Erasmus University Rotterdam

Erasmus School of Economics

Ergonomic Workforce Scheduling: A Multi-Objective Approach

Master Thesis

Econometrics and Management Science:

Operations Research and Quantitative Logistics

Author

Hazel Alberts, 416297

Supervisor

Dr. W. van den Heuvel

Supervisors ORTEC

J. Augustijn

A. van de Wetering, MSc

Second assessor

Dr. D. Galindo Pecin

November 27, 2020

Abstract

In this thesis, we present a multi-objective solution approach for ergonomic workforce scheduling. We show that we are able to create balanced schedules with respect to multiple conflicting healthful scheduling criteria. The multi-objective approach presented is able to provide a good approximation of the Pareto curve. Also, it is able to find non-supported Pareto points, which are out of reach for methods that use a weighted sum objective function such as the one used in ORTEC Workforce Scheduling. The approach in this thesis combines goal programming with a metaheuristic. The goal programming component provides an initial solution as well as ideal solutions. The initial solution is used as a starting point for the metaheuristic. An ideal solution is found for each healthful scheduling goal. The ideal solution with respect to a goal is the solution to a mixed-integer program with only this goal in the objective function. These ideal solutions are used to measure the fitness of solutions found by the metaheuristic. As the metaheuristic component, we implement the falling tide algorithm and a simulated annealing algorithm. The falling tide algorithm is shown to perform better than a simulated annealing approach. Also, we add a Variable Neighbourhood Search component to falling tide and find that this improves its performance. Using the combined approach described, we are able to make schedules with a good trade-off among several objectives.

Contents

1 Introduction

ORTEC provides its customers in the healthcare industry with Workforce Scheduling software (OWS). As the name suggests, this software is used for making schedules for healthcare employees. So, it is a solution approach to solve the well-known Nurse Scheduling Problem. The NSP is the problem of assigning shifts to nurses in an optimal way. This is done under a given set of hard constraints and some objective function. Solutions to the NSP are also applicable outside of the medical field. The objective of this thesis is to provide a solution to the Multi-Objective Nurse Scheduling problem, presenting planners with flexible timetables that contribute positively to overall employee health.

The extent to which the scheduling process is automated differs per planner. Some planners lean more towards the manual planning side while others make use of the optimizer module to varying degrees. In either case, the resulting schedule may fall short in terms of customer satisfaction. This is despite the fact that the program allows for the objective function to be tailored to the needs of the user when the optimizer is used. So, we seek to provide an alternative algorithm that provides the planner with a result that consists of several options to choose from.

The current optimizer may benefit from some additional key performance indicators that are conducive to health. In the current framework, it is possible to weight different KPIs so as to create a trade-off in a single objective function. This does mean that the optimizer provides a single schedule as its solution. The planner does have the option to change individual shifts in the schedule of the resulting solution. However, there are many variables, making it difficult to make effective changes manually.

So, it may be useful to provide the customer with several options of good schedules with respect to the KPIs. The first aim of this thesis is to provide schedules that take health indicators into account. To do this we will use an alternative method which provides some Pareto optimal schedules that the customer may choose from. We approach the problem as a multi-objective one, where each health indicator is translated to a soft constraint that the algorithm attempts to satisfy as well as possible. As mentioned, in OWS these indicators are part of one objective function with weights, resulting in a weighted objective value at termination. Conversely, in the case of the approach is this thesis, each goal is considered in a separate objective function. This approach creates solutions that are the result of a trade-off among these goals, resulting in several schedules, which each score differently on the health indicators. Also, the methods used here are able to find some solutions that an approach with a single weighted objective function cannot.

Moreover, the planners do not have direct insight into the healthfulness of their schedule. Especially in the case of a largely manually constructed schedule, providing insight into health effects such as sleep deprivation is likely to be beneficial. Sleep deprivation may result from working many night shifts or irregularity of shifts, disrupting the employee's sleep pattern. A metric encompassing a variety of health indicators will give the planner a better understanding of their scheduling choices, possibly leading to healthier schedules for the employees. Therefore, the second goal of this research is to provide a useful metric for assessing schedule healthfulness. This metric is the fitness value of the schedule and is made up of a combination of health criteria.

The structure of this thesis is as follows. Section 2 describes the problem in detail. Section 3 described the motivation behind this research and the academic and practical relevance. We present the existing literature on healthful scheduling as well as the NSP in section 4. The mathematical formulation for the mixed-integer program is given in section 5. The methodology is given in section 6. The results are presented in section 7. Finally, we provide a discussion and conclude in section 8.

2 Problem Description

To achieve the aims described in section 1, we will use criteria to ascertain the healthfulness of schedules. The following are the health indicators as described by Groen et al. (2019). These have been translated from Dutch.

2.1 Shift Scheduling Criteria

The criteria treated in this section are the focus of this thesis. They are health indicators that can be influenced by the way that existing shifts are scheduled. These existing shifts are created by the healthcare institutions and will not be changed in this thesis. These criteria are translated to mathematically formulated soft constraints. These formulations are given in Section 5.6.

1. Allowing a maximum of 2-3 consecutive night shifts.
2. Allowing a maximum of 3-4 shifts in a row (before a rest day).
3. Ensuring 2 rest days after a sequence of 4-5 shifts and 3 rest days after each 6-shift sequence.
4. Regularity: it is desirable to have a familiar pattern of shifts
5. A forward rotating shift system is advised. This means that each sequence of shifts goes from a day shift to an evening shift to a night shift.
6. Speed of rotation: it's advised to have 2 to 3 identical shifts in a row.
7. Two breaks of 15 minutes per shift and one of 30 minutes.

Criterion 7 is an exception and is incorporated in the model as hard constraints. All given shifts have a standard 30 minute break. So, to adhere to criterion 7 and have an hour break in total, we add 30 minutes to the relevant rest constraints in section 5.5. The details will be given in that section.

2.2 Shift Property Criteria

The following indicators are determined by properties of the shifts. As mentioned, shifts are created by the healthcare institutions and we will not make changes to them. So, it is beyond the scope of this thesis to investigate changes of these properties. We present these criteria to indicate that they are relevant and advised by research in the field of ergonomic workforce scheduling.

8. Shift length should be limited in case of especially demanding work.
9. The maximum length of a shift should be 9 hours and 8 hours for night shifts
10. The starting time of a shift should typically be no earlier than 7.00 AM.

2.3 Institution Policy Criteria

The following criteria are inevitably dependent on the policies of the relevant healthcare institution. So, they are also outside of the scope of this thesis.

11. Predictability: Last-minute schedules changes should be limited
12. For shifts in the evenings and weekends it's advisable to take personal preference of the employees into account.

Criterion 12 can be implemented as hard constraints. Specifically, if an employee does not wish to work on a particular day, a hard constraint can be added that enforces this wish. The same can of course be done in case the employee does wish to work on any given day. Although we do not do this in this thesis, this would entail adding a simple hard constraint that fixes the value of the relevant variable.

2.4 Use of Health Criteria in the Model

The criteria presented in the previous sections will be used as soft constraints in the model. We will also refer to them as goals. The algorithm described in section 6 will minimize the deviation from these goals. This implies that the algorithm attempts to adhere to them as strictly as possible.

Although some of the criteria contradict one another, this does not present a problem. For example, criterion 2 demands that there are no more than 4 shifts in a row and criterion 3 demands a minimum of 3 rest days after a 6-shift sequence. This is to ensure that, should soft constraint 2 be violated, we still require adequate rest after a 5- or 6-shift sequence.

3 Motivation and Relevance

In this section, we discuss the motivation behind this thesis as well as its practical and academic relevance.

The motivation of this research is twofold. First, there is the direct benefit to healthcare employees. As described in sections 2, this thesis focuses on solving a multi-objective NSP, where the objectives are health criteria. So, the use of this algorithm for scheduling is expected to have a positive effect on employee health and well-being. Second, there is the possibility of alleviating some of the pressure due to the capacity challenges commonly experienced in this sector. These challenges are a result of healthcare institutions being understaffed.

According to the Dutch Employee Insurance Agency (UWV), the staff shortages across the healthcare sector are at a level that is leading to excessive workloads for employees. This is a well-known problem in the Netherlands which is regularly discussed within the government and Dutch society. In section 4.1, we discuss research on the topic of ergonomic scheduling. It is clear that excessive workloads have a negative effect on healthful scheduling indicators. Part of the resulting negative health effects for employees may be mitigated by an increase in focus on their well-being through scheduling.

At the time of writing this thesis, the world is in the midst of the COVID-19 pandemic. The situation has brought the underlying issues of staff shortages to the forefront. This clearly adds to the motivation of this research.

Furthermore, this research has practical as well as academic relevance. The practical relevance relates to the use of workforce scheduling software by healthcare institutions. As mentioned in section 2, ORTEC has a workforce scheduling product to solve the well-researched Nurse Scheduling Problem. The current method is a Genetic Algorithm combined with Variable Neighbourhood Search. In this thesis, we provide an alternative algorithm for solving the NSP.

The method used in this thesis has not been used before by ORTEC in its work scheduling software. Also, it differs from the current optimizer in a number of ways. It consists of a goal programming approach and a metaheuristic. The former is used to create initial solutions. The metaheuristic component is the falling tide algorithm (Li et al., 2012), which is an approach based on the great deluge algorithm (Dueck, 1993). The method provides the planner with a number of solutions from which they can choose. This flexibility is likely to be important in terms of practical applicability as the customer will have more control over the outcome. They may therefore be more satisfied with the resulting schedule.

Section 4.2 discusses academic literature on the NSP, including other heuristic approaches. There are only a couple of papers that present an approach to NSP as a multi-objective problem with separate objectives instead of one weighted objective function. Furthermore, the falling tide algorithm in particular has not been extensively researched since its introduction by Li et al. (2012). So, it is relevant to investigate the performance of this method in a different context, where the constraints and data differ. Also, in this thesis we add a VNS component to the algorithm and compare it to a simulated annealing approach. These two elements form an addition to existing research as they have not been investigated before in the literature in combination with a falling tide approach to the NSP problem.

4 Literature

We will approach the described problem as a Nurse Scheduling Problem (NSP). This is an NP-hard problem. Many papers have been published about algorithms which attempt to find solutions that are computationally viable. In this section we aim to put into context the approach used in this thesis. In section 4.1, we discuss research on healthful scheduling, including the basis for the health indicators used as goals in the model. In section 4.2 we discuss the literature on work scheduling and justify the use of the falling tide algorithm in the context of the existing literature. First, we discuss the basis for a heuristic approach in section 4.2.1. Then, in 4.2.2, we discuss the existing research on multi-objective approaches as well as the basis for using an adaptation of the falling tide algorithm.

4.1 Health Indicators

The criteria for scheduling healthfully are based on research in the area of shift work and health. Groen et al. (2019) summarize several studies on this topic and present the criteria for a healthful schedule used in this thesis. These indicators are partially based on the findings of Knauth & Hornberger (2003). They make a number of ergonomic recommendations that they expect will have positive health effects. These are based on, among other things, reducing sleep deprivation, allowing for adequate social contact, reducing the probability of accidents happening, preventing negative long-term health effects, and providing sufficiently long periods of leisure time.

Hulsegge et al. (2020) compare some health indicators between shift workers and non-shift workers. They find that, compared to non-shift workers, shift workers were more likely to suffer from burnout or distress if they perceived their schedules to impact their private lives negatively. A similar difference is found in employees that are generally dissatisfied with their schedules. In the case of burnout, the opposite effect is found for shift workers who are satisfied and/or experiencing no impact of work schedules on their private life.

Gerber et al. (2017) find negative effects of different types of physical activity on the probability of developing sleep complaints. Furthermore, they find that shift work in itself does not necessarily contribute to the probability of developing sleep complaints.

Åkerstedt (2003) argues that there is no concrete evidence that the direction of shift rotation matters when it comes to sleep quality. This contradicts a recommendation made by Knauth & Hornberger (2003) to adopt forward rotation so as to reduce circadian rhythm disruption as well as findings by Sallinen & Kecklund (2010).

4.2 Algorithm

In general, there is a hierarchy to consider when designing an algorithm. Ideally, we would use a polynomial time algorithm. However, as stated before, the Nurse Scheduling Problem is known to be NP-hard. When a polynomial time algorithm is not an option, an alternative is to use a fast non-polynomial time algorithm. In the case of the NSP, this is often a Branch and Price approach, for instance in Legrain et al. (2019). This type of approach may still be too slow for large instances and is not convenient for a multi-objective approach. The next best option would be a fast approximation algorithm. After analyzing the literature, we conclude that there seems to be no such algorithm for the NSP. Importantly, the use of a metaheuristic is desirable here due to the multi-objective nature of the problem. There are also exact multi-objective approaches, most notably goal programming. However, this approach is known to be time consuming and does not perform well in terms of the objective value (Li et al., 2012). Moreover, it produces one solution. As discussed in the problem description, my goal is to give the planner several solutions, thereby providing a flexible result. In this thesis, we will use goal programming for an initial solution and combine it with a metaheuristic approach for a complete, well-performing algorithm.

Here, we discuss some heuristic approaches that have been studied before and we will substantiate the choice of the falling tide algorithm.

4.2.1 Heuristic Approaches

Sarkar et al. (2019) present a study comparing a variety of approaches to the NSP. They found that metaheuristics performed better than other considered methods regarding, among other factors, cost minimization and time complexity. The other considered methods included using a mathematical programming approach. Several of the investigated meta heuristics use some variation of an evolutionary algorithm (EA). Also, Mutingi & Mbohwa (2014) and Inoue et al. (2003) use evolutionary algorithms as an approach to this problem. Peters et al. (2019) present an evolutionary algorithm that finds a better objective value than a MIP approach in the same runtime. Compared to an upper bound on the objective value they find a 1.7% gap in the EA case and a 10.5% gap with the MIP approach, having used a time limit of 5 hours. Lin et al. (2015) describe a Memetic Algorithm, a type of evolutionary algorithm. This algorithm differs from some of the other evolutionary algorithm approaches mentioned in that there is a clear focus on the preference of the nurses. Jafari & Haleh (2019) use both a MIP and a simulated annealing (SA) approach to solve the NSP with a focus on nurse preferences. They find good quality solutions using the SA approaching, using the MIP bounds as benchmarks.

For a general literature review on personnel scheduling, we refer the reader to Van den Bergh et al. (2013).

4.2.2 Multi-Objective Approaches

Tan et al. (2019) propose a multi-objective MIP approach to schedule emergency room resident in the West China Hospital of Sichuan University, treating employees' personal preferences as soft constraints. The approach consists of two stages and is used to improve the shift schedules, which were previously created manually. The authors state that their new method helps relieve work pressure and increase service quality in the hospital.

Caballini & Paolucci (2019) consider two types of MIP multi-objective approaches. One in which 3 objectives are combined into one with scalar weights and one in which the problem is solved 3 times, each time with a single objective function in the order of importance. The latter is known as a hierarchical optimization approach. They use their approach to minimize health risks at an Italian port in Genoa.

Burke et al. (2012) use a pareto-based search for a multi-objective approach to the NSP. They begin by creating feasible solutions using an Improved Squeaky Wheel Optimization (ISWO) search along with a repair heuristic. This is to create solutions that satisfy all the hard constraints. Once they obtain feasible solutions, the next step is a simulated annealing-based meta-heuristic. This is used to get solutions to adhere to the so-called soft constraints in an approximately pareto optimal way. In the case of this thesis, the health criteria are examples of such constraints. Burke et al. (2012) compare their results to three other approaches and find that they are able to find better objective values when looking at a single weighted sum objective. This is not the only advantage of this approach, that is, they are able to satisfy almost all soft constraints and provide a range of non-dominated schedules. However, there is a method that provides even better results, namely the falling tide-based approach in Li et al. (2012).

Li et al. (2012) propose the falling tide algorithm, an approach that combines goal-programming and meta-heuristics to find (approximately) pareto-optimal solutions to the NSP applied in a Dutch hospital. When compared to other methods as Burke et al. (2012) did, they find high quality solutions. In fact, when comparing their method to others using a single weighted objective value, they found their method to find lower (better) objective values in much faster runtimes. Three of the compared methods were a hybrid GA approach, a hybrid VNS approach (Post & Veltman, 2004) and an Integer Programming-based VNS approach (Burke et al., 2010). These are the same methods that were considered in Burke et al. (2012). Li et al. (2012) find a lower objective value than Burke et al. (2012), again in considerably less time. Interestingly, in this last case Li et al. (2012) also compare results on a multi-objective level. So, they compare the approximate Pareto sets of the two approaches using the two set coverage metric, an existing performance metric for multi-objective approaches. This metric shows to which extent solutions from one set dominate those of the other. Both sets are approximations of the Pareto front (Li et al., 2012). Specifically, the metric is used to calculate what percentage of solutions in Burke et al. (2012) are dominated by one or more solutions in Li et al. (2012). They find that for all data sets used, over 70% of solutions found by Burke et al. (2012) are dominated by solutions found by the falling tide algorithm. Moreover, they investigate the resemblance of their approximated pareto front to the true pareto front and find that they are likely close due to the very limited violation of soft constraints.

Li et al. (2012) describe their approach to be useful in problems with many constraints. Also, instead of addressing a simplified version of the NSP, they've made an approach that is adjustable to real-world scenarios (Li et al., 2012) by, among other things, accounting for many soft constraints in an efficient way. The outcome provides the customer with a number of solutions to choose from as opposed to a single solution with an optimal weighted objective value. The authors conclude that their method is likely suitable to be adapted for problems with different sets of constraints. So, we expect using constraints based on Groen et al. (2019) will yield good results.

As an addition to the existing literature, we will adapt the approach by Li et al. (2012). The authors mention that there is room for further research into performance improvement of the falling tide algorithm. My adaptation of the algorithm is explained in more detail in section 6.

5 Mathematical Formulation

This formulation is used for the goal programming part of the algorithm. The objective function as well as an explanation of the goal programming approach is given in section 6.

5.1 Definitions

Here, we give some definitions of terms used in definitions of the parameters and sets.

- A day k is a *rest day* for employee i if they are not assigned any shifts that start on day k.
- Unless otherwise specified, the *starting time* and *ending time* of a shift are relative to the length of the planning period. A starting time of 50 indicates that the shift starts at 2:00 on the third day.

5.2 Parameters

Here, we describe all the parameters used in the mathematical formulaton.

l_s = the length of shift s in hours

b_s = starting time of shift s in hours

e_s = ending time of shift s in hours

d_s = starting day of shift s

g_s = ending day of shift s

n_s = parameter with value 1 if s is a night shift, 0 otherwise

w_s^t = week day parameters with value 1 if the shift takes place on weekday t.

v_{sp} = parameter with value 1 if shift s requires skill p

y_{ip} = parameter with value 1 if employee i has skill p

a_i = weekly availability of employee i in hours

5.3 Sets

In this section, we present all the relevant sets along with their definitions.

5.3.1 Short Descriptions

I = the set of employees (indexed by i)

S = the set of shifts in the planning horizon (indexed by s)

D = the set of days in the planning horizon (indexed by d)

W = the set of weeks in the planning horizon (indexed by w)

M = the set of months in the planning horizon (indexed by m)

N_d = the set of night shifts starting on day d (indexed by s)

S_d = the set of shifts starting on day d (indexed by s)

P = the set of skills (indexed by p)

T^t = the set of weekdays of type t in the planning horizon e.g. T^{Sun} is the set of Sundays in the planning horizon (indexed by γ)

5.3.2 Detailed Descriptions

The following sets are given more detailed descriptions.

S^r = pairs of shifts of which the time in between is not more than the time required for rest. Specifically, it should hold for the set $\{(s, s') : b_{s'} - e_s \leq r, e_s \leq b_{s'}, s, s' \in S\}$.

S^b = the set of pairs of shifts (s, s') that take place on two consecutive days, with shift s' taking place after shift s, and the starting time of shift s' being more than 3 hours earlier than that of shift s relative to the length of a day. Specifically, it should hold for the set $\{(s, s') : b_{s'} - b_s \leq 21, s \in S_d, s' \in S_{d+1}, d \in D\}$.

S^e = the set of pairs of shifts (s, s') that take place on two consecutive days, with shift s' taking place after shift s, and the starting time of shift s' and s being unequal relative to the length of a day. Specifically, it should hold for the set $\{(s, s') : b_{s'} - b_s \neq 24, s \in S_d, s' \in S_{d+1}, d \in D\}$.

5.4 Variables

In this section, we describe the decision variables used in the mathematical formulation.

$$x_{is} = \begin{cases} 1 & \textit{if employee } i \textit{ is assigned to shift } s \\ 0 & \textit{otherwise} \end{cases}$$

$$r_{id} = \begin{cases} 1 & \textit{if employee } i \textit{ has a rest day on day } d \\ 0 & \textit{otherwise} \end{cases}$$

$$z_i \in \mathbb{B} : \text{ used for if-then constraints (equations 9 and 10)}$$

5.5 Hard Constraints

Here, I present the hard constraints. These apply to all instances. The soft constraints can be found in section 5.6 .

5.5.1 General Hard Constraints

Complete each shift exactly once

These constraints ensure that each shift is completed and that no individual shift is completed more than once.

$$\sum_{i \in I} x_{is} = 1 \qquad \forall s \in S \tag{1}$$

Complete one shift per day

We assume that each employee should complete no more than one shift a day. The following constraint enforces this.

$$\sum_{s \in S_d} x_{is} \leq 1 \qquad \forall d \in D,\ i \in I \tag{2}$$

Each employee works according to their availability

Employees have contracts that stipulate their weekly availability. This constraint ensures they do not work longer hours than allowed by these contracts.

$$\sum_{s \in S_w} l_s \cdot x_{is} \leq a_i \qquad \forall i \in I\ w \in W \tag{3}$$

Connecting x and r variables

r_{id} is a rest variable indicating an entire day is a rest day.

$$r_{id} = 1 - \sum_{s \in S_d} x_{is} \qquad \forall i \in I,\ d \in D \tag{4}$$

5.5.2 Labour Regulation Hard Constraints

The following are hard constraints that are relevant for the departments considered in this thesis. These follow from Dutch labor regulations. To limit the number of explicit hard constraints needed, labor regulations that are not applicable to the shifts of the data sets used in this thesis are omitted. For instance, many labor regulations apply to on-call shifts, which are not needed in the departments considered in this thesis. The selection of hard constraints in order to comply with labor regulations was done in accordance with expert judgement of employees of ORTEC and planners that work in the field of shift scheduling for the healthcare industry. However, some regulations are omitted for the sake of simplicity.

Shifts must be completed by employees with matching skill set

$$x_{is} \leq \sum_{p \in P} y_{ip} \cdot v_{sp} \qquad \forall\, i \in I,\ s \in S \tag{5}$$

Minimum rest time after night shift

There should be at least 14 hours of rest between a night shift and the next shift. We add half an hour to adhere to criterion 7 in section 2.1.

$$x_{is} + x_{is'} \leq 1 \qquad \forall i \in I,\ (s, s') \in S^{14.5} \tag{6}$$

Maximum average working time per week

The total working time per week has to be less than 55 hours on average.

$$\sum_{s \in S} l_s x_{is} \leq 55 \cdot |W| \qquad \forall i \in I \tag{7}$$

Daily uninterrupted rest

According to labour regulations, employees should have 11 hours of uninterrupted rest per day. As a slight simplification, we implement the following constraint which enforces 11 hours of rest between each shift. We add half an hour to adhere to criterion 7 in section 2.1.

$$x_{is} + x_{is'} \leq 1 \qquad \forall i \in I,\ (s, s') \in S^{11.5} \tag{8}$$

Working time for more than 16 night shifts

Labour regulation: In a period of 16 weeks, the maximum average working time per week is 40 hours if the number of night shifts in this period exceeds 16.

$$\sum_{s \in S} n_s x_{is} - |W| \leq M_1 z_i \qquad \forall i \in I \tag{9}$$

$$\sum_{s \in S} l_s x_{is} - 40|W| \leq M_2(1 - z_i) \qquad \forall i \in I \tag{10}$$

Where $M_1 = max\{\sum_{s \in S} n_s x_{is} - |W|\} = |S| - |W|$ and $M_2 = max\{\sum_{s \in S} l_s x_{is} - 40|W|\} = \sum_{s \in S} l_s - 40|W|$

Number of Sundays Off

Labour regulation states that each employee should have 13 Sundays off in every 52 weeks. This constraint amounts to a little over 1 Sunday off per month on average. Because our goal is to create ergonomic schedules, we round this up to 2 Sundays per month.
In the case of a planning horizon of less than one month, this constraint is not active because fractional days off are not possible.

$$\sum_{s \in S} w_s^{Sun} x_{is} \leq |T^{sun}| - 2|M| \qquad \forall i \in I \tag{11}$$

5.6 Soft constraints

In this section, we formulate the criteria from Groen et al. (2019) as mathematical constraints. Also, we present the transformation of these constraints to soft constraints. Soft constraints are constraints that do not need to be satisfied in order to obtain a feasible solution. Violating these constraints leads to a penalty in the objective function. A higher frequency of violations of these soft constraints leads to a higher penalty. The mathematical equations that determine these penalties are presented here.

To include the soft constraints in the objective function, we use maximization operations. This is necessary to ensure that only unwanted deviations from constraint are penalized. However, this clearly leads to nonlinear terms in the objective function, meaning the formulation as a whole is no longer linear. There are straight-forward methods to linearize these maximization terms. The method used for this is described in appendix A.

For each constraint, we present the formula for the deviation from the goal d^x with the relevant subscripts. f_x is the sum of all these deviations, making it the total deviation from goal x.

Soft constraint 0: Max 2 night shifts in a row

For each employee and each day d in the planning period, a penalty is incurred if there is a series of 3 or more consecutive night shifts starting on day d.

$$\sum_{s \in (N_d \cup N_{d+1} \cup N_{d+2})} x_{is} \leq 2 \qquad \forall i \in I, d \in \{1, .., |D| - 2\} \tag{12a}$$

$$d^0_{id} = max\{0, \sum_{s \in (N_d \cup N_{d+1} \cup N_{d+2})} x_{is} - 2\} \tag{12b}$$

$$f_0 = \sum_{i \in I} \sum_{d \in \{1,..,|D|-2\}} d^0_{id} \tag{12c}$$

Soft constraint 1: Rest day after a series of 4 shifts

Here, there is a penalty for each series of 4 shifts that is not followed by a rest day.

$$\sum_{d=l}^{l+4} r_{id} \geq 1 \qquad \forall l \in \{1, .., |D| - 4\}, i \in I \tag{13a}$$

$$d^1_{il} = max\{0, 1 - \sum_{d=l}^{l+4} r_{id}\} \tag{13b}$$

$$f_1 = \sum_{i \in I} \sum_{l=1}^{|D|-4} d^1_{il} \tag{13c}$$

Soft Constraint 2: 2 rest days after a series of 5 shifts. Here, we ensure that 2 rest days are required in the case of a series of 5 shifts. Binary variables are used for this if-then mechanism.

The following binary variables are added to the model.

$$\eta_{il} \in \{0, 1\} \qquad \forall l \in \{1, ..., |D| - 6\},\ i \in I \tag{14}$$

$\eta_{il} = 1$ when there is no rest day in a series of 5 days starting on day l.

$\eta_{il} = 0$ when there is a rest day in a series of 5 days starting on day l.

A series of 4 shifts may only occur if the relevant η variable is equal to 0.

$$\sum_{d=l}^{l+4} r_{id} \geq (1 - \eta_{il}) \qquad \forall\, l \in \{1, ..., |D| - 6\},\ i \in I \tag{15}$$

If η_{il} is indeed equal to 0 for some i and l, then the series of 5 shifts must be followed by 2 rest days. If this is not the case, a penalty is added.

$$\sum_{d=l+5}^{l+6} r_{id} \geq 2\eta_{il} \qquad \forall l \in \{1, ..., |D|-6\},\ i \in I \tag{16a}$$

$$d_{il}^2 = max\{0, 2\eta_{il} - \sum_{d=l+5}^{l+6} r_{id}\} \tag{16b}$$

$$f_2 = \sum_{i \in I} \sum_{l=1}^{|D|-6} d_{il}^2 \tag{16c}$$

Soft Constraint 3: 3 rest days after a series of 6 shifts

This soft constraint is similar to soft constraint 2. Instead of 2 rest days after 5 shifts, we want to encourage 3 rest days after each series of 6 shifts. In this case, the binary variables are given by ϵ_{il} as formulated in equation 17.

$$\epsilon_{il} \in \{0, 1\} \qquad \forall l \in \{1, ..., |D|-8\},\ i \in I \tag{17}$$

$\epsilon_{il} = 1$ when there is no rest day in a series of 6 days starting on day l.

$\epsilon_{il} = 0$ when there is a rest day in a series of 6 days starting on day l.

The following constraint follows the same logic as the one given by equation 15.

$$\sum_{d=l}^{l+5} r_{id} \geq (1 - \epsilon_{il}) \qquad \forall\, l \in \{1, ..., |D|-8\},\ i \in I \tag{18}$$

Finally, we formulate the soft constraint in equations 19a through 19c.

$$\sum_{d=l+6}^{l+8} r_{id} \geq 3\epsilon_{il} \qquad \forall l \in \{1, ..., |D|-8\},\ i \in I \tag{19a}$$

$$d_{il}^3 = max\{0, 3\epsilon_{il} - \sum_{d=l+6}^{l+8} r_{id}\} \tag{19b}$$

$$f_3 = \sum_{i \in I} \sum_{l=1}^{|D|-8} d_{il}^3 \tag{19c}$$

Soft constraint 4: Regularity of shift pattern

Here, we attempt to let the employee start their shift at the same time each day. S^e is the set of pairs of shifts taking place on consecutive days that do not start at the same time.

$$x_{is} + x_{it} \leq 1 \qquad \forall i \in I,\ (s, t) \in S^e \tag{20a}$$

$$d^4_{i(s,t)} = max\{0, x_{is} + x_{it} - 1\} \tag{20b}$$

$$f_4 = \sum_{i \in I} \sum_{(s,t) \in S^b} d^4_{i(s,t)} \tag{20c}$$

Soft constraint 5: Forward Rotation

If employee i works on day k and day $k+1$, the shift worked on day $k+1$ should not start more than 3 hours earlier than the one worked on day k. Recall that S^b is the set of shift pairs on consecutive days, where the shift on day $k+1$ starts more than 3 hours earlier than the one on day k.

$$x_{is} + x_{it} \leq 1 \qquad \forall i \in I,\ (s,t) \in S^b \tag{21a}$$

$$d^5_{i(s,t)} = max\{0, x_{is} + x_{it} - 1\} \tag{21b}$$

$$f_5 = \sum_{i \in I} \sum_{(s,t) \in S^b} d^5_{i(s,t)} \tag{21c}$$

Soft constraint 6 and 7: Speed of Rotation

Here, the aim is to get at least 3 shifts in a row on consecutive days. To do this, we use two goals. The first goal is to ensure no separate single shifts. The second is to ensure no separate series of 2 shifts.

Let W represent a work day and R represent a rest day. The following sets of equations enforces the soft constraint that avoids sequences of the type R W R.

$$1 - r_{ik-1} + r_{ik} - r_{ik+1} \geq 0 \qquad \forall i \in I,\ k \in \{1, ..., |D| - 1\} \tag{22a}$$

$$d^6_{ik} = max\{0, r_{ik-1} - r_{ik} + r_{ik+1} - 1\} \tag{22b}$$

$$f_6 = \sum_{i \in I} \sum_{k=1}^{|D|-1} d^6_{ik} \tag{22c}$$

The following sets of equations enforces the soft constraint that avoids sequences of the type R W W R.

$$r_{ik} + r_{ik+1} - r_{ik-1} - r_{ik+2} \geq -1 \qquad \forall i \in I,\ k \in \{1, ..., |D| - 2\} \tag{23a}$$

$$d^7_{ik} = max\{0, r_{ik-1} - r_{ik} - r_{ik+1} + r_{ik+2} - 1\} \tag{23b}$$

$$f_7 = \sum_{i \in I} \sum_{k=1}^{|D|-2} d^7_{ik} \tag{23c}$$

6 Methodology

6.1 Introduction

The purpose of the methodology presented here is to approximate the Pareto front. That is, the algorithm attempts to find Pareto-optimal solutions which make up this front. In fact, it finds points that are not strongly dominated by others. Because the algorithm gives no guarantee of finding all existing solutions, these points, that are not dominated within the set of solutions found by the metaheuristic, may not all be actual Pareto points. Although the points found in this thesis are approximate Pareto points, we will refer to them as Pareto points for simplicity. The following two definitions are relevant for the methodology presented here.

For any two solutions a and b we write $a \prec b$ if a strongly dominates b. This means that $f_t(a) \leq f_t(b) \, \forall t \in T$ and $\exists \, t : f_t(a) < f_t(b)$. We write $a \preceq b$ if a weakly dominates b. For this, we only require that $f_t(a) \leq f_t(b) \, \forall t \in T$. $f_t(x)$ denotes some objective value of x, in this case the one that corresponds to the tth soft constraint.

6.2 The Methods

We present a method based on Li et al. (2012). It is a combination of goal-programming and a metaheuristic approach for global search. Goal programming is essentially an extended MIP approach used to handle multiple objectives. The global search algorithm is called the falling tide algorithm and is based on an older approach called the Great Deluge algorithm (Dueck, 1993).

As the name suggests, the algorithm can be explained using the the analogy of a falling tide, where solutions are represented by shells that someone collects. As the tide falls the sea level lowers, but it rises periodically due to waves hitting the beach (Li et al., 2012). These waves may bring shells to the beach but may also take some shells. The shell collector stays as close to the sea as possible while these waves hit and dissipate. After some predetermined time, the collector stops and may come back later to try again as desired. Each try is a run in the algorithm.

In the algorithm, the sea level is denoted by B. In each run, the starting level has a random component indicating that the level differs depending on the moment at which the search is

started. As the tide falls, the water level B lowers. When a new wave hits the shore, the water level B rises temporarily. This happens at the end of each wave in the algorithm.
Shells (solutions) that are close to the sea have lower fitness values. The collector stays as close to the sea as possible so that, as the sea level lowers, they pick up only the solutions with lower fitness values.

To obtain an initial solution, a goal programming model is used. In goal programming, some constraints of the problem are simply treated as constraints, so they are hard constraints (must be met for a feasible solution). These are the ones required by Dutch legislation (*Arbeidstijdenwet*, 2020) as well as demands made by the relevant institution. Another set of constraints are called goals i.e. soft constraints. These are the health criteria in this thesis. In order to adhere to the latter, they are combined into an objective function. More specifically, the sum of their slack variables (deviation from the goal) is minimized. The total deviations are denoted by $f_1, ..., f_T$. The weights that each goal is given is given by p_i. So, the objective function becomes:

$$z = p_1 \cdot f_1 + ... + p_T \cdot f_T \tag{24}$$

This goal programming model is used to ensure that all hard constraints hold at this stage. For the creation of the initial solution, the objective function is simply set to 0 as we are looking for a feasible solution and are not yet interested in obtaining a "good" solution. This initial solution will be improved upon in the falling tide algorithm.

Goal programming has a second use in this algorithm. Namely to create the so-called ideal vector $v = (f_1(u_1), f_2(u_2), ..., f_T(u_T))$ where u_t is the solution providing the optimal objective value for the tth soft constraint. For goal t, $f_t(x)$ is a function of the deviation from the goal. Each element of the vector v is the ideal objective value w.r.t. goal t. This ideal objective value of a goal t^* is obtained by solving the problem as a MIP, where p_t is set to 0 for all t except t^*.

The ideal values described above are used in the fitness metric to assess the quality of candidate solutions. The exact metric used to assess fitness may differ in each run. It is selected randomly from a set of 3 metrics. These are Manhattan distance, Euclidean distance, and Chebyshev distance. These are known distance measures which can all be represented as $L_p(x) = \{\sum_{t=1}^{T} [w_t(f_t(x) - f_t(u_t))]^p\}^{1/p}$ where T is the number of soft constraints. The parameter p takes on the values 1, 2 and ∞ for the Manhattan, Euclidean and Chebyshev distances respectively. w_t is the weight of goal t, representing its importance when compared to other goals. In practice, these follow from the preferences of the user.

6.3 Falling Tide

The following is the algorithm adapted from Li et al. (2012).

6.3.1 Initialization

1. The number of runs is given by N_{run}, the number of waves is given by N_{wav}, and the number of levels is given by N_{lev}.

2. The goal programming model is used to obtain the initial solution x_0 as well as the ideal objective-value vector $v = (f_1(u_1), f_2(u_2), ..., f_T(u_T))$. For the former, the p_t values are chosen as follows: $p_t = 0$ for all $t \in \{1, ..., T\}$. For the latter, the model is run T times and we set $p_{t'} = 0$ for all $t' \in \{1, ..., T\}$ and $p_t = 1$ at the tth time.

Algorithm 1: Falling Tide

Result: R

$R = \{x_0\}$; ▷ Initialize the set of Pareto points

$k \leftarrow 0$;

for $i \in \{1, ..., N_{run}\}$ **do**

  Randomly express user's preferences on each objective (i.e. the weight vector $w = [w_1, ..., w_T]^T$), and randomly select a p metric from a uniform distribution metric where $p \in \{1, 2, \infty\}$. Each element of the weight vector is selected from unif{0,10}, the discrete uniform distribution;

  Set the current solution x to x_0, calculate the fitness of x as: $L_p(x) = \{\sum_{t=1}^{T}[w_t(f_t(x) - f_t(u_t))]^p\}^{\frac{1}{p}}$, and define the initial level as $B = \frac{L_p(x)}{\alpha}$, where the restart coefficient α is a random number between 0 and 1, selected from unif[0,1);

  for $j \in \{1, ..., N_{wav}\}$ **do**

    Set the regression rate $\Delta B = \frac{L_p(x)}{N_{lev}}$;

    for $k \in \{1, ..., N_{lev}\}$ **do**

      Construct a new solution $y \in N_k^r(x)$;

      while *y violates at least one of the hard constraints* **do**

        Construct a new solution $y \in N_k^r(x)$;

      end

      Calculate the fitness of y as $L_p(y) = \{\sum_{t=1}^{T}[w_t(f_t(y) - f_t(u_t))]^p\}^{\frac{1}{p}}$;

      if $L_p(y) \leq L_p(x)$ *or* $L_p(y) \leq B$ **then**

        Accept y and replace x with y;

        if $L_p(y) < L_p(x)$ **then**

          $k \leftarrow 0$; ▷ Return to smallest neighbourhood when a good solution is found

        end

        if $\nexists s \in R$ *s.t.* $s \prec y$ **then**

          $R \leftarrow R \cup \{y\}$; ▷ Add y to R if no existing solution in R dominates it

        end

        if $\exists t \in R$ *s.t.* $y \prec t$ **then**

          $R \leftarrow R \setminus \{t\}$; ▷ Remove solutions from R that are dominated by y

        end

      else

        $k \leftarrow 1 + (k \bmod K)$; ▷ Go to a larger neighbourhood

      end

      Lower level $B = B - \Delta B$;

    end

    Reset $B = (\beta + 1)f(y)$, where the water-rerising rate β is a random number generated between 0 and 1, selected from unif[0,1);

  end

end

Constructing new solutions New solutions are constructed (lines 9 and 12) using a neighbourhood search defined by vertical moves or swaps between two employees. As an extension to the original algorithm from Li et al. (2012), we use a variable neighbourhood search. If a search in a particular neighbourhood is not successful we switch to another (larger) neighbourhood temporarily as on line 26. The possible neighbourhoods are moving a shift to another employee, swapping a pair of shifts between employees, and swapping two or more shifts with two or more other shifts. The neighbourhoods are numbered 0 through K, where $K = |R|$ i.e. the number of days in the planning period. So, elements of $N_0(x)$ are those solutions that can be obtained by moving a job from one employee to another. Elements of $N_5(x)$ are those that can be obtained from swapping 5 jobs between 2 employees. $N_k^r(x)$ with a superscript r indicates that a random element from the neighbourhood is chosen. In this case, the employees whose jobs are moved or swapped are chosen from the discrete uniform distribution. Also, the days to be swapped are chosen randomly from the planning horizon, specifically from unif$\{1,|R|\}$.

We compare this falling tide approach with VNS to the approach from Li et al. (2012). In the latter case, the number of days to be swapped is chosen randomly (from the uniform distribution). The difference between the two methods is mainly that the VNS method systematically switches to a larger neighbourhood if it is not successful in a smaller one. Also, it does so only when it is not successful in a smaller neighbourhood. So, as long as acceptable solutions are found in a smaller neighbourhood, the search will stay in that neighbourhood.

Runs, Waves, and Levels A run (starting on line 4) consists of N_{wav} waves. Each run has its own user preferences, that is, the weight vector is drawn at the start of the run. Also, each run begins with the same initial solution x_0. The initial level is different in each run according to the restart coefficient α. This parameter simply introduces some randomness to the starting level $B = \frac{L_p(x)}{\alpha}$.

Each wave, which starts on line 7, has a single regression rate. This is the rate at which the level decreases in each level iteration on line 28. At the end of each wave, the level is reset (line 30) so that the level is not exclusively decreasing within each run. So, in each wave the level starts out relatively high and decreases as the wave progresses, making the probability of accepting the new solution lower. A new solution is always accepted if its fitness is better than that of the current solution.

In each level, one new solution is found. If this solution violates any hard constraints, a new solution is created until a solution is found that doesn't. The acceptance step comes after this. Only if the solution is accepted according to the criteria described, the set of potentially non-dominated solutions R is updated. This does not necessarily mean that y is added to R. This is done only if y is not dominated by any other solution in R.

6.4 Variation of Parameters

The resulting set of solutions R is an approximation of the Pareto front. We ensure this by adding exclusively non-dominated (Pareto optimal) solutions to the set and removing existing solutions as soon as they become dominated. A solution that is not dominated by any other possible solution is called non-dominated or Pareto optimal.

The set R is an approximation because there is no guarantee of finding all possible solutions. This follows directly from the fact that falling tide is a metaheuristic. This implies that some solutions we find to be non-dominated may in fact not be Pareto optimal because a solution that dominates it could exist without being found by falling tide.

We do attempt to find as many non-dominated solutions as possible. One important vehicle for this is the variation of p by which a metric for evaluating fitness is chosen. The first metric is given by $p = 1$ resulting in the formula $L_1(x) = \sum_{t=1}^{T} w_t(f_t(x) - f_t(u_t))$. In section 6.2, we mentioned that this is called the Manhattan distance. In general, the Manhattan distance uses absolute values, which would result in $L_1(x) = \sum_{t=1}^{T} |w_t(f_t(x) - f_t(u_t))|$. However, this is not necessary because we know that the ideal function value is the lowest possible value. Therefore, $w_t(f_t(x) - f_t(u_t))$ is always non-negative. Here, the effect of the deviation on the fitness level is proportional to the size of the deviation. This is completely contrary to the case of $p = \infty$, where only the largest of the T deviations has an effect on the fitness value. Namely, $L_\infty(x) = \max_{t=1,\dots,T} w_t(f_t(x) - f_t(u_t))$.

We will illustrate the implications of this with the help of figures 1 and 2, where we consider the case of two goals for simplicity. Using $L_1(x)$ we can never find a Pareto point on the Pareto front between points A and B. This is because the iso-value relationship (points that have the same fitness value) is linear and A and B are closer to the origin, meaning they have a lower (better) fitness value. This situation occurs when the relationship between the goals is concave. The iso-value relationship is represented by the grey line. All points on this line have the same fitness value. It is a line due to the fact that the L_1 norm is linear.

No matter what values w_1 and w_2 are given, the points on the concave arc cannot be reached with this metric. On the other side of the spectrum $L_\infty(x)$ considers only the goal with the highest product of weight and deviation. So, by varying w_t and w'_t, point C may, for instance, be reached. This illustrates that points on a concave curve (relationship between two objectives) can be reached by the L_∞ norm due to the fact that there is essentially no compromise among the objectives. Only the largest weight-deviation product is considered. This is why it is useful to use both the L_1 and L_∞ norm in falling tide. The L_2 (Euclidean) norm is also used. This norm provides a fairer compromise among the objectives than both L_1 and L_∞, as higher deviations from the ideal value lead to a higher penalty to the fitness level. This is what leads us to use these 3 norms, altering p from run to run.

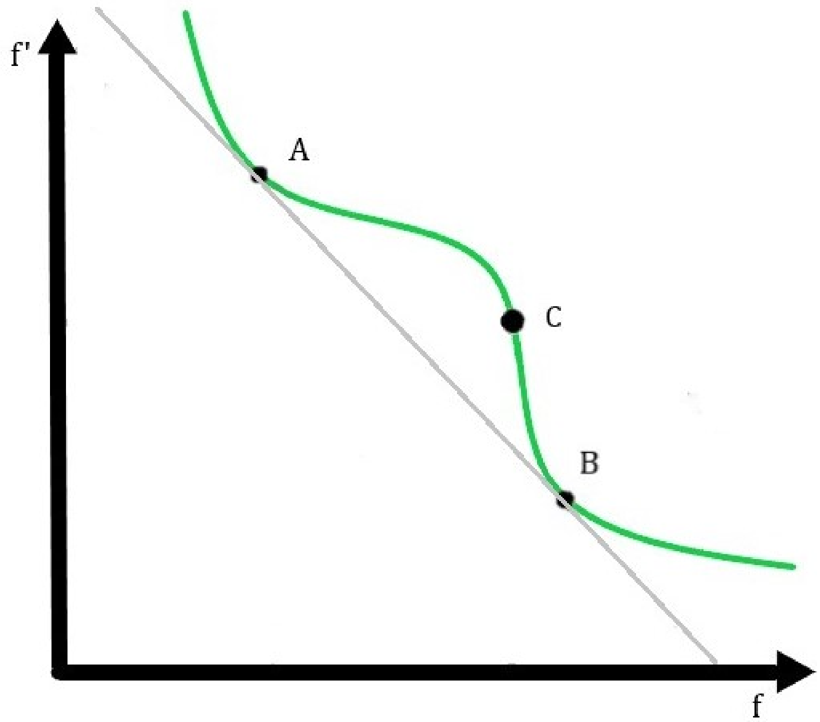

Figure 1: L_1 metric approximation of the Pareto front

Inspired by an image from Li et al. (2012)

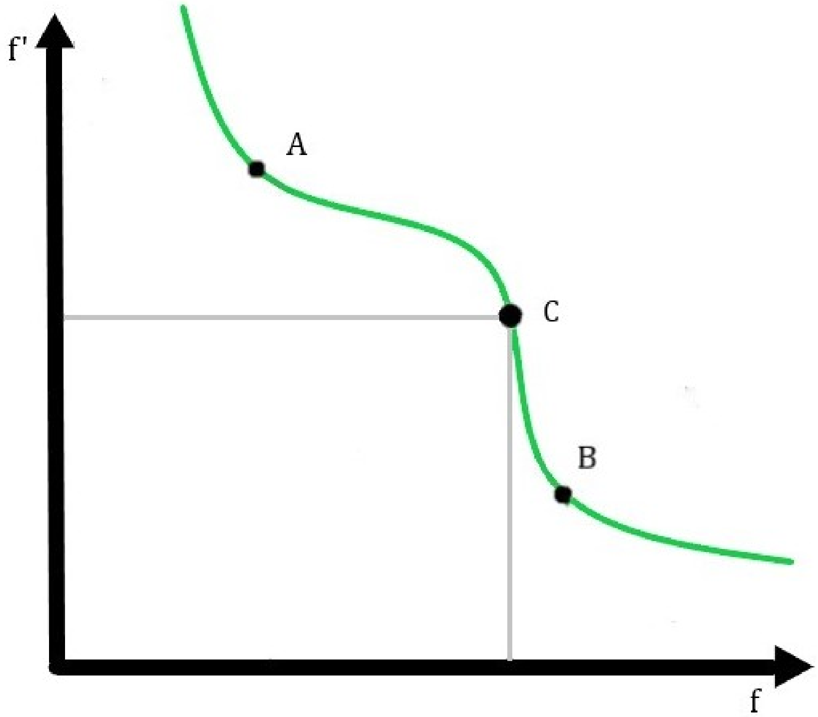

Figure 2: L_∞ metric approximation of the Pareto front

Inspired by an image from Li et al. (2012)

6.5 Intensification and Diversification

In order to find the best possible solutions using a metaheuristic, intensification and diversification components are essential. Intensification helps to find local optima while diversification attempts to let the algorithm escape from or avoid local optima so that other local optima can be found. In the falling tide algorithm, intensification is achieved by finding solutions with a fitness value lower the value of B, given that B has a sufficiently low value. In the earlier runs of each wave, while the water level is still high, this in fact acts as a diversification mechanism because many solutions are accepted. The new wave starts its search using the solution x from the previous wave. The search is thus intensified from wave to wave although the starting level is perturbed slightly each time due the random variable β. Diversification occurs in one way between each run as the metric may change and the initial level B is reset with a large degree of randomness due to the new metric and the random variable α. Another mechanism for diversification is variation of the neighbourhood that is searched.

6.6 Simulated Annealing

As a second method, we implement a multi-objective version of the simulated annealing algorithm. We have created this algorithm by adding some elements of the falling tide algorithm by Li et al. (2012) to the well-known simulated annealing algorithm. It is given by algorithm 2. The initial solution x_0 is given by the goal programming method just as in falling tide. The metric for determining fitness $L_p(x)$ is also the same as in the original method. A number of runs N_{run} is determined beforehand. The neighbourhood $N(x)$ is defined by swapping a random number n jobs.

Algorithm 2: Simulated Annealing

Result: R

$R = \{x_0\}$; ▷ `Initialize set of Pareto points`

$x \leftarrow x_0$;

$\tau \leftarrow 100$; ▷ `Initialize temperature`

for $i \in \{1, ..., N_{run}\}$ **do**

  Randomly express user's preferences on each objective (i.e. the weight vector $w = [w_1, ..., w_T]^T$), and randomly select an L_p metric where $p \in \{1, 2, \infty\}$;

  Construct a new solution $x' \in N(x)$;

  while x' *violates at least one of the hard constraints* **do**

    Construct a new solution $x' \in N(x)$

  end

  Calculate the fitness of x' as $L_p(x')$;

  Let $\Delta(x', x) = L_p(x') - L_p(x)$;

  if $\Delta(x', x) < 0$ **then**

    $x \leftarrow x'$;

  else

    Select random $r \in [0, 1)$

    if $r < e^{-\frac{\Delta(x', x)}{\tau}}$ **then**

      $x \leftarrow x'$

    end

  end

  if $\nexists s \in R$ *s.t.* $s \prec x$ **then**

    $R \leftarrow R \cup \{x\}$; ▷ `Add` x' `to` R `if no solution in` R `dominates it`

  end

  if $\exists t \in R$ *s.t.* $x \prec t$ **then**

    $R \leftarrow R \setminus \{t\}$; ▷ `Remove solutions from` R `that are dominated by` x'

  end

  Select random $\alpha \in (0, 1)$;

  $\tau \leftarrow \alpha\tau$;

end

A simulated annealing algorithm uses a so-called temperature, denoted by τ in this case. The temperature is lowered from run to run. The purpose of this it to lower the chance of accepting a solution that does not have an improved fitness value. As can be seen on line 12, a solution that provides a better fitness value is always accepted. However, there is also a chance of accepting a solution when it does not have a better fitness value than the previous solution. This can be seen on line 16. On this line we also see that the acceptance of such a solution depends on a random component as well as the temperature τ. In earlier runs, we attempt to find a diverse number of solutions by accepting relatively many solutions that do not have an improved fitness value. Because the temperature (τ) is lowered in each run, we accept less and less solutions that do not have a better fitness value than the previous best solution. This allows us to find better quality solutions in later runs.

6.7 Comparing the Metaheuristics

The comparison of the metaheurstics will be done on a multi-objective level using the two set coverage metric, also known as the C-metric. It is a well-known metric, described by Knowles & Corne (2002) among others. This metric uses the fraction of solutions that are weakly dominated by another set. When comparing two sets, if 60% of solutions of method B are weakly dominated by one or more solutions of method A, the value of this metric is 0.6.
The metric is calculated as follows

$$C(A, B) = \frac{|\{b \in B | \exists a \in A : a \preceq b\}|}{|B|} \tag{25}$$

If we choose falling tide as method A and simulated annealing as method B, a high value of the C metric would indicate that relatively good solutions are found by algorithm A. This metric those not allow for objectively comparing the approximate Pareto front to the exact Pareto front. Metrics that achieve this do exist, but cannot be used here due to the NP-hard nature of the Nurse Scheduling Problem. The complexity of the problem implies that the exact Pareto front cannot be determined in polynomial time.

6.8 Non-supported Pareto Points

As mentioned in section 1, the methods described in this section are able to find Pareto points that cannot be found using a single-objective weighted sum approach. Specifically, the Pareto optimal solutions found by a weighted sum approach are called *supported* solutions (Czyzżak

& Jaszkiewicz, 1998). The implication of this is that some Pareto points can not be found in the case of a non-convex Pareto curve. The points that cannot be found are non-supported Pareto points. This has been mentioned in research by, among others Ehrgott (2008) and Emmerich & Deutz (2018). When the Pareto curve is non-convex, the same situation occurs that is illustrated by figure 1 where we see that only points A and B are found. When using a weighted objective approach, no combination of weights is able to find solutions on the arc between A en B. As mentioned in section 6.4, this is due to the linear relationship among the goals in the objective function. Using the variation of the p metric, the approach in this thesis is able to reach non-supported Pareto points.

7 Results

7.1 Introduction

In this section, we perform a number of experiments to investigate the performance of the methods presented in section 6.

In section 7.2, we describe the data instances used. In section 7.3, we present the results of the goal programming algorithm. In section 7.4, we vary the use of the fitness metrics. We do this to determine the best setting of the p metric for each instance. In section 7.5 we present the results of experiments where we compare the three metaheuristic methods. Our purpose is to find the metaheuristic that is best able to approximate the Pareto front.

7.2 Data

We use 5 data sets. We describe their characteristics in table 1. The data has been obtained by ORTEC from the institutions "Radboud UMC" (RUMC) and "Groene Hart Ziekenhuis" (GHZ) in the Netherlands. The departments are described below. Note that instances 1 and 5 are the same department with different planning horizons. In the case of instance 1, 2, 4, and 5, we added some extra night shifts to the data to make the problem instances more challenging for the algorithm. Historical schedules of these departments reveal that the number of shifts scheduled is often higher than that of the official shift requirements. So, the instances considered here are realistic. In the case of instance 2, the hard constraint pertaining to skills (given by equation (5)) is not used because skills have not been correctly implemented in this data set.

1. Nursing department for Neurology at RUMC
2. Nursing department for Haematology at RUMC
3. Lung Disease department at GHZ
4. Emergency department at GHZ
5. Nursing department for Neurology at RUMC

Instance	Planning Horizon (in weeks)	Number of Employees	Number of Shifts
1	1	41	156
2	1	60	197
3	1	33	67
4	1	66	109
5	4	41	692

Table 1: Description of the Data Instances

7.3 Goal Programming Results

Table 2 shows the goal values found by the goal programming approach. The initial values are those used by the metaheuristics to start with. They are found by solving the MIP without an objective function. The ideal values are the results of solving the MIP with one objective at a time. These ideal values are lower bounds on the relevant objective value. We also attempted solving the MIP with a weighted objective function containing all the goals. No result was found due to the MIP gap still being 99% after several hours of running. So, it does not seem realistic to solve the problem using a MIP. This substantiates the use of a metaheuristic to solve this problem.

Goal	0	1	2	3	4	5	6	7
Initial Values	7	2	1	0	39	0	6	10
Ideal Values	0	0	0	0	0	0	0	0

Table 2: Initial and Ideal Goal Values Instance 1

Goal	0	1	2	3	4	5	6	7
Initial Values	12	6	2	0	32	0	38	9
Ideal Values	0	0	0	0	0	0	0	0

Table 3: Initial and Ideal Goal Values Instance 2

Goal	0	1	2	3	4	5	6	7
Initial Values	0	0	0	0	7	0	23	9
Ideal Values	0	0	0	0	0	0	0	0

Table 4: Initial and Ideal Goal Values Instance 3

Goal	0	1	2	3	4	5	6	7
Initial Values	0	0	0	0	13	0	47	6
Ideal Values	0	0	0	0	0	0	0	0

Table 5: Initial and Ideal Goal Values Instance 4

Goal	0	1	2	3	4	5	6	7
Initial Values	96	68	88	80	168	10	77	53
Ideal Values	0	0	0	0	0	0	0	0

Table 6: Initial and Ideal Goal Values Instance 5

7.4 Varying the Fitness Metric

In this section, we describe all the results of the fitness metric experiment. The goal of this experiment is to determine the best parameter settings of the p metric in the metaheuristic. We choose among using one of the 3 metrics ($p = 1$, $p = 2$ or $p = \infty$) separately and using all 3 metrics at once. The results presented in this section pertain to instance 1. Tables with C-metric values of this experiment for the remaining data sets can be found in appendix B. The number of runs (50) used is the same for all instances. Instance 5 is not included in this experiment due to the exceedingly long runtime. This long runtime is due to its larger size.

We use the falling tide algorithm with VNS for all runs in this section. We then decide on the best choice of p metric(s) using data about the Pareto points as well as the C-metric for comparing the 4 settings.

A time limit of 30 seconds per run is implemented. Also, when no new non-dominated solutions are found for 3 levels in a row, the algorithm immediately moves on to the next run. For each random component in the algorithm, the same seed is used for all compared methods. So, the stream of random numbers is the same for the methods. This is done to mitigate differences in performance due to chance. The parameter settings described here are used in all experiments conducted in this thesis.

Table 7 describes all the runs performed for this experiment. Here, we see some discrepancy in the runtimes despite an equal number of runs. We see that in the case of using $p = \infty$ metrics, the runtime is longest and the number of Pareto points found is highest. As mentioned, a run is cut short if no new Pareto point is found for 3 levels. In the case of using $p = \infty$ this happens less often, making the runtimes longer in total.

p-Metric	1	2	∞	All
Number of Pareto Points	14	12	24	22
Number of Runs	50	50	50	50
Runtime/seconds	798	717	879	853

Table 7: Descriptions of the Runs of the Fitness Metric Experiment Instance 1

Table 8 shows the C-metric values of each pair of p metrics. Values in the table should be read as representing the value $C(A, B)$ where set A is given by the row of the value and set B is given by the column it is in. The set of Pareto points R_{all} found by the algorithm using all metrics weakly dominates 17% of points found by the run using only $p = 2$. Only 8% of points in the R_∞ are dominated by this set. Also, the algoirthm with only $p = \infty$ dominates 43% of points found by $p = 1$. Using all metrics leads to the worst performance in the case of this instance. On average, 13% of the points in R_∞ are dominated by another set. Also, the points in the set dominate 51% of points in another set on average. It is clear that the algorithm using $p = \infty$ performs best. p metrics 2 and "all" dominate only 8% of points in R_∞. This means that the rest of the points are not found by the single metrics at all. This follows from the fact that the C-metric is calculated using weak domination.

Metric	1	2	∞	all	Average
1	-	0.25	0.21	0.59	0.35
2	0.14	-	0.08	0.23	0.15
∞	0.43	0.42	-	0.68	0.51
all	0.14	0.17	0.08	-	0.13
Average	0.24	0.28	0.13	0.50	

Table 8: C-Metric Values for Each Metric Pair

Tables 9 and 10 show the minimum, maximum, mean, and variance of the goal values found by each metric. Although the results show that it is best to use $p = \infty$, the other metrics still find many points that are not found when using only $p = \infty$. This is especially the case for $p = 1$. This explains why this metric outperforms using $p = \infty$ on, for example, goal 2.

	p=1				p=2			
Goal	Min	Max	Mean	Variance	Min	Max	Mean	Variance
0	7	15	11.14	4.90	7	18	13.08	13.54
1	2	5	3.71	0.68	2	13	7.33	9.33
2	1	3	1.50	0.42	1	6	3.67	2.42
3	0	0	0.00	0.00	0	0	0.00	0.00
4	31	39	36.29	4.22	31	39	35.75	6.39
5	0	1	0.93	0.07	0	1	0.92	0.08
6	6	13	9.93	4.07	5	13	7.92	6.99
7	6	11	8.29	2.07	6	12	9.08	4.08

Table 9: Pareto Point Descriptive Statistics for Metrics $p = 1$ and $p = 2$

	$p = \infty$				All metrics			
Goal	Min	Max	Mean	Variance	Min	Max	Mean	Variance
0	7	18	13.42	12.51	7	17	14.18	7.11
1	1	14	8.21	16.17	2	10	8.18	5.49
2	0	8	4.50	7.04	1	7	5.41	2.73
3	0	0	0.00	0.00	0	0	0.00	0.00
4	28	39	32.13	9.24	32	39	34.95	3.66
5	0	2	1.00	0.09	0	1	0.82	0.16
6	6	14	10.54	6.26	6	19	13.95	14.33
7	6	11	8.25	1.85	3	11	7.09	6.85

Table 10: Pareto Point Descriptive Statistics for Metrics $p = \infty$ and all p values

The result that $p = \infty$ leads to the best Pareto set also holds for most of the other instances. The exception is instance 2. In this case, we conclude that using all metrics is the best option because only 10% of the Pareto points in the set produced by this p metric is dominated by the other options on average. This is the lowest out of all four settings.

Figures 3 through 6 show one graph for each metric configuration using the data from instance 1. Due to the multi-objective nature of the problem, it is not possible to plot the relationship among all goals. So, we provide two-dimensional graphs of the goals with the highest variance because these are the most interesting cases. At the two-dimensional level, some points are dominated by others. This does not imply that they are dominated at the 8-dimensional level because this is of course depends on the values of *all* the goals. All the illustrated points are non-dominated within the set found by the metaheuristic.

In the case of $p = \infty$, goal 0 and 1 have the highest variance. We see that there seems to be a positive relationship between these goals. This is entirely plausible because these two goals do not directly contradict each other. As a reminder, soft constraint 0 aims to allow no more than 2 night shifts in a row while the purpose of soft constraint 1 is to have one rest day after each series of 4 shifts. Both these penalties are lower in the case of less consecutive shifts. This explains the shape of the scatter plot. The same argument can be made for figure 4. We see that the same goals are plotted here and the plot shows that the relationship between the Pareto points is similar to that in 5. We also see more points in 5 because the experiment with this metric found more Pareto points. In figure 3, we present the values of goals 0 and 4 found by $p = 1$. Firstly, we see that this experiment has not produced many Pareto points. Also, we see that, in general, points with a higher penalty on goal 4 have a lower pentaly on goal 0. The intention of goal 4 is to have the employee start at the same time in case they work consecutive shifts. In the case of night shifts, this conflicts with goal 0. For example, 3 night shifts in a row would lead to a penalty with respect to goal 0 but no penalty on goal 4. In figure 6, we again plot a different pair of goals. The plot suggests a positive relationship between goals 0 and 6. Goal 6 is to avoid single shifts. The positive relationship between these goals does not follow directly from their definitions.

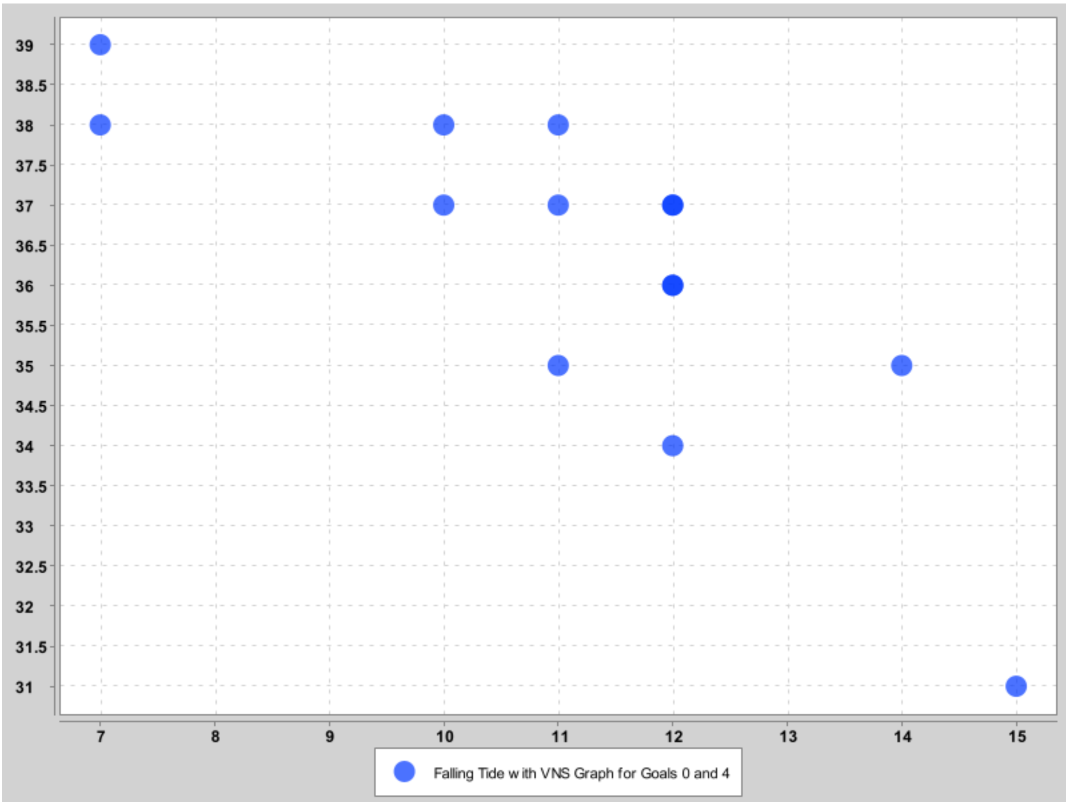

Figure 3: Scatter plot of Goal 4 against Goal 0 with Metric $p = 1$

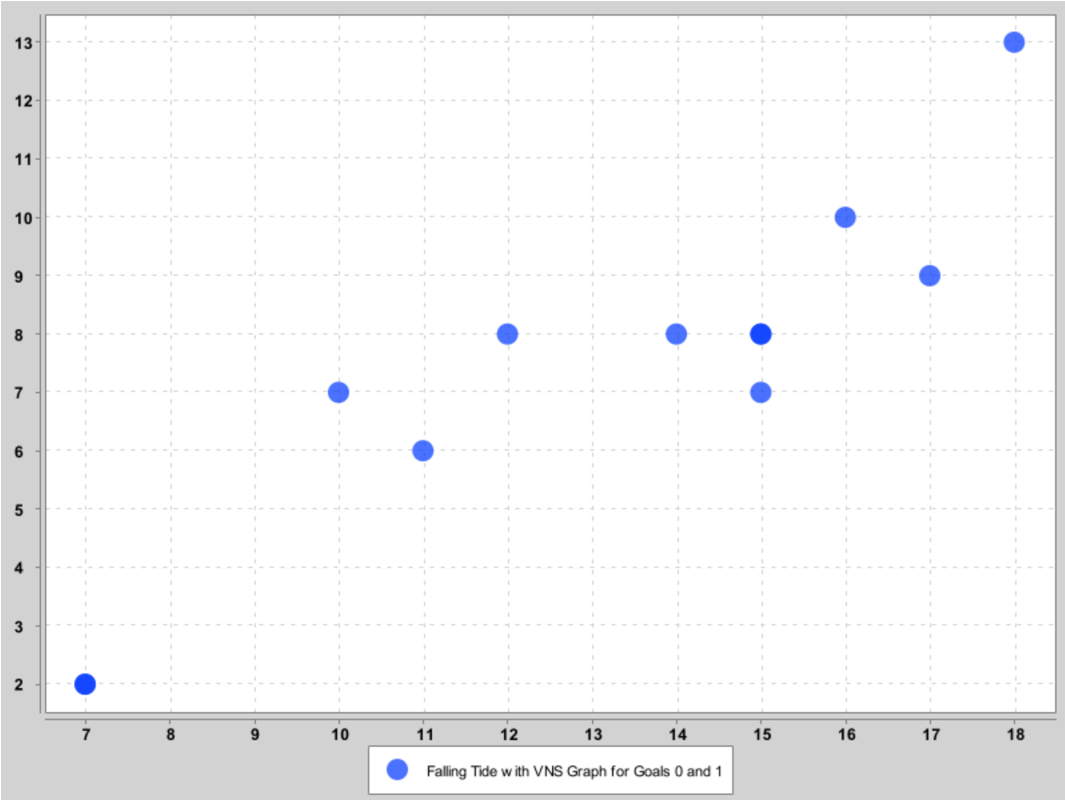

Figure 4: Scatter plot of Goal 1 against Goal 0 with Metric $p = 2$

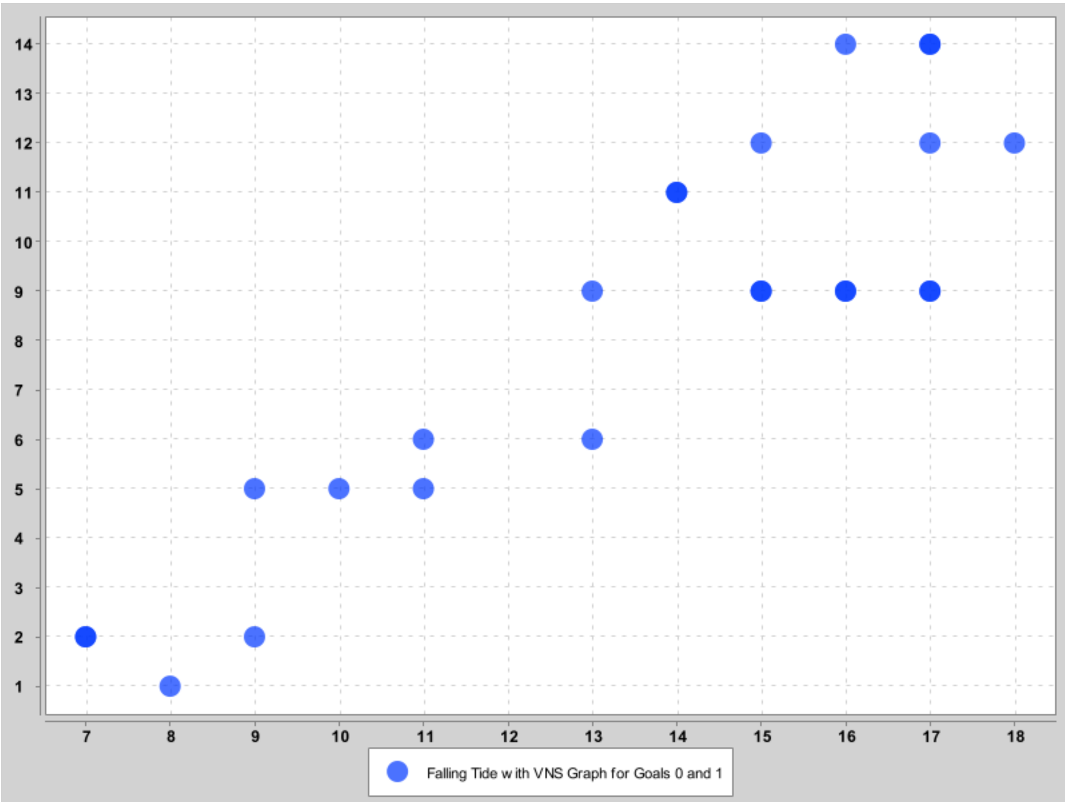

Figure 5: Scatter plot of Goal 1 against Goal 0 with Metric $p = \infty$

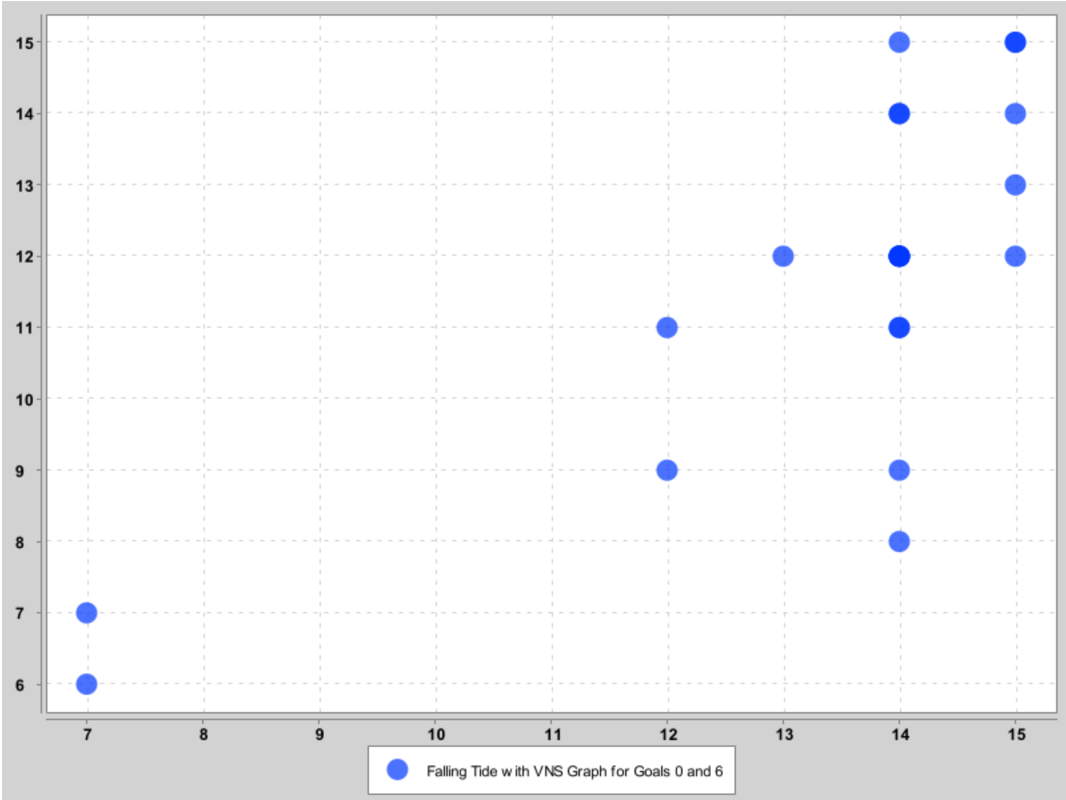

Figure 6: Scatter plot of Goal 6 against Goal 0 all metrics

7.5 Comparing Methods

In the following section, we present numerical results for 3 methods: falling tide with VNS, falling tide without VNS, and simulated annealing. This is done to investigate the benefit of the VNS component and to compare the falling tide algorithm to an alternative metaheuristic. For each instance, we use the setting of the p metric that was shown to perform best in the fitness metric experiment. For instance 1, 3, and 4 this is $p = \infty$. For instance 4, we also use $p = \infty$ because the data pertains to the same department as instance 1. For instance 2 we use all p metrics.

In table 11, we describe the runs for instance 1. Similar tables for all other instances can be found in appendix B. The number of runs is the same for each instance except for instance 5. In the case of instance 5, we only compare falling tide with VNS with the simulated annealing method. Also, we use a lower number of runs, namely 20 and 40 runs respectively. This is done because of the extremely long runtime of this data set.

Method	Falling Tide with VNS	Falling Tide without VNS	Simulated Annealing
Number of Pareto Points	41	21	23
Number of Runs	100	100	250
Runtime/seconds	1960	2027	1505

Table 11: Descriptions of the Runs of the Algorithm Comparison Experiment Instance 1

First, we discuss the differences between the two falling tide methods. Overall, the method with VNS outperforms that without VNS. Less near-Pareto optimal points are found by the method without VNS in the case of instance 1. Tables 12, 14, 15, 16, and 17 show that $C(R_{withVNS}, R_{withoutVNS})$ is substantially higher than $C(R_{withoutVNS}, R_{withVNS})$ for all instances. For example, in the case of instance 1, the set of Pareto points found by falling tide without VNS dominates only 2% of points found by falling tide with VNS, while $C(R_{withVNS}, R_{withoutVNS})$= 0.19. So, the method with VNS weakly dominates almost 20% of the points found by falling tide without VNS. Interestingly, the method without VNS does beat falling tide with VNS on most of the goals when considering the average values for instance 1 (see table 13). This suggests that the 80 % of points that are not dominated by falling tide with VNS perform well on goals 0,1,2,6, and 7.

VNS is used in general to stay in small, easily explored neighbourhoods when possible and to switch to larger neighbourhoods when necessary to escape local optima. In general, a systematic change of neighbourhoods is preferable. A smaller change in the schedule is less likely to lead to infeasibilities, saving runtime in general. So, staying in small neighbourhoods, which leads to making smaller changes, is preferred when possible. This mechanism is likely what leads to a better performance of falling tide with the VNS component as compared to the one without. However, in the method without VNS, the size of the neighbourhood also changes due to the fact that the number of days switched is chosen randomly. This already helps to escape local optima. This explains why this algorithm still performs relatively well and why the VNS component may not lead to a better performance in some other instances.

Next, we consider the simulated annealing metaheuristic. In all but one instance, falling tide with VNS finds as many or more Pareto points than simulated annealing (tables 11, 18, 19, 20, and 21). The exception is instance 5. This could be due to the low number of runs in this case. We do see that the single point found by falling tide with VNS in this instance is not dominated by the points found by simulated annealing (table 17). Table 12 shows that, on average, 20% of the points found by this algorithm are dominated by the others for instance 1. In the case of instance 4, this number is 59% (table 16). Conversely, in instance 1, it dominates only 5% of values of the Pareto set of falling tide with VNS. For instances 2, 4, and 5 this number is 0, as can be seen in table 14, 16, and 17. So, the falling tide algorithm with VNS has produced better results than the simulated annealing algorithm in general.

Method	Falling Tide with VNS	Falling Tide without VNS	Simulated Annealing	Average
Falling Tide with VNS	-	0.19	0.13	0.16
Falling Tide without VNS	0.02	-	0.26	0.14
Simulated Annealing	0.05	0.24	-	0.14
Average	0.04	0.21	0.20	

Table 12: C-Metric values of the Algorithm Comparison Experiment Instance 1

Method	Falling Tide with VNS				Falling Tide without VNS				Simulated Annealing			
Goal	Min	Max	Mean	Variance	Min	Max	Mean	Variance	Min	Max	Mean	Variance
0	7	18	14.51	9.81	7	20	12.19	7.86	7	17	13.91	9.54
1	1	14	7.63	11.14	2	13	5.52	4.46	2	13	7.96	8.95
2	0	8	4.05	5.10	0	5	2.76	1.39	1	6	4.09	1.45
3	0	0	0.00	0.00	0	0	0.00	0.00	0	0	0.00	0.00
4	23	39	29.20	18.81	29	39	32.76	7.29	26	39	30.39	14.61
5	0	2	0.83	0.20	0	3	1.95	0.75	0	3	2.09	0.90
6	6	21	13.24	15.29	6	17	11.14	5.33	6	17	12.83	7.97
7	6	11	8.15	1.53	5	10	7.00	1.80	4	10	6.13	2.66

Table 13: Pareto Point Descriptive Statistics for each Method for Instance 1

Method	Falling Tide with VNS	Falling Tide without VNS	Simulated Annealing	Average
Falling Tide with VNS	-	0.28	0.21	0.24
Falling Tide without VNS	0.10	-	0.13	0.11
Simulated Annealing	0.00	0.34	-	0.17
Average	0.05	0.31	0.17	

Table 14: C-Metric Values Algorithm Comparison Experiment Instance 2

Method	Falling Tide with VNS	Falling Tide without VNS	Simulated Annealing	Average
Falling Tide with VNS	-	0.67	0.33	0.50
Falling Tide without VNS	0.00	-	0.00	0.00
Simulated Annealing	0.63	0.11	-	0.37
Average	0.31	0.39	0.17	

Table 15: C-Metric Values Algorithm Comparison Experiment Instance 3

Method	Falling Tide with VNS	Falling Tide without VNS	Simulated Annealing	Average
Falling Tide with VNS	-	0.86	0.64	0.75
Falling Tide without VNS	0.27	-	0.55	0.41
Simulated Annealing	0.00	0.14	-	0.07
Average	0.14	0.50	0.59	

Table 16: C-Metric Values Algorithm Comparison Experiment Instance 4

Method	Falling Tide with VNS	Simulated Annealing	Average
Falling Tide with VNS	-	0.14	0.14
Simulated Annealing	0.00	-	0.00
Average	0.00	0.14	

Table 17: C-Metric Values Algorithm Comparison Experiment Instance 5

Figures 7 through 9 show scatter plots of the 2 goals with the highest variance for each method using the data from instance 1. Plots for instance 2-5 can be found in appendix C. There is no scatter plot for the falling tide with VNS approach using instance 5 because only 1 point was found. Most of the plots have parts that suggest a concave relationship between the given goals. This implies a situation as described using figures 1 and 2 in section 6.4. So, using $p = \infty$ allows the algorithm to find points that are unreachable for the other metrics. This is consistent with the fact that the algorithm using $p = \infty$ performs relatively well when compared to the other metrics, as explained in 7.4. However, this conclusion is not definitive because we should keep in mind that the graphs only show the relationship for a limited number of goals.

Although there is no direct conflict between goals 4 and 6, figure 8 shows a negative relationship. A possible explanation is that there may not be many consecutive shifts in general. In that case, the value of goal 4 is low because less consecutive shifts means that there are by definition less consecutive shifts that do not start at the same time. At the same time a lower number of consecutive shifts makes it more likely that there are more single shifts, which causes a penalty on goal 6. The relationship between 0 and 4 is negative just as we saw in section 7.4. The same explanation given in that section applies here.

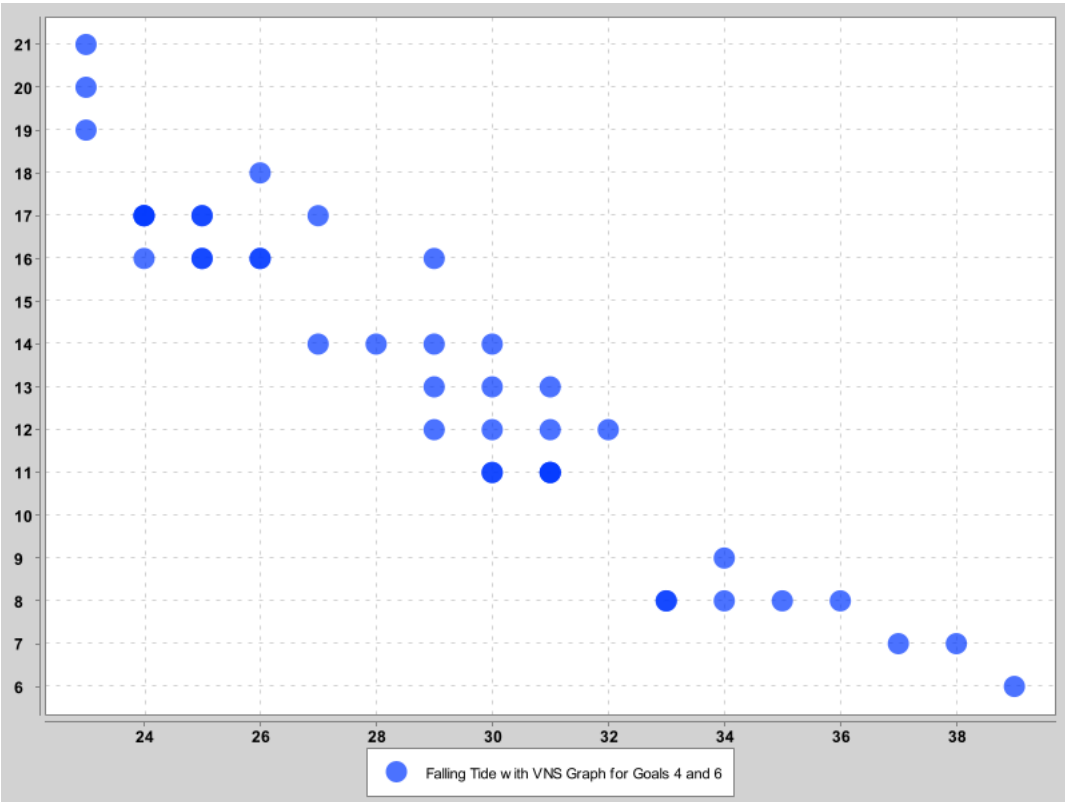

Figure 7: Scatter Plot of Goal 6 against Goal 4 Falling Tide with VNS Instance 1

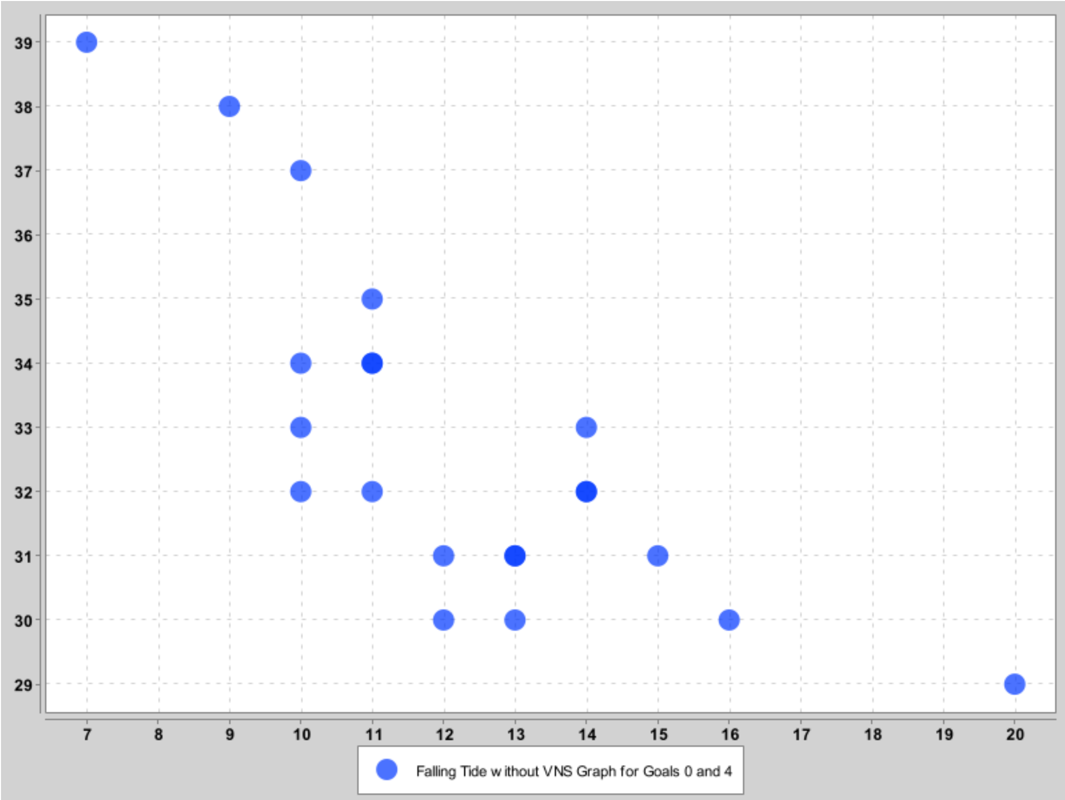

Figure 8: Scatter Plot of Goal 4 against Goal 0 Falling Tide without VNS Instance 1

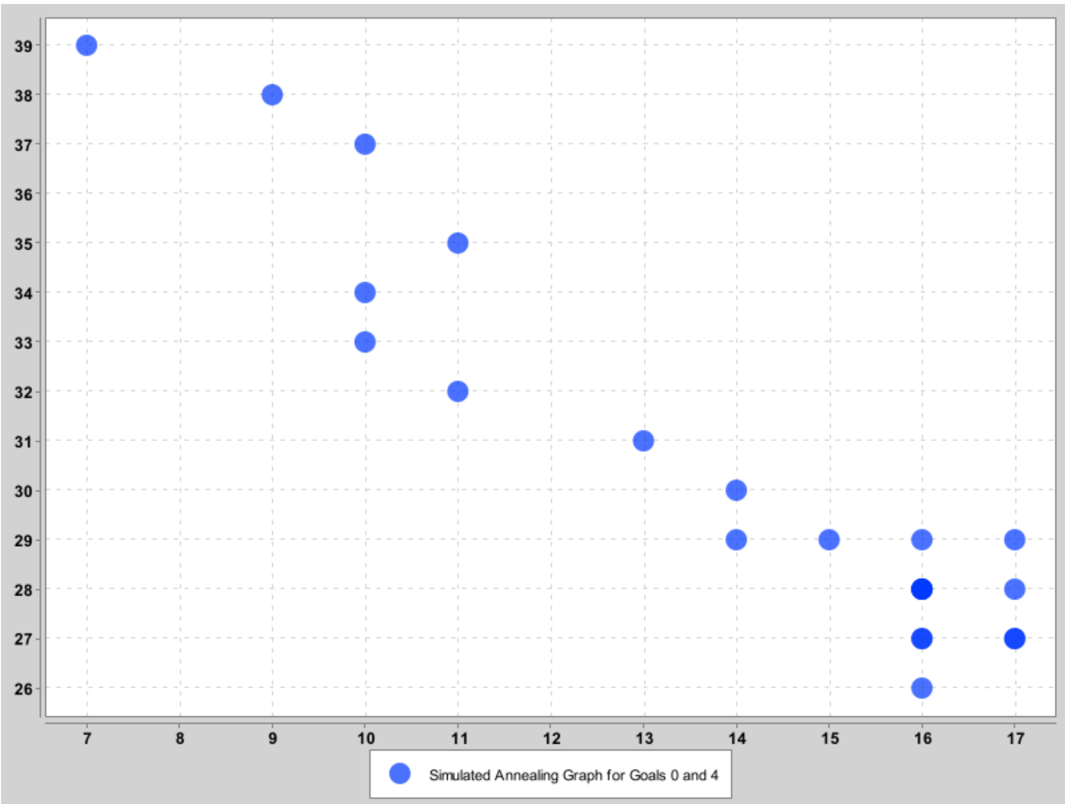

Figure 9: Scatter Plot of Goal 4 against Goal 0 Simulated Annealing Instance 1

8 Discussion and Conclusion

In this thesis, we have developed an approach for solving the multi-objective Nurse Scheduling Problem (NSP). The NSP is the problem of assigning shifts to nurses under a given set of constraints. It is an NP-hard problem. This research was conducted at ORTEC, which has developed a product that provides solutions to the NSP with a single weighted sum objective function. The approach in this thesis is able to provide healthcare institutions with several flexible schedules. Also, the objectives used in the multi-objective approach of this thesis are all aimed at making ergonomic schedules. We have used 8 objectives based on literature in the field of ergonomic workforce scheduling. To tackle the multi-objective nature of the problem considered in this thesis, we have incorporated metaheuristics in our approach. Also, we have used a goal programming approach to create ideal and initial solutions. Goal programming is a mixed-integer program method with soft constraints in the objective value. As mentioned, the purpose of these soft constraints is to make ergonomic schedules. The ideal solutions created using goal programming provide are the best possible solutions with respect to each single objective. The initial solution is a feasible solution obtained by solving the MIP with no objective function. This initial solution is the starting solution used by the metaheuristics. These metaheuristics use a neighbourhood search to create new and better solutions. To assess the fitness of these new solutions, 3 different fitness metrics (denoted by p) are used. This approach, where we vary the fitness metric, allows us to find non-supported Pareto points. These are points on concave parts of the Pareto curve. These points are out of reach for any approach using a single weighted sum objective function. The output of the metaheuristic is a set of Pareto points, each representing a balanced schedule in terms of the 8 goals.

We have implemented two types of metaheuristics. The first is the Falling Tide algorithm (Li et al., 2012). The second is a multi-objective Simulated Annealing algorithm. In the case of Falling Tide, we have also implemented a version with a Variable Neighbourhood Search component. This component was not used in Li et al. (2012).

We varied the settings of the p metric to find the one that leads to the best performance of the metaheuristics. The results indicate that using $p = \infty$ metrics is the best option. In 4 out of 5 data sets, using this metric leads to the best Pareto set. In the case of instance 2, using all 3 metrics proved to be the best option. Scatter plots of the approximated Pareto front indicate that the front contains concave parts. This is an explanation for the fact that $p = \infty$ performs relatively well.

We compared Falling Tide with VNS to the one without VNS to determine whether there is an added value to the VNS component. In general, this proved to be the case. For all 5 instances, we found that the C-metric indicated a superior performance of Falling Tide with VNS to that without. We also compared Falling Tide to Simulated Annealing. We found that Falling Tide outperforms Simulated Annealing. In 3 of the 5 instances, the Pareto set found using Simulated Annealing weakly dominated none of the points found by Falling Tide with VNS. Also, in most instances Falling Tide with VNS finds more Pareto points than Simulated Annealing. For example, in instance 1, Falling Tide with VNS finds 41 points and Simulated Annealing finds 23.

A limited percentage of Pareto points found by the Falling Tide with VNS were dominated by the other metaheuristics on average. This is one indication that the set of points is a good approximation of the Pareto front. Moreover, the variation of the p metric and weights (w_t) has been used to find points that are evenly distributed over the front. This is another reason to conclude that we have successfully approximated the Pareto curve.

The main limitation of this research is the lack of knowledge about the actual Pareto front. This limitation is difficult to overcome due to the nature of the problem. The NP-hard property of the NSP is one of the major reasons we have used a metaheuristic to approach this problem. The NP-hardness also means it is difficult to know how close the approximated Pareto front is to the actual Pareto front. One way to reduce the uncertainty would be to generate some Pareto points using the MIP approach with a weighted objective function. We attempted to do this in this thesis, but the runtimes were too long to obtain any result. One important reason for this is that the objectives used conflict with one another. An opportunity for future research is to overcome this limitation, for example by designing a more efficient MIP approach that is able to generate some Pareto points. Then the approximated Pareto front could be compared to these points to get more certainty about the quality of the approximation.

Another limitation is that we have not considered historical data of the healthcare departments. Using historical data, hard constraints as well as soft constraints could be made more realistic. For example, data about rest days in the past could influence the need for rest days during the planning period. Future research could incorporate this historical data to a model that better represents the actual scheduling situation.

Other opportunities for future research relate to the hard constraint in the model. The list of hard constraints that ensure labour regulations are met is not complete in this thesis. Future research might expand the list of constraints. Another type of constraint that could be added is that of specific wishes by employees.

We conclude that a combination of a goal programming and Falling Tide with VNS is useful for solving the multi-objective NSP and approximating the pareto front. Using the method in this thesis, we were able to make a good trade-off among many objectives. As a result, we have found Pareto points that are more balanced when compared to single-objective approaches.

References

Åkerstedt, T. (2003). Shift work and disturbed sleep/wakefulness. *Occupational medicine*, *53*(2), 89–94.

Arbeidstijdenwet. (2020). https://wetten.overheid.nl/BWBR0007671/2020-01-01. (Accessed: 2020-05-22)

Burke, E. K., Li, J., & Qu, R. (2010). A hybrid model of integer programming and variable neighbourhood search for highly-constrained nurse rostering problems. *European Journal of Operational Research*, *203*(2), 484–493.

Burke, E. K., Li, J., & Qu, R. (2012). A pareto-based search methodology for multi-objective nurse scheduling. *Annals of Operations Research*, *196*(1), 91–109.

Caballini, C., & Paolucci, M. (2019). A rostering approach to minimize health risks for workers: An application to a container terminal in the italian port of genoa. *Omega*, 102094.

Czyzżak, P., & Jaszkiewicz, A. (1998). Pareto simulated annealing—a metaheuristic technique for multiple-objective combinatorial optimization. *Journal of Multi-Criteria Decision Analysis*, *7*(1), 34–47.

Dueck, G. (1993). New optimization heuristics: The great deluge algorithm and the record-to-record travel. *Journal of Computational physics*, *104*(1), 86–92.

Ehrgott, M. (2008, 12). Multiobjective optimization. *AI Magazine*, *29*, 47-57. doi: 10.1007/978-0-387-76635-5_6

Emmerich, M. T., & Deutz, A. H. (2018). A tutorial on multiobjective optimization: fundamentals and evolutionary methods. *Natural computing*, *17*(3), 585–609.

Gerber, M., Lindwall, M., Börjesson, M., Hadzibajramovic, E., & Jonsdottir, I. H. (2017). Low leisure-time physical activity, but not shift-work, contributes to the development of sleep complaints in swedish health care workers. *Mental Health and Physical Activity*, *13*, 22–29.

Groen, S., Kolkman, M., & de Leede, J. (2019). Duurzaam individueel roosteren: van idealisme naar realisme.

Hulsegge, G., van Mechelen, W., Proper, K. I., Paagman, H., & Anema, J. R. (2020). Shift work, and burnout and distress among 7798 blue-collar workers. *International Archives of Occupational and Environmental Health*, 1–9.

Inoue, T., Furuhashi, T., Maeda, H., & Takaba, M. (2003). A proposal of combined method of evolutionary algorithm and heuristics for nurse scheduling support system. *IEEE Transactions on Industrial Electronics*, *50*(5), 833–838.

Jafari, H., & Haleh, H. (2019). Nurse scheduling problem by considering fuzzy modeling approach to treat uncertainty on nurses' preferences for working shifts and weekends off. *Journal of Optimization in Industrial Engineering*.

Knauth, P., & Hornberger, S. (2003). Preventive and compensatory measures for shift workers. *Occupational medicine*, *53*(2), 109–116.

Knowles, J., & Corne, D. (2002, 06). On metrics for comparing nondominated sets. In (Vol. 1, p. 711-716). doi: 10.1109/CEC.2002.1007013

Legrain, A., Omer, J., & Rosat, S. (2019). A rotation-based branch-and-price approach for the nurse scheduling problem. *Mathematical Programming Computation*, 1–34.

Li, J., Burke, E. K., Curtois, T., Petrovic, S., & Qu, R. (2012). The falling tide algorithm: a new multi-objective approach for complex workforce scheduling. *Omega*, *40*(3), 283–293.

Lin, C.-C., Kang, J.-R., & Hsu, T.-H. (2015). A memetic algorithm with recovery scheme for nurse preference scheduling. *Journal of Industrial and Production Engineering*, *32*(2), 83–95.

Mutingi, M., & Mbohwa, C. (2014). Healthcare staff scheduling in a fuzzy environment: A fuzzy genetic algorithm approach..

Peters, J., Stephan, D., Amon, I., Gawendowicz, H., Lischeid, J., Salabarria, L., . . . others (2019). Mixed integer programming versus evolutionary computation for optimizing a hard real-world staff assignment problem. In *Proceedings of the international conference on automated planning and scheduling* (Vol. 29, pp. 541–554).

Post, G., & Veltman, B. (2004). Harmonious personnel scheduling. In *Proceedings of the 5th international conference on the practice and automated timetabling* (pp. 557–559).

Sallinen, M., & Kecklund, G. (2010, 03). Shift work, sleep, and sleepiness - differences between shift schedules and systems. *Scandinavian Journal of Work, Environment —& Health*, *36*, 121-133. doi: 10.5271/sjweh.2900

Sarkar, P., Chaki, N., & Chaki, R. (2019). A study of resource optimization for nurse scheduling problem. In *2019 4th international conference on computer science and engineering (ubmk)* (pp. 757–761).

Tan, M., Gan, J., & Ren, Q. (2019). Scheduling emergency physicians based on a multiobjective programming approach: A case study of west china hospital of sichuan university. *Journal of healthcare engineering*, *2019*.

Van den Bergh, J., Beliën, J., De Bruecker, P., Demeulemeester, E., & De Boeck, L. (2013). Personnel scheduling: A literature review. *European journal of operational research*, *226*(3), 367–385.

A Linearization of Maximization Expression

The expression to be linearized is of the form

$$\delta = max\{0, \zeta\} \tag{26}$$

where ζ is a variable and therefore δ is a variables. To linearize this expression we add the following constraints to the model.

$$\delta \geq 0 \tag{27a}$$

$$\delta \geq \zeta \tag{27b}$$

B Additional Tables

Method	Falling Tide with VNS	Falling Tide without VNS	Simulated Annealing
Number of Pareto Points	29	32	24
Number of Runs	100	100	250
Runtime/seconds	4672	5168	5489

Table 18: Descriptions of the Runs of the Algorithm Comparison Experiment Instance 2

Method	Falling Tide with VNS	Falling Tide without VNS	Simulated Annealing
Number of Pareto Points	8	9	3
Number of Runs	100	100	250
Runtime/seconds	120	138	103

Table 19: Descriptions of the Runs of the Algorithm Comparison Experiment Instance 3

Method	Falling Tide with VNS	Falling Tide without VNS	Simulated Annealing
Number of Pareto Points	11	7	11
Number of Runs	100	100	250
Runtime/seconds	765	720	585

Table 20: Descriptions of the Runs of the Algorithm Comparison Experiment Instance 4

Method	Falling Tide with VNS	Simulated Annealing
Number of Pareto Points	1	14
Number of Runs	20	40
Runtime/seconds	5090	24361

Table 21: Descriptions of the Runs of the Algorithm Comparison Experiment Instance 5

Metric	1	2	∞	all	Average
1	-	0.41	0.14	0.14	0.23
2	0.81	-	0.14	0.14	0.37
∞	0.06	0.14	-	0.0	0.07
all	0.19	0.14	0.14	-	0.16
Average	0.35	0.23	0.14	0.10	

Table 22: C-Metric Vales Metric Comparison HEMAVAE00

Metric	1	2	∞	all	Average
1	-	0.60	0.30	1.00	0.63
2	0.60	-	0.30	0.60	0.5
∞	0.70	0.80	-	0.70	0.73
all	1.00	0.60	0.30	-	0.63
Average	0.77	0.67	0.30	0.77	

Table 23: C-Metric Vales Metric Comparison Longgeneeskunde

Metric	1	2	∞	all	Average
1	-	0.71	0.25	1.00	0.65
2	0.83	-	0.25	0.83	0.64
∞	1.00	0.86	-	1.00	0.95
all	1.00	0.71	0.25	-	0.65
Average	0.94	0.76	0.25	0.94	

Table 24: C-Metric Vales Metric Comparison Spoedeisende Hulp

C Additional Graphs

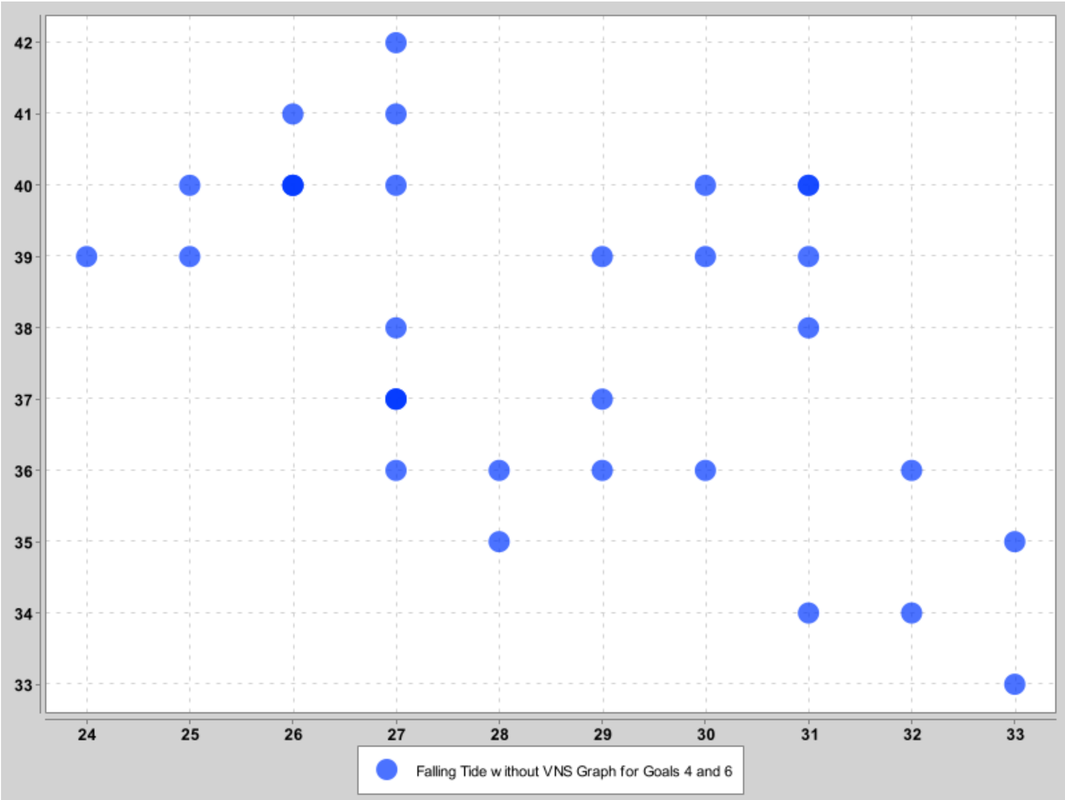

Figure 10: Scatter Plot of Goal 6 against Goal 4 for Falling Tide without VNS Instance 2

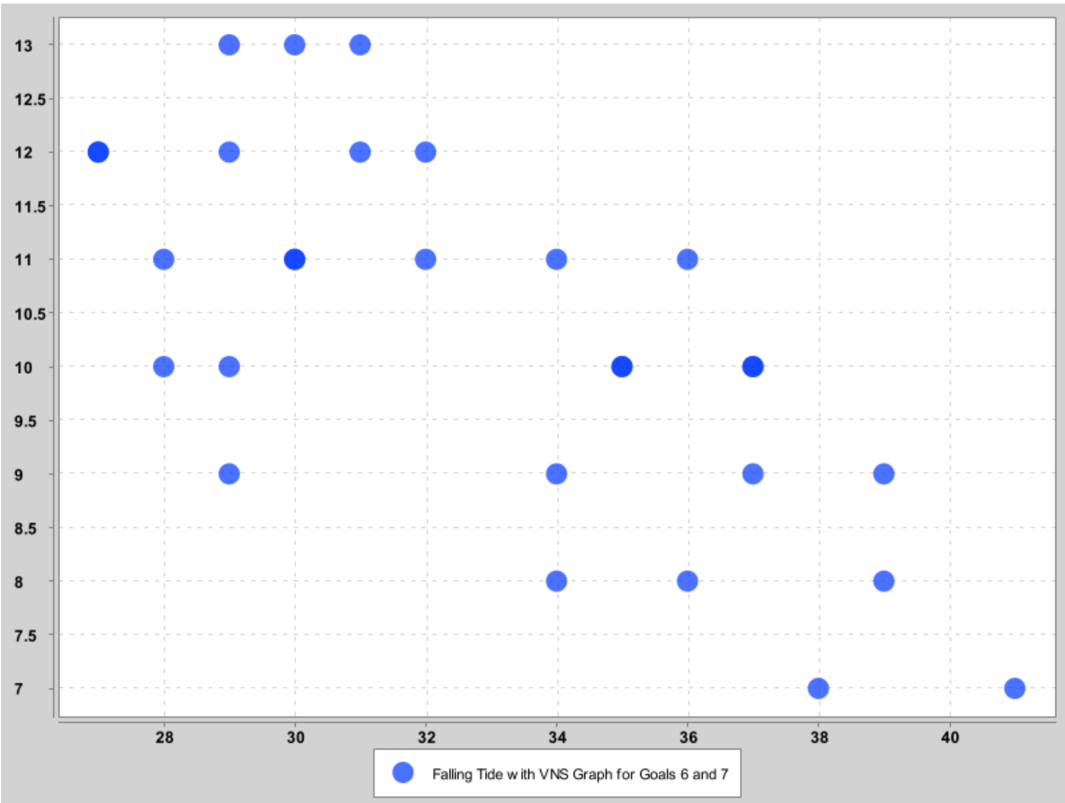

Figure 11: Scatter Plot of Goal 7 against Goal 6 for Falling Tide with VNS Instance 2

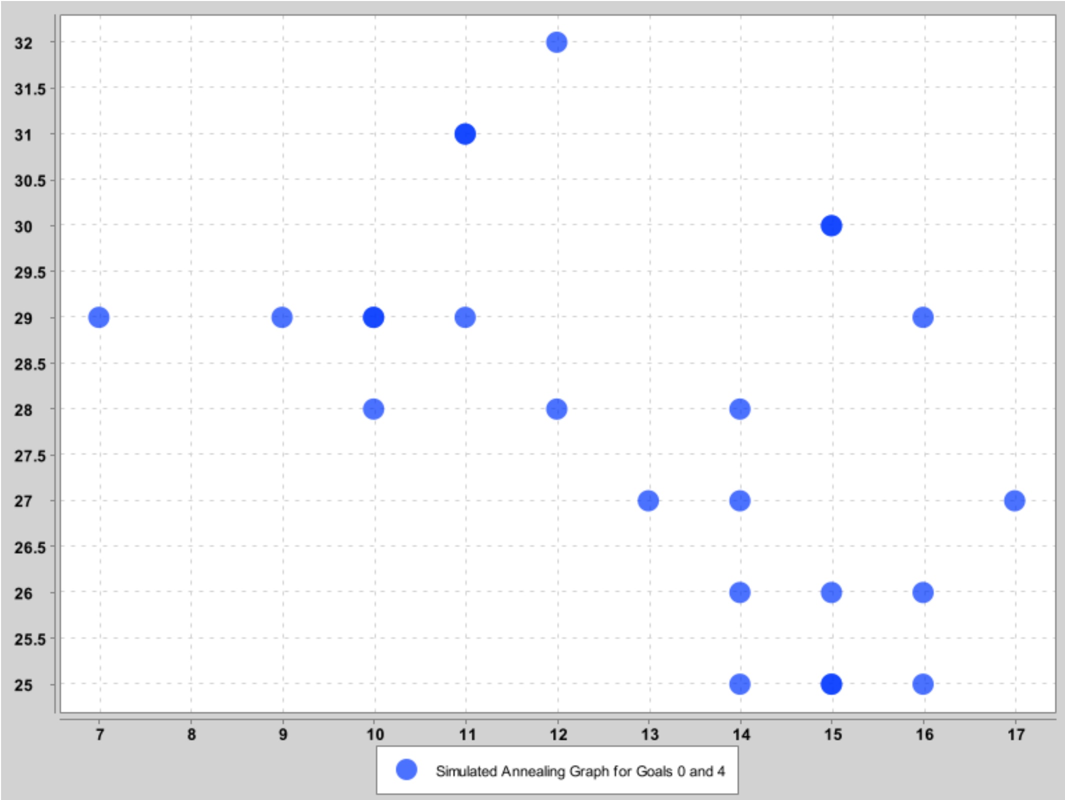

Figure 12: Scatter Plot of Goal 4 against Goal 0 for Simulated Annealing Instance 2

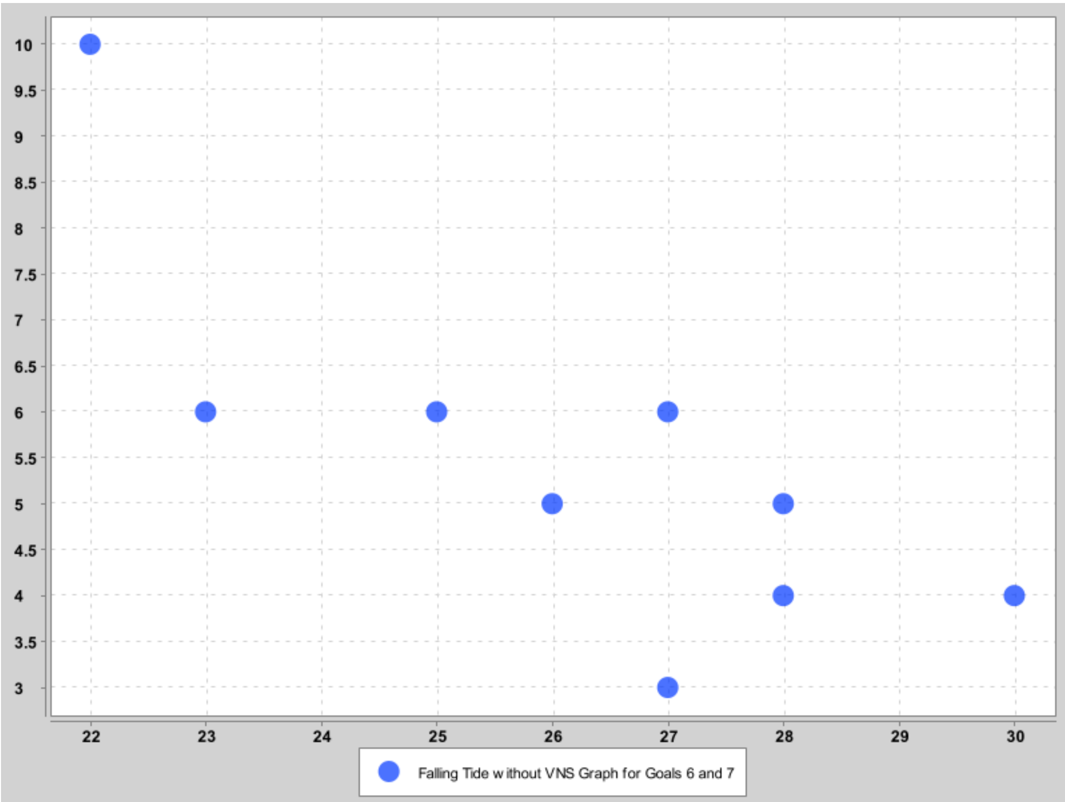

Figure 13: Scatter Plot of Goal 7 against Goal 6 for Falling Tide without VNS Instance 3

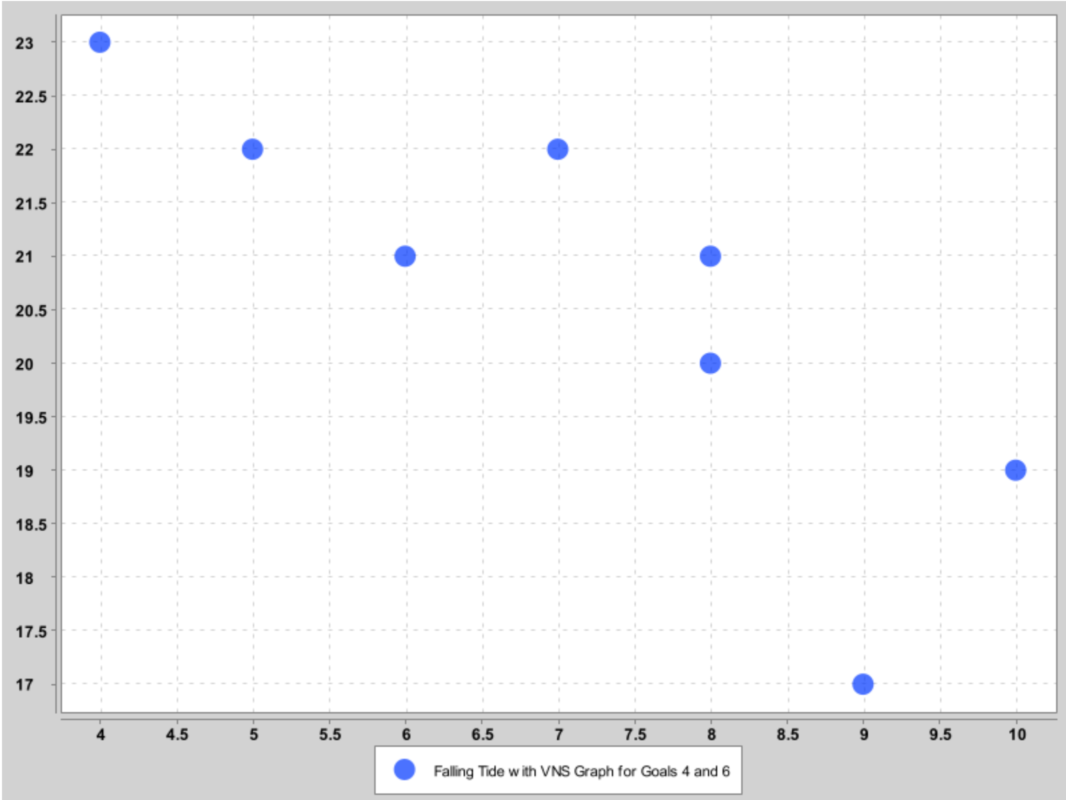

Figure 14: Scatter Plot of Goal 6 against Goal 4 for Falling Tide with VNS Instance 3

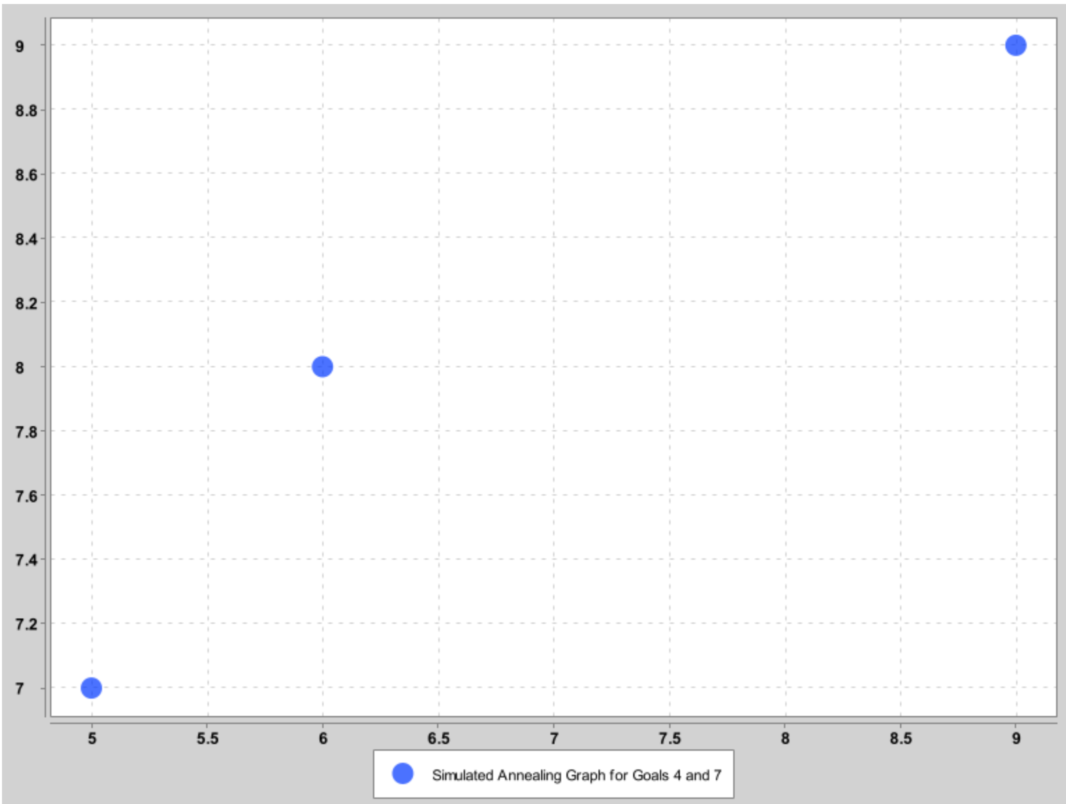

Figure 15: Scatter Plot of Goal 7 against Goal 4 for Simulated Annealing Instance 3

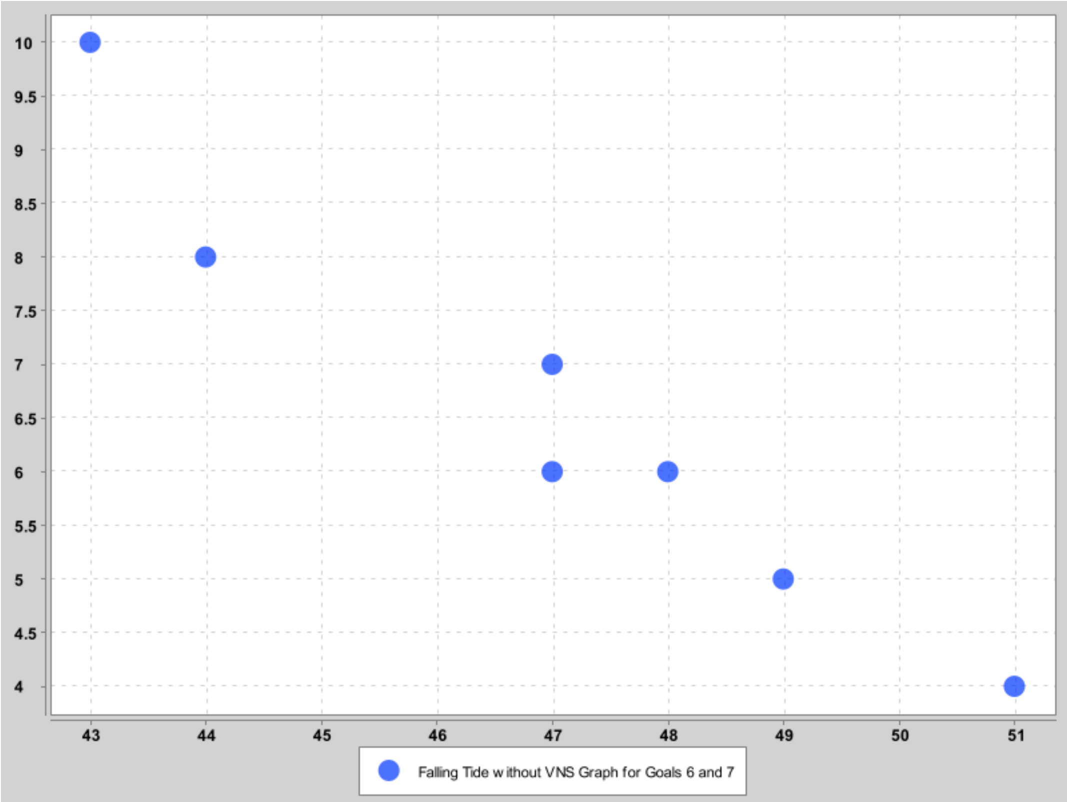

Figure 16: Scatter Plot of Goal 7 against Goal 6 for Falling Tide without VNS Instance 4

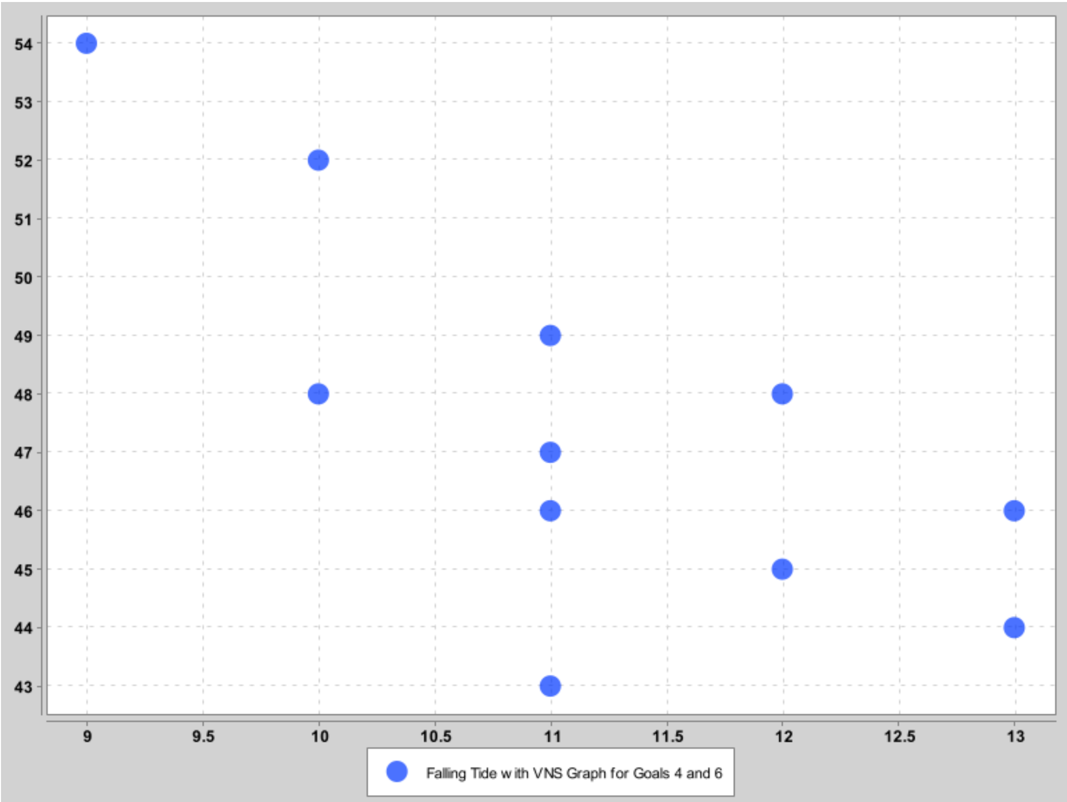

Figure 17: Scatter Plot of Goal 6 against Goal 4 for Falling Tide with VNS Instance 4

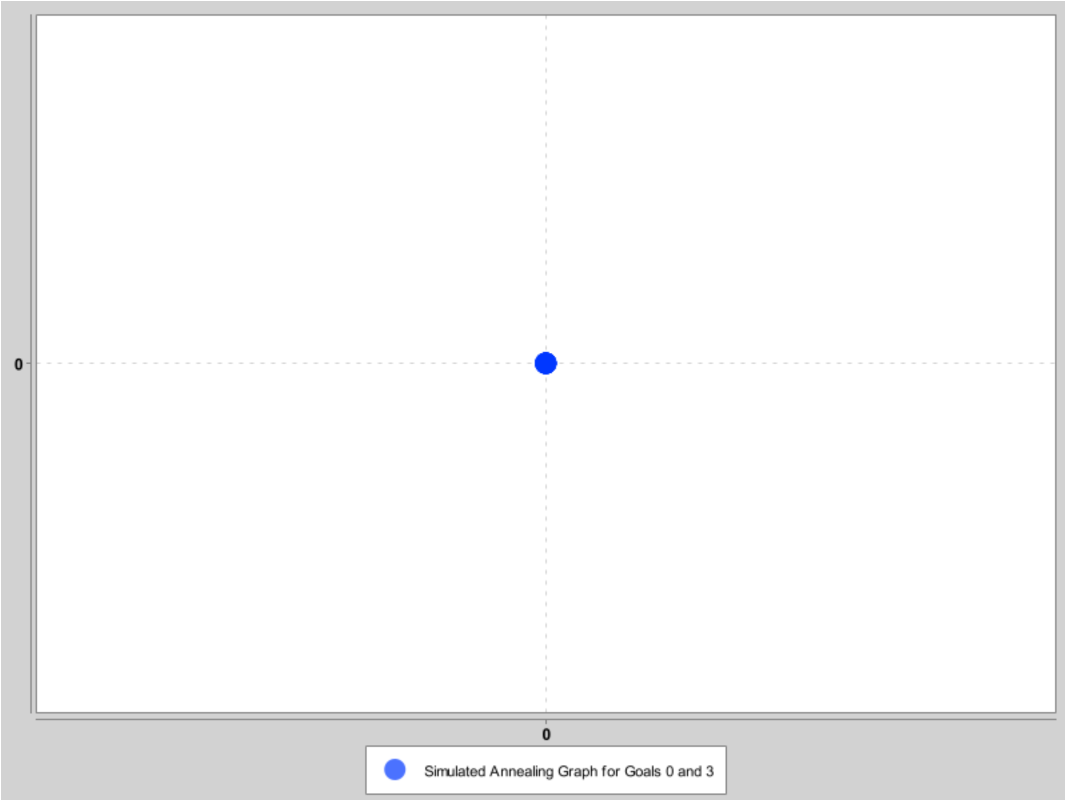

Figure 18: Scatter Plot of Goal 3 against Goal 0 for Simulated Annealing Instance 4

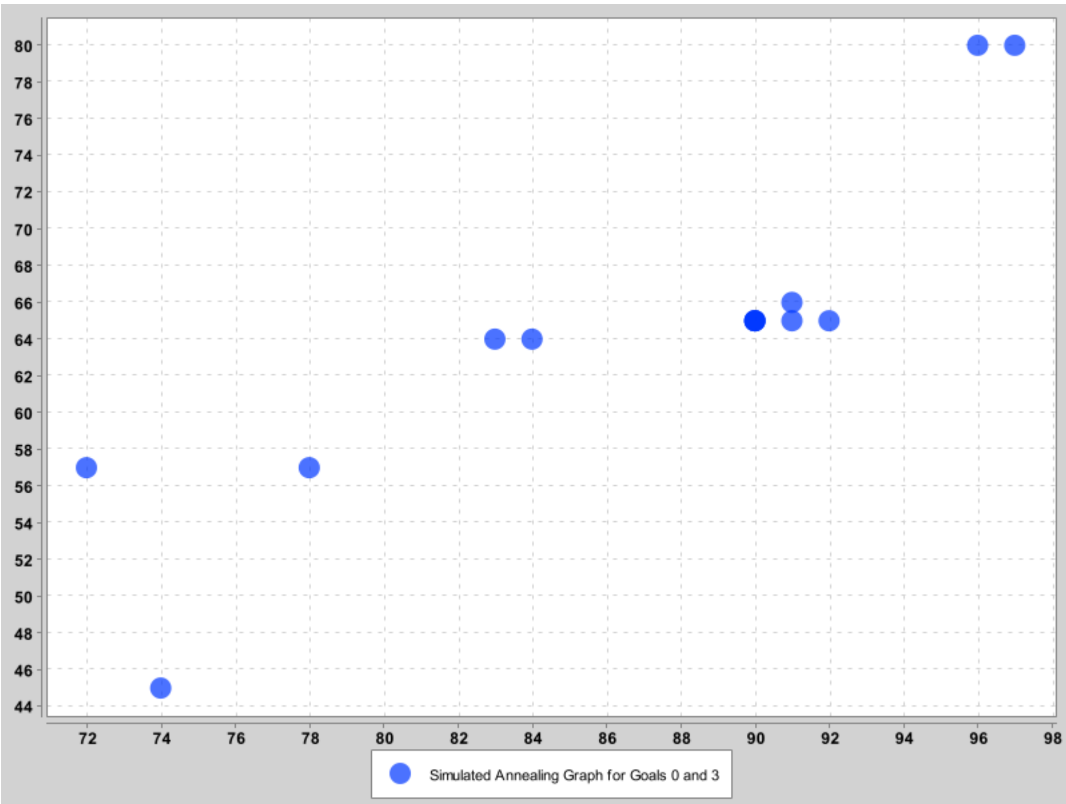

Figure 19: Scatter Plot of Goal 3 against Goal 0 for Simulated Annealing Instance 5