.10$ meters NAP[1]. There are small fluctuations around this water level of ± 5 cm. The water level is kept at this level to ensure vessels can safely navigate the canal at all times [23]. The water level at sea is dependent on tides, so it oscillates with a high variance. The water level at sea oscillates between approximately $+3.0$ m NAP and -2.2 m NAP.

Locks can be enhanced with mechanical additions to reduce water usage or limit the entry of high salinity water. Examples of this are a water-saving basin, a bubble screen, or a sill [24]. However, since this work considers the specific locks in Terneuzen, the lock mechanisms are fixed and therefore not a variable in an optimisation algorithm. The addition of new mechanics to the existing locks is deemed outside the scope of this work.

[1]NAP: *Normaal Amsterdams Peil*, is the reference water level of 0 *m* in the Netherlands.

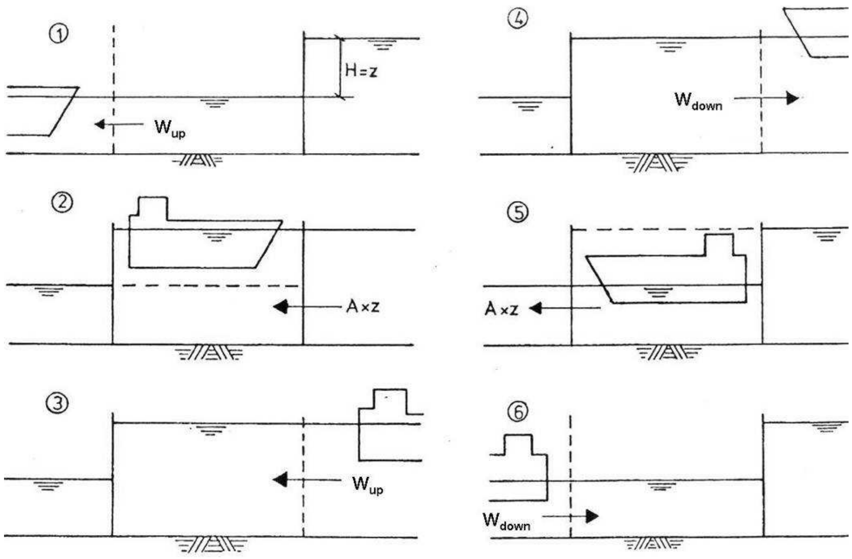

Figure 2.1: Water loss of an upstream and downstream cycle of a lock [2].

2.2 Water Flow

Water flows through a lock from the higher water level to the lower water level, as depicted in Figure 2.1. To increase the level of the water inside the lock, water from the higher water level is released into the lock, as shown in the left subfigures. To lower the water level inside the lock, water is released from inside the lock to the lower water level, as shown on the right side of the figure. A lockage causes the loss of water from the higher water level and causes mixing of the water on both sides of the lock.

2.2.1 Water Loss

The water level in the Western Scheldt is generally lower than the Terneuzen–Gent canal water. This means that a full lockage cycle loses canal water to the Western Scheldt. When the difference in water level is larger, the amount of water lost from the canal is also greater. To compute the amount of water lost in a levelling cycle, we can use this formula [2]:

$$W_{loss} = A \cdot z + W_{up} - W_{down}$$

Where:

A	horizontal area lock chamber
z	difference in water level, water head
W_{up}	amount of water which is displaced by vessels going upstream
W_{down}	amount of water which is displaced by vessels going downstream

This formula considers a lockage as a full cycle consisting of four phases:

1. Opening the Lock doors to the Western Scheldt, to allow vessels to leave and enter the lock to and from that side.

2. Closing the lock doors and levelling the lock chamber's water level to the Terneuzen–Gent canal level.

3. Opening the Lock doors to the Terneuzen–Gent canal, to allow vessels to leave and enter the lock to and from that side.

4. Closing the lock doors and levelling the lock chamber's water level back to the Western Scheldt water level.

In Macomi's model, a lockage is considered one-sided; a lockage is either phases 1, 2, and 3 for upstream lockages (shown on the left side in Figure 2.1), or phases 3, 4, and 1 for downstream lockages (shown on the right side in Figure 2.1, labelled 4, 5, and 6 respectively). Water loss is considered over a full lock cycle, meaning two lockages by Macomi's lockage definition. Therefore, we split the water loss computation over two lockages, giving the following two formulas:

$$W_{lossUp} = A \cdot z + W_{up}$$
$$W_{lossDown} = -W_{down}$$

The usage of these formulas is further discussed in Section 5.1.

The relevant net water loss is the difference between the water discharged through the lock to the Western Scheldt and the water discharged into the Terneuzen–Gent canal. We consider water loss as the total water discharged through the North Sea Locks, as this is a direct consequence of the lockage schedule, independent of historical data. The historical discharge data reflects past lock operator decisions, dependent on conditions relevant at that time. Using this data to determine the water loss would assume their decisions are independent of the North Sea Lock operations, which is not the case. An effect of our definition is that a lockage's water loss is large, while part of the water loss is natural discharge from the locks surrounding Gent. This part of the water loss has a net loss of zero, as there is an equal amount of water discharged from Gent as discharged through the locks. This inflated water loss amount does not influence the comparison of lockage schedules.

2.2.2 Salinity

The high-salinity water from the Western Scheldt, with high salt levels caused by the direct connection to the North Sea, is mixed into the Terneuzen–Gent canal's fresh water. A lockage mixes fresh water from the canal side of the lock and salt water from the Western Scheldt side of the lock. This mixture of fresh and salt water inside the lock is released into the water outside of the lock. When the lock door to the Terneuzen–Gent canal is opened, this means the canal's water increases in salinity level.

To compute the expected change in salinity, we use the ZeeSluisFormulering[2] (ZSF) [25]. ZSF is a model created and utilised by Deltares [26]. It is adopted in practice by Rijkswa-

[2]"Zeesluisformulering" translates to Sea Lock Formulation

terstaat[3] and used in related hydraulic engineering research [10, 24]. Performance measures show high accuracy in predicting the salt transfer through locks [27]. Based on the following input parameters, which ZSF finds crucial in the computation of the salt intrusion: the lock dimensions; the sea and canal water level and salinity; the ship volume through the lock in both directions; and the opening times of the lock doors, ZSF can compute, among other values, the amount of salt that enters each side of the lock. The lock dimensions and the water and salinity levels are publicly available from Rijkswaterstaat [28]; the other values are estimated, as described in 5.1. In our algorithm, we minimise the amount of salt entering the Terneuzen–Gent canal, as computed by ZSF.

ZSF is an elaborate research into a simplification of the high salinity water intrusion through lockages [26]. The salt intrusion computation does not match the exact mixing process in the lock. The mixing of water inside and outside the lock chamber is complex. For the purposes of this research, the exact details are unnecessary to reach conclusions on the macro level of the salt intrusion. Based on a suggestion from the ZSF documentation, we assume the initial salinity of the lock chamber to equal the average of the sea and canal water salinities. This is as if we assume a steady state for the lock chamber salinity prior to a lockage.

ZSF's method consists of four phases: 1. levelling the lock water level to the canal water level, 2. opening the lock doors to the canal for vessels to leave/enter the lock, 3. levelling the lock water level to the Western Scheldt water level, and 4. opening the lock doors to the Western Scheldt for vessels to leave/enter the lock.

We use a per-lockage computation to compute the salt intrusion for a lockage as defined by Macomi. A lockage translates to three phases of ZSF: either phases 2 (open to canal), 3 (level to sea), and 4 (open to sea) for a lockage in the downstream direction; or phases 4 (open to sea), 1 (level to canal), and 2 (open to canal) for a lockage in the upstream direction.

Besides the difference in salinity levels between both sides of the lock, the most important factor in salt transition is the discharge of fresh water into the canal, as higher discharge means that the high salinity water is 'flushed' out of the canal [29]. ZSF does not reason about discharge, as it considers a lake as the freshwater side of the lock. Therefore, we add discharge as a separate component to the computation of a lockage's salt intrusion.

Stratification likely also plays an important role in the exact computation of the mass salt intrusion [30, 31]. These variables are excluded from this study because the stratification data are not available and are difficult to estimate. Since stratification is considered less critical for the salt intrusion computation, and we do not need the exact value of the salt intrusion to guide us towards a more environmentally friendly lockage planning, we assume complete mixing inside the lock after a cycle.

Since the sea and canal mix, other variables in the canal are also affected. For example, oxygen and acidity are affected by seawater. These variables have no critical effects on the water quality in the Terneuzen–Gent canal [32, 33] and were not included in the expert discussion to guide lock operations with a decision support system [34]. Therefore, they are deemed irrelevant for this thesis.

[3] Rijkswaterstaat are the Dutch water authorities, largely responsible for lock operations and water management of the Netherlands.

2.3 Simulated Annealing

Simulated annealing (SA) [35, 36] is an optimisation technique based on annealing in metallurgy. Annealing is a controlled way of cooling a material to influence its physical properties. The SA algorithm adapts this approach to model how quickly the algorithm narrows its search space to local solutions. This is called the cooling schedule.

SA starts with an arbitrary solution to the problem under optimisation. The algorithm proceeds to make alterations, called mutations, to the current solution to find new solutions. The quality of solutions can be measured by scoring a solution. The goal of the algorithm is to iteratively find new solutions that score better than the current solution. The cooling schedule causes the algorithm to sometimes select mutations which score worse than the current solution. When the temperature of the cooling schedule is high, the probability of selecting worse-performing solutions is higher. This allows the search algorithm to cover a broader area of the search space, avoiding getting stuck in local optima.

Scoring a solution is done using an **evaluation function**. The evaluation function is problem-specific, meaning it depends on the problem under optimisation. The evaluation function in lockage scheduling assigns a score to a schedule of lockages and vessels. The score is dependent on multiple dependent variables. In lockage scheduling, important variables are the vessel delays caused by lock waiting times, the water loss due to lockages, and the amount of salt entering the freshwater canal. The evaluation function scores the schedule based on how large the negative consequences are of each of these variables under the current schedule.

Besides the scoring function, SA has a cooling schedule and uses mutation selection. **The cooling schedule** describes at what temperature the simulated algorithm starts and how this temperature changes over time. The temperature determines, based on some predetermined acceptance probability, if worse solutions are accepted. This thesis also experiments with the stopping criteria as part of the cooling schedule. SA does not necessarily continue until the global optimum is found. The stopping criterion dictates when SA stops looking for new solutions. The stopping criterion is closely related to the temperature. When the temperature approaches zero, accepting new solutions becomes rarer, leading to eventual termination of the algorithm. **Mutation selection** describes what mutations are made to the current solution to explore new solutions, as well as what solution to start the search from.

2.4 Macomi's Model

Macomi has created a model which simulates vessels through the Terneuzen–Gent canal. Part of this simulation model is lockage scheduling, using SA. The hyper-parameters, such as evaluation function weights, cooling schedule α, and mutation operator probabilities, are selected empirically. The SA model schedules which vessels should enter which lock at a given time. This results in a schedule with lockages for a given time period. A lockage can be empty or contain one or more vessels.

2.4.1 Evaluation Function

Macomi implemented multiple objective functions to score solutions. Each of the functions prioritises a different goal to optimise. The objectives currently under optimisation are: minimising the number of lockages; minimising the total vessel delay; minimising the maximum delay of a single vessel; minimising the preference punishment scores of vessels, which are higher if a vessel is not scheduled in its preferred lock; and Macomi has a basic implementation for the minimisation of water loss. This thesis extends this list of objectives with a more extensive water loss computation and minimisation of salt intrusion. Each of the separate components is combined in a weighted sum to form the evaluation function.

The main tradeoff in lockage scheduling is vessel delay against water loss and salt intrusion, also referred to in this thesis as the environmental parameters. The minimisation of total vessel delay and maximum vessel delay will be the main drivers of the vessel delay results. The water loss and salt intrusion scorers will have the largest impact on the environmental parameters. The other scorers do impact the vessel delays and environmental variables, so we do keep them in the evaluation function, but we do not compare their direct contribution in this thesis to keep the tradeoff between vessel delay and environmental parameters clear.

Therefore, a simplified version of the evaluation function is as follows:

$$F_{\mathrm{S}} = W_{\mathrm{delay}} \cdot \sum_{l \in S} \mathrm{Delay}_l + W_{\mathrm{water\ loss}} \cdot \sum_{l \in S} \mathrm{WaterLoss}_l + W_{\mathrm{salt\ intrusion}} \cdot \sum_{l \in S} \mathrm{SaltIntrusion}_l$$

Where F_{S} is the fitness value of schedule S, and l represents a lockage in S. W represents the weight assigned to each component. The weights are selected manually before running the simulation and represent how important the associated variable is in the current simulation.

2.4.2 Cooling Schedule

Macomi uses a geometric cooling schedule, which decreases with a constant factor α every second iteration. The temperature T starts at a value of 100, α is set at 0.993. Giving the following formula for T at iteration i:

$$T_{i+1} = 0.993 T_i$$

A new solution i is accepted with the following probability:

$$P_i = e^{-\Delta_i / T_i}$$

Where Δ_i is the difference in fitness between the new solution and the current solution. Meaning that the probability of accepting a new solution becomes smaller as Δ increases, or as the temperature decreases.

Macomi's SA algorithm terminates when no improvement over the current solution is found for 1000 iterations. This leads the algorithm to exploit the found solutions, as it only terminates when it is not possible to find a mutation which improves the current solution.

2.4.3 Mutation Selection

To explore new solutions, the current solution is mutated. In lockage scheduling, mutations are changes in the assignments of vessels to lockages. The mutations are selected from one of five predetermined mutation operators:

- Move Vessel: Moves the vessel to a random existing lockage.
- Move Vessel Greedy: Moves the vessel to an existing lockage near the vessel's arrival time.
- Swap Vessels: Swaps the vessel with another vessel in a random lockage.
- To New: Creates a new lockage for the vessel at a different time in the same lock.
- Change Visit: Move the vessel to a different lock at a similar time as the original lockage.

The operators are selected by weighted random selection. The operator weights are empirically decided and fixed between simulations.

For all operations, the requirements of vessels are taken into account. Vessels which are too large for the smaller locks will not be scheduled in these locks. Vessels moving upstream cannot be scheduled in downstream lockages. An empty lockage is a candidate for removal. If an empty lockage is preceded or followed by another empty lockage, both lockages are removed. An empty lockage is also not bound by the time requirements of vessels, making them flexible between the preceding and succeeding lockages.

Chapter 3

Problem Statement

In lockage scheduling, many variables play a role. An optimal schedule should create minimal waiting time for vessels. Macomi has constructed an SA model which allows scheduling the locks based on arriving vessels to minimise their waiting time. The model also includes a basic measure to minimise water loss, but the environmental awareness of the lockage scheduling can be extended. In addition to the environmental variables, we believe improvements can be made to the optimisation algorithm. This chapter describes which variables play a crucial role in creating an optimal lock schedule. Then, we discuss how Macomi's current SA algorithm can be improved while considering the environmental variables.

3.1 Lockage Effects on Water Scarcity and Quality

The North Sea Locks in Terneuzen form a passage between higher salinity Western Scheldt water, also referred to as seawater in this thesis due to its high salinity, and the freshwater Terneuzen–Gent canal. Lockages cause freshwater from the Terneuzen–Gent canal to be lost to the sea. Lockages also cause high-salinity seawater to enter the canal. Both events have negative consequences for the maintenance of freshwater. The loss and contamination of fresh canal water should be minimised.

3.1.1 Freshwater Scarcity

Freshwater is limited in availability. It is crucial to important infrastructures, such as drinking water, agriculture, industry, and nature [8]. We should aim to reduce wasting freshwater, especially during droughts, caused by low discharge from the rivers flowing into the Terneuzen–Gent canal and by high temperatures with little rainfall. Such conditions are strengthened by global warming [37], making this an intensifying problem.

Hardly any water is lost when a lockage is scheduled at a time when the water levels of the Western Scheldt and the Terneuzen–Gent canal are equal. However, one lockage of the New Lock – the largest of the North Sea Locks – can lose over $100,000,000$ litres[1] (equivalent to 40 Olympic swimming pools) of fresh water when scheduled at an inefficient moment. Such losses should be avoided.

In the Netherlands, safety measures for important hydraulic structures and vulnerable nature are deemed paramount during freshwater shortages [38]. A shortage of freshwater can cause damage to dikes, which reduces their structural integrity. This could lead to malfunctions during periods of high water [39]. Vulnerable nature can also suffer from freshwater shortage. Nature in the Netherlands, and surrounding Terneuzen, is very diverse and unique in biodiversity and ecological values. This nature is under pressure from, among others, nitrogen deposition, urbanisation, and desiccation; amplified by global warming [40]. Efficient water management is crucial in sustaining these locations.

Other infrastructures heavily reliant on freshwater are drinking water facilities and energy suppliers [38]. Both these facilities fall under the critical infrastructure of the Netherlands, meaning disruptions can be detrimental to the national and European economy and safety [41]. Careful management of water loss is crucial to protect these infrastructures.

Drought also has large consequences on inland vessel transport surrounding locks. We further discuss the direct consequences on lock operations in Section 3.1.2. Besides direct restrictions on lock operations, low discharge periods cause a lower water level, forcing limitations on vessel cargo loads. This means more vessels are required for the same cargo. Together with a narrowed navigation channel, this leads to increased vessel traffic and thus longer waiting times at the locks [42].

Periods of drought are also strongly connected to the salinisation of the freshwater canal. During high discharge, high salinity water is 'flushed' out of the freshwater canal [29]. Low water allows the high salinity water to further intrude inland. We further discuss salinisation in Section 3.1.3.

The water losses can be minimised by efficient scheduling of lockages. A method to minimise the water loss is to schedule lockages at high tide moments. During high tide, the sea level and the canal water level differ only slightly, reducing the water lost. However, high tide only occurs approximately two times a day. Scheduling lockages only at such times can mean long waiting times for vessels arriving at low tide. The locks could also become overcrowded at high-tide moments when many vessels are scheduled in lockages at those times. This can cause large delays for vessels [3].

Another method for reducing water loss is by reducing the number of lockages by combining multiple vessels in a single lockage. This causes delays for vessels that have to wait for other vessels to arrive at the lock or to leave the lock post lockage.

Since Terneuzen has 3 locks, varying in size, another possibility to reduce water loss is to schedule vessels to a smaller lock. The locks have requirements for maximum vessel size [43], so switching to another lock is not always a viable option. Rescheduling vessels to smaller locks can also cause the smallest lock to be overloaded with vessels, again leading to vessel delays.

[1] Approximated by: Water loss = surface area New Lock $\times$ greatest difference in water level between sea and canal in November 2025 $= 23,485 \times 4.44 = 104,273.4 m^3$.

The delay of vessels and the minimisation of freshwater loss are dependent variables [10]. This highlights the importance of an optimisation model to find a solution which optimises both.

3.1.2 Lock deactivation

The water level in the Terneuzen–Gent canal has to be maintained at a minimum height [23]. This assures vessels can safely navigate the canal. During periods of drought, meaning the discharge into the Terneuzen–Gent canal is consistently smaller than the discharge through the locks to the Western Scheldt, the locks can be deactivated, or their activity can be limited to reduce water loss and maintain an acceptable water level [44, 45].

We observe a direct consequence of low water on the operation regulations of the locks in Terneuzen. The locks have previously been disabled for two to four hours around low tide to avoid large water losses [46, 47]. The time windows of the disabled locks can be improved using efficient lockage scheduling.

Deactivating locks or limiting their activity can cause significant vessel delays; force more strict cargo regulations; or cause the rescheduling of vessels to a different port [48]. Deactivating locks should be avoided at all costs, highlighting the importance of regulating the canal water effectively.

With the rise of global warming, drought occurrences arise more frequently [49]. The lock scheduling in Terneuzen should be robust against such occurrences, highlighting the importance of a well-performing planning tool to keep the locks active.

Conversely, in periods of high discharge, locks can also be disabled. To regulate the abundance of water in the canal, one of the locks can be left open to release water to the Western Scheldt [34]. In periods of high discharge or when such periods are approaching, having more lockages is beneficial. The 'loss' of water to the Western Scheldt is considered beneficial to reduce the water level in the canal. Efficient lock operations during these high-discharge periods can also be scheduled using an optimisation algorithm.

3.1.3 Increased Salinity Level

While lockages cause a loss of freshwater, the high salinity seawater also flows in the reverse direction. This has negative effects on agriculture, ecosystems, industries, and drinking water.

The ecosystems surrounding the Terneuzen–Gent canal rely on fresh water to maintain balance. Many plants in such a freshwater ecosystem have a maximum amount of salt they can handle in water. An increase in the salinity level of the water can cause negative effects on certain more vulnerable plants, or even cause species to go locally extinct [50]. Such biodiversity loss has negative effects on the ecosystem functioning and resilience [51]. A similar concern holds for agriculture that depends on the Terneuzen–Gent canal for irrigation. Crop yields and soil quality might suffer when water in the canal becomes too saline [52].

High salinity water also negatively affects industries by increasing metal corrosion. Industries using water for cooling, agriculture using water for irrigation, and water infrastructure all deal with an increased rate of metal corrosion when in contact with water. Water with higher levels of salt leads to increased reactivity with oxygen, causing faster oxidation [53]. This means faults and, therefore, maintenance occur more often [54].

Water from rivers and canals in the Netherlands is directly and indirectly, through groundwater, filtered and used as drinking water. Water with increased salinity levels is harder for filtering systems to clean, and water filtering industries have even been forced to shut down the intake of water due to increased salinity [55].

Global warming causes more severe and widespread periods of drought [3, 49]. In periods of drought and low discharge from rivers, it is shown that the salinity of inland water, such as the Terneuzen–Gent canal, is increased [29, 56]. This causes the aforementioned negative side effects of high-salinity water to be amplified in periods where freshwater is already scarce.

It is important to minimise the salinisation of the freshwater Terneuzen–Gent canal. Macomi's current model does not consider salinisation as one of the variables in lock scheduling to minimise. Extending the model with a salinity measurement could extend the breadth of the model.

3.2 Existing Model

Macomi has implemented a model which uses SA to schedule vessels through the locks in the Terneuzen–Gent canal, as described in Section 2.4. The model minimises the delay of vessels and contains a basic function to compute water loss. However, this computation can be extended, and the salt intrusion is not yet considered in optimisation. Furthermore, Macomi's SA algorithm can potentially be improved by making the hyper-parameters adaptive to the specific problem instance.

3.2.1 Environmental Variables

The inclusion of environmental variables is not complete in Macomi's model. The current computation of the water loss of a lockage does not consider water displacement by vessels. The model also does not account for the salt intrusion, which is caused by lockages.

Vessels inside a lock during the lockage displace water in the lock chamber. As described in Section 2.2.1, this causes extra water loss for vessels moving upstream, while it reduces the amount of water loss for downstream lockages [2]. Downstream lockages with multiple vessels should therefore use less water than empty downstream lockages, under the same circumstances. This consideration is not implemented in Macomi's current model.

Besides the litres of water lost in a lockage, another important factor determining the cost of lockages is the salinity of the canal water. Macomi's current model does not consider the flow of high salinity water to compute the cost of a lockage. Adding such a parameter would broaden the model to include all crucial environmental considerations.

3.2.2 Fixed Simulated Annealing Parameters

Macomi's SA hyper-parameters are fixed between simulations. This means that for different optimisation problems, the same parameters are used. While these generalised parameters can perform well for certain problem instances, they may lack adaptability to specific ones [5]. The exact empirical parameter selection methodology is not documented. Empirically selecting SA parameters has been shown to lead to weaker performing models [6]. As operator requirements change over time, this leads Macomi to time-consuming recalibration of parameters that may no longer perform optimally for the current problem.

In the current model, the weight of the water loss needs to be set by the operator prior to running each simulation. This has a large impact on the model's outcome, as shown in previous experiments by Macomi. However, the weight is represented by a dimensionless scalar, which does not have a clear meaning for inexperienced users of the system. Reducing the cruciality of the weight, by making it adaptive, would reduce the experience required to use the system.

Adapting the SA parameters based on problem-specific variables can improve the possibly outdated empirically selected parameters. It also leads to an SA approach tailored to specific problem instances. We expect this could improve the performance of the optimisation algorithm.

Chapter 4

Related Work

Optimisation is a thoroughly studied domain in computer science. This chapter discusses SA practices applied to the scheduling of lockages. We discuss two research fields. First, research on lockage scheduling and the parameters that influence the cost of lockages. Then, SA and its adaptive variants.

4.1 Lockage Scheduling

The lockage scheduling problem is understudied. Most work considers the scheduling of vessels, with the lock(s) as a secondary focus. The locks are mainly seen as an obstacle to the vessel schedule. Therefore, the more extensive lockage variables, such as their environmental impact, are often not considered. We discuss several studies related to the lockage scheduling problem to get an idea of the current state of the academic landscape.

4.1.1 Environment Parameters

Work regarding lockage scheduling and its environmental impact is limited. We discuss research regarding the environmental impact of lock operations. These papers highlight the importance of this research field and provide insight into what parameters are considered crucial in the environmental conditions of a lock.

Wu et al. [7] highlight the importance of environmental awareness in lock scheduling. They provide a solution for the *Green Lock Scheduling Problem* (GLSP), where they focus on minimising the environmental impact of lockage scheduling. Their work approaches the GLSP by minimising vessels' carbon emissions. Despite the focus on minimising the environmental impact of lock scheduling, the water loss and salt intrusion are not considered.

A study by Bakker and Van Koningsveld [10] explores the connection between the number of lockages and the salinity of the water in the Terneuzen–Gent canal. This work shows that clustering vessels compactly in a lockage to reduce the number of lockages limits saltwater intrusion, but it introduces high vessel delays. Their study uses a theoretical model with simplified assumptions and simulations in TU Delft's OpenTNSim to simulate vessel behaviour. While their work models the physical consequences of lock usage, it does not implement an optimisation framework. We build upon their findings and use SA to find improved schedules with reduced environmental impact for the locks in Terneuzen.

Research on the lock situation in Bremen considers water loss and energy usage in their lockage scheduling, emphasising the importance of these parameters in lockage scheduling [57]. In the described lockage process, a water pump is used to pump water back into the port area. The energy usage of this pump can be minimised if the water level in the port area remains at an acceptable level, which can be achieved by locking at moments when the (sea)water level is high. Indicating similar requirements to our water loss minimisation. To model this system, the authors propose a multi-agent system with an agent for each individual variable in the scheduling problem, such as each vessel, each lock, or the harbour basin. The paper suggests a model to solve the scheduling problem, while focusing on minimising the vessel waiting times and reducing energy usage. Simulations with the model have not yet been performed, but are expected to show how effective water management with lockages can be maintained. The multi-agent system approach is highly dependent on the specific lock agents, making it complicated to adapt to other lock facilities. We expect SA to give a more general solution to lockage scheduling and will show the tradeoff between vessel delays and environmental impact with experimental results.

4.1.2 The Scheduling Problem

The lockage scheduling optimisation problem is sparsely studied [58], as it is a niche optimisation problem. However, the lockage scheduling problem is closely related to more general forms of NP-hard scheduling problems [11]. These general scheduling problems are studied more intensively [59].

Passchyn [12] did extensive research into lock scheduling. In his work, he draws the similarities between lock scheduling and the machine scheduling problem, an NP-hard optimisation problem which is studied more extensively. In this comparison, locks are mapped to machines, and vessels are the jobs that these machines have to process. This work discusses multiple versions of the lock scheduling problem, such as a single lock, multiple locks in a sequence, or multiple locks in parallel. For the latter, which is most similar to the situation in Terneuzen, he proposes a dynamic programming approach which schedules vessels with a runtime of $O(mn^m)$, for m vessels and n chambers.

A similar mapping from the lockage scheduling problem to machine scheduling is used in work by Verstichel et al. [13]. In this work, they construct a mixed integer linear programming model which can be solved to optimality on small problem instances.

Both papers show the high complexity of the lockage scheduling problem and show that it can be solved to optimality for smaller problem instances. However, the environmental impact of lockages is not considered despite its crucial role in lockage planning, and the proposed dynamic and mixed integer linear programming approaches are not feasible for the large and complex real-world situation in Terneuzen. A metaheuristics-based optimisation approach, such as SA, is therefore necessary to get good results in an acceptable time for larger problem instances.

A study by Zhang et al. [60] on the two dams of the Three Gorges Project in China uses SA, in combination with local search, to demonstrate the feasibility and optimality of a lockage schedule of the five locks in the project. In their approach, they minimise vessel tardiness, minimise empty lockages, and maximise the balance between the workload of the locks. Minimising empty lockages can be considered a basic form of water loss and salt intrusion minimisation. To solve the lock scheduling problem, they draw similarities with the *Flexible Manufacturing System* (FMS) problem, which is a more extensive version of the machine scheduling problem mentioned previously, yet it is less complex than the lockage scheduling problem. The authors highlight that their model is limited to deterministic navigation conditions and emphasise the need for a model to adapt to the stochastic dynamic environment of locks. Their model does not account for direct water loss or salt intrusion parameters. Therefore, the trade-off between increased vessel delays and the reduction of environmental impact is not discussed.

4.2 Adaptive Simulated Annealing

Simulated Annealing is a widely adopted optimisation method, due to its straightforward algorithm. The initial version is attributed to Kirkpatrick et al. [35]. It is studied extensively in a broad domain. This section discusses research with SA using adaptive parameters, which alleviates the burden of manually selecting good parameters.

Previous work on adaptive parameters for SA shows that they can produce accurate results, competitive with well-crafted empirically selected parameters [17, 61]. We discuss several papers which evaluate adaptive parameters in scheduling optimisation tasks.

4.2.1 Domain Driven

This thesis aims to improve lockage scheduling for the North Sea Locks. This is a specific problem. Therefore, besides the general adaptive SA approaches we will discuss, we can consider utilising the domain-specific information to guide the SA parameters. This section discusses related work exploring these domain-driven adaptive SA approaches. This research field is mainly focused on neighbourhood operations and initialisation procedures. We take inspiration from these domain-specific approaches for our implementation of an adaptive evaluation function and adaptive neighbourhood control.

Musharavati and Hamouda [14] exploit auxiliary knowledge in manufacturing process planning. Multiple SA algorithms are extended with heuristic and meta knowledge of the planning problem and compared to a basic version of the SA algorithm. The version extended with auxiliary knowledge shows significant improvements over the basic algorithm. This work uses domain-specific knowledge on scheduling, such as generating a more efficient initial solution and reusing previous schedules in new solutions. These heuristics are based on neighbourhood operations (i.e. the generation of a new solution). The heuristics suggested are for process planning and are not applicable to the water variables in lock scheduling. Further parameters beyond neighbourhood control are not considered.

Domain-specific knowledge can also be used to inflate or deflate evaluation scores. Based on knowledge of scheduling, major inefficiencies in a solution can be detected, guiding the mutation selection to avoid such solutions [15]. This algorithmic approach also leads to improved performance.

The domain-specific methods are not directly applicable to the lockage scheduling problem of the North Sea locks, but show that the methodology can be effective. In our study, we will discuss the effects of problem-specific water/lock heuristics on a wider range of SA parameters to see if these adjustments follow the same trend of improving SA.

4.2.2 Evaluation Function

Macomi's SA algorithm uses a weighted sum evaluation function. The evaluation function is the objective to optimise; we have no way to score an evaluation function. Therefore, the evaluation function itself cannot be optimised. However, the function currently is static, while the lockage scheduling problem is a dynamic problem. We can make the evaluation function dynamically reactive to the lock situation, following the idea of the domain-driven approaches, which inflate evaluation weights [15].

Research on such dynamic evaluation functions is rare. Eiben et al. [62] discuss three approaches to dynamically change penalty coefficients for a different metaheuristic, the evolutionary algorithm: a function for the weights over time, feedback from the search process, and self-adapting weights. The former does not reduce the effort of empirical selection of weights; the latter is specific to the workings of evolutionary algorithms. Therefore, we adapt the evaluation function weights based on domain-driven feedback in the search process; this methodology is described in Section 5.2.

4.2.3 Cooling Schedule

A large part of adaptive SA research centres around the cooling schedule. The temperature change in standard SA follows a geometric decay function with fixed parameters. These parameters can be adapted based on the search state to form a more effective algorithm [63]. We discuss research showing the benefits of an adaptive cooling schedule for SA. Then, we describe how we selected the Modified Lam annealing approach for our cooling schedule.

Azizi and Zolfaghari [64] compare an SA algorithm with an adaptive cooling schedule against a standard SA algorithm and find that the adaptive cooling approach outperforms the baseline. In their research, they find that a combination of adaptive simulated annealing with a tabu-list is significantly superior to the baseline simulated annealing. They demonstrate that adjusting the temperature based on the search trajectory helps escape local maxima and therefore improves SA. The experiments are performed on several job scheduling problems, which have similarities with lockage scheduling. However, they remain theoretical, which means that the results on a practical problem, such as the North Sea Lock situation, are unknown.

Triki et al. [65], in their research on the cooling schedule, show empirically that most classical cooling schedules perform similarly. This indicates that the basic geometric cooling schedule functions as an interchangeable baseline. A new adaptive method is proposed that outperforms these classical approaches on the travelling salesperson problem. This shows that an adaptive cooling schedule can improve the SA algorithm. The results are based on artificial test functions, which might not translate to lockage scheduling. Therefore, this thesis adopts a better-established and more widely tested, well-performing approach called Lam annealing.

Lam annealing is a well-established approach for adapting the cooling schedule [63]. It was initially proposed by Lam and Delosme [66]. In their cooling schedule, they change the size of the neighbourhood to allow increased or decreased exploration. Since it is difficult to define a dynamic neighbourhood, their approach is modified by Swartz [16], causing the neighbourhood function to no longer be changed during search, giving the later formalised Modified Lam annealing schedule [67], which only adjusts the temperature based on the ideal Lam acceptance rate. This model has been shown to perform well over multiple domains [17, 18, 19]. We implement a computationally more efficient approach [68]. The methodology is described further in Section 5.3.

4.2.4 Neighbourhood Control

The neighbourhood control is responsible for selecting the next mutated solution based on the current solution. Creating a mutated solution is problem-specific, as a solution is defined within the problem's domain. Therefore, we explore an adaptive neighbourhood approach using domain-driven heuristics to determine how a mutation is constructed. Since these heuristics are highly dependent on the North Sea Lock situation and Macomi's implementation, there is no related work to directly follow. However, as described in Section 4.2.1, previous work in related fields has shown that the use of problem-specific knowledge can improve mutation selection. The heuristics we use for the first adaptive neighbourhood approach are based on domain knowledge and are further described in Section 5.4.2.

Possible other approaches, independent of the problem under optimisation, for adaptive neighbourhood control are the adaptive large neighbourhood search [69] or a reinforcement learning-based hyper-heuristic approach [70]. These approaches require more tuning and hyper-parameters. They are more difficult to directly adopt into Macomi's existing model. Therefore, we opt to evaluate a less complex, problem-independent, memory-based approach, which we evaluate alongside the domain-driven approach.

We adopt a strategy from the literature. Yu et al. [20] propose Adaptive Neighbourhood Simulated Annealing (ANSA) for a fleet vehicle routing problem. ANSA aims to guide the selection of the next mutation. It does so by keeping a memory of what operators reach good fitness in previous mutations. This memory can be used to influence the probabilities of selecting each mutation operator in the future. ANSA performs well on the fleet vehicle routing problem. A similar memory-based neighbourhood control also shows strong results on university course timetabling problems [21]. These memory approaches are easily generalisable to other problem classes. We will evaluate an adapted version for the North Sea Lock scheduling problem. The methodology for this algorithm is described in Section 5.4.1.

4.3 Research Gap

Throughout all academic work on lock scheduling, we see a similar pattern. The sources mainly focus on scheduling the vessels efficiently. Most models contain no parameters related to water loss or salt intrusion. If they do, they are basic indicators which do not cover the full lock scheduling problem, and the concrete effects are not discussed. Much research also focuses on modelling the lock scheduling problem. This work can be used to prove feasibility, but it does not show how the North Sea Lock scheduling problem can be solved efficiently.

Although it is known that tuning parameters is a tedious but important aspect of SA, there is a lack of experimentation with adaptive parameters. The adaptive cooling schedule has been extensively researched, providing the well-established Modified Lam Annealing approach. However, the evaluation function and neighbourhood control are more sparsely studied. Work on adaptive SA also rarely discusses domain-driven heuristics, as such algorithms are not generalisable. However, we expect that for a specific problem, such as the North Sea Lock scheduling problem, using problem-specific information to find a solution can be beneficial.

This work aims to evaluate adaptive variants of all parameters which are crucial in lock scheduling to alleviate the parameter tuning efforts. With these new algorithms, we aim to reduce the water loss and salt intrusion in the North Sea Locks to reduce the complete environmental impact. We discuss the effect of the reduced environmental impact on the lockage schedule's vessel delays.

Chapter 5

Methodology

This chapter describes how each of the adaptive parts of the SA algorithm is created. We introduce five experiments and explain how we approach their implementation. The first experiment compares Macomi's original model without water loss and salt intrusion parameters, as a baseline, against an implementation with the environmental parameters present. This experiment produces the new baseline with water loss and salt intrusion for the remaining four experiments. The following three sections explain how the SA approaches with adaptive weights, temperature, or neighbourhood control are constructed. Finally, we create a fully adaptive SA algorithm with a combination of the best performing approaches from the adaptive weights, temperature, and neighbourhood control to evaluate if their successes accumulate.

5.1 Experiment 1: Water Loss and Salt Intrusion Parameters

Based on our Background (Chapter 2) and Problem Statement (Chapter 3), we conclude that water loss and salt intrusion are the critical environment-based parameters in a lockage of the locks in Terneuzen. Therefore, we add these parameters to the SA algorithm. This extended SA algorithm will form the baseline in further experiments.

To extend the SA algorithm with water loss and salt intrusion, following the existing implementation by Macomi, we compute a fitness value per lockage. Then we extend the evaluation function – the weighted sum of fitnesses – to add the water loss fitness and the salt intrusion fitness.

5.1.1 Water Loss

Macomi has created a basic implementation to compute the water loss fitness of a lockage, following the formula:

$$V_l = L_{\text{Lock}_l} \cdot W_{\text{Lock}_l} \cdot (\text{Canal}_t - \text{Sea}_t)$$

With V_l: the volume of water lost by upstream lockage l. L_{Lock_l} and W_{Lock_l}: respectively the length and width of the lockage's lock, Lock_l. $(\text{Canal}_t - \text{Sea}_t)$: the water level of the canal minus the water level of the sea, at time t of lockage l.

This computation for V_l is extended with the water displacement from the vessels [2], as described in Section 2.2.1. We estimate the volume of the water displacement by vessels in the lockage based on the following formula [71]:

$$V_i = L_i \cdot B_i \cdot T_i \cdot C_b$$

With:

- V_i: the water displacement volume of vessel i.
- L_i: the vessel length.
- B_i: the vessel beam.
- T_i: the vessel draught.
- C_b: the block coefficient describing the hull fullness.

Since C_b for vessels is not available in the data, it is estimated by a constant value of 0.5. This simplification neglects differences in hull shape between vessels, making the estimation less precise. However, the water displacement is only a minor part of the water loss computation, meaning such an estimation is sufficient for a rough estimate. The volume of water lost to sea is transformed into a fitness value.

5.1.2 Salt Intrusion

The salt intrusion fitness value is computed using the Zeesluisformulering [25]. Most input values can be directly taken from the lockage data (i.e. lockages time and direction) or from measurements from Rijkswaterstaat (i.e. sea/canal water level and salinity).

Remaining input values are estimated. We estimate the volume of the water displacement by vessels and barges in the lockage the same way as in the water loss computation, as described in the previous section. We estimate the time the lock doors are opened based on the formula by Bakker [72]:

$$T_{\text{open}}(x) = 1.5 + 1.0 + (x-1) \cdot 5.0 + t_{\text{pass}} + 15.0 + (x-1) \cdot 15.0 + t_{\text{pass}} + 1.5$$

With $T_{open}(x)$: the time the doors are opened for a lockage with x vessels. t_{pass} is the time required for the vessels to pass through the lock (approximately one minute for the average vessel). The constants 1.5 and 1.0 represent the door opening and closing times and the preparation time, respectively. The constant 15.0 is the time allocated per inbound vessel for entering the lock and securing the vessel. Lastly, 5.0 the extra time per outgoing vessel, as a buffer waiting time for vessels leaving the lock shortly after one another.

5.1.3 Weights

To determine the weights for the water loss and salt intrusion, we perform experimental runs. We try several weights and compare the contribution of water loss and salt intrusion to the total fitness score. Based on these experimental runs, we adjust the water loss and fitness score to contribute similarly to the vessel delay, for a weight of 1.

The experiment compares several weights ranging from 0.05 to 10 against the baseline, allowing us to systematically find the preferred weight for the water loss and salt intrusion. We further elaborate on this in Section 6.2. We give water loss and salt intrusion the same weight throughout the experiments, as we deem them equally important in this thesis, and this limits the number of weight configurations to test.

5.2 Experiment 2: Adaptive Weights

A core element of SA is the evaluation function. Since the evaluation function is the goal to optimise, the function itself cannot be optimised. However, to make the cost function adaptive, we can make the weights adapt based on the state of the lock environment at the time of each lockage. In Macomi's implementation, the evaluation function is a sum of weighted components. A simplified version, with the extension of the environmental parameters, is:

$$F_{\mathrm{S}} = W_{\mathrm{delay}} \cdot \sum_{l \in S} \mathrm{Delay}_l + W_{\mathrm{water\ loss}} \cdot \sum_{l \in S} \mathrm{WaterLoss}_l + W_{\mathrm{salt\ intrusion}} \cdot \sum_{l \in S} \mathrm{SaltIntrusion}_l$$

Where F_{S} is the fitness value of schedule S, and l represents a lockage in S. W represents the weight assigned to each component.

Optimising the weights W_i would lead to $W_i = 0$ or $W_i = -\infty$, depending on the bounds of the weights, since this leads to the lowest value for the evaluation function. However, this would not allow SA to find new solutions that improve the lock schedule. Therefore, SA would no longer work.

To make the cost function adaptive, we can make the weights adapt to the state of the lock environment at the time of a lockage. This results in the following evaluation function:

$$F_{\mathrm{S}} = W_{\mathrm{delay}} \cdot \sum_{l \in S} \mathrm{Delay}_l + \sum_{l \in S} w_{\mathrm{water\ loss}}(t) \cdot \mathrm{WaterLoss}_l + \sum_{l \in S} w_{\mathrm{salt\ intrusion}}(t) \cdot \mathrm{SaltIntrusion}_l$$

Where $w_{\mathrm{water\ loss}}(t)$ and $w_{\mathrm{salt\ intrusion}}(t)$ represent the weight for the respective variables at time t of the lockage. Since this thesis focuses on the environmental conditions of lock scheduling, the water loss and salt intrusion weights are adapted. The weight for the vessel delay remains static, as vessel delays do not depend on the environmental conditions of the lock.

5.2.1 Adaptive Water Loss

The adaptive water loss weights $w_{\mathrm{water\ loss}}$ are computed as follows:

$$w_{\mathrm{water\ loss}}(t) = \max\big(-1.0,\ \min\big(2.0,\ \delta_{\mathrm{canal\ sea}}(t) + w_{\mathrm{drought}}(t) + w_{\mathrm{discharge}}(t)\big)\big)$$

With:

- $w_{\text{water loss}}(t)$: the adaptive water loss weight at time t of the lockage. The value is bounded to the range of $[-1.0,\ 2]$, to keep the weights within reasonable bounds. A negative value indicates that the Terneuzen–Gent canal water level is below the sea water level, meaning a lockage is beneficial, and thus the weight is negative.

- $\delta_{\text{canal sea}}(t)$: the difference between the water level in the Terneuzen–Gent canal and the sea. This value is normalised between $[-1.0,\ 2]$ with the differences between the water levels of the year.

- $w_{\text{drought}}(t)$: the drought value at time t of the lockage.

- $w_{\text{discharge}}(t)$: the discharge value at time t of the lockage.

The drought and discharge values at time t of a lockage are computed with a sliding window of three days. When the average water level in the Terneuzen–Gent canal or the average discharge of the time window is below a threshold, we say that the lockage is at a moment of drought, or low-discharge, respectively. The drought threshold is 2.1m, as this is the ideal water level of the canal. Since there is no ideal value for discharge, its threshold is the average discharge over the year. We increase the weight of water loss when lockages happen during drought or low-discharge, as these circumstances make reducing the loss of water more crucial.

When the canal water level is over the drought threshold, or the discharge is higher than the discharge threshold, then their respective values are negative. When the water level is consistently higher than the ideal 2.1m and when discharge is consistently higher than average, this means that there is too much water in the canal. Lockages at such times are beneficial; therefore, we decrease the weight of the water loss. Similarly, when the discharge is high, this also signals an abundance of water, making lockages at these moments beneficial.

5.2.2 Adaptive salt intrusion

The adaptive salt intrusion weights $w_{\text{salt intrusion}}(t)$ are computed as follows:

$$w_{\text{salt intrusion}}(t) = \max\big(0,\ \min\big(2.0,\ \delta_{\text{sea canal}}(t) + \delta_{\text{salinity}}(t) - w_{\text{discharge}}(t)\big)\big)$$

With:

- $w_{\text{salt intrusion}}(t)$: the adaptive salt intrusion weight at time t of the lockage. The value is bounded to the range of $[0,\ 2]$, to keep the weights within reasonable bounds. This value cannot be negative, as lockages are always disadvantageous for salt intrusion.

- $\delta_{\text{sea canal}}(t)$: the difference between the sea level and the water level in the Terneuzen–Gent canal.

- $\delta_{\text{salinity}}(t)$: the difference between the salinity of the sea and the Terneuzen–Gent canal.

- $w_{\text{discharge}}(t)$: the normalised discharge value at time t of the lockage over the yearly data. Discharge is subtracted, as a higher discharge means salt water is 'flushed away' [29], so this should lead to a lower weight.

All components are normalised between [0, 1] with minimal and maximal values of the yearly data.

5.3 Experiment 3: Adaptive Temperature

The second key feature of SA is the temperature (T). We can adaptively control T by adjusting it to get an acceptance rate close to an ideal acceptance rate. We adjust T following the work by Swartz [16], which was refined by Boyan [67] to create the now-called Modified Lam Annealing schedule. We implement it following the optimised version from Cicirello [68].

T is increased or decreased when the actual acceptance rate is below or above the ideal acceptance rate, respectively. The ideal acceptance rate starts at 1, while the actual acceptance rate starts at 0.5, causing T to increase. This allows exploration and acceptance of new, possibly worse, solutions. The ideal acceptance rate declines exponentially for the first 35% of the allowed iterations, concurrently decreasing T. Then, for 50% of the iterations of the search, the ideal acceptance rate is fixed at 0.4. During the last 15% of the iterations, it exponentially declines towards 0, continuously decreasing T to stop accepting worse solutions than the current solution, thus exploiting good solutions towards a local optimum.

The Modified Lam Annealing schedule uses a fixed number of iterations; we name this stopping criterion *Fixed*. Macomi's SA terminates after finding no improvements for 1000 iterations. Therefore, we construct a second hybrid approach for SA with adaptive T, following the Modified Lam Annealing with Macomi's stopping criterion, which we name *No Improvement*. This approach still uses a hyper-parameter for the number of iterations, but after the specified iterations, the ideal acceptance rate is fixed at 0, continuously decreasing T. The algorithm then continues until no improvements are found for 1000 iterations.

5.4 Experiment 4: Adaptive Neighbourhood Control

The final crucial part of SA that we make adaptive is the neighbourhood control. Neighbourhood control happens by the selection of the new mutation based on the current solution. We can make this mutation selection adaptive by steering what mutation operator is selected.

Macomi's SA has five mutation selection operators, with their own probability of being selected:

- Move Vessel: Moves the vessel to a random existing lockage. Probability: $\frac{1}{3}$.

- Move Vessel Greedy: Moves the vessel to an existing lockage near the vessel's arrival time. Probability: $\frac{1}{3}$.

- Swap Vessels: Swaps the vessel with another vessel in a random lockage. Probability: $\frac{1}{6}$.
- To New: Creates a new lockage for the vessel at a new time in the same lock. Probability: $\frac{1}{12}$.
- Change Visit: Move the vessel to a different lock at a similar time as the original lockage. Probability: $\frac{1}{12}$.

An operator is selected randomly following their probabilities. The probabilities are assigned empirically by Macomi. To adapt the Neighbourhood control, we change these probabilities. We keep the probabilities greater than 0 at all times, allowing SA to theoretically explore all solutions that the operators combined can produce.

5.4.1 Memory

We evaluate two approaches of adaptive neighbourhood control. The first follows from work by Yu et al. [20]. This approach changes the weights of the operators based on the fitness of the solutions found by that mutation. The algorithm keeps a memory of the fitness resulting from each operator. The operators that find solutions with better fitness get an increased probability of being selected for future mutations.

The fitness history of an operator is stored as a weighted average of the last produced fitnesses of that operator. This weighted average is computed following the Exponential Moving Average (EMA) [73] computation. EMA assigns the highest weight to the most recent observation, following this formula:

$$EMA_t = \alpha \cdot P_t + (1 - \alpha) \cdot EMA_{t-1}$$

With EMA_t as the current weighted average, P_t as the last produced fitness, and α as the recency bias. For the experiments, we evaluate the adaptive memory with varying values for α. Lower values encapsulate a longer history of values, reducing the variance, and causing the EMA_t value to be minimally dependent on the most recent observation. A larger α indicates a strong bias towards the most recent observation, causing more variance in the EMA_i.

5.4.2 Environment

The second approach to adaptive neighbourhood control is based on the environmental conditions of the lock. Based on the heuristics of what makes a lockage fitter, we adjust the mutation selection operator weights. Four environmental signals are computed to measure how favourable the current lock conditions are.

Salt intrusion pressure is high when the water level of the sea exceeds the Terneuzen–Gent canal level, since this causes more saline water to enter the canal during a lockage. **Water loss pressure** is high when the canal level exceeds the tide level, as this indicates more fresh water lost per lockage. **Salinity pressure** is high when the salinity difference between the sea and the lake is large, because this indicates a larger increase in salinity of the canal. **Discharge pressure** is high when the river discharge is low, reflecting drought conditions where fresh water is scarce, meaning the loss is more costly, and this reflects the lack of flushing of the high salinity seawater. These four signals are combined into a single scalar, which represents how environmentally favourable the current moment is for performing a lockage.

The environmental scalar drives the selection probability of selecting the mutation operators. *Move Vessel* is favoured under high environmental pressure and when the current lockage only contains one vessel. The rationale is that when lockage occurs under negative environmental conditions, vessels should be moved away from the current lockage, since we prefer not to send vessels through the locks under negative conditions. Moving a vessel from a lockage containing only one vessel leaves the lockage as a candidate for removal, reducing the number of lockages. The lockage is then also no longer bound by vessel timing requirements, giving more freedom for the time of the lockage.

Swap Vessels is favoured under low environmental pressure and when the lockage contains two or more vessels. The rationale is that under good conditions, a full lockage should be kept full, so only the category of vessels should be swapped, and the number of vessels in the lockage should remain equal.

Move To New Lockage is increased when the lockage only contains one vessel. Creating a new lockage will most likely increase water loss and will always increase salt intrusion. However, creating a new lockage is more acceptable when the current one is underutilised and thus a candidate for removal. This could lead to the possible removal of the lockage and give more freedom to the time of the lockage.

Change Visit is favoured under low environmental pressure, as having vessels in a lockage at this time is beneficial, so only reassigning them to a different lock keeps the vessel at the superior time.

The intensity of the environment-based heuristics is guided by a strength parameter. A strength of zero results in Macomi's empirically selected operator probabilities; a higher intensity weight amplifies the differences between the probabilities of the adaptive operators.

5.5 Experiment 5: Fully Adaptive Simulated Annealing

As the final experiment, we combine the adaptive weights, temperature and neighbourhood control to create a fully adaptive SA algorithm.

The fully adaptive SA consists of the best-performing adaptive approaches from the previous experiments. For the adaptive weights, we select the weight which outperforms the baseline on water loss and salt intrusion, without a substantial increase in vessel delay. Based on the pairwise experiment between the adaptive temperature with a fixed number of iterations and the adaptive temperature with Macomi's stopping criteria, we select the better approach. Then, based on the comparison with the baseline, we can select the preferred number of iterations for the cooling schedule. Finally, we use the adaptive neighbourhood control that has the clearest superiority over the baseline and use the parameter which scores best; the best α in the case of the memory approach or the superior intensity weight for the environment approach.

Chapter 6

Experiment Design

This chapter describes how we test the performance of each of the adaptive SA approaches. The experiment design is divided into five experiments, as described in Chapter 5. The baseline experiment compares Macomi's original SA implementation, as the baseline, to the extended version with environmental parameters. The preferred configuration from the first experiment functions as the baseline for the further evaluation of the adaptive approaches. First, we discuss what data we use to run the experiments on. Then, this chapter describes the parameters on which each experiment is evaluated and how we test for significant results.

6.1 Data

Macomi has an internal simulation pipeline which constructs an arrival schedule for vessels at the North Sea Locks. These arrival schedules are the input for the SA algorithm to construct a lock plan. All arrival schedules are simulated on 2018 data, as this is the data available from Macomi.

The complete dataset used to evaluate the algorithms consists of 20 such arrival schedules. The arrival schedules are distributed over the four seasons, giving five schedules during summer, autumn, winter, and spring. The schedules are divided over the seasons, as some of the experiments depend on the environmental conditions of the locks. These conditions differ between seasons. During summer and autumn, droughts are more common and discharge from Gent is lower. During winter and spring, droughts are uncommon, and discharge is higher. The distinction between seasons for the input data gives more diverse data, and allows for the comparison of algorithms during a specific season.

The SA algorithm treats previously scheduled vessels and lockages as constraints. An arrival schedule with hardly any vessels scheduled before running SA makes planning of the current vessel with SA superficial. This leads to an uninteresting lockage schedule with few lockages and no conflicts while finding the solutions. With such SA runs, the effects of the algorithm adjustments are not visible. Therefore, we select arrival schedules for the experiments where a significant number of vessels are scheduled before the SA algorithm runs.

When comparing multiple algorithms, all algorithms run on the same input data and with the same random seed. This reduces the variance of the experiments, increasing the power of the significance tests. We describe, for each experiment, how we test for significant differences in the results.

6.2 Experiment 1: Water Loss and Salt Intrusion Parameters

To measure the effects of adding water loss and salt intrusion to the cost function, we compare eight configurations with different water loss and salt intrusion weights against the original SA implementation by Macomi. We explore the following weights: 0.05, 0.1, 0.2, 0.5, 1, 2, 5, and 10. These values are selected to cover a large range of weights for the water loss and salinity. Since the experimental runs show that there is a relatively large decrease in water loss and salt intrusion for weights smaller than one, we explore more weight values in the lower range.

The fitness value of SA with water loss and salt intrusion is inflated because of the added components to the evaluation function. Therefore, we compare the performance of the configurations and the baseline on the amount of water lost in m^3, the amount of salt entering the canal in kg and the total vessel delay in minutes.

6.2.1 Baseline Statistical Test

To test if the new configuration results are significantly different from the baseline, we perform a statistical test. For all following experiments where we compare several configurations of an algorithm approach against a baseline, as is the case in this experiment, we evaluate the significance of the pairwise differences between the configuration and the baseline.

For each configuration result, we compute the difference with the result of the baseline result on the same input data. For each configuration, this gives 20 differences between the configuration and the baseline. We test whether these differences are normally distributed with the Shapiro-Wilk test.

If the differences are normally distributed, we can use the paired t-test [74] to see if the difference between the configuration and the baseline is significant. The paired t-test is a strong statistical test, but it requires normally distributed data. Since we perform multiple comparisons, the probability of getting a significant result by chance increases. Therefore, we perform a correction of the p-value using the Benjamini-Hochberg (BH) procedure [75]. The BH procedure reduces the number of false positives when testing multiple hypotheses.

If the differences are not normally distributed, we test significance using the Wilcoxon signed rank-test [76]. This procedure does not require normally distributed data, but it is weaker than the paired t-test. Again, after we compute the p-values, we correct them with the BH procedure.

6.2.2 Hypothesis

With the added water loss and salt intrusion parameters, we expect to see a decrease in the volume of water lost to the sea, and we expect a decrease in the amount of salt entering the

Terneuzen–Gent canal. This result will be the effect of fewer lockages and lockages timed at different moments. We predict that a side effect of this change in the vessel schedule is that the vessel delay increases. With fewer lockages and lockages at moments beneficial for water loss and salinity, vessels could potentially be planned in less optimal lockages, increasing their delay. This hypothesis is grounded in the work by Bakker and Van Koningsveld [10].

Based on the experimental runs, we expect the configuration with weights of 1 to have an equal contribution of water loss and salt intrusion to the total fitness, as the vessel delay contributes to the total fitness. The lowest weight, 0.05, will likely have minimal effects on the vessel delay, water loss and salt intrusion, as such a small weight keeps the priority on minimising the vessel delay, similar to Macomi's original implementation. By contrast, the weight of 10 will show the most extreme results, showing a strong decrease in water loss and salt intrusion at the cost of high vessel delays. We expect the baseline for the future experiments to have weights of 0.5, showing a clear decrease in water loss and salinity, but keeping a focus on minimising vessel delays.

6.3 Experiment 2: Adaptive Weights

We evaluate eight configurations of the adaptive weights, each with a different initial weight. These initial weights are equal to the weights used to find our baseline with static weights: 0.05, 0.1, 0.2, 0.5, 1, 2, 5, and 10. We evaluate the performance of SA with adaptive weights based on three sub-experiments.

In the first experiment, we compare the eight configurations against the baseline resulting from experiment 1. This shows whether the adaptive weights can outperform the baseline. Then, we compare the eight configurations against the baseline for each of the four seasons. This experiment indicates whether the effect of the adaptive weights is larger under particular seasonal circumstances. Lastly, we do a pairwise comparison of each static weight configuration with the matching adaptive weight configuration. This experiment allows us to conclude whether SA with adaptive weights is generally better than SA with static weights.

We compare the results on the amount of water lost in m^3, the amount of salt entering the canal in kg, and the total vessel delay in minutes.

6.3.1 Paired Statistical Test

For the comparison of the first and second subexperiments: adaptive weight SA against the baseline, we check for significance following the same procedure as described for Experiment 1 (Section 6.2.1).

For the pairwise comparison of the static weights SA against the adaptive weights SA, we perform a pooled pairwise significance test. We match each static weight configuration with the adaptive weight configuration with the same weight. The differences between the elements in each pair are pooled together, giving 8 (configs) $\times$ 20 (runs per config) $=$ 160 differences in total. The Shapiro-Wilk test tells whether the data is normally distributed, leading to either a one-sample paired t-test or the Wilcoxon signed-rank test.

6.3.2 Hypothesis

In the experiment against the baseline, we expect a decrease in the water loss and the salt intrusion, as the adaptive weights can inflate the importance of water loss and salt intrusion. Especially during crucial periods of drought or low discharge, we expect a decrease in water loss and salt intrusion. Since these periods are mostly during summer and autumn, the adaptive weights SA approach will be more effective during these seasons.

We anticipate the pairwise experiment to show that adaptive weights improve the performance of SA with respect to the water loss and salt intrusion. Similarly to Experiment 1, where the weights of water loss and salt intrusion increase, the vessel delay might increase due to the inflated cost of the water loss and salt intrusion.

6.4 Experiment 3: Adaptive Temperature

We evaluate two approaches for the adaptive temperature. The first is the Modified Lam Annealing with a fixed number of iterations (*Fixed*) [17]. The second uses the same cooling schedule, but with Macomi's stopping criteria (*No Improvement*). Both approaches have five configurations with a varying number of iterations N.

For the *Fixed* approach, the algorithm is terminated after N iterations. For the *No Improvement* algorithm, N represents the number of iterations for the ideal acceptance rate to approach 0. After N iterations, the ideal acceptance rate remains zero, meaning T decreases towards zero until no new solutions are accepted for 1000 iterations.

Macomi's original SA algorithm generally runs for circa 4000 to 5000 iterations, so we select the following values of N: 500, 1000, 2500, 5000, and 7500, as a range around these values. We expect the adaptive temperature approach to improve the baseline, so we evaluate more configurations with a lower number of iterations. 7500 iterations is approximately the maximum number of iterations used by Macomi's original cooling schedule to terminate.

Both approaches are compared against the baseline, aiming to show whether the adaptive temperature SA improves the baseline. Then, we do a pairwise comparison of the two approaches for the adaptive temperature to show which is superior. We compare the results on the fitness of the best solution and the number of iterations, as these are the variables influenced by the temperature.

6.4.1 Statistical Tests

To test the significance of the results of the comparison between each adaptive temperature approach and the baseline, we follow the same procedure as described in Experiment 1 (Section 6.2.1).

For the pairwise comparison between the *Fixed* and *No Improvement* adaptive temperature SA, we use the pooled pairwise comparison as described in Experiment 2 (Section 6.3.1). We match the configurations of both approaches in terms of the number of iterations N for the cooling schedule.

6.4.2 Hypothesis

For the *Fixed* algorithm, we expect the configurations with a low number of iterations (500 and 1000) to terminate with higher fitness solutions. We expect the configurations with slightly fewer or a similar number of iterations (2500 and 5000) to reach comparable or slightly better fitness in a similar number of iterations. An added benefit of the *Fixed* approach is that it gives the practitioner more control over the runtime of SA, as running SA until no improvements are found for 1000 iterations leads to a large variance in runtime. We expect the configuration with 7500 iterations to find solutions of higher fitness, but in significantly more iterations.

We expect the *No improvement* approach to run significantly longer than the *Fixed* approach. We do expect this algorithm to find solutions with better fitness than the baseline, as the adaptive temperature allows for more exploration early in search, while allowing the same exploitation at the end of the search process. Increased exploration leads to finding a broader range of solutions, and therefore, possibly better solutions.

In the pairwise comparison, we will see that the runtime of the adaptive temperature with the *Fixed* stopping criterion is significantly lower than the runtime of the *No Improvement* criterion. As a tradeoff for the lower runtime, we anticipate worse fitness values for SA with the *Fixed* criterion.

6.5 Experiment 4: Adaptive Neighbourhood Control

We evaluate two approaches for the adaptive neighbourhood, as described in Sections 5.4.1 and 5.4.2. We test five configurations of the memory approach and five configurations of the environment approach.

We test the memory approach with the values: 0.05, 0.1, 0.2, 0.3, and 0.5 for α to cover a large range of recency biases. For EMA, the values are generally between 0.1 and 0.3 in the literature, as these weights give a balanced weight for the most recent observation and the history. We select a broader range to highlight the effects more clearly.

We evaluate domain-driven adaptive neighbourhood control with intensity weights of: 0.2, 0.5, 1, 2, and 5, where the smaller values aim to show realistic adjustments to the operator probabilities, and the larger weights aim to clearly show the effects of the adaptive neighbourhood control. Since the environmental neighbourhood is dependent on the season, we also compare season-wise.

We compare the results on the fitness of the best found solutions and the number of iterations, as these are the parameters affected by the adaptive neighbourhood control. For the environment-based approach, we also examine the amount of water lost in m^3, the amount of salt entering the canal in kg and the total vessel delay in minutes, as the environment heuristics are aimed at improving the water loss and salt intrusion.

6.5.1 Statistical Test

For the comparison of both adaptive neighbourhood control approaches against the baseline, we check significance following the procedure as described in Experiment 1 (Section 6.2.1).

6.5.2 Hypothesis

For the memory approach, we expect to see the same number of iterations as the baseline since the number of iterations should not be affected by different operators. We do anticipate better fitness values, because the mutation operators that have proven to work well previously by finding low fitness solutions are used more often and thus low fitness solutions can be found more often.

For the environment approach, we expect to reduce the salt intrusion and water loss of the found solution, because mutation operators are selected following heuristics beneficial for water loss and salt intrusion. The mutation operators guided by heuristics are not directly disadvantageous for the timing of vessels, so we do not expect a large increase in vessel delays. This means that the overall fitness of the solutions should decrease.

6.6 Experiment 5: Fully Adaptive Simulated Annealing

We do a pairwise comparison between the fully adaptive SA algorithm and the baseline. This will show if the sum of well-performing components also outperforms the baseline.

We compare the results on the number of iterations, the amount of water lost in m^3, the amount of salt entering the canal in kg and the total vessel delay in minutes. We do not evaluate on the fitness from the evaluation function, as the adaptive weights mutate the weights of the components in the evaluation function, causing the fitness value to no longer be comparable with the static fitness function from the baseline.

6.6.1 Statistical Test

We test if the difference between the baseline and the fully adaptive algorithm is significant, as described in Section 6.3.1, by computing the pooled differences between the results of the fully adaptive algorithm and the baseline results.

6.6.2 Hypothesis

We expect adaptive SA to find better solutions, especially for water loss and salinity, in fewer iterations. The separated parts of the complete algorithm have shown improvements against the baseline. We expect the sum of the components to also give the sum of the results, since the adaptive components act on secluded parts of the SA algorithm. This would lead to their respective improvements accumulating to a greater improvement of the baseline.

Chapter 7

Results

This chapter describes the results of the experiments described in Chapter 5. Experiment 1 will determine the baseline for the further experiments. Experiments 2, 3, and 4 compare a singled-out adaptive component against that baseline. The final experiment explores whether the components of adaptive SA complement each other and whether a completely adaptive SA algorithm improves the baseline. We discuss the meaning of each experiment's result and conclude by discussing adaptive SA in general.

7.1 Experiment 1: Water Loss and Salt Intrusion Parameters

Experiment 1 aims to find a baseline for future experiments. Macomi's lock planning algorithm is extended with water loss and salt intrusion parameters. We evaluate eight configurations with varying weights for the new parameters. Figure 7.1 shows the comparison of the configurations against the baseline.

All differences in vessel delay, water loss, and salt intrusion between the configurations and the baseline are significant. The smallest weight for the water loss and salt intrusion variables shows an immediate strong decline over the baseline in litres of water loss and kilograms of salt entering the canal. The weight of 0.05, on average, causes 19.31% less water loss and a 22.83% decrease in salt intrusion, leading to a 51.76% vessel delay increase. This shows that by putting minimal focus on water loss and salt intrusion, there is already much to gain. The amount of water loss and salt intrusion decreases with increasing weights. However, as the weights of the environmental variables increase, the amount of vessel delay increases more rapidly than the environmental impact decreases. The configuration with a weight of 0.5, which we hypothesised as the ideal configuration, introduces a reduction of 24.44% in water loss and 31.44% in salt intrusion, causing a 309.45% increase in vessel delay.

These results show that our hypothesis was correct; adding water loss and salt intrusion to the objective function leads to decreased water loss and salt intrusion. As a consequence of this, vessel delays increase significantly. This is also supported by the results of the work by Bakker and Van Koningsveld [10].

For future experiments, we select the configuration with weights of 0.2 as the baseline,

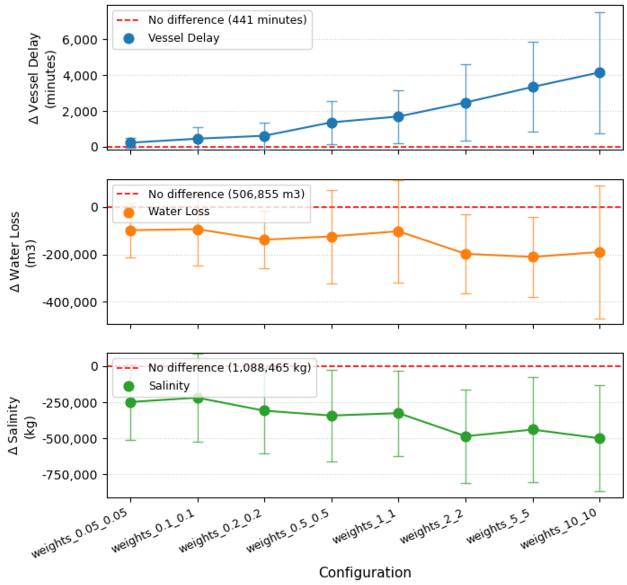

Figure 7.1: SA with water loss and salt intrusion compared to Macomi's original model as baseline. The error bars represent the variance over the pairwise differences.

reducing water loss by 27.20% and salt intrusion by 28.37%, at the cost of an increase in vessel delays of 139.54%. This configuration shows a relatively large decrease in water loss and salt intrusion, while keeping the vessel delay somewhat within the range of the baseline's vessel delays. We select the configuration with weight 0.2 over the smaller weights, as it puts more emphasis on water loss and salt intrusion, allowing us to see their effects more clearly. The selected weight is lower than our expected weight, since the effect on water loss, salt intrusion, and vessel delay of the environmental variables was larger than expected.

7.2 Experiment 2: Adaptive Weights

Experiment 2 compares SA with adaptive weights against the baseline with static weights. We compare eight configurations with varying starting weights against the baseline with weight 0.2 to see whether adaptive weights can improve the baseline. Then, we compare each SA with adaptive weights against SA with static weights by pairing the configurations with the same (starting) weight.

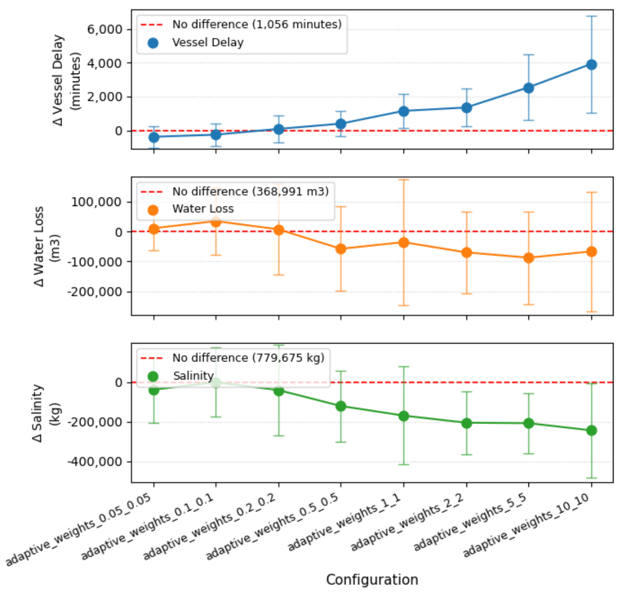

Figure 7.2: SA with adaptive weights compared to the baseline with static weights fixed at 0.2.

7.2.1 Baseline Comparison

Figure 7.2 shows the comparison of the adaptive weights configurations against the baseline. In the comparison with the baseline, we observe that the configuration with the same start weight as the baseline reaches very similar results, with no significant differences. The configurations with smaller weights of 0.05 and 0.1 also show no significant differences for the water loss or the salt intrusion. However, these configurations do significantly decrease the vessel delays by 36.24% and 24.48%, respectively. The results for the water loss metric are not significant for any of the configurations. Salt intrusion and vessel delays are significantly affected by the start weights > 0.2. Vessel delays increase, while salt intrusion decreases for higher weights.

The comparison of the adaptive weights SA against the baseline for each individual season shows hardly any significant differences. Only for the largest start weights 2, 5, and 10, there is a significant improvement in salt intrusion by the adaptive weights approach. This difference can be explained by the large weights producing more extreme results. All other season-wise results are not significantly different from the baseline, due to the small sample size of five runs per season.

We expected to see a decrease in water loss and salt intrusion with the adaptive weights, since the adaptive weights focus on these parameters. However, water loss is not significantly affected by the adaptive weights. The configuration with the same start weight as the static baseline weight also did not show significant differences. This indicates that the

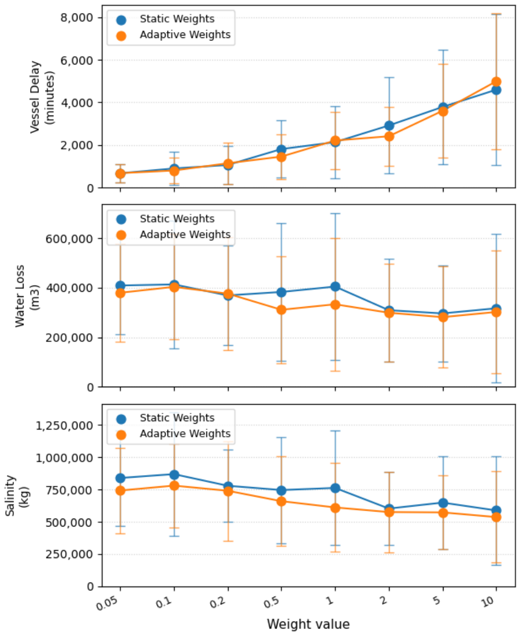

Figure 7.3: Comparison of the static weights SA and the adaptive weights SA, paired on the same (start) weight for water loss and salt intrusion.

adaptive weights do not have a large influence on the found solutions. However, a comparison between the adaptive weights approach and a single baseline does not clearly show the advantages and disadvantages of the algorithmic difference. Therefore, to explore these differences further, we do a pairwise experiment between the two algorithms.

7.2.2 Pairwise Comparison

The results of the pairwise experiment are shown in Figure 7.3. The pooled statistical test shows no significant differences for the vessel delays, as visible in the figure. Both water loss and salt intrusion decrease significantly by the adaptive weights approach. Between the different configurations, the adaptive approach differs in water loss, fluctuating between $+1.96\%$ and -18.77%. The salt intrusion shows an even more consistent performance increase where the adaptive approach improves the salinity by -4.54% to -19.85% between configurations.

The results of water loss and salt intrusion of the pairwise comparison are consistent with the hypothesis and show the difference between the static and adaptive weights more clearly. Since the adaptive weights dynamically change the weights of these parameters under crucial lock conditions, they improve with adaptive weights. We anticipated the vessel delays to increase because inflated water loss and salt intrusion weights could signify fewer lockages or lockages at inefficient moments.

However, the adaptive weights do not merely inflate the weights of water loss and salt intrusion. Under beneficial lock conditions, their weights are decreased, and the emphasis shifts towards vessel delay costs. The reallocation of priority from water loss and salt intrusion towards vessel delays causes the effect on vessel delay to cancel out over all runs. Vessel delay is alternately prioritised and deprioritised depending on environmental conditions, resulting in similar vessel delays for the adaptive weights SA to the static weights baseline.

Water loss and salt intrusion benefit asymmetrically from shifting the priority based on the environmental conditions. At moments where a lockage has an increased risk of high salt intrusion or water loss, the priority of these parameters is increased. At moments less crucial to these parameters, the priority is shifted back to the vessel delays. This results in an overall reduction of water loss and salt intrusion.

These results highlight the effectiveness of adaptive weights. The domain-specific lockage heuristics allow parameters to be prioritised at their respective crucial moments. Deciding on the heuristics requires domain knowledge and is not generalisable, but adaptive weights can be generalised and show promising results.

7.3 Experiment 3: Adaptive Temperature

The second component of SA that we make adaptive is the temperature (T). We evaluate two approaches and compare them to the baseline. Both approaches adapt T with the Modified Lam annealing schedule [67]. The first approach uses the same stopping criteria as described in this work, with a fixed number of iterations; we name this stopping criterion *Fixed*. The second approach adopts Macomi's stopping criterion, which terminates after finding no improvement in the found solution for one thousand iterations; we name this criterion *No Improvement*.

7.3.1 Fixed Number of Iterations Stopping Criterion

The first sub-experiment evaluates the *Fixed* adaptive temperature SA approach. The results are shown in Figure 7.4.

The baseline uses, on average, approximately 5000 iterations. The *Fixed* adaptive temperature with 500, 1000, and 2500 iterations, therefore, uses significantly fewer iterations than the baseline. All configurations, except for 7500 iterations, score significantly worse than the baseline in fitness. This means that even the configuration of 5000 iterations, a similar amount to the baseline, does not improve the fitness of the found solutions. The configuration with the largest runtime has significantly more iterations and reaches fitness that is not significantly different from the baseline.

The trend in the results of the configurations follows expectations; longer runtime translates to reaching better solutions. However, contrary to the hypothesis, adapting the cooling schedule does not improve the performance of SA for the lock scheduling problem. The adaptive temperature allows more exploration early in the search for a broader range of solutions. However, since the focus is more on exploration, later in search, the found solutions are not exploited to the fullest.

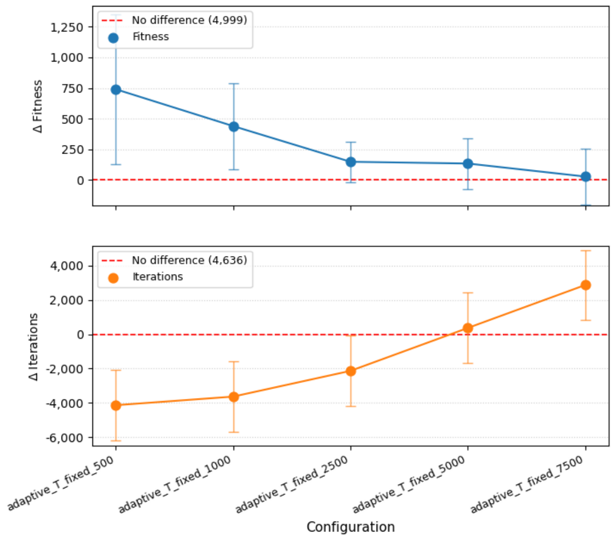

Figure 7.4: *Fixed* adaptive temperature SA, compared to the baseline on fitness and number of iterations. The variance in runtime follows only from the baseline variance.

SA with the Modified Lam Annealing schedule cannot directly improve the baseline of the lockage scheduling problem. The adaptive temperature gives the practitioner more control over the runtime of the algorithm, so if this is a high priority, the adaptive temperature can be useful. However, following the increased exploration early in the search, exploitation of found solutions is not optimal. This leads to finding worse solutions than the original geometric cooling schedule with a stopping criterion of finding no improvements for 1000 iterations.

7.3.2 No Improvement Stopping Criterion

The second sub-experiment evaluates the *No Improvement* adaptive temperature. The number of iterations for each configuration represents the number of iterations for the ideal acceptance rate to approach zero. After the assigned iterations, the ideal acceptance rate is fixed to zero. Then, SA continues until it finds no improvements for one thousand iterations, matching the baseline's stopping criterion. The results are shown in Figure 7.5.

We observe that the configurations all behave similarly. Each configuration runs for significantly more iterations than the baseline. Despite the extra iterations and the increased focus on exploration early in the search, none of the configurations reaches significantly better solutions than the baseline. The results signify that the *No Improvement* stopping criterion reduces control over the runtime. All configurations require around 7000 iterations or more, even though the ideal acceptance rate is fixed at zero for the largest portion of the runtime.

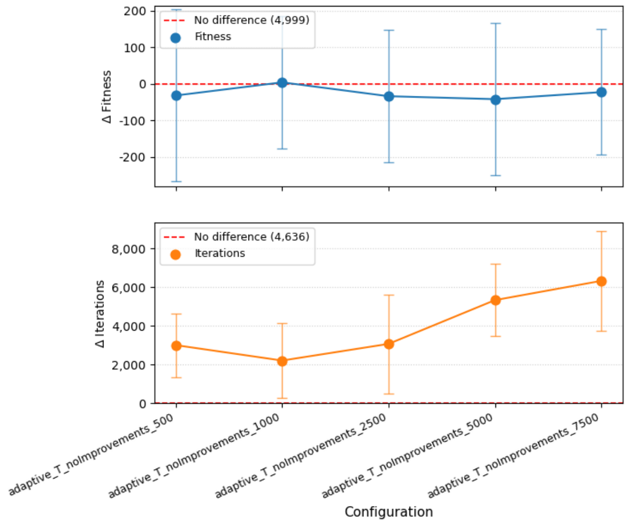

Figure 7.5: *No Improvement* adaptive temperature SA, compared to the baseline on fitness and number of iterations.

The results show that even with a stopping criterion that allows exploitation of the found solutions, the adaptive temperature does not clearly outperform the baseline. Since the cooling schedule from the Modified Lam Annealing schedule cools more slowly than the geometric schedule from the baseline, SA accepts solutions more often, increasing the runtime for SA with this stopping criterion. Despite this increased runtime, the fitness does not improve significantly, showing the Modified Lam algorithm is not effective for Macomi's lock scheduling problem.

7.3.3 Pairwise Comparison

We compare the *Fixed* and *No Improvement* stopping criteria for adaptive temperature SA. We perform a pairwise comparison matched on the number of iterations for the cooling schedule. The results are shown in Figure 7.6.

The results show consistently better fitness for the *No Improvement* approach. This increase in fitness is at the cost of significantly more iterations. As the number of iterations for the *Fixed* approach nears that of the *No Improvement* stopping criterion, we observe a decrease in the difference in fitness. The results do not show a definitively better approach, as the better performance of the *No Improvement* condition can be attributed to the longer runtime.

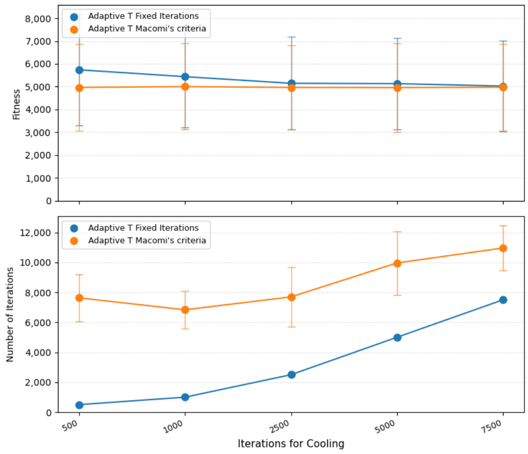

Figure 7.6: Pairwise comparison in number of iterations and fitness of *Fixed* and *No Improvement* adaptive temperature SA.

7.4 Experiment 4: Adaptive Neighbourhood Control

The final adaptive SA component is the neighbourhood control. Neighbourhood control is handled by the mutation operators, which are selected with a certain probability. In the baseline, these probabilities are fixed; the adaptive approaches change them. The first approach bases the probabilities on the fitness of previous solutions found by the operator. Finding solutions with better fitness results in increased probability of being selected. The second approach is guided by environmental conditions. Heuristics prioritise operators beneficial under the environmental conditions at the time of the lockage.

7.4.1 Memory

First, we compare the memory-based approach against the baseline. We evaluate five configurations, each with varying values for α. Higher values for α indicate that the memory prioritises the most recent solution, so the operator weights are affected most by the last seen fitness. The results are depicted in Figure 7.7.

The results show no significant differences between the baseline and any of the adaptive neighbourhood control configurations with memory. A general trend in the results seems to be that the adaptive memory approach uses fewer iterations (decrease of 7.11% to 16.45%), leading to solutions that are slightly worse, ranging from 0.92% to 4.18% higher fitness. This could be the consequence of selecting less favourable mutation operators despite them reaching good fitness previously. SA terminates after no improvement is found for 1000 iterations. This means that a mutation operator should find an improvement over the current solution to extend the search. By selecting worse mutation operators, SA terminates sooner and therefore reaches fewer iterations and worse fitness.

Figure 7.8 shows the weights of the mutation operators over the iterations. Because runs terminate after different numbers of iterations, averages are computed per iteration

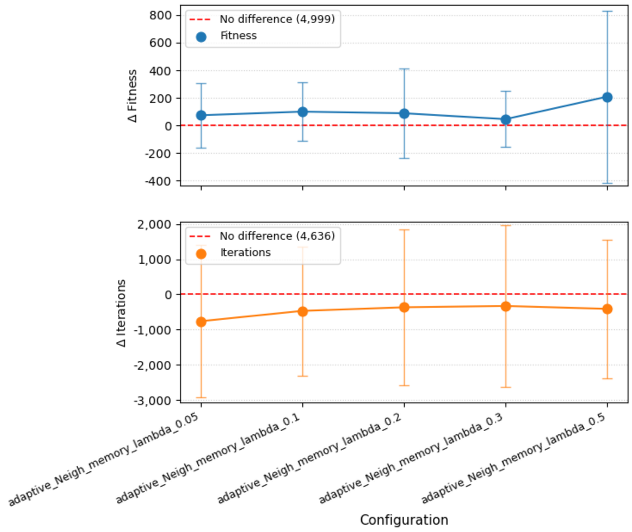

Figure 7.7: Adaptive neighbourhood control with memory of best-performing mutation operators compared to the baseline on fitness and number of iterations.

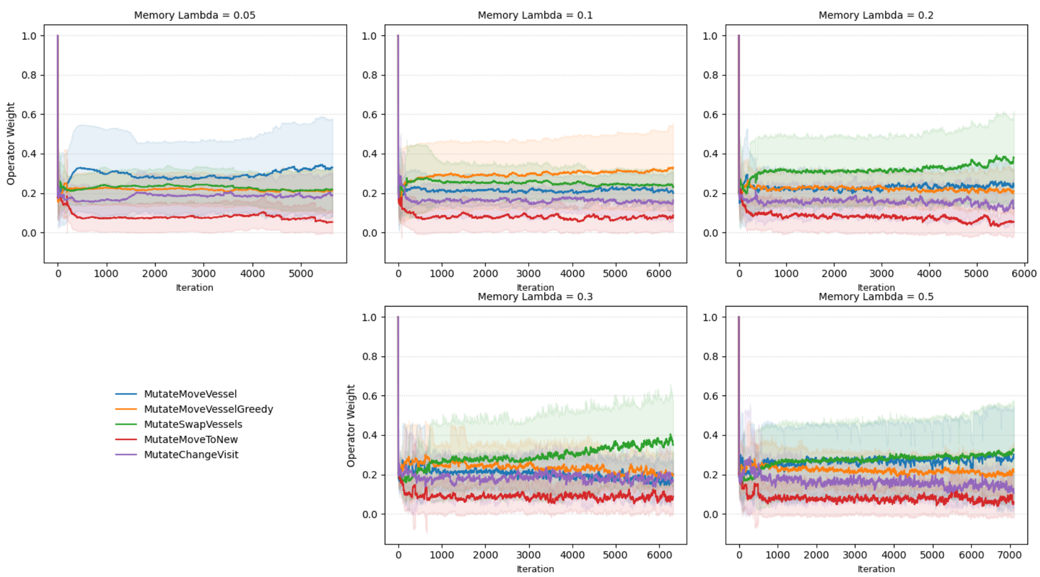

Figure 7.8: The averaged weights of the mutation selection operators over the iterations for the memory approach.

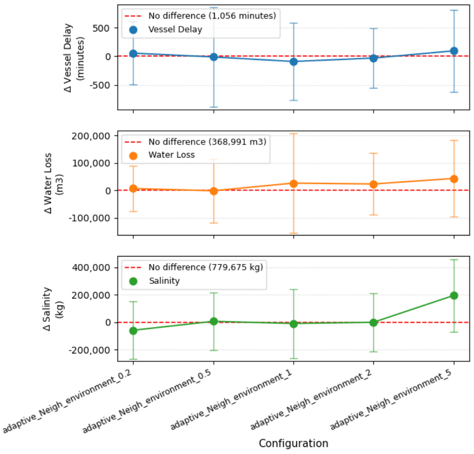

Figure 7.9: Comparison of adaptive neighbourhood control guided by environmental heuristics against the baseline, on vessel delay, water loss, and salt intrusion.

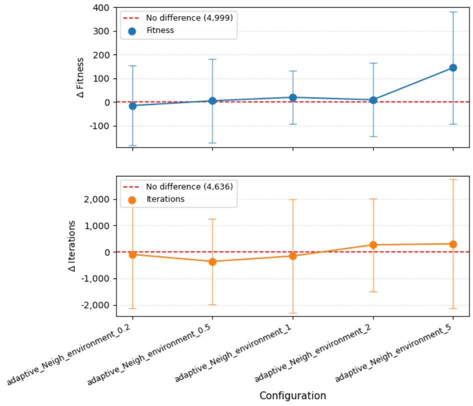

Figure 7.10: Comparison of adaptive neighbourhood control guided by environmental heuristics against the baseline, on fitness and the number of iterations.

using all available observations. Iterations for which fewer than half of the runs remained were excluded from the figure to avoid unreliable estimates based on a small number of observations.

We can see that the operators all start with similar weights. After a small number of iterations, the operator weights disperse. Especially, the *Move to New* operator produces consistently worse solutions. The *Change Visit* operator also consistently underperforms compared to the remaining three operators. The operator weights vary between 0.047 and 0.329, showing that the mutation operator selection is influenced strongly by the memory. As expected, larger values for α increase the variance of the weights, as it depends more strongly on the last observation, but the values show similar patterns to lower α configurations.

We can conclude that the memory approach, as described in Yu et al. [20], does not work well for Macomi's lockage scheduling problem. The memory does find operators that generally construct better solutions, but selecting these operators more frequently does not lead to better solutions over the full SA run.

7.4.2 Environmental Conditions

The second experiment adapts the mutation operator probabilities based on the lockage's environmental conditions. Heuristics guide what operator is preferred at a certain lockage time. We evaluate several intensities of the heuristics, where a higher intensity weight signifies a more significant increase in the probability of the preferred operator. The results of comparing these configurations against the baseline are shown in Figures 7.9 and 7.10.

The season-wise comparison based on five runs per season did not provide any significant results. Since the variance of the results is too high, the five runs are not sufficient to draw any conclusions.

The results on the aggregated data in Figure 7.9 show mostly non-significant results. Only the salt intrusion is significantly lower for the intensity weight 0.2 and significantly worse for the intensity weight of 5. Figure 7.10 only shows one significant difference with the baseline for the fitness of the configuration with intensity weight 5. This configuration shows a worse performance than the baseline in all parameters, leading to the conclusion that mutation operator selection strongly guided by heuristics does not lead to improved performance.

These results contradict our hypothesis. Since the environmental heuristics are aimed at water loss and salt intrusion, we expected a decrease in these parameters. However, except for the salt intrusion of the smallest intensity weight, they do not change significantly from the baseline. Mutation operator selection guided by heuristics prioritising certain operators based on the environmental conditions of the lockage, as we defined, shows minimal to no improvement for the fitness of the solutions.

7.5 Experiment 5: Fully Adaptive Simulated Annealing

The final experiment evaluates the fully adaptive SA algorithm. Each of the adaptive components is selected from the best-performing configurations of previous experiments.

7.5.1 No Improvements Stopping Criterion

For the adaptive weights, we use a start weight of 0.05. Generally, adaptive weights outperformed the static weights in reducing water loss and salt intrusion. The start weight of 0.05 also reduces the vessel delay significantly. Since the adaptive temperature approaches did not find a configuration which generally improves the baseline, we select the *No Improvements* adaptive temperature approach with Macomi's stopping criterion. We assign 2500 iterations for the cooling schedule to reach zero. This configuration did not perform significantly worse than the baseline, and the adaptive temperature configurations with Macomi's stopping criterion show better results than the configurations with fixed iterations, even when the number of iterations is close. For the adaptive neighbourhood, we select the configuration guided by environment heuristics and intensity weight 0.2. This is the only adaptive neighbourhood configuration which reaches (non-significantly) better fitness than the baseline and has significantly fewer salt intrusion.

This fully adaptive SA takes significantly more iterations than the baseline, because we use Macomi's stopping criteria with the slower adaptive cooling schedule. The combination of adaptive components with the longer runtime allows the algorithm to find solutions with significantly less vessel delay. Water loss and salinity do not show a significant difference from the baseline. This indicates an improvement over the baseline, but this improvement may be attributed to the increased number of iterations of the adaptive approach. Therefore, we evaluate a configuration with a similar number of iterations to the baseline.

7.5.2 Fixed Iterations

Since the increase in performance can be credited to the extra iterations, we evaluate the same configuration but with a fixed number of 5000 iterations, a similar amount to the baseline. The evaluation of the configuration with adaptive temperature with 5000 iterations shows that it does not improve the fitness of the found solutions, nor does it decrease performance. However, it does come with the benefit of increased control over the runtime and a strong reduction of the runtime variance.

The 5000 fixed iterations for the Modified Lam Annealing, combined with the adaptive weights and neighbourhood, show no significant differences with the baseline. This shows that the reduced performance from the adaptive cooling schedule can be compensated for by the adaptive weights and neighbourhood. This gives an algorithm which reaches similar performance to the baseline, without the variance in runtime.

7.5.3 Post-Hoc Environment-Driven Simulated Annealing

Since the only adaptive configurations which showed a concrete improvement over the baseline are the environment-driven adaptive weights and the environment-driven adaptive neighbourhood, we perform a post-hoc experiment to evaluate a final environment-driven configuration. The environment-driven algorithm uses the baseline's geometric cooling schedule, combined with adaptive weights with start weight 0.05, and adaptive neighbourhood guided by heuristics with strength 0.2.

Compared against the **original baseline**, the environmental approach achieves a significant reduction in vessel delays of 42.71% compared to the baseline. The increases in water loss (+1.99%) or salt intrusion (+0.24%) are also significant. These results suggest that the environmental conditions approach provides an improvement over the baseline.

However, the increased performance in vessel delays for the environment-driven algorithm can be attributed to the selected start weight of the adaptive weights. For the environment-driven approach, the weights start at 0.05, while the baseline uses 0.2 as the fixed weight. This means that the baseline generally prioritises the water loss and salt intrusion over the vessel delays.

To isolate the effect of the adaptive mechanisms and the neighbourhood control, we can compare the environmental approach against a **baseline with fixed weights of** 0.05 for the environmental variables. These results show a performance increase for the vessel delays (-9.56%), the water loss (-7.98%) and the salt intrusion (-6.95%). However, these improvements do not reach individual statistical significance, which weakens the claim of improved performance for the environment-driven approach.

The observed performance improvements are consistent with the intended behaviour of the environment-driven approach. The increased performance may be attributed to the shifting in priority between the environmental variables and the vessel delay. Under less demanding lockage conditions, the adaptive weights assign relatively higher weights to vessel delays, leading to fewer delays. Under crucial conditions, the environmental variables benefit from the adaptive weights. Simultaneously, the environmental heuristics may select neighbourhood operators that, besides improved environmental impact, contribute to better performance for the vessel delays.

7.6 Discussion

We perform five experiments leading to a fully adaptive SA algorithm. The aim of the experiments is to see if they outperform the baseline. This section discusses the results in a broader scope, emphasising the most significant findings and their implications.

7.6.1 Baseline

The baseline is determined in Experiment 1, where we extend Macomi's base SA implementation with water loss and salt intrusion. The results show that the smallest tested start weight of 0.05 decreases the loss of water in the intrusion of salt significantly, making the North Sea Locks more environmentally friendly. The water loss and salt intrusion parameters cause the lockages to be scheduled at less preferable times for vessels and to be filled with more vessels. This leads to an increase in vessel delay, confirming the direct tradeoff between environmental impact and scheduling efficiency reported by Bakker and Van Koningsveld [10]. For the further experiments, we select the configuration with weights equal to 0.2, giving a reduction of 27.20% and 28.37% in water loss and salt intrusion at the cost of an increase in vessel delays of 139.54%.

7.6.2 Adaptive Components

The following three experiments isolate the effect of making individual SA components adaptive. We evaluate adaptive versions of the weights in the cost function, the temperature in the cooling schedule, and the neighbourhood control against the baseline from experiment 1.

Adaptive weights produce the clearest and most consistent improvement of the three adaptive components. The pairwise comparison between the adaptive weights and the static weights from the baseline shows a significant reduction in water loss and salt intrusion, without affecting the vessel delays. The key mechanism responsible for the performance increase is the dynamic shifting of the priority of the environmental variables and the vessel delay in the cost function. For the vessel delays, the shifting of priority is arbitrary, leading to equivalent vessel delays for static and adaptive weights. However, the priority increase and decrease happen at crucial moments for the environmental variables. This causes the water loss and salt intrusion to decrease by their adaptive weights. These results are domain-specific and not directly generalisable outside lockage scheduling, but they demonstrate that adaptive cost function weights grounded in domain knowledge can produce meaningful performance gains.

Adaptive temperature following the Modified Lam Annealing schedule, with a fixed number of iterations, does not improve the quality of the solutions for the North Sea Lock scheduling problem. The configurations with fewer or a similar number of iterations to the baseline consistently performed significantly worse than the baseline. Only the configuration with 7500 iterations, roughly 50% more than the baseline, achieved similar performance to the baseline, highlighting the correlation between runtime and solution quality. The issue with the Modified Lam Annealing is that it spends too much time in the exploratory phase of the search. The ideal acceptance rate that the Modified Lam Annealing defines follows from generic test functions, where the longer exploration might be beneficial. However, this does not necessarily reflect the optimal search behaviour for the lockage scheduling problem. The practical value of the adaptive temperature does not lie in solution quality, but in control over the runtime. The variant with a fixed number of iterations eliminates the large runtime variance of Macomi's stopping criterion. This can be valuable when predictable scheduling computation times are required.

Adaptive neighbourhood control shows mixed results depending on the approach. The memory-based adaptive neighbourhood guided, which increases the probability of selecting historically well-performing mutation operators, does not show concrete improvements in performance. A likely explanation for this is the Markovian nature of SA. Selection and evaluation of the next mutation are only based on the current solution and the temperature. Historical performance of operators, therefore, is not a reliable predictor for the performance of future operators. The operator weights do diverge meaningfully over iterations, with *Change Visit* and *Move To New* consistently assigned lower weights, but selecting the stronger operators more frequently does not translate to improved final solutions, suggesting that operator diversity plays an important role in the search dynamics.

The environment-driven approach shows a marginal improvement over the baseline. The smallest intensity weight of 0.2 produces significantly less salt intrusion, with insignificant differences for the remaining variables, making it favourable over the baseline. The higher intensity weights constrain the search too much, leading to worse performance, again indicating that diversity of mutation operators is crucial in the search. The heuristics guiding mutation operator selection are selected based on domain knowledge and under the assumption that certain operators are more beneficial under certain environmental conditions. These prerequisites may not be completely fulfilled, leading to only a minimal improvement of the algorithm.

7.6.3 Fully Adaptive Simulated Annealing

The final experiment combines the three best-performing adaptive configurations into a single fully adaptive SA algorithm. Since the adaptive components each influence a distinct part of SA procedure, the expectation was that their individual benefits could be accumulated to achieve a superior SA configuration. However, the experiment's results show that, in the general case, the sum of parts does not lead to the sum of successes.

The fully adaptive SA uses the Modified Lam Annealing cooling schedule with Macomi's stopping criterion, meaning it uses significantly more iterations than the baseline. These extra iterations lead to a significant reduction in vessel delays, but this performance increase can be attributed to the increased runtime rather than the adaptive components. When the fully adaptive SA is evaluated with a fixed budget of 5000 iterations, a similar number to the baseline, it shows no significant improvements. Even though each adaptive component acts in a secluded part of SA, they influence the same parameters. This causes the effects of the components to counteract, leading to no clear improvement over the baseline.

The most compelling result of the final experiment is the environment-driven SA configuration, which combines adaptive weights and environment-driven neighbourhood control with Macomi's geometric cooling schedule. Compared to a weight-matched baseline with environment variable weight of 0.05, this configuration achieves improvements of 9.56% in vessel delays, 7.98% in water loss, and 6.95% in salt intrusion. These individual differences do not reach statistical significance. However, the consistent improvements across all metrics are promising. The results suggest that the environment-driven components complement each other, where the adaptive weights penalise critical moments, and the adaptive neighbourhood selects operators that steer away from these moments. The priority shifting back to vessel delays at non-critical moments causes vessel delay performance to increase simultaneously.

The experiments lead to three conclusions. First, incorporating environmental variables into the objective function produces significant reductions in water loss and salt intrusion, at the cost of increased vessel delays. Second, domain-driven adaptive mechanisms outperform general-purpose ones for the specific lockage scheduling problem of the North Sea Locks. The adaptive weights and environment-driven neighbourhood both show improvements, while Modified Lam Annealing and memory-based neighbourhood control, adopted from the literature, do not transfer effectively to this problem. Third, the practitioner's priority should guide component selection; when runtime predictability is valued, the fixed-iteration adaptive temperature is beneficial; when solution quality is the priority, the environment-driven adaptive weights provide the best improvements. A fully adaptive one-size-fits-all configuration does not emerge from the experiments, but the environment-driven SA shows potential towards an algorithm that is both more environmentally aware and operationally competitive with the existing Macomi baseline.

Chapter 8

Conclusions and Future Work

This chapter gives an overview of the thesis's contributions. We will reflect on the results of the environmentally aware lockage scheduling and the adaptive variants and draw conclusions based on these results. Finally, ideas for future work are discussed.

8.1 Contributions

This thesis aims to measure the effect of adaptive simulated annealing parameters on the minimisation of the environmental impact of the North Sea Lock lockage scheduling problem. To achieve this, we find what environmental variables are important in a lockage's cost function. Then, we measure the effects of adding these variables to the SA lockage scheduling algorithm. Finally, to alleviate the importance of manual tuning of SA hyper-parameters, we compare several adaptive SA approaches.

Based on a review of the hydraulic and ecological literature, we identify the loss of freshwater to the Western Scheldt and the intrusion of saline water into the freshwater Terneuzen–Gent canal as the critical environmental variables in a lockage of one of the North Sea Locks.

Extending Macomi's SA model with water loss and salt intrusion as environmental variables with the smallest weight already shows a great reduction in freshwater loss and the intrusion of saline water. This shows that significant gains in environmental impact can be achieved with efficient lockage scheduling. The reduction in environmental impact comes at the cost of significant vessel delays, quantifying the direct tradeoff between the environmental impact of a lockage schedule and the vessel delays caused by the schedule, as identified by Bakker and Van Koningsveld [10]. This tradeoff forms a design parameter, allowing the operator to balance the value of the environmental impact and the admissible increase in vessel delays.

With the crucial environmental variables identified and included in the model, we further extend the model by making the SA parameters adaptive. This alleviates the significance of the cumbersome, but crucial, hyper-parameter tuning process, while potentially improving performance. The effect of the adaptive hyper-parameters depends on whether the adaptive parameters approach is guided by domain-specific knowledge or by a general-purpose algorithm.

We evaluate two general-purpose adaptive SA approaches: Modified Lam Annealing for the adaptive cooling schedule and memory-based mutation operator selection for the adaptive neighbourhood control. Although both show promising results on different optimisation problems, they do not perform well for the North Sea Locks lockage scheduling optimisation problem. Modified Lam Annealing focuses on exploration of the search space, at the expense of the baseline's exploitation, leading to performance decreases. The memory-based mutation operator selection allows SA to utilise operators more frequently that show good historical performance. Even though the algorithm does identify operators which consistently outperform others, selecting these operators more frequently does not improve performance, indicating the importance of variety in the mutation selection. These findings show that general-purpose algorithms with strong performance on benchmark problems do not guarantee good results for specific real-world optimisation problems, such as the North Sea Locks optimisation problem.

However, domain-driven hyper-parameters do improve the model's solution quality. We evaluate adaptive cost function weights, which prioritise variables based on lock conditions. These results show that strategic shifting of these weights under crucial or favourable lockage times can reduce environmental impact, without influencing vessel delays. We combine the adaptive weights with adaptive neighbourhood control by mutation operator selection guided by environmental knowledge. This configuration shows promising performance across vessel delay, water loss, and salt intrusion, indicating that using problem-specific domain knowledge in optimisation is valuable.

Answering the main research question: adaptive SA parameters, when guided by domain-specific environmental knowledge, can meaningfully reduce the environmental impact, in the form of freshwater loss and salt intrusion, of the North Sea Locks. The greatest improvements are achieved by adapting cost function weights based on the lock conditions. The findings indicate that established general-purpose algorithms are not by definition the best approach for specific optimisation problems. For specific problems, investing in problem-specific adaptive mechanisms is advised.

8.2 Reflection

This thesis presents a quantitative evaluation of adaptive simulated annealing for environmentally-aware lock scheduling. However, some limitations call for reflection.

The input data for the experiments was selected to allow evaluation per season, since the environmental impact of lockages has strong seasonal dependencies. Water loss and salt intrusion are particularly problematic during summer drought periods, while excessive discharge is more likely to be a concern under high-water winter conditions. The domain-driven adaptive SA approaches are designed to account for these circumstances, so we expect that evaluation per season would highlight this functionality. Unfortunately, due to time constraints, we only performed five experiments per season, leading to insignificant results. The experiments on the aggregated data potentially mask the season-specific behaviour.

A second limitation concerns the hyper-parameters of the adaptive approaches. While the adaptive approaches reduce the sensitivity to empirically selected fixed hyper-parameters by adjusting them dynamically during the search, the adaptive approaches also depend on hyper-parameters, which were tuned empirically in this thesis. Tuning these parameters was not exhaustive, so better parameters may exist. In addition, this tuning was performed independently for each adaptive component. When multiple adaptive components are combined, their hyper-parameters may interact in ways that individual tuning does not account for. It is therefore possible that the configurations do not use optimal hyper-parameters for their adaptive algorithm.

8.3 Future work

Several directions for future work follow naturally from the findings and limitations of this thesis.

Following directly from the discussed limitations of this thesis, we vouch for further experiments focused on each season to obtain statistically significant results. Given the dependency of lockage scheduling on the seasons, results highlighting the algorithm's performance during drought and high-water seasons would be interesting.

To improve the second limitation in this research, we suggest parameter tuning of combined adaptive approaches. The fully adaptive algorithm uses the hyper-parameters tuned independently. Tuning could be more extensive. These parameters furthermore do not guarantee good performance when combined. Systematic tuning of the parameters of the adaptive algorithms could potentially show more significant improvements.

Finally, a more extensive evaluation of Macomi's full vessel simulation pipeline, which includes the SA lockage scheduling, would be interesting. This would further highlight the real-world impact of the extended SA algorithm. The preliminary full simulation results are promising, so a systematic study at this level would be a natural next step.

Bibliography

[1] North Sea Port. (2025) New lock terneuzen. North Sea Port. Accessed 3 June 2026. [Online]. Available: https://www.northseaport.com/en/project/new-lock-terneuzen

[2] W. Molenaar *et al.*, “Hydraulic structures: Locks,” 2020, lecture notes, Chapter 2.1.1 (original version 3 March 2011).

[3] P. Van Parys, “Nautisch café - Over water,” Tech. Rep., 2 2024.

[4] Central Commission for Navigation on the Rhine, “Inland waterway cargo handling in ports,” 2025, accessed: 2026. [Online]. Available: https://inland-navigation-market.org/chapitre/5-manutention-fluviale-dans-les-ports-2/

[5] W. G. Jackson, E. zcan, and R. John, “Tuning a simulated annealing metaheuristic for cross-domain search,” Donostia-San Sebastian, Spain. [Online]. Available: https://nottingham-repository.worktribe.com/output/871495

[6] P. Cabrera-Guerrero, G. Guerrero, J. Vega, and F. Johnson, “Improving simulated annealing performance by means of automatic parameter tuning,” *Studies in Informatics and Control*, vol. 24, no. 4, pp. 419–426, 2015. [Online]. Available: https://doi.org/10.24846/v24i4y201506

[7] Z. Wu, B. Ji, and S. S. Yu, “Modeling and solution algorithm for green lock scheduling problem on inland waterways,” *Mathematics*, vol. 12, no. 8, 2024. [Online]. Available: https://www.mdpi.com/2227-7390/12/8/1192

[8] R. Van der Brugge and R. C. de Winter, “Deltascenario’s 2024 - zicht op water in nederland,” 2024.

[9] Informatiepunt Leefomgeving. (2025) Verzilting – oorzaken, gevolgen en maatregelen. IPLO. Accessed: 2026-05-17. [Online]. Available: https://iplo.nl/thema/water/beheer-watersysteem/verzilting-oorzaken-gevolgen-maatregelen/

[10] F. P. Bakker and M. van Koningsveld, “Tool to evaluate countermeasures at locks to limit freshwater losses and saltwater intrusion while minimizing waiting times for

shipping," in *Proceedings of the 35th PIANC World Congress 2024*, J. S. Schoonees, Ed. PIANC, 2024, pp. 1558–1563.

[11] P. Brucker, Y. N. Sotskov, and F. Werner, "Complexity of shop-scheduling problems with fixed number of jobs: a survey," *Mathematical Methods of Operations Research*, vol. 65, no. 3, pp. 461–481, 2007. [Online]. Available: https://doi.org/10.1007/s00186-006-0127-8

[12] W. Passchyn, "Scheduling locks on inland waterways," Ph.D. dissertation, KU Leuven, 2016.

[13] J. Verstichel, P. De Causmaecker, F. Spieksma, and G. Vanden Berghe, "The generalized lock scheduling problem: An exact approach," *Transportation Research Part E: Logistics and Transportation Review*, vol. 65, pp. 16–34, 2014, special Issue on: MODELING, OPTIMIZATION AND SIMULATION OF THE LOGISTICS SYSTEMS. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1366554513002123

[14] F. Musharavati and A. Hamouda, "Simulated annealing with auxiliary knowledge for process planning optimization in reconfigurable manufacturing," *Robotics and Computer-Integrated Manufacturing*, vol. 28, no. 2, pp. 113–131, 2012. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0736584511000950

[15] N. M. Sadeh and Y. Nakakuki, "Focused simulated annealing search: An application to job shop scheduling," *Annals of Operations Research*, vol. 63, pp. 77–103, 1996.

[16] W. Swartz, "Automatic layout of analog and digital mixed macro/standard cell integrated circuits," Ph.D. dissertation, Yale University, New Haven, Connecticut, May 1993.

[17] V. Cicirello, "On the design of an adaptive simulated annealing algorithm," 09 2007.

[18] D. Delahaye, S. Chaimatanan, and M. Mongeau, "Simulated annealing: From basics to applications," in *Handbook of Metaheuristics*, M. Gendreau and J.-Y. Potvin, Eds. Springer, 2019, pp. 1–35.

[19] C. Cabrera, M. Fine-Morris, M. Pokross, K. Kish, S. Michalczyk, M. Cahn, H. Klei, and M. F. Russo, "Dynamically optimizing experiment schedules of a laboratory robot system with simulated annealing," *SLAS Technology*, vol. 19, no. 6, pp. 517–527, 2014.

[20] V. F. Yu, P. Jewpanya, A. P. Redi, and Y.-C. Tsao, "Adaptive neighborhood simulated annealing for the heterogeneous fleet vehicle routing problem with multiple cross-docks," *Computers & Operations Research*, vol. 129, p. 105205, 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0305054820303221

[21] H. Tarawneh and M. Ayob, "Adaptive neighbourhoods structure selection mechanism in simulated annealing for solving university course timetabling problems," *Journal of Applied Sciences*, vol. 13, pp. 1087–1093, 07 2013.

[22] L. Ingber, "Adaptive simulated annealing (asa): Lessons learned," 2000. [Online]. Available: https://arxiv.org/abs/cs/0001018

[23] M. Schultz van HaegenMaas Geesteranus and M. van Infrastructuur en Milieu, "Ontwerp Tracébesluit Nieuwe Sluis Terneuzen," Tech. Rep. VNZTR1577, 4 2015. [Online]. Available: https://www.commissiemer.nl/docs/mer/p28/p2875/2875-026otb.pdf

[24] F. P. Bakker, J. W. de Ruiter, A. J. van der Hout, and M. van Koningsveld, "Accurately forecasting saltwater intrusion through navigation locks requires nautical traffic simulation modelling," *Ocean Engineering*, vol. 355, p. 124918, 2026. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0029801826007523

[25] T. Vreeken and O. Weiler. (2021) Documentatie zeesluisformulering (libzsf). Deltares. [Online]. Available: https://libzsf.readthedocs.io/en/latest/index.html

[26] O. Weiler, T. O'Mahoney, and R. Burgers, "Zoutindringing schut- en spuissluizen," Kennisprogramma Natte Kunstwerken, Tech. Rep., 7 2019. [Online]. Available: https://www.nattekunstwerkenvandetoekomst.nl/document/opzet-en-verkenning-zeesluisformulering/

[27] N. Zuiderwijk, S. Maijvis, and T. O'Mahoney, "Validatierapport zeesluisformulering," Deltares, Tech. Rep., december 2025.

[28] Rijkswaterstaat. (2026) Waterbeheer. [Online]. Available: https://waterinfo.rws.nl/thema/Waterbeheer

[29] B. Biemond, V. Vuik, P. Lambregts, H. E. de Swart, and H. A. Dijkstra, "Salt intrusion and effective longitudinal dispersion in man-made canals, a simplified model approach," *Estuarine, Coastal and Shelf Science*, vol. 298, p. 108654, 2024. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0272771424000416

[30] B. Biemond and R. J. Labeur, "Adjustment time of the lock exchange for a linearly stratified lock and ambient," *Environmental Fluid Mechanics*, vol. 26, no. 1, p. 9, 1 2026. [Online]. Available: https://doi.org/10.1007/s10652-026-10073-5

[31] A. N. Shabrin and B. A. Ovrutskii, "Water exchange of stratified fluids through a lock," *Hydrotechnical Construction*, vol. 9, no. 8, pp. 765–770, 8 1975, translated from Gidrotekhnicheskoe Stroitel'stvo, No. 8, pp. 37–40, August, 1975. [Online]. Available: https://doi.org/10.1007/BF02382239

[32] I. Van Tongeren, personal communication, rijkswaterstaat.

[33] M. Deschamps, personal communication, departement Mobiliteit en Openbare Werken.

[34] C. Martens, G.-J. Liek, D. Maes, H. Haas, M. Deschamps, L. Dekker, H. van't Westeinde, M. Vermeirssen, H. Verbeek, and K. Storm, "Building a decision support system for the terneuzen locks: combining optimal management for water and shipping," in *PIANC-World Congress Panama City, Panama 2018*, 2018.

[35] S. Kirkpatrick, C. Gelatt, and M. Vecchi, "Optimization by simulated annealing," *Science (New York, N.Y.)*, vol. 220, pp. 671–80, 06 1983.

[36] P. J. M. van Laarhoven and E. H. L. Aarts, *Simulated annealing*. Dordrecht: Springer Netherlands, 1987, pp. 7–15. [Online]. Available: https://doi.org/10.1007/978-94-015-7744-1_2

[37] M. Mens, J. Pouwels, O. Weiler, J. Delsman, and Y. Huismans, "De zoetwaterbalans van laag Nederland in een warmer klimaat," Deltares, Tech. Rep. 11210362-000-ZWS-0001, 2 2024. [Online]. Available: https://www.deltares.nl/expertise/publicaties/zoetwaterbalans-van-laag-nederland-in-een-warmer-klimaat

[38] M. van Infrastructuur en Waterstaat, "Verdeling water bij droogte," 7 2025. [Online]. Available: https://www.rijkswaterstaat.nl/water/waterbeheer/droogte-en-watertekort/verdeling-water-bij-droogte

[39] F. van den Berg and A. Koelewijn, "Degradatie van dijken door dieren en droogte : uitwerking observaties graverijen en scheuren," Deltares, Tech. Rep., december 2025.

[40] Staatsbosbeheer, "In dienst van de toekomst," 11 2025. [Online]. Available: https://www.staatsbosbeheer.nl/-/media/07-over-staatsbosbeheer/ondernemingsplan-staatsbosbeheer-2026-2030.pdf

[41] M. van Justitie en Veiligheid, "Vitale infrastructuur en Wet weerbaarheid kritieke entiteiten," 3 2026. [Online]. Available: https://www.nctv.nl/onderwerpen/v/vitale-infrastructuur

[42] M. van Infrastructuur en Waterstaat, "Maatregelen voor de scheepvaart," 3 2026. [Online]. Available: https://www.rijkswaterstaat.nl/water/waterbeheer/droogte-en-watertekort/maatregelen-voor-de-scheepvaart

[43] H. Jungen, P. Specht, J. Ovens, and T. Friedrich, "State-of-the-art and baseline for the seamless use cases," SEAMLESS Project, Tech. Rep. Deliverable D2.1, 2024, accessed: 2025-02-19. [Online]. Available: https://www.seamless-project.eu/wp-content/uploads/2024/04/D2.1-State-of-the-art-and-baseline-for-the-SEAMLESS-Use-Cases_v1.0.pdf

[44] Rijkswaterstaat, "Operationeel Watermanagement - Kanaal Gent-Terneuzen," Rijkswaterstaat, Tech. Rep. WVL0723ZB174F, 7 2023. [Online]. Available: https://iplo.nl/publish/pages/188980/watersystemen-kanaal-gent-terneuzen.pdf

[45] E. Van Zanten, L. Hermans, V. Vuik, and P. Lambregts, "Verzilting kanaal gent-terneuzen." [Online]. Available: https://www.vliz.be/imisdocs/publications/392086.pdf

[46] Omroep Zeeland, "Binnenvaartschippers verwachten flinke files bij nieuwe sluis terneuzen," Online news article, augustus 2022, retrieved 1 May 2026. [Online]. Available: https://www.omroepzeeland.nl/nieuws/14872973/binnenvaartschippers-verwachten-flinke-files-bij-nieuwe-sluis-terneuzen

[47] J.-K. Verschuure, "Tijdelijke maximale diepgangbeperking scheepvaart gent-terneuzen," Online news article, augustus 2025, retrieved 1 May 2026. [Online]. Available: https://www.flows.be/scheepvaart/2025/08/tijdelijke-maximale-diepgangbeperking-scheepvaart-gent-terneuzen/?gdpr=accept

[48] W. Schouppe, "Minder vracht op schepen kanaal Gent-Terneuzen door droogte, gestremd verkeer aan de sluizen — VRT NWS: nieuws," 8 2022. [Online]. Available: https://www.vrt.be/vrtnws/nl/2022/08/04/minder-vracht-op-schepen-kanaal-gent-terneuzen-door-droogte-ges/

[49] A. Dai, "Increasing drought under global warming in observations and models," *Nature Climate Change*, vol. 3, pp. 52–58, 2013. [Online]. Available: https://doi.org/10.1038/nclimate1633

[50] M. H. Moreira, N. H. They, L. R. Rodrigues, L. Alvarenga-Lucius, and A. Pita-Barbosa, "Salty freshwater macrophytes: the effects of salinization in freshwaters upon non-halophyte aquatic plants," *Science of The Total Environment*, vol. 857, p. 159608, 2023. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0048969722067080

[51] M. S. Schuler, M. Cañedo-Argüelles, W. D. Hintz, B. Dyack, S. Birk, and R. A. Relyea, "Regulations are needed to protect freshwater ecosystems from salinization," *Philosophical Transactions of the Royal Society B: Biological Sciences*, vol. 374, no. 1764, p. 20180019, 2018.

[52] M. Ahmed, Z. Tóth, and K. Decsi, "The impact of salinity on crop yields and the confrontational behavior of transcriptional regulators, nanoparticles, and antioxidant defensive mechanisms under stressful conditions: A review," *International Journal of Molecular Sciences*, vol. 25, no. 5, p. 2654, 2024. [Online]. Available: https://doi.org/10.3390/ijms25052654

[53] K. Zakowski, M. Narozny, M. Szocinski, and K. Darowicki, "Influence of water salinity on corrosion risk—the case of the southern baltic sea coast," *Environmental Monitoring and Assessment*, vol. 186, no. 8, pp. 4871–4879, 2014. [Online]. Available: https://doi.org/10.1007/s10661-014-3744-3

[54] B. Valdez, J. Ramirez, A. Eliezer, M. Schorr, R. Ramos, and R. Salinas, "Corrosion assessment of infrastructure assets in coastal seas," *Journal of Marine*

Engineering & Technology, vol. 15, no. 3, pp. 124–134, 2016. [Online]. Available: https://doi.org/10.1080/20464177.2016.1247635

[55] M. Jorritsma, "Verzilting: een bedreiging voor ons drinkwater?" 8 2025. [Online]. Available: https://www.drinkwaterplatform.nl/verzilting-drinkwater/

[56] J. Lee, B. Biemond, H. de Swart, and H. A. Dijkstra, "Increasing risks of extreme salt intrusion events across european estuaries in a warming climate," *Communications Earth & Environment*, vol. 5, no. 1, p. 60, 2024. [Online]. Available: https://doi.org/10.1038/s43247-024-01225-w

[57] T. Schindler, C. Greulich, D. Bode, A. Schuldt, A. Decker, and K.-D. Thoben, "Towards intelligent waterway lock control for port facility optimisation," in *Dynamics in Logistics*, M. Freitag, H.-D. Haasis, H. Kotzab, and J. Pannek, Eds. Cham: Springer International Publishing, 2020, pp. 32–41.

[58] J. Verstichel, P. De Causmaecker, and G. V. Berghe, "Scheduling algorithms for the lock scheduling problem," *Procedia - Social and Behavioral Sciences*, vol. 20, pp. 806–815, 2011, the State of the Art in the European Quantitative Oriented Transportation and Logistics Research – 14th Euro Working Group on Transportation & 26th Mini Euro Conference & 1st European Scientific Conference on Air Transport. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1877042811014698

[59] E. Mokotoff, "Parallel machine scheduling problems: A survey," *Asia - Pacific Journal of Operational Research*, vol. 18, no. 2, pp. 193–242, 11 2001. [Online]. Available: https://www.proquest.com/scholarly-journals/parallel-machine-scheduling-problems-survey/docview/204764435/se-2

[60] X. Zhang, X. Yuan, and Y. Yuan, "Improved hybrid simulated annealing algorithm for navigation scheduling for the two dams of the three gorges project," *Computers & Mathematics with Applications*, vol. 56, no. 1, pp. 151–159, 2008. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0898122107008346

[61] L. Fraga-Gonzalez, R. Fuentes-Aguilar, A. García-González, and G. Sanchez-Ante, "Adaptive simulated annealing for tuning pid controllers," *AI Communications*, vol. 30, pp. 1–16, 08 2017.

[62] A. Eiben, R. Hinterding, and Z. Michalewicz, "Parameter control in evolutionary algorithms," *IEEE Transactions on Evolutionary Computation*, vol. 3, no. 2, pp. 124–141, 1999.

[63] F. Romeo and A. Sangiovanni-Vincentelli, "A theoretical framework for simulated annealing," *Algorithmica*, vol. 6, no. 1, pp. 302–345, Jun 1991. [Online]. Available: https://doi.org/10.1007/BF01759049

[64] N. Azizi and S. Zolfaghari, "Adaptive temperature control for simulated annealing: a comparative study," *Computers & Operations Research*, vol. 31, no. 14, pp. 2439–2451, 2004. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0305054803001977

[65] E. Triki, Y. Collette, and P. Siarry, "A theoretical study on the behavior of simulated annealing leading to a new cooling schedule," *European Journal of Operational Research*, vol. 166, no. 1, pp. 77–92, 2005, metaheuristics and Worst-Case Guarantee Algorithms: Relations, Provable Properties and Applications. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0377221704003388

[66] J. Lam and J.-M. Delosme, "Performance of a new annealing schedule," in *Proceedings of the 25th ACM/IEEE Design Automation Conference*. IEEE Computer Society Press, 1988, pp. 306–311.

[67] J. A. Boyan, "Learning evaluation functions for global optimization," Ph.D. dissertation, Carnegie Mellon University, Pittsburgh, Pennsylvania, August 1998.

[68] V. A. Cicirello, "Optimizing the modified lamannealing schedule," *EAI Endorsed Transactions*, 2020.

[69] S. Ropke and D. Pisinger, "An adaptive large neighborhood search heuristic for the pickup and delivery problem with time windows," *Transportation Science*, vol. 40, no. 4, pp. 455–472, 2006.

[70] E. Burke, M. Hyde, G. Kendall, G. Ochoa, and E. Özcan, *A classification of hyper-heuristic approaches*, 01 2010, pp. 449–468.

[71] C. Barrass and D. Derrett, "Chapter 5 - form coefficients," in *Ship Stability for Masters and Mates (Seventh Edition)*, seventh edition ed. Oxford: Butterworth-Heinemann, 2012, pp. 37–43. [Online]. Available: https://www.sciencedirect.com/science/article/pii/B9780080970936000050

[72] F. P. Bakker, personal communication, tU Delft Civil Engineering and Geoscience.

[73] S. Nison, *Japanese Candlestick Charting Techniques*, 2nd ed. New York Institute of Finance, 2001.

[74] Student, "The probable error of a mean," *Biometrika*, vol. 6, no. 1, pp. 1–25, March 1908.

[75] Y. Benjamini and Y. Hochberg, "Controlling the false discovery rate: A practical and powerful approach to multiple testing," *Journal of the Royal Statistical Society: Series B (Methodological)*, vol. 57, no. 1, pp. 289–300, 1995.

[76] F. Wilcoxon, "Individual comparisons by ranking methods," *Biometrics Bulletin*, vol. 1, no. 6, pp. 80–83, December 1945.

Appendix A

AI Disclosure

In the process of this research, Artificial Intelligence (AI), in the form of Large Language Models, was used. In the exploratory phase of getting to know Macomi's codebase, ChatGPT (GPT-5) is used to help understand existing code by summarising code snippets into text.

Claude Sonnet 4.6 is also used to assist in writing the Python code for the data processing pipeline and the implementation of the statistical tests on the experiment results. The statistical methodology, including the choice of tests, the interpretation of results, and all conclusions drawn from them, was determined and verified independently by the author.

While writing the thesis, ChatGPT and Claude are used for small changes in sentence structure or grammar to fit the formal thesis style. ChatGPT also helped in formatting LaTeX objects such as figures and tables. Grammarly is used to correct spelling and grammar mistakes.

AI was at no point used to generate new ideas. Output by AI models is always manually checked for correctness. All ideas and conclusions are original.