UKRAINIAN CATHOLIC UNIVERSITY

MASTER THESIS

A multifactorial optimization of personnel scheduling in fleets of seagoing vessels

Author:
Oleksandr SMYRNOV

Supervisor:
Dr Rupert SMALL

A thesis submitted in fulfillment of the requirements for the degree of Master of Science

in the

Department of Computer Sciences
Faculty of Applied Sciences

Lviv 2020

Declaration of Authorship

I, Oleksandr SMYRNOV, declare that this thesis titled, "A multifactorial optimization of personnel scheduling in fleets of seagoing vessels" and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed: _____

Date: _____

UKRAINIAN CATHOLIC UNIVERSITY

Faculty of Applied Sciences

Master of Science

A multifactorial optimization of personnel scheduling in fleets of seagoing vessels

by Oleksandr SMYRNOV

Abstract

The maritime industry is huge and consists of a lot of complex processes. It is a consequence of the fact that the maritime industry provides most of the goods transportation. During transportation, people serve the vessel. And here the problem is raised of the optimal distribution of crew on vessels.

This problem can be solved by formalizing the integer programming problem. In practice, we saw that solving this problem is time-consuming, since there are a large number of free variables. This makes the solution inapplicable to the end-user.

In this work, we describe the approach to speed up a solution of crew optimization for the maritime industry using the Rolling Time Horizon technique.

Our approach is 3.5 times faster than the benchmark, and deviates from the optimal solution by less than 1%.

Acknowledgements

I am grateful to people who helped me with this project: Rupert Small, Andrii Rogovyi, and other members of 90POE, Oleksii Molchanovskyi, Oleksandr Zaytsev. Also special gratitude to Data Science squad in 90POE, especially to Rupert Small, who provided supervision and supported from the start till the end of the project.

I am grateful to Ukrainian Catholic University and Oleksii Molchanovskyi for the great Data Science Master's program. And I want to say thank you to the company "Ninety Percent of Everything" for giving me the opportunity to complete my Master's thesis with them.

Contents

List of Figures

List of Tables

Dedicated to my family, especially for grandfather-captain

Chapter 1

Introduction

1.1 Description of the problem

More than 90% of produce in the world is transported by vessels. Vessels are continuously moving from port to port, and people are serving them along the way. Current standards clearly describe the number of people and skills needed to service each vessel. At the same time, people have a different set of skills and live in different parts of the world. Accordingly, in large companies with a large number of vessels and employees the task emerges of optimally assigning crew to each vessel.

For each vessel, you need to recruit a team with the most suitable set of skills, while spending the least amount of money on moving the sailor from home to the vessel and back. For this, integer linear programming methods are used. Experiments have shown that in production, the solution of the corresponding integer linear programming problem can be found, but it takes a lot of time.

Because of this, a hypothesis has arisen about accelerating the search for solutions using the Rolling Time Horizon technique. This method does not consider the entire period of time (for example, a year) for which it is necessary to assign people optimally on vessels, but uses shorter time intervals (for example, months), considering that the intermediate stages lie on the way to a global optimum.

The goal of the project is to implement and check with real data whether the Rolling Time Horizon approach will provide acceleration in the search for the optimum, and investigate the deviation from the optimal solution for the maritime industry.

1.2 Outline of the work

After the introduction, in Chapter 2 - background description - we make a general overview of the maritime industry, where the crewing process is described and its features summarised.

Chapter 3 provides a description of related works: resource optimization and rolling time horizon approach usage.

In Chapter 4 we discuss approach and solution, and we made an overview of existing methods for solving optimization problems. After, we introduce our solution designed for the maritime industry.

In Chapter 5 - implementation and evaluation - we describe the working pipeline and compare the obtained results with the benchmark.

Chapter 2

Background

2.1 Types of vessels

Most of the products in the world are transported by sea: grain, oil, food, clothes, vehicles, chemicals, etc. For different products exist specific kind of vessels. For example, a tanker transports oil and fuel. This kind of vessel is designed specifically for fluids and has such differences with other vessel types: inside the ship, it has an inner tank with a doubled shell, where the fluid is filled (the reason is the risk of damage to the hull and cargo leakage).

FIGURE 2.1: Different types of vessels: left - tanker; central - bulk carrier; right - container ship
Source: *Different kind of vessels*

Grain is mostly transported by bulk carrier. A bulk carrier has a large hold, where the product (for example, grain) is filled. Most of the products like clothes, vehicles, electrical devices are transported by container ship. Refrigerated vessels transport food, and have refrigerators for storing perishable goods [United Nations, 2019].

So the maritime industry has a huge variety of ships which are designed in a different way and have different purposes.

2.2 Working on a vessel

Every vessel has a group of people that provides its functionality – the crew. The crew consists of the sailors with different ranks and duties: a captain, the first mate - the third mate, chief engineer, seaman, cook, etc. The number of crew members, positions, the experience and education of each person depends on the vessel type [Sekimizu, 2010].

For some types of vessels, there is a requirement of a large crew, such as cruise liner; for vessels that serve oil towers, the crew is smaller. But the general pipeline of the workers is the same, and it is different from usual work on the land. A sailor works for continues periods that usually 3-8 months. In such a working period, the sailor is transported on the vessel, and stays there till the end of the contract. The

sailor sleeps and lives on the ship, and have the only possibility to visit the land when the vessel is on the port (in most countries, sailors can visit the port-city without a visa). After the contract ends, the sailor is transported back to the land where the person has a rest till the next contract (depending on the contract, the person can work three months after three months rest or work on other conditions).

2.3 Crewing process

In the maritime industry exists the process called crewing. Crewing solves a similar problem as a recruiting – finding labor power for the employer. But it has a lot of differences because of maritime industry species. During the enrolling of the person for a job, such cases should be checked: requirements met by experience and education; ability to work under the vessel flag; forming documents for the job; organization of the sailor transfer from the home to the vessel. So the process is more complicated, and more factors should be included.

To transfer the sailor on the vessel, the company required to provide to sailor a transfer (most common – a flight) to the port where the sailor will join the crew, and accommodation in a hotel if needed (for example, when vessel did not come to the port in time). So this process is costly if the procedure performed not in an optimal way.

2.4 Crewing features

During education and passing the courses, sailors acquire licenses that show the ability of a person to perform some job. These licenses are international, so the documents from the different countries are comparable. Licenses are temporal, so the sailors should renew their skills from time to time.

Also, there are requirements for work experience for each particular role on particular vessel types for some jobs (for example, to become the first mate on a tanker, you need to work as the second mate on the tanker, the experience on bulk carrier can be irrelevant).

In the maritime industry, there are additional constraints that are unusual for common works, such as a person can not work on the vessel more than 11 months in a row; some countries do not hire people from Philippines, so they cannot be part of a crew under some flags. These additional constraints make the crewing process more complex.

Chapter 3

Related Works

3.1 Optimization of resources

The job scheduling problem is widely used in different fields: manufacturing (the most common one), scheduling of operating rooms at the hospital, energy systems, etc. The general problem seems similar for all of these cases – optimization of resources. But the specifics of jobs and workers influence the approach for the problem and its complexity. For example, the job might have a flexible schedule or fixed start and end time; the job may be stopped and performed after some time or not (even with another worker); jobs may be similar and permutable, or jobs complexity may vary, so that not all workers can perform all the jobs.

Workers specialisation also might be different: workers can be uniform or with different qualifications; workers might be able to work all the time or in some specific open ranges; workers might have restrictions on working time or not etc.

All of these factors are influencing the complexity of the optimization problem and affects the approach to solve the problem [Kroon, Salomon, and Van Wassenhove, 1995].

3.2 Types of optimization problems

For edge cases, the solution to the problem can be simple. For such problems when all setup consists of n workers and j jobs, which any workers can do, and each job has a value w_j, there exists a transformation from the problem of optimization to the problem of coloring interval graph. So we can find a solution in polynomial time [Arkin and Silverberg, 1987].

When exists relation of the order between the workers, so the worker $w_i < w_j$ means that worker w_j can perform all the jobs that worker w_i can do, but not vice versa, exist polynomial time second order approximation of the solution [Bhatia et al., 2003].

For other cases, scheduling jobs with fixed start and end with more than 1 type of jobs is NP-hard task (without considering trivial cases)[Kolen et al., 2007].

In the context of the maritime industry, not all the sailors have a similar experience, education, licenses, etc. So there exists a subset of jobs that require specific workers. Also, workers can not be arranged (there is no ordering between a chief officer and a chief engineer). This implies that the problem that we consider is the problem of the third type: NP-hard.

3.3 NP-hard problems

Since our problem is NP-hard, the common method to solve such optimization tasks is branch and bound method [Clausen, 1999]. The algorithm considers all possible solutions in the complete space to find the optimal one. The idea of the algorithm is providing two operations: branching and bounding. Branching means cutting the complete space into the some subset; bounding – finding maximal and minimal solution of the problem for the branch. If in minimization problem some branch has minimal value higher than maximal in another branch, it means that the first branch could be eliminated from the consideration.

Since the branch and bound method is modification of the brute force, it takes a lot of time to find a optimal solution. It also was confirmed in practice.

3.4 Decomposition of integer linear programming problems

The research was aimed at speeding up the existing the solution. For such cases, decomposition methods could be used. There exist a lot of methods that provide decomposition of optimization task: Rolling Time Horizon, Bender decomposition, Lagrangian decomposition, bi-level decomposition. These approaches enable one to solve not the whole problem, but divide it into smaller parts, for which finding the solution takes less time.

We considered the Rolling Time Horizon technique for this purpose since it is the method used for a wide spectrum of tasks.

3.5 Rolling Time Horizon usage

3.5.1 Energy Hub System

Marquant, Evins, and Carmeliet, 2015 optimized Urban Energy Hub System costs on energy production by the formulation of the Mixed Integer Linear programming problem for the hub. Energy hub takes as input energy from different sources: wind, solar panels, gas, grids. After, the hub converts input energy into the goods that consumers require – heat, cold water, or electricity. The consumers demand changes during the day, week, and season, so changes sequence of the hub parts activation – either convert energy to electricity, or to heat, or to cool.

To optimize the process, the authors formed a mixed integer linear programming problem for the yearly functioning of the energy hub with hourly discretization. For each hour of the year there are a few free variables that describe each part of the energy hub, so the problem consists of a huge number of components. It makes the process of finding the solution quite long. The authors used the Rolling Time Horizon technique to decompose the problem into the smaller part, for which the solution could be found much faster. The results show that the rolling time horizon approach can reduce the computational time from 7 to 100 times without significant quality difference.

3.5.2 Aircraft Traffic flow management

Samà, D'Ariano, and Pacciarelli, 2012 presented research about the scheduling of traffic in the airport area by solving the problem using the rolling time horizon

technique. The authors created a pipeline that consists of two modules: the first solves the problem of routing decisions, the second - timing decisions. In routing decisions were an optimized process of choosing a specific air corridor for a plane. The timing determined the passing timing of the plane in the airport nearby. The authors compared their results of the optimization task with the existing rule FIFO, and the resulting algorithm gave improvements. In order to speed up the solution, the authors used the Rolling Time Horizon technique to decompose and speed up the optimization task. As a result, Rolling Time Horizon increased the speed performance 7-10 times.

Chapter 4

The approach and solution

4.1 Overview of existing methods

4.1.1 Optimization problems

The Optimization problem is a problem of finding a minimal or maximal value of a function

$$f : R^n \Rightarrow R$$

Usually there exist some additional constrains, so the optimization problem is defined in the following way:

$$\min_{x \in R^n} / \max f(x)$$

$$c_i(x) \leq 0,\ i \in IC$$

$$c_e(x) = 0,\ e \in EC$$

where IC - set of inequality constraints, EC - set of equality constraints, $c : R^n \Rightarrow R$

Optimization problems classification

There are classifications over the class of optimization problems. Generally, the optimization task could be divided in Mixed Integer NonLinear Programming (MINLP) and Nonlinear programming [Moving-horizon and Hydroformylierungsanlage, 2015]. In the MINLP optimization problem exist part of constraints of some variables that require integers values. General formulation looks similar:

$$\min_{x,y} / \max f(x, y)$$

$$c_i(x, y) \leq 0, where\ i \in IC$$

$$c_e(x, y) = 0, where\ e \in EC$$

$$x \in R^n, y \in Z^k$$

where IC - set of inequality constraints, EC - set of equality constraints.

If functions f, c_i, c_e are linear than the problem becomes Mixed Integer Linear Programming. If there is no $x \in R^n$ variables than the problem is stated as Integer Linear Programming problem.

Nonlinear programming with only continuous variables also are divided into different classes: whether the functions f, c_i, c_e are differentiable – have first and second derivatives.

The next characteristic that divides problems into classes is whether the objective function f is convex or not. This will affect if the objective function has a local optimum or not.

For quadratic programming exist specific solutions and derivatives could be found in an easy way.

And Linear programming problem when the functions f, c_i, c_e are linear.

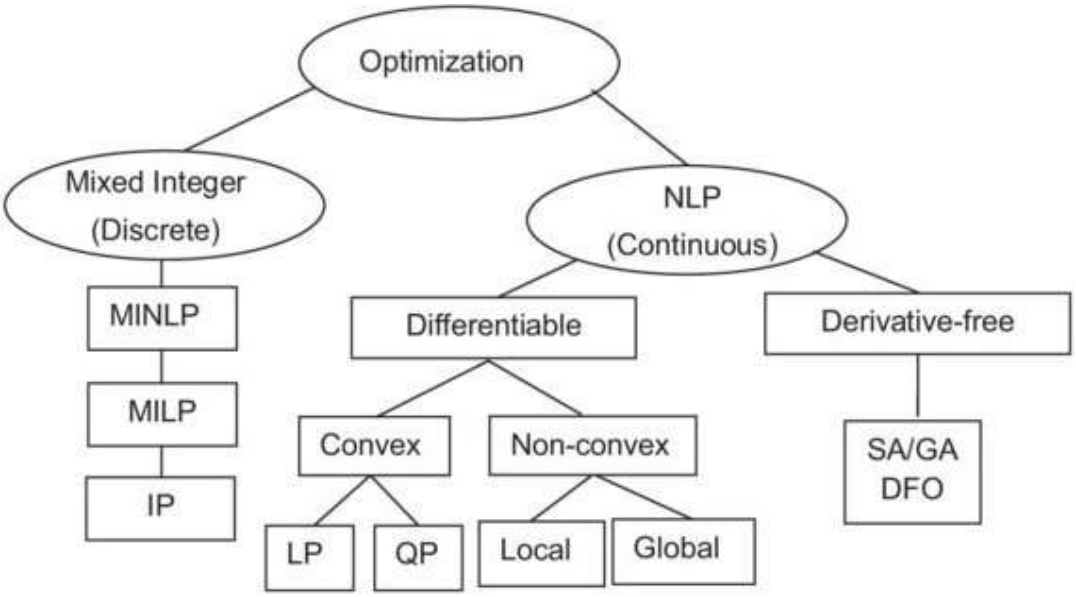


FIGURE 4.1: Classification of optimization problems
Source: *Classification of optimization problems*

In our formulation of the problem, we have integer linear programming task, since variables x show if the sailor is assigned for some particular job or not, so the possible values are $x \in \{0, 1\}$.

Optimization problems solvers

Exist quite a lot of products that solve optimization problems (MINLP, MILP, IP, etc) in an automatic way. Mostly, they are written in low-level languages such as C or C++ for higher performance and have API for easier usage by the end-user. These solvers differ in the class of problems that they solve, compatibility with programming languages, performance on problems of different dimensionalities, commercial usage, or open-source.

For example, CPLEX - the leading commercial solver, released by IBM - provides nice performance, but it is expensive. LINDO is compatible with Excel (also commercial). COIN-OR is a project that provides a lot of free solvers, and its purpose is to foster open-source collaborations of research and industry [Donoghue, 2015].

Some of the solvers have their own modeling language (CPLEX solver – OPL language, FICO solver – Mosel language). Modeling languages transform initial optimization problem into solver language, and return results of solution back in an understandable way (such functionalities also provide high-level languages, such as Python).

Comparison of packages performance:

TABLE 4.1: Performance of different packages on N-Queens problem [Bruen and Dixon, 1975]
Source: *Integer Programming models using different mathematical modelling packages*

n	PuLP	Python-MIP	JuMP
100	0.24	0.97	0.45
200	1.43	0.18	0.19
300	4.97	0.37	0.38
400	12.37	0.72	0.74
500	24.7	1.25	1.23
600	43.88	2.02	1.99
700	69.77	3.04	2.94
800	105.04	4.33	4.26
900	150.89	5.95	5.83
1000	206.63	8.02	7.76

In our case, we used Pulp package with CBC solver.

4.1.2 Rolling Time Horizon

The Rolling Time Horizon (RTH) is an approach that is widely used to increase the performance of finding an optimal solution in an optimization task from a computational (timing) point of view. The Rolling Time Horizon could be used for the optimization of work of Energy Hub [Marquant, Evins, and Carmeliet, 2015] or for Air Traffic Control [Samà, D'Ariano, and Pacciarelli, 2012]. Also, the RTH technique is used in such domains as Pulp Distribution [Bredstrom and Rönnqvist, 2006] and Gas supply chains [Zamarripa et al., 2016].

The general idea is to solve the optimization task with fewer time variables, so instead of solving a complex problem for a whole time period, the problem is solved for consecutive intervals on which the time period is divided. Each interval consists of fewer time variables, so the solution for such sub-problem could be found more easily.

The whole time period is divided into successive sub-problems (with intersection). For each sub-problems the optimization task is solved. The schema of the RTH approach described in the picture below:

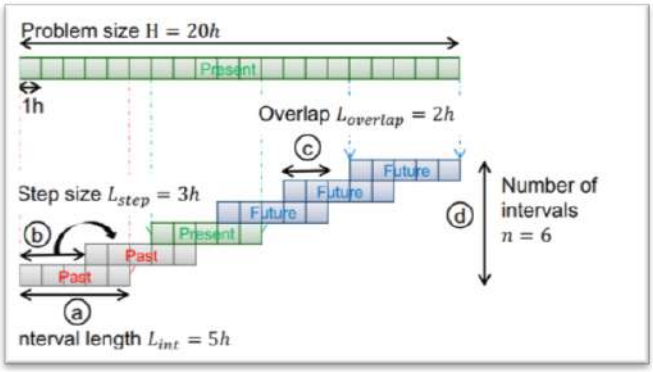


FIGURE 4.2: Rolling Time Horizon schema
Source: Marquant, Evins, and Carmeliet, 2015

The intersection is needed to make it possible for the consequent sub-problem to use the result of the previous sub-problem. The following sub-problem uses the intersection as an initial state for the optimization task.

4.2 The solution

4.2.1 Formalization of terms

Jobs forming

Each vessel must have the required crew to serve its functionality. These requirements defined by "Principles of minimum safe manning", so there exist a set of positions that should be permanently filled. Also, there are restrictions on the sailor working on the vessel for more than 11 months. Based on these rules and information about current and the previous sailors' trips on different ranks, new jobs are generated after the expected ending of the voyage of the previous sailor on a particular duty. The lengths of the generated job depend on the sailors' contract.

Compliance function

Sailors have different education and working experience. It makes them differently compliant with jobs and ranks. To measure this compliance, the function C was defined.

$$C(person\ information, job\ requirements) \Rightarrow R$$

Function C takes into account persons' private information, education, experience on different vessels within the company, and measures the match between sailor and position on the vessel ($C(sailor_i, job_j) = 0$ if the $sailor_i$ is not matching for the job_j; and the more suitable the sailor, the greater the value of the function). So the one person can have different compliance values for the same role on different vessels, and a different values for positions within one vessel for different roles.

Complex Job

Complex Job is a job that consists of two parts. And either we can assign two different sailors for two parts of complex job, or we can assign one sailor for both parts of the complex job, and reduce costs on flights (to take off the previous sailor from the vessel, and put on next sailor on the vessel).

4.2.2 Optimization problem for crew assignment

The approach of crew assignment was implemented for a maritime company on the market. As described before, jobs were generated for the next half of the year (using information about historical and current voyages). After, for each sailor from the base of the company was calculated value of compliance function C.

The optimization of crew assignment was formed as maximization of total match between sailors and vacancies:

$$\max_x \sum_{i=0}^{N} \sum_{j=0}^{M} C_{i,j} * x_{i,j}$$

s.t.

$$\sum_{i=0}^{N} x_{i,j} \leq 1, \forall j$$

$$\sum_{j=0}^{M} x_{i,j} \leq 1, \forall i$$

$$\sum_{j=0}^{M} (x_{i,j} + \sum_{k\ in\ CJ_i} w_k * x_{k,j}) \leq 1, for\ i \in CJ$$

$$x_{i,j} \in \{0,1\}, \forall i,j$$

where $x_{i,j}$ - indicator of assignment of sailor j for job i,
N – number of jobs,
M – number of sailors,
CJ – set of Complex Jobs,
w – weight of part of Complex Job,
$C_{i,j}$ - compliance function value for sailor j for job i.
The first constraint allows assigning a sailor for no more than 1 job.
The second constraint allows assigning a job for no more than 1 sailor.
The third constraint allows assigning either Complex Job or two parts of Complex Job, but not simultaneously.

For this setup, solution of the task of optimization of the crew on vessels takes a lot of time, since time variables produce a lot of free variables in the optimization task (on job start each sailor could be assigned, so we have Cartesian product of the jobs and sailors, and a complexity of the integer linear programming task rises exponentially with increment in number of variables [Domínguez-Muñoz et al., 2011]). So there was a need to speed up the existing solution.

4.2.3 Rolling Time Horizon for crew assignment

On each iteration of the Rolling Time Horizon for crew assignment, we consider the set of jobs $INRVL$ that starts at the interval, set of jobs that fall in the intersection of the successive rolls - $INTRS$, set of unassigned sailors – US. On each roll we solve the optimization problem that looks similar to the whole optimization problem:

$$\max_x \sum_{i \in INRVL} \sum_{j \in US} C_{i,j} * x_{i,j}$$

s.t.

$$\sum_{i \in INRVL} x_{i,j} \leq 1$$

$$\sum_{j \in US} x_{i,j} \leq 1$$

$$\sum_{j \in US} (x_{i,j} + \sum_{k \ in \ CJ_i} w_k * x_{k,j}) \leq 1, for \ i \in CJ$$

$$x_{i,j} \in \{0, 1\}$$

where $x_{i,j}$ - indicator of assignment of sailor j for job i,
$INRVL$ – set of jobs that starts at the interval,
US – set of unassigned sailors,
CJ – set of Complex Jobs in $INRVL$,
w – weight of part of Complex Job,
$C_{i,j}$ - compliance function value for sailor j for job i.

After the solution of the optimization problem for one iteration, we consider two sets of assignments: for $\{jobs|jobs \in INRVL \setminus INTRS\}$, we fix the assigned sailors as the final solution; for assignments $\{jobs|jobs \in INTRS\}$, we put the assignments as the initial state for the next roll.

4.2.4 Heuristics for sailors prioritization

Licenses expiring

If we consider compliance function C, we will see that it works well when it's used for the solution of the whole optimization problem at once, since information about all variables is included in the problem formulation. But when we apply the Rolling Time Horizon approach, we can face issues.

For example, we have job A and job B with similar requirements. Job A starts in May, job B starts in October. We have sailors $\{a_1, b_1\}$ with similar experience, but licenses of $\{a_1\}$ expires earlier than $\{b_1\}$, so both sailors equally compliant for job A, and $\{a_1\}$ less compliant than $\{b_1\}$ for job B . If we assign $\{b_1\}$ for job A during the first roll period and then assign $\{a_1\}$ for job B, the value of $\{a_1\}$ on job B will be lower (because of expiring licenses), so we get lower objective function value.

A specially chosen function called a "heuristic" can be used to solve this kind of issue. A heuristic is a technique to produce a better solution by modification of the problem, but not necessarily the optimal one. An example heuristic, to avoid the above issue introduced by utilising the Rolling Time Horizon approach, would be to define a heuristic function based on crew licence expiry dates. For each job there is requirement for particular number of licenses, some of them being interchangeable. So for each sailor we can find the set of required licenses S that has maximal expiring date. After, in the set S we find the license with minimal expiring date, and put it as

$$FLE \ (first \ license \ expiring) := min\_expiring\_date(S)$$

So heuristic is calculated in way:

$$h(person \ information, job \ requirements) = \frac{1}{1 + FLE - Job \ Start \ Date}$$

And modify function of compliance:

$$\hat{C}(person\ info., job\ req.) = C(person\ info., job\ req.) + h(person\ info., job\ req.)$$

In this case, sailors with earlier licenses expiring will have a higher priority to jobs with an earlier start.

Flights costs

The transporting of the sailors to land and from the land on the vessel might be costly. Ships are constantly moving around the globe, and sailors live in different countries. So here the optimization of the additional costs might be. Using the planned route of the vessel, we can consider the port at which a new job on the vessel will start. Based on this information, information about sailor homeland, and using Skyscanner API, additional priorities was added for the sailors that have cheaper flights to the destination port for a particular job.

4.2.5 Complex Jobs in Rolling Time Horizon

As described before, Complex Job is a job that consists of two parts, and we can either assign Complex Job for one sailor or two parts for different sailors.

Since we do not consider the whole problem in Rolling Time Horizon, it was necessary to implement a new logic for Complex Jobs. There are 3 possible cases with Complex Job (CJ) and rolls:

1. CJ and both parts of the CJ in roll, but the second part is in the intersection
2. One part of the CJ and CJ in the roll, and the second is not
3. CJ and it's both parts in the roll, and not in the intersection

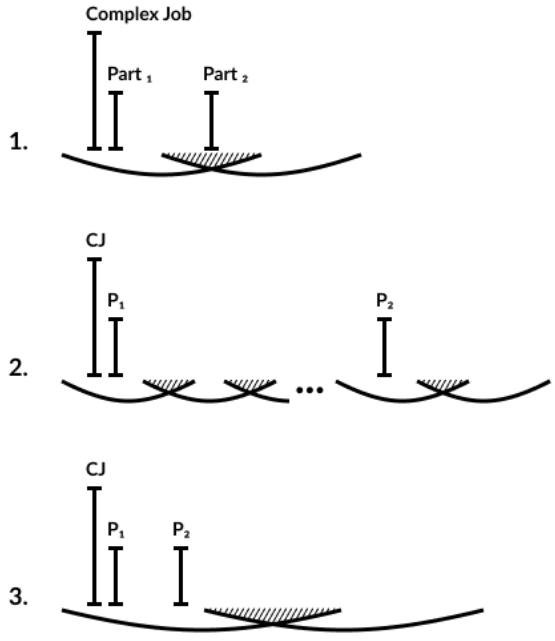


FIGURE 4.3: Complex Jobs possible cases in the Rolling Time Horizon

So we can consider during the rolls 3 or 2 jobs: CJ, CJ_{part1}, and CJ_{part2}; or only CJ_{part1} and CJ_{part2} - it depends on the case.

In case 1 and case 2 we should exclude CJ from consideration in the RTH, and leave only CJ_{part1} and CJ_{part2}.

For case1: during the first roll, if we assign a sailor for the Complex Job, it means that during the next roll, when CJ_{part2} is in the initial state, and assignment can change, we will not be able to change the assignment. It conflicts with the RTH logic. So in this case, during the first roll we consider only CJ_{part1} and CJ_{part2}, and during the second roll make able to assign sailor from CJ_{part1} to CJ_{part2} with additional value. For case2: analogously - we should be able to assign another sailor to CJ_{part2}, so CJ should be excluded. For case3: we consider CJ, CJ_{part1}, and CJ_{part2} simultaneously. So during the roll, we can assign either one sailor for CJ, or two different sailors for CJ_{part1} and CJ_{part2}.

4.2.6 Transfer of the solution into jobs to end-user

After the solution of the problem using Rolling Time Horizon, we should post-process the results in order to form the correct assignments. It includes joining of the assignment of one sailor to CJ_{parts}, and transforming it into CJ assignment. After, we transform the assignment of the sailor for the formal jobs to the information about the vessel, starting and ending date, rank of the person on assignment, etc.

The provided solution enables us to speed up the optimization problem solving, automate the process of assignment of sailors to the jobs on vessels, and save the money required on the existing crewing process.

Chapter 5

Implementation and evaluation

5.1 Data description

For the project we used real data from one of the largest maritime companies in the world. The data included:

- Description of the vessels:
  - Type of the vessel (tanker, bulk carrier, etc.)
  - Flag of the vessel (country)
  - The optimal crew on the vessel
- Historical routes of the vessels
- Person information:
  - Person licenses
  - Person working experience
  - Person private information
- Prices on the flights

5.2 Implementation process

Remark: the database configuration, the process of data extraction, jobs generation, compliance function calculation, flight heuristic, benchmark solution, and front-end output was implemented by members of 90POE team.

5.2.1 Requirements for the application

The existing application should work in a secure way and provide an easy-to-use interface to users for work with. The application should give end-users information about the optimal distribution of sailors of vessels for the company. As input, the software acquires information about previous and current trips and descriptions of the sailors' experience and licenses. After getting input data, the software generates approximate jobs on the period, that user input. Then starts the process of measuring compliance function C between jobs and sailors. Based on the function value, and jobs, we solve the integer linear programming task of optimal assignment of sailors to jobs. The solution of the optimization task is used to plot the resulting sailors' assignment as Gantt charts for end users. The Gantt charts show the information in which sailor is assigned on which vessel, when and in which role.

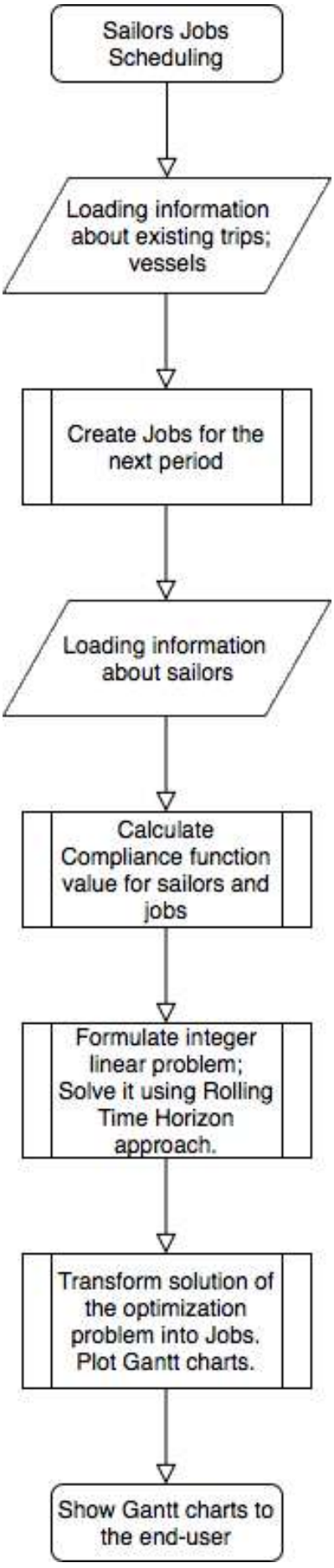


FIGURE 5.1: Application pipeline

5.2.2 Hardware and software

The solution was made using the programming language Python 3, and many packages that made able to perform different parts of the product. The data extracting, transforming, and loading (ETL) were made using Pandas library. Pandas make able to connect to the database and download the data samples. After downloading, Pandas DataFrame provides different methods for easy implementation for data preprocessing, features generation, data aggregation, etc.

The integer linear problem was defined and solved using Pulp package. In our setup, we used CBC solver, since we worked with the integer linear programming problem. Pulp also has a payment version of solvers, which perform the faster calculation. After, using Flask package, the front-end part was created, where Gantt charts are plotted using Plotly package. All the calculations and implementation of the work were made on MacBook Pro 2017 laptops, with processor 2.3 GHz Intel Core i5, 8 GB RAM, macOS operating system.

5.2.3 Technology transfer

The developed technology could be transferred to other maritime companies since the standards are commonly used in the world (and could be modified if needed); sailors information is standard too – because licenses are unified in the world; vessel types are similar. So by changing the data sources, specific rules of the company, and API's details, it could be used in another systems.

5.3 Evaluation

The performance of the Rolling Time Horizon was considered from the objective function value point of view, and time of solution. The objective function value was calculated as the total sum compliance values for the resulting assignments of sailors for the jobs. It means the assignments with the maximal matching between sailors skills and jobs requirements.

$$objective\ function\ value = \sum_{(i,j)\in A} C_{i,j}$$

where A - set of pairs of assignments (i, j) (sailor j assigned for job i),
$C_{i,j}$ - compliance function value of sailor j for job i.

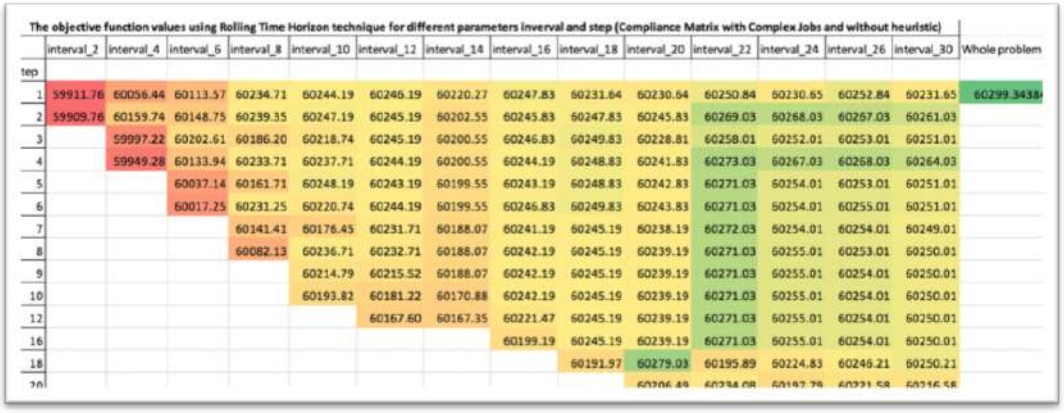
The time of solution was calculated as time required on forming and finding the optimum for the integer linear programming problem, since the other steps (data loading, preprocessing, plotting the results, etc.) are present in any other approach, and it is similar.

Such evaluation we did for each pair $(step, interval)$, so for $step = i$, $interval = j$ we solved the optimization problem using Rolling Time Horizon technique, evaluated the performance from the objective function value and time of solution.

The benchmark is the solution of the whole optimization problem at once.

We also made a comparison of results with and without heuristic to see if the function h for jobs prioritization for each particular sailor, depending on licenses expiring, increases the objective function value for the Rolling Time Horizon approach.

For the compliance matrix without the heuristic, we have the following results for the objective function values:

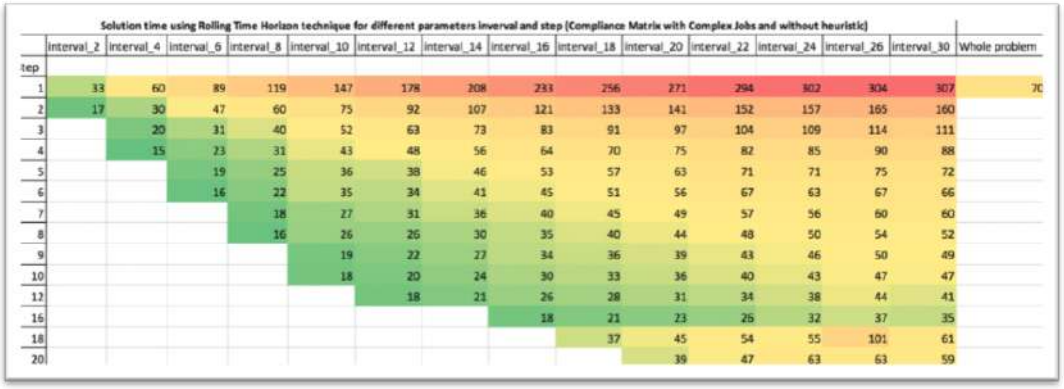


The objective function values using Rolling Time Horizon technique for different parameters inverval and step (Compliance Matrix with Complex Jobs and without heuristic)

tep	interval_2	interval_4	interval_6	interval_8	interval_10	interval_12	interval_14	interval_16	interval_18	interval_20	interval_22	interval_24	interval_26	interval_30	Whole problem
1	59911.76	60056.44	60113.57	60234.71	60244.19	60246.19	60220.27	60247.83	60231.64	60230.64	60250.84	60230.65	60252.84	60231.65	60299.3438
2	59909.76	60159.74	60148.75	60239.35	60247.19	60245.19	60202.55	60245.83	60247.83	60245.83	60269.03	60268.03	60267.03	60261.03	
3		59997.22	60202.61	60186.20	60218.74	60245.19	60200.55	60246.83	60249.83	60228.81	60258.01	60252.01	60253.01	60251.01	
4		59949.28	60133.94	60233.71	60237.71	60244.19	60200.55	60244.19	60248.83	60241.83	60273.03	60267.03	60268.03	60264.03	
5			60037.14	60161.71	60248.19	60243.19	60199.55	60243.19	60248.83	60242.83	60271.03	60254.01	60253.01	60251.01	
6			60017.25	60231.25	60220.74	60244.19	60199.55	60246.83	60249.83	60243.83	60271.03	60254.01	60255.01	60251.01	
7				60141.41	60176.45	60231.71	60188.07	60241.19	60245.19	60238.19	60272.03	60254.01	60254.01	60249.01	
8				60082.13	60236.71	60232.71	60188.07	60242.19	60245.19	60239.19	60271.03	60255.01	60253.01	60250.01	
9					60214.79	60215.52	60188.07	60242.19	60245.19	60239.19	60271.03	60255.01	60254.01	60250.01	
10					60193.82	60181.22	60170.88	60242.19	60245.19	60239.19	60271.03	60255.01	60254.01	60250.01	
12						60167.60	60167.35	60221.47	60245.19	60239.19	60271.03	60255.01	60254.01	60250.01	
16								60199.19	60245.19	60239.19	60271.03	60255.01	60254.01	60250.01	
18									60191.97	60279.03	60195.89	60224.83	60246.21	60250.21	
20										60206.49	60234.08	60197.79	60221.58	60216.58	

FIGURE 5.2: The objective function value for different parameters of the RTH and Compliance Matrix with Complex Jobs and without Heuristic

From a timing point of view:



Solution time using Rolling Time Horizon technique for different parameters inverval and step (Compliance Matrix with Complex Jobs and without heuristic)

tep	interval_2	interval_4	interval_6	interval_8	interval_10	interval_12	interval_14	interval_16	interval_18	interval_20	interval_22	interval_24	interval_26	interval_30	Whole problem
1	33	60	89	119	147	178	208	233	256	271	294	302	304	307	70
2	17	30	47	60	75	92	107	121	133	141	152	157	165	160	
3		20	31	40	52	63	73	83	91	97	104	109	114	111	
4		15	23	31	43	48	56	64	70	75	82	85	90	88	
5			19	25	36	38	46	53	57	63	71	71	75	72	
6			16	22	35	34	41	45	51	56	67	63	67	66	
7				18	27	31	36	40	45	49	57	56	60	60	
8				16	26	26	30	35	40	44	48	50	54	52	
9					19	22	27	34	36	39	43	46	50	49	
10					18	20	24	30	33	36	40	43	47	47	
12						18	21	26	28	31	34	38	44	41	
16								18	21	23	26	32	37	35	
18									37	45	54	55	101	61	
20										39	47	63	63	59	

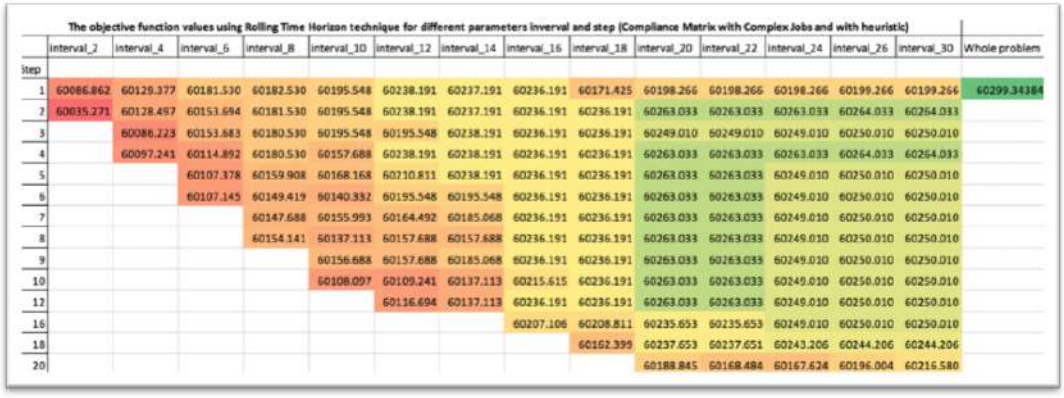
FIGURE 5.3: Solution time for different parameters of the RTH and Compliance Matrix with Complex Jobs and without Heuristic

Analogously, we evaluated the compliance matrix with the heuristic. For the optimization task used compliance function with the heuristic to prioritize sailors

$$\hat{C}(person\ info., job\ req.) = C(person\ info., job\ req.) + h(person\ info., job\ req.)$$

but the provided evaluation made on compliance function without heuristic C to make results comparable.

The results for the objective function values:

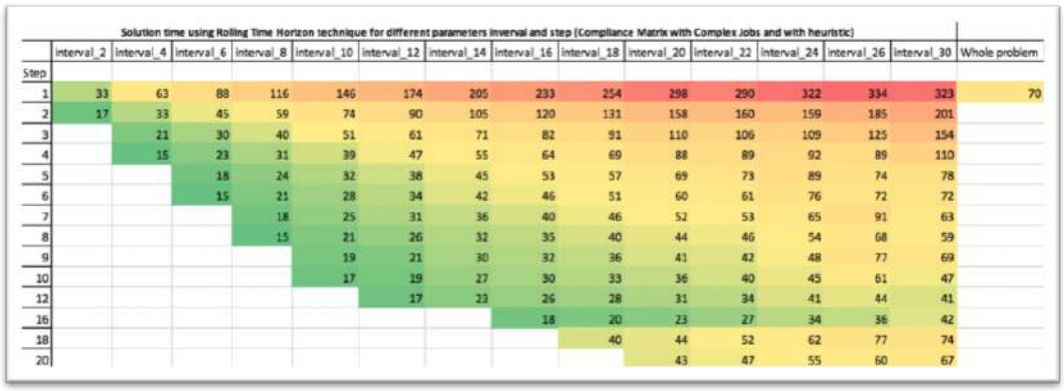


The objective function values using Rolling Time Horizon technique for different parameters inverval and step (Compliance Matrix with Complex Jobs and with heuristic)

Step	interval_2	interval_4	interval_6	interval_8	interval_10	interval_12	interval_14	interval_16	interval_18	interval_20	interval_22	interval_24	interval_26	interval_30	Whole problem
1	60086.862	60129.377	60181.530	60182.530	60195.548	60238.191	60237.191	60236.191	60171.425	60198.266	60198.266	60198.266	60199.266	60199.266	60299.34384
2	60035.271	60128.497	60153.694	60181.530	60195.548	60238.191	60237.191	60236.191	60236.191	60263.033	60263.033	60263.033	60264.033	60264.033	
3		60086.223	60153.683	60180.530	60195.548	60195.548	60238.191	60236.191	60236.191	60249.010	60249.010	60249.010	60250.010	60250.010	
4		60097.241	60114.892	60180.530	60157.688	60238.191	60238.191	60236.191	60236.191	60263.033	60263.033	60263.033	60264.033	60264.033	
5			60107.378	60159.908	60168.168	60210.811	60238.191	60236.191	60236.191	60263.033	60263.033	60249.010	60250.010	60250.010	
6			60107.145	60149.419	60140.332	60195.548	60195.548	60236.191	60236.191	60263.033	60263.033	60249.010	60250.010	60250.010	
7				60147.688	60155.993	60164.492	60185.068	60236.191	60236.191	60263.033	60263.033	60249.010	60250.010	60250.010	
8				60154.141	60137.113	60157.688	60157.688	60236.191	60236.191	60263.033	60263.033	60249.010	60250.010	60250.010	
9					60156.688	60157.688	60185.068	60236.191	60236.191	60263.033	60263.033	60249.010	60250.010	60250.010	
10					60108.097	60109.241	60137.113	60215.615	60236.191	60263.033	60263.033	60249.010	60250.010	60250.010	
12						60136.694	60137.113	60236.191	60236.191	60263.033	60263.033	60249.010	60250.010	60250.010	
16								60207.106	60208.811	60235.653	60235.653	60249.010	60250.010	60250.010	
18									60162.399	60237.653	60237.651	60243.206	60244.206	60244.206	
20										60188.845	60168.484	60167.624	60196.004	60216.580	

FIGURE 5.4: The objective function value for different parameters of the RTH and Compliance Matrix with Complex Jobs and with Heuristic

From a timing point of view:

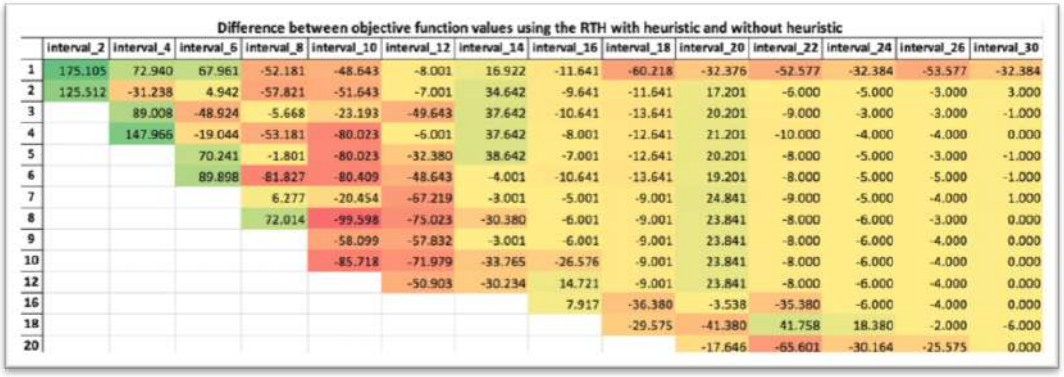


Solution time using Rolling Time Horizon technique for different parameters inverval and step (Compliance Matrix with Complex Jobs and with heuristic)

Step	interval_2	interval_4	interval_6	interval_8	interval_10	interval_12	interval_14	interval_16	interval_18	interval_20	interval_22	interval_24	interval_26	interval_30	Whole problem
1	33	63	88	116	146	174	205	233	254	298	290	322	334	323	70
2	17	33	45	59	74	90	105	120	131	158	160	159	185	201	
3		21	30	40	51	61	71	82	91	110	106	109	125	154	
4		15	23	31	39	47	55	64	69	88	89	92	89	110	
5			18	24	32	38	45	53	57	69	73	89	74	78	
6			15	21	28	34	42	46	51	60	61	76	72	72	
7				18	25	31	36	40	46	52	53	65	91	63	
8				15	21	26	32	35	40	44	46	54	68	59	
9					19	21	30	32	36	41	42	48	77	69	
10					17	19	27	30	33	36	40	45	61	47	
12						17	23	26	28	31	34	41	44	41	
16								18	20	23	27	34	36	42	
18									40	44	52	62	77	74	
20										43	47	55	60	67	

FIGURE 5.5: Solution time for different parameters of the RTH and Compliance Matrix with Complex Jobs and with Heuristic

The difference between objective functions using the RTH with heuristic and without it:



Difference between objective function values using the RTH with heuristic and without heuristic

	interval_2	interval_4	interval_6	interval_8	interval_10	interval_12	interval_14	interval_16	interval_18	interval_20	interval_22	interval_24	interval_26	interval_30
1	175.105	72.940	67.961	-52.181	-48.643	-8.001	16.922	-11.641	-60.218	-32.376	-52.577	-32.384	-53.577	-32.384
2	125.512	-31.238	4.942	-57.821	-51.643	-7.001	34.642	-9.641	-11.641	17.201	-6.000	-5.000	-3.000	3.000
3		89.008	-48.924	-5.668	-23.193	-49.643	37.642	-10.641	-13.641	20.201	-9.000	-3.000	-3.000	-1.000
4		147.966	-19.044	-53.181	-80.023	-6.001	37.642	-8.001	-12.641	21.201	-10.000	-4.000	-4.000	0.000
5			70.241	-1.801	-80.023	-32.380	38.642	-7.001	-12.641	20.201	-8.000	-5.000	-3.000	-1.000
6			89.898	-81.827	-80.409	-48.643	-4.001	-10.641	-13.641	19.201	-8.000	-5.000	-5.000	-1.000
7				6.277	-20.454	-67.219	-3.001	-5.001	-9.001	24.841	-9.000	-5.000	-4.000	1.000
8				72.014	-99.598	-75.023	-30.380	-6.001	-9.001	23.841	-8.000	-6.000	-3.000	0.000
9					-58.099	-57.832	-3.001	-6.001	-9.001	23.841	-8.000	-6.000	-4.000	0.000
10					-85.718	-71.979	-33.765	-26.576	-9.001	23.841	-8.000	-6.000	-4.000	0.000
12						-50.903	-30.234	14.721	-9.001	23.841	-8.000	-6.000	-4.000	0.000
16								7.917	-36.380	-3.538	-35.380	-6.000	-4.000	0.000
18									-29.575	-41.380	41.758	18.380	-2.000	-6.000
20										-17.646	-65.601	-30.164	-25.575	0.000

FIGURE 5.6: Difference in objective functions values between the RTH with heuristic and without it

As we can see, the heuristic shows nice performance on short intervals and gives increment to the rolling approach, without using the intersection of sub-problems.

5.3.1 Heuristic for hyperparameters choosing

For choosing parameters for the rolling time horizon approach we defined the function that takes into account difference with benchmark from objective function value, and the time of solution. It calculates as a weighted sum of deviation from the optimal solution for the whole problem and increment in time required to solve the optimization problem. For alpha=0.99 I calculated the heuristic (since deviation from the solution for the whole problem less than 1%, and in other way we will just take into account time difference).

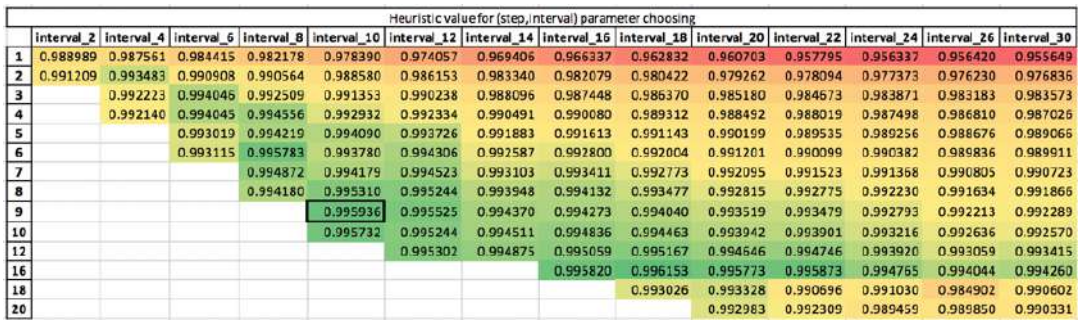
$$V := objective\ function\ value\ for\ the\ whole\ problem\ (benchmark)$$

$$T := time\ required\ for\ the\ whole\ problem\ solution\ (benchmark)$$

$$h_p(obj.f.value_{step,interval}, sol.time_{step,interval}) \Rightarrow R$$

$$h_p(.,.) = \frac{obj.f.value_{step,interval}}{V} * \alpha + (1 - \frac{sol.time_{step,interval}}{T}) * (1 - \alpha)$$

We propose parameters interval=10, step=9.



Heuristic value for (step,interval) parameter choosing

	interval_2	interval_4	interval_6	interval_8	interval_10	interval_12	interval_14	interval_16	interval_18	interval_20	interval_22	interval_24	interval_26	interval_30
1	0.988989	0.987561	0.984415	0.982178	0.978390	0.974057	0.969406	0.966337	0.962832	0.960703	0.957795	0.956337	0.956420	0.955649
2	0.991209	0.993483	0.990908	0.990564	0.988580	0.986153	0.983340	0.982079	0.980422	0.979262	0.978094	0.977373	0.976230	0.976836
3		0.992223	0.994046	0.992509	0.991353	0.990238	0.988096	0.987448	0.986370	0.985180	0.984673	0.983871	0.983183	0.983573
4		0.992140	0.994045	0.994556	0.992932	0.992334	0.990491	0.990080	0.989312	0.988492	0.988019	0.987498	0.986810	0.987026
5			0.993019	0.994219	0.994090	0.993726	0.991883	0.991613	0.991143	0.990199	0.989535	0.989256	0.988676	0.989066
6			0.993115	0.995783	0.993780	0.994306	0.992587	0.992800	0.992004	0.991201	0.990099	0.990382	0.989836	0.989911
7				0.994872	0.994179	0.994523	0.993103	0.993411	0.992773	0.992095	0.991523	0.991368	0.990805	0.990723
8				0.994180	0.995310	0.995244	0.993948	0.994132	0.993477	0.992815	0.992775	0.992230	0.991634	0.991866
9					0.995936	0.995525	0.994370	0.994273	0.994040	0.993519	0.993479	0.992793	0.992213	0.992289
10					0.995732	0.995244	0.994511	0.994836	0.994463	0.993942	0.993901	0.993216	0.992636	0.992570
12						0.995302	0.994875	0.995059	0.995167	0.994646	0.994746	0.993920	0.993059	0.993415
16								0.995820	0.996153	0.995773	0.995873	0.994765	0.994044	0.994260
18									0.993026	0.993328	0.990696	0.991030	0.984902	0.990602
20										0.992983	0.992309	0.989459	0.989850	0.990331

FIGURE 5.7: Heuristic values h_p for different parameters $(step, interval)$

Chapter 6

Conclusions

6.1 Summary

The crew assignment pipeline was developed to automate the process of assigning crew to available jobs onboard vessels. It can decrease costs on crewing and logistics, speed up the procedure of finding a person to a job in the maritime industry. During the assignment, the pipeline takes into account not just the current recruiting needs, but also future opening vacancies.

The algorithm acquires information about voyages, vessels, flight prices, crew experience and education, and provides optimal assignment for the next period of time, defined by the user. It performed using a formalization of the crewing task into a Integer Linear Programming problem and solution of the problem.

We implemented the approach of Rolling Time Horizon for crew assignment in the maritime industry. It increased the time performance by 3.5 times with a deviation of less than 1% from the optimal solution, in comparison with the benchmark. To do it, we implemented the specific logic of Complex Jobs for the Rolling Time Horizon; introduced the example of the heuristic that can be used for prioritization of elements in the Rolling Time Horizon approach; defined heuristic for choosing parameters of the Rolling Time Horizon approach.

The obtained results prove the research hypothesis and meet with research expectations.

6.2 Future work

As the future improvement of the work, we consider using solvers with higher performance. It should provide similar results with the current one from the objective function value point, but with a higher speed. But it might be costly, since a lot of them are commercial.

Also, it is possible to divide sailors into groups by possible jobs, so the sailors within each group will be compliant for some subset of the jobs. And then solve the optimization problem within the group. Such decomposition could increase performance too, since for each optimization problem there will be fewer free variables.

The objective function value might also include additional factors: cooperation of the sailors, working on a particular vessel in the past, etc. On the one hand, it will make the optimization problem more complicated, but on the other, it will include more information in the model.

And one of the main plans for the future — testing of the solution with end-users.

Bibliography

Arkin, Esther M and Ellen B Silverberg (Sept. 1987). "Scheduling jobs with fixed start and end times". In: *Discrete Appl. Math.* 18.1, pp. 1–8. DOI: 10.1016/0166-218X(87)90037-0.

Bhatia, Randeep et al. (2003). "Algorithmic Aspects of Bandwidth Trading". In: *Automata, Languages and Programming*. Springer Berlin Heidelberg, pp. 751–766. DOI: 10.1145/1219944.1219956.

Bredstrom, David and M Rönnqvist (Dec. 2006). "Supply Chain Optimization in Pulp Distribution Using a Rolling Horizon Solution Approach". URL: dx.doi.org/10.2139/ssrn.1145184.

Bruen, A and R Dixon (1975). *The n-queens problem*. DOI: 10.1016/0012-365X(75)90079-5.

Clausen, J (1999). "Branch and bound algorithms-principles and examples". In: *Department of Computer Science, University*. URL: imada.sdu.dk/~jbj/heuristikker/TSPtext.pdf.

Domínguez-Muñoz, Fernando et al. (Nov. 2011). "Selection of typical demand days for CHP optimization". In: *Energy Build.* 43.11, pp. 3036–3043. DOI: 10.1016/j.enbuild.2011.07.024.

Donoghue, John R (2015). *Comparison of Integer Programming (IP) Solvers for Automated Test Assembly (ATA)*. URL: files.eric.ed.gov/fulltext/EJ1109657.pdf.

Kolen, Antoon W J et al. (Aug. 2007). "Interval scheduling: A survey". In: *Nav. Res. Logist.* Lecture Notes 54.5, pp. 530–543. DOI: 10.1002/nav.20231.

Kroon, Leo G, Marc Salomon, and Luk N Van Wassenhove (Apr. 1995). "Exact and approximation algorithms for the operational fixed interval scheduling problem". In: *Eur. J. Oper. Res.* 82.1, pp. 190–205. DOI: 10.1016/0377-2217(93)E0335-U.

Marquant, Julien F, Ralph Evins, and Jan Carmeliet (Jan. 2015). "Reducing Computation Time with a Rolling Horizon Approach Applied to a MILP Formulation of Multiple Urban Energy Hub System". In: *Procedia Comput. Sci.* 51, pp. 2137–2146. DOI: 10.1016/j.procs.2015.05.486.

Moving-horizon, Echtzeit-Optimierung und and Zustandsschätzung für eine Hydroformylierungsanlage (2015). "Real-time Optimization and Moving-horizon State Estimation for a Hydroformylation Plant". In: DOI: 10.13140/RG.2.1.1737.9686/1.

python-mip.readthedocs.io. *Integer Programming models using different mathematical modelling packages*. URL: python-mip.readthedocs.io/en/latest/bench.html.

Resolution, A. "Principles of minimum safe manning". In: *Principles of safe manning* (). URL: www.imo.org/en/OurWork/HumanElement/VisionPrinciplesGoals/Documents/890.pdf.

Samà, Marcella, Andrea D'Ariano, and Dario Pacciarelli (Oct. 2012). "Optimal Aircraft Traffic Flow Management at a Terminal Control Area during Disturbances". In: *Procedia - Social and Behavioral Sciences* 54, pp. 460–469. DOI: 10.1016/j.sbspro.2012.09.764.

Sekimizu, K (2010). "Stcw A Guide For Seafarers. International Transport Workers' Federation". In: URL: www.mptusa.com/pdf/STCW_guide_english.pdf.

semanticscholar.org. *Classification of optimization problems*. URL: www.semanticscholar.org/paper/Classification-of-Optimization-Problems-3-1-.-2-of/df184bb6a4315f1caeb378ad15988307595ca36e.

ship-technology.com. *Different kind of vessels*.

United Nations (Jan. 2019). *Review of Maritime Transport 2018*. en. UN. URL: unctad.org/en/PublicationsLibrary/rmt2018_en.pdf.

Zamarripa, Miguel et al. (Mar. 2016). "Rolling Horizon Approach for Production–Distribution Coordination of Industrial Gases Supply Chains". In: *Ind. Eng. Chem. Res.* 55.9, pp. 2646–2660. DOI: 10.1021/acs.iecr.6b00271.