Kylian Kropf

Supporting non-expert users in modelling and understanding AI: an interactive CP approach

Master Thesis Computer Science

Supporting non-expert users in modelling and understanding AI: an interactive CP approach

Bringing the power of advanced optimisation in employee scheduling to small and medium sized organisations

by

Kylian Kropf

to obtain the degree of Master of Science
at the Delft University of Technology,
to be defended publicly on Thursday, March 14, 2024 at 10:30 AM.

Student number:	5366968	
Project duration:	November 1, 2022 – August 31, 2023	
Thesis committee:	Dr. N. York-Smith,	TU Delft, Supervisor
	Dr. M. Tielman,	TU Delft, Committee member
	R. Bormans,	Dyflexis B.V., Company supervisor

This thesis is confidential and cannot be made public until December 31, 2024.

An electronic version of this thesis is available at https://repository.tudelft.nl/.

TU Delft

Abstract

This thesis proposes and develops an interface and model in which advanced optimisation for general employee scheduling is made available to non-experts in computer science or optimisation. The interface teaches, guides, configures, dynamically creates a constraint programming (CP) model, iteratively improves, decreases black box properties, increases trust in the outcome, and complies with relevant European Union Artificial Intelligence law. The objective of this study is to allow a wider range of organisations to take advantage of CP techniques, with the potential to greatly improve efficiency, reduce unfairness, meet company goals, and improve employee satisfaction.

Employees are assigned to personalised shifts based on the expected demands of departments within an organisation, these are set by the domain expert in the field of employee scheduling. Next, to tailor the model to the organisation's needs, the domain expert is guided in setting both the restrictive assumptions and priorities of shift assignment. To optimise the generated optimal schedule, it is encouraged to create and compare multiple sets of configurations and subsequent schedules. Historical demand data, organisation structure, contract information, and employee preferences are included.

Multiple visual design iterations have been made, after which a working interface has been developed and improved iteratively in conjunction with experts in the field. Validations with external domain experts from various industries and organisations have shown that the interface performed effectively in supporting the objectives. Further research can be done to improve the speed of solving, implement diversity of solutions, support for highly custom constraints based on natural language, or interface reusability for other optimisation problems.

Keywords: combinatorial optimisation, multi objective optimisation, general personnel scheduling problem, domain expert systems, understandable AI, XAI

Preface

Before diving into the core of this master thesis, I wish to express my profound gratitude and appreciation to a number of individuals and institutions, for without them this research would not have been possible.

First and foremost, I wish to express my deepest gratitude to my thesis supervisor, Neil York-Smith. His expertise and insightful feedback have been instrumental in shaping my work and helping me navigate the challenges of this field. I fondly look back at our collaboration over the past 9 months.

I am grateful to Dyflexis and all the employees involved for their collaboration and support during this project. Their expertise and resources enhanced the quality and relevance of my research. In particular, I would like to thank Renier, my supervisor at Dyflexis; you proved to be a smart person who is excellent at giving feedback and being a great sparring partner. I would like to take this opportunity to pay a tribute to Matthijs and Thomas for their unconditional support over the past 6 years. Without you, I would not be where I am right now, both personal and professionally. You are a role model for me.

Also I wish to express my heartfelt appreciation to the TU Delft Algorithmics Faculty members, dr. Jip J. Dekker from Monash University Australia and Linnea Stjerna from KTH Royal Institute of Technology whose expertise, insightful feedback, and suggestions have helped me improve my work.

Lastly, but by no means least, my profound thanks to my friends and family. Your unwavering faith, endless encouragement, and constant love served as my foundation during this challenging but rewarding journey. This achievement is as much yours as it is mine.

All I have left to say is have fun reading! I tried my best to make the next 35 pages accessible and easy to understand.

Kylian Kropf
The Hague, August 2023

Contents

1 Introduction

This thesis aims to address the knowledge gap in making advanced optimisation available to non-experts in computer science. The interface should support domain experts in the field of employee scheduling to tailor the algorithm developed to their organisation-specific needs and support the understandability of the outcome. The lessons learnt from this thesis can support making the power of optimisation available for a much wider variety of organisations.

Employee scheduling is a process that every company owner, manager, or team leader has to deal with on a regular basis. The scheduling process involves assigning shifts to individual employees based on a variety of factors such as organisational needs, employee availability, labour regulations, and preferences of the employees and organisation. This can be a challenging task, as employees are the absolute basis for an organisation to achieve goals. A good schedule contains a careful trade-off between the goals of the organisation and the personal preferences of employees within predefined legal or organisation-specific boundaries.

In order to accelerate the progress of this thesis and work towards a concrete problem, this study has been conducted in collaboration with Dyflexis B.V., from now on referred to as the hosting company, a company specialised in employee scheduling software for small and medium-sized organisations.

1.1. Problem description

Optimisation techniques provide computer scientists and experts in operations research with an extensive but complicated tool to produce an optimal combination of factors to a problem. This problem is tailored beforehand to a specific organisation and circumstance. Increased access to accessible high-speed computing and improved modelling techniques have improved the speed and efficiency, and thus the real-world impact of this optimisation. These improvements help the overall efficiency of organisations already enjoying the advantages of optimisation while widening the efficiency gap with non-users.

Due to technical complexity, the usage of these optimisation tools is mostly limited to organisations that can command large amounts of resources [Van den Bergh et al., 2013]. This often means that optimisation is not accessible to domain experts in small and medium-sized organisations. This is why there is a need for a software solution that is tailored to their organisation-specific needs and constraints. The main difficulty lies in the fact that this software solution must support the actual domain export in understanding what optimisation is and what potential pitfalls are while gaining trust in the outcome. Only when this is understood can one unlock the full potential. Moreover, the software solution must comply with the relevant laws for the actual users in a production setting.

1.2. Background

1.2.1. Employee scheduling problem

In the employee scheduling problem, an employee is assigned to a shift within a department in a time interval; the tasks within this shift are not taken into consideration [Ernst et al., 2004]. For example, an employee can be assigned to the cashier department on Friday morning between 6 AM and 11 AM. The queue of actual tasks to perform in this 5-hour shift is not relevant to the schedule. Shifts have a starting time and an ending time, and a single employee is assigned to the shift. For multiple employees at the same moment, multiple single shifts have to be assigned. If no employee is assigned to a shift, this is called an open shift. Shifts follow certain constraints, e.g. related to the rest time or minimal and maximal length; the law is always the upper bound, but usually, organisations follow looser shift times. These constraints can differ per department and, therefore, can get complicated.

1.2.2. Employees

An employee can have a contract with a fixed number of hours per week (e.g., 24 hours) or a variable number of hours per week, optionally with a maximum number of hours. An employee can be scheduled in multiple departments during the week, or even during the day. The departments in which an employee is available for assignment are predefined. Also predefined for every employee is the availability per hour; three options are possible: available when an employee prefers to work on that time frame, unavailable when an employee is not available, and indifferent, which is the default state when none of the first two options have been defined. Finally, for each employee, the hourly rate is provided to calculate the expected cost of the employee according to the schedule.

1.2.3. Current scheduling approach

Currently, for customers from the hosting company, employee scheduling is a manual process that requires a lot of time. The scheduler, or domain expert, first has to create all shifts and then assign one of the employees to the individual shifts. Both of these actions are a manual process. The number of employees that need to be scheduled at a specific time for a specific department is determined beforehand based on the experience of the scheduler. The expected workload also called the forecast, is an important measure, as inaccurate forecasting can lead to unnecessary over- or under-staffing or costly delays.

Company objectives such as maximised output are often not taken into account, let alone achieving an optimal shift assignment. The current scheduling process is often a concentrated process within organisations; this makes it an error-prone process. This can lead to unfair situations, for example, when employees close to the scheduler are favoured in the shift assignment process. If the desired schedule cannot be met, the scheduler must personally reach out to the employees to ask if they are available to work and fill the gaps.

Most of the processes related to the scheduling process can be automated, e.g. employees can enter their availability themselves or the schedules are distributed automatically. The expected workload can be predicted by domain-based exports, for example, weather predictions.

1.2.4. Existing solutions

Automatic employee scheduling is not new [Dantzig, 1954], and some specific problems such as the nurse rostering problem (NPR) have been studied in the literature for several decades and can have their own designated efficient algorithms. These algorithms aim to find a (near) optimal solution considering predefined objectives like minimal costs or employee happiness. Unfortunately, small changes in these problems, including organisation-specific rules or adding regulatory changes, often make these designated algorithms unusable [Van den Bergh et al., 2013]. These algorithms are very technical, therefore, hard to configure, and are typically used by experts in computer science or operations research. For medium and small organisations, the significant costs and complexity of implementing these solutions can be a barrier to entry.

1.2.5. Constraint programming

One of these advanced approaches for implementing optimisation is constraint programming (CP). CP is a mathematical technique used to model and solve complex real-world problems. Instead of step-by-step instructions, it involves:

- defining a set of hard constraints, which are conditions that must be satisfied for a given solution to be valid
- defining a set of soft constraints and their weights which will be used to measure the quality of the solution guides the algorithm to the best solution

These two combined are called the model. The model is fed into a mathematical solver that automatically searches for a solution that satisfies all the constraints simultaneously while maximising predefined heuristics. This is great for employee scheduling optimisation, as it can enforce rules dynamically without worrying about how to solve them. The processes of configuring the input, programming, improving the model itself, dealing with errors, and interpreting the result are highly technical and are only available to those highly technical skilled in computer science. All this has to be set up in close cooperation with the domain experts who are responsible for solving the aimed problem.

1.2.6. European artificial intelligence act

Recently, the European Council has adopted its common position on the Artificial Intelligence Act. The proposal defines AI systems as software that is developed to *"generate outputs such as content, predictions, recommendations, or decisions influencing the environments they interact with" [EP, 2023]*. Under this definition, a CP model used for employee scheduling would be considered an AI system.

In the proposed regulations, AI systems will need to undergo conformity assessments before being put into use. Conformity assessments will determine whether an AI system meets the technical requirements and safety standards set out in the regulations. This has the objective of requiring AI providers to provide clear and concise information to users about the AI systems they use and how the requirements for their internal function have been defined to ensure ongoing monitoring, transparency, and reporting.

A big part of the Act is dedicated to data quality to use training data while training machine learning models. CP does not rely on the need for historical data to calculate a feasible schedule; this is an advantage since the potential bias included in the training set is ruled out. The chapters related to data quality will not be taken into account in this thesis.

1.3. Goal and research questions

This research aims to explore and establish methods that improve the understanding and usability of CP for users without a background in computer science or optimisation. The focus is on developing a user-friendly interface which teaches, guides, configures, dynamically creates a CP model, iteratively improves, decreases black box properties, increases trust in the outcome, and complies with relevant EU laws.

More concrete, a more elaborate version of the general employee scheduling problem is used. The objective of this specific problem is to allow a wider range of organisations to take advantage of CP techniques, with the potential to greatly improve efficiency, reduce unfairness, meet company goals and improve employee satisfaction.

These goals lead to the main research question of this thesis:

1. Is it possible to create an interface for configuring a CP model for organisation-specific rostering problems in a general way where the user interface will explain the impact of its constraint and objective function to non-expert users?

The subquestions of this thesis are:

1. How to show the preview of changing constraints to the user?
2. How can the user interface deal with explaining the impact of changing constraints or the objective function on the outcome?
3. Can the interface comply with the EU AI Act?

The main research goal will be reached if:

1. Training of the domain expert beforehand will not exceed 2 hours
2. Modelling should take a maximum 1 hour to get to a workable outcome (customer input like contracts, and availability need to be done beforehand; everything that needs to be in place for manual schedule)
3. Actual customers (domain-expert) say they will use it

1.4. Scope and limitations

The focus of this project will be on the person responsible for the employee scheduling process in small and medium-sized organisations. It is assumed that this person has limited technical knowledge and computer skills; however, will be a domain expert in managing employees and configuring the employee schedule of the organisation. The interaction will be limited to a single organisation location with multiple departments. Moreover, the assumption is made that employee details (contract information, availability, skills) and organisation details (department names, historical workload) are available and imported beforehand. In addition, the focus will be on setting up the CP model. Other parts of the process, such as publishing the schedule or informing the employees, are beyond the scope of the study. Concerning the European AI Act, the focus will be on complying with the aspects related to the interface.

1.5. Outline

The further structure of this thesis is guided by the research questions. In Chapter 2, the state of the available literature will be discussed to gain insight into the state-of-the-art and identify the literate gaps which this thesis can fill. In Chapter 3 the methodology is presented, which is executed in Chapter 4. The evaluations of the working prototype are described in Chapter 5. Finally, Chapter 6 provides conclusions, recommendations, and directions for future research.

2

Literature Review

For the purposes of this thesis, we will focus on what is relevant considering the background and goal of this research presented in Chapter 1. The purpose of the literature study is to gain insight into existing knowledge and identify gaps in the literature that can be fulfilled in this thesis. Section 2.1 discusses past research on algorithms to solve employee scheduling problems. Section 2.2 describes Constraint Programming and how it can support solving employee scheduling problems. Section 2.3 focusses on supporting trust when designing automation solutions.

2.1. Employee scheduling problem

The employee scheduling problem has been extensively researched. Ernst et al., 2004 reviewed about 700 articles on the subject published until 2004. Van den Bergh et al., 2013 extended this by reviewing 293 articles published from 2004 until 2012. Given this vast body of work, conducting a review of all existing literature on employee scheduling would not only be time-consuming but also might offer limited new insights given the extensive analyses already performed. If the literature from 2012 and beyond has significant new insights, this will be included.

2.1.1. Directions of research

First it is important to understand the different directions of research in employee scheduling. Van den Bergh et al., 2013 classified this into four categories, each of which will be summarised below together with a description of the general employee scheduling problem we are considering.

Personnel characteristics, decision delineation and shift definitions

A distinction should be made between individual employees based on characteristics such as their skill set, contract type as part-time or full time; although the fast majority of papers study full-time only. Shift definitions include the degrees of flexibility which the shift structure has, e.g. is overlap allowed and are the shift start and end times flexible.

In this thesis, the interface should be applicable to a wide variety of organisations. This means that for every employee the contract hours can be specified, or left empty when the employee has a variable-hours contract. The skill is determined based on the department in which an employee has permission to work in. The structure of the shift can be determined individually for each department.

Constraints, performance measures and flexibility
Distinction should be made between hard-constraints and soft-constraints, the most important constraint that was found in almost 75% of the reviewed papers is the coverage constraint [Van den Bergh et al., 2013]. Most of these prohibit under-planning in general or those with specific skills. Under-planning can not always be avoided in real-world scenario's, e.g. when not enough employees are available. In this thesis, the coverage constraints will be a soft constraint. This is to make sure that no solution at all will be found when, for example, only 7 of the desired 8 employees are available.

Scheduling breaks is often omitted from employee scheduling problems [Van den Bergh et al., 2013]. The timing of the brakes can be crucial, as some departments require a minimum number of employees to be available. Thompson and Pullman [Thompson and Pullman, 2007] surveyed 64 articles in which planning breaks in advance is compared to scheduling breaks in real time. Scheduling them in advance results in more complexity, more costly schedules, and less productive workforce. Since the current process does not incorporate scheduling breaks either, breaks will not be taken into consideration.

Tracking the quality of the generated schedule can be done on a variety of metrics, depending on the objectives and priorities of the organisation, e.g., minimising the number of employees, total costs or the number of hours over / under scheduled. To support a wide variety of organisations, the preferences of the employees will also be taken into account in this thesis.

Solution method and uncertainty incorporation
The literature review by Van den Bergh et al., 2013 showed that mathematical programming techniques are widely used. Other categories are simulation, constraint programming (CP), and queueing. Highly constrained employee scheduling problems offer an ideal framework for the use of CP methods due to the ease of adding or changing constraints.

Main categories of uncertainty are:

- Demand: not certain if the predicted workload will be met
- Arrival: broken equipment or arrival of calls

- Capacity: No-shows or sickness of employees

Most papers reviewed by Van den Bergh et al., 2013 take into account demand and/ or arrival, capacity uncertainty is only taken into account in five of the papers. In this thesis, uncertainty will not be directly incorporated, as it is not a fully automated process and the experience of domain experts will be required to determine the desired workload.

Application area and applicability of research
Van den Bergh et al., 2013 found that half of the articles considered are developed for the transportation industry, the nurse scheduling problem ranks second in popularity. Testing is done mostly with real-world data; when the research was actually implemented, details of the process or the observed results were hardly ever provided. Comparing of the solution is often benchmarked against similar problems that lack diversity. This is probably caused by the lack of integration of related software system(s).

2.2. Constraint programming (CP)

One of the techniques used to solve employee scheduling is CP. The field of CP is evolving fast, to keep this overview concise, this chapter covers three research areas of CP which can be useful in making CP more accessible for non-expert users [Wallace, 2020].

2.2.1. Visualisation of a CP solver

Because CP is a declarative approach to problem-solving, it is often difficult to visualise how a solver will approach the search space of the model [Howell et al., 2018]. To support the understanding of expert users, one way is to visualise the search tree (e.g. Gist) or network in which the nodes represent the variables of the problem and the edges represent the constraints between those variables [Mak, 2007]. The existing tools and literature focus mainly on improving the understanding ability of the developer [Goodwin et al., 2017]. For non-expert users, the literature is limited to focussing on the solution process and how CP-solvers work, and also the scope of these specific problems is limited [Simonis et al., 2010].

2.2.2. Debugging a CP model

Even for an expert, debugging unexpected behaviour in a model can be difficult. For example, discovering the reason why not all hard constraints were found, also known as an unsatisfactory result. One way to debug a constraint programming solver after an unsatisfiable result is to carefully inspect the constraints that have been specified for the problem. This can help identify inconsistencies or errors in the constraints, such as redundant or contradictory constraints or constraints that are too restrictive and prevent the solver from finding a solution. This process can be speeded up and made more accessible with techniques such as findMUS that automatically search for a Minimal Unsatisfiable Subformula or MUS [Nadel et al., 2013]. Another way to debug a constraint programming solver is to use tools that allow you to walk through the search process and see how the solver explores the space of possible solutions [Simonis et al., 2010, Naish, 1997, Simonis et al., 2000]. This can help identify any problems with the search algorithm, such as incorrect heuristics or other issues that prevent the solver from finding a solution. By understanding the constraints and the search process, you can better identify and fix any problems that may prevent the solver from finding a solution [Simonis and Aggoun, 2000]. Problems related to invalid input can be tackled in the user interface (UI) or with asserts before running the solver. The literature does not specifically describe how to prevent hard-constraint violations for nonexperts based on infeasible input.

2.2.3. Trust in the outcome of CP solver

Using feedback to increase the transparency of a system is critical in building trust in optimisation systems [Glass et al., 2008, Liu et al., 2020], as well as any limitations that may be present in the system. Additionally, it is helpful to ensure that the constraints and objective function used by the model are clearly defined and well understood by the user. This can be achieved through careful specification and documentation of constraints, as well as regular testing and validation to ensure that it is operating as intended [Liu et al., 2022]. Furthermore, providing user-friendly tools and interfaces to define and modify constraints can also help improve trust in the outcome by allowing users to understand and control the constraints that are being used more easily [Goodwin et al., 2017]. A 2020 study analysed how people create representations of constraint problems [Zhu et al., 2020]. The observations revealed patterns allowing creating better UIs that improve and broaden access to constraint modelling to non-specialists. These patterns include:

- Combine different forms of expressions
- Support the process rather than just the outcome
- During validation sessions: leveraging implicit information as a source of data

Surprisingly, there has been little work to describe how to develop and maintain models. The authors are not aware of any papers in the last decade.

2.2.4. User inteface in optimisation modelling

Despite the critical role of optimisation methods in various applications, there is a lack of research on the integration of these methods into user interfaces. One notable exception is the work of Borissova and Mustakerov, 2016, who delved into the presentation of optimisation outcomes in wind farm allocation through graphical user interfaces. Their exploration underscores the importance and potential of making intricate computational processes more accessible and intuitive to users.

2.3. Trust in automation systems

In contrast to the UI in optimisation modelling, there is sufficient research on automation systems in general. The consensus is that trust includes three components [Hardin, 2006].

1. There must be a truster to give the trust and someone to accept it

2. The accepter must have an incentive to perform the task
3. There must be a possibility that the receiver will fail to perform

This applies to interpersonal relationships such as human-automation relationships. Although trust in technology has some parallels with interpersonal trust, there are differences. One potential reason for these similarities is that, to some extent, users place the trust of automation systems in the designers of these systems [Oglesby et al., 2014]. The differences are that in interpersonal trust depends on the ability, integrity, or benevolence of a trustee [Mayer et al., 1995]. Also, contrary to relationships between people, users of systems are commonly biased and think machines are perfect [Dzindolet et al., 2003]. However, this trust can dissolve quickly after system errors, as relationships with automated system dependability and predictability replace faith as the primary basis of trust [Madhavan and Wiegmann, 2007]. Hoff and Bashir, 2014 identified 3 sources in human automation trust:

- The human user: e.g. culture, age, gender, personality
- The environment:
  - Internal: e.g. self-confidence, subject matter expertise, mood
  - External: e.g. perceived benefits, organisation setting, framing of task
- The automated system itself:
  - Pre-existing knowledge: e.g. expectations, reputation, understanding of system
  - Initial system performance: e.g. reliability, validity, predictability, usefulness
  - Design features: e.g. appearance, ease-of-use, communication, transparency

Although trust is crucial in regulating the user's dependence on automated systems, it is impossible to speculate on the influence of each of the sources indicated above due to the vast range of paradigms examined.

Too much trust in automation is dangerous [Parasuraman and Riley, 1997], as it can lead to disuse, e.g., to overlook mistakes. The same article also found that better operator knowledge of how the automation solution works results in a more appropriate use of automation. The Goldilocks trust dilemma is how to design applications that implement the right amount of trust. Liu et al., 2022 focusses on improving trust in the automated system itself, as this takes into account the understanding and experience of users. Other researchers have suggested that interaction leads to increased trust [Meignan et al., 2015]. Herlocker et al., 2000 found that explaining the reasons why automation systems are imperfect resulted in greater trust.

2.4. Conclusion and gaps

To summarise this overview of the literature, especially non-flexible subproblems of employee scheduling problems have been studied extensively. Furthermore, constraints related to labour law, contractual obligations, or specific organisations are only covered by a small number of studies. Van den Bergh et al., 2013 noted that many research projects do not make it until implementation in practice, one of the main reasons being the lack of integration with other software systems.

Constraint Programming is one of the options to create a scalable and flexible general employee scheduling model. Since debugging unsatisfiable results for nonexpert users is hard, this should be prevented at all times. Notable research has not been conducted on supporting nonexpert users in CP. More generally, in automation, trust in the outcome is an essential factor for the domain experts to use and keep using solutions. Trust can be increased through interaction, explaining the pitfalls, considering the sources of trust, and taking into account the experience of the user.

Based on this literature survey, and to the best of the authors' knowledge, this thesis proposes the first interface which has the following combined properties:

- configure a CP model specifically developed for non-experts
- general scheduling solution, supporting organisation specific configuration
- increases the trust in the outcome
- flexible technical setup to add additional constraints in future
- complies to relevant European Union AI legislation
- has integration with existing software systems
- validate the working prototype with domain experts

The proposed interface is specifically designed to suit the needs of domain experts in the field of employee scheduling; however, we believe that the same principles will be more widely applicable to create models for specific optimisation problems.

3 Methodology

The approach of this research consists of several components, namely participatory design, actual software development, and multiple rounds of validations throughout the process. This methodology was chosen to ensure the development of a user-friendly tool that aligns with the final goals of this thesis.

A significant part of this research was conducted in collaboration with a development team of the hosting company. Often, academia lacks both these resource capabilities and real-world test beds to which commercial organisations have access, and this collaboration effectively fills that gap. This collaboration allowed for the actual development of a web-based proof of concept.

The support team allocated to this project consisted of:

1. Senior User Interface Designer: the person who is an expert in designing user interfaces which are intuitive and user-friendly. Does not have a deep understanding of the technicalities of constraint programming but understands how to translate complex systems into easy-to-understand and navigate user interfaces. Works in close collaboration with the final decision-maker to iteratively improve the designs.

2. All-round Software Developer: this individual is proficient in several programming languages and has experience in all stages of the software development process. The main responsibility is to implement the visual designs and set up the connection between the front-end and the back-end solver framework.

3. Product Owner: the stakeholder representative for this project, is mainly responsible for overseeing the agile process and forming the bridge between the research team and other stakeholders.

This research team, with its mix of expertise, is a representative reflection of a software development team, as it exists in many organisations. This makes the team not only well-equipped to tackle this research project but also to contribute to academic literacy.

3.1. Constraint programming model

The initial phase is to test the feasibility of setting up the MiniZinc CP model. The objectives of this stage are to assess the flexibility, and speed, and gain insights into the potential solver to be utilised. Based upon a fictive organisation, the model and a set of testing instances will be set up. To validate the feasibility, the set of testing instances will be used to test its ability to find optimal solutions within a reasonable time frame. An essential part of this is the solver selection in which the speed of different instances is tested upon solvers like OR-Tools, HiGHS, and Gurobi.

3.2. Initial requirements

Based on the technical options from the CP-Model, the European AI Act, and (design) suggestions from relevant literature, the initial requirements will be set. These will be both technical requirements and requirements related to the process or flows.

3.3. Participatory design

The next phase involves participatory design, in which the research team and future end users collaborate in an iterative manner. At an early stage, both the requirements and the first visual design are discussed with internal domain experts from the hosting company. Later sessions are also conducted with external domain experts from both a variety of industries and organisational sizes. These informal sessions serve as an open platform to communicate their needs; due to time limitations, no surveys will be conducted. The iterative nature of these design sessions will allow for continuous improvement and fine-tuning of the software design.

3.4. Development

Following the participatory design phase, the next step is the actual software development. The goal of this phase is to create a working prototype based on the most recent visual designs. The software will be developed as a web application using widely accepted programming languages. The software development process will be conducted in two-week sprints in line with agile principles. When the prototype is finished, it will be extensively tested by the development team to minimise the number of bugs.

3.5. Final validation

In the final stage, the created prototype will be validated by domain experts in employee scheduling, these domain experts are actual customers of the hosting company. Validation involves creating an actual schedule in the prototype software and filling out a questionnaire.

The development team loaded the relevant company variables into the prototype before the start of the evaluation. The validation process is divided into two parts:

1. Usability validation: To assess ease of use and intuitiveness, users are asked to configure a real schedule for their own organisation by setting the hard and soft constraints.

2. Quality validation: The effectiveness of the CP-model is evaluated using a post-test questionnaire.

The post-test questionnaire reflects the following research questions on a scale 1 - 6, followed by an open field to explain their choice.

Questionnaire question	**RQ**
The tool explains well what the settings do and what impact changes have on the final schedule.	2
The tool explains well what the priorities with which you can steer and the impact of changing the priorities on the final schedule	2
The available settings are in line with the wishes of our organisation	MQ
The available priorities with which you can steer are in line with the wishes of our organisation	MQ
Comparing different configurations and the underlying grids is useful	2
The tool gives me sufficient confidence in efficient but also fair schedules	MQ
The schedule generated is of sufficient quality to use	MQ
Configuring the settings and priorities for my departments takes less than an hour	MQ
I would use this tool in my work	MQ

The mean research question (MQ) and sub-question 2 will be answered positively if all the external domain experts affirm the questions to be sufficient (4) or higher. The answers to sub-questions 1 and 3 will be formed during the development process and are discussed in Chapter 6.

3.6. Ethical considerations

The research will be conducted fully in line with the ethical guidelines of the TU-Delft. The Data Management Plan (DMP), Human Research Ethics Checklist (HREC), and the Informed Consent form have been approved by the Ethics Committee of the TU-Delft. All participants provided their informed consent before participating, and the data will be anonymised to protect their identity.

4 Development

4.1. Identification of user needs

Setting the user's needs is an important step in the design process. This ensures that the software is developed for intuitive use, efficiently works towards solving the problem, and takes the research questions into account. We clearly defined the user group in the target domain. This target group has been selected based on the characteristics of their employee schedule; The schedule must have at least one of the following properties:

- High variety in which individual employees are scheduled
- High variety in the demand of labour

This removes irrelevant departments, where the schedule is too constant, from the scope. Examples are office departments, as the employee schedule will be very constant. To narrow down the scope even further, the UI will focus solely on the person responsible for creating the employee schedule in the current manual process. Also, it will be assumed that schedules will be created for one location at a time. Allowing multiple locations will only introduce additional complexity and thus will not contribute to increasing the learning curve. In summary, the domain of the schedule is: a single location, with two or more departments and two or more employees.

After we had a clear understanding of our target audience, we moved on to understand their needs. Several internal interviews at the hosting company were conducted with domain experts to determine the main variables on which a schedule is currently created. Next, a review of existing scheduling solutions has been performed to discover which basic settings in existing general solutions have been implemented. Combining these will lead to a list of constraints, which must be present in the user interface.

Another critical aspect of identifying user needs was to get insight into the context in which the software would be used. This includes, for example, the devices used, the frequency of use, and other concurrent tasks that the user might be doing. Since the new approach is completely different from the manual way of scheduling, we hypothesised that the configuration will be more work compared to the manual approach. To speed up and scope the UI design process, the focus of the front end is on desktop screens only.

The upcoming European AI Act will have both direct and indirect implications on the user interface. According to the Act, high-risk AI systems, such as systems used for employee scheduling, must be transparent, tractable, and under human oversight. The AI Act describes the following concrete measures related to the front-end:

- the interface should be designed to clearly communicate the scheduling decisions
- the interface includes features that allow users to maintain oversight of the algorithm's operations
- users can oversee their functioning, measures should guarantee that the built-in operational constraints cannot be overridden
- it should be made clear to users that they are dealing with an AI system, but only if this is not obvious already
- users should remain aware of the possible tendency of automatically relying or overrelying on the output produced by the system (automation bias)
- the AI system shall be accompanied by instructions for appropriate use that include concise, complete, correct and clear information that is relevant, accessible and comprehensible to users
- users should fully understand the capacities, limitations and reasonably foreseeable misuse

The implications mentioned above, combined with the sub-research questions and the technical options from the created MiniZinc model, will make up the initial requirements. It is important to note that the previous steps are not a one-time process. As we move through the design iterations, the software evolves over time. We continuously reviewed and updated our understanding of user needs. User feedback in the regular domain was a critical component of this iterative process.

4.2. User flow

To support the user interface designer in understanding the initial requirements, a visual representation of the user activities, user tasks, and the underlying user stories has been created. The representation in Figure 4.1 is specifically for the employee scheduling problem, but most steps can be applied to other combinatorial optimisation problems.

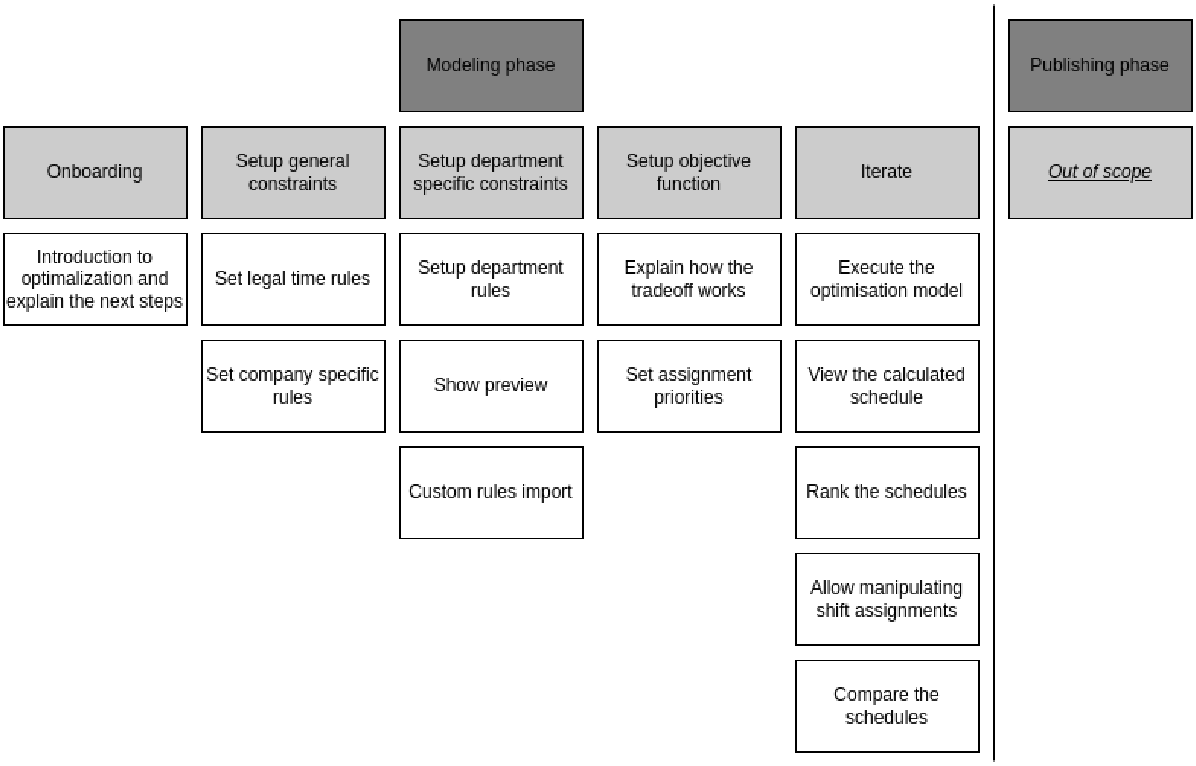

Figure 4.1: User flow

4.3. Visual designs

This section presents three versions of the interface design, which were developed in cooperation with the user interface designer. The intention was to learn from each design cycle by addressing the shortcomings, guided by feedback from domain experts in the subsequent iteration.

4.3.1. First design iteration: initial prototype

The initial design of the user interface was based on the basic features of the linear process presented in the user flow. The available inputs were based on the setup in the requirements. A two-panel layout was used, in which the user could input the necessary hard-constraint variables into the left panel and a preview would appear in the right panel accordingly.

Moreover, this design did not include elements which can answer the sub-research questions below:

- How can the UI deal with explaining the impact of changing constraints or the objective function on the final outcome?
- Can the interface comply with the EU AI Act?

4.3.2. Second design iteration: new approach

The second design of the user interface tried to solve the flaws of the first iteration by introducing a six-step linear flow in which the domain expert is supported in configuring the constraints. In addition, more advanced options for customising the schedule were added together with several onboarding widgets, explaining to future users what can be expected.

The prototype was evaluated using heuristic evaluation with internal feedback from domain experts at the hosting company. Heuristic evaluation is a usability inspection method in which a small set of expert evaluators assess the user interface to identify potential usability issues and areas for improvement. These individuals were unfamiliar with this project and had no further involvement. This internal feedback showed that:

- this initial design was too complicated and visually distracting
- the modern design is much more visually appealing compared to the first iteration
- not displaying the schedule grid during the configuration was not received well as it made the settings very abstract
- order of the settings was hard to understand, as this diverges from the current manual approach

No improvements have been made related to the two sub-research questions mentioned in the initial prototype.

4.3.3. Third design iteration: refinement and final prototype

Before the third iteration of the design started, several days had been spent trying to understand the mental model of the domain users. A mental model is the future user's perspective and understanding of how a product or system should operate [Carroll and Olson, 1988]. This is based on their previous experiences and learning. The goal of getting these insights was to align the preconceived expectations of the domain users and to reduce the learning curve. Next to this, the hard constraints which need to be presented in the UI have been translated into textual 'conversations' between the domain user and the UI. This with the goal of getting insights into all the different scenarios that can occur related to the hard constraints. All the requirements and constraints have been included in the confidential part of the Appendix.

The aligned mental model, conversations and feedback from the second iteration have been used to set up the third design. This design was much more comprehensive and, therefore, took several weeks to design. Based on the feedback received, we:

- changed the language from English to Dutch
- copied the terms from the existing manual process
- re-introduced the scheduling grid throughout the entire process

- removed the concept of a linear process and allowed a more 'free form' while configuring
- use real historical data for the preloading part of the configuration
- include the working hours act, and deviate in favour of the employee
- clarified the difference between soft and hard constraints
- clarified the assumptions made based on the imported employee data
- remove the setting: prioritize this department
- added an overview option, for comparing configurations

After careful consideration, we decided not to implement the following feedback:

- Forecasting option in which the number of employees per hour can be estimated: although useful, this is out of scope considering the scalability across industries and time limitations.
- Allow flexible planning intervals of 60, 30 or 15 minutes: would be useful in the real world, it is not worth changing it at this time.
- Add opening hours in the layout: this is the old way of thinking, to support the learning curve, we left this out.
- Option for custom boundaries, like 'only three-weekend shifts per 8 weeks': the scope of the prototype is a one-week schedule.

Concerning sub-research question 2, we found that it is hard to show a preview in which the changes in the constraint or the objective function are reflected. Due to the complexity of the problem, it is difficult to generate a preview of the outcome promptly.

The designs for the third iteration have been validated both internally and, for the first time in this project, also validated with experts in the external domain. As with the initial design, the second phase incorporated user testing with heuristic evaluation. These evaluations have been conducted with the same group of internal domain experts at the hosting company. For the external experts, we visited several companies and showed them the concepts of the visual designs. The feedback of both the internal and external experts was aligned, below is a summary of the feedback:

- at first glance, not sure how and where to start when
- unclear in which units the bars and graphs are
- ethical warning note is unclear in some settings
- the term 'AI' seems to be distracting and sets false expectations

- unclear which data/variables are pre-imported
- the design looks very fresh and simple
- displaying the scheduling horizontally is an improvement compared to the current approach
- all experts would like to see it in action to come up with additional feedback

The domain expert feedback has been taken into account and implemented in the final visual designs. The final validation, based on the working proof of concept, will be discussed in Chapter 5: Experimental Results.

4.4. Technical setup

The next step is the actual software development; in this phase, the software developer develops the prototype based on the final visual designs. This phase had to be completed in a maximum of 5 weeks.

Due to the hard deadline in the external domain-user validation sessions, some features which had been included in the visual designs have not been incorporated in the prototype. These not developed features are listed below, and were selected based upon the trade-off on time necessary for development and missed impact:

- Onboarding widgets for first-time users: this will be done verbally during the validation sessions
- Day-requirements, which allow for selecting employees with specific skills or characteristics like safety training or possession of keys: too much work, we rather had other constraints included
- Custom soft-constraints: out of scope, since the domain experts won't be able to program this

4.4.1. Front-end setup

To speed up the development process, the design is focused on a desktop-only user interface, specifically for screens with a resolution of 1920 x 1080. The Vue.js front-end framework was used in combination with PrimeVue to build the user interface based on existing components. Existing libraries Chart.js and Sakai were used for data visualisation. For both testing and development and final validation purposes, the application was built to operate in a local environment only.

4.4.2. Back-end setup

The back-end system is built on a Node.js environment, which functions as a bridge between the front-end HTTP requests and the execution of MiniZinc models. Node.js was used primarily to manage incoming HTTP requests from the front end, creating an asynchronous, event-driven infrastructure that handles the client-server interactions.

To load the details of the organisation's employees' three JSON files will be exported from the scheduling software of the hosting company. These files contain:

1. The locations subgroups and departments
2. Employee details like availability, contract details or the departments in which the employee can work in
3. A historical schedule of a representative week

The constraints defined within the front end are systematically extracted using JavaScript, which then transfers these constraints to a DataZinc file (.dzn). This file serves as the input for our MiniZinc model. Depending on the data received from the front end, additional constraints for department-specific requirements are dynamically created and injected into the existing DataZinc model. This flexible design allows for a wide selection of constraints without the need for major alterations to the core model.

After the DataZinc files are compiled, Node.js proceeds to execute a command-line interface (CLI) command to the MiniZinc software. When the model is solved, producing an output in JSON format, this response is then returned directly back to the front-end without any modifications, maintaining a 1-1 mapping between the MiniZinc output and the front-end. This direct transmission ensures the accuracy and reliability of the results presented.

4.4.3. MiniZinc model

The foundation of the project lies within a MiniZinc model that was built well before the design phase. This preliminary model was developed based on a solid understanding of the fundamental principles of the employee scheduling problem. It offered a conceptual blueprint on which to refine and build further.

For post-validation of the model solution accuracy, we adopted an iterative approach to enhance and fine-tune the model. Each iteration consisted of executing the model, reviewing the results, making the necessary adjustments to the constraints or the decision variables, and reevaluating. This continuous cycle allowed us to identify potential weak points in the model, optimise its performance, and ultimately improve its accuracy in finding solutions to the defined problem. Also, see the subsection ”Performance Improvements” below.

Hybrid model

To allow for a generic solution, which will be suitable for a range of industries and organisation sizes, the model has been set up using two principles:

1. The DataZinc file, with all the customer-specific variables and also including inputs for all non-department-specific constraints.

2. The MiniZinc model, including a solid base and a dynamic component, allows for optional injection of department-specific constraints.

Performance improvements

The first versions of the model were functional but operated at an unacceptable slow pace. The challenge of proving optimality, an integral part of our objective, was a task that significantly slowed the execution time. Several improvements, see below, had to be made to allow usable results on time.

We found that symmetry issues, often a common problem in constraint programming, did not seem to make a noticeable dent in our model performance. A significant breakthrough occurred when the profile compilation option was explored within the MiniZinc IDE. This feature allowed us to dissect the model's operation and identify bottlenecks contributing to the slow performance. By incorporating global constraints and combining existing constraints, it reduced the complexity of the problem and the search space, leading to dramatic speed improvements.

A significant part of our success is due to Jip Dekker from Monash University. His advice and guidance provided the direction needed to accelerate the model. His insights informed our strategy, helping us to reduce runtime and make the project much more efficient.

Diversity of solutions

Diversity in optimisation problems refers to the availability of multiple distinct solutions that satisfy the constraints and the objective function. The emphasis on diversity provides a range of optimal solutions, each slightly different but equally valid, considering the objective function. This concept is relevant for employee scheduling problems, as an optimal schedule is not just about meeting objectives but also accommodating employee preferences, balancing workloads and ensuring fairness.

We aimed to implement this diversity using a framework provided by Linnea Stjerna from KTH Royal Institute of Technology. Her framework served as an excellent starting point, helping us to integrate the concept of diversity into our model and exploration process. Our initial tests with the diversity framework showed promise. We were able to generate a range of three distinct solutions which were in different contexts. However, when it came to applying this to real-world instances, the execution time soared, rendering it impractical for operational use.

Given the unfeasible execution times, we decided to put the concept of diversity on hold for now, focussing on improving the performance of the model before revisiting this promising approach. While diversity of solutions remains an intriguing idea for future development.

Solver configuration

Selecting the appropriate solver is a crucial aspect of the process in constraint programming. Different solvers have varying degrees of efficacy in dealing with particular types of problems, and some can handle certain constraints or objectives more effectively than others.

In this project, we compared three distinct solvers: Google OR Tools, Gurobi, and HiGHS. Each of these solvers comes with its own set of strengths, but in terms of speed, Gurobi outperformed the other two, see the table below. Gurobi demonstrated significant speed advantages, making it the most efficient tool for solving our problem. The choice of solver was only the first step. Gurobi's performance was further enhanced by fine-tuning the solver parameters. In particular, adjusting the MIPgap helped to control the quality of the solutions, and thus speed up the solver. At the same time, we set Gurobi to use only one thread, which improved performance on certain problems. Finally, we implemented a time limit of 5 minutes for the solver. This parameter helped avoid spending an excessive amount of time on exceedingly complex instances, thereby ensuring that the solver returned a feasible solution within a reasonable time frame.

Model	Solver	Solver setting	Avg time(s)	Best found
Restaurant	Gurobi 10.0.1	Threads: 1	1800	43
Restaurant	Gurobi 10.0.1	Threads: 4	1800	53
Restaurant	OR-Tools 9.6	Threads: 1	1800	3383
Restaurant	OR-Tools 9.6	Threads: 4	1800	229
Restaurant	HiGHS 1.5.1	Threads: 1	1800	-
Restaurant	HiGHS 1.5.1	Threads: 4	1800	-
Retail	Gurobi 10.0.1	Threads: 1	64	453
Retail	Gurobi 10.0.1	Threads: 4	48	453
Retail	OR-Tools 9.6	Threads: 1	1800	467
Retail	OR-Tools 9.6	Threads: 4	411	453
Retail	HiGHS 1.5.1	Threads: 1	1800	-
Retail	HiGHS 1.5.1	Threads: 4	51	453
Warehousing	Gurobi 10.0.1	Threads: 1	18	1415
Warehousing	Gurobi 10.0.1	Threads: 4	14	1415
Warehousing	OR-Tools 9.6	Threads: 1	28	1415
Warehousing	OR-Tools 9.6	Threads: 4	12	1415
Warehousing	HiGHS 1.5.1	Threads: 1	168	1415
Warehousing	HiGHS 1.5.1	Threads: 4	142	1415

4.5. Fine-tuning of prototype

Upon completion of developing the working prototype, a validation at the hosting company with internal domain experts. The feedback we received indicated that the prototype needed some textual and visual adjustments. Textual changes mainly involved rewording certain explanations and instructions to make them clearer and more concise. In terms of visual changes, the following adjustments have been made:

- Change line chart to bar chart
- Add title of the axis in the chart
- Sort the employees alphabetically
- Round the imported historical times to the closest hour
- Make the loading icon more prominent
- Added a hidden priority in the model to prioritize departments with one or two employees

5 Experimental results

In this chapter, a detailed analysis of the results derived from the prototype testing is provided. The sessions included domain experts from three different industries. The objective was to explore the usability, utility, and effectiveness of the tool designed in real-world settings and to obtain tangible data to support efficiency. Not all the results have a direct link with the research questions, but are still relevant for potential future research.

5.1. Context and objectives

The testing process involved three different organisations, each from a unique industry: the warehousing sector (Org1), retail (Org2), and restaurant (Org3). These organisations were chosen to ensure a broad spectrum of application scenarios and a diversity of requirements. All selected organisations had between 100 and 500 employees. The domain experts participating in the session within these organisations are all involved in the scheduling process. In every session, two or three domain experts were present, during the sessions they were tasked as a group while testing the prototype.

5.2. Experimental protocol

The validation sessions were set up in the same structured way. First, each organisation gave an introduction to automatic scheduling and onboarding of the prototype, including the benefits and potential pitfalls. After this, it was made clear that the prototype is not the final version, and we want to validate the broader configuration process and the degree of applicability for their organisation. The domain experts were then given a tutorial on how to use the tool, followed by a handover in which the domain experts were challenged to configure all the settings for the organisation themselves. To find potential shortcomings in the prototype, no questions could be asked during the configuration process. After creating and generating several configurations and schedules, they were asked to fill out a 10-question survey.

5.3. Result analysis

5.3.1. Warehousing (org 1)

The organisation operates in the warehousing sector and employs more than 500 individuals at different locations in the same city. The employees come from a variety of backgrounds, highlighting its diverse workforce. A key operational characteristic is a strict adhesion to fixed timestamps for the departure of trucks and the starting times of shifts, which streamlines logistics and ensures consistency in operations. However, this regimented schedule also results in employees having to work long hours on peak days to manage the increased demand and to maintain the organisation's commitment to timely deliveries. During these situations, third-party temporary workers can be hired. This scenario illustrates the challenges and demands that organisations face in the warehousing sector, balancing operational efficiency and employee satisfaction.

Org1 domain experts reported that the prototype is easy to understand and configure after an introduction. The prototype expects to provide significant efficiency improvements over existing methods, and experts particularly praise the capability to handle complex scheduling problems. However, some minor missing functionalities, as well as the slow speed of solving, were pointed out. Also, it was pointed out that the configuration of the constraints should be done by the HR department, rather than by the employees who are responsible for the current manual scheduling process. The reason they mentioned is questionable whether these employees are still needed in the future.

5.3.2. Retail (org 2)

The organisation in focus is a medium-sized retail entity employing a total of 50 personnel, including fixed-contract employees and flexible student workers. The workload is fairly consistent week-by-week, except for public holidays, during which the student workforce is ramped up to manage the increased demand, while the fixed contract employees see a reduction in their working hours. This helps to manage the workload effectively and also provides a much-needed respite to the permanent employees during holidays. The organisation operates with a flexible work schedule that allows employees to choose their starting and ending times, with shift lengths varying between 5 and 9 hours. This flexibility is instrumental in accommodating the needs of the student workforce, whose availability can vary depending on their other commitments.

In Org 2, the domain experts found that the learning curve is steep; however, the entire configuration is thought out and easy to use when it is understood. While they acknowledged the potential, they suggested incorporating more domain-specific priorities / soft constraints for better usability in the retail sector. For future versions, they recommended looking into the option to predict the workload of departments rather than relying on the experience of the scheduler to set the number of employees per hour for every department.

5.3.3. Grand Café (org 3)

The Grand Café considered is a renowned establishment that employs approximately 100 individuals, mainly flexible students. The workload at the Grand Café varies considerably from day to day and week to week, with weather being the most significant variable that affects customer inflow and, consequently, the staff's workload. As a result, starting and ending times for the employees are flexible, with shift lengths typically ranging between six and ten hours, which allows the restaurant to operate efficiently while accommodating the students' schedules. This dynamic work environment demands a well-coordinated and flexible workforce, capable of adapting to fluctuating demands while providing exceptional service to customers.

The representative of the restaurant sector found that the prototype is highly effective in optimising the workload. The new approach towards scheduling was received with much interest, although hard to understand at first glance. They suggested implementing additional priorities / soft constraints since the prototype result contained unnecessary department switches. They expect an impressive efficiency improvement, however, need to use it in a real-life scenario to confirm this.

5.4. Feedback and suggestions

In general, the prototype was well received in the three industries. Domain experts value the potential to solve employee scheduling problems efficiently. However, they also provided constructive feedback on the importance of any potential future in-app onboarding and additional industry-specific features. These feedback points are invaluable for future potential refinements and improvements to the tool.

The prototype showed significant promise in configuring company-specific problems in various industries into a CP model that can be used to improve efficiency. The next chapter will provide a thorough discussion of these results in relation to the research objectives and the existing literature in the field.

Results of the survey questions:

1. The tool explains the priorities with which you influence and the impact of changing the priorities on the final schedule well

- Warehousing: *6/6; The examples in the introduction help to understand how the priorities work*
- Retail: *6/6; No elaboration provided*
- Restaurant: *5/6; Also a matter of using and testing it yourself*

2. The available settings are in line with the wishes of our organisation

- Warehousing: *4/6; Hard to say at this moment, does give insights to efficiently schedule*
- Retail: *5/6; Would be interesting to include revenue insights in future versions*
- Restaurant: *5/6; Switching departments not desirable*

3. Comparing different configurations and the underlying grids is useful

- Warehousing: *4/6; Do not really understand the value added*
- Retail: *6/6; Yes, this allows us to compare the different configurations, maybe also add costs in the future*
- Restaurant: *4/6; Hard to say at this moment*

4. The available priorities with which you can steer are in line with the wishes of our organisation

- Warehousing: *5/6; Absolutely, if all the settings are correct*
- Retail: *6/6; No elaboration provided*
- Restaurant: *4/6; Trust yes, but real experience is still missing*

6. The tool gives me sufficient confidence in efficient but also fair schedules

- Warehousing: *5/6; Absolutely, if all settings are correct*
- Retail: *6/6; No elaboration provided*
- Restaurant: *4/6; Trust yes, real experience is still missing*

7. The schedule generated is of sufficient quality to use
   - Warehousing: *5/6; Seems like it based on this session*
   - Retail: *6/6; No elaboration provided*
   - Restaurant: *5/6; Looks good so far*

8. Configuring the settings and priorities for my departments takes less than an hour
   - Warehousing: *4/6; Not sure if this is the case for big departments*
   - Retail: *6/6; No elaboration provided*
   - Restaurant: *6/6; Saves a lot of time*

9. I would use this tool in my work
   - Warehousing: *5/6; Based on today, yes*
   - Retail: *6/6; You have to get used to it, but once you understand it this will save a lot of time and create better schedules*
   - Restaurant: *5/6; I assume that I will use it*

Summary of the survey questions:

Question	**Min**	**Max**[1]	**Avg**
1. The tool explains well what the settings do and what impact changes have on the final schedule	5	5	5
2. The tool explains well the priorities with which you influence and the impact of changing the priorities on the final schedule	5	6	5.7
3. The available settings are in line with the wishes of our organisation	4	5	4.7
4. The available priorities with which you can steer are in line with the wishes of our organisation	5	5	5
5. Comparing different configurations and the underlying grids is useful	4	6	4.7
6. The tool gives me sufficient confidence in efficient but also fair schedules	4	6	5
7. The schedule generated is of sufficient quality to use	5	6	5.3
8. Configuring the settings and priorities for my departments takes less than an hour	4	6	5.3
9. I would use this tool in my work	5	6	5.3

[1]Scale from 1 - 6

6
Conclusions and future work

This chapter provides an overview of the contributions of the project. Furthermore, a reflection on the results, conclusions, and recommendations will be presented. Finally, ideas for future work will be discussed.

6.1. Conclusions

6.1.1. Main research question

The main research question of this thesis was formulated as follows: Is it possible to create an interface for configuring a CP model for organisation-specific rostering problems in a general way where the user interface will explain the impact of constraint and objective function to non-expert users?

This goal will be reached if:

- Training beforehand will not exceed 2 hours
- Modelling should take max 1 hour to get to a workable outcome (customer input like contracts, and availability need to be done beforehand; everything which needs to be in place for manual schedule)
- Actual customers (domain-users) say they will use it

This thesis proposes a flexible interface that is connected to a developed hybrid CP model. A requirement study has been conducted to get an overview of the soft- and hard constraints that have to be taken into consideration for a generalised model, i.e., labour laws, contractual agreements, department-specific constraints, and employee preferences. These constraints have been modelled into a MiniZinc CP model to check technical feasibility. In the subsequent phase, the interface has been designed in an iterative manner, which has been validated and improved with both internal domain experts from the hosting company and external domain experts. The development of a working interface and two-way integration with the CP model and subsequent internal validation to the final phase, in which the prototype has been validated with external domain experts from various industries.

We can conclude that this prototype performed effectively in the organisations where it was tested, using real-world datasets taken from existing software systems and their own organisation-specific constraints. This is validated for external domain experts in the field of employee scheduling, the survey results confirm that the goal was achieved.

6.1.2. How to show the preview of changing constraints to the user?

To answer this sub-question, we first need to consider how the prototype is designed to manage the constraints. The iterative design process conducted showed that an effective user interface lies in the ability to simplify the process of setting up and adjusting the constraints. This involves proper onboarding, intuitive design, and a clear explanation of the functionalities. However, it is important to note that while the configuration process can be designed to be easy to understand, it is not always feasible to provide a preview of the results within a reasonable time due to the complexity of the problem. In theory, there are certain constraints for which a fast preview could be possible; in practise these constraints are usually among the simpler ones; e.g. when a shift must be between 5 and 8 hours, the only possible outcomes are 5, 6, 7, or 8-hour shift lengths. As a result, the impact of providing a preview for these particular constraints would be minimal.

The domain expert feedback has shown that even with the absence of a preview feature, they are still able to understand the implications of changing constraints. This suggests that while a preview feature might be useful in some instances, it is not a critical component for understanding. The ease of configuration and user adaptability seem to compensate for the lack of a direct preview.

In conclusion, despite the intuitive appeal of offering a preview feature of changing constraints, it is not always practical, beneficial, or adding value. Rather, it should focus on designing a simple, logical, and effective user interface. The current user interface design is sufficient for its purpose; however, there could be added value to the preview functionality for some constraints in potential future versions.

6.1.3. How can the user interface deal with explaining the impact of changing constraints or the objective function on the final outcome?

These questions should be answered in relation to the iterative process rather than individual results. The domain expert needs to create and run several different configurations in which they alter the constraints, while the input is kept consistent. This setup allows one to assess the influence of different constraints on the same input scenario. The consistent improvement of constraint values, driven by the feedback from the comparison of outcomes, leads to the enhancement of constraints' input values. The prototype provides a summary screen in which key metrics of the individual schedules can be compared, providing an indirect way to visualise the impact of changes, giving deeper insights, and enabling a more nuanced understanding of the constraint programming process. This is formally confirmed by domain experts, as the survey shows that both the explanation of constraints and the summary functionality have a positive impact on the final outcome.

6.1.4. Can the interface comply with the EU AI Act?

When evaluating the compliance of the tool's interface with the EU AI Act, it is essential to specify the aspects of the tool we are considering. In this case, our assessment is limited to the front end of the tool, without including the back end or the associated procedures. Legally speaking, it remains a subjective matter whether the interface genuinely complies with the EU AI Act. Ultimately, a definitive conclusion can only be reached by a legal authority or a judge, who can assess its alignment with the Act in a legal context.

However, based on our preliminary assessment:

- The essential requirements of the EU AI Act as detailed in section 4.1 have been considered and incorporated. These elements can be found either directly within the front-end interface or in the introduction/onboarding provided to users before their interaction with the tool.

- Our confidence in this assessment is confirmed by feedback from domain experts. The survey involving these experts confirmed that they observed the presence of the aforementioned bullet points, aligning the front end with the stipulations of the Act.

In conclusion, while only a legal authority can definitively confirm compliance with the EU AI Act, preliminary indicators suggest that the front-end interface of the tool aligns with the bullet points stated in section 4.1, as confirmed by domain experts.

6.2. Discussion and reflection

Central to this research is the investigation of the possibility of creating an interface to configure a constraint programming model for organisation-specific rostering problems. The primary objective was to find out whether the user interface could appropriately make the impact of constraints and the objective function clear to users without specialised knowledge in this area.

The results obtained through the evaluation of the working prototype by domain experts were promising. These experts have indicated that it is possible to create such an interface. Their evaluation not only confirmed the theoretical feasibility of the proposed interface but also provided valuable feedback to refine it further.

For the organisations involved in the validation, the significance of these findings cannot be overstated. By demonstrating the capability to develop such an interface, Dyflexis B.V., the hosting company, possesses the potential to implement it independently. The implementation of this could have a direct, major effect on the lives of hundreds of thousands of people in the Netherlands. On a broader level, this research has a tangible impact with far-reaching consequences on the accessibility of advanced optimisation methods. Lessons learnt during the execution of this investigation can be used to develop similar optimisation tools.

However, like all studies, this research is not without limitations. Some of the notable limitations include the following:

- The actual schedules generated using the CP model interface were not implemented in real-world scenarios. Therefore, the practical feasibility and potential issues that arise from these schedules remain uncharted.
- Related company processes, such as allowances that could affect the final roster, were not included in the study. This exclusion might lead to a divergence between theoretical and practical outcomes.
- The study focuses primarily on labour that requires immediate deployment. Hence, there might be room for improvement in types of rostering problems in which the labour has a broader deadline.
- The research did not fully account for the entire AI Act of the EU. Given the widespread influence and legal implications of the Act, this could pose challenges in the broader implementation of the tool.

The next steps to further improve the current interface could be:

- User Onboarding: As the tool is designed for non-experts, the next crucial step would be to develop an intuitive user onboarding process, as this is currently done in person.
- Integration with existing IT Infrastructure: To make the tool more adaptable and functional, it is essential to integrate it more into the existing IT landscape. This would ensure seamless data exchange, increased usability, and overall coherence with other systems in use.
- Incorporation with business intelligence (BI) tooling: To maximise the full potential of the tool, integrating it with BI tools can provide invaluable insights and improved analytical capabilities.

In conclusion, while this research has paved the way for a revolutionary tool that has the potential to impact many lives, the journey ahead, filled with refinements, integrations, and legal compliances, promises even more advancements in the domain of rostering.

6.3. Directions of future research

Improving solving speed: One of the main objectives of any optimisation solution is efficiency. Although the current design offers satisfactory performance, there is, especially in large instances, room for improvement. Future work can be orientated towards streamlining the model, refining search heuristics, or exploring parallel and distributed computing methods to increase the solving speed as currently only one thread was used. Speed improvements would greatly enhance the applicability of the solution in real-time scenarios where quick decisions are paramount.

Diversity of solutions: Although this research has laid the foundation for generating optimal solutions, it is crucial to consider the diversity of these solutions, especially in cases where multiple near-optimal solutions are acceptable. Expanding research to ensure a richer diversity of solutions could provide users with a broader spectrum of choices, catering to various scenarios and preferences.

Interface re-usability: The designed interface, with its robustness and flexibility, can serve as a blueprint for other combinatorial optimisation problems. Elements of this interface, including visualisation tools, decision support mechanisms, and user interaction modules, can be repurposed and customised to fit other similar problem domains. Such reusability can expedite the development process for subsequent projects and ensure a consistent user experience.

Highly custom constraints based on natural language: A groundbreaking area for future research is the translation of natural language or typed explanations by domain experts into actionable soft or hard constraints. By developing a system that can interpret and 'codify' domain expert insights and preferences directly, the optimisation process can become even more adaptive.

7

Informal advice for comparable research

This section presents practical, non-scientific advice for future researchers interested in making optimisation algorithms accessible to domain experts. Through a combination of personal experience and informal insights, this section aims to provide tips which can help to make the process more efficient. Looking back, I wish someone had given me this advice at the beginning of this thesis.

Change management. Introducing a new workflow to the domain experts often requires a change in their existing workflow. From the experiences in this thesis, it is essential to clearly understand what steps of the process remain the same, will be removed and are new. Domain experts have worked with the same process for many years, and some of these steps are covered by the algorithm now. You need to understand this has an impact. In the first design iteration, we made the mistake of not introducing the new steps of the scheduling process. For users without any knowledge about the new process, this is confusing. In our situation, users mainly struggled to understand that scheduling is no longer about the individual employees, but the number of employees required with a specific skill at a specific timeframe.

Preload data inputs. Start by preloading data from their existing systems into the interface; this allows the users to shift their time and attention to understanding the new process. For example: we extracted the amount of employees required per timeframe from a historical schedule. If a specific variable was not present in the old process, try to estimate the value from the old process.

Visualize. Your algorithm is still a complicated piece of software which has many variables. Providing a soft landing is very important, as there is only one chance to leave a first impression. By presenting both the algorithmic inputs and outputs in an intuitive, visual format, you help domain experts understand how the optimisation algorithm behaves. This also helps in fine-tuning it to suit their specific needs. While designing the interface, try to ignore existing visualisations, as this might have evolved inefficiently over time. We found that the visualisation of comparing multiple generated outcomes is very useful.

Not a magical box. Set realistic expectations right from the start. Although optimisation algorithms can significantly enhance decision-making and efficiency, they are not infallible or magical solutions. Educate the domain experts on the nature of these tools: they require ongoing input, refinement, and understanding. Clarify that initial results might not be perfect and that the true value of these algorithms is realized through iterative improvements and sustained engagement.

Plan for iterations. Prepare for multiple iterations, both in terms of the algorithm's interface and its underlying parameters. The first version is rarely the final one. Gather feedback regularly from the domain experts and be open to making adjustments. This iterative process is crucial for fine-tuning the algorithm to meet the specific needs and preferences of its users. Schedule these iterations as part of the implementation plan, highlighting that each round brings the tool closer to its optimal form and functionality.

Bibliography

Borissova, D., & Mustakerov, I. (2016). A framework for designing of optimization software tools by commercial api implementation. *Int. Journal of Advanced Engineering, Management and Science*, *2*, 1790–1795.

Carroll, J. M., & Olson, J. R. (1988). Mental models in human-computer interaction. *Handbook of human-computer interaction*, 45–65. https://doi.org/10.1016/B978-0-444-70536-5.50007-5

Dantzig, G. B. (1954). A comment on edie's "traffic delays at toll booths". *Journal of the Operations Research Society of America*, *2*(3), 339–341. https://doi.org/10.1287/opre.2.3.339

Dzindolet, M. T., Peterson, S. A., Pomranky, R. A., Pierce, L. G., & Beck, H. P. (2003). The role of trust in automation reliance. *International Journal of Human-Computer Studies*, *58*(6), 697–718. https://doi.org/10.1016/s1071-5819(03)00038-7

EP. (2023). Regulation of the european parliament and of the council laying down harmonised rules on artificial intelligence (artificial intelligence act) and amending certain union legislative acts.

Ernst, A., Jiang, H., Krishnamoorthy, M., & Sier, D. (2004). Staff scheduling and rostering: A review of applications, methods and models [Timetabling and Rostering]. *European Journal of Operational Research*, *153*(1), 3–27. https://doi.org/10.1016/S0377-2217(03)00095-X

Glass, A., McGuinness, D. L., & Wolverton, M. (2008). Toward establishing trust in adaptive agents. *Proceedings of the 13th International Conference on Intelligent User Interfaces*, 227–236. https://doi.org/10.1145/1378773.1378804

Goodwin, S., Mears, C., Dwyer, T., de la Banda, M. G., Tack, G., & Wallace, M. (2017). What do constraint programming users want to see? exploring the role of visualisation in profiling of models and search. *IEEE Transactions on Visualization and Computer Graphics*, *23*(1), 281–290. https://doi.org/10.1109/TVCG.2016.2598545

Hardin, R. (2006). *Trust: Key concepts*. Polity Press Cambridge. https://doi.org/10.1007/s11615-003-0046-8

Herlocker, J. L., Konstan, J. A., & Riedl, J. (2000). Explaining collaborative filtering recommendations. *Proceedings of the 2000 ACM conference on Computer supported cooperative work*. https://doi.org/10.1145/358916.358995

Hoff, K. A., & Bashir, M. (2014). Trust in automation. *Human Factors: The Journal of the Human Factors and Ergonomics Society*, *57*(3), 407–434. https://doi.org/10.1177/0018720814547570

Howell, I., Woodward, R., Choueiry, B. Y., & Bessiere, C. (2018). Solving sudoku with consistency: A visual and interactive approach. *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI-18*, 5829–5831. https://doi.org/10.24963/ijcai.2018/852

Liu, J., Dwyer, T., Tack, G., Gratzl, S., & Marriott, K. (2020). Supporting the problem-solving loop: Designing highly interactive optimisation systems. https://doi.org/10.48550/arXiv.2009.03163

Liu, J., Marriott, K., Dwyer, T., & Tack, G. (2022). Increasing user trust in optimisation through feedback and interaction [Just Accepted]. *ACM Trans. Comput.-Hum. Interact.* https://doi.org/10.1145/3503461

Madhavan, P., & Wiegmann, D. A. (2007). Similarities and differences between human–human and human–automation trust: An integrative review. *Theoretical Issues in Ergonomics Science*, *8*(4), 277–301. https://doi.org/10.1080/14639220500337708

Mak, A. B. (2007). *Constraint graph visualization*. https://api.semanticscholar.org/CorpusID:10546951

Mayer, R. C., Davis, J. H., & Schoorman, F. D. (1995). An integrative model of organizational trust. *The Academy of Management Review*, *20*(3), 709. https://doi.org/10.2307/258792

Meignan, D., Knust, S., Frayret, J.-M., Pesant, G., & Gaud, N. (2015). A review and taxonomy of interactive optimization methods in operations research. *ACM Transactions on Interactive Intelligent Systems*, *5*(3), 1–43. https://doi.org/10.1145/2808234

Nadel, A., Ryvchin, V., & Strichman, O. (2013). Efficient mus extraction with resolution. *2013 Formal Methods in Computer-Aided Design*, 197–200. https://doi.org/10.1109/FMCAD.2013.6679410

Naish, L. (1997). Oz explorer: A visual constraint programming tool. In *Logic programming: The 14th international conference* (pp. 286–300). https://doi.org/10.1007/3-540-61756-6_108

Oglesby, J. M., Stowers, K., Leyva, K., Dietz, A., Sonesh, S., Burke, S., & Salas, E. (2014). Assessing human-automation system safety, efficiency, and performance. *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, *58*(1), 1149–1153. https://doi.org/10.1177/1541931214581240

Parasuraman, R., & Riley, V. (1997). Humans and automation: Use, misuse, disuse, abuse. *Human Factors: The Journal of the Human Factors and Ergonomics Society*, *39*(2), 230–253. https://doi.org/10.1518/001872097778543886

Simonis, H., & Aggoun, A. (2000). Search-tree visualisation. In P. Deransart, M. V. Hermenegildo, & J. Małuszynski (Eds.), *Analysis and visualization tools for constraint programming: Constraint debugging* (pp. 191–208). Springer Berlin Heidelberg. https://doi.org/10.1007/10722311_8

Simonis, H., Aggoun, A., Beldiceanu, N., & Bourreau, E. (2000). Complex constraint abstraction: Global constraint visualisation. In P. Deransart, M. V. Hermenegildo, & J. Małuszynski (Eds.), *Analysis and visualization tools for constraint programming: Constraint de-*

bugging (pp. 299–317). Springer Berlin Heidelberg. https://doi.org/10.1007/10722311_13

Simonis, H., Davern, P., Feldman, J., Mehta, D., Quesada, L., & Carlsson, M. (2010). A generic visualization platform for cp. In D. Cohen (Ed.), *Principles and practice of constraint programming – cp 2010* (pp. 460–474). Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-642-15396-9_37

Thompson, G. M., & Pullman, M. E. (2007). ”scheduling workforce relief breaks in advance versus in real-time”. *”European Journal of Operational Research”*, *181*(1), 139–155. https://doi.org/10.1016/j.ejor.2006.05.018

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. https://doi.org/10.1016/j.ejor.2012.11.029

Wallace, M. (2020). *Building decision support systems: Using minizinc*. Springer International Publishing. https://doi.org/10.1007/978-3-030-41732-1

Zhu, X., Nacenta, M. A., Akgun, O., & Nightingale, P. (2020). How people visually represent discrete constraint problems. *IEEE Transactions on Visualization and Computer Graphics*, *26*(8), 2603–2619. https://doi.org/10.1109/tvcg.2019.2895085

A

Initial requirements

The list of initial requirements set by domain expert interviews, literature studies, and the EU AI Act can be found in the confidential version of the thesis.

B

Drafts of design iterations

The different designs iterations and final design can be found in the confidential version of the thesis.

C

Final MiniZinc Model

The final constraint programming model in MiniZinc can be found in the confidential version of this thesis.