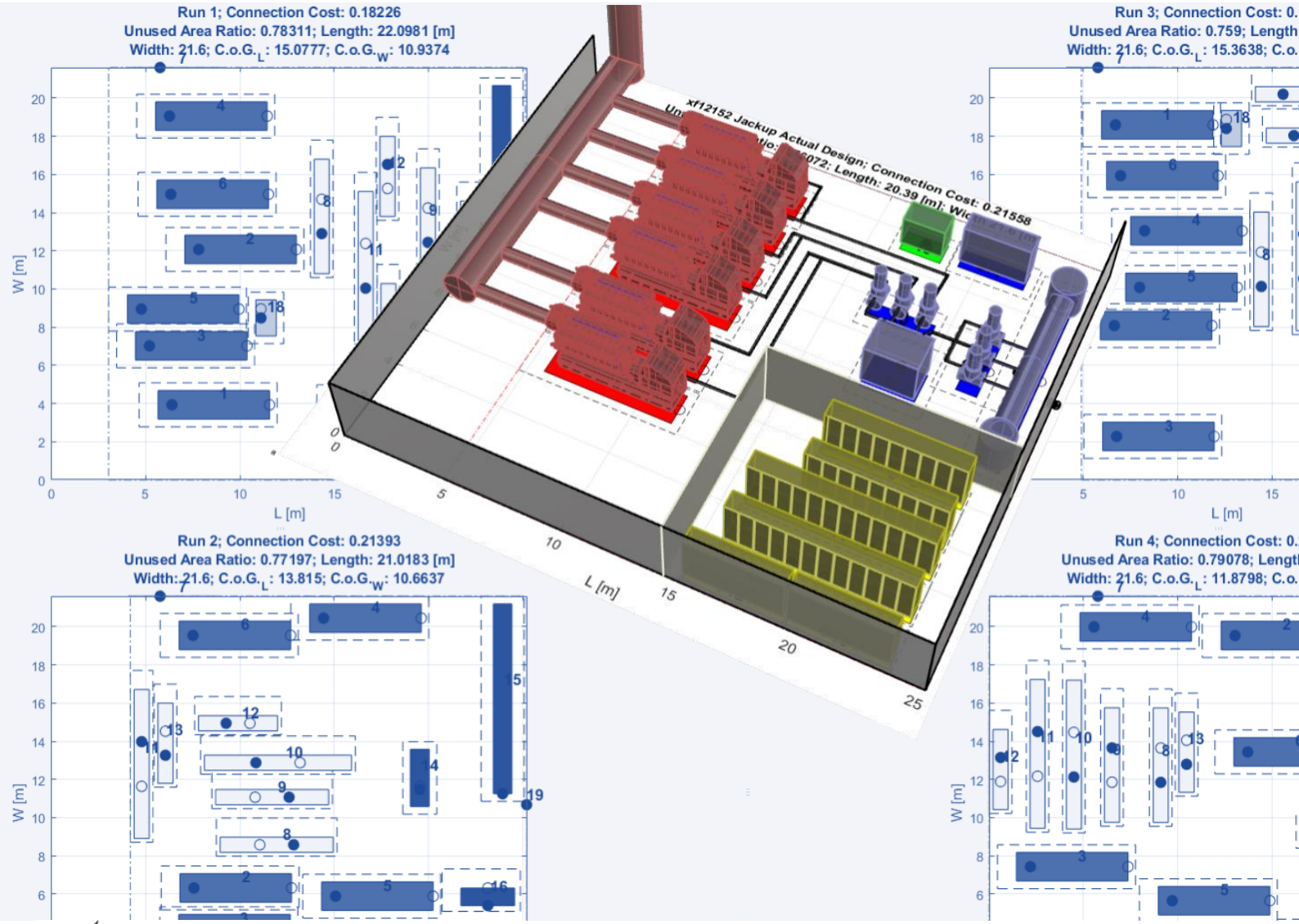
A Design Tool for Machinery Space Arrangement

MSc Thesis Report
SDPO.19.004.m.

R. van der Bles



TU Delft Delft University of Technology

VUYK DESIGN TO FUNCTION.

A Design Tool for Machinery Space Arrangement

MSc Thesis Report
SDPO.19.004.m.

by

R. van der Bles

in partial fulfilment of the requirements for the degree of

Master of Science
in Maritime Technology

at Delft University of Technology to be defended publicly on April 10th, 2019 at 14:00.

Student number	4119401
Project duration	March 28th, 2018 – April 10th, 2019

Thesis committee

Prof.ir. J.J. Hopman	Delft University of Technology
Dr. A.A. Kana	Delft University of Technology
Dr. B. Atasoy	Delft University of Technology
N.R. Mayenburg	Vuyk Engineering Rotterdam B.V.

Summary

The subject of this master thesis is the development of a design tool to help Vuyk Engineering Rotterdam B.V. arrange the machinery spaces (engine room, switchboard rooms, separator rooms) aboard their vessels early in their design process. It applies the field of automatic design generation to a step in Vuyk's design process to save time and easily explore alternatives.

First the general design process at Vuyk and the design of the engine room and other machinery spaces in particular are investigated. This is followed by a review of literature on design of machinery spaces, both in maritime applications in ships and in non-maritime process plant lay-out design, to compare theory and practice at Vuyk and look for new insights. Next to the design process itself, the characteristics of ship concept design tools are investigated in literature, leading to a focus on concept exploration and various methods how to deal multiple objectives. This results in the design requirements for the design tool for machinery space arrangements and a focus on the most relevant requirements to consider in the limited time frame of this thesis, to limit the scope.

The best modelling method and resolution approach for the problem are investigated. The Facility Layout Problem and Particle Swarm Optimization algorithm are identified as the best options. The model is defined accordingly.

The capabilities of the model for the design tool are tested in a concept design of a jackup vessel. Initial runs of the model led to problems for the Particle Swarm Optimization algorithm, as its gets stuck in local minima while trying to solve the problem. This severely limits the solutions diversity.

Several adaptations for the model are developed to overcome this problem. The particle swarm algorithm is adapted with several mutations, allowing it to better escape the local minima thus creating more diverse, better optimized solutions.

The model is also improved with an additional grouping constrained and a pathing filter to better account for the identified machinery space design drivers.

The resulting tool is applied in a case study on the early concept design phase for a dredge pump room and connected engine room for a trailing suction hopper dredger. The tool is used to evaluate early design choices in space arrangements for a diesel-direct drive of the dredging pump and jet pump, in comparison to alternative diesel-electric solutions with three V16 diesel-generator-sets, four V12 sets or six V8 generator sets.

The application of the tool leads to valuable insights for the arrangement options for the machinery spaces. The developed tool is shown to be a useful addition to the early concept design of machiney space arrangement.

Contents

List of Figures

List of Tables

1 Introduction

The subject of this master thesis is the development of a design tool to help Vuyk Engineering Rotterdam B.V. arrange the machinery spaces (engine room, switchboard rooms, separator rooms) aboard their vessels early in their design process. It applies the field of automatic design generation to a step in Vuyk's design process to save time and easily explore alternatives. In the following sections first the problem will be introduced, including the company Vuyk, a general description of an Engine Room and the concept of a design tool for automatic generation of machinery space arrangements. In the next section the research objective and questions are defined, followed by an overview of the structure of the thesis.

1.1. Problem introduction

Vuyk Engineering Rotterdam B.V. is a company which, among other activities, provides one-off vessel designs of a wide variety. Vuyk expressed interest in the possibilities of automatic ship design generation in their early design stages, specifically applied to engine rooms.

When early in their design process the first general arrangement is developed, one of the first spaces that is analysed in more detail is the engine room with its associated machinery spaces, such as switchboard and separator rooms. An example of a topview of an engine room is presented in figure 1.1.

Several concepts are developed for the layout of this room, based on the experience of the Vuyk engineers. Subjects like ventilation airflow in and out or the need for different room dimensions to properly fit all the equipment can have a substantial impact upon the design of the vessel and thus are important to determine early in the design process. Developing these different machinery space concepts is a very time-consuming endeavour, that takes several iterations before the general arrangement of the vessel is finalized.

1.1.1. General Description of an Engine Room

In order to understand the problem first a general sense of what an engine room is is needed.

The Engine Room is the machinery space in a vessel that contains the combustion engines (mostly diesel engines) that produce the power for the propulsion of the vessel and for other energy consumers like for example dredge pumps in a dredger.

Generally spoken engine rooms are characterized by the diesel engine(s) that are positioned in that space, with a number of alternatives:

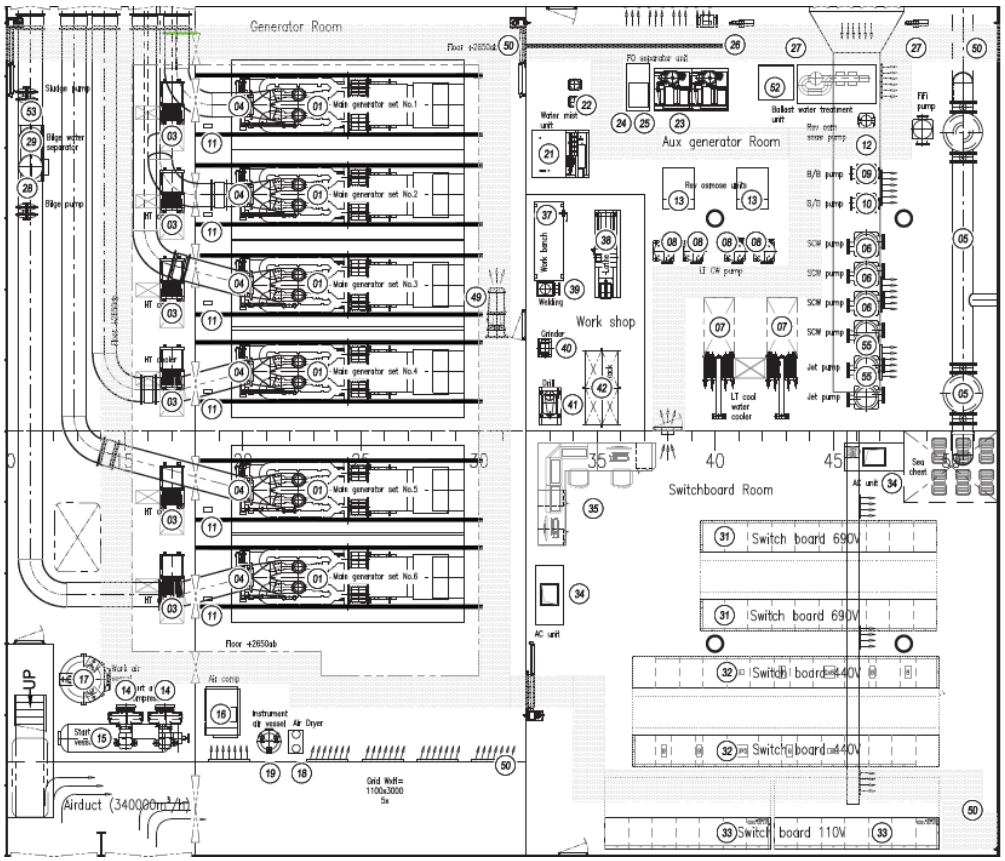


Figure 1.1: Example of an engine room arrangement of a Jackup vessel

- one large 2-stroke slow speed diesel engine driving a propellor shaft, usually without a gearbox in between, positioned in the aft part of the vessel;
- A smaller, more compact 4-stroke medium speed diesel engine connected to a gearbox that is driving the propellor shaft, also positioned in the aft part of the vessel;
- - variants with two or more medium speed diesel engines, driving through one or more gear-box(es) one or more shaft(s), positioned in the aft part of a vessel; a variant of this type is an engine room with medium speed diesel engines driving the shaft of a dredge pump and/or a jet pump in a dredger vessel (that are designed regularly by Vuyk); in this case the engine room can also be positioned in the forward part of the vessel;
- diesel-electric engine room concepts, that have quite different characteristics: usually a number of diesel-generator sets of the medium speed type are positioned in the engine room, as shown in figure 1.1, providing electric power to large switch boards that can be positioned in the same engine room or in a separate switch board room; the electric power is transmitted to electric motors that drive the propeller shaft(s) and /or dredge pumps.

The diesel-electric concept provides the designer of the vessel with more freedom to position the engine room not necessarily down below in the aft part of the hull, but possibly on a higher deck level. The shape of this kind of engine room may be rectangular, not influenced by the hull lines and the height may be just one or two deck levels.

Vuyk quite often applies the diesel-electric concept in dredgers and offshore vessels, with rather rectangular engine rooms.

A variety of systems and their components can be found in the machinery spaces aboard an engine room:

Next to the main diesel engines, various types of other machinery are positioned in the engine room, like pumps for the fuel-, lubrication- and cooling water systems, coolers/heat-exchangers in the cooling systems, etc. An exhaust gas system with large diameter piping may take quite some space in the engine room between the diesel engines and the exhaust casing with silencers and other devices. Other items in engine rooms are the various tanks for fuel, lubrication oil, bilge water and others, in different sizes. The design of the engine room is the process in which the designer determines the best arrangement of all machinery in the required space. The issues mentioned above will be taken into account in the development of the design tool for the arrangement of machinery in the engine room and other machinery spaces

1.1.2. Concept of a Design Tool for Automatic Generation of Machinery Space Arrangements

Automatic ship design generation is an area of ongoing research both in the academic world and in the industry. It can cover a range from generating entire (low detail) ship concepts to a smaller, more detailed problems concerning a single space, or even a high detail application in a space such as pipe routing through an engine room.

A design tool which automatically generates the machinery space concepts, and scores their performance to certain criteria, could help speed up this step in the design process while also aiding in more easily exploring alternative arrangements for this specific space aboard the vessel. The well-scoring concepts would be presented to the designer who can judge these concepts not just on their own, but in relation to the whole vessel as well.

As this concerns a step early in the design process, there is still uncertainty about exact components that need to be included and their specifications of the machinery. Assumptions need to be made about which components need to be modelled and their properties, and how to account for components which are not modelled. The design rationale of why machinery rooms are arranged a certain way needs to be investigated and incorporated into the design tool.

As Vuyk designs a wide range of ship types, and each individual design faces different challenges, it is hard to try to optimize the machinery space concepts for a generic scenario applicable to each design case. Therefore, to keep the tool useful for Vuyk, the performance of a design along different characteristics will be quantified after which the designer can decide what the pertinent drivers are for the specific design case.

1.2. Research objective and questions

From the problem introduction, the research objective is formulated:

> *Automatically generate arrangements of machinery spaces in an early design phase and quantify their performance in the overall design perspective of the vessel, in order to support the designer in selecting a feasible arrangement for a machinery space in this early design phase.*

In order to reach this goal, the following research questions will be addressed:

- 1. What is the design process and rationale of the Vuyk designers when developing these initial concepts for the machinery spaces?
- 2. Which components will and will not be modelled in this phase, and why? (E.g. main engine, generators, separators, switchboards, pumps, compressors)
  - a. What are the properties of the modelled components that need to be taken into account? How can these be determined?
  - b. How can the components that are not modelled still be accounted for in the design?

- 3. Which generation/optimization method for the arrangements will be used for the design tool?
- 4. How is the performance of a machinery space defined?

1.3. Structure of the Thesis

After the introduction of the problem that is the subject of this thesis and the research objective and questions in the first chapter, in chapter 2 the Problem Definition is elaborated. The chapter starts with the investigation of the general design process at Vuyk and the design of the engine room and other machinery spaces in particular. This is followed by an investigation of literature on design of machinery spaces, both in maritime applications in ships and in non-maritime process plant lay-out design, to compare theory and practice at Vuyk and look for new insights. Next to the design process itself, the characteristics of ship concept design tools are investigated in literature, leading to a focus on concept exploration and various methods how to deal multiple objectives. Chapter 2 results in the design requirements for the design tool for machinery space arrangements and a focus on the most relevant requirements to consider in the limited time frame of this thesis, to limit the scope.

In chapter 3 the solution directions are identified for modelling of layout problems. The analysis of the possible solutions leads to the choice of the most promising modelling method and resolution approach.

Chapter 4 describes how the model is defined according to the facility layout problem, its objectives and constraints, and the workings of the particle swarm optimization method that was selected as the resolution approach.

The capabilities of the model for the design tool are tested in chapter 5 for a concept design of a jackup vessel. Initial runs of the model lead to adaptations and improvements of the functionality of the tool, described in this chapter.

The resulting tool is applied in chapter 6 for a case study on the early concept design phase for a dredge pump room and connected engine room for a trailing suction hopper dredger. The tool is used to evaluate early design choices in space arrangements for a diesel-direct drive of the dredging pump and jet pump, in comparison to alternative diesel-electric solutions with three V16 diesel-generator-sets, four V12 sets or six V8 generator sets.

2

Problem Definition

In order to successfully tackle a problem, it's context and parameters need to be understood. This chapter outlines the process of defining the problem definition. This starts with an investigation into the design process at Vuyk, in which first the general design process is discussed and then the engine room design in particular. Secondly the relevant literature on machinery space design from both the maritime field and other fields is presented. After this the literature on the concept exploration process is investigated. Finally this chapter culminates in a set of design requirements for a method or tool for machinery space arrangement which are concluded from the information presented in this chapter.

2.1. The Design Process

To answer the first research question "What is the design process and rationale of the Vuyk designers when making these initial concepts for the machinery spaces?", several interviews were held with the naval architects and marine engineers at Vuyk to identify the overall design process followed at Vuyk and the particulars of designing the machinery spaces. The following sections document the results of these interviews.

2.1.1. The Overall Design Process

The design process of a new vessel is divided in three phases: the concept design, the contract design (named basic design at Vuyk), and the detail design Tupper (2013). In figure 2.1 these main design phases are illustrated, as practised at Vuyk.

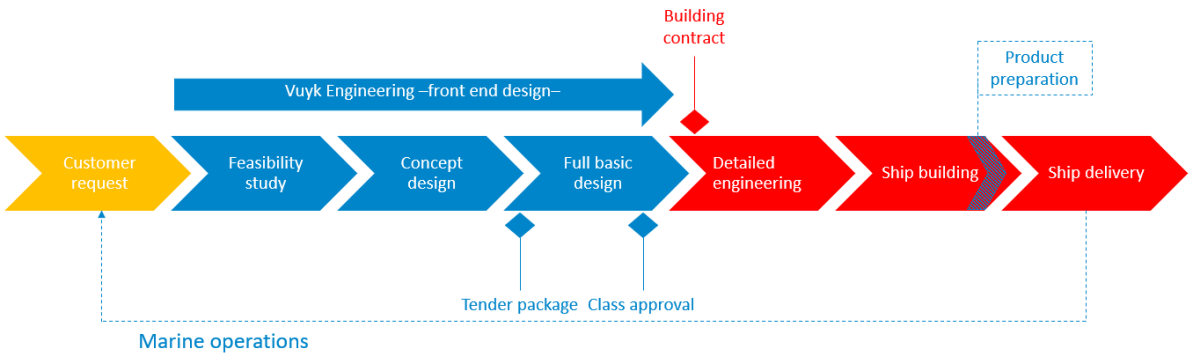


Figure 2.1: Phases of ship design as used by Vuyk (van den Berg, 2018).

After a new project starts following the customer request, Vuyk typically performs the front end design consisting of a feasibility study, the concept design and the basic design. This results in a class approved design of sufficient detail to get a newbuilding contract at a shipyard, who then performs the detailed engineering and builds the vessel.

The design phases generally follow the ship design spiral, as presented in fig 2.2, where a design is iterated upon until it is refined enough to enter the next design phase. Whenever the concept changes substantially, a new iteration is needed to calculate the impact of the change upon the design. As this spiral is used by Vuyk to communicate its design process to its customers, it explicitly consists of the deliverables contractually agreed to with a client, such as a main cross section or system diagrams. The engine room is influenced by several of the design steps in the spiral. A space is allocated in the general arrangement, the machinery can be a significant factor in the light ship weight estimate, the hull form often limits the shape of the engine room, speed and power calculations pose requirements for the machinery, and so on.

The next sections detail how the front end design phases shown in figure 2.1 are handled at Vuyk from the start of the project.

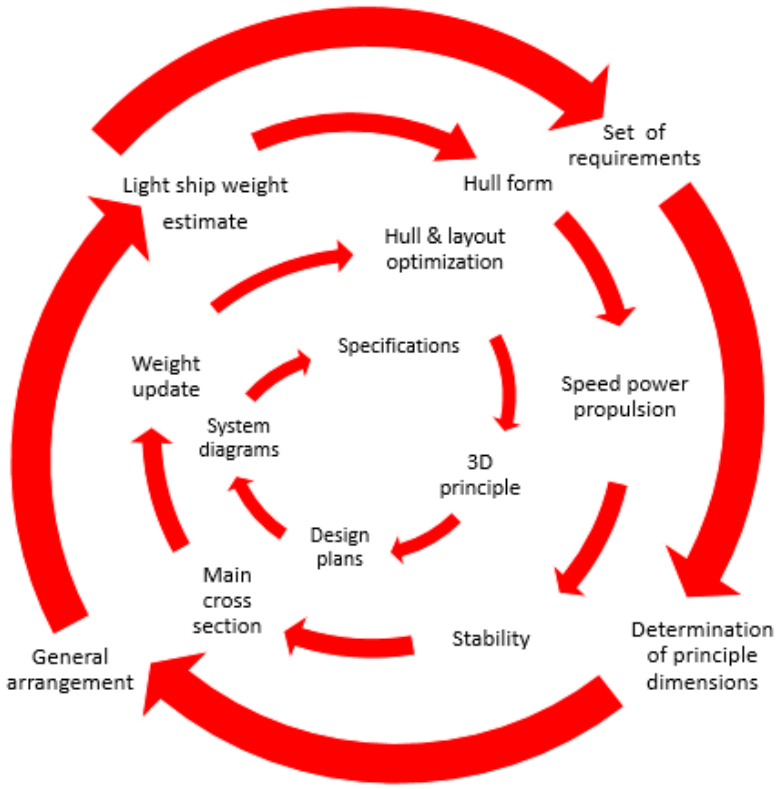


Figure 2.2: The ship design spiral used by Vuyk (van den Berg, 2018).

Customer Request

In the initial talks with the client a design brief is constructed. This is a short document, of only 2-3 pages, which outlines the design wishes and requirements of the client and serves as a starting point for the design process. Both negotiable (e.g. the ship will be around 100m) and non-negotiable (e.g. as the client wants to sail in certain areas, the draught of the vessel must not exceed 6m) design requirements are formulated. If the client has specific wishes regarding the machinery spaces, e.g. location aboard the vessel, propulsion preferences, fuel preferences, machinery brands or specific machinery, these are established in this phase.

Concept Design

Although Vuyk formally distinguishes a feasibility study phase, as is seen in figure 2.1, in practice this is considered to be a part of the initial concept design phase. With the design brief established the concept design is started by a small team, which initially only consist of the project manager, who at Vuyk is a naval architect, and one additional engineer. The concept design phase normally takes about two to three months. The process starts with estimating the main dimensions, the weight of the vessel, and drawing up a first concept based on rough calculations and reference vessels. As more work can be done in parallel, more people gradually join the project, but usually the team stays small in this design phase, not exceeding four or five members.

The concept design phase is generally regarded as the most important design phase as it establishes the true requirements the designer is too meet. Also, in this phase a large percentage of the total ship cost is committed, although the actual spendings, in both time and money, are small relative to the rest of the project (Tupper, 2013). Figure 2.3 illustrates this committed cost, together with the available design freedom and the gathered problem knowledge, trough the design stages. By increasing the known problem knowledge as early as possible, while the design freedom to handle identified problems is still available, risk in the design process can be reduced and the process as a whole is made more efficient.

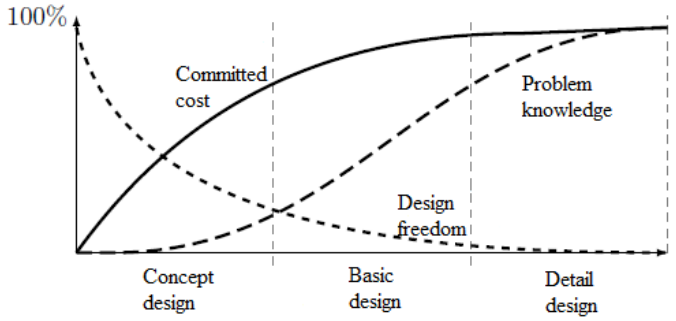


Figure 2.3: Commited cost, problem knowledge and design freedom trough the design stages. Adapted from Mavris and DeLaurentis (2000).

The goal of the concept design phase is twofold: 1) to show that the client's wishes result in a feasible design, and 2) to design a vessel which the client can show to his management, investors or own clients in order to convince them that this is a worthwhile design to pursue further. To reach this goal several deliverables are contractually agreed to for the concept design:

- General arrangement
- Lines plan
- Tank plan
- Electric Load Balance
- Electric single line
- Midship section
- Provisional stability calculations
- Speed and power calculations

Most of these deliverables are related to the machinery spaces on board of a vessel: Speed and power calculations necessitate choices in the propulsion setup, (preliminary) equipment is specified in the electric load balance and single line, areas are allocated and (some) equipment is placed in the general arrangement, which is of course influenced by the other deliverables. Some additional items can be contractually agreed, depending on their importance to a particular ship type or specific wishes by the client. Examples are items such as a motion analysis and dynamic positioning calculations for a heavy lift vessel.

The level of detail to which the engine room is elaborated to during the concept design may vary for different projects, depending on the time available for the concept design, the availability of the personnel, the complexity of the vessel and the wishes of the client. When a minimal amount of attention is paid to the machinery spaces, the naval architect allocates a space in which he positions the engines, generators and routes the ventilation inlets and outlets. On the other hand, when the space is a priority

and the personnel is available, after about two weeks a machinery specialist can join the team to start the design of the machinery spaces. He designs several concepts based on the information available and can adapt one or more of these when more information becomes known. This early integration of the machinery spaces into the design process ensures that potential issues, such as the space allocation for all the needed components or the ventilation requirements of the engine room can be tackled early and avoid costly redesigns later in the design process.

Place of the Design Tool in the Design process

The early concept design phase is identified as the phase where a new machinery arrangement tool can add the most value to the design process of Vuyk. There are several reasons for this: the ability to speed up the acquisition of problem knowledge early in the design process and to mitigate the risk of costly redesigns later on in the project. An important facet of this problem knowledge is accurately estimating the required area needed for a machinery space to support an optimal use of the space available in the general arrangement. Underestimating this area can lead to these costly redesigns in a later phase. Also the ability to save time while reducing the required man-hours of the experienced marine engineers is seen as important. This can either speed up this design phase, or allow for more time to be spend on either this or another part of the ship design in this important design phase.

In order to give the context of the whole design process the other design phases are also described below.

Basic Design

With the concept design finished the client has a clear idea whether his requirements result in a feasible vessel design, what this vessel looks like, its important dimensions and characteristics, and an estimate of the building and life-cycle costs. If the client wants to move forward the basic design starts. This design phase lasts approximately six to nine months. More people, mostly specialists, join the project team which can consist of up to 15 people during the busiest project months.

At Vuyk the basic design culminates in a design which has been approved by a class society and a flag state, which ensures that the design follows all the relevant rules and regulations. The class society and flag are chosen by the client, although Vuyk has their own experiences with these organisations and can advise a client on this choice if needed, as some organisations have slightly different rules from others.

The goal of this design phase is a design of sufficient detail to present to a shipyard and get an accurate quotation for the building price of the vessel. A more detailed design allows a yard to make a more detailed building cost estimation, reducing their uncertainty margin which would drive up the price.

The design of the engine room is continued from the concept design, with the necessary design steps depending upon the level of detail which was achieved in the concept design. As the systems are designed and the components are specified it becomes clear whether all the components will fit in the allocated space, or if a (likely costly) redesign is needed. The earlier this is established, the easier it is to accommodate the necessary changes in the rest of the design.

Detail design

After the basic design is finished the detail design of the vessel remains. The client may approach several shipyards and enters negotiations with one or several parties that may build the vessel based on the basic design. Vuyk rarely performs the detail design of a vessel. Most often the builder of the vessel executes this design phase, as they can work closely with their production department to incorporate their specific expertise and the particulars and constraints of the yard.

Figure 2.2 showed the iterative nature of the ship design process, and figure 2.3 shows that late in the design process there is still a significant amount of problem knowledge learned. Often, at the end of a design project the designer already has knowledge on how to improve the design but this cannot be implemented any more due to the lack of design freedom at that stage. This knowledge is taken with

the designer and can be applied to similar designs in future projects.

Due to the duration of the detail design phase and the building process, there is often a gap of at least two years between the end of a project for Vuyk and the moment the vessel is delivered to the owner. During this time Vuyk has of course immersed itself in other projects. As such there is only limited opportunity to gather detailed feedback on the real life performance of a vessel. This is further complicated if the client was not a regular customer of Vuyk, in which case it is possible that no feedback gets back to Vuyk at all, from either the detail design or the actual operation of the vessel.

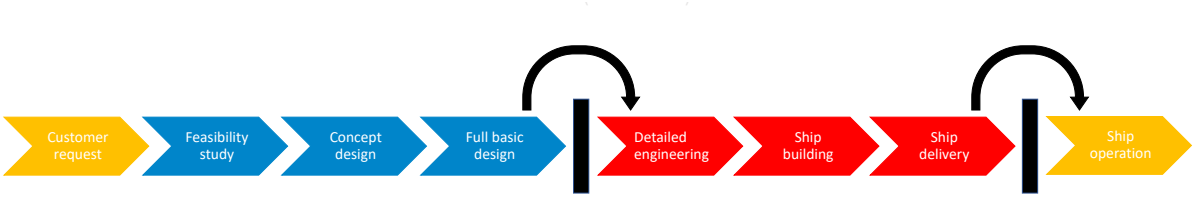


Figure 2.4: 'Walls' between the design stages

This limited communication between these design phases, and also the communication gap between the building and the operation of the vessel is often referred to as 'over the wall'. This is illustrated in figure 2.4, where walls are placed between the different design phases and the building and operation phases. In ship design the complexity of the design problem is addressed by not dealing with all the aspects of the design at each stage. After a phase has been finished, the result is 'thrown over the wall' to be used by the next team. There is no flow backwards between these walls, which can lead to inefficiencies due to missing information and the need to re-work some aspects of the design.

At Vuyk the same team handles the concept and basic design of a vessel, expanding over the project duration as described earlier in this section, so no wall exists between these design phases. Then the project generally gets 'thrown over the wall' to a yard which will perform the detailed engineering and building of the vessel. Finally the vessel is delivered to the client, which then operates the vessel. This can also be seen as an "over the wall" step, as the crew of the vessel gains insight in the design during the operating phase of the vessel.

2.1.2. Designing the Engine Room and Associated Machinery Spaces

After determining the general design phases above, now the way Vuyk handles the design of a engine rooms and machinery spaces is investigated.

For Vuyk a detailed look into the machinery space arrangement can start anywhere from 2 weeks into a project to the beginning of the basic design phase. Generally, at least the following information can be considered known:

- An area within the design is allocated for the engine room
- The locations of accesses to this area are determined
- Air inlets and outlets are positioned
- Total engine power is estimated
- Total electrical power is estimated
- The number of propellers, and thus driveshafts have been established
- Initial speed-power calculations have been performed, from which preliminary propeller properties are known

Depending on the client some additional information is possibly available. Sometimes the client already has procured equipment to be used in the vessel, or the client has specific requests due to, for example, past experiences or a wish for a certain level of uniformity across his fleet. This uniformity can be using the same brands of equipment he already uses in other vessels, due to existing connections and

contracts with those companies or specialized training of the crew. This uniformity can also express itself in a wish for certain equipment to be located in specific places in order to allow for crew to easily familiarize themselves with the new vessel. The client can also have a policy for additional comfort above that defined by the rules, which in an engine room can express itself in broader than normal entries and exits, less steep stairs which take additional space or a lower maximum temperature level.

From the required power from the main engines a lot of information can be derived. If no engines have been specified by the client, they are estimated from the main required power. A gearbox can be sized if it is needed. The lubrication oil, fuel oil and cooling systems are all connected to and dependent upon the engines and can thus be estimated once the main engines are determined. The same goes for the ventilation system and starting air system. The electrical system of course depends on the amount of electrical power installed.

Depending upon the engine and fuel types selected, special fuel treatment systems need to be available or scrubbers and selective catalytic reduction (S.C.R.) units are needed to treat the exhaust gasses.

It is of course important that the engine room arrangement fits in the allocated space. If this is impossible the naval architect should know this as soon as possible, so the overall vessel design can be altered as necessary. In order to get a satisfactory design, the experience of the designers at Vuyk identifies three main design drivers:

- Accesses
- Ventilation
- The necessary space for maintenance

Accesses consist of entryways to other spaces on the same level (doorways) and other levels (stairs). They take up space, influence the airflow in the rooms, and as the rules and regulations have pushed for more comfortable accesses (larger doors and less steep stairs), looking at the reference vessels can be deceptive.

The engine room ventilation is a system which has a considerable impact upon the vessel design, with the inlets and outlets taking up considerable space. The exhaust system especially, with the scrubbers, SCR units and silencers taking up space for vessels that need them to comply with new rules and regulations. After the sizing of the ventilation system is done they are often positioned in direct communication with the naval architect. New regulations as well as advancing technologies make the use of reference vessels deceptive as well.

The necessary space for maintenance is not only the space directly around a piece of equipment so that it can be maintained, for which often recommendations are set by the manufacturer, but also routes through the space for spare parts or whole replacements to move and reach the equipment. The need for this is often impacted by the owners wishes. For example, a lot of dredging vessels operate in remote areas with a lot of equipment operating regularly in overload and at excessive working hours, and thus an above average failure rate. Owners of these vessels often want to be able to do a lot of maintenance themselves, and sometimes require even the main engines to be able to be replaced without disassembly. This requires an exit path for this large piece of equipment, which can considerably influence the design of an engine room and the rest of the vessel.

Generally, in the first concepts only the big components are placed: the main and auxiliary engines with their associated gearboxes, ventilation entries, exits and shafts and fuel day tanks. The above mentioned accesses and ventilation up- and downtakes are allocated in direct dialogue with the naval architect. Then, as the concept design phase progresses and more information is available and decisions are made, the necessary auxiliary systems are positioned.

Components that are not specified yet are initially sized from reference vessels. Only when certain information is known is it possible to design a component to specification. Even then, there can be substantial design changes which lead to the redesign or reselection of components. Also, the client can and often will specify or change components during the detailed design and building process depending

on manufacturer or system installer offers. This requires careful selection of the dimensions of the engine room in the first concept design phase, where the designer takes additional space and margins into account for possible future changes and additional machinery.

When a particular system is designed there are a variety of ways to handle redundancy requirements or trade-offs. An example of a trade-off in the cooling system is the relation between pump size and cooler size. A smaller pump requires a larger cooler, resulting in higher initial cost for a lower operational cost.

2.2. Design of machinery spaces in literature

Now that the design process and rationale for machinery space design as practised by Vuyk is known, literature is investigated to see whether it agrees with Vuyks practices and can offer new insights. First the literature from the maritime field is discussed. Secondly literature from other fields is investigated.

2.2.1. Ship Machinery Space Related Literature

Klein Woud and Stapersma (2002) mention that the dimensions of machinery spaces are dictated by both the overall ship design and the volume and clearances necessary for maintenance and overhaul of the machinery in the space. They state that the arrangement of the machinery in a space follows a limited number of straightforward considerations:

- A mechanical drive propulsion plant is located in such a way that it can be connected to the propulsors.
- Auxiliary equipment is located in the direct vicinity of the main equipment it has to support, to reduce cabling and piping.
- Some equipment needs to be located low in the machinery space, such as certain pumps to enable good suction behaviour.
- Some equipment needs to be located high in the machinery space, such as expansion tanks to maintain static pressure or ventilation ducting and fans.
- Much equipment does not have strict location requirements, such as air compressors and vessels, switchboards and chillers. They are located at convenient places from a total ship layout point of view. Here considerations may be locations of weight and centre of gravity, and vicinity to consumers.
- Sufficient space should be available for access, control, monitoring and maintenance of equipment.

Most of these guiding rules can be implemented as constraints for the optimization problem, once the specific equipment that is subjects to these constraints has been identified.

Babicz (2015) states that it is necessary to investigate the layout of the engine room(s) from the very beginning of a design. He considers the following items of special importance in early engine room arrangement:

- Ventilation
- Transport ways
- Escapes
- Maintenance hatch
- Space for maintenance

This list is very similar to the design drivers identified at Vuyk in section 2.1.2, with the items more explicitly separated. The convenience of maintenance was also defined by Wu et al. (1998) to be an important objective in engine room design, next to the minimization of the space of the engine room and the reproducibility of the pipes connecting the machinery.

Another factor that Babicz (2015) identifies is the coordination between the design of the engine room and the accommodation block, as it is often located above the engine room. Finally, Babicz (2015) considers detailed engine room arrangement with the specification of machinery and equipment as part of the basic design. As the tool is determined to operate in the concept design phase it is acceptable to limit the level of detail.

Aside from determining why machinery is placed in a certain way, which machinery must be placed and the sizing of this machinery is also an important topic. As mentioned in section 2.1 in practice the equipment that is not yet specified in an early design stage is mainly sized from comparable vessel designs. If this information is not available the sizing of components is done according to rules of thumb known by the designer, or by applying regressions formulas, tables or graphs based on historical data, such as those found in (Klein Woud and Stapersma, 2002).

Stapersma and Vos (2015) developed an first principle based sizing for diesel engines, gearboxes and electrical machines. Hu (2016) expanded this method to include centrifugal pumps and plate, shell and tube heat exchangers as well. This method has two advantages over the traditional regression analyses of machine dimension data as a function of speed and/or power characteristics. It is possible using this method to explore the influence of (future) technology on the dimensions of equipment, and it is possible to introduce an uncertainty analyses to the design process.

2.2.2. Relevant Literature from Other Fields

As only limited literature from the maritime field was available, literature from other relevant fields was investigated as well, as there are other fields that deal with the problem of designing machinery spaces. The closest field that was found with relevant literature available is the field of process plant layout designs. Moran (2017) wrote the book on practical process plant layout design. The information in this section can all be attributed to this source.

Moran defines a process plant as “a complete set of process units and direct supporting infrastructure required to provide a total operational function to produce a product or products”. The discipline of layout design is concerned with the spatial arrangement and its interconnections. The process units and supporting infrastructure contain equipment similar to that found in ship engine rooms, such as pumps, separators, piping and ducts, filters, etc. With these definitions in mind there is a clear similarity between the ship machinery space arrangement problem and the process plant layout problem, which makes the knowledge from this source useful and possibly applicable to this problem.

Moran considers layout design practice to be based in two parts: 1) codes and standards and 2) design experience. While codes and standards form constraints which are necessary to comply to, design experience is more difficult to grasp.

There are several reasons why Moran considers good layout design as important. Moran argues that “good layout practice plays a vital part in the ongoing commercial success of a project, by making the plant safe and efficient to construct, operate, and maintain, while making effective use of the land available”. Moran remarks that a “well-thought-out layout also contributes to successful planning of the design and construction stages of a project. This avoids costly redesigns, both in time and money, in the later stages of the project. Getting the layout right on paper before construction starts will minimize the possibility of this”. This was of course also remarked earlier in section 2.1 in regards to the importance of the engine room design relative to the ship design process as a whole. Moran remarks that “good layout will not compensate for bad process design, but a bad layout can easily lead to an unsuccessful or unsafe plant”.

It is noted that while 2D software is still common in the industry, 3D modelling software has several advantages. It reduces drafting errors and inconsistencies, and makes it easier to spot clashes. It can also simplify coordination between several disciplines or people unfamiliar with engineering drawings.

In the early design stages existing guidelines for separation distances in plant design are a very useful

method to produce designs in a timely manner, without all the specifications needed for more advanced calculation methods.

The earlier mentioned professional experience provides several notes and guidelines for making a good design. There are of course several factors at play, which are grouped under the labels of cost, safety and robustness. Moran considers making “informed compromises between cost, safety, and robustness” the essence of good design.

The theoretical minimum space that a process plant can occupy is greater than the sum of all its components. Various constraints influence this, such as the need for clearances for access during operation, maintenance or construction. Subject to these constraints, according to Moran “the most economical plot layout is generally that in which the spacing of the main equipment minimizes interconnecting pipework and structural steelwork”. Also, as a rule, “an attractively laid out plot with equipment in rows is also economically laid out. However, changes in appearance on aesthetic grounds must not conflict with the requirements of operability, maintainability, or safety”. Process plants on land often have enough space to do this, they often are outdoors on a single level. This is a big difference with engine rooms aboard ships that are mostly allocated in confined spaces.

Moran’s observations on “operational convenience” are also interesting for the design of engine room arrangements in ships. He considers this “a very important factor in achieving safe and reliable operation. , reducing the chances of making mistakes and increasing the probability of a malfunction being detected early. As with operational considerations, the layout designer should arrange equipment to facilitate safe maintenance. Maintenance, which is made safe and easy, is more reliable, is often quicker and saves downtime. In the long run, this provides ample repayment for the thought and care given at the layout stage”. In process plants the equipment is either repaired where it stands, in which case space around the equipment is necessary, or it needs to be able to be decoupled, transported to and from a workshop and recoupled again. This needs space around the equipment as well as a means and route of transport, the same as in ship engine rooms and important to take into account in the development of the design tool for machinery space arrangement.

2.3. Concept Exploration

In the concept design phase (section 2.1.1) one or several concept designs of the vessel are created to meet the requirements set forth by the client. ‘Concept exploration’ is considered to be the process of identifying different ways to meet the known design requirements, and learn from these solutions what good and bad ways are, possibly leading to identifying new design requirements. This section first defines the characteristics of concept exploration tools and then details the two main methods of concept exploration: sequential and concurrent exploration. Finally it discusses how to deal with multiple, possibly conflicting, objectives during the exploration process.

2.3.1. Characteristics of ship concept design tools

Andrews (2011) identified the most relevant desirable characteristics of ship concept design tools. This research relates to naval ship design which has a concept design phase which is typically much longer and requires more elucidation of design requirements than the concept design phase of commercial vessels at Vuyk. Also his work relates to the concept design phase of a complete vessel instead of the specific part of the design of the machinery spaces. Still, the mentioned characteristics concerning ship concept design tools are considered valuable and useful for the development of the tool for machinery space arrangements for Vuyk. Duchateau (2016) summarizes these characteristics Andrews identified as:

- ”Believable solutions, that is, generated solutions should be technically feasible and sufficiently descriptive (e.g., they must obey the laws of physics, the basic principles of naval architecture, and the necessary rules and regulations).
- Coherent solutions, that is, a tool should produce more than a solely numerical description of

performance and cost (e.g., a visual representation of the solution).

- Open and responsive methods, that is, the opposite of a ”black-box” or rigid decision systems. Tools and methods should respond to those issues that are deemed important to the stakeholders.
- Revelatory insights, that is, identifying likely design drivers early on to aid the concept exploration process.
- Creative approach, that is, encouraging radical ”out-of-the-box” solutions and a wide design exploration to push requirement elucidation boundaries.”

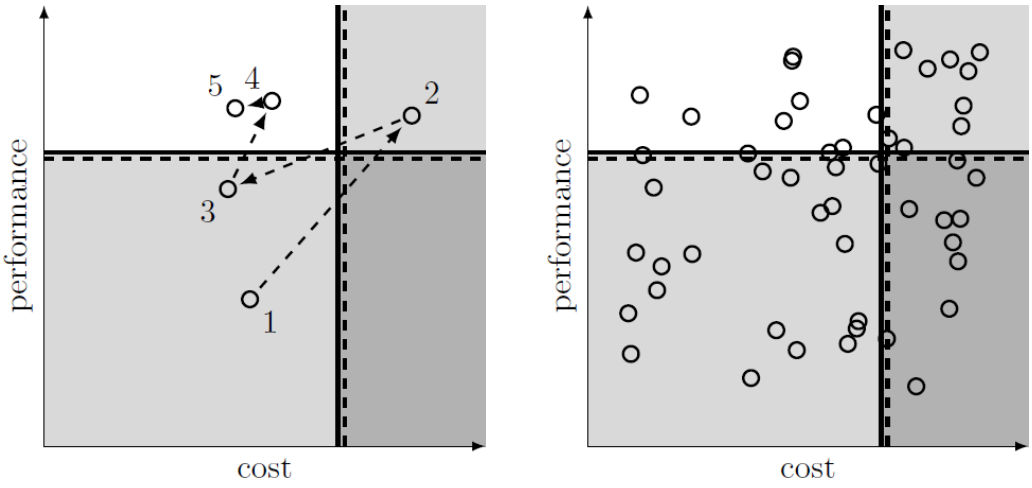
As far as possible the design tool for machinery space arrangements for Vuyk will be developed according to these characteristics: generating feasible solutions, with a visual representation, supporting the designer with an open and responsive method, encouraging to explore various alternative concepts including radical “out-of-the-box”

In the next paragraph the two principal methods of concept exploration are presented.

2.3.2. Sequential and concurrent concept exploration

Sequential exploration follows the traditional design spiral, which Vuyk also uses in their design process and is shown in figure 2.2. A single design is made, from which lessons are learned which are used to alter the or create an entirely new design in the next iteration. As the design process iterates the design space is explored until an acceptable balance of design objectives is found. This is illustrated in figure 2.5a. Creating these alterations or new designs takes considerable effort, which means that only a limited number can be made and this leaves large and potentially high-performing areas of the solution space unexplored Duchateau (2016). Aside from this, the exploration of this solution space depends heavily upon the initial design as this is the starting point for following iterations. Another disadvantage of iterating upon a single design is that it can result in distorted final solutions Van Oers (2011).

This method is in contrast to concurrent concept exploration. With this method a large number of designs are generated at the same time to widely populate the solution space, as is seen in figure 2.5b. Due to the computational power available nowadays this is now an alternative to sequential exploration. However, the models that are used to generate these designs normally lead to solutions that are less detailed when compared to the manually designed sequential solutions. This disadvantage is offset by the larger exploration of the solution space, which covers a larger set of varying design options to explore more regions of potential interest. Interesting designs can be refined to a suitable level of detail in later stages of the concept design phase.



(a) Sequential exploration (b) Concurrent exploration

Figure 2.5: Illustration of sequential and concurrent approaches to concept exploration Duchateau (2016)

While Vuyk mostly uses a sequential approach for the concept design phase as a whole, the design of the engine room is sometimes handled in a more concurrent way by a specialized marine engineer who designs several concepts for this space in an early stage, as was discussed in 2.1.1. However, this is not done for most designs, it depends upon the right personnel being available and the amount of man-hours available for the concept design as a whole and for the machinery spaces in particular.

A design tool that contributes to making this concurrent exploration approach less time-intensive and bringing its benefits to the general engine room design in the concept design phase, would add value to Vuyks design process.

2.3.3. Dealing with multiple-objectives

In the previous sections of this chapter multiple objectives and design drivers are identified, which later will be used to create the objective functions governing the optimization/search algorithms of the tool in section 4.1.3. But first a method to deal with these (possibly) conflicting objectives is needed. Duchateau (2016) identified three ways to deal with conflicting objectives for a optimization/search algorithm:

- a-priori
- a-posteriori
- progressively or gradually

The names of these methods refer to the moment in the exploration process when the relative importance of the different objectives is determined: before the search algorithm is run (a-priori), afterwards (a-posteriori) or gradually during the concept exploration.

The a-priori method combines the several objectives into a single objective function. To successfully do this a-priori information is needed about which objectives are important, and what importance these objectives have relative to each other. A common a-priori method is the weighted sum method, which directly adds all the objectives together, each multiplied with its own weighing coefficient to denote its relative importance. This method then results in a single 'best' solution according to this objective function containing all the weighted objectives.

The a-posteriori method does not need this a-priori information about the relative importance of the objectives. Instead it presents the multiple best solutions along the Pareto-front, leaving it to the designer to judge afterwards the trade-off between the objectives and select the solution that best deals with these. While the name of this method is a-posteriori and the relative importance of the objectives is indeed performed after the optimization, it is still needed to define the objectives that are optimized beforehand.

The Pareto-front that results from an a-posteriori method, is the set of 'nondominated' solutions among all the found solutions during the optimization. A solution is 'dominated' if there is another solution that scores better on all the objectives, and non-dominated if there are not any solutions that score better on all the objectives. This is illustrated in figure 2.6.

An a-posteriori method is preferred over an a-priori method, as the relative importance of different design objectives can vary per design, and it is also leaves the different design objectives separate instead of adding them together. This makes it easier for the designer to extract knowledge from the design space. This means that the optimisation algorithm must be capable of handling multiple objectives at once, if possible presented in a Pareto-front graph.

Progressive methods use the information that a user learns during the optimization process in order to then guide this process based on this information, in a direction that the users deems interesting. Manual sequential concept exploration can be seen as a progressive method, as each new iteration contains the lessons learned in the previous ones. Manually running a concurrent exploration method in which the user alters the objectives and constraints between iterations to account for new insight is also a form of a progressive method.

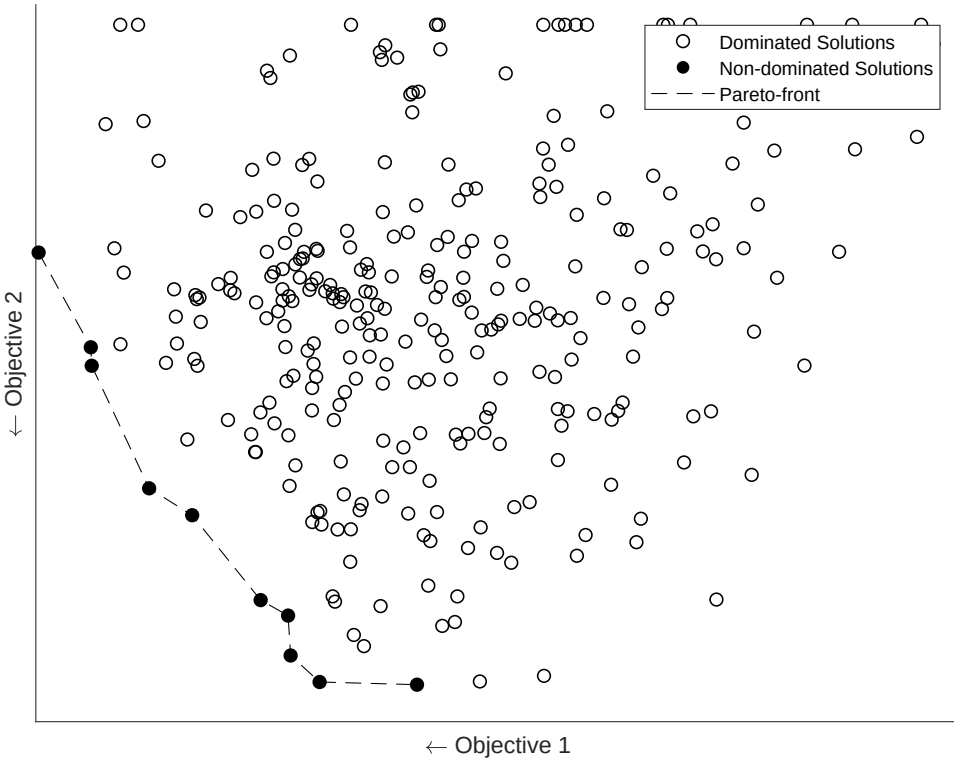


Figure 2.6: Illustration of a Pareto-front

In regards to a-priori or a-posteriori methods, progressive methods can actually make use of elements of either depending on its implementation. Progressive methods are defined by their interaction with the user, which can show itself to tuning the weights of an a-priori weighted sum method between runs or changing the number and definition of objectives and constraints of a a-posteriori method.

The combination of a progressive method with an a-posteriori method seems promising for the design tool to be developed.

2.4. Design Requirements for the Design Tool for Machinery Space Arrangement

From the previous sections in this chapter the design requirements for the tool are compiled. These requirements can be divided in several categories: there are requirements that govern how the tool should be used, requirements about the results from the tool and modelling requirements that need to be met in order to accurately get to these results.

To summarise, the design tool should be:

- Usable early in the design process (Concept design phase).
- Usable by a Naval Architect in an open and responsive way.
- Run in a reasonable amount of time.
- Contain a basic database of reference-components.
- Enable the input of new, specific components.
  - Specific components
  - Generate components from regression formulas or rules of thumb

The desired results should:

- Accurately estimate the necessary space needed.
- Explore the solution space to present several initial layout designs.
- Be technically feasible and visually represented.

Furthermore the model needs to:

- Be able to handle multiple decks
- Be able to handle relationships between components
- Be able to handle several objectives
- Account for the identified main design drivers
  - Accesses and escapes
  - Transport routes and walkways
  - Space for maintenance
  - Separation distances
  - Ventilation

As was established in section 2.1.1 the application of such a tool has the most value early in the design process, in the concept design phase. This means that the tool needs to be able to work with uncertain or limited information, as only a limited amount of problem knowledge is available when the model is used.

In the early design stage there are several sources of information for the components that a user wants to place in the tool: those specified by the client, those already determined by an engineer, and temporary components and 'reserved spaces' approximated from reference vessels, regression formulas or rules of thumb.

In order to accommodate these different sources of information it is important that the user can easily add components manually to the tool. These could then be saved to a database which is then available for future use. To ease the design process regression formulas or rules of thumb for several component types could also be programmed.

From a work-flow perspective it is important that the runtime of the tool after the input has been selected is reasonable. Generally a runtime of seconds or minutes is not considered disruptive, and allows for quick iteration and tuning by the user to explore the solution space of the problem and achieve desirable results. However, if the runtime is quite long (more than a few hours), it becomes feasible to run it overnight without disrupting the normal work-flow. Such a large runtime does not allow for easy iteration, so if the runtime is that large the solution space of the problem should be sufficiently explored to only necessitate the one run to achieve workable results.

The desired goal of the tool is to facilitate design exploration for the engine room layout design, to bring more detail to the concept design stage without necessitating more man-hours in order to reduce the risk of costly redesigns later in the design process. In order to do this is, it is important that the area needed by the selected input set of machinery is accurately estimated, and that the design space is sufficiently explored by the generated layout designs.

It is acceptable to simplify the hull shape to a rectangular form. Components placed in the layout can be simplified to rectangular blocks as well.

An engine room is often spread over multiple decks, and the model should support the designer to generate alternative arrangements for the different decks.

As was discussed in section 2.3 a model which can handle multiple objectives is desired to better facilitate concept exploration.

Finally in order to create a usable design for the machinery space arrangements, the various design drivers discussed in sections **??** and **??** need to be accounted for.

It is considered quite ambitious to develop a design tool that fulfils the complete set of design requirements as mentioned above within the time frame of a Master Thesis. In the next section a prioritization in requirements is discussed to limit the scope.

2.5. Limiting the scope

With the design requirements set up it is important to prioritize these. Because of the inherent time constraints of the thesis research it is unlikely that all the identified requirements can be met. In order to limit the scope it was decided to a list of considerations is presented, which touches upon some design requirements set up for the tool and some other considerations presented in the earlier sections:

- The focus of this research is the arrangement problem, and as such the selecting and sizing of components and the creating of a comprehensive database of input components is not considered to be a part of the tool.
- As a large part of the vessels Vuyk designs are diesel-electric vessels, and these type of engine rooms allow for relatively less constrained arrangements, conforming to the hull shape is not necessary.
- The cost of an arrangement is not quantified and thus ignored as a metric.
- As the tool will aid in creating early concept designs, connections between components such as pipes are not routed by the model.
- The model will initially consider only a single deck.
- As Vuyk works with 2D arrangements in the early design phase the tool does too.
- while separation distances identified as important and widely used metric in process plant layout design literature, this is not used in the maritime industry and thus this data is not readily available for the engine room components. This design driver is neglected as it is also considered superfluous when an necessary space for maintenance around a component is already defined.

3

Solution Directions

With a clearer understanding of the problem that needs to be solved to develop a design tool for machinery space arrangements , a solution direction can be identified. In this chapter first the best way to model a machinery space is researched. After a modelling method has been decided upon, the method to solve this model is investigated.

3.1. Models for layout problems

As the problem that this research focusses on is a layout arrangement problem, first the best method to model model a machinery space is researched.

3.1.1. Ship Arrangement Problem

A natural starting point for the search of a possible solution was Van Oers (2011), who performed his PhD thesis on the automatic generation of ship concept designs using an geometric packing approach to model the problem and a genetic algorithm to search the solution space for feasible arrangements. While the focus is of course on the whole vessel instead of a single room, there are clear similarities in the problem. Due to the iterative nature of the design process van Oers used an optimization algorithm (called a search algorithm in his work, as it is used to search the solution space). In order to utilize this algorithm a design process was proposed as seen in figure 3.2.

Van Oers (2011) remarked on the iterative nature of the ship design process, which he laid out in figure 3.1. This is a iterative loop that either a designer or software can follow to create the description of a design that meets certain design requirements. A description of the vessel is created, after which the performance of this is predicted. The performance is analysed against a set of predefined design requirements, after which it is decided whether the design meets the requirements or whether it needs to be adapted.

When this process is automated an algorithm decides on the design change if one is deemed necessary, after which it alter or creates a new design description. In order for the algorithm to do this, a parametric model of the design is needed so that the algorithm can change the design by altering the parameters of the description. The process van Oers designed can be seen in figure 3.2. Here the process is initialized by generating a random set of input parameters, which are used to create the first description of the design when they are plugged into the parametric model of the ship. The ship performance of this ship description is then analyzed using certain performance prediction tools. The relevant information from these tools is then used to calculate the ratings for the design, which take the form of objectives and contraints. If a stopping criterion is not met, these objectives and constraints are used by an

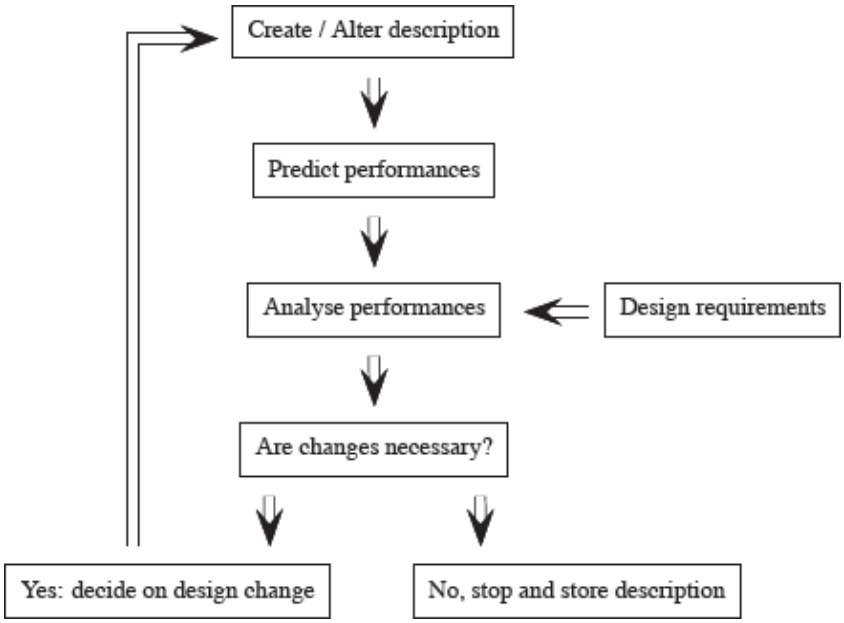


Figure 3.1: The simple design loop as laid out by Van Oers (2011)

optimization algorithm (called a search algorithm by van Oers) to create a new set of input parameters, which are then used to begin the loop anew until the stopping criterion is met.

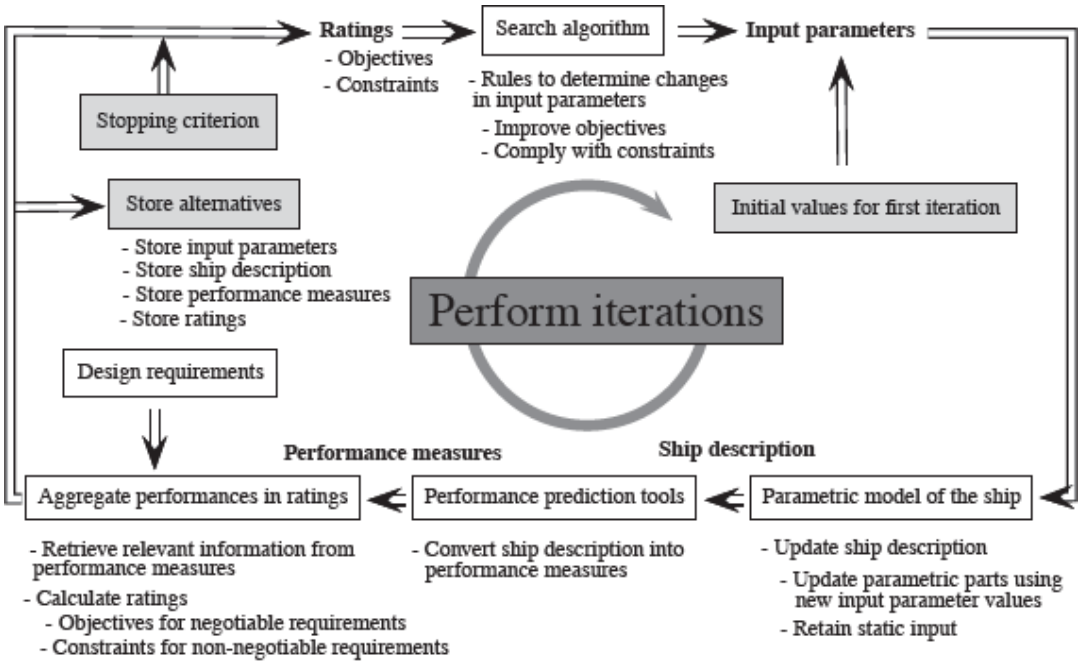


Figure 3.2: The design process using a search algorithm as laid out by Van Oers (2011)

In order to use this approach several elements need to be developed or selected:

- A parametric description of the design
- A way to analyse the performance of the design
- A way to aggregrate these performances into ratings
- The search/optimization algorithm

This method was developed by van Oers and later this research was continued by Duchateau (2016),

who used it as the basis for the Interactive Evolutionary Concept Exploration Method (IECEM). In this method the designer can interactively steer the concept exploration effort based on generated designs which are considered desirable and promising by updating the objective functions throughout the design process.

Unfortunately the code that is available from these projects is considered to be unusable for this research. Understanding the thousands of lines of code, dozens of special functions, rebuilding or adapting the parametric model and performance prediction tools is an infeasible challenge within the timeframe of an MSc-thesis.

In his research Gillespie (2012) argues that the ship arrangements problem is more about managing relationships than about managing space (exceptions aside, such as the submarine); managing the relationships among shipboard elements is considered the true challenge of integrating compartments and systems into a cohesive and functional early-stage ship layout. By understanding and working with these relationships, feasible layouts should be able to be generated with less iteration.

Gillespie states that there is no “right” arrangement, only good and bad ones. As an experienced naval architect is assumed capable of identifying a quality layout upon seeing it, he desired to keep the human designers in the loop. Rather than simply accepting the single highest-scoring solution from an optimization, a set of high-quality solutions are retained to be analysed by human designers.

3.1.2. Packing problem

The packing problem is a general name for a set of similar problems. Packing problems are closely related to cutting problems and together they appear under a variety of names in literature, such as the cutting stock or trim loss problem, bin or strip packing problem, vehicle, pallet or container loading problem, nesting problem, knapsack problem, among others (Dyckhoff, 1990). The classic bin packing problem considers how various objects of different sizes can be packed into a finite amount of containers in a way that minimizes the number of containers used, or the size of the container.

Van Oers (2011) describes geometric packing as placing several geometric objects in a larger space, such that 1) all the objects overlap completely with a larger positioning space, i.e. they are packed within an envelope, and 2) all objects prevent unwanted overlap among themselves so that geometric objects cannot unintendedly occupy the same space. A simple visual illustration of geometric packing can be seen in fig 3.3.

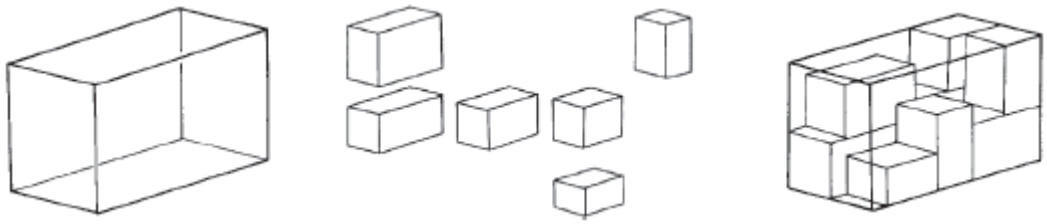


Figure 3.3: Packing example of the container loading problem. Left is the packing envelope, centered are the objects to be packed, and right is a possible solution. Adapted by Van Oers (2011) from Dyckhoff (1990)

Van Oers (2011) uses geometric packing to create a ship description in the ship arrangement problem, which was further discussed above.

Fadel and Wiecek (2015) investigated layout optimization in 3D for realistic engineering problems. They describe the problem of packing 3D free-form objects within a specific free-form envelope as the most general case of the configuration design problems involving multiple constraints, multiple criteria, and mixed continuous and discrete variables. The constraints and objectives are also a combination of linear, quadratic, non-linear, multi-modal, continuous or discontinuous functions which are often not analytically available. They advocate a clear divide between the objects, anatomic solids which cannot be taken apart, and the system, defined as an aggregation of objects that can move with respect to each

other. The objective of the problem is to design the system, placing the components inside an envelope, while satisfying a set of constraints and maximize one or a set of objectives. Problems are divided between compact packing, which only objective is minimizing the volume, and non-compact packing. For non-compact packing the objectives in their engineering problems are system level characteristics, such as volume, inertia, heat transfer or maintainability.

3.1.3. Facility layout problem

The Facility Layout Problem (FLP) has been investigated by researchers in various different engineering fields (Lee et al., 2005) over the past few decades and as such knows several definitions. Drira et al. (2007) defines a facility layout as an arrangement of everything needed for production of goods or delivery of services. A facility is here defined as an entity that facilitates the performance of any job. The FLP seeks to produce a facility layout such as to best satisfy its goal. This can for example be done by reducing material handling costs, work in process, lead times, utilizing existing space more effectively, making plants adaptive to future changes, or provide a healthy, convenient and safe environment for employees Ahmadi et al. (2017).

Layout problems are known to be complex and like the geometric packing problem, the facility layout problem is known to be NP-Complete (Ahmadi et al., 2017; Drira et al., 2007). This means that for these problems there is no known optimal algorithm that runs in a reasonable amount of time. However, while algorithms cannot guarantee that the optimal solution is found, good solutions can still be found within a reasonable time frame.

The FLP could be considered a special case of the packing problem, as the basis is still positioning a certain number of predefined objects within a certain packing envelope. However, the FLP is always considered as a two dimensional problem, and can have different or more objectives than just minimizing the packing envelope. The main focus of the research for the FLP has been applied to manufacturing applications, where material handling costs are a significant factor in the total operating expenses. As such, minimizing the material handling cost has been considered as the common objective in the literature (Ahmadi et al., 2017), although other objectives have also been considered depending on the problem that is modelled. Some other common objectives are, depending on the application of the FLP: adjacency relationships, installation costs, building costs, rearrangement costs, and others (Ahmadi et al., 2017; Drira et al., 2007)

The FLP has been used to approach a variety of problems in various different engineering fields, and as such knows a wide variety of modelling and resolution approaches. Figure 4.2 in the next chapter shows a tree representation of research efforts into these approaches for the multi-floor facility layout problem (MFLP), which is a variety of the general FLP which arranges the facilities on multiple floors. While most research concerns single-floor facility layout problems, research into the multiple-floor variation of the facility layout problem has become increasingly popular as it is more applicable to the real-world scenarios (Ahmadi et al., 2017).

3.1.4. Choosing the Most Suitable Model

The Facility Layout Problem has been identified as a very suitable problem for modelling the machinery spaces that are the subject of this research. It has mainly been developed for manufacturing applications, which also deal with the arrangement of specialized machinery. Thus, it is easily able to comply to several of the identified design requirements for the tool (section 2.4), such as the handling of relationships between the components, incorporating multiple objectives and handling multiple floors. It also has predefined ways to deal with several of the identified design drivers, such as the space for maintenance. As it the method is regularly used to deal with machinery, there are known ways to deal with the modelling specifics of machines.

For these reasons the Facility Layout Problem is chosen as the modelling method for the machinery space arrangement problem.

3.2. Resolution Approaches

With the Facility Layout Problem selected as the way to model the machinery space arrangement problem, the way to solve this model is now investigated. In figure 3.4 an overview of the various resolution approaches for the multiple-floor facility layout problems is shown. The multiple-floor version of the facility layout problem is a special instance of this problem, which is considered significantly more complex. While the research in this thesis is limited to a single-floor application, a method that is suitable to be adapted to the multiple-floor instance of the problem is chosen to make further development of the tool possible.

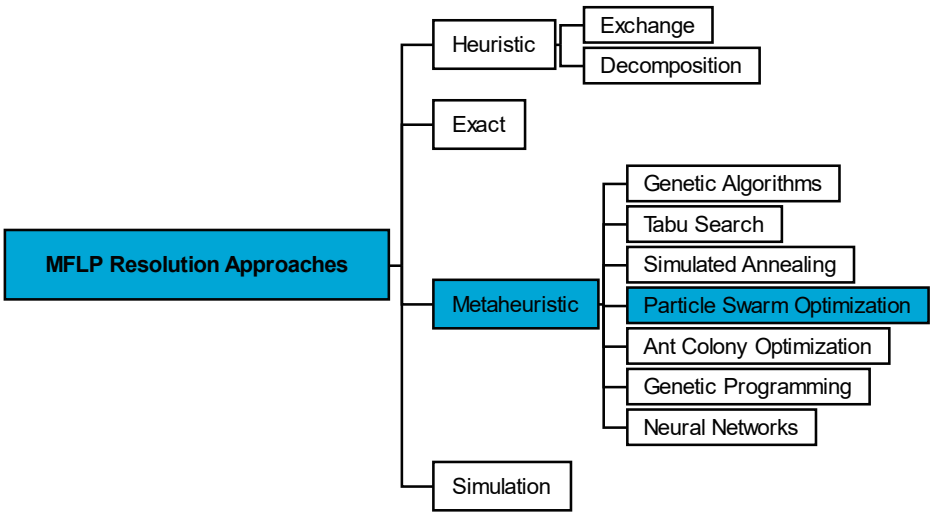


Figure 3.4: Tree representation of resolution approaches for the multiple-floor facility layout problem (MFLP). Adapted from (Ahmadi et al., 2017; Drira et al., 2007).

While several categories of resolution approaches are identified and shown in figure 3.4, they are not all well researched in literature, applicable to every instance of the facility layout problem, or able to find a good solution within a reasonable time-frame.

Layout problems are considered to be complex, with a large amount of variables and constraints, and a very large solution space. As mentioned above both packing and the facility layout problem are NP-complete. As such, metaheuristics are a good way to approach this optimization, to get a sufficiently good answer within a reasonable amount of computation time. Metaheuristics sample a set of solutions from the solution space which is too large to be completely sampled within a reasonable amount of time.

There are a host of metaheuristics available, particularly those inspired by nature and human behaviour. Examples of newer ones include those inspired by cuckoos, bees, the formation of galaxies, musical composition or even colonisation by imperial nations. However, these relatively recent metaphor-inspired metaheuristics have been criticised by the research community for failing to be significantly better then other metaheuristics while distracting researchers from finding the best way forward in this field of study Brownlee and Woodward (2015).

As such, three of the relatively older modern metaheuristics are considered for this research, that are well understood and have proven themselves in an engineering environment. These are the genetic algorithm (GA), particle swarm optimization (PSO) and simulated annealing (SA). Various implementations off these algorithms are readily available as open source projects, or are already implemented in popular programming environments such as MATLAB or Python.

Genetic algorithms, first proposed by Holland (1975), are inspired by the natural selection process, where a population of solutions is evolved toward better solutions. These solutions have been encoded into chromosomes. Each iteration of the algorithms is called a generation, and by manipulating the solutions over successful generations the population is evolved towards a good solution. Biological inspired operations are used in this evolution process. Selection is used to promote solutions with ”better” objective function scores, crossover between solutions is used to find new and hopefully better solutions, and sometimes solutions are randomly mutated. Sometimes elitism is used to keep the best solutions in the population over successful generations.

Simulated annealing was introduced by Kirkpatrick et al. (1983). It takes its inspiration from the annealing of metals: if a metal is quenched quicky, atoms are locked into high-energy states that are local optima, while if a metal is annealed slowly, atoms will escape from these local optimum solutions and move to the global optimum. For the algorithm, this translates to allowing the occasional ”hill-climbing”, e.g. accepting a worse solution than the current best in order to escape local minima or optima. The chance that a worse solution is accepted is related to the ”temperature” variable. At the start there is a high temperature, and thus a high likelihood that a worse solution is accepted in order to escape a local minimum. As the solution is slowly cooled, the algorithm becomes more picky. The way the solutions are cooled is called the cooling scheme. Unlike genetic algorithms or particle swarm optimization, the simulated annealing algorithm is not population based, i.e. it only calculates one solution at a time.

Particle Swarm Optimization is a somewhat more recent metaheuristic, first introduced by Kennedy and Eberhart (1995). It is based on a behavioural analogy approach to minimization, where the algorithm models swarms of animals such as a flock of birds, a swarm of bees, or a school of fish. As such, it is a population based algorithm, like the genetic algorithms. At each iteration of the algorithm each member of the swarm is evaluated, after which it moves in the direction dictated by a combination of its inertia, its own personal best previous position in the solution space, and the best known position based on the knowledge of the whole swarm.

While genetic algorithms and simulated annealing have been more applied to both packing and facility layout problems (Ahmadi et al., 2017; Duchateau, 2016; Fadel and Wiecek, 2015; Lee et al., 2005; Van Oers, 2011; Wu and Appleton, 2002), particle swarm optimization has been applied to the facility layout problem and is noted as a promising candidate for further research (Ahmadi et al., 2017).

Choosing the Right Resolution Approach

After investigating these various resolution approaches for the Facility Layout Problem the decision was made to use an particle swarm optimization algorithm. This decision was based on the following arguments:

- It is of comparable speed (if not more so) to the other metaheuristics.
- It is relatively easy to implement.
- It only has a few tunable parameters in the algorithm.
- The solution vector is easy to work with.
- As a population based algorithm it is relatively suitable for concept exploration as a search algorithm instead of a pure optimization algorithm.
- It is identified as a promising algorithm to solve the facility layout problem.

3.3. Chosen Modelling Method and Resolution Approach

With the modelling method for the engine room arrangement problem determined as the facility layout problem and the resolution approach for this model chosen as the particle swarm optimization, the specifics of these two methods can be defined and the tool can be built. The description of these specifics is found in the next chapter.

4

Model Description

In this chapter first the Facility Layout Problem for the machinery arrangement is defined in section 4.1. This is followed by the definition of the objectives and constraints of the optimization problem. Then the Particle Swarm Optimization algorithm which is used to solve the defined model is described in section 4.2. Finally, the way the user interacts with the model is described in section **??**.

4.1. Defining the Facility Layout Problem

With the Facility Layout Problem chosen as the model for the engine room layout problem, it is important to define the specific instance of the facility layout problem that best describes this problem. Figure 4.2 gives an overview of the various options when modelling a problem as a facility layout problem, with the choices made for this thesis highlighted in blue. This overview has mainly been adapted from Ahmadi et al. (2017) which concerns the multiple-floor facility layout problem. While the model will initially concern only one floor, in order to facilitate adapting the model to multiple floors in a later stage the research on the multiple-floor facility layout problem is taken into account while developing the model for this thesis.

It is important to use a correct distance metric, as is seen in figure 4.2 the rectilinear, or city-block distance, is generally used in the FLP. It has a very easy implementation and somewhat accounts for the movement around other facilities, as they are considered impassible. This is illustrated in figure 4.1.

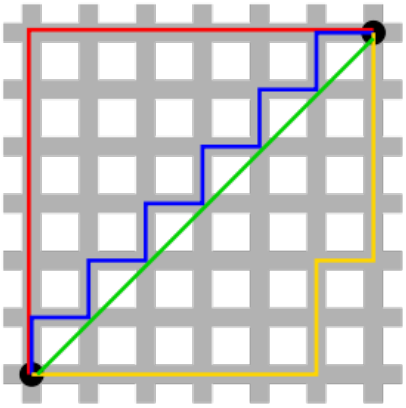


Figure 4.1: Figure illustrating Manhattan versus Euclidean distance. The red, blue, and yellow lines all have the same length (6+6=12 blocks), whereas the green line has length $\sqrt{6^2 + 6^2} \approx 8.4853$ blocks. (Psychonaut, 2006)

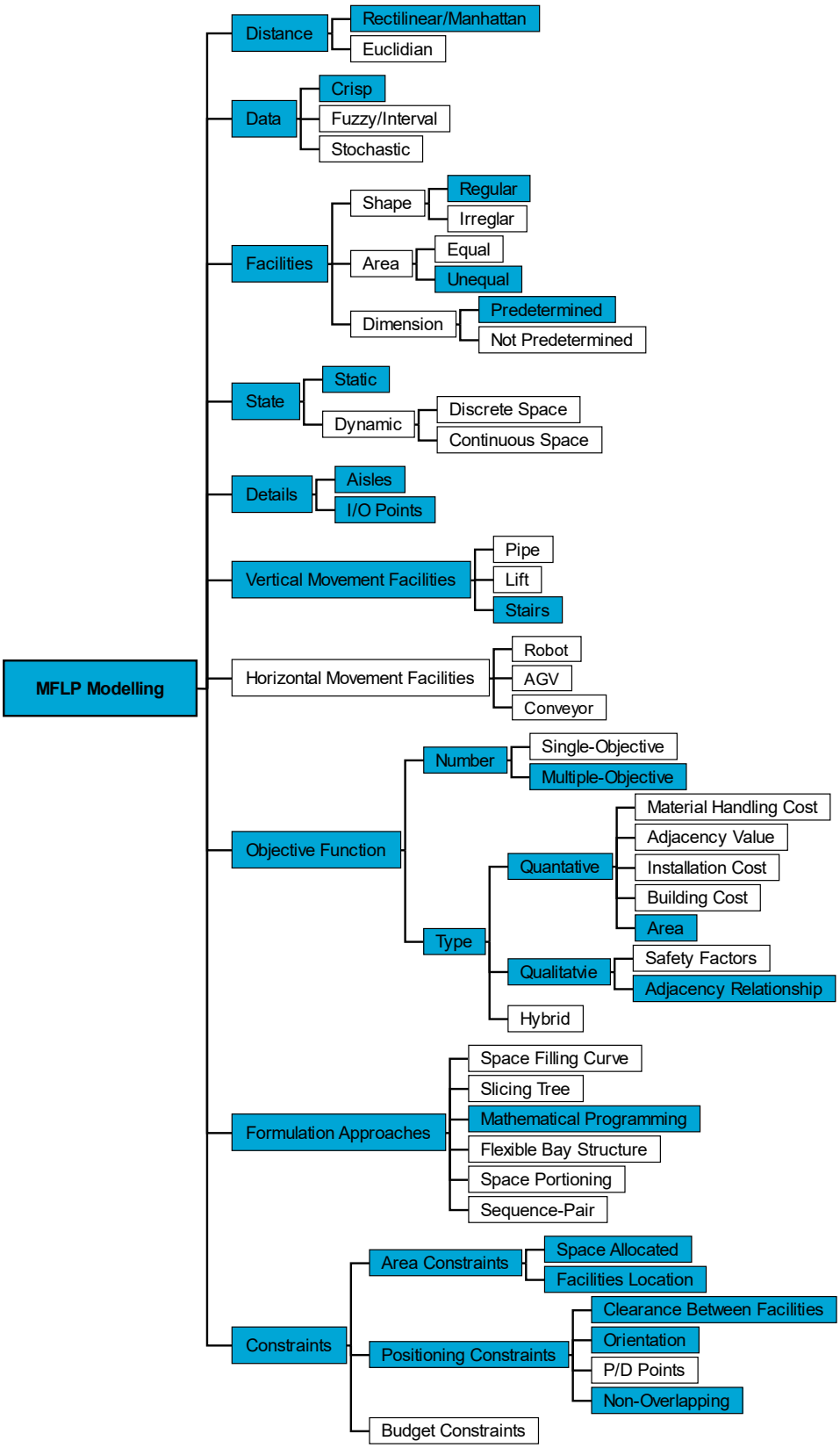


Figure 4.2: Tree representation of modelling choices in the multiple-floor facility layout problem (MFLP). The selected choices are highlighted in blue. Adapted from (Ahmadi et al., 2017; Drira et al., 2007).

The data used in this research is crisp, as data on specific or reference components is considered known. Fuzzy data to qualify the uncertainty in component selection or the relations between components has been considered, but decided against as the advantages do not weigh up against the (time) cost of implementation.

The facilities can be simplified to a regular rectangular shape. As the components are not all the same they have unequal areas, and as the components are defined before the optimization is run the facilities have predetermined dimensions.

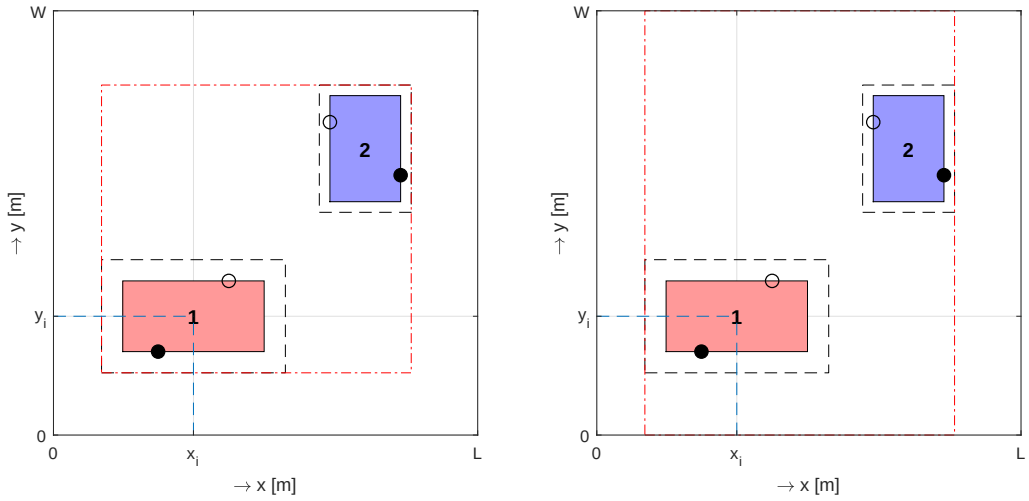
The facility layout problems distinguishes two states, the static and dynamic ones. In the dynamic facility layout problem the objectives or contraint can vary over time, e.g. a manufacturing hall which changes the product it produces. In the static facility layout problem these are constant. This problem is static.

Both of the details, aisles and in- and output points are present. When in and output points are present their location is used for the distance calculation between the facilities, instead of the centre of the facilities which is normally used. By using these the orientation of the facilities can also be accounted for in the FLP, e.g. engines who face forward as they receive fuel from that direction and output exhaust gasses to and exhaust gas uptake defined behind them.

Aisles are of course important as the need for walkways and transport routes trough the machinery space has been identified in sections 2.1.2 and 2.2.

Facility layout problems can be formulated both discretely and continuously. Discrete formulation simplify the problem but also limit solution space significantly. As the dynamic multiple-floor facility layout problems are very complex these are only modelled discretely. However, for static problems most recent research use continuous formulations due to the limits of discrete formulations (Ahmadi et al., 2017).

Mathematical programming outperforms the other continuous formulation methods. It easily allows for the incorporation of various constraints of the problem, such as fixed location of facilities. In mathematical programming various properties of the facilities which are relevant to the problem such as positions and orientations are represented by variables.



(a) The envelope minimizes both the length and the width of the machinery space.

(b) The envelope only minimizes the length of the machinery space.

Figure 4.3: Example of two modelled components and the area envelope.

A simple reprentation of two components can ben seen in figure 4.3. The dimensions of the overall area are defined by the length L and width W of the engine room, placed on the x- and y-axes. The

position of a facility i is determined by the coordinates x_i and y_i of its centre. The actual space needed for the configuration is shown by the red dotted line around the components. This can either show the minimum area, as can be seen in the left side of the figure, or only the minimum length of the engine room. As most engine rooms cover the whole width of their ships this can be a more useful representation of the area needed for the engine room.

4.1.1. Component Properties

Figure 4.4 show a facility in more detail. A facility i has a length l_i and a with w_i. Around the facility a clearance area is defined by clearance CL_i, in order to account for the space needed around component. Several reasons for this clearance space are identified in sections 2.1.2 and 2.2, such as a maintenance area or space need for safe operation.

With the implementation of some component types it became clear that, while it is acceptable to simplify these components to rectangular boxes, having the same clearance on each side of these components leads to large differences with actual designs. This is because some modelled components, notably switchboards and plate-coolers, have their clearances mainly for maintenance purposes. This maintenance is performed on a specific side of the component, on which space is needed. The other sides of the component are however allowed to be close to other components or walls. By reimplementing the clearance in a way that it can be separately defined for each side of the components reality is better approximated and actual designs are better modelled for comparison without incurring the penalty function. An illustration of the clearance can be seen in figure 4.4b.

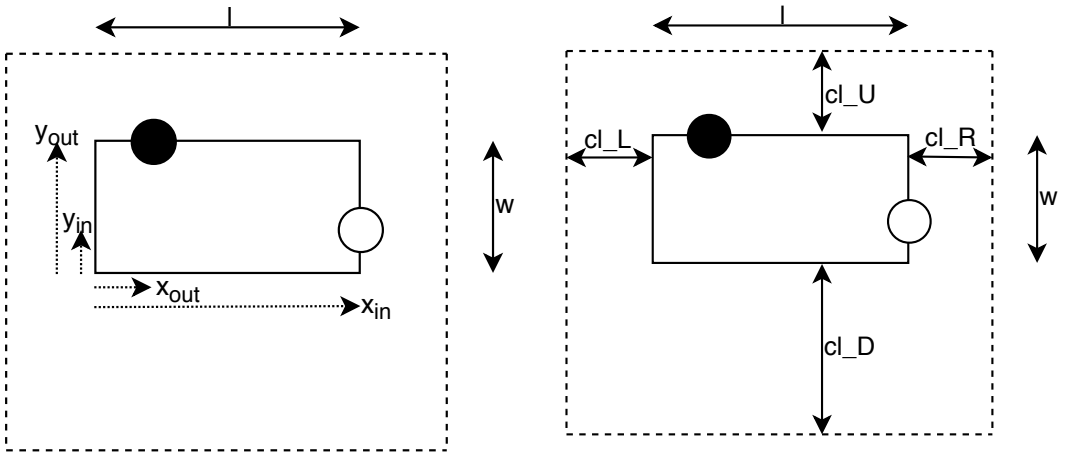
Each facility has an input and an output gate, whose location is defined as a percentage of the length as width of the facility by the variables x_{in}, y_{in}, x_{out} and y_{out}. This is illustrated in figure 4.4a, where for example $x_{in} = 1 * l$ and $y_{in} = 0.33 * w$. These input and output gates are the connection points from which the distances to the other components are calculated. While the terminology input and output is used, since they are defined by the user modelling actual components as facilities for the model the user can choose what connection point they represent. When a more complex component which itself has more than one input and one output is simplified for the model it could be more accurate to view these input and output gates as connection points instead.

Aside from these visually present variables several other variables are defined for each facility. Every facility has an rotation r, which defines whether is is 0, 90, 180 or 270 degrees rotated. A weight g can be assigned so that the centre of gravity can be automatically calculated.

If it is necessary for a facility to be located in a certain part of the floor then the position can be constrained with the $x_{min}, x_{max}, y_{min}$ and y_{max} properties. Similarly the rotation can be constrained between r_{min} and r_{max}.

Interfaces

There is one special component type, which is the interface. An interface is the connection of the defined engine room with the rest of the vessel and as such is almost always located on one of the limits of the defined room. Because of this, interfaces are ignored when calculating the envelope of a layout, as they would otherwise set the envelope on the limit they are placed on.



(a) Length, width, and in- and output locations of a component

(b) Clearances around the component are defined up, down, left and right of the component

Figure 4.4: Illustration of several of the component properties

Relationship Matrix

When defining a components it is important to note that these do not exist in isolation. They have relations to the components around them, or to the vessel trough an interface, that influence their placement in the layout design. These relations are often physical connections, for example the piping between a pump and a heat exchanger. But they are not always such a literal physical connection, for example the need for certain components to be located near each other from a maintenance perspective, or when components have a different working temperature so that they necessitate placement together in a separate room.

In this model these relations are implemented using a qualifying relationship matrix. Such a relationship matrix, sometimes also called a relationship chart, is a common approach for qualifying relations in layout design. Evans† et al. (1987) concludes that layout design algorithms that make use of a relative, qualifying relation input for instead of estimated material flows or connection costs, quantifying relations, are generally easier for designers to use because of the less detailed input that is required. An example of a relationship matrix can be seen in figure 5.3, which shows the relationship matrix defined for the components that serve as input in the case studies performed in chapter 5.

For the values in the relationship matrix those proposed by Lee et al. (2005) are used, which are give in table 4.1. The defined relationship matrix is used by the first objective function to score the found layouts and thus guides the optimization algorithm.

RM_{ij}	Meaning
0	it is *undesirable* for facilities i and j to be located close together
1	it is *unimportant* for facilities i and j to be located close together
2	it is *ordinary* for facilities i and j to be located close together
3	it is *important* for facilities i and j to be located close together
4	it is *especially important* for facilities i and j to be located close together
5	it is *absolutely necessary* for facilities i and j to be located close together

Table 4.1: Values of the relationship matrix and their meaning Lee et al. (2005)

The value of RM_{ij} is the connection value between the defined output of component i and the input of component j. As the components and their input and output gates are defined by the designer, it is up to them to define the relation or connection that is present between the input and output gates of these components and what values these should have. This is one of the main ways that the designer can input their design rationale into the model.

4.1.2. Defining the optimization problem

The Facility layout problem is an optimization problem. An optimization problem can be stated as (Rao, 2009):

$$\text{Find } X = \{x_1..x_n\} \text{ which minimizes } f(X) \tag{4.1}$$

subject to the constraints:

$$g_j(X) \leq 0, j = 1, 2..., m \tag{4.2}$$

$$l_j(X) = 0, j = 1, 2..., p \tag{4.3}$$

Here X is the design vector with n variables, $f(X)$ is the objective function, $g_j(X)$ are the inequality constraints and $l_j(X)$ are the equality constraints of the problem.

The design vector X contains all the variables of the problem. The layout of the FLP is determined by the variable positions of the defined components. Each component has three properties that define its location in the layout: its x- and y-coordinates and its r variable that controls its rotation in steps of 90 degrees. Therefore the constructed design vector for the problem with n defined component is:

$$X = \{x_1..x_n, y_1..y_n, r_1..r_n\} \tag{4.4}$$

Where first all the x-coordinates, then all the y-coordinates and finally the rotations of each component are defined. As each component has three properties, the problem has $3n$ dimensions.

In section 2.4 the various design requirements are summarized, with among them the various design drivers and component relations. In order to incorporate these in the model they need to be added as either objectives or constraints. Objectives relate to the things that the designer wants to archieve with a design, such as a minimal needed area. Constraints are things that the model needs to comply to, e.g. two components cannot physically be in the same space. The following sections discuss these objectives and constraints for the model.

4.1.3. Objective functions

The general formulation of the optimization problem in equation 4.1 optimizes a single objective function $f(X)$. However, as was discussed in 2.3.3 in order to comply to the design requirements and facilitate solution space exploration multiple objectives are desirable. Therefore the general optimization statement is adapted to:

$$\text{Find } X = \{x_1..x_n\} \text{ which minimizes } \sum_{i=1}^{k} f_i(X) \tag{4.5}$$

subject to the constraints:

$$g_j(X) \leq 0, j = 1, 2..., m \tag{4.6}$$

$$l_j(X) = 0, j = 1, 2..., p \tag{4.7}$$

In which $i = 1..k$ objectives $f_i(X)$ are optimized at the same time. With this adaption in place the objective functions can now be defined.

Two objective functions are chosen. The first objective guides the relative placement of the facilities with respect to each other, and the second objective strives to minimize the total area of the layout arrangement.

First objective: Connection Cost

One of the reasons for choosing the Facility Layout Problem is the modelling method was its ability to handle relations between components. Defining these relations allows the designer to input his or her design rationale into the model.

In the Facility Layout Problem this can be handled by an objective function which implements a relative weight factor between the placement of facilities. A typical formulation for such an objective function for a continuous FLP with defined in and outputs is given by Drira et al. (2007):

$$Minimize: f_1 = \sum_{i=1}^{N}\sum_{j=1}^{N} RM_{ij} * d_{ij} = RM_{ij} * (|x_{out} - x_{in}| + |y_{out} - y_{in}|) \tag{4.8}$$

The variable d_{ij} is the distance between the output of component i and the input of component j. As can be seen the formula calculates a rectilinear distance, which is illustrated in figure 4.1.

Here RM_{ij} is the relative weight factor that denotes the importance that the distance between the output of component i and the input of component j is minimized, which is discussed in section 4.1.1. By constructing the relationship matrix RM the designer can input his design rationale of which components need to be close to each other. For most relations between components this will be heavily influenced by the cost of the necessary connections between two components, e.g. the cost of the fuel line between an engine and a fuel pump.

For each output-input pair the distance is calculated and multiplied by the user defined relative weight factor of the connection. As the values of the relationship matrix are constant in a single run of the model, it can only minimize the objective function by reducing the distance between the right output-input pairs.

In the special case that no in- and output locations are defined the distances are calculated between the centres of each component and the relationship matrix RM is symmetrical. However, when in- and outputs are defined RM is not symmetrical as the rotation of each facility influences the location of the in- and outputs and thus also influences the distance. By defining the relation between in- and outputs the rotation of the facilities becomes an influencing variable for this objective.

Second objective: Unused Area Ratio

An important objective identified in chapter 2 is accurately assessing the space needed for a layout. Space often comes at a premium, and space not used by necessary systems in the engine room can be used for actually revenue generating systems or other optimization of the overal design of the vessel. This desire for an minimum area is offset by the risk of costly redesigns in later design stadia if too little space is assigned during the concept design space. As such, minimizing the area that the components lends itself excellently to an objective function, where the designer can interpret the results and see the effects of certain area sizes upon the layout. Yarpiz (2015) defines an area minimization objective which is used here:

$$\text{Minimize: } f_2 = \frac{A_{unused}}{A_{envelope}} = \frac{A_{envelope} - A_{components}}{A_{envelope}} \tag{4.9}$$

$$A_{componenents} = \sum_{i=1}^{N} l_i * w_i \tag{4.10}$$

$$A_{envelope} = l_{envelope} * w_{envelope} \tag{4.11}$$

In which the envelope is defined as shown in figure 4.3. This objective calculates the ratio between the enveloped area that is not used by the facilities and the total area enveloped by the facility layout.

As the total area of the components is constant, the only way to minimize this function is to place the facilities closer together to reduce the total enveloped area of the facilities.

Restating the first objective function

It is desirable to make sure that the value of the objective functions is in the same order of magnitude, and independent of the problem size (dimensionless). This is because penalty functions will be used to force compliance to the constraints. If the values of the objective functions is in the same order of magnitude then the same penalty function can be used for both objective functions. Furthermore, if the values of the objective functions are independent of the problem size the penalty functions will not need to be tuned again when the size of your problem changes.

The size of the defined problem influences equation 4.8 by of the amount of components, which affects C_{ij}, or the size of the placement area, which affects the distances in d_{ij}. As equation 4.9 is a ratio the value of this objective will always be between 0 and 1. The first objective function will be adapted to obtain values of the same order of magnitude. This can be done as the objective is a qualitative one, and does not quantify an actual cost.

In order to scale equation 4.8 both the relationship matrix variable C_{ij} and the distance variable d_{ij} need to be scaled. The distance calculation part of the objective also needs to be made dimensionless. This is accomplished in the following way:

$$\text{Minimize: } f_1 = \sum_{i=1}^{N}\sum_{j=1}^{N} \frac{RM_{ij}}{RM_{tot}} * \frac{d_{ij}}{L+W} = \frac{RM_{ij}}{RM_{tot}} * \frac{(|x_{out} - x_{in}| + |y_{out} - y_{in}|)}{L+W} \tag{4.12}$$

where

$$RM_{tot} = \sum_{i=1}^{N}\sum_{j=1}^{N} RM_{ij}$$

4.1.4. Constraints

The facility layout problem poses several constraints inherent to layout problems, concerning the physical placement of the components in the space: two components cannot occupy the same space and thus may not have overlap, and components cannot be placed outside of the limits of the defined space.

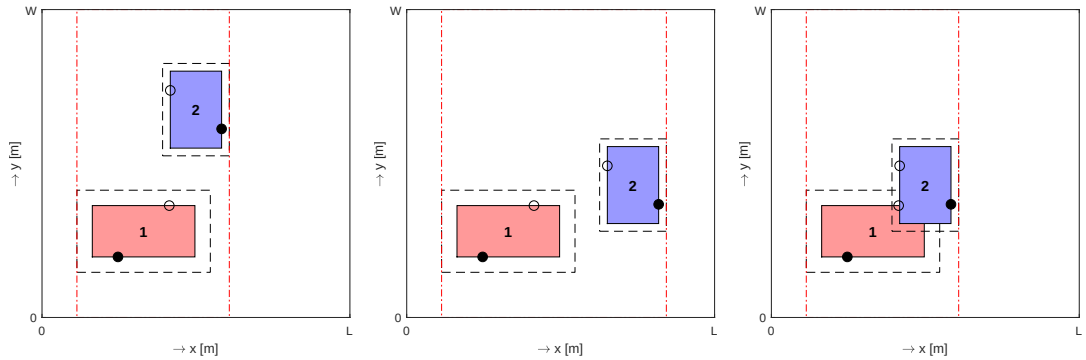
Overlap of components

Yarpiz (2015) proposes equations 4.13 - 4.15 to calculate the overlap between two components i and j. This constraint allows the clearances of the components to overlap, but not the components themselves. The overlap on both the x and y axis is calculated, and if overlap exists on both axes the constraint is violated. This is demonstrated in figure 4.5. As the clearance of the facilities can be uneven the model is adapted to this change. This is done by determining the relative position of the two components for which the overlap is calculated beforehand, and then inputting the relevant clearance for the component into the formula.

$$A_{ij} = max(\Delta X_{ij}, \Delta Y_{ij}) \leq 0 \tag{4.13}$$

$$\Delta X_{ij} = max(0, 1 - \frac{|x_i - x_j|}{\frac{l_i + l_j}{2} + max(CL_i, CL_j)}) \tag{4.14}$$

$$\Delta Y_{ij} = max(0, 1 - \frac{|y_i - y_j|}{\frac{w_i + w_j}{2} + max(CL_i, CL_j)}) \tag{4.15}$$



(a) Overlap on the x-axis is allowed (b) Overlap on the y-axis is allowed (c) Overlap on both axes, the constraint is violated

Figure 4.5: Example of component overlap

This constraint calculates the overlap matrix A, in which the overlap between all components for a layout is saved. During the particle swarm optimization this constrained is enforced using a penalty function.

Enforcing the overlap constraints with a penalty function

There are several ways to handle the constraints of a particle swarm optimization. One of the most used techniques is the penalty function (Helwig, 2010). Yarpiz (2015) provides an implementation for this.

When using a penalty function the objective functions are modified to include the penalty function term so that the problem can then be solved as an unconstrained minimization problem. Two broad approaches for penalty functions can be distinguished, interior and exterior penalty functions. Interior penalty functions apply their penalty when the boundary of the feasible region is approached, so that this boundary into the infeasible region can never be crossed. This is illustrated in 4.6a. Exterior penalty functions apply their penalty once the boundary into the infeasible region has been crossed, as is illustrated in 4.6b.

The interior penalty has one very large disadvantage for the formulated problem, namely that once the boundary to the infeasible region has been crossed it has no mechanism to get back into the feasible region. As the population is randomly generated at the start of the particle swarm algorithm, there is no guarantee that the initial population does not violate the constraints and thus is within the infeasible region. It is even very likely that the initial patricles are infeasible in a problem where the total area of all facilities is somewhat close to the placement area. Considering this, external penalty functions are used in this research.

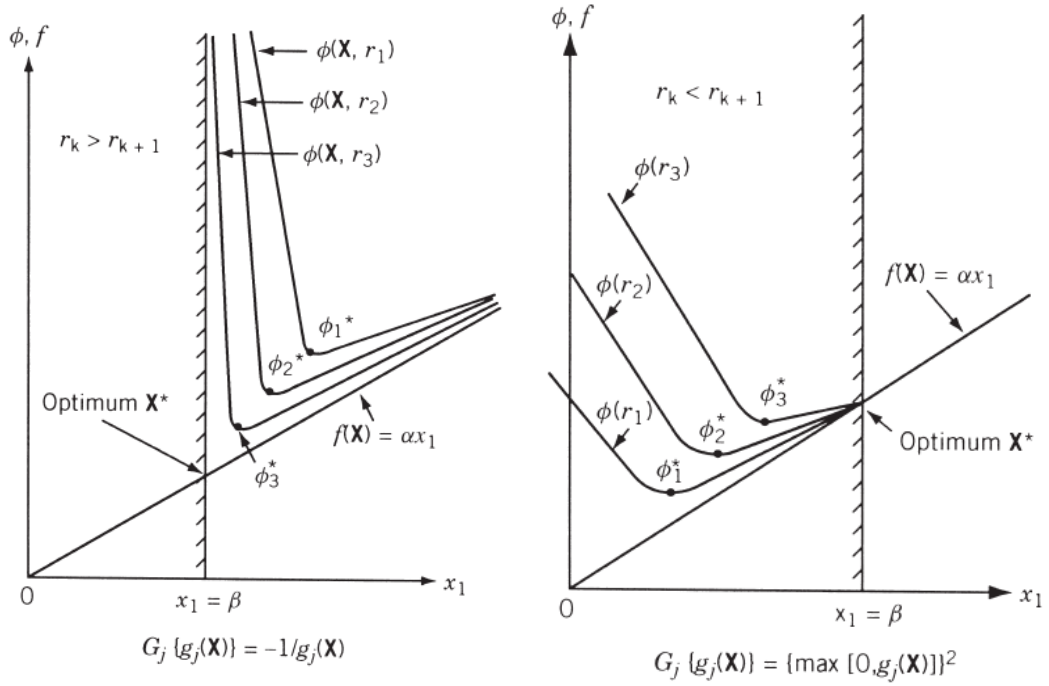
A general formulatio of an exterior penalty function is (Rao, 2009):

$$f_{new}(x) = f(x) + r_k \sum_{i=1}^{m} (g_j(x))^q \tag{4.16}$$

Here r_k is a positive penalty parameter, the exponent q in a nonnegative constant, and $g_j(X)$ is defined as:

$$g_j(x) = max(g_j(x), 0) \tag{4.17}$$

So that $g_j = 0$ when the constraint is satisfied, and $g_j > 0$ when the constraint is violated.



(a) Interior penalty function

(b) Exterior penalty function

Figure 4.6: Illustration of the interior and exterior penalty function Rao (2009)

By choosing the value of the parameters r_k and q the behaviour of the penalty function, and thus of the optimization, can be tuned. Tuning the parameter properly is important; if they are chosen too low, then exploring infeasible space in not sufficient penalized and particles will spend a lot of iterations exploring this infeasible space. If the parameters are chosen too high then crossing the boundary of the feasible region into the infeasible region will incur such a heavy penalty that particles will be unable to to explore disconnected feasible regions (Helwig, 2010).

For this thesis both objective functions are formulated to be in the same order of magnitude so that the same penalty function tuning can be used for both objective functions.

Enforcing the area limits

The other constraint is that the components can not be placed outside of the limits of the defined room. This is a so called box-constraint, for which the PSO has a standard implementation using a repair function method Helwig (2010). It is called a box constraint because the variables in the decision vector that determine the x- and y-positions of each component must fall between the values determined by the box, or the limits of the room. These upper and lower bounds for the x- and y-values of each component are the limits of the room by default, but can be changed by the designer if they already have in mind where a certain component must be placed. A component can be constrained to a certain area of the room in this way, or even to a single feasible position. This is illustrated in figure 4.7.

4.1.5. The model and optimization problem defined

With the facility layout problem model and the optimization problem defined this model could then be built and coded. Instead of starting from scratch, an implementation of the facility layout problem solved with particle swarm optimization was found (Yarpiz, 2015). This implementation is provided by Yarpiz, which is aimed to be a resource of academic and professional scientific source codes and tutorials. Source codes provided by Yarpiz, are all free to use for research and academic purposes, and free to share and modify, as well. This implementation of the facility layout problem already has several properties of the specific instance of the problem which best models the engine room arrangement

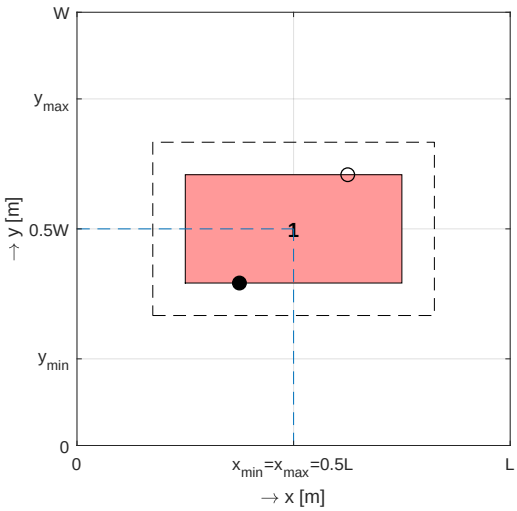


Figure 4.7: A component that is free on the y-axis, but has a set x-coordinate of 0.5L. The minimum and maximum y-coordinates not 0 or W, as they denote the location of the centre of the component and are thus limited by the width and clearance of the component.

problem modelled and coded, which were referenced in the previous sections. As such, it serves as a good starting point when implementing and coding the defined model.

4.2. Particle Swarm Optimization algorithm

In section 3.2 the particle swarm optimization algorithm was chosen as the resolution approach for the facility layout problem. This section describes the basic workings of this algorithm and some of its more advanced properties. The information in this section is attributed to Heris (2016).

The particle swarm optimization algorithm was inspired by the behaviour of groups of animals, such as a school of fish or a flock of birds. A population of particles is randomly initialized. As the algorithm iterates, all the particles move trough the solution space and try to find the best solutions. This is illustrated in figure 4.8, where a PSO with three decision variables finds the optimal solution (represented by the star) at iteration N.

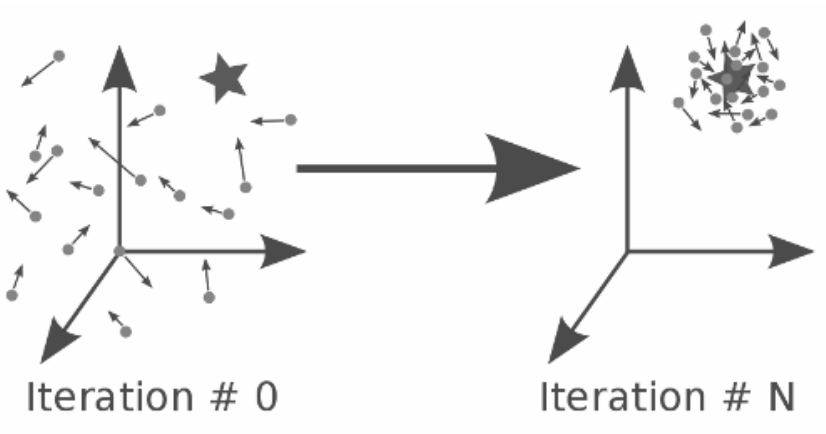
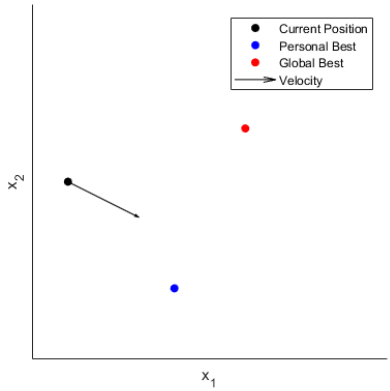


Figure 4.8: Illustration of the Particle Swarm Optimization Process (Pagmo Development Team, 2017).

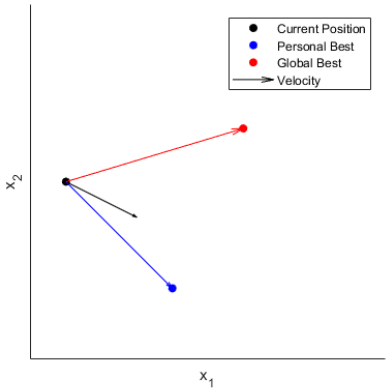
An iteration step for a single particle is illustrated in figure 4.9. Each particle has several properties, shown in figure 4.9a: the decision variable vector defines the position pos, the particle has a current

velocity vel. Furthermore, each particle remembers the personal best solution it encountered during the optimization $pBest$, and knows the global best solution out of all solutions encountered by all particles during the optimization, $gBest$.

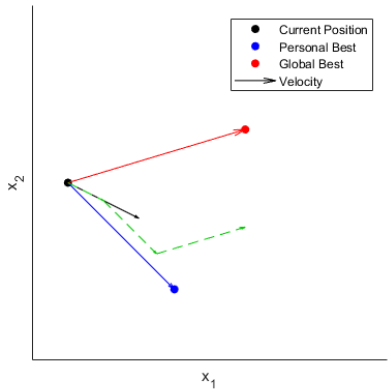
The vectors between the current position and the personal and global best are calculated (figure 4.9b), and by scaling these two vectors and the current velocity with the predefined PSO parameters and them adding them together the new velocity is found (figure 4.9c). Finally the new position in the solution space is calculated by adding this new velocity to the current position (figure 4.9d).



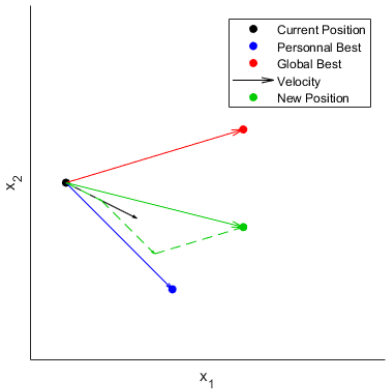
(a) Known information at current iteration it



(b) Calculate the vectors to the particles personal best and the swarms global best.



(c) Scale the vectors based on the PSO parameters and add them together.



(d) The new position in the solution space at $it + 1$ is found.

Figure 4.9: Visual representation the particle swarm optimization algorithm with two decision variables x_1 and x_2 and a population of one particle. Adapted from Heris (2016).

Mathematically this is governed by two equations: equation 4.18 calculates the new velocity of particle i at iteration $t + 1$ and equation 4.19 updates the position of particle i at iteration $t + 1$:

$$vel_i(t+1) = w * vel_i(t) + c_1 * r_1 * (pBest_i(t) - pos_i(t)) + c_2 * r_2 * (gBest_i(t) - pos_i(t)) \quad (4.18)$$

$$pos_i(t+1) = pos_i(t) + vel_i(t+1) \quad (4.19)$$

Equation 4.18 consists of three parts: the inertia component, the cognitive component and the social component. Each of these components has a tuning parameter, respectively w, c_1 and c_2. Choosing these parameters is the main way to influence the behaviour of the particle swarm algorithm.

The first term $w * vel_i(t)$ is called the inertia term, in which the parameter w is called the inertia coefficient. This term accounts for the velocity that the particle has when calculating the new position. The parameter w typically has a value between 0.4 and 0.9, in which a lower value encourages exploitation of the area close to the particle in the solution space, while a higher value encourages the exploration of new areas in the solution space.

The second term $c_1 * r_1 * (pBest_i(t) - pos_i(t))$ is called the cognitive components, as it lets the particle learn from its own past by steering it towards the best solution it encountered itself.

The final term, $c_2 * r_2 * (gBest_i(t) - pos_i(t))$ is called the social component as it governs how the swarm of particles 'communicates' with each other, by steering each particle towards the best solution that the whole swarm has encountered.

The parameters c_1 and c_2 in the cognitive and social components of equation 4.18 are called the acceleration coefficients and typically have a value between 0 and 2. The numbers r_1 and r_2 have a random value between 0 and 1, and are newly determined at each step of the iteration.

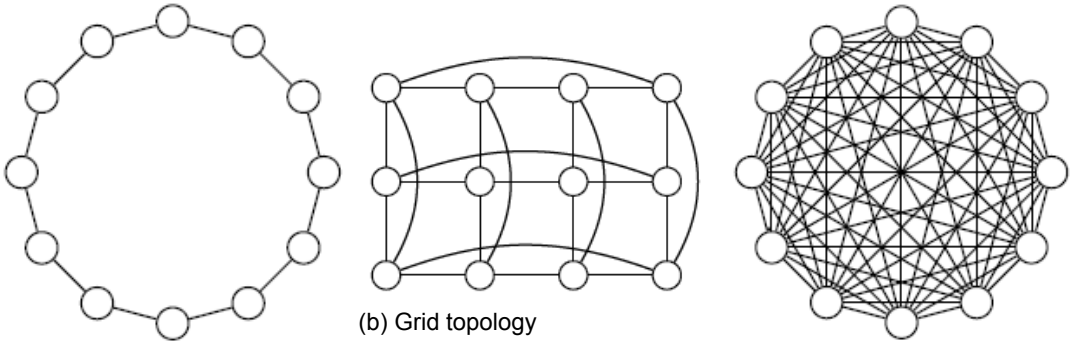
Helwig (2010) references for values of c_1 and c_2 of $c_1 = c_2 = \frac{(w+1)^2}{2}$

This basic version of the algorithm is not very complex, being governed by only the two equations 4.18 and 4.19 which have only three tunable parameters, w, c_1 and c_2. Aside from these the number of particles in the swarm and the maximum number of iterations need to be defined.

More complex versions of the PSO algorithm introduce different neighbourhood topologies, constraints, multiple objectives, (possibly problem specific) heuristics and mutations, and other adaptations.

4.2.1. Neighbourhood topologies

In equation 4.18 three components are used to calculate the new velocity of a particle: the current velocity, the personal best of the particle and the global best of the swarm. This global best is how the particles in the swarm share information with each other. For this information sharing the topology of the swarm is important, i.e. how the particles are connected. The three most common options for these topologies are shown in figure 4.10.



(a) Ring topology (b) Grid topology (c) Fully connected topology

Figure 4.10: Different topologies for the particles in the swarm, ordered from least to most information shared. (Helwig, 2010)

In most PSO algorithms a fully connected swarm is used which corresponds to the global best shown before. This type of topology has the fastest convergence as the best solutions are instantly available to guide every particle. When one of the other topologies is used one speaks not of a global best, but of a local best as information is only shared locally with a particles own neighbours. In a ring topology each particle only has two neighbours, while in a grid topology each particle is connected to four other particles. The more connections a topology has, the faster information spreads throughout the swarm and the faster it converges. However, a slower spread of information allows particles to explore the

solution space more before converging upon the best found region. In practice, the ring topoloy is considered too slow, but the grid topology is recommended as an alternative to the fully connected topology Helwig (2010).

4.2.2. Upper and lower bounds

Upper and lower bounds allow us to impose an upper and a lower limit on the values of variables in the design vector. These limits bound the solution space that a particle can move in during the optimization. In the model these are used to enforce that each facility lies within the preallocated area, or to limit the placement of a facility within the area. A facility can for example be free to move within the whole area, be set to have an x-coordinate of larger than $\frac{L}{2}$ and so is placed forward in the engine room, or even be locked into a x- or y position when both the upper and lower bound on an axis are the same.

is used to bound each facilities x-coordinate, y-coordinate and rotation options.

These bounds are implemented with a repair function implementing the following rules:

- If a design variable is higher than its maximum value, set it to its maximum value and reverse the direction of that design variable in the velocity vector
- If a design variable is lower than its minimum value, set it to its minimum value and reverse the direction of that design variable in the velocity vector

These rules change not only the position of a particle which has left the search space defined by the boundaries, but also inverses the velocity component of the axis on which it left the search space to pull the particle back into the feasible region. This is a common repair algorithm for particle swarm boundary handling (Helwig, 2010).

4.2.3. Velocity clamping

In order to limit the step size of each particle at each iteration velocity clamping can be used. When applied, the direction of the calculated velocity vector remains the same, but the magnitude is limited to a set maximum. In this case the maximum is defined as a percentage of the size of the search space. While applying velocity clamping does is not necessary for the swarm to converge, its application can improve the performance of the algorithm Helwig (2010). The velocity clamping is implemented using a similar rule to the above boundary handling rules:

- If the velocity of a particle is larger than a set limit, set the magnitude of the velocity of the particle to the limit.

4.2.4. Multiple objectives

In the algorithm every iteration a global best solution must be chosen, or if a different neighbourhood topology is chosen a local best must be selected from a group. But with multiple objectives there is not a single metric to choose the best solution from anymore.

Instead of a single global best that is updated after each iteration, a repository with nondominated solutions is kept which is updated every iteration. At each iteration all nondominated particles of that iteration are checked against this repository of best found solutions. From this new group the nondominated solutions are kept as the new repository, and the dominated solutions are thrown out.

When at the next iteration a new global best must be selected, this is done from this repository of best found answers.

In order to maintain solution diversity in the selected global best, instead of selecting a solution at random from the repository, a hypercube grid method is used to group similar solutions together. One

of these groups is then selected at random, and from this group a random global best for the iteration is chosen.

A grid is made on the two objective axes, linearly spaced between the maximum and minimum objective values in the repository. For each solution it is calculated in which grid cell it is located. Then a random grid cell containing solutions is selected, and from this grid cell a random solution is selected as the global best.

The amount of cells is the same in both directions, and is set as a parameter at the start of the simulation. Martínez-Cagigal (2017) suggests a grid of 20x20.

The amount of solutions relative to the grid resolution determines how 'full' the cells are likely to be, and how 'similar' the solutions are before they are in a cell together.

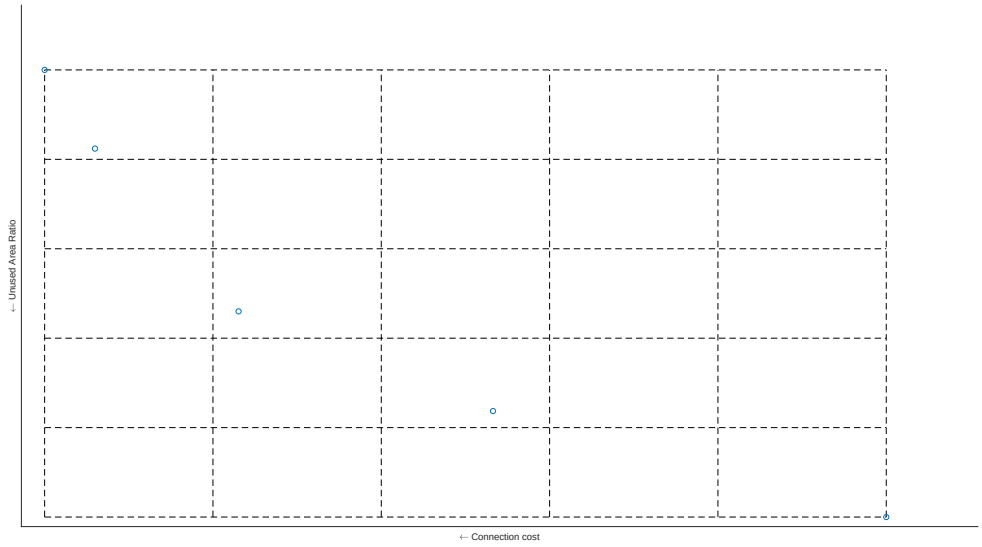


Figure 4.11: Illustration of a 5x5 grid on a Pareto front of five solutions. Four cells have a solution in them; the two leftmost solutions are placed in the same cell. For the next iterations one of these cells will be chosen randomly, after which a random solution from this cell is chosen as the global best for that iteration.

An existing implementation of the Multiple Objective Particle Swarm Optimization algorithm by Martínez-Cagigal (2017) was used for this research.

5

Model Testing and Adaptations

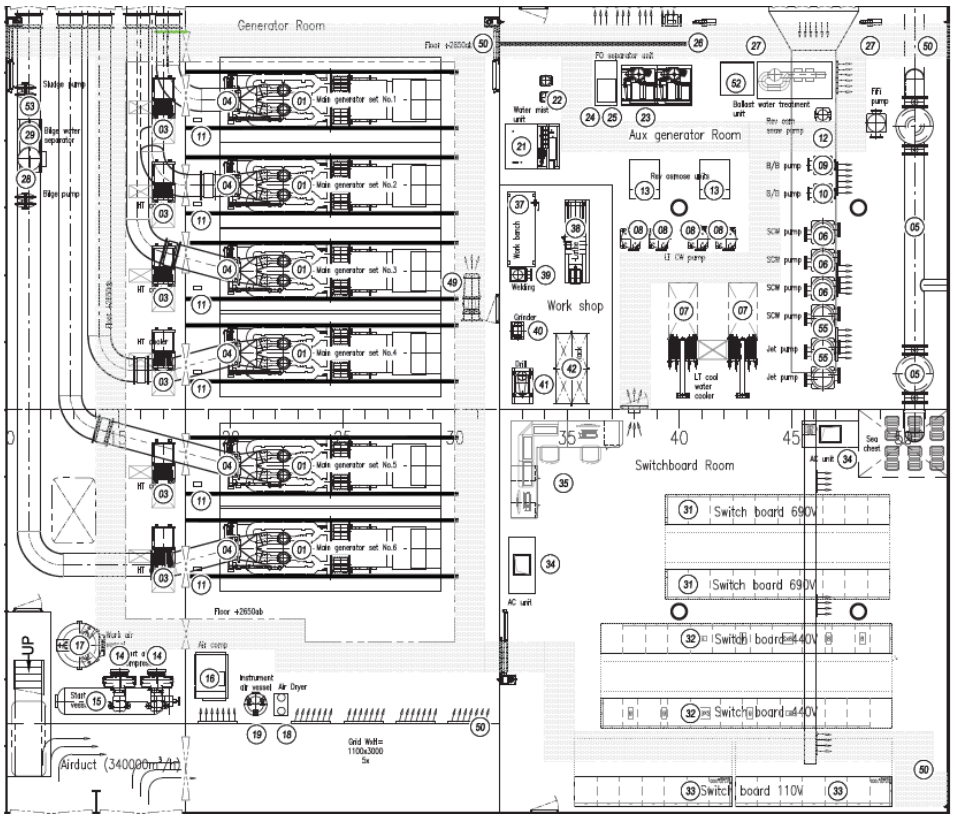
Now that the model has been defined in chapter 4 it is important to test the capabilities of the design tool for machinery space arrangement. To do this a Vuyk design of a jackup vessel has been selected to serve as input to test the capabilities of the tool and identify its shortcomings. First the input for this defined, after which the tool is run and its results are analysed. Various additions to the algorithm called 'mutations' are developed to increase the performance of the tool. A new grouping constraint is developed and implemented to better allow a designer to better implement necessary component relations. Finally a pathing filter is developed and added to the tool to better account for the design drivers of machinery space design.

5.1. Setup

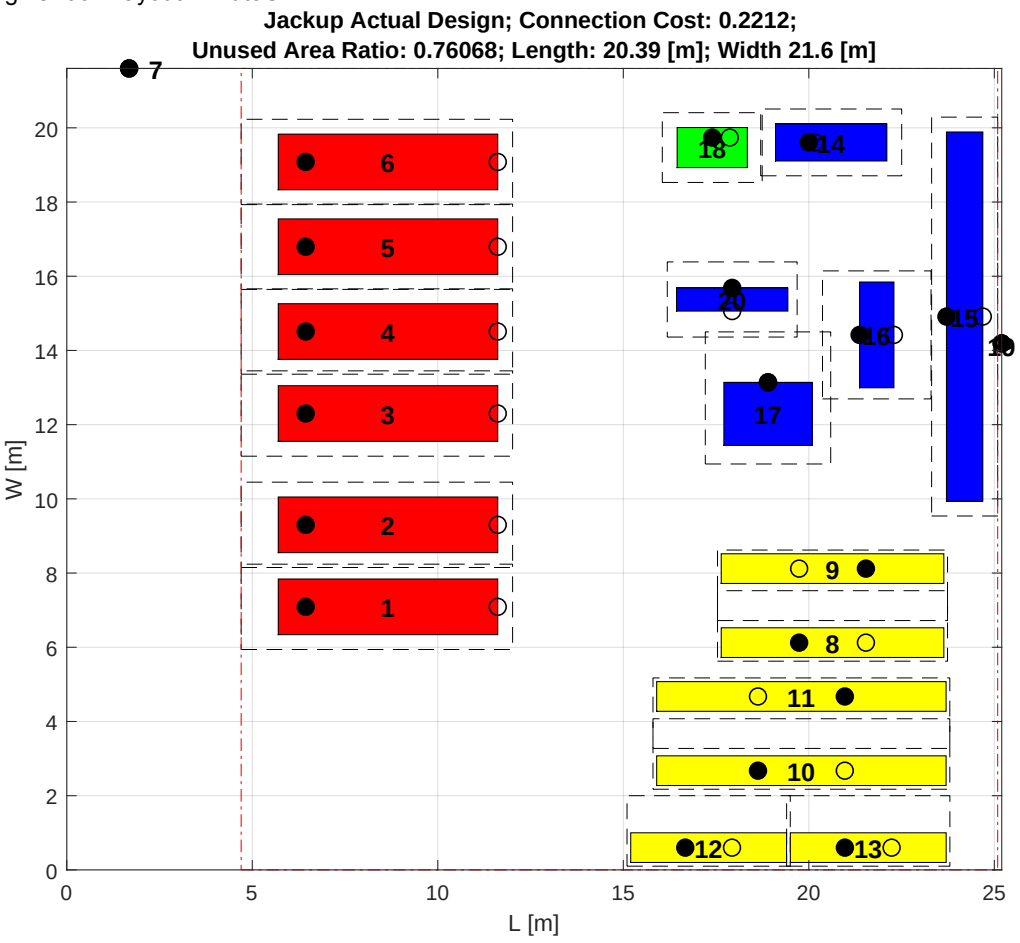
In order to test the tool a Vuyk design was selected. This design is a concept design of a jackup vessel. As it is a concept design it properly represents the level of detail Vuyk desires at the end of the concept design phase. The design also has several other properties which make it a suitable for testing the model: it has a single-deck engine room, a large amount of components which leads to a large possible solution space of potential layout solutions, and it is diesel-electric, which imposes less limits on the positional constraints of the engines as these do not need to be connected to a propeller or gearbox shaft.

5.1.1. Defining Input

The actual design of the vessel can be seen in figure 5.1a. The accompanying figure 5.1b shows the same design recreated using the model, with its accompanying scores on the two objective functions. Designs generated by the model will be compared to this actual design in order to get a sense for what realistic solutions are.



(a) Engine room layout in AutoCAD



(b) The design recreated for the model

Figure 5.1: Actual design of the xf12152 Jackup vessel

Components

The most relevant components need to be chosen in order to sufficiently model the design to result in figure 5.1b. Based on interviews with various Vuyk engineers these components that generally have the most impact upon layout design were established. From these the components that are modelled were selected. The details of these components are shown in table 5.1.

In addition to their properties, all the components defined in this table are also given a number (first column) and a colour (the last column) to make them clearly identifiable when they are plotted. This can be seen in the plot in figure 5.1b.

This design is a concept design that was never built, and as such most components present in the arrangement are mostly reference components taken from previous designs. This is a common practice as was discussed in section 2.1.2. All the measurements of these components have been taken from the general arrangement drawings of the jackup design, while the estimates for the weights were provided by Vuyk.

Classification			General Physical properties			Clearances				In and output location				Other
			Length	Width	Weight	Left	Right	Up	Down	Xin	Yin	Xout	Yout	Plot
Nr.	Component	Type	[mm]	[mm]	[kg]	[mm]	[mm]	[mm]	[mm]	[0..1]	[0..1]	[0..1]	[0..1]	Color
1	Caterpillar 3516B	Genset	5916	1500	13500	1000	400	400	400	1	0.5	0.13	0.5	red
2	Caterpillar 3516B	Genset	5916	1500	13500	1000	400	400	400	1	0.5	0.13	0.5	red
3	Caterpillar 3516B	Genset	5916	1500	13500	1000	400	400	400	1	0.5	0.13	0.5	red
4	Caterpillar 3516B	Genset	5916	1500	13500	1000	400	400	400	1	0.5	0.13	0.5	red
5	Caterpillar 3516B	Genset	5916	1500	13500	1000	400	400	400	1	0.5	0.13	0.5	red
6	Caterpillar 3516B	Genset	5916	1500	13500	1000	400	400	400	1	0.5	0.13	0.5	red
7	Ventillation exit	Interface	4800	1	0	0	0	0	0	0.35	0.5	0.35	0.5	black
8	Switch board 690V	Switch board	6000	800	7000	100	100	1000	100	0.65	0.5	0.35	0.5	yellow
9	Switch board 690V	Switch board	6000	800	7000	100	100	1000	100	0.65	0.5	0.35	0.5	yellow
10	Switch board 440V	Switch board	7800	800	10000	100	100	1000	100	0.65	0.5	0.35	0.5	yellow
11	Switch board 440V	Switch board	7800	800	10000	100	100	1000	100	0.65	0.5	0.35	0.5	yellow
12	Switch board 110V	Switch board	4200	800	4000	100	100	1000	100	0.65	0.5	0.35	0.5	yellow
13	Switch board 110V	Switch board	4200	800	4000	100	100	1000	100	0.65	0.5	0.35	0.5	yellow
	Ballast Water	Ballast Water												
14	Treatment Unit	Treatment Unit	3000	1000	2000	400	400	400	400	0.35	0.5	0.3	0.5	blue
15	Crossover	Crossover	9950	980	2251.3	400	400	400	400	0.5	0	0.5	1	blue
16	SCW Pump Skid	Pump	925	2850	2000	1000	1000	300	300	1	0.5	0	0.5	blue
17	LT Heat Exchanger	Heat exchanger	2380	1700	3600	500	500	1360	500	0.5	1	0.5	1	blue
18	Fuel Seperator unit	Separator	1892	1080	1100	400	400	400	400	0.75	0.75	0.5	0.75	green
19	Crossover exit	Interface	1	324	0	0	0	0	0	0.5	0.7	0.5	0.7	black
20	LT CW pump skid	Pump	625	3000	2000	700	700	250	250	0	0.5	1	0.5	blue

Table 5.1: Properties of the modelled components of the Jackup design

The only components which are specified by the drawings are the six generator sets, which are Catepillar 3516B generator sets. These are the red components numbered 1 to 6.

Components 8 to 13 are the six switchboards and are plotted in yellow. They form a substantial part of a diesel-electric design such as this one. They mainly need clearance to the front of them in order to perform maintenance but can be tightly packed in the other directions.

The next four components in the table are part of the cooling system. The crossover (15) takes in seawater which is pumped by the seawater cooling pumps (16) to the low temperature heat exchanger (17). The low temperature cooling pumps (20) provide the other cooling water flow that goes into the heat exchanger. These pumps are modelled as one pump-skid component instead of separately, as often applied in practical engine room design. The advantage for the tool is that defining too many separate components would make the problem too high-dimensional which increases the difficulty for the algorithm to find feasible solutions. The low-temperature cooling systems cools the high-temperature cooling system which is connected to the engines. The high temperature heat exchangers are placed directly behind the engines and are not modelled separately in order to prevent the mentioned component bloat. Instead the clearance behind the engines is extended to account for these.

Aside from these power generation, electrical and cooling system components, two other components are added to increase the diversity and account for other important engine room systems. These are a ballast water treatment unit (14) and a fuel separator unit (18) containing two separators for the Marine Diesel Oil that treat the fuel for the generator sets (1-6).

Aside from the regular components two interfaces with the vessel are defined: an exhaust ventilation exit for the generator sets to connect to, and an crossover exit interface through which the crossover supplies an adjacent room. The exhaust ventilation exit is defined together with the naval architect overseeing the general design, as discussed in section 2.1.2, and is vital in guiding the placement of the generator sets. The crossover exit interface is defined to encourage the crossover to be placed close to the adjacent space.

Position constraints

In table 5.2 the position constraints of the components are defined. These are left relatively open initially. The interfaces (7,19) are constrained to their position along their respective walls, with a rotation constraint to match the direction of the wall.

Classification			Position constraints					
Nr.	Component	Type	Xmin [0..1]	Xmax [0..1]	Ymin [0..1]	Ymax [0..1]	Rmin [0..1]	Rmax [0..1]
1	Caterpillar 3516B	Genset	0	1	0	1	0	1
2	Caterpillar 3516B	Genset	0	1	0	1	0	1
3	Caterpillar 3516B	Genset	0	1	0	1	0	1
4	Caterpillar 3516B	Genset	0	1	0	1	0	1
5	Caterpillar 3516B	Genset	0	1	0	1	0	1
6	Caterpillar 3516B	Genset	0	1	0	1	0	1
7	Ventillation exit	Interface	0	0.2	1	1	0.1	0.1
8	Switch board 690V	Switch board	0	1	0	1	0	1
9	Switch board 690V	Switch board	0	1	0	1	0	1
10	Switch board 440V	Switch board	0	1	0	1	0	1
11	Switch board 440V	Switch board	0	1	0	1	0	1
12	Switch board 110V	Switch board	0	1	0	1	0	1
13	Switch board 110V	Switch board	0	1	0	1	0	1
	Ballast Water	Ballast Water						
14	Treatment Unit	Treatment Unit	0	1	0	1	0	1
15	Crossover	Crossover	0.9	1	0.95	1	0.4	0.4
16	SCW Pump Skid	Pump	0	1	0	1	0	1
17	LT Heat Exchanger	Heat exchanger	0	1	0	1	0	1
18	Fuel Seperator unit	Separator	0	1	0	1	0	1
19	Crossover exit	Interface	1	1	0	1	0.1	0.1
20	LT CW pump skid	Pump	0	1	0	1	0	1

Table 5.2: Position constraints of the modelled components of the Jackup design

Relationship Matrix

The relationship matrix that is discussed in section 4.1.1 is presented in table 5.3. The values have been determined according to table 4.1. A few relations are deemed *absolutely necessary* and so have a value of 5: The generator sets outputs are connected with the exhaust exit input, the crossover is connected to the crossover exit, the seawater cooling pumps are connected to the low temperature heat exchanger which is connected to the low temperature cooling pumps. Furthermore the switchboards are connected according to the single line diagram: there are two sets of three switchboards, were a 690V switchboard is connected to a 440V switchboard which is then connected to a 110V switchboard.

The switchboards of the same voltage are also connected to each other, but by a less severe connection of the value 3: *important*. After some initial test all the connections between switchboards were set to a minimum value of 3. The switchboards in this design are separated from the other components because they need to be placed in a climatised space, the Switchboard Room. To encourage this in the relationship matrix their connection value between each other is set to a minimum of 3 while the

value to the other connections is set to 0. Only the 690V switchboards are each connected to three generator sets according to the provided single line diagram. They are connected to the input of the generator sets; while the electrical power is obviously an output of the generator, the location of the outputted power is the alternator which is where the input gate of the generator set is placed. The output gate of the generator sets is then used for the exhaust ventilation of the diesel engines.

The low temperature cooling pumps are connected to the engine output by a *especially important* connection, as the high temperature heat exchangers for the engines are also located there but are not modelled separately. The fuel separators are connected by the same relationship value to the engines.

The two interface-type components are only connected to the components they interface with, and as mentioned the switchboards are only connected to each other or their engine for the 690V switchboards. The rest of the relationships between the components have either an *ordinairy* or an *unimportant* relationship.

Relationship Matrix: the values in this matrix denominate the relavite importance that the in- and outputs of two components are placed near eachother	Comp 1: Caterpillar 3516B	Comp 2: Caterpillar 3516B	Comp 3: Caterpillar 3516B	Comp 4: Caterpillar 3516B	Comp 5: Caterpillar 3516B	Comp 6: Caterpillar 3516B	Comp 7: Ventillation exit	Comp 8: Switch board 690V	Comp 9: Switch board 690V	Comp 10: Switch board 440V	Comp 11: Switch board 440V	Comp 12: Switch board 110V	Comp 13: Switch board 110V	Comp 14: Ballast Water Treat	Comp 15: Crossover	Comp 16: SCW Pump Skid	Comp 17: LT Heat Exchanger	Comp 18: Fuel Seperator unit	Comp 19: Crossover exit	Comp 20: LT CW pump skid
From this component's ouput																				
Comp 1: Caterpillar 3516B	0	2	2	2	2	2	5	0	0	0	0	0	0	1	1	1	1	2	0	2
Comp 2: Caterpillar 3516B	2	0	2	2	2	2	5	0	0	0	0	0	0	1	1	1	1	2	0	2
Comp 3: Caterpillar 3516B	2	2	0	2	2	2	5	0	0	0	0	0	0	1	1	1	1	2	0	2
Comp 4: Caterpillar 3516B	2	2	2	0	2	2	5	0	0	0	0	0	0	1	1	1	1	2	0	2
Comp 5: Caterpillar 3516B	2	2	2	2	0	2	5	0	0	0	0	0	0	1	1	1	1	2	0	2
Comp 6: Caterpillar 3516B	2	2	2	2	2	0	5	0	0	0	0	0	0	1	1	1	1	2	0	2
Comp 7: Ventillation exit	2	2	2	2	2	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Comp 8: Switch board 690V	4	4	4	1	1	1	0	0	3	5	3	3	3	0	0	0	0	0	0	0
Comp 9: Switch board 690V	1	1	1	4	4	4	0	3	0	3	5	3	3	0	0	0	0	0	0	0
Comp 10: Switch board 440V	0	0	0	0	0	0	0	3	3	0	3	3	3	0	0	0	0	0	0	0
Comp 11: Switch board 440V	0	0	0	0	0	0	0	3	3	3	0	3	3	0	0	0	0	0	0	0
Comp 12: Switch board 110V	0	0	0	0	0	0	0	3	3	3	3	0	3	0	0	0	0	0	0	0
Comp 13: Switch board 110V	0	0	0	0	0	0	0	3	3	3	3	3	0	0	0	0	0	0	0	0
Comp 14: Ballast Water Treat	1	1	1	1	1	1	0	0	0	0	0	0	0	0	2	3	2	2	2	2
Comp 15: Crossover	1	1	1	1	1	1	0	0	0	0	0	0	0	2	0	5	2	2	2	2
Comp 16: SCW Pump Skid	1	1	1	1	1	1	0	0	0	0	0	0	0	3	5	0	2	2	2	2
Comp 17: LT Heat Exchanger	1	1	1	1	1	1	0	0	0	0	0	0	0	2	2	2	0	2	0	5
Comp 18: Fuel Seperator unit	4	4	4	4	4	4	0	0	0	0	0	0	0	2	2	2	2	0	0	2
Comp 19: Crossover exit	0	0	0	0	0	0	0	0	0	0	0	0	0	0	5	0	0	0	0	2
Comp 20: LT CW pump skid	2	2	2	2	2	2	0	0	0	0	0	0	0	2	2	2	2	2	2	0

To this component's input

Table 5.3: Relations of the modelled components of the Jackup design

5.1.2. Initial runs

Several initial runs were performed to observe the behaviour of the model. All runs are performed with a particle swarm size of 60 particles. The amount of particles in a swarm is generally not found to have a large influence on the performance of a particle swarm optimization (Helwig, 2010). This was confirmed by initial testing of the model. However, the swarm size should be chosen with the dimensionality and characteristics of the problem in mind. As the problem as defined is high-dimensional (there are 20 defined components, or facilities, for a total of 60 dimensions), a large swarm size is chosen.

Figure 5.2 shows an example of the results from one of these early runs, presenting for all feasible arrangement solutions the reference values for the Connection cost and for the Unused Area Ratio in a graph. These are the two main objectives that are discussed in 4.1.3. The reference value for the Unused Area Ratio is indicating the resulting length of the machinery space arrangement. This particular run was performed with the settings of $w = 0.6$ and $c_1 = c_2 = \frac{(w+1)^2}{2} = 1.28$ for a total of 100 iterations. There are 2649 feasible solutions found in this run. The actual solution shown in figure 5.1b is represented by the star symbol in the lower left hand of the figure.

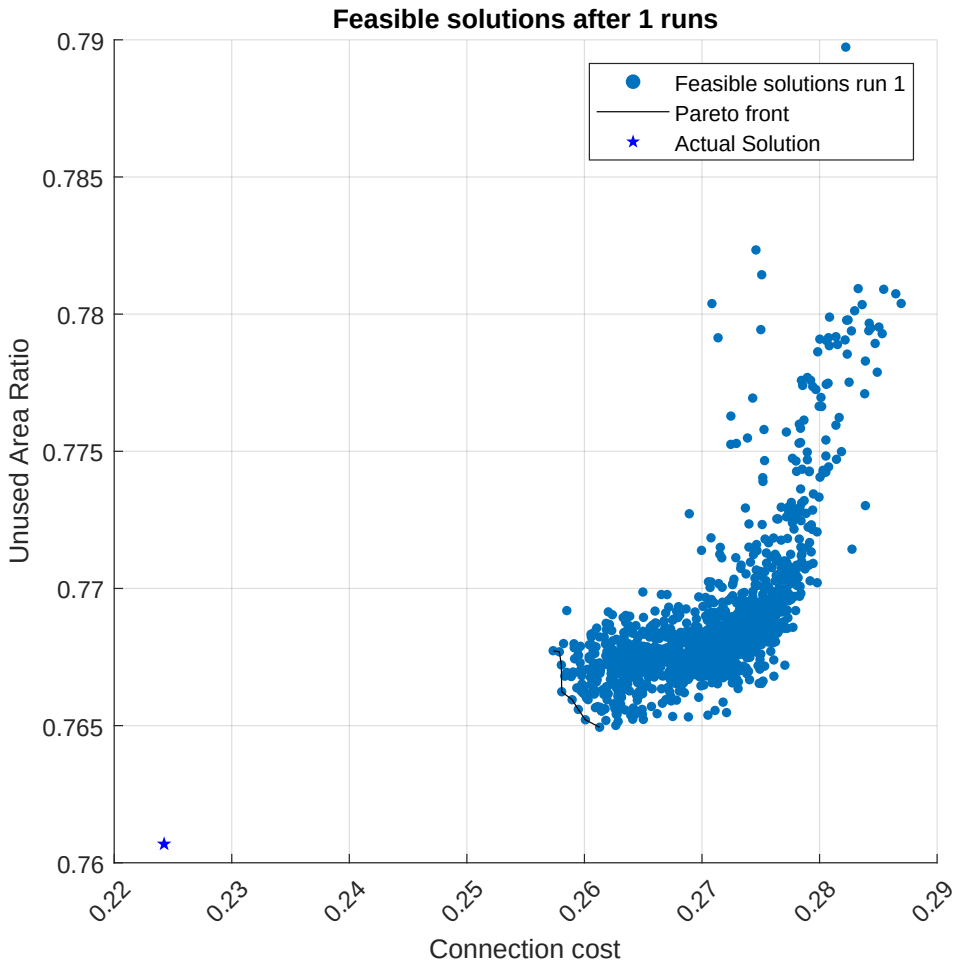


Figure 5.2: An example of the results found in the initial runs.

Figure 5.3 shows the convergence of the objectives of the example run. In iteration 38 of the algorithm the first feasible solution is found which is seen by the significant drop of both objectives. This is the first solution without a penalty. After this small gains are made in the further iterations.

The Pareto front in figure 5.2 is represented by the line along the lower-left corner of the solution-dots. One of the layouts corresponding to the Pareto front is plotted in figure 5.4. The other layouts of the

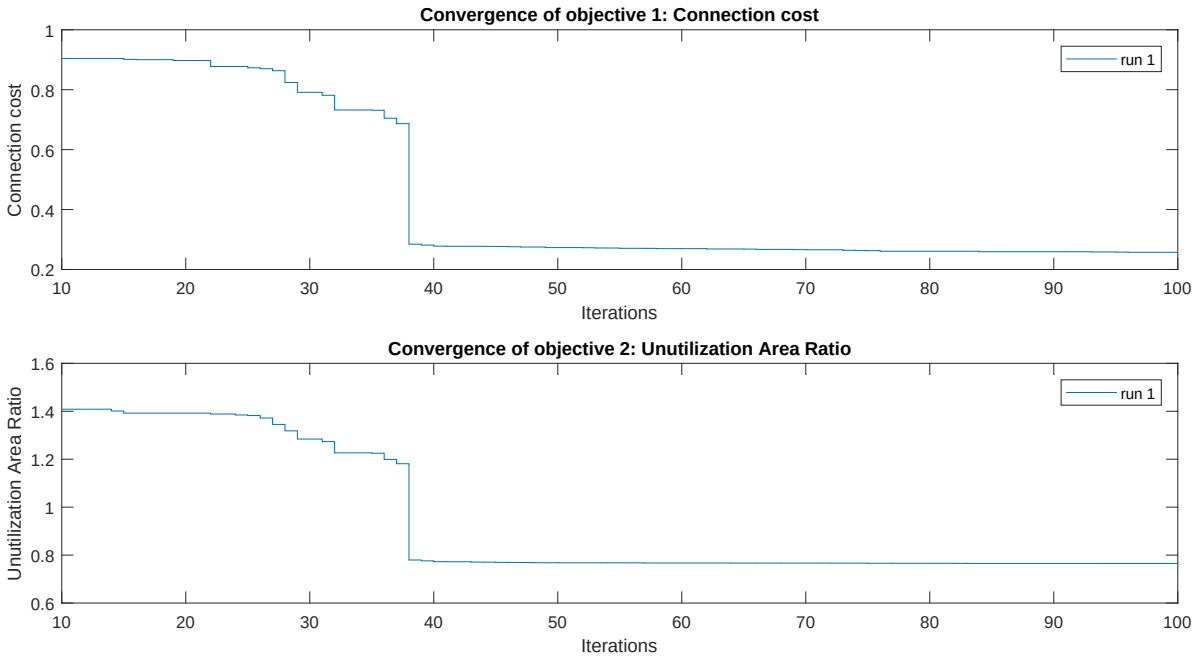


Figure 5.3: The convergence of the objectives of the example run

Pareto front can be found in figures A.1 and A.2 in appendix A. There is no meaningful difference found in the layouts of the Pareto front, as can be seen in these figures.

The initial solutions are that are found by the tool are plotted in figure A.3 in appendix A. While the components are a bit more spread out and some have another rotation, the final solutions of the Pareto front can clearly be seen in these initial layouts. There is no relevant variation in the arrangements

After several exploratory runs of the model it became clear that the behaviour that is shown in the above example is the same for all performed runs: the algorithm is unable to escape the first local minimum it finds: after the random initialization there is almost never a feasible solution, which means that every particle has a penalty. As the particles start to explore the solution space one eventually encounters a feasible region, which then incurs no penalty and is thus significantly better. The algorithm explores this particular feasible region to find the best solution. However, the particle is then unable to escape this feasible region and thus does not find any diverse, or meaningfully different, solutions.

In order to solve this problem, the first step was to retune the penalty function which enforces the overlap constraint. This penalty function was discussed in section 4.1.4. When the penalty function is tuned too high the algorithm gets stuck in these local minima's, but gradually lowering the penalty function values of r_k and q does not result in finding more diverse solutions. Instead, a run of the model has a higher likelihood of not finding any feasible solutions at all, while if it does find solutions it still has the same local minima problems. For the tool the penalty function was tuned to be able to distinguish between feasible and infeasible solutions, and the local minima problems were tackled in the following sections.

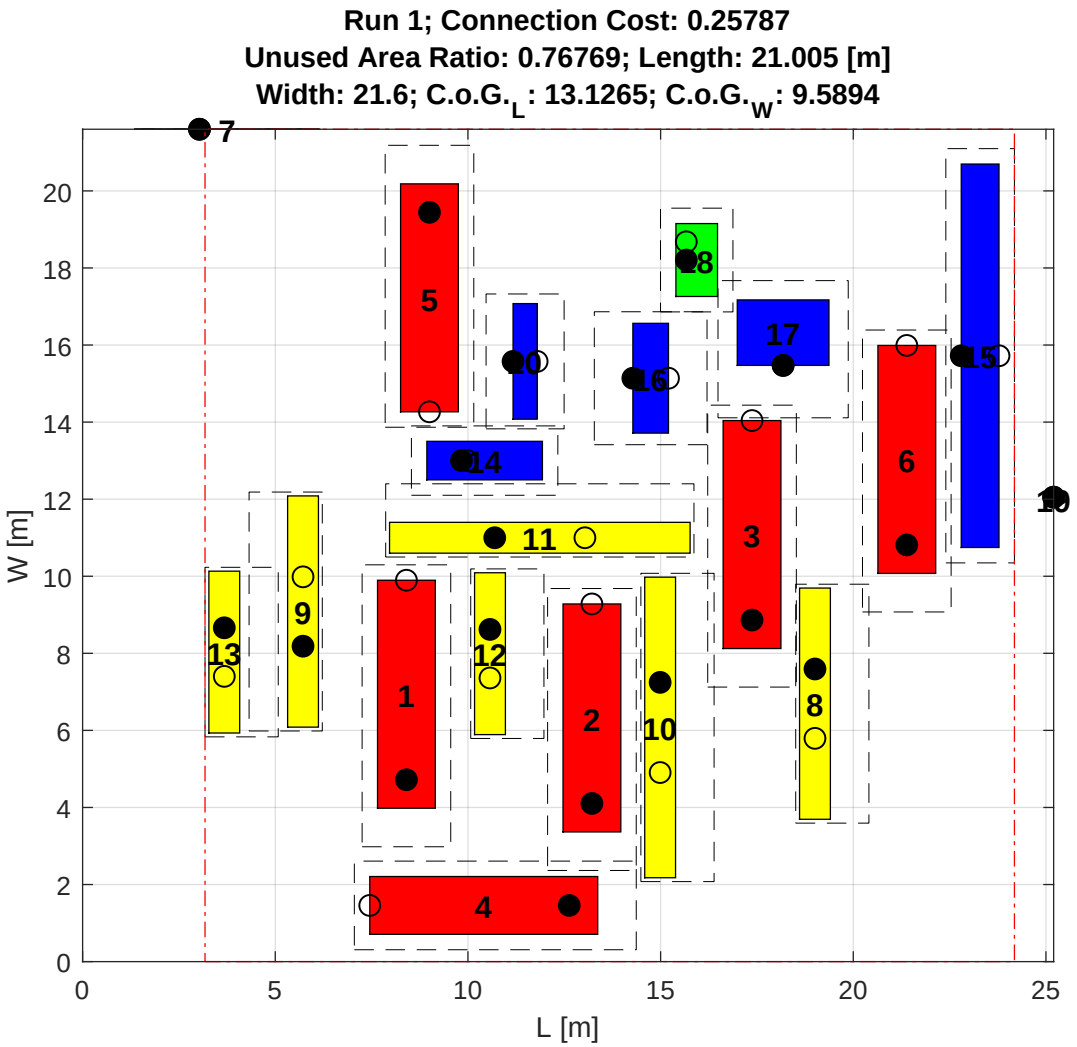


Figure 5.4: Example Pareto front layout from the Pareto front of figure 5.2. The other Pareto front layouts of this run can be found in appendix A

5.1.3. Diversifying Solutions

In order to find diverse solutions, the tool can run the model multiple times instead of once for each setting of the MOPSO parameters. As each run finds a different local minimum which it gets stuck in, each finds a different solution. This way various meaningfully different solutions are found. Also, because of the random nature of the algorithm, the effect of changing the MOPSO parameters is more clear when looking at various runs of the model with the same settings. For each run one random member of its Pareto front is plotted, together with its Pareto scatter plot and the convergence plot of the objective functions.

Figures 5.6, 5.5 and 5.7 show the results of this approach for 4 runs of the model with the same settings as the run performed earlier in this section. The settings and the amount of feasible results found in each run can be found in table 5.4. The scatter plot of feasible solutions found by the runs shown in figure 5.6 shows that the result of a run depends highly upon the initial found feasible region that the algorithm then explores.

Figure 5.7 shows a layout from an arrangement solution that is part of the Pareto front (lower left corner in the graph) of each of the four runs. While each of these “Pareto layouts” within a run are very similar(as each of these runs explored a different local minimum of the solution space), the four layouts presented are very different.

w	$c_1(w)$	$c_2(w)$	Number of feasible results			
			Run 1	Run 2	Run 3	Run 4
0.60	1.28	1.28	1345	1828	2191	1534

Table 5.4: Results of the example run 4 times with the same settings

While the behaviour of the algorithm is unfortunate, it is still interesting to investigate what the right MOPSO parameters are to find solutions. It can be investigated whether they can be tuned to allow the algorithm to escape the local minima, or if they have an noticeable effect on the exploration of these local minima. Also, there are alterations that can be made to the algorithm which need to be investigated, intending to help the algorithms escape these local minima.

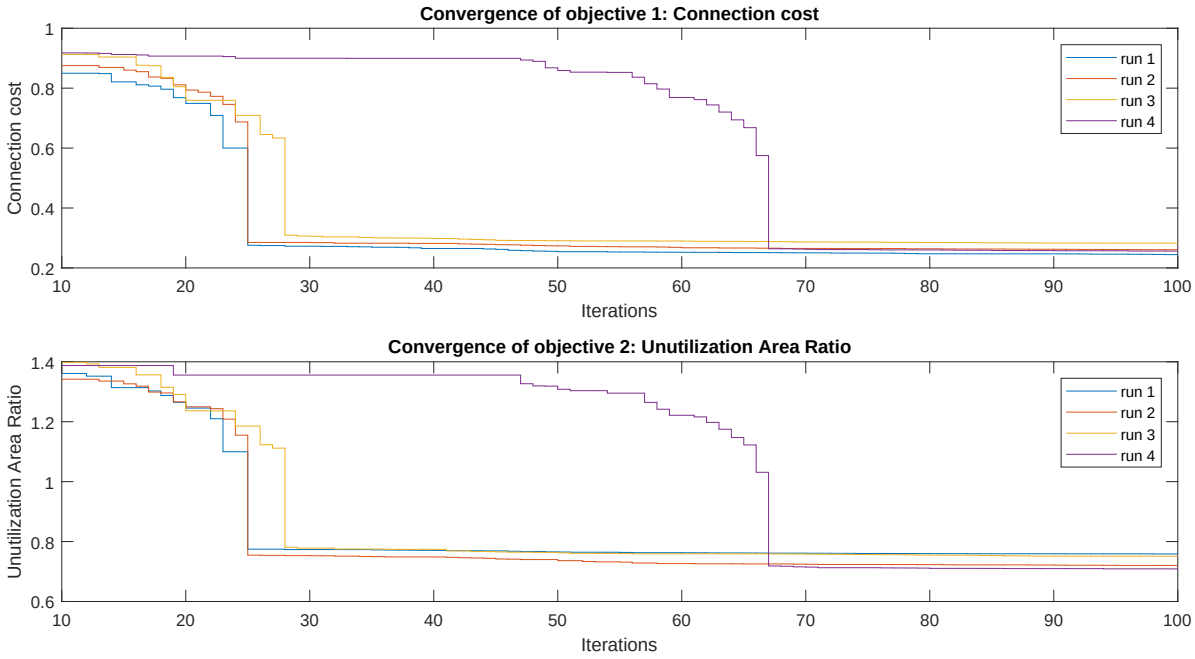


Figure 5.5: The convergence of the objectives

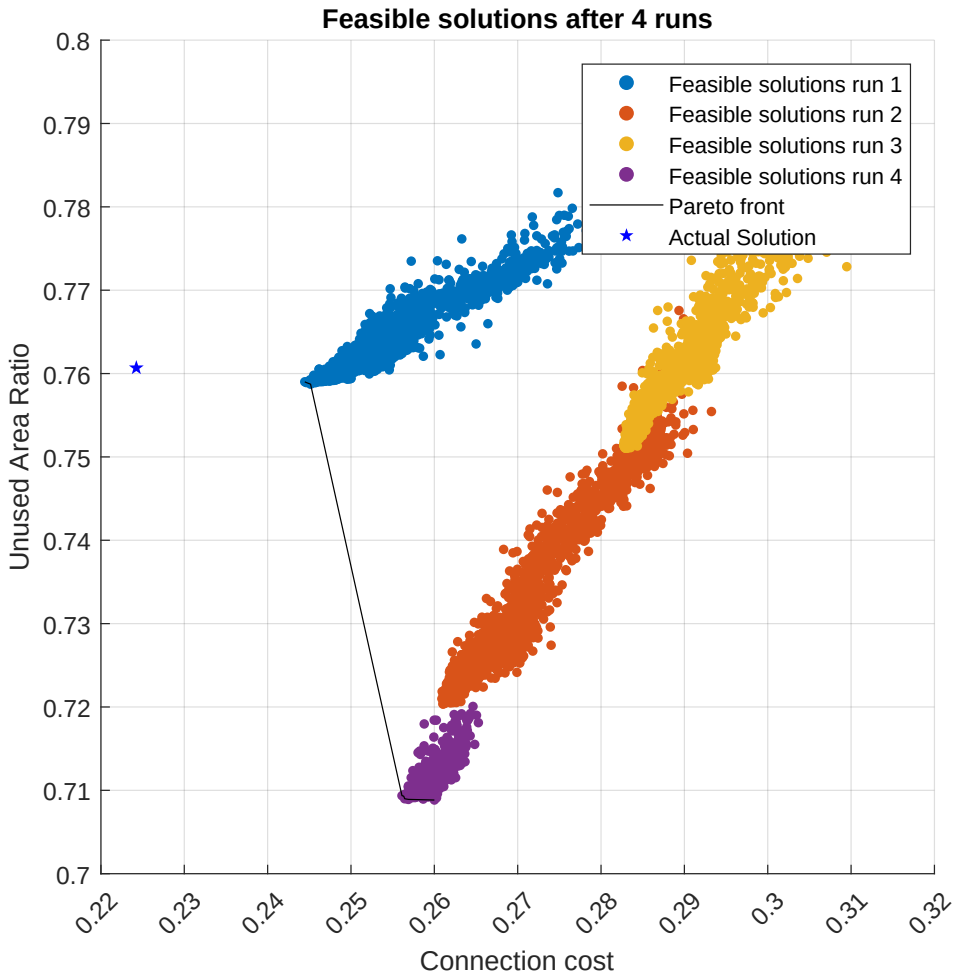
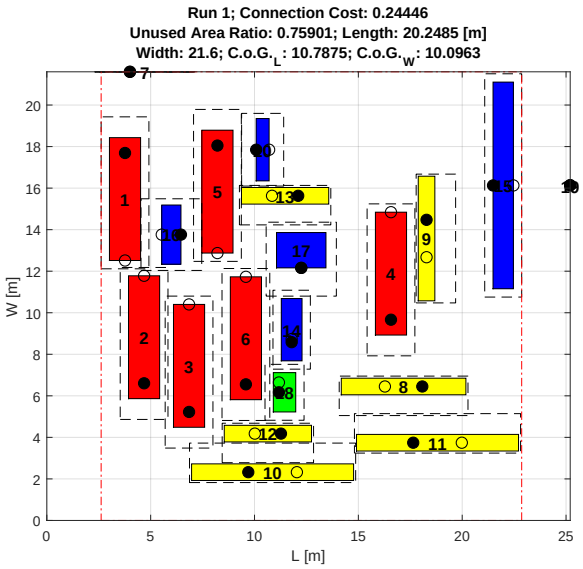
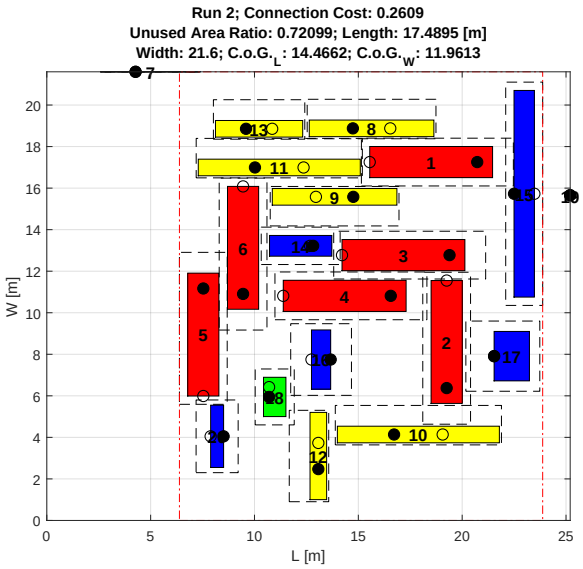


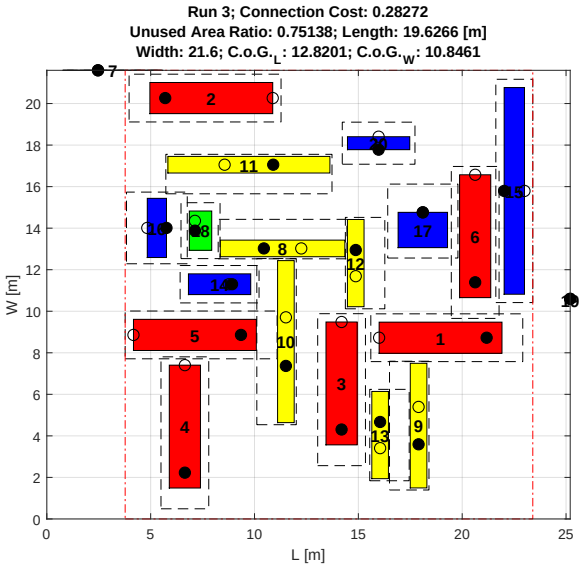
Figure 5.6: Four runs of the model performed with identical settings, shown in table 5.4.



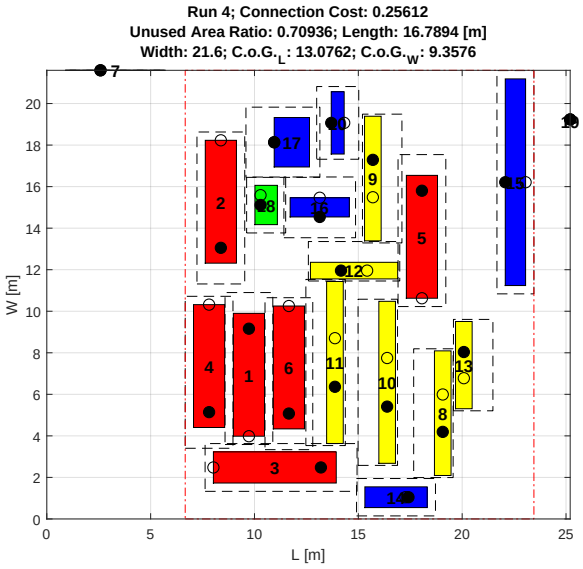
(a) Pareto front layout of the first run



(b) Pareto front layout of the second run



(c) Pareto front layout of the third run



(d) Pareto front layout of the fourth run

Figure 5.7: A randomly selected Pareto front layout for each of the four runs seen in figure 5.6. As each of these runs explored a different local minima of the solution space, the four shown solutions are meaningfully different.

5.2. Investigating the Effect of the PSO Parameters

First, in order to get a broad overview of the settings, the model is run over the acceptable range of the inertia coefficient w with the cognitive coefficient c_1 and its social coefficient c_2 dependable upon the value of w by the formula $c_1 = c_2 = \frac{(w+1)^2}{2}$ (Helwig, 2010).

w	$c_1(w)$	$c_2(w)$	Number of feasible results			
			Run 1	Run 2	Run 3	Run 4
0.40	0.98	0.98	0	0	0	0
0.45	1.05	1.05	4352	0	3988	0
0.50	1.13	1.13	0	2102	3400	4522
0.55	1.20	1.20	938	2593	2696	2000
0.60	1.28	1.28	1345	1828	2191	1534
0.65	1.36	1.36	97	529	378	370
0.70	1.45	1.45	0	83	0	46
0.75	1.53	1.53	25	0	0	0
0.80	1.62	1.62	0	0	0	0
0.85	1.71	1.71	0	0	0	0
0.90	1.81	1.81	0	0	0	0

Table 5.5: Number of feasible solutions found while varying w with $c_1(w) = c_2(w)$ for 100 iterations

The corresponding Pareto scatter plots can be found in figures A.4 and A.5 in appendix A.

Section 4.2 explained that low values of w encourage exploitation of the area close to the particle, while a high value encourages the exploration of new areas in the solution space. This corresponds to what can be seen in the Pareto scatter plots of this run: the low values of w that find results have a very narrow scatter plot for each run. The higher w gets the more spread out the scatters of the individual runs get. However, upon investigation of the solutions, they are still essentially the same solution which indicates that the same local minimum has not been escaped. When the value of w gets above 0.65 the model has trouble finding feasible solutions at all.

Because a higher w should be promising the model was run again but with significantly more iterations for the w values of 0.70-0.90, to see if the results are different when the algorithm is given more iterations to find a feasible region. The results of these runs can be found in table 5.6. The Pareto scatter of the solutions can be found in figure A.6 of appendix A. While the higher number of iterations allow some higher settings to find feasible solutions, the problems with diversity persist: in effect only a single meaningful solution is found.

w	$c_1(w)$	$c_2(w)$	Number of feasible results			
			Run 1	Run 2	Run 3	Run 4
0.70	1.45	1.45	249	133	191	72
0.75	1.53	1.53	31	99	0	33
0.80	1.62	1.62	0	24	0	17
0.85	1.71	1.71	0	0	0	0
0.90	1.81	1.81	0	0	0	0

Table 5.6: Number of feasible solutions found while varying w with $c_1(w) = c_2(w)$ for 500 iterations

In order to investigate if the effects of the three parameters w, c_1 and c_2 could be identified, a series of runs was performed where two of the parameters were set while the third was varied over the range of the parameter identified in section 4.2. For w this range is between 0.4 and 0.9, while for c_1 and c_2 this range is between 0 and 2. An overview of these runs is shown in tables 5.7, 5.8 and 5.9. The

accompanying Pareto scatter plots, convergence plots and layout plots can be found in appendix A.

w	c_1	c_2	Number of feasible results			
			Run 1	Run 2	Run 3	Run 4
0.40	1.28	1.28	2925	3501	3735	0
0.45	1.28	1.28	3397	2384	3375	2831
0.50	1.28	1.28	2069	1891	3149	2953
0.55	1.28	1.28	2513	1960	2406	1752
0.60	1.28	1.28	1345	1828	2191	1534
0.65	1.28	1.28	571	1104	1131	574
0.70	1.28	1.28	134	223	626	283
0.75	1.28	1.28	79	69	211	99
0.80	1.28	1.28	0	12	47	54
0.85	1.28	1.28	0	0	0	1
0.90	1.28	1.28	0	0	0	0

Table 5.7: Number of feasible solutions found while varying w and keeping c_1 andc_2 constant

w	c_1	$c_2(w)$	Number of feasible results			
			Run 1	Run 2	Run 3	Run 4
0.60	0.00	1.28	1323	1557	3739	2068
0.60	0.20	1.28	2311	1506	2272	1804
0.60	0.40	1.28	1968	1749	2574	2360
0.60	0.60	1.28	1949	1746	1917	2207
0.60	0.80	1.28	0	1993	2445	2065
0.60	1.00	1.28	1166	1928	1855	0
0.60	1.20	1.28	1010	2424	2393	1557
0.60	1.40	1.28	1164	1358	1134	1784
0.60	1.60	1.28	402	648	969	305
0.60	1.80	1.28	361	835	171	0
0.60	2.00	1.28	305	41	267	0

Table 5.8: Number of feasible solutions found while varying c_1 and keeping w and $c_2(w)$ constant

w	$c_1(w)$	c_2	Number of feasible results			
			Run 1	Run 2	Run 3	Run 4
0.60	1.36	0.00	0	0	0	0
0.60	1.36	0.20	0	0	0	0
0.60	1.36	0.40	0	0	0	0
0.60	1.36	0.60	0	0	0	3255
0.60	1.36	0.80	2982	2575	2494	1800
0.60	1.36	1.00	2032	1978	2597	0
0.60	1.36	1.20	1605	1258	2499	0
0.60	1.36	1.40	426	600	768	267
0.60	1.36	1.60	0	165	258	0
0.60	1.36	1.80	0	226	24	0
0.60	1.36	2.00	0	83	22	19

Table 5.9: Number of feasible solutions found while varying c_2 and keeping w and $c_1(w)$ constant

When parameter c_2, the social coefficient, is too low no solutions can be found. Without sufficient

communication in the swarm there is no drive for each particle to explore outwards of the infeasible region in which they are initialized and so do not encounter feasible regions at all.

When the cognitive coefficient c_1 is varied all cases produce feasible results, but higher values seem to indeed encourage local exploitation for a more broad scatter plot. When set too high the number of feasible solutions again diminishes.

Higher values of each parameter lead to a more broad scatter, where the particles take larger steps each iteration and so find more diverse solutions. However, these also increase the chance that a step through the solution space is too big and the resulting solution is infeasible. This leads to less solutions being found as the values of the parameters increase until they are not found at all.

While the Pareto scatter plots of some runs clearly cover a larger amount of the solution space than others, not one set of parameter values achieves to cover a substantial amount of the solution space. Closer investigation of the resulting solutions in each of these runs, shows that the same behaviour that was present in the initial runs is found for all performed runs: after all the particles are initialized in an infeasible region, they start to explore the solution space as the algorithm starts iterating. Given the correct settings a first feasible solution, that is a solution with no component overlap, is found. The algorithm then starts to explore this feasible region and optimizing this particular configuration. However, it can only do this by moving the components slightly. Components can never move across each other to create a meaningfully different solution, as this requires the particles to move through an infeasible region where components have overlap, which it cannot accomplish. This is a problem that can not be solved by tuning the MOPSO parameters.

Relaxing the penalty function that enforces the overlap constraint should allow the particles to move through infeasible regions after they get stuck in a feasible one, but doing this gives the algorithm too little ability to distinguish between feasible and non-feasible regions which causes no feasible solutions to be found at all.

Whether the resulting solutions manage to approach or surpass the initial solution designed by Vuyk is entirely dependent on the placement of the components relative to each other in the encountered local minima. The algorithm can mainly optimize the first encountered feasible solutions by placing the components closer together, which has a large effect on the unused area ratio objective. This generally only has a small effect on the connection cost, the distances are shortened but the components are not ordered more logically to reduce connection cost in a more meaningful way. This explains why most runs that find feasible solutions have a clear downward trend on the Unused Area Ratio, but cannot reduce the Connection Cost value, which would be necessary to approach the objective value of the initial Vuyk design.

In the next section several mutations are added to the MOPSO algorithm in an effort to allow the algorithm to escape these local minima and thus create diverse solutions.

5.3. Mutations

In order to explore the solution space more broadly by enabling the algorithm to escape the local minima it gets stuck in, several mutations are implemented.

The term ’mutation’ is used when working with genetic algorithms (discussed in section 3.2), which are among the most-used meta-heuristics and as such the term mutations is also used when discussing other meta-heuristics, such as the particle swarm optimization. The term still applies well to this case as well, as it describes additions to the MOPSO algorithm that mutate selected particles in the swarm to change their position in the solution space.

Four mutations are implemented in the model and are discussed below. Each of these can be switched on or off independently. First the effect of each mutation is investigated separately, after which promising combinations can be identified and consequently investigated.

The initial investigation is done by performing a sweep of w from 0.4 to 0.9 with $c_1 = c_2 = \frac{(w+1)^2}{2}$ (Helwig, 2010) in order to quickly check the whole spectrum of possible parameters.

5.3.1. Mutation: Replace

The implemented MOPSO code already had an uniform mutation option build-in, which replaces a number of particles, depending on the mutation factor u_{mut}, with a randomised new particle. After this replacement a position boundary check is performed to enforce compliance to the position constraints.

By creating new randomly generated particles the algorithm can escape local minima in the solution space that it cloud get stuck in. While such a particle is very unlikely to be feasible, by travelling through the solution space from this new location it can discover previously encountered feasible regions and so explore interesting regions of the solution space that the algorithm does not otherwise encounter.

w	$c_1(w)$	$c_2(w)$	Number of feasible results			
			Run 1	Run 2	Run 3	Run 4
0.40	0.98	0.98	2763	882	0	3175
0.45	1.05	1.05	2436	0	1947	0
0.50	1.13	1.13	0	594	2274	2002
0.55	1.20	1.20	1940	1316	2001	1424
0.60	1.28	1.28	893	0	1057	1093
0.65	1.36	1.36	175	0	569	314
0.70	1.45	1.45	34	17	65	50
0.75	1.53	1.53	13	0	53	0
0.80	1.62	1.62	0	0	0	0
0.85	1.71	1.71	0	0	0	0
0.90	1.81	1.81	0	0	0	0

Table 5.10: Number of feasible solutions found with mutation: Replace, a mutation percentage of 20% while varying w and $c_1(w) = c_2(w)$

The scatter plots belonging to the runs shown in table 5.10 are figures A.13 and A.14 in appendix A.

The behaviour of the swarms is very similar to the normal observed behaviour. In the initial run with a mutation percentage of 20% there seems to have been no to little effect. Therefore the run was repeated with a higher muation percentage of 40%, leading to comparable results. These results can be seen in table 5.11 and in figures A.15 and A.16 in appendix A. Even higher mutation percentages were investigated but did not lead to more promising results.

The mutation does not have the intended effect. A newly, randomly generated particle has only a very slim chance to be feasible, and when travelling through the solution space is very unlikey to encouter a new feasible region before that particle arrives back in the local minimum that the swarm has found.

As the mutation does not have the intended effect it is not implemented in the final model.

w	$c_1(w)$	$c_2(w)$	Number of feasible results			
			Run 1	Run 2	Run 3	Run 4
0.40	0.98	0.98	1753	0	2763	0
0.45	1.05	1.05	0	0	2483	1928
0.50	1.13	1.13	0	2189	0	2257
0.55	1.20	1.20	1150	2001	1904	0
0.60	1.28	1.28	1032	1590	1426	486
0.65	1.36	1.36	0	367	0	300
0.70	1.45	1.45	0	33	75	15
0.75	1.53	1.53	0	11	0	11
0.80	1.62	1.62	0	0	0	0
0.85	1.71	1.71	0	0	0	0
0.90	1.81	1.81	0	0	0	0

Table 5.11: Number of feasible solutions found with mutation: Replace, a mutation percentage of 40% while varying w and $c_1(w) = c_2(w)$

5.3.2. Mutation: Swap

The investigated second mutation is the swap mutation. At each iteration, for each particle this mutation swaps the place of two randomly selected facilities. The first of these is randomly selected from the list of facilities, after which the second facility is selected from those facilities not identical to the first one. After the two facilities have been swapped, the position constraints are enforced anew and the new solution is evaluated. If it dominates the old solution it replaces it, otherwise the old solution is kept.

By forcing a swap with this mutation the swapped facilities do not need to move through the infeasible region where they overlap other facilities in order to get to a significantly different position. This should enable the algorithm to find more diverse solutions.

w	$c_1(w)$	$c_2(w)$	Number of feasible results			
			Run 1	Run 2	Run 3	Run 4
0.40	0.98	0.98	0	0	0	0
0.45	1.05	1.05	39	3	0	0
0.50	1.13	1.13	0	53	20	0
0.55	1.20	1.20	0	0	0	0
0.60	1.28	1.28	0	0	5	0
0.65	1.36	1.36	0	0	0	0
0.70	1.45	1.45	0	0	0	0
0.75	1.53	1.53	0	0	0	0
0.80	1.62	1.62	0	0	0	0
0.85	1.71	1.71	0	0	0	0
0.90	1.81	1.81	0	0	0	0

Table 5.12: Number of feasible solutions found with mutation: Swap while varying w and $c_1(w) = c_2(w)$

Table 5.12 shows the amount of feasible solutions for the first runs with this mutation enabled. Because relatively little results were found, the run was performed again with more iterations, but this did not lead to better results.

Unfortunately this mutation does not have the intended effect. The model has a lot more difficulty finding feasible results, and those found are not more diverse than without the mutation. Due to the relatively large amount of components packed in the machinery area, and these components having a large variety in size, switching two components almost always lead to a new solution that has significant

overlap and is thus is dominated by the old solution.

As the mutation does not have the intended effect it is not implemented in the final model.

5.3.3. Mutation: Improve Nondominated Particles

The Heris (2016) PSO implemtation included a local search heuristic which has been adapted to a mutation in the MOPSO to improve the quality of the solutions that the algorithm finds. In the original PSO each iteration the global best value was improved, but this is not possible in a MOPSO. It is found that the best results are obtained when every iteration the a random number of the nondominated particles of the swarm are improved. This is done in the following steps:

1. Select a nondominated particle
2. Sort all facilities in a random order.
3. Select the first facility
4. Move the selected facility in each of nine defined movement possibilities. These movements cannot exceed the position constraints and the length of the movement is randomly chosen between 0 and half the maximum movement length of L or W. The movement possibilities are:
   - 4.1 no movement
   - 4.2-4.5 Move in each of the four main directions (up, down, left, right)
   - 4.6-4.9 Move in each of the four diagonal directions (up-right, up-left, down-right, down-left)
5. After each movement is performed the facility is rotated to all orientations which its constraints allow.
6. For each rotation the solution is evaluated.
7. The solutions from these nine movement-rotation combinations are checked for domination against each other.
8. One nondominated solution is selected to continue from.
9. Select next facility in the order and go back to step 4.
10. End when all facilities have been moved

This mutation performs a lot changes and evaluations on the particle. Especially the large amount of evaluations add to the runtime of the program. But, as the mutation is only performed on a single or small amount of particles each iteration the actual impact on the total runtime is still within reasonable limits. Implementing this mutation doubles the runtime of the program to about 120 seconds.

The results of the initial investigation into this mutation can be found in table 5.13 and figures A.19 and A.20 in appendix A. One example scatter plot is shown in figure 5.8.

While it won't help with improving the swarm as a whole to get all the particles to explore more diverse regions of the solution space, it leans into the optimization part of exploring the solution space by improving a single or small amount of particles each iteration. Due to this optimizing the algorithm is finally able to escape the local minima that it otherwise gets stuck in. The mutation enables the movement of facilities in big enough steps to actually pass each other and so manages to create diverse solutions. This also enables the optimization of the connection cost objective better, as now the distance between important inputs and outputs of facilities can be better minimized by placing the relevant components near each other.

It is noticeable that now tens of feasible solutions are found, instead of the hundreds or thousands feasible solutions that sometime were found before this mutation was implemented. Also the solutions are not as clumped together any more, instead these tens of solutions are generally more spread out. This is clearly illustrated in figure 5.8. When a feasible solution is found, it is very likely that this particle dominates the swarm and is thus selected to be mutated, which results in a significant change in its position in the solution space. Before when a feasible solution was found all the particles would

w	$c_1(w)$	$c_2(w)$	Number of feasible results			
			Run 1	Run 2	Run 3	Run 4
0.40	0.98	0.98	89	43	201	586
0.45	1.05	1.05	64	53	43	64
0.50	1.13	1.13	66	38	55	303
0.55	1.20	1.20	28	38	45	137
0.60	1.28	1.28	32	60	59	44
0.65	1.36	1.36	27	54	22	46
0.70	1.45	1.45	24	40	35	53
0.75	1.53	1.53	45	38	29	38
0.80	1.62	1.62	23	27	38	29
0.85	1.71	1.71	46	41	34	32
0.90	1.81	1.81	27	22	41	33

Table 5.13: Number of feasible solutions found with mutation: Improve Nondominated Particles while varying w and $c_1(w) = c_2(w)$

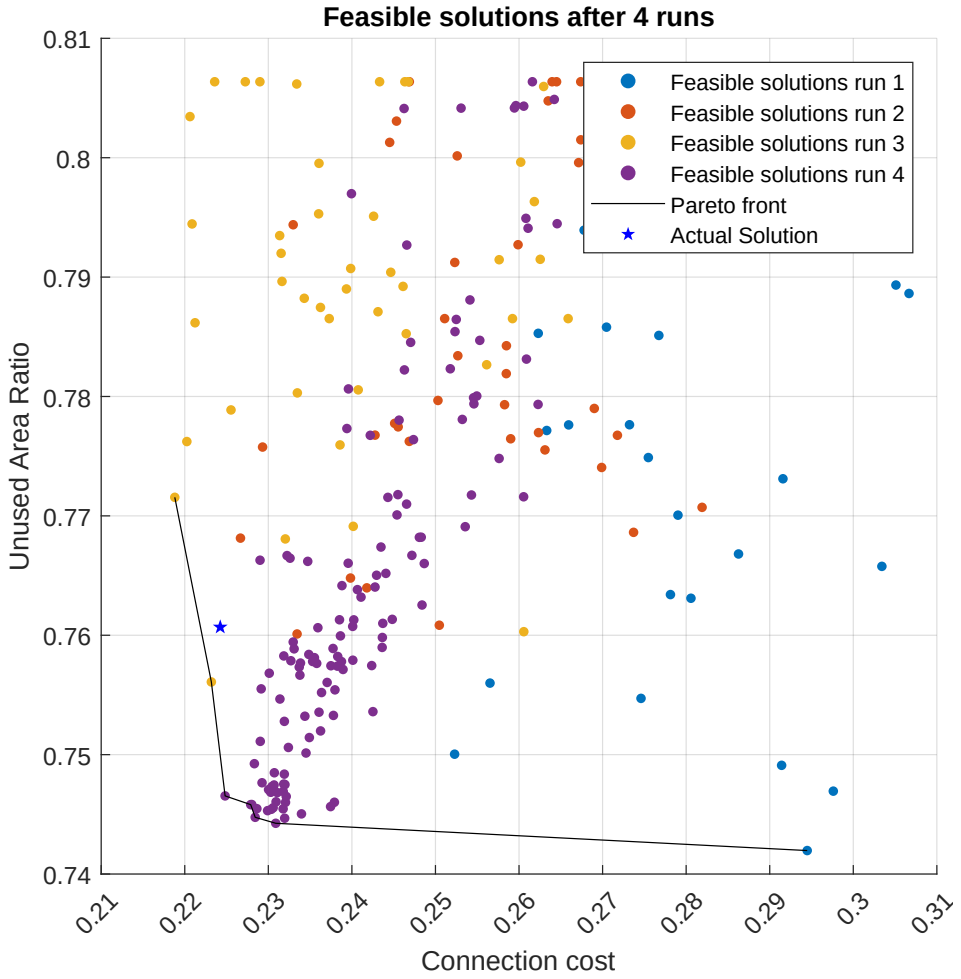


Figure 5.8: Scatter plot of the runs with $w = 0.5$ and $c_1 = c_2 = 1.20$ in table 5.13.

converge to that position in the solution space and optimize the local minimum without resulting in meaningful change because of the reasons outlined in the previous section. This did lead to a lot of feasible solutions. With the larger and more sudden steps this behaviour does not appear as much when this mutation is enabled, although it can still be observed in run 4 of figure 5.8.

While a single run covers the solution space better than before, its actual exploration of the solution space is still limited. Therefore multiple runs with the same settings are still performed.

Due to the positive effects this mutation has on the model it is implemented in the final version.

5.3.4. Mutation: Improve Infeasible Particles

The last discussed mutation tries not to focus on the optimization aspect, but on the search aspect of exploring the solution space with the particle swarm. It tries to repair infeasible particles each iteration to a feasible solution. And in doing so to allow the whole swarm to find more diverse feasible solutions.

It does this by applying the facility movement function written for the previous mutation to a different set of facilities. Multiple variant were explored, in which for each infeasible particle a different amount of facilities are moved each iteration. This mutation performed best when all the components which have overlap are moved by the mutations

In this way the mutation acts similar to a repair function, which is a way to handle constraints in a particle swarm optimization (Helwig, 2010). It is however not a repair function however, as the mutation does not guarantee that it results in a feasible solution.

The process of the improve infeasible particles mutation is:

1. Identify all infeasible particles.
2. Select the first particle
3. Identify all facilities with overlap.
4. Order from most to least overlap.
5. Perform steps 4-10 from the improve nondominated particle mutation.
6. Select the next particle
7. End when all particles in the list have been mutated

An overview of the results of the runs can be seen in table 5.14. The scatter plots belonging to the runs shown in table 5.13 are figures A.21 and **??** in appendix A. An example of one of these scatter plots in shown in figure 5.9.

w	$c_1(w)$	$c_2(w)$	Number of feasible results			
			Run 1	Run 2	Run 3	Run 4
0.40	0.98	0.98	229	249	157	300
0.45	1.05	1.05	228	310	230	303
0.50	1.13	1.13	276	230	268	287
0.55	1.20	1.20	263	255	166	177
0.60	1.28	1.28	290	317	229	208
0.65	1.36	1.36	270	245	314	257
0.70	1.45	1.45	208	279	250	238
0.75	1.53	1.53	263	231	322	246
0.80	1.62	1.62	248	278	273	203
0.85	1.71	1.71	243	259	296	241
0.90	1.81	1.81	234	276	216	230

Table 5.14: Number of feasible solutions found with mutation: Improve Infeasible Particles variant 1 while varying w and $c_1(w) = c_2(w)$

As this mutation is performed on every infeasible solution every iteration, it has a very significant impact on the runtime, extending it to 1050+ seconds.

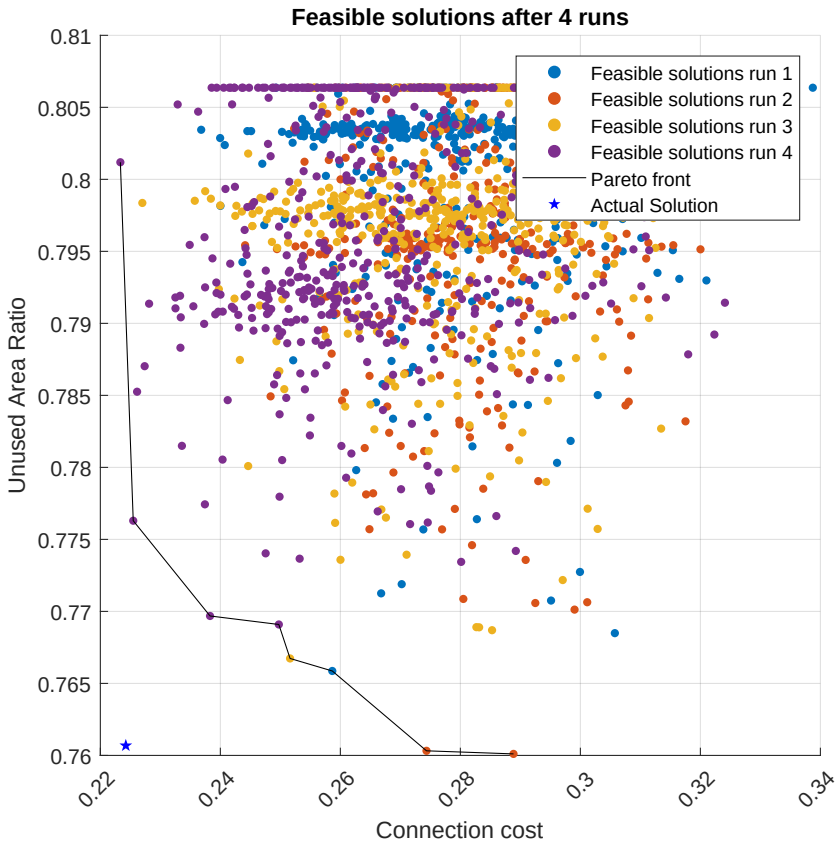


Figure 5.9: Scatter plot of the runs with $w = 0.5$ and $c_1(w) = c_2(w)$ in table 5.14.

With this mutation enabled the algorithm finds a lot more diverse solutions within a single run. However, the objective scores of these solutions rarely approach those of the actual solution designed by Vuyk. This is due to the mutations focus on repairing rather than optimizing. Infeasible particles are repaired by the mutation, sometimes to a feasible state. But once the feasible state is achieved it the algorithm still is not able to optimize this particle other than placing the facilities closer together. It is still stuck in the same local minima. And because the mutation disrupts the normal swarm behaviour where all the particles flock towards and then optimize the initially found feasible solution, the local minima does not really get explored. The results of this mutation are a lot of initially found feasible solutions. These solutions are diverse, but also unoptimized and the algorithm lacks the tools to do so. In order find one that scores comparably or better than the actual solution the algorithm essentially needs to get lucky with repairing an infeasible particle.

Due to the positive effects this mutation has on the model the first variant is implemented in the final version.

5.3.5. Combining Mutations

Both the improve nondominated particles mutation and the improve infeasible particles mutation improve upon the standard algorithm. As they both improve the results in a notably different way a run is performed with both mutations enabled to investigate whether they compliment each other.

As the improve infeasible particles mutation has a very significant effect on the runtime of the model the number of particles was lowered to speed up the process. Helwig (2010) provides an equation for the minimum number of particles scaled by the number of dimensions of the problem:

$$Np = 10 + 2 * sqrt(2 * D) = 10 + 2 * sqrt(2 * 60) = 32 \tag{5.1}$$

This does indeed significantly lower the runtime while the improve infeasible particles mutation is implemented, while there are still enough particles to generate diverse results.

Based on a large number of experiments with the model the settings that show the most interesting results are determined. Four of these experiments are presented in table 5.15.

w	$c_1(w)$	$c_2(w)$	Number of feasible results			
			Run 1	Run 2	Run 3	Run 4
0.5	0.5	0.5	205	158	143	123
0.5	1	1	132	157	151	160
0.5	1.5	1.5	241	172	177	191
0.5	2	2	203	211	176	198

Table 5.15: Number of feasible solutions found with mutations: Improve Nondominated Particles and Improve Infeasible Particles variant 1 enabled.

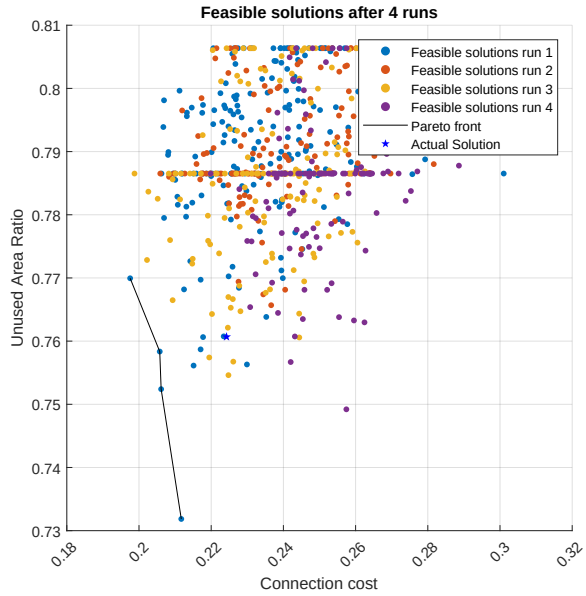
In figure 5.10 the scatter plots of the feasible solutions are presented corresponding to table 5.15. In these runs the acceleration coefficients c_1 and c_2 are varied in large steps to show their influence on the exploration of the search space.

As can be seen in this figure the interaction between the introduced mutations and the particle swarm algorithm seems best when the PSO coeffecients are set relatively low. When the coeffecients are set too high the particles take too large a step though the solution space at each iteration to find good results. This is the same behaviour that was observed as before the mutations were added.

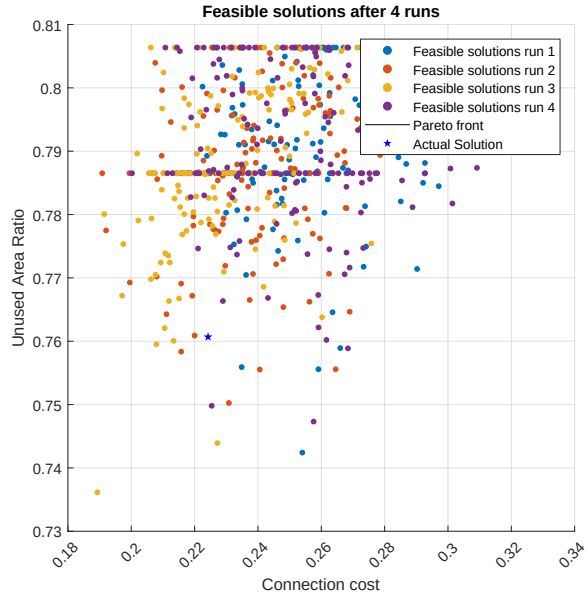
It is notable that in all four plots the model hits a barrier at a Unused Area Ration of 0.7865. This barrier is also present in most other runs that were performed for the jackup vessel. This is due to the x-position constraint of the crossover component. In all these solutions the crossover is placed as far back in the engine room as its constraint allows and determines the length of the engine room, were the starting point of the engine room is determined by a facility that is placed at length 0. This can be seen in figure 5.11a. The only way to minimize the Unused Area Ratio below this line is to place the facility that determines the starting point of the engine room length more forward. This is illustrated in figure 5.11b.

In order to identify the effect of both mutations a run was performed where the coefficients were set to zero. This effectively disables the velocity update equation of the algorithm and thus makes it purely rely on the introduced mutations. The results of these runs can be seen in figure 5.12.

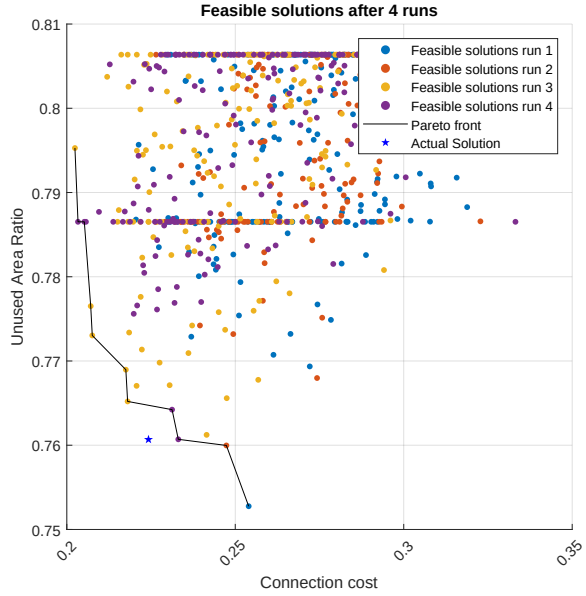
The model still finds results with these settings, and they are relatively well optimized. However, they also lack diversity. The results mainly come from the improve nondominated particle mutation, which is performed each iteration on the nondominated particles. Without the particle swarm movement it is very likely that the same particle is nondominated at each iteration, which means this particle is selected to be mutated every iteration. This lead to a relatively well optimized feasible solution, but also to a low solution diversity.



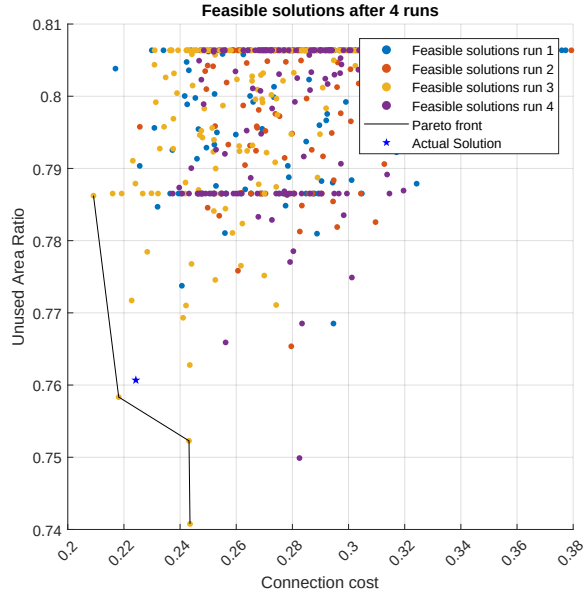
(a) $w = 0.50; c_1 = c_2 = 0.5$



(b) $w = 0.50; c_1 = c_2 = 1.0$

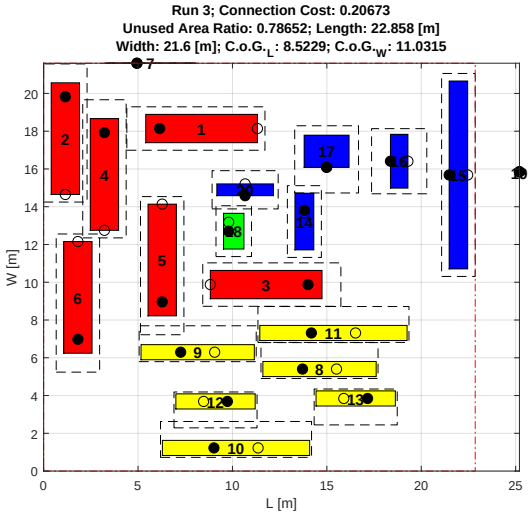


(c) $w = 0.50; c_1 = c_2 = 1.5$

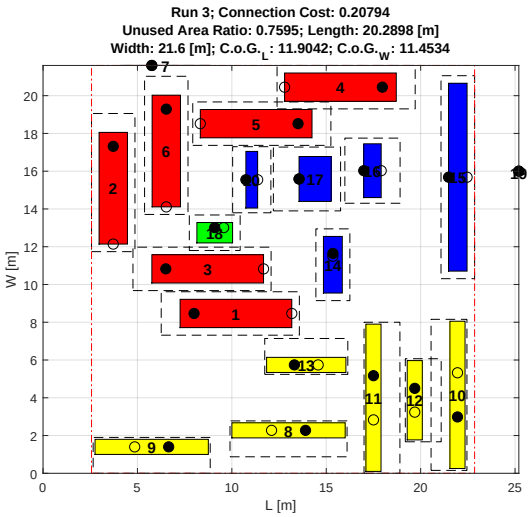


(d) $w = 0.50; c_1 = c_2 = 2.0$

Figure 5.10: Scatter plot of solutions found with mutations: Improve Nondominated Particles and Improve Infeasible Particles variant 1 enabled.



(a) This layout is where the model hit the Unused Area Ratio barrier of 0.7865. The generator facility 2 defines the starting length of the engine room at 0, and the crossover facility 15 defines the ending length of the engine room for a total length of 22.858. This is the furthest back placement that the crossover position constraints allow.



(b) This layout in the same run has passed the Unused Area Ratio barrier. The crossover facility 15 is still placed as far back as its position constraints allow, but the engines that decide the starting point of the engine room have been placed further forwards.

Figure 5.11: Two layouts plotted from run 3 shown in figure 5.10b

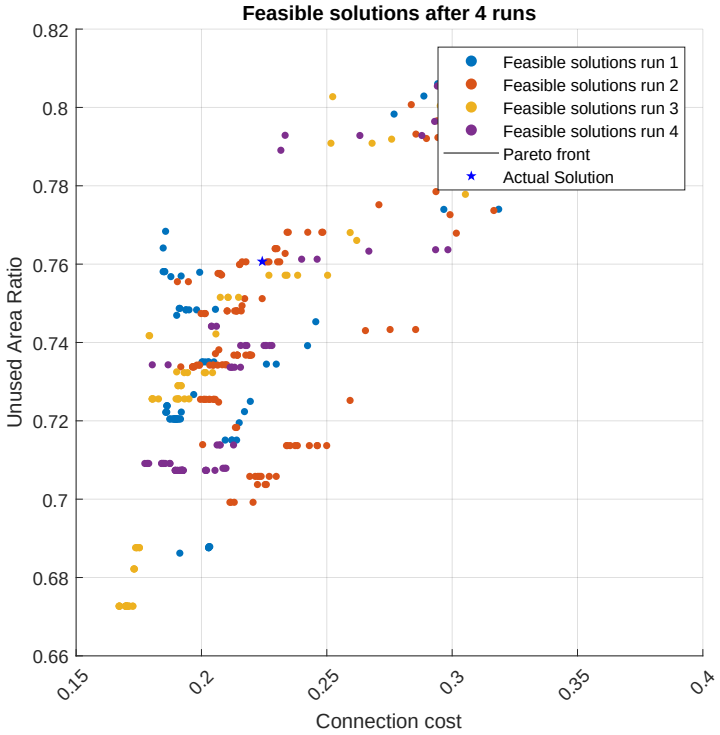


Figure 5.12: Scatter plot with both mutations and the coefficient all set to $w = c_1 = c_2 = 0$. This makes the algorithm rely purely on the mutations.

5.3.6. Conclusions about the PSO coefficient settings

Due to the random nature of the PSO algorithm and the mutations it is difficult to determine the influence of the PSO coefficients w, c_1 and c_2 upon the behaviour of the model. The improve nondomintated particles and improve infeasible particles mutations allow for better exploration than the bare PSO algorithm. They even generate results pureply by themselves, as is shown in figure 5.12. However, to create diverse results within a run the influence of these mutations must be balanced against the swarm behaviour. Testing showed that generally low settings for the PSO coefficients provide the best balance between the swarm behaviour and the implemented mutations, although this is very dependant on the randomness of the individual runs.

Because of this it was decided that the settings that the model is run with are $w = c_1 = c_2 = 0.5$.

5.3.7. Revisiting constraints

Now that a working algorithm has been identified the results need to be investigated. As can be seen in the plotted layouts in figure 5.11 the algorithm succeeds in placing the facilities together according to the guidance provided by the relationship matrix. However, several unwanted results can still be seen in these plots.

One such thing can already be handled by the model. The engines of the generator sets should generally be placed in the longitudinal ship direction for operability reasons. This can simply be enforced by the setting the rotation position constraint of facility one through six to 0.1. The updated position constraints can be seen in table 5.16.

Classification			Position constraints Expressed in percentage of Length and Width space					
Nr.	Component	Type	Xmin [0..1]	Xmax [0..1]	Ymin [0..1]	Ymax [0..1]	Rmin [0..1]	Rmax [0..1]
1	Caterpillar 3516B	Genset	0	1	0	1	0.1	0.1
2	Caterpillar 3516B	Genset	0	1	0	1	0.1	0.1
3	Caterpillar 3516B	Genset	0	1	0	1	0.1	0.1
4	Caterpillar 3516B	Genset	0	1	0	1	0.1	0.1
5	Caterpillar 3516B	Genset	0	1	0	1	0.1	0.1
6	Caterpillar 3516B	Genset	0	1	0	1	0.1	0.1
7	Ventillation exit	Interface	0	0.2	1	1	0.1	0.1
8	Switch board 690V	Switch boa	0	1	0	1	0	1
9	Switch board 690V	Switch boa	0	1	0	1	0	1
10	Switch board 440V	Switch boa	0	1	0	1	0	1
11	Switch board 440V	Switch boa	0	1	0	1	0	1
12	Switch board 110V	Switch boa	0	1	0	1	0	1
13	Switch board 110V	Switch boa	0	1	0	1	0	1
14	Ballast Water Treatme	Ballast Wat	0	1	0	1	0	1
15	Crossover	Crossover	0.9	1	0.95	1	0.4	0.4
16	SCW Pump Skid	Pump	0.5	1	0	1	0	1
17	LT Heat Exchanger	Heat excha	0	1	0	1	0	1
18	Fuel Seperator unit	Separator	0	1	0	1	0	1
19	Crossover exit	Interface	1	1	0	1	0.1	0.1
20	LT CW pump skid	Pump	0	1	0	1	0	1

Table 5.16: Updated position constraints with the rotation constraint for the generator sets.

The results of the runs with these new constraints is shown in figure 5.13.

The layout in figure 5.14a is similar to actual solution: the generator sets in the back aligned over the width of the engine room, the switchboards are all at starboard and the cooling systems at port side. The connection cost of this layout is in fact lower than that of the actual design, as the low temperature cooling water pump skid and the 690V switchboards both connect to the generator sets and are placed closer to these generators.

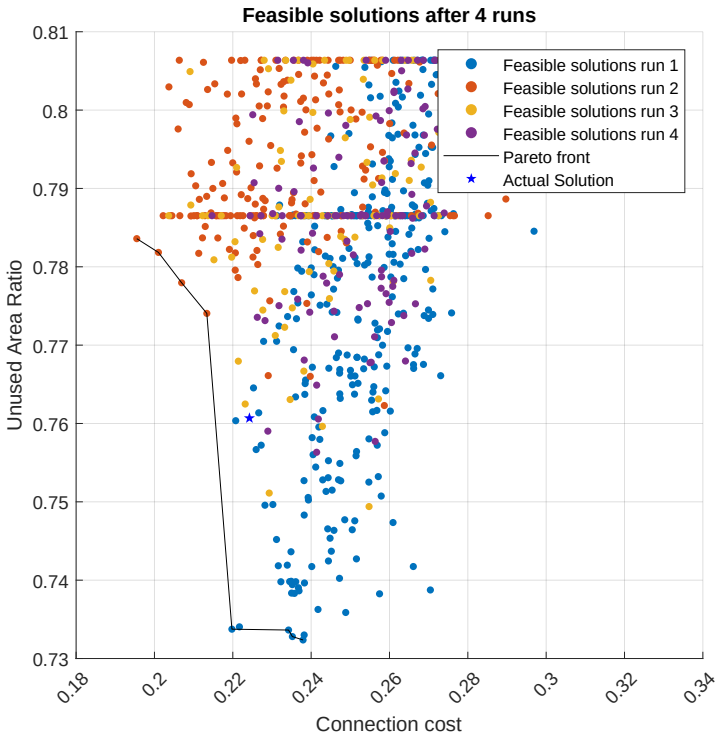
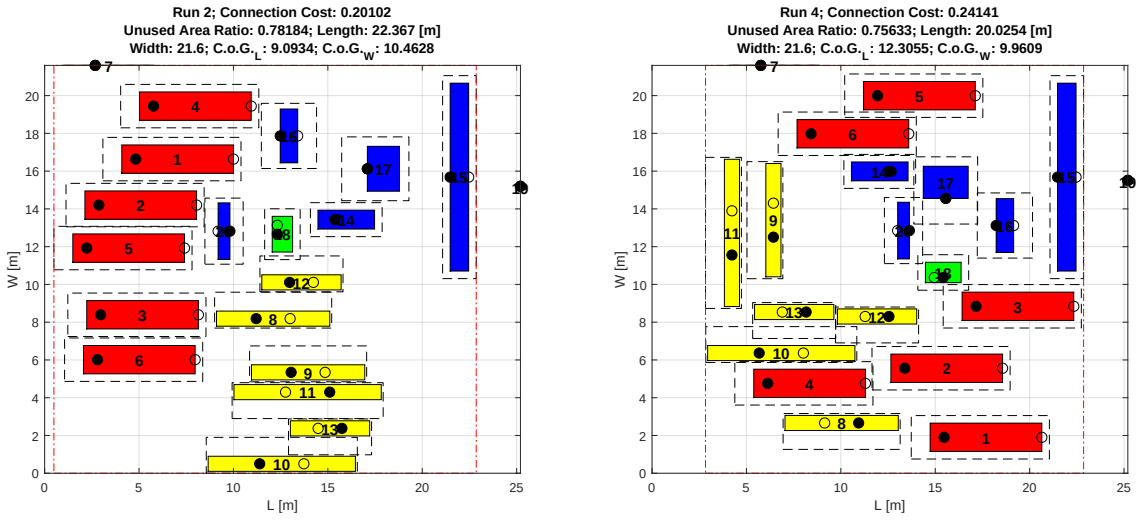


Figure 5.13: Scatter plot of the feasible solutions found with the updated position constraints shown in table 5.16.



(a) Pareto front layout from run 2

(b) Nondominated layout from run 4

Figure 5.14: Two layouts plotted from figure 5.13

The layout in figure 5.14b is one of the best layouts found in run 4 plotted in figure 5.13. While some logic according to the relationship matrix can be observed, such as the placement of the cooling systems, or the engines being placed close to the 690V switchboards, it is obvious that chaotic placement of the generator sets and the switchboards is not good design.

In order to remedy these deficiencies two additions to the model have been developed. The first is a grouping constraint which is discussed next in section 5.4. The second is a pathing module which is discussed in section 5.5.

5.4. Grouping Constraint

Sometimes it is necessary, instead of just desirable, that a certain set of components ('facilities' in the model) are placed in close proximity to each other without interference of other facilities. Because this is not sufficiently achieved by the relationship matrix a new constraint has been developed, which allows a set of components to be assigned to a $group$. The envelope around the different components of the group is calculated, and a check for overlap is performed between this envelope and all facilities not belonging to the defined $group$. This will incur a penalty when another component is placed in between two facilities in the same group.

In the input table an extra column is defined where every facility can be allocated to a group. By default all the facilities are placed in group 1, which doesn't activate the constraint. If a facility is placed in another group this constraint is activated.

Every iteration, an extra envelope around all the components in the group is calculated. This is done by taking the minimum and maximum x- and y-coordinates of all the facilities in the group. Then, using the overlap equations 4.14 and 4.15 the overlap between the group and all facilities not belonging to the group is calculated. This calculated overlap is then used in a penalty function to enforce the grouping constraint.

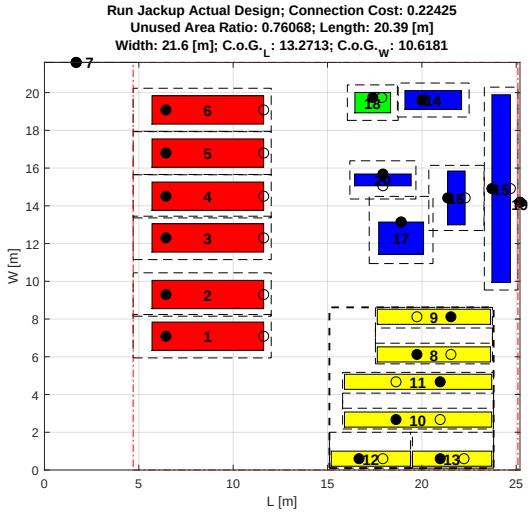
With this constraint added to the model, the new input for the jackup vessel can be defined. Aside from the default group 1, in this case two new groups are defined. The first group are the switchboards, facilities 8-13. These need to be placed in a separate space from the other components, as they need to be in a climatised space. The second new group is the crossover and the crossover exit, facilities 15 and 19. By placing these in a group together it is ensured that no components will be placed forward of the crossover, and the crossover position constraint can now be relaxed to allow it to be placed further back in the engine room. This enables the model to minimize the Unused Area Ratio objective. This newly defined input can be seen in table 5.17.

Classification			Position constraints						Other
Nr.	Component	Type	Xmin [0..1]	Xmax [0..1]	Ymin [0..1]	Ymax [0..1]	Rmin [0..1]	Rmax [0..1]	Group
1	Caterpillar 3516B	Genset	0	1	0	1	0.1	0.1	1
2	Caterpillar 3516B	Genset	0	1	0	1	0.1	0.1	1
3	Caterpillar 3516B	Genset	0	1	0	1	0.1	0.1	1
4	Caterpillar 3516B	Genset	0	1	0	1	0.1	0.1	1
5	Caterpillar 3516B	Genset	0	1	0	1	0.1	0.1	1
6	Caterpillar 3516B	Genset	0	1	0	1	0.1	0.1	1
7	Ventillation exit	Interface	0	0.2	1	1	0.1	0.1	1
8	Switch board 690V	Switch board	0	1	0	1	0	1	2
9	Switch board 690V	Switch board	0	1	0	1	0	1	2
10	Switch board 440V	Switch board	0	1	0	1	0	1	2
11	Switch board 440V	Switch board	0	1	0	1	0	1	2
12	Switch board 110V	Switch board	0	1	0	1	0	1	2
13	Switch board 110V	Switch board	0	1	0	1	0	1	2
	Ballast Water	Ballast Water							
14	Treatment Unit	Treatment Unit	0	1	0	1	0	1	1
15	Crossover	Crossover	0.6	1	1	1	0.4	0.4	3
16	SCW Pump Skid	Pump	0	1	0	1	0	1	1
17	LT Heat Exchanger	Heat exchanger	0	1	0	1	0	1	1
18	Fuel Seperator unit	Separator	0	1	0	1	0	1	1
19	Crossover exit	Interface	1	1	0	1	0.1	0.1	3
20	LT CW pump skid	Pump	0	1	0	1	0	1	1

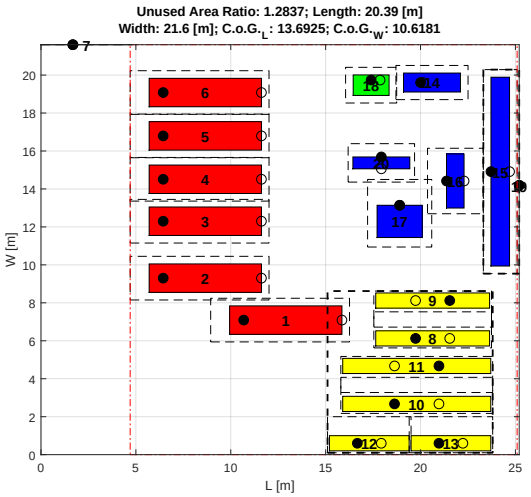
Table 5.17: Updated position constraints with the assigned groups and the relaxed crossover constraint.

With this newly defined input the model is run anew. The scatter plot of this run can be observed in figure 5.16.

As can be seen in figure 5.17 the grouping constraint enables the successful grouping of components.



(a) The actual design with the grouping envelope plotted.



(b) Violation of the defined grouping constraint.

Figure 5.15: Illustration of the grouping constraint

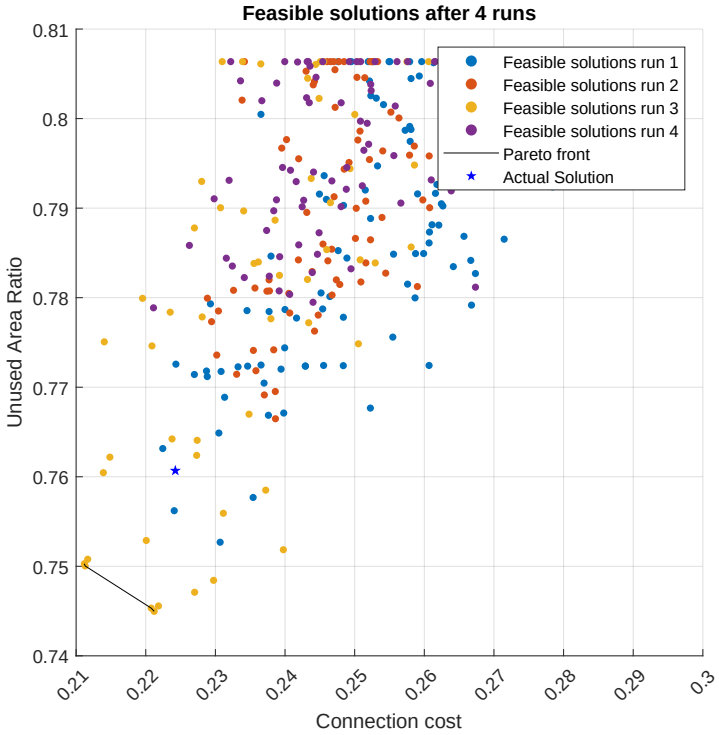
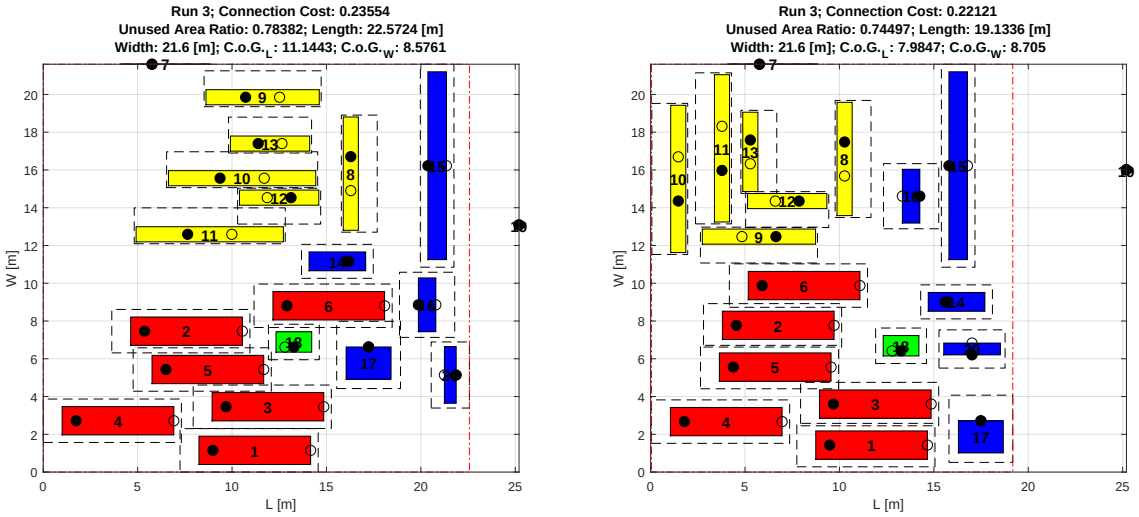


Figure 5.16: Scatter plot of the feasible solutions of the grouping runs

However, constraining the model in this way comes at a cost in solution diversity. Where the implemented mutations helped with escaping the local minima, in the case that such a large group of facilities is constrained to a group, the mutations are unable to help the model escape from the newly introduced local minima.

While the mutations, in particular the improve nondominated particles mutation, are able to move the components sequentially they are still moved one at a time. It is too big a change from the starting layout for the mutation to change the location of the group of components as a whole.



(a) The first found feasible solution of run 3 in figure 5.16. (b) Pareto front solution of run 3 in figure 5.16.

Figure 5.17: Illustration of the grouping constraint

This can be seen is figure 5.17, where the left hand figure 5.17a show the first found feasible solution of run 3 plotted in figure 5.16. While still somewhat chaotic, is it better than the initial solutions found without this grouping constraint. The ...

The right hand side figure 5.17b shows a layout from the Pareto front of run 3, which is also the Pareto front of all the runs combined. The similarity to the first found feasible solution is clearly visible. The group of switchboards was able to be moved backwards because the free space was there, but it hasn't changed it's position relative to the other components, which is the same for all solutions found in the runs. A group of this size thus captures the model into a local minimum again, but the designer can run the model again if he wishes to investigate solutions with the group in another section of the engine room.

The placement of facilities within the group was improved with respect to both objectives, with facility 8 and 9 moving closer to the generator sets to which they are connected. The placement of the other facilities has also improved. While generator set facilities 1, 3 and 4 have very little room to move even with the mutations, the other generator set facilities, especially 5 and 6, move closer to the exhaust facility and to the switchboards they are connected to. Because the crossover is now placed in a group with the crossover exit and its position constraint has been relaxed, it can now be placed further backwards than before, allowing for better optimization of the area minimization objective while keeping the crossover as (one of) the most forward component. This use of the grouping constraint works well.

As mentioned before, the connection between the engines of the generator sets and the exhaust ventilation point are very important. This connection already has the highest value of 5 in the relationship matrix, but there are still several runs where the model is incapable of placing all the generator sets near the exhaust ventilation exit. That is why a new input is defined, where all the generator sets are grouped together with the exhaust ventilation exit. In order to not 'over-constrain' the problem the switchboards are no longer placed in a group, instead the model is trusted to be able to place them logically together when the generator sets are no longer capable of mixing with them.

The changed input for the grouping for these runs can be seen in the most right hand column in table 5.18.

With this newly defined input the model is run anew. The scatter plot of this run can be observed in figure 5.18.

Figure 5.19 shows an example of the progress made during the third run of the model, from one of

Classification			Position constraints						Other
Nr.	Component	Type	Xmin [0..1]	Xmax [0..1]	Ymin [0..1]	Ymax [0..1]	Rmin [0..1]	Rmax [0..1]	Group
1	Caterpillar 3516B	Genset	0	1	0	1	0.1	0.1	2
2	Caterpillar 3516B	Genset	0	1	0	1	0.1	0.1	2
3	Caterpillar 3516B	Genset	0	1	0	1	0.1	0.1	2
4	Caterpillar 3516B	Genset	0	1	0	1	0.1	0.1	2
5	Caterpillar 3516B	Genset	0	1	0	1	0.1	0.1	2
6	Caterpillar 3516B	Genset	0	1	0	1	0.1	0.1	2
7	Ventillation exit	Interface	0	0.2	1	1	0.1	0.1	2
8	Switch board 690V	Switch board	0	1	0	1	0	1	1
9	Switch board 690V	Switch board	0	1	0	1	0	1	1
10	Switch board 440V	Switch board	0	1	0	1	0	1	1
11	Switch board 440V	Switch board	0	1	0	1	0	1	1
12	Switch board 110V	Switch board	0	1	0	1	0	1	1
13	Switch board 110V	Switch board	0	1	0	1	0	1	1
	Ballast Water	Ballast Water							
14	Treatment Unit	Treatment Unit	0	1	0	1	0	1	1
15	Crossover	Crossover	0.6	1	1	1	0.4	0.4	3
16	SCW Pump Skid	Pump	0	1	0	1	0	1	1
17	LT Heat Exchanger	Heat exchanger	0	1	0	1	0	1	1
18	Fuel Seperator unit	Separator	0	1	0	1	0	1	1
19	Crossover exit	Interface	1	1	0	1	0.1	0.1	3
20	LT CW pump skid	Pump	0	1	0	1	0	1	1

Table 5.18: Updated position constraints with the assigned groups and the relaxed crossover constraint.

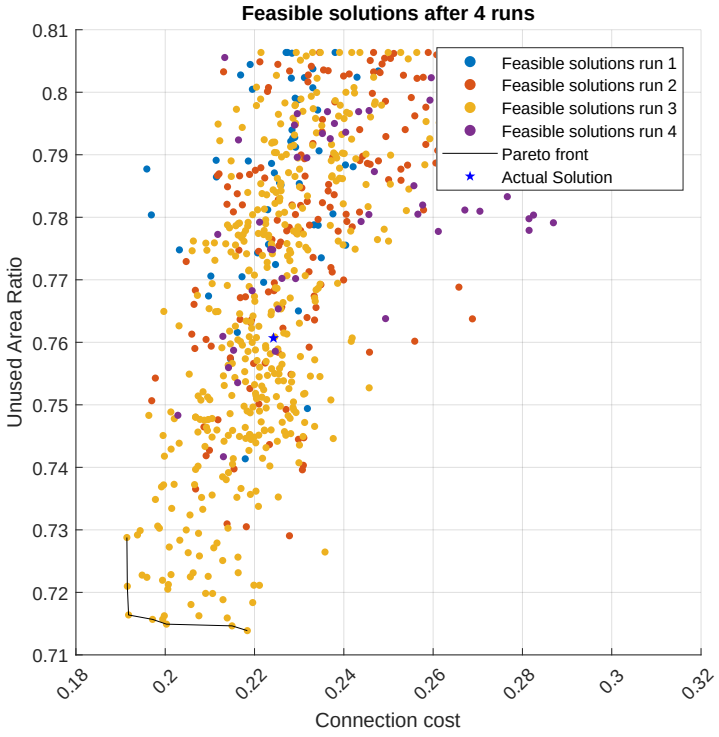
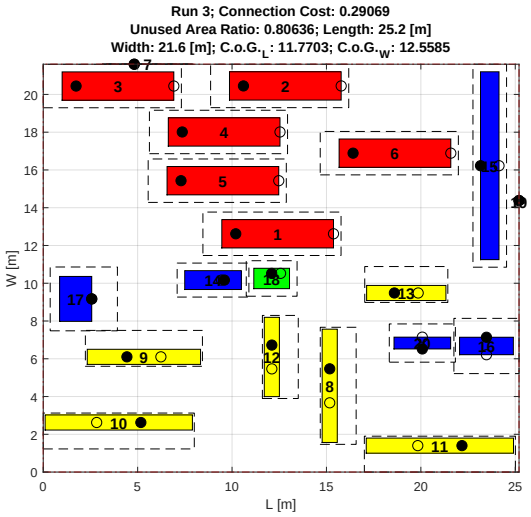
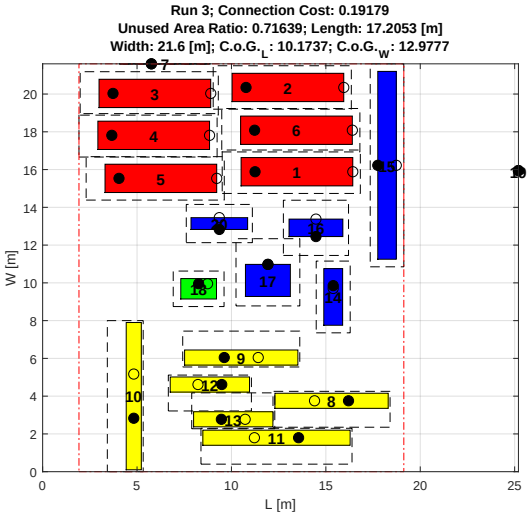


Figure 5.18: Scatter plot of the feasible solutions of the grouping runs with the revised constraints shown in table 5.18.

the first found solutions to a Pareto front solution. Because the exhaust ventilation exit is included in the group, all the gensets cluster around this point. Once the run found the 2x3 configuration it did not deviate from it. All the feasible solutions in run 3 that dominate the actual solution have this configuration, and differ only in the arrangement of the other facilities. But due to the mutations there is still variety in the placement of these facilities. Most solutions that dominate the actual solution even separate the switchboards from the other facilities even though they are not grouped, successfully following the values defined in the relationship matrix.



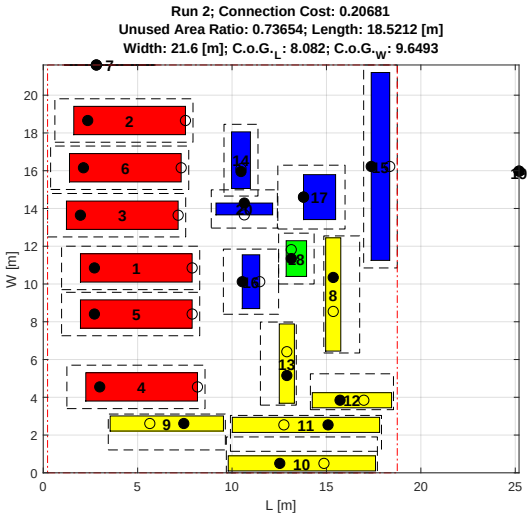
(a) One of the first found feasible solution of run 3 in figure 5.18.



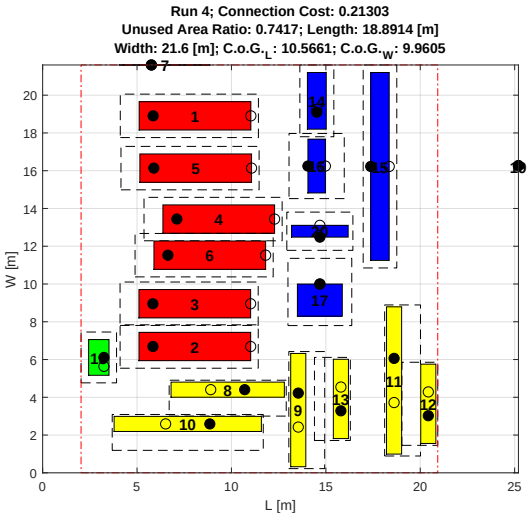
(b) Pareto front solution of run 3 in figure 5.18.

Figure 5.19: Plots of feasible solutions belonging shown in figure 5.18.

The other runs all converged on a grouping of the engines that is the same as in the actual Vuyk design solution, as is illustrated by the Pareto front solutions plotted of run 2 and 4 plotted in figure 5.20.



(a) Pareto front solution of run 2 in figure 5.18.



(b) Pareto front solution of run 4 in figure 5.18.

Figure 5.20: Plots of feasible solutions belonging shown in figure 5.18.

Overall the grouping constraint is a useful addition to the model. While it tends to over-constrain the problem, which makes the solution diversity suffer, it allows for better implementation of designer intentions than the position constraints alone.

5.5. Pathing Module

In section 2.4 the design requirements for the tool were determined, one of which is that the model must account for the main design drivers identified earlier in chapter 2. These main design drivers (as clarified in that chapter) are:

1. Accesses and escapes
2. Transport routes and walkways
3. Space for maintenance
4. Separation distances
5. Ventilation

When analysing the layouts that were generated in the previous sections, these design drivers have only been accounted for in a limited fashion.

Accesses and escapes such as doors or stairwells can be defined as facilities in the model. But aside from reserving space by modelling them as a facility with a clearance around them they would not interact with the model.

Transport routes and walkways are also accounted for in a limited fashion. The clearances around the facilities generally allow for personnel movement through the engine room, and when this is not the case this can easily be identified by a designer looking at the model. For transport routes this is not as apparent though, as different engine room components require different paths of different widths to be able to transport spare parts or entire replacement components through the engine room.

The *space for maintenance* around a component is accounted for by the ability to define a clearance for this purpose around the facility.

While the *separation distances* may be a useful design driver in the design of a process plant, in section 2.5 it was decided to not pursue the implementation of this design driver.

The *ventilation* design driver impacts various different aspects of the vessel layout. One of the main ways is by the necessary air in- and outlets that need to be placed in the vessel. But these are not present in the engine room layout itself, although their position can be taken into account by defining an outlet point (for ventilation and exhaust lines) and assigning relations, which was done with the jackup design in this chapter. Another important ventilation aspect which does affect the layout of the engine room is the routing of the exhaust lines from the engines to the defined exhaust outlet point. While it was initially decided to not route and model the individual connection between facilities, the exhaust ventilation is very substantial in size and is the reason that the connection between the engines and the exhaust ventilation point has a very high RM_{ij} value. Due to the size of this connection it cannot be neglected in every case.

While most of the main design drivers are accounted for, either completely or partly, there is a clear gap in the model. This concerns the unused or negative space left in a layout arrangement: is it possible to route all the necessary connections? Can all the facilities actually be reached by their necessary replacement components or units? When accesses and escapes are modelled, can a path from one to the other be guaranteed? In order to solve these questions these 'pathing requirements' must be incorporated into the model.

5.5.1. Solution Directions for the Pathing Problem

There are multiple ways to tackle this problem. The facility layout problem, when aisles are incorporated, usually handles it by predefining pathways such as Lee et al. (2005) did when applying this problem to vessel design. However, predefined aisles would heavily influence the solution space, limiting the exploration ability of the model. Also, the most obvious implementation of this method with the used model definition involves enforcing the overlap constraints between pathway and facility with a penalty function, which has already shown to limit the search and optimization ability of the model.

Another approach would be changing the FLP-formulation and algorithm to a nested particle swarm optimization. This would enable the model to generate an aisle layout in the first layer of the optimization, and then place the components for every aisle layout in a second PSO layer. However, this would

increase the runtime of the model drastically and also requires significant development time which was not available.

A third approach to this problem is to analyse the generated layouts to check if they incorporate the necessary pathways. Instead of predefining aisle structures or generating them during the optimization, an extra check is implemented for all the feasible solutions that the model finds.

In the research area of pipe routing network optimization techniques are a popular method Liu (2015). This field also deals with the analysis of the unused area of a layout arrangement. The main difficulty of network optimization methods is to properly define the nodes of the graph. Once a graph has been constructed, common network optimization techniques can be used to best solve the problem. While in this case the problem is not how to best route the pipes in this stage of the machinery space design, the same methods can be used to check whether paths exists between defined entries and exits, or engines and exhaust ventilation exits.

A common way to define the nodes is the escape graph method Liu (2015), which is illustrated in figure 5.21. It can guarantee a rectilinear shortest collision-free path in a space with obstacles. While it is not necessary to find the shortest path to comply to the design drivers, it is a good method to translate a machinery space to a graph in order to check if there is *a* path present where it needs to be.

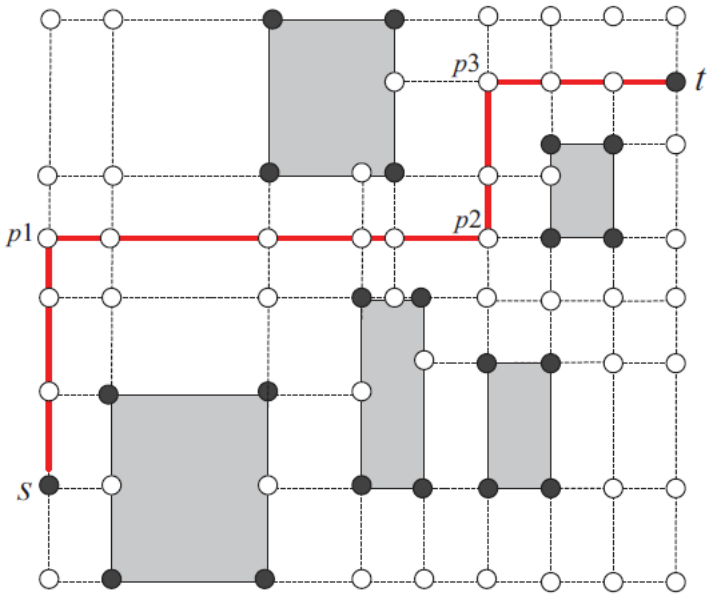


Figure 5.21: Illustration of an escape graph (Liu, 2015).

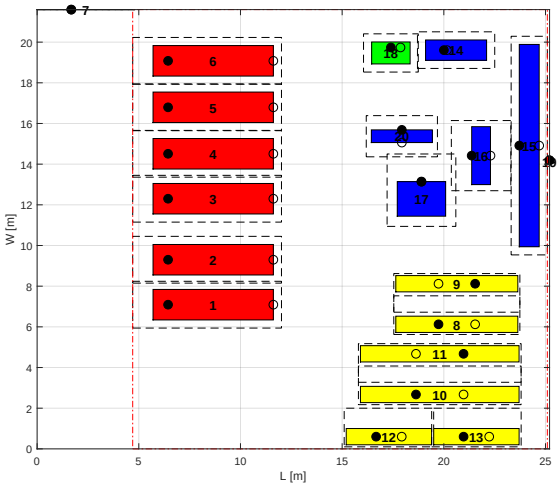
After each feasible solution is translated to a network diagram, for each node the horizontal and vertical space around it are calculated. Then a subgraph can be created of all the nodes of a certain predefined width, and a check is performed to see if there is a path present between the defined start and endpoints. Solutions which violate this are declared infeasible and removed from the feasible set.

5.5.2. Creating the Escape Graph

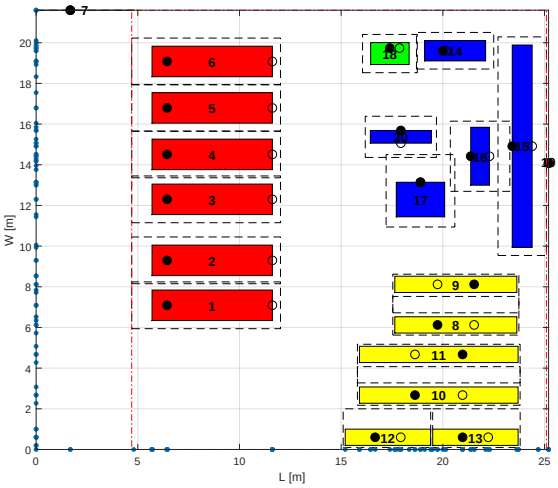
As can be seen in figure 5.21 the corners of the defined area for the components are used as the basis for the nodes in the escape graph. Because the start and endpoints used are the input and output gates of the facilities, it is important to make sure that they also exist as nodes in the created graph and are therefore taken in to account when constructing the graphs as well.

Figure 5.22 shows the process of creating an escape graph of an arrangement. In the implementation chosen for this problem, the input gate, output gate and two lower corners of each facility are projected on the x-axis, and the input gate, output gate and two left corners of each facility are projected onto

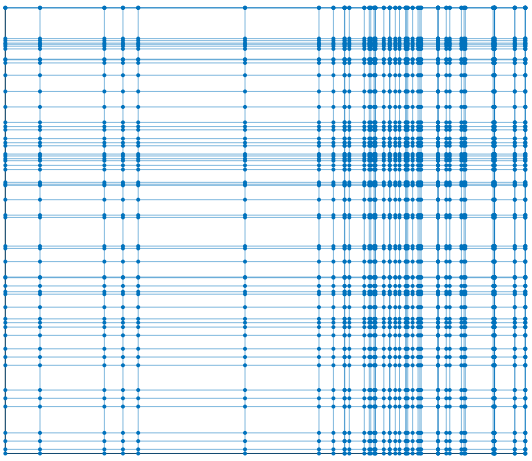
the y-axis. This is shown in figure 5.22b. The nodes created on these axes are then used to create a grid over the whole space. This grid contains the nodes on which the escape graph is based. All these nodes are then connected with the nodes directly horizontally and vertically adjacent. The value for these connecting edges is the distance between the nodes. With the nodes and edges defined in this way, the graph can be plotted, which is shown in figure 5.22c. In order to prevent paths from moving through the facilities the nodes and edges that lie within the facilities are removed. This results in the final escape graph shown in figure 5.22d. In this particular graph the edges of the facilities are highlighted in orange for clarity.



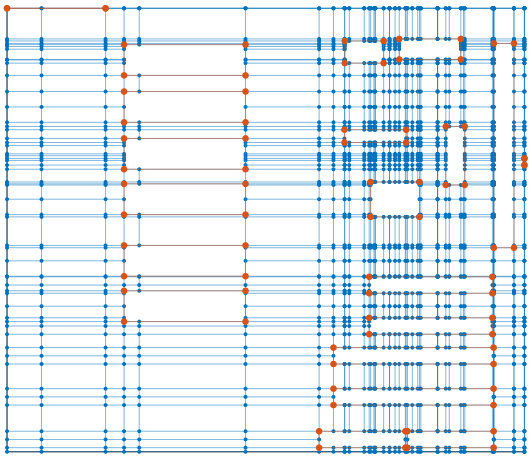
(a) The actual design



(b) Nodes projected on the x- and y-axis



(c) Graph created from the grid based on the projected nodes



(d) Final escape graph with the edges and nodes within the facilities deleted. The edges of the facilities are highlighted in orange for clarity

Figure 5.22: Illustration of the process of generating an escape graph for an arrangement.

After the graph is constructed the free space around each node is calculated, both horizontally and vertically. This is defined by the length that is free on the respective axis before either a facility or the area boundary is reached. So the bottom-left node in figure 5.22b has a horizontal space of the whole length of the engine room, or 25.6m, and a vertical space of the whole engine room, or 21.6m. While the bottom-left node of facility 1 in the same figure has a horizontal free space of 5.7m and a vertical space of 6.34m, which are the distances to the bounding area walls.

The constructed escape graph is now used to check whether a path of the predefined width exists between two defined facilities.

In order to do that, a new pathing matrix is defined in the input, which contains the minimum path

widths between all inputs and outputs of the components. This is the same structure that is used for the relationship matrix, only now instead of the RM_{ij} value a minimum path width value PM_{ij} is defined.

When the input or output gate of a facility is located within the facility its node has been removed from the graph. Therefore the node closest to the input or output gate is selected as the respective start or endpoint of the path.

For every defined path-width, a new subgraph is created from the escape graph. This subgraph contains only those nodes which have both the necessary horizontal and vertical space to comply to the defined minimum path-width requirement. From this subgraph then a check is performed to see if the starting and ending nodes still exist, and if they do exist then whether a path between them exists. If this is not the case the solution is removed from the feasible set.

5.5.3. Defining new Input

For the defined components of the jackup design the new pathing matrix is defined in figure 5.19. For this design only a necessary path from the output gate of the engines of the generator sets to the exhaust gate is defined. The exhaust lines have a diameter of 800 mm, as indicated in the matrix.

Pathing matrix: the values in this matrix give the requirement for the width of a path in [mm] that must exist between two componets	To this component's input																			
From this component's ouput	Comp 1: Caterpillar 3516B	Comp 2: Caterpillar 3516B	Comp 3: Caterpillar 3516B	Comp 4: Caterpillar 3516B	Comp 5: Caterpillar 3516B	Comp 6: Caterpillar 3516B	Comp 7: Ventillation exit	Comp 8: Switch board 690V	Comp 9: Switch board 690V	Comp 10: Switch board 440V	Comp 11: Switch board 440V	Comp 12: Switch board 110V	Comp 13: Switch board 110V	Comp 14: Ballast Water Treatment	Comp 15: Crossover	Comp 16: SCW Pump Skid	Comp 17: LT Heat Exchanger	Comp 18: Fuel Seperator unit	Comp 19: Crossover exit	Comp 20: LT CW pump skid
Comp 1: Caterpillar 3516B	0	0	0	0	0	0	800	0	0	0	0	0	0	0	0	0	0	0	0	0
Comp 2: Caterpillar 3516B	0	0	0	0	0	0	800	0	0	0	0	0	0	0	0	0	0	0	0	0
Comp 3: Caterpillar 3516B	0	0	0	0	0	0	800	0	0	0	0	0	0	0	0	0	0	0	0	0
Comp 4: Caterpillar 3516B	0	0	0	0	0	0	800	0	0	0	0	0	0	0	0	0	0	0	0	0
Comp 5: Caterpillar 3516B	0	0	0	0	0	0	800	0	0	0	0	0	0	0	0	0	0	0	0	0
Comp 6: Caterpillar 3516B	0	0	0	0	0	0	800	0	0	0	0	0	0	0	0	0	0	0	0	0
Comp 7: Ventillation exit	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Comp 8: Switch board 690V	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Comp 9: Switch board 690V	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Comp 10: Switch board 440V	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Comp 11: Switch board 440V	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Comp 12: Switch board 110V	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Comp 13: Switch board 110V	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Comp 14: Ballast Water Treatn	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Comp 15: Crossover	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Comp 16: SCW Pump Skid	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Comp 17: LT Heat Exchanger	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Comp 18: Fuel Seperator unit	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Comp 19: Crossover exit	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Comp 20: LT CW pump skid	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 5.19: Pathing matrix.

5.5.4. Applying the Pathing Module

As this addition to the model acts as a filter, its impact is shown by applying it to the results of the previous sections.

First the pathing filter is applied to the runs in which the grouping constraints were enabled, shown in figure 5.18 and 5.16. In both cases barely any feasible solutions were removed. To illustrate this the feasible solutions before and after application of the pathing filter are plotted side by side in figure 5.23. The Pareto front doesn't move, and most of the removed solutions were dominated by the actual solution designed by Vuyk.

As the results of the runs with the grouping constraints were already shown to be relatively orderly, this is not an unexpected result. Most of the solutions which were removed were made infeasible because one of the facilities was placed so close to the exhaust ventilation point that it prevented the point from being reached. In the case where the switchboards were grouped, which are the runs plotted in figure 5.23, some solutions where also made infeasible due to a generator set being enclosed by other

facilities.

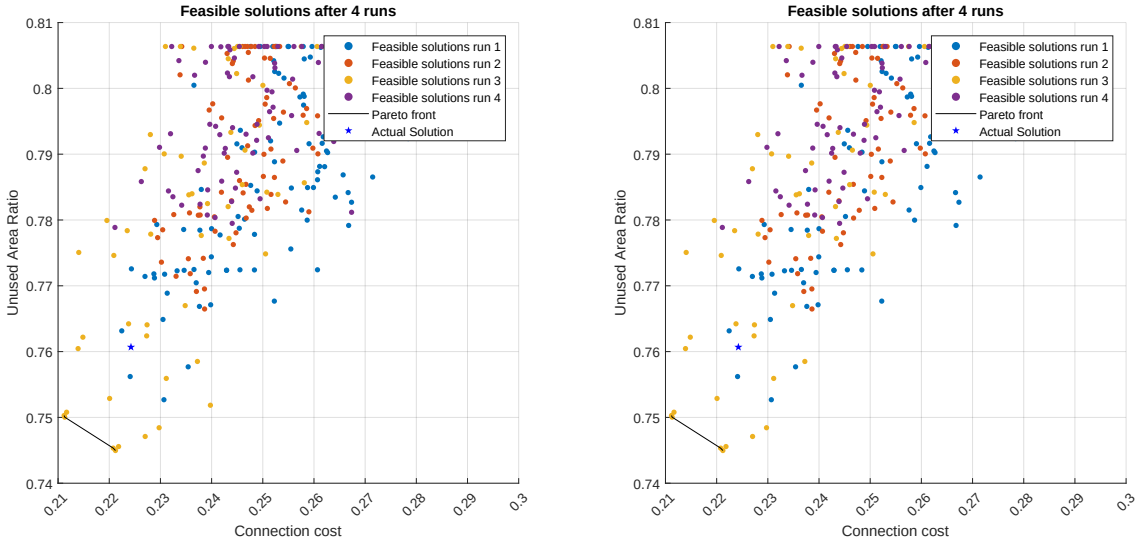


Figure 5.23: Side by side comparison of the feasible solutions removed by the pathing filter. The left hand figure is figure 5.18, the right hand figure are the feasible results remaining after the pathing filter was applied. Barely any results were removed.

Similar results are obtained when applying the pathing filter to the runs in which only the generator set orientation was constrained. The comparison between these results are seen in figure 5.24. It is notable that in this case a Pareto front layout was removed, although overall the module does not filter out many solutions.

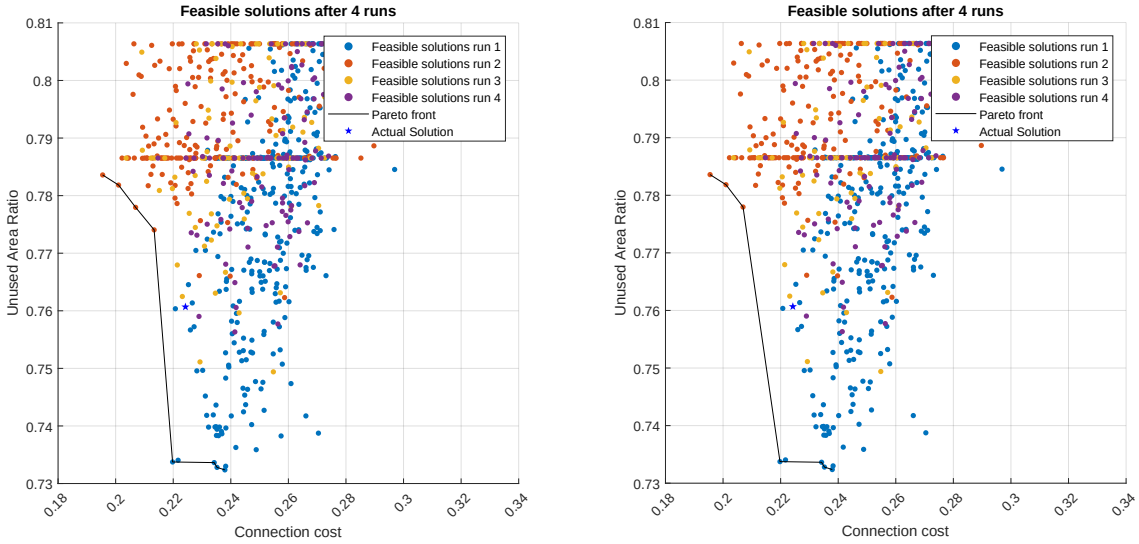
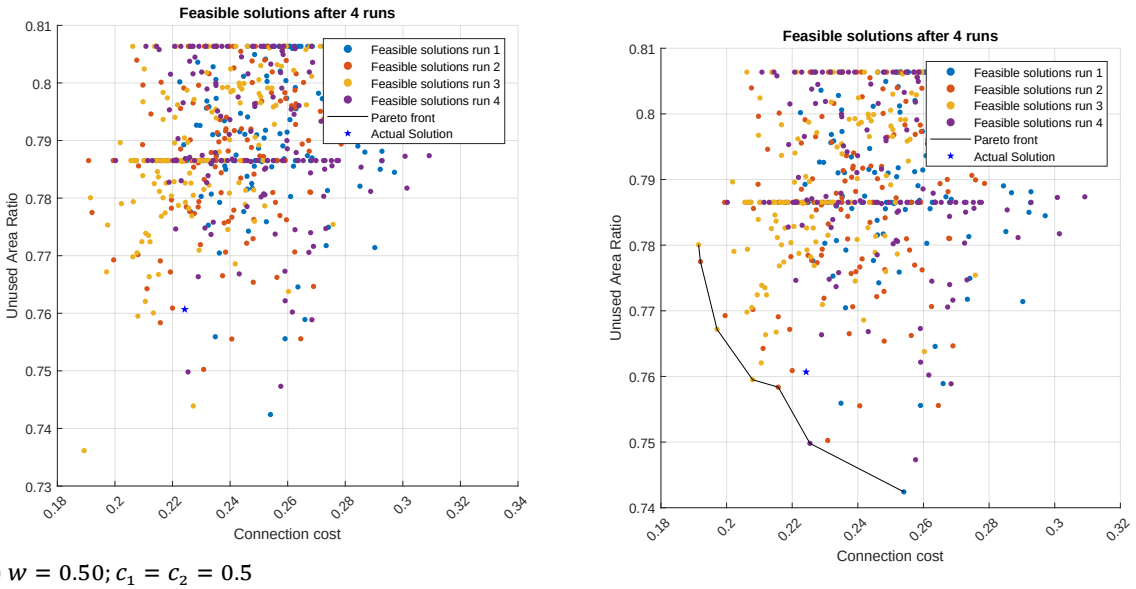


Figure 5.24: Side by side comparison of the feasible solutions removed by the pathing filter. The left hand figure is figure 5.13, the right hand figure are the feasible results remaining after the pathing filter was applied.

When investigating the effect on the least constraint runs that were performed, more, and better scoring, feasible solutions were removed. Notable is the comparison shown in figure 5.25. The original run has one solution which dominates all the others, but this solution is not feasible. Several of the other best scoring solutions are also removed. Thus an entirely new Pareto front is created.

In general the more constrained the model is, the more orderly the solutions become, reducing the effect of the pathing filter when checking these exhaust pathing requirements. When performing an exploration with minimal constraints it is a useful addition to filter out infeasible solutions. It has only been applied with respect to exhaust ventilation routing, and for future applications it would be inter-



(a) $w = 0.50; c_1 = c_2 = 0.5$

Figure 5.25: Side by side comparison of the feasible solutions removed by the pathing filter. The left hand figure is figure 5.10b, the right hand figure are the feasible results remaining after the pathing filter was applied.

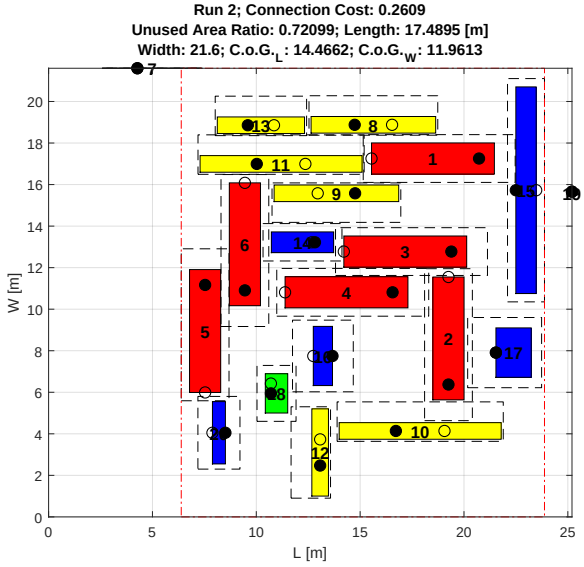
esting to apply this method to the other design drivers that deal with paths through the engine room, e.g. checking whether personnel paths exist towards a modelled escape, or paths for spare parts from a modelled store towards specified components.

5.6. Conclusions Model Testing and Adaptations

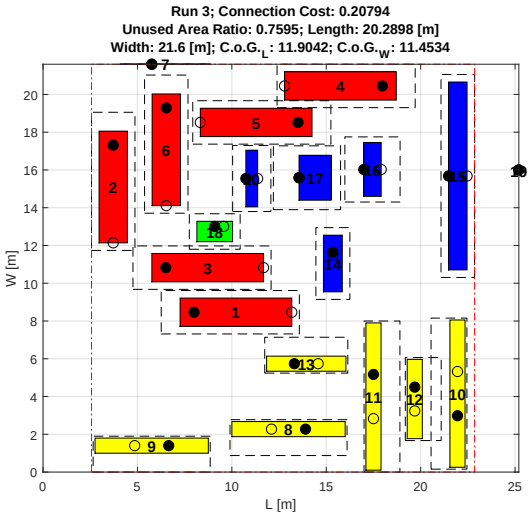
In this chapter the model as described in chapter 4 was tested and improved. First the input for this case was defined. It was found that for the model as defined in chapter 4 the penalty function method causes problems that the particle swarm algorithm cannot solve by itself. The swarm converges upon local minima in the solution space and cannot escape it. Tuning the penalty function did not prove the solution to this problem, as when less penalty was imposed the model was unable to properly distinguish feasible and infeasible solutions. Tuning the parameters of the algorithm also did not allow to model to escape these local minima.

Several alterations were made to combat this, called mutations. Two mutations were found to improve the model: the improve nondominated particles mutation and the improve infeasible particles mutation. When the algorithm was able to present diverse results these results were investigated. A new constraint was developed and implemented to group facilities together and better enforce the designers intent. Finally a new pathing module was implemented to better account for several machinery space design drivers.

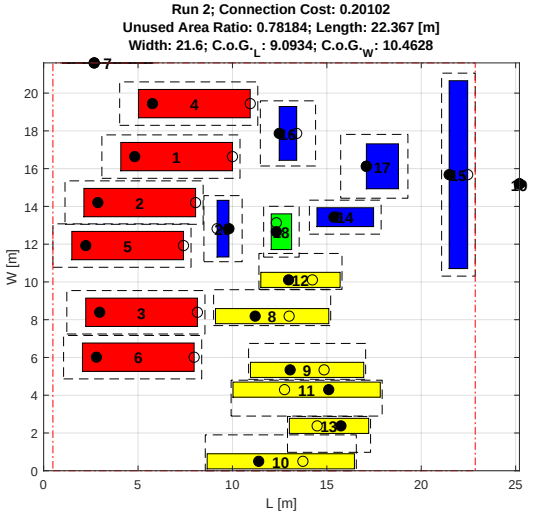
The additions to the model have improved the quality of the found solutions. When the algorithm got stuck in these local minima only very chaotic layouts were found, as only the local minima could be explored without the ability to reorder facilities. Such a layout is plotted in figure 5.26a. After the mutations were applied to enable to model to reorder the facilities and escape these local minima, the model was capable of optimizing the layouts to the designers intent represented by the relationship matrix. Such a layout is show in figure 5.26b. With the constraints added to constrict the orientation of the generator sets the model was able to better look for more realistic layouts. Here already solutions were found that resemble the actual solution designed by Vuyk. With the addition of the grouping constraint the model could be better forced to generate solutions which are less chaotic, but this comes at a cost of solution diversity. This is a very usable addition if the designer already has good ideas of the kind of layouts that are interesting to explore. Such a layout is shown in figure 5.26d.



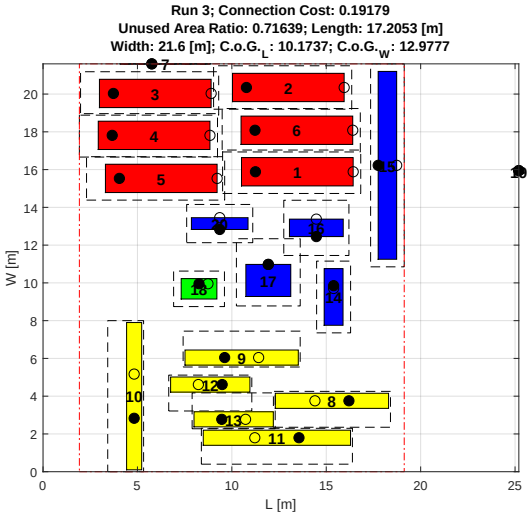
(a) Layout found by the initial runs, see figure 5.7b.



(b) Layout found after the implementation of the mutations, see figure 5.11b



(c) Layout found after revised constraints, see figure 5.14a.



(d) Layout found after grouping generator sets, see figure 5.19b.

Figure 5.26: Illustration of found layouts throughout the process of improving the model.

6

Model Application

Now that a working version of the model is available a new case study is performed to determine the usability for the designer.

In section 2.1 the overall design process of Vuyk and the design of the engine room within this process was discussed. In order to show how the tool helps the designer early in this design process it is used to investigate the effect of various engine configuration options that are possible for this design, and their effects on the arrangement.

The investigated design is of a trailing suction hopper dredger (TSHD), with two machinery spaces in the forward part of the ship: an engine room and a dredge pump room that can be combined to one space for the application of the tool.

The engine room of the example design alternative of Vuyk contains three diesel generator sets that provide the main power for the vessel. Next to the engine room is a pump room containing a dredge pump and a jet pump, which are specialized dredging equipment. These require considerable power to be driven. In the example concept design of Vuyk a solution is presented in which these pumps are driven directly by the diesel engines that also drive the generators. This example design is shown in figure 6.1.

As the vessel is diesel-electric, it is interesting to investigate in an early design phase the possibility of electric driven dredge and jet pumps, and the effect on the arrangement and size of both machinery spaces.

Within the design scenario for an electric driven dredge and jet pump, there are new possibilities for the amount of generator sets providing the power for the vessel. Generally, a larger number of generator sets is better able to be tuned to the specific required power of the various operational profiles of the TSHD. The arrangements of three different possibilities are investigated: a case with the same three diesel generators, a case with four diesel generators and a case with six diesel generators.

It is assumed that the design choice for a diesel electric vessel already has been made, with electric propulsion of the propellers. The total required power is known, and for the engine and pump room an area is assigned in the initial general arrangement, of a length of 16m and a width of 21.5m. The machinery space arrangements of the four design scenario's are investigated to determine the length requirement for the combined machinery space, and to investigate if the assigned length could be shortened.

The required power for this vessel can be delivered by the three diesel-generator sets in the Vuyk design. These are Caterpillar 3516B generator sets, so three engines in the 3500-series with each 16 cylinders. The total power is thus generated by 48 cylinders. A comparable power can be generated

by the same 48 cylinders spread over a different amount of engines: four Caterpillar 3512B generator sets with 12 cylinders each or six Caterpillar 3508B generator sets with 8 cylinders each.

So the four investigated scenarios are:

1. Three 3516B generator sets and diesel-direct driven pumps
2. Three 3516B generator sets and electric driven pumps
3. Four 3512B generator sets and electric driven pumps
4. Six 3508B generator sets and electric driven pumps

For all these scenarios the tool is used with the settings determined in chapter 5. The values of the particle swarm optimization coefficients w, c_1 and c_2 are all 0.5. The algorithm is run for 100 iterations. The number of particles for each scenario is determined by the minimum recommended particle formula presented earlier in section 5.3.5: 27 particles for the first three scenario's and 29 particles for the fourth scenario. For all the runs of the tool the 'improve nondominated particles' and 'improve infeasible particles' mutations are enabled.

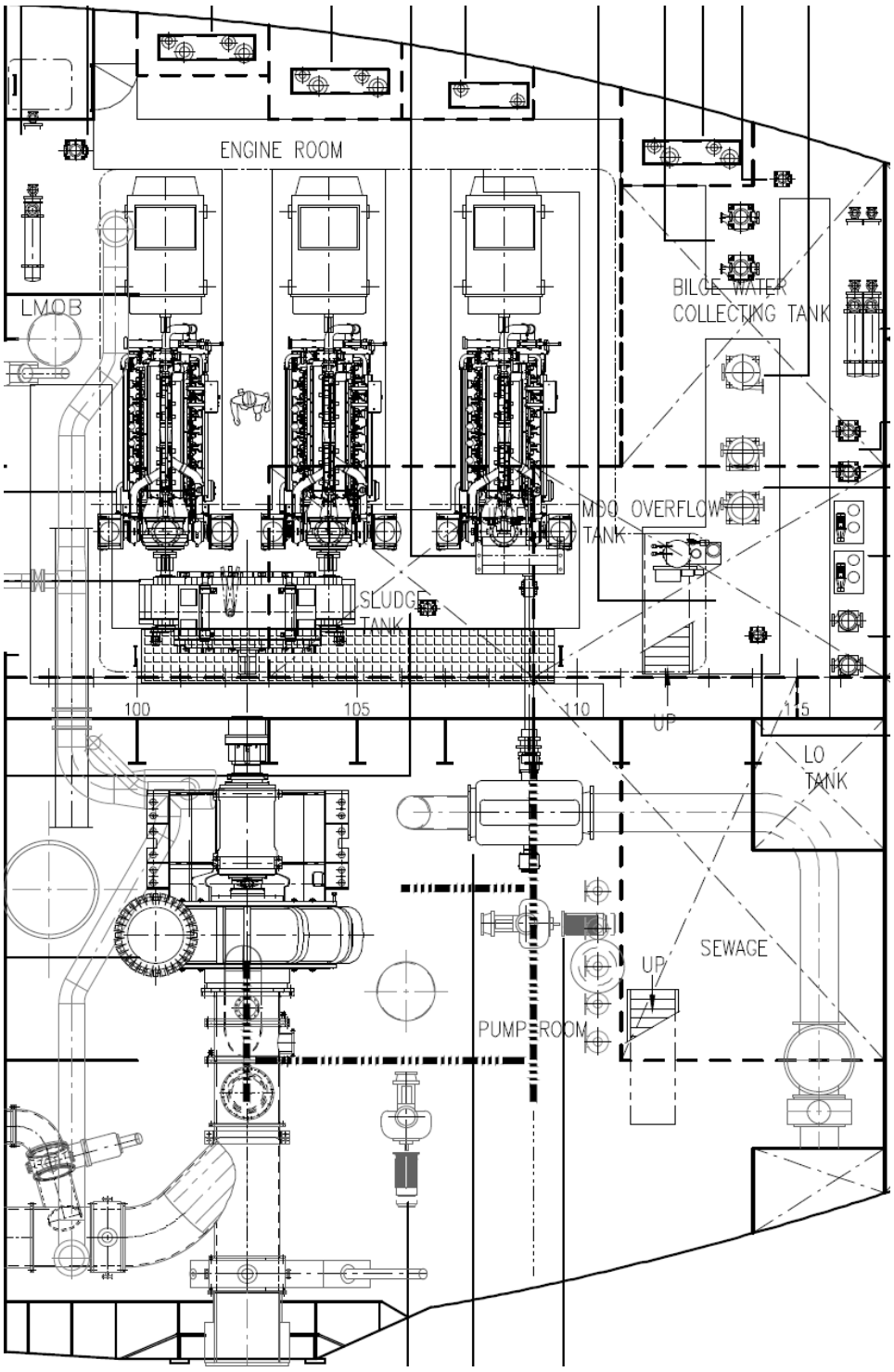


Figure 6.1: The engine room arrangement of the TSHD design.

6.1. Scenario 1: three 3516B generator sets and diesel-direct driven pumps

6.1.1. Input Scenario 1

Figure 6.1 shows the example design of Vuyk for the scenario in which the dredge and jet pump are directly driven by the diesel engines of the generator sets. There is a gearbox which connects the first two diesel engines with the dredge pump, and a gearbox which connects the third diesel engine with the jet pump. The diesel engines are also connected to the generators which provide the power for the propulsion- and other systems.

This design is used to define the components that are taken into account for this investigation. These components are presented in table 6.1 with their properties that are measured from the example design and otherwise provided by Vuyk.

Classification			General Physical properties			Clearances				In and output location				Other
			Length	Width	Weight	Left	Right	Up	Down	Xin	Yin	Xout	Yout	Plot
Nr.	Component	Type	[mm]	[mm]	[kg]	[mm]	[mm]	[mm]	[mm]	[0..1]	[0..1]	[0..1]	[0..1]	Color
1	Genset 1	Generator Set	6095	1500	18800	1000	600	1000	1000	0.5	0.5	0	0.5	red
2	Genset 2	Generator Set	6095	1500	18800	1000	600	1000	1000	0.5	0.5	0	0.5	red
3	Genset 3	Generator Set	6095	1500	18800	1000	600	1000	1000	0.5	0.5	0	0.5	red
4	Dredge pump	Pump	6000	3000	51000	2500	1000	1000	1000	0	0.5	0	1	blue
5	Jetpump	Pump	3500	2000	16000	1000	1000	1000	1000	0.2	0	0.2	1	blue
6	Sea inlet DP	Inlet	2000	1000	3000	100	100	400	0	0.5	0.5	0.5	1	blue
7	Sea inlet JP	Inlet	2000	1000	3000	100	100	400	0	0.5	0.5	0.5	1	blue
8	Funnel	Interface	4000	1	1	0	0	0	0	0.5	0.5	0.5	0.5	white
9	Water Pump Skid	Pump skid	5600	1200	2000	250	250	700	700	0.5	0	0.5	1	green
10	Fuel pump skid	pump skid	4500	900	2000	250	250	700	700	0.5	0	0.5	1	green
11	MDO tank interface	Interface	2000	1	1	0	0	0	0	0.5	0.5	0.5	0.5	white

Table 6.1: Properties of the modelled components of the direct-drive run

The first three components are the three Caterpillar 3516B generator sets. While these are the same generator sets that are used by the jackup vessel which was used in the previous chapter, in this design the generators are connected to the other end of the engine as the engines are also used to drive the pumps. In the plots of the arrangements these components are plotted in red.

Component four is the dredge pump, which is connected to the two diesel engines with a gearbox. The length and width of the component are those of the combined pump and gearbox. The dredge pump requires a large clearance around it, particular near its inlet where an inspection device needs to be fitted. The dredge pump is connected to the suction pipe inlet, here modelled as a generic sea inlet as component six.

Aside from the dredge pump, the pump room also contains a jet pump which is modelled as component number five. It is connected by a gearbox to the third diesel engine. This pump also requires a large clearance, and is connected to a sea inlet which is modelled as component seven. Both pumps and their inlets are plotted in blue.

A funnel for the exhaust ventilation has already been positioned in the design and is modelled as interface facility eight.

Component nine and ten are two pump skids which are modelled in order to ensure that enough space is reserved in the arrangements. The first pump skid contains various water pumps such as gland- and flushing pumps. The second pump skid contains the fuel pumps which supply the generator sets with marine diesel oil (MDO). This pump skid is connected to interface facility eleven, the MDO tank interface. This models the connection to the MDO tanks that are placed forward of the engine room. Both pump skids are plotted in green.

Table 6.2 shows the position and grouping constraints for the components in the first scenario. The constraints are mainly enforced using the grouping constraints.

Only the funnel and the water pump skid remain in the default group 1, all other components are assigned to a group. Group 2 consists of the first two generator sets, the dredge pump and its inlet.

Classification			Position constraints Expressed in percentage of Length and Width space						Other
Nr.	Component	Type	Xmin [0..1]	Xmax [0..1]	Ymin [0..1]	Ymax [0..1]	Rmin [0..1]	Rmax [0..1]	Group
1	Genset 1	Generator Set	0	1	0	1	0.4	0.4	2
2	Genset 2	Generator Set	0	1	0	1	0.4	0.4	2
3	Genset 3	Generator Set	0	1	0	1	0.4	0.4	3
4	Dredge pump	Pump	0	0.5	0	1	0.4	0.4	2
5	Jetpump	Pump	0	1	0	1	0	1	3
6	Sea inlet DP	Inlet	0.25	0.25	0	0	0.1	0.1	2
7	Sea inlet JP	Inlet	0.3	1	0	0	0.1	0.1	3
8	Funnel	Interface	0	0	0.5	0.5	0.4	0.4	1
9	Water Pump Skid	Pump skid	0	1	0	1	0	1	1
10	Fuel pump skid	pump skid	0	1	0	1	0	1	4
11	MDO tank interface	Interface	1	1	0.75	0.75	0.4	0.4	4

Table 6.2: Constraints of the modelled components of the direct-drive run

Together with their rotation constraints, this ensures that they will be positioned in line with each other.

Group 3 contains the third generator set, the jet pump and its inlet, for the same reasons.

Lastly the fuel pump skid and its interface are assigned to group 4. This is to ensure the forward placement of the fuel pump skid, without enforcing a too limiting position constraint.

In table 6.3 the relationship matrix is presented. The dredge pump and jet pump are connected by the most important connection of value five to their inlets. The same connection value is used to declare the relationship between the generator set output gate and the funnel. They have a connection value of zero with the other components as these are placed outside of the pump room. The fuel pump skid has an important connection to its interface with a value of four. Furthermore the fuel pump skid and the generators have the normal connection value of two in their relation. This is also the connection between the generators.

Relationship Matrix: the values in this matrix denominate the relavite importance that the in- and outputs of two components are placed near eachother		To this component's input										
		Comp 1: Genset 1	Comp 2: Genset 2	Comp 3: Genset 3	Comp 4: Dredge pump	Comp 5: Jetpump	Comp 6: Sea inlet DP	Comp 7: Sea inlet JP	Comp 8: Funnel	Comp 9: Water Pump Skid	Comp 10: Fuel pump skid	Comp 11: MDO tank interface
From this component's ouput	Comp 1: Genset 1	0	2	2	0	0	0	0	5	0	0	0
	Comp 2: Genset 2	2	0	2	0	0	0	0	5	0	0	0
	Comp 3: Genset 3	2	2	0	0	0	0	0	5	0	0	0
	Comp 4: Dredge pump	0	0	0	0	0	0	0	0	0	0	0
	Comp 5: Jetpump	0	0	0	0	0	0	0	0	0	0	0
	Comp 6: Sea inlet DP	0	0	0	5	0	0	0	0	0	0	0
	Comp 7: Sea inlet JP	0	0	0	0	5	0	0	0	0	0	0
	Comp 8: Funnel	0	0	0	0	0	0	0	0	0	0	0
	Comp 9: Water Pump Skid	0	0	0	0	0	0	0	0	0	0	0
	Comp 10: Fuel pump skid	2	2	2	0	0	0	0	0	0	0	0
	Comp 11: MDO tank interface	0	0	0	0	0	0	0	0	0	4	0

Table 6.3: Relationship Matrix for scenario 1

Table 6.4 shows the pathing matrix for scenario 1. Paths are defined from the generator sets to the funnel, and between the dredge and jet pump and their respective inlet.

6.1.2. Arrangements Scenario 1

In the first scenario the arrangements are investigated for the diesel-direct driven dredge and jet pumps and these arrangements are analysed on the resulting length and the connection costs.

This case is very constrained, as the engines and pumps need to be in line with each other and with the pump inlets.

Pathing matrix: the values in this matrix give the requirement for the width of a path in [mm] that must exist between two componets	Path to input of component: Comp 1: Genset 1	Comp 2: Genset 2	Comp 3: Genset 3	Comp 4: Dredge pump	Comp 5: Jetpump	Comp 6: Sea inlet DP	Comp 7: Sea inlet JP	Comp 8: Funnel	Comp 9: Water Pump Skid	Comp 10: Fuel pump skid	Comp 11: MDO tank interface
Path from output component: Comp 1: Genset 1	0	0	0	0	0	0	0	800	0	0	0
Comp 2: Genset 2	0	0	0	0	0	0	0	800	0	0	0
Comp 3: Genset 3	0	0	0	0	0	0	0	800	0	0	0
Comp 4: Dredge pump	0	0	0	0	0	0	0	0	0	0	0
Comp 5: Jetpump	0	0	0	0	0	0	0	0	0	0	0
Comp 6: Sea inlet DP	0	0	0	0	1000	0	0	0	0	0	0
Comp 7: Sea inlet JP	0	0	0	0	0	600	0	0	0	0	0
Comp 8: Funnel	0	0	0	0	0	0	0	0	0	0	0
Comp 9: Water Pump Skid	0	0	0	0	0	0	0	0	0	0	0
Comp 10: Fuel pump skid	0	0	0	0	0	0	0	0	0	0	0
Comp 11: MDO tank interface	0	0	0	0	0	0	0	0	0	0	0

Table 6.4: Pathing Matrix for scenario 1

Figure 6.2 shows the scatter plot of the feasible solutions found by the tool.

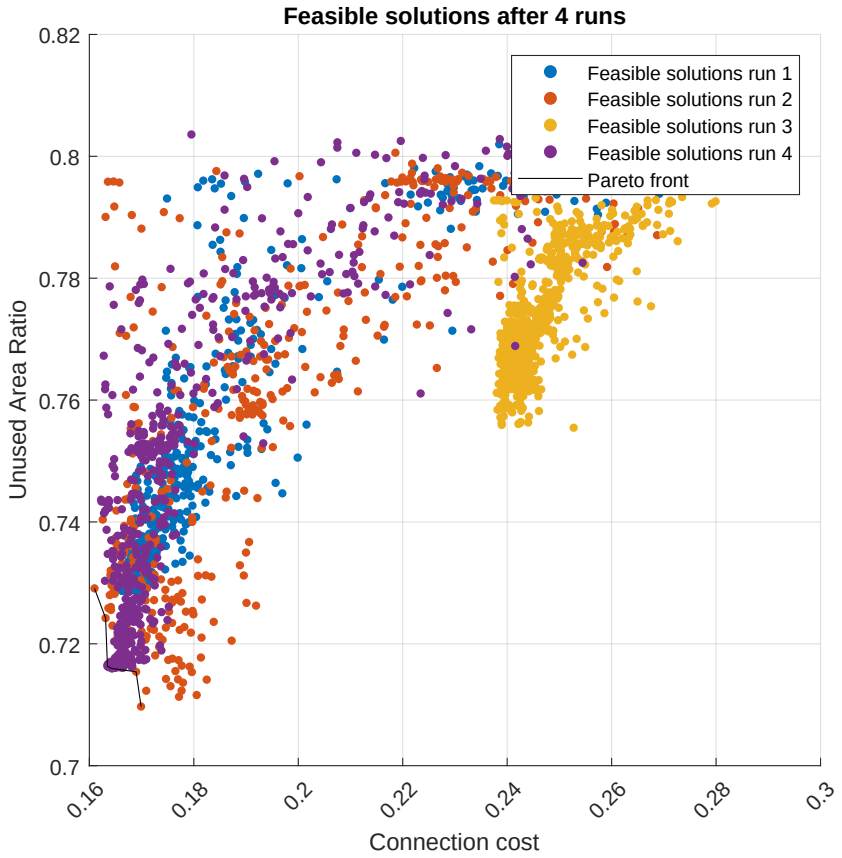
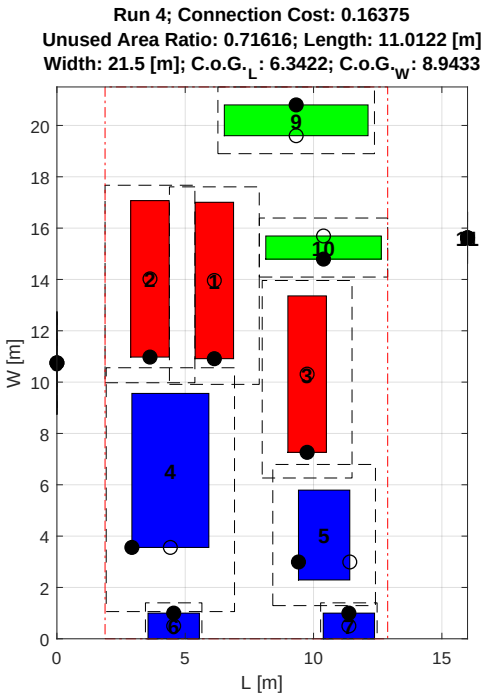


Figure 6.2: Feasible solutions found for the direct drive configuration

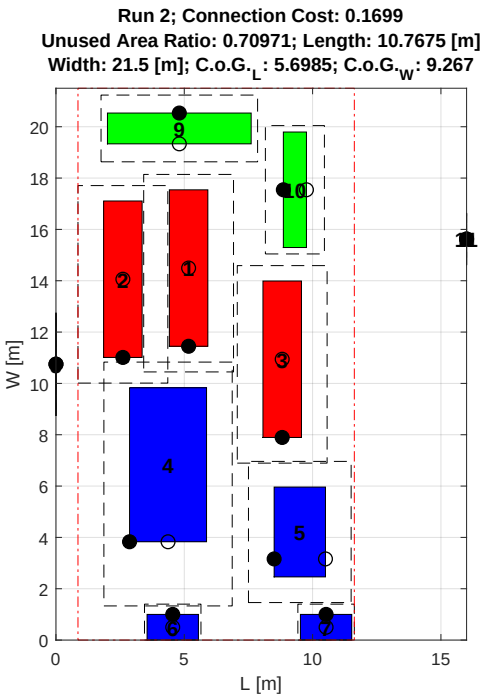
As the input was very constrained the output has little variance, as shown in figure 6.3.

Figure 6.3 shows two arrangements from the Pareto front of the runs. The left arrangement was found by run 4, plotted in purple in the scatter plot. Is has relative balanced objective values. The right arrangement was found by the second run and as it has the lowest Unused Area Ratio found by the model, it has the shortest length for the combined machinery spaces.

While this arrangement is the shortest one that was found, the length of the machinery spaces is defined by the frame spacing of the vessel, which is 700mm. This means that the shortest arrangement and the more balanced one still fit in the same number of frames, namely 16 frames for a combined machinery



(a) A Pareto front solution which balances Unused Area Ratio and Connection Cost



(b) The Pareto front arrangement with the shortest engine room length

Figure 6.3: Example Pareto front arrangements for scenario 1.

space length of 11.2m.

This is considerably shorter than the example design shown in figure 6.1, which has a length of 20 frames (14m).

The reasons that the arrangements generated by the tool are shorter are twofold. The first reason is that the box coolers for the cooling water of the diesel engines are not modelled in the tool. In the Vuyk example scenario these are placed at port-side against the ship hull. These coolers would prevent pump skid number nine being placed close to the port-side shell plating of the vessel. These coolers could be modelled if a further iteration of this scenario would be investigated.

Secondly, the third diesel engine is positioned further starboard than both other generator sets. In the example design this was not possible due to the straight watertight longitudinal bulkhead separating the two machinery spaces. If the designer could fit a step in that longitudinal bulkhead to the starboard side, the most forward engine can be placed further to starboard, as is presented by the tool in the generated arrangements, thus creating extra space on port-side leading to a shortening of the engine room. By using the tool the designer could be attended to the possibility of shortening the engine room.

Because the diesel engines already drive generators, in the next scenario the option of electrical-driven pumps is investigated.

6.2. Scenario 2: Three 3516B generator sets and electric driven pumps

In scenario 2 the dredge and jet pump are fitted with an electric motor instead of being driven diesel-direct by the engines of the generator sets. The effect this has upon the arrangements is investigated.

6.2.1. Input Scenario 2

The dredge pump is sized with an electric motor of the same length as the gearbox that was fitted in the previous scenario. While the electric motor will be smaller than the gearbox, the width of the rectangular area of the component is still determined by the dredge pump itself. This results in a component with the same measurements as in the previous case. For the jet pump the same reasoning is used and thus properties of the components for this case remain the same and can be read from table 6.1.

The constraints for this case do change, as the pumps are no longer connected to the generator sets. This results in the constraints shown in table 6.5. The orientation of the generator sets in now free, as is the orientation of the jet pump. The grouping for this case is significantly different, as the generator sets are no longer grouped with the pumps they were connected to in the previous case. The dredge and jet pump are still grouped with their respective inlets. Also the fuel pump skid is still grouped with its interface.

Classification			Position constraints						Other
Nr.	Component	Type	Xmin [0..1]	Xmax [0..1]	Ymin [0..1]	Ymax [0..1]	Rmin [0..1]	Rmax [0..1]	Group
1	Genset 1	Generator Set	0	1	0	1	0	1	1
2	Genset 2	Generator Set	0	1	0	1	0	1	1
3	Genset 3	Generator Set	0	1	0	1	0	1	1
4	Dredge pump	Pump	0	0.5	0	1	0.4	0.4	2
5	Jetpump	Pump	0	1	0	1	0	1	3
6	Sea inlet DP	Inlet	0	0.3	0	0	0.1	0.1	2
7	Sea inlet JP	Inlet	0.3	1	0	0	0.1	0.1	3
8	Funnel	Interface	0	0	0.5	0.5	0.4	0.4	1
9	Water Pump Skid	Pump skid	0	1	0	1	0	1	1
10	Fuel pump skid	pump skid	0	1	0	1	0	1	4
11	MDO tank interface	Interface	1	1	0.75	0.75	0.4	0.4	4

Table 6.5: Constraints table for scenario 2

The pathing remains the same for this scenario, and is thus defined by table 6.4.

6.2.2. Arrangements Scenario 2

The scoring of the generated arrangements are presented in the scatter plot shown in figure 6.4.

Figure 6.5 shows three distinct arrangement variants from the Pareto front that the tool generated.

The arrangement in figure 6.5a is the Pareto front solution that has the lowest connection cost, but this arrangement requires quite some space: the length of the arrangement is necessitating 19 frames (13.3m). This is caused by the orientation of the generator sets: while it allows the generator sets to be placed as close to the funnel as possible, and the fuel pump skids to be placed relative close to all generator sets, the longitudinal positioning of generator set nr.3 requires a lot of space.

However, as can be seen in the Pareto front arrangement in figure 6.4 a relatively small increase in connection cost can significantly lower the unused area Ratio, resulting in shorter arrangements. Two of these shorter Pareto front solutions are discussed next.

Figure 6.5b presents an arrangement with transverse positioning of the generator sets, which length is 17 frames (11.9m). While compared to the previous solutions with the directly driven pumps, in this scenario the generator sets can be placed closer together as they do not need to be in line with the pumps. The jet pump also can be rotated to have its connection with the sea inlet minimized, which is preferred.

Figure 6.5c presents a solution with longitudinal positioning of the generator sets, which is generally preferred for operability reasons. The dredge and jet pumps line up nicely with their sea inlets. In this arrangement the length needed by these pumps determines the length of the combined machinery

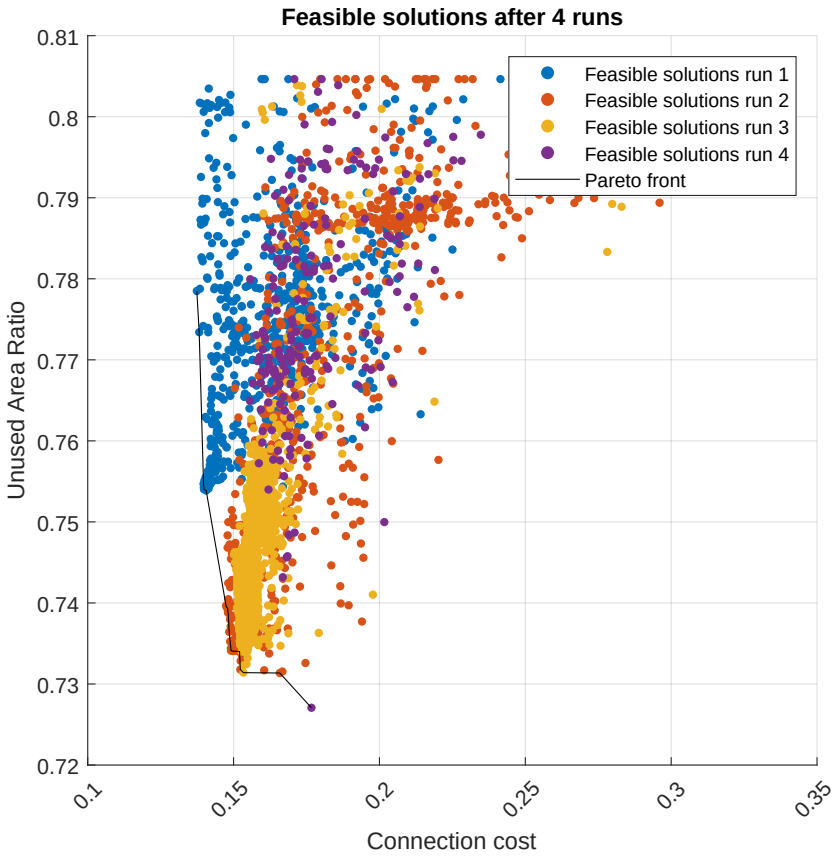
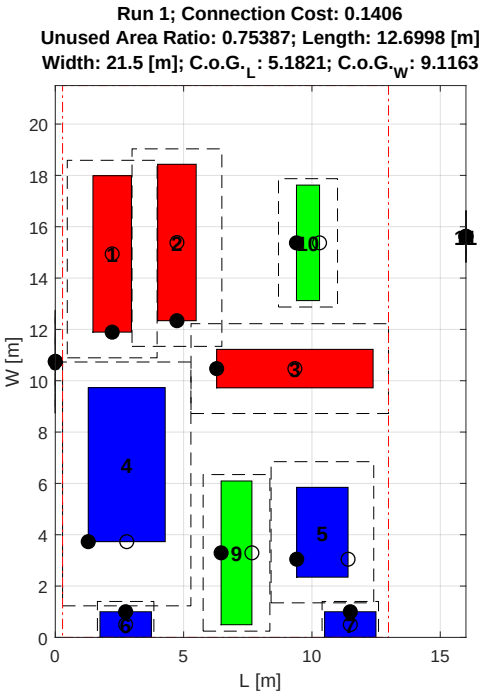
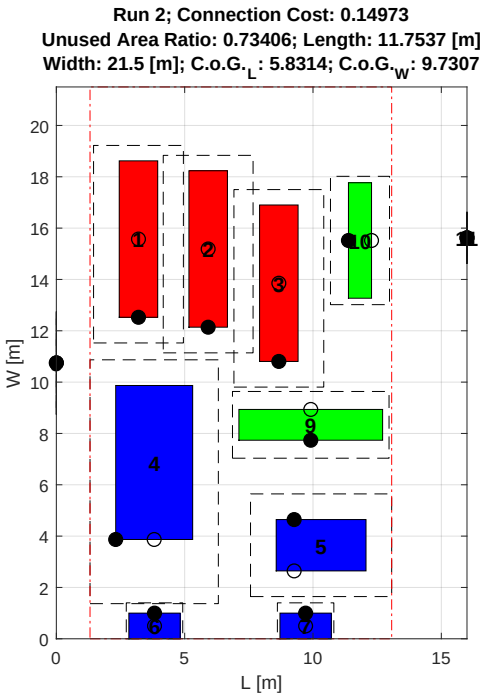


Figure 6.4: Feasible solutions found for the three generator set electric configuration

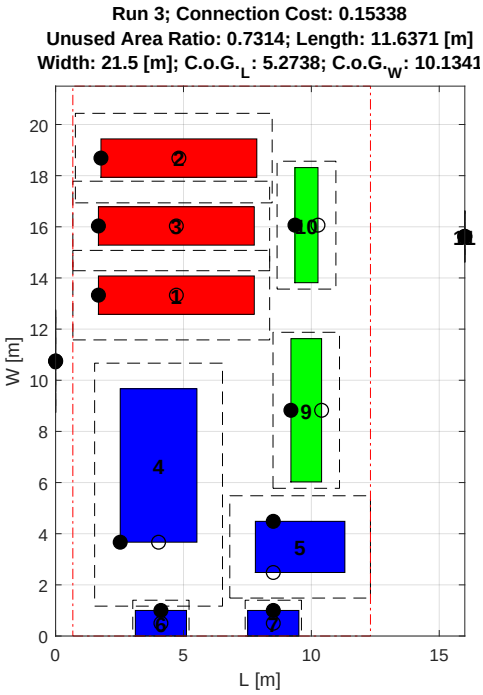
space, which fits within 17 Frames (11.9m). By placing these pumps further aft the arrangement could possibly even be shortened to fit within 16 frames (11.2m), leading to an interesting arrangement



(a) Shortest exhaust line connection



(b) A Pareto front solution which balances Unused Area Ratio and Connection Cost



(c) The Pareto front arrangement with the shortest length

Figure 6.5: Example Pareto front arrangements for scenario 2.

6.3. Scenario 3: Four 3512B generator sets and electric driven pumps

In scenario 3 the generators sets are changed to four 3512B generator sets to provide the power for the vessel.

6.3.1. Input Scenario 3

As the 3512B generators sets are four cylinders shorter than the 3516B ones, they have a shorter length. This is reflected in the input presented in table 6.6. Aside from the different generators, the other components for this scenario remain the same as for the previous scenarios.

Classification			General Physical			Clearances				In and output location				Other
Nr.	Component	Type	Length [mm]	Width [mm]	Weight [kg]	Left [mm]	Right [mm]	Up [mm]	Down [mm]	Xin [0..1]	Yin [0..1]	Xout [0..1]	Yout [0..1]	Plot Color
1	Genset 1	Generator Set	4842	1500	14975	1000	600	1000	1000	0.5	0.5	0	0.5	red
2	Genset 2	Generator Set	4842	1500	14975	1000	600	1000	1000	0.5	0.5	0	0.5	red
3	Genset 3	Generator Set	4842	1500	14975	1000	600	1000	1000	0.5	0.5	0	0.5	red
4	Genset 4	Generator Set	4842	1500	14975	1000	600	1000	1000	0.5	0.5	0	0.5	red
4	Dredge pump	Pump	6000	3000	51000	2500	1000	1000	1000	0	0.5	0	1	blue
5	Jetpump	Pump	3500	2000	16000	1000	1000	1000	1000	0.2	0	0.2	1	blue
6	Sea inlet DP	Inlet	2000	1000	3000	100	100	400	0	0.5	0.5	0.5	1	blue
7	Sea inlet JP	Inlet	2000	1000	3000	100	100	400	0	0.5	0.5	0.5	1	blue
8	Funnel	Interface	4000	1	1	0	0	0	0	0.5	0.5	0.5	0.5	white
9	Water Pump Skid	Pump skid	5600	1200	2000	250	250	700	700	0.5	0	0.5	1	green
10	Fuel pump skid	pump skid	4500	900	2000	250	250	700	700	0.5	0	0.5	1	green
11	MDO tank interface	Interface	2000	1	1	0	0	0	0	0.5	0.5	0.5	0.5	white

Table 6.6: Properties of the modelled components of scenario 3

The 3512B generator sets are modelled with the same constraints and relations as the 3516B generator sets in scenario 2, and thus the constraints-, relationship matrix and pathing table remain the same as for scenario 2 aside from the additional generator set.

6.3.2. Arrangements Scenario 3

The feasible solutions found for this scenario are presented in the scatter plot figure 6.6.

Figure 6.7 shows two arrangements from the Pareto front for this run. Notably, there are no solutions where all the generator sets are placed either longitudinally or vertically, their orientation is always mixed.

In the arrangement shown in figure 6.7a the generator sets are placed quite compact and relatively close to the centreline , which leaves more space for the box coolers with are not yet modelled. Observing the arrangement it is noted that the length could be shortened by placing the generator sets and pump skids further aft. The arrangement now takes 19 frames (13.3m), but the designer could easily shorten this to 18 frames (12.6m), inspired by this generated arrangement.

If shortening the engine room length would be beneficial for other spaces or aspects in the general design of the vessel, the arrangement in figure 6.7b is interesting. This arrangement fits in 17 frames (11.9m). The water pump skid could be placed further aft or transversely between the generator set 1 and 2 and the fuel pump skid to make space for the box coolers which are not modelled. The arrangement shows that 17 frames is the minimal length required for an arrangement with four 3512B generator sets.

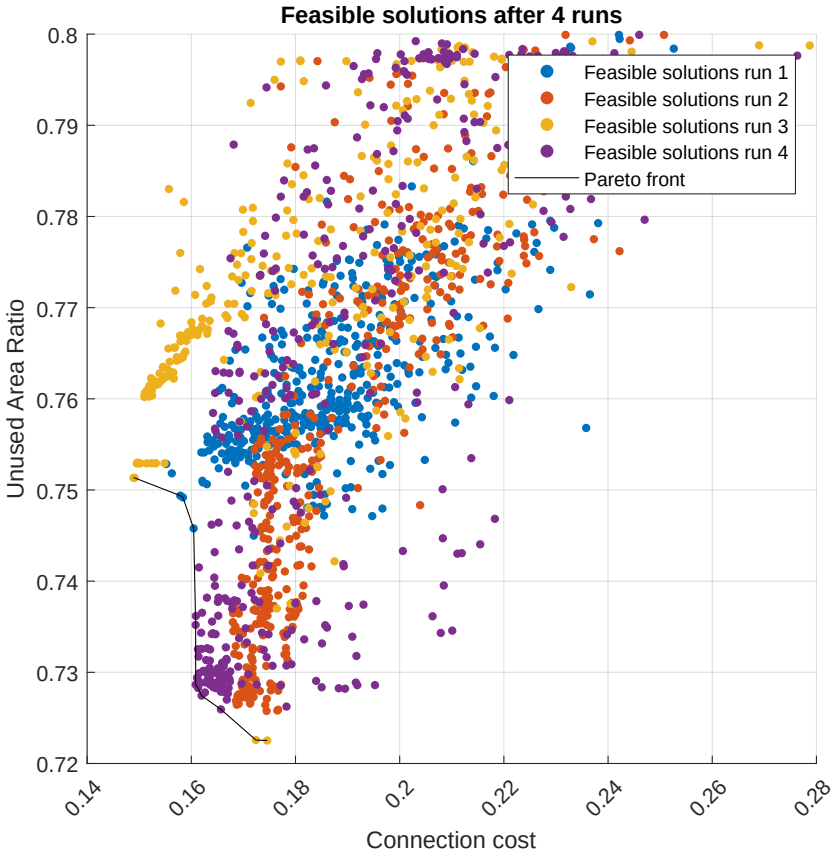
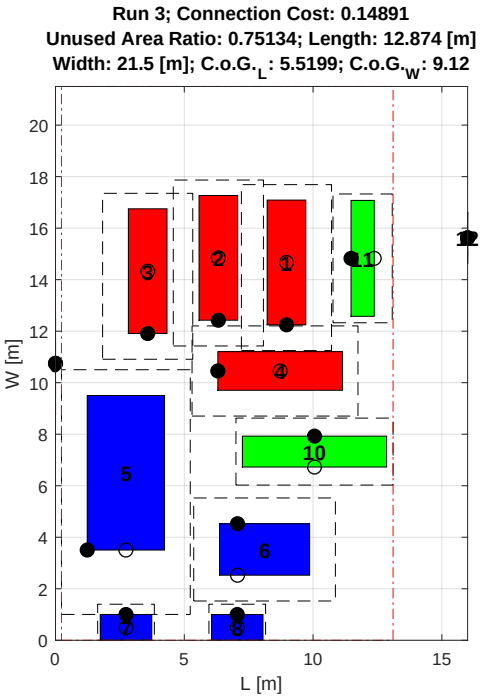
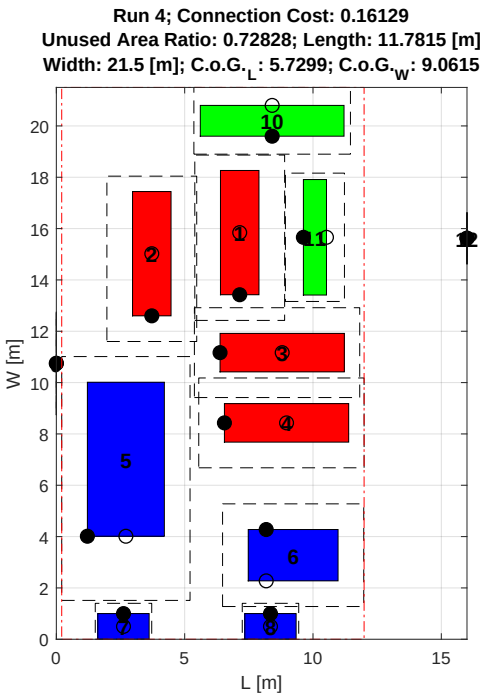


Figure 6.6: Feasible solutions found for scenario 3



(a) The Pareto front arrangement with the lowest Connection Cost



(b) A Pareto front solution which balances Unused Area Ratio and Connection Cost

Figure 6.7: Example Pareto front arrangements for scenario 3

6.4. Scenario 4: six 3508B generator sets and electric driven pumps

This final scenario uses six 3508B generator sets to provide the power for the vessel.

6.4.1. Input Scenario 4

With only eight cylinders the 3508B generator sets are the shortest that are investigated, which can be seen in table 6.7.

The 3508B generator sets are modelled with the same constraints and relations as the generator sets in scenario 2 and scenario 3, and thus the constraints-, relationship matrix and pathing table remain the same as for these scenarios aside from the added generator sets.

Classification			General Physical			Clearances				In and output location				Other
Nr.	Component	Type	Length [mm]	Width [mm]	Weight [kg]	Left [mm]	Right [mm]	Up [mm]	Down [mm]	Xin [0..1]	Yin [0..1]	Xout [0..1]	Yout [0..1]	Plot Color
1	Genset 1	Generator Set	4000	1500	12360	1000	600	1000	1000	0.5	0.5	0	0.5	red
2	Genset 2	Generator Set	4000	1500	12360	1000	600	1000	1000	0.5	0.5	0	0.5	red
3	Genset 3	Generator Set	4000	1500	12360	1000	600	1000	1000	0.5	0.5	0	0.5	red
4	Genset 4	Generator Set	4000	1500	12360	1000	600	1000	1000	0.5	0.5	0	0.5	red
5	Genset 5	Generator Set	4000	1500	12360	1000	600	1000	1000	0.5	0.5	0	0.5	red
6	Genset 6	Generator Set	4000	1500	12360	1000	600	1000	1000	0.5	0.5	0	0.5	red
7	Dredge pump	Pump	6000	3000	51000	2500	1000	1000	1000	0	0.5	0	1	blue
8	Jetpump	Pump	3500	2000	16000	1000	1000	1000	1000	0.2	0	0.2	1	blue
9	Sea inlet DP	Inlet	2000	1000	3000	100	100	400	0	0.5	0.5	0.5	1	blue
10	Sea inlet JP	Inlet	2000	1000	3000	100	100	400	0	0.5	0.5	0.5	1	blue
11	Funnel	Interface	4000	1	1	0	0	0	0	0.5	0.5	0.5	0.5	white
12	Water Pump Skid	Pump skid	5600	1200	2000	250	250	700	700	0.5	0	0.5	1	green
13	Fuel pump skid	pump skid	4500	900	2000	250	250	700	700	0.5	0	0.5	1	green
14	MDO tank interface	Interface	2000	1	1	0	0	0	0	0.5	0.5	0.5	0.5	white

Table 6.7: Properties of the modelled components of the six generator set electric configuration

6.4.2. Arrangements Scenario 4

The feasible solutions found for this scenario are presented in figure 6.8. Due to the larger number of defined components the scatter is spread more broadly, but the solutions ‘hit a clear wall’ with regards to minimizing the connection cost.

Figure 6.9 shows two Pareto front arrangement that were generated by the tool. The right hand arrangement is the shortest arrangement, which corresponds to the lowest Pareto front solution in the scatter plot. It just fits within 18 frames (12.6m). The left hand arrangement is the Pareto front solution directly above this one, and requires and extra frame (19 frames, 13.3m).

With six generator sets the engine room area becomes quite crowded, not leaving any space for the box coolers which are not modelled. Should this scenario be pursued in a future design iteration, then the cooling solution would require the attention of the designer.

While both arrangements are quite unrealistic, they also present a clear design direction that is interesting to investigate for the generator set arrangement. Figure 6.9a clearly shows that two of the 3508B generator sets fit transversely above the dredge pump. This gives direction for a design where the six generator sets are arranged in two rows of three transversely placed generator sets at the port side of the vessel. The other interesting option is an arrangement based on figure 6.9a, where two generator sets are positioned transversely against the aft bulkhead, and the other four generator sets are positioned longitudinally directly forward of these transverse generator sets.

To investigate the two rows of three transversely placed engines a series of runs with additional position and grouping constraints was performed. In order to produce the desired arrangement the constraints

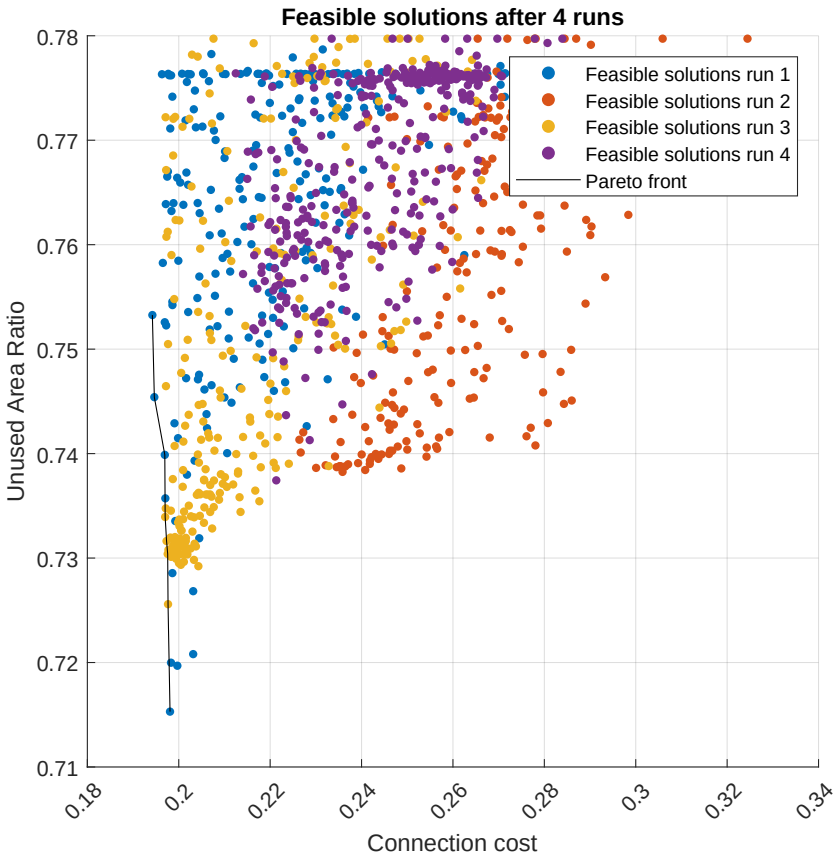
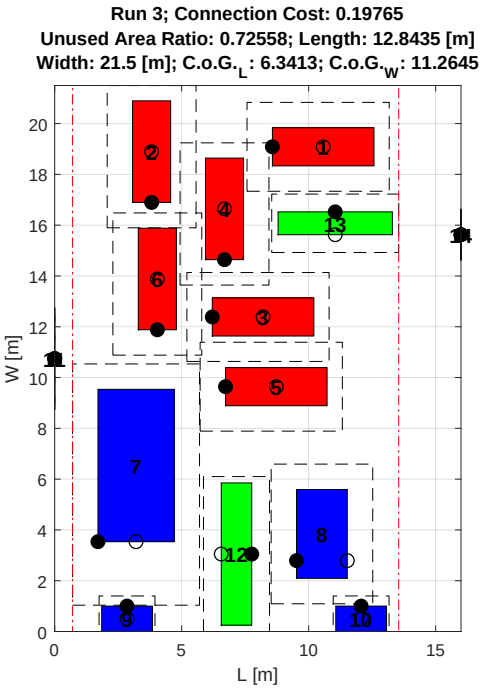
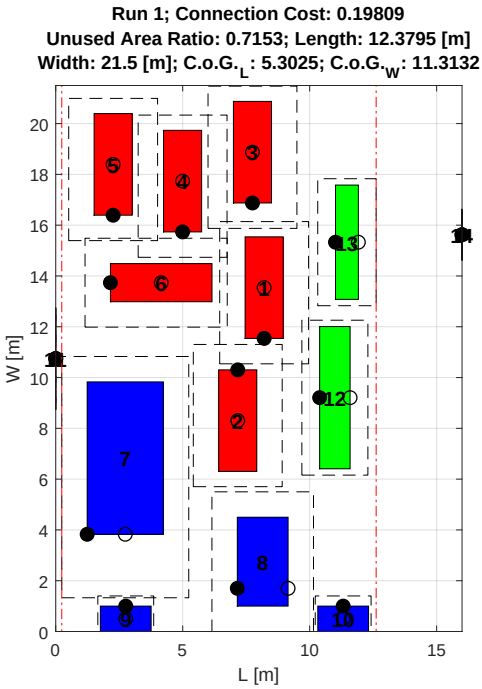


Figure 6.8: Feasible solutions found for scenario 4.



(a) The Pareto front arrangement with the second shortest length



(b) The Pareto front arrangement with the shortest length

Figure 6.9: Example Pareto front arrangements for scenario 4.

had to be defined so tightly that essentially no other arrangement was possible. These constraints are shown in table 6.8.

Classification			Position constraints						Other
Nr.	Component	Type	Xmin [0..1]	Xmax [0..1]	Ymin [0..1]	Ymax [0..1]	Rmin [0..1]	Rmax [0..1]	Group
1	Genset 1	Generator Set	0	0.8	1	1	0.4	0.4	5
2	Genset 2	Generator Set	0	0.8	1	1	0.4	0.4	5
3	Genset 3	Generator Set	0	0.8	1	1	0.4	0.4	5
4	Genset 4	Generator Set	0	0.6	0.3	0.8	0.4	0.4	5
5	Genset 5	Generator Set	0	0.6	0.3	0.8	0.4	0.4	5
6	Genset 6	Generator Set	0	0.6	0.3	0.8	0.4	0.4	5
7	Dredge pump	Pump	0	0.5	0	1	0.4	0.4	2
8	Jetpump	Pump	0.3	1	0	1	0	1	3
9	Sea inlet DP	Inlet	0	0.3	0	0	0.1	0.1	2
10	Sea inlet JP	Inlet	0.3	1	0	0	0.1	0.1	3
11	Funnel	Interface	0	0	0.5	0.5	0.4	0.4	1
12	Water Pump Skid	Pump skid	0.4	1	0	1	0.4	0.4	1
13	Fuel pump skid	pump skid	0.4	1	0	1	0.4	0.4	4
14	MDO tank interface	Interface	1	1	0.75	0.75	0.4	0.4	4

Table 6.8: Constraints to create the 3x2 generator set arrangement shown in figure 6.10.

This results in the interesting arrangement shown in figure 6.10. This arrangement has a significantly lower Unused Area Ratio, and with a length of 11.8097m just fits within 17 frames (11.9m). As only the constraints have changed compared to the previous runs in this scenario, the connection cost of this arrangements can be compared to those in figure 6.9. This arrangement has better connection cost, which was initially quite difficult for the tool to minimize. If this objective has a large priority, the designer could consider to rotate the lower row of generator sets 180 degrees and position the funnel further towards the port side of the vessel, to minimize the exhaust line distance. This would require the designer to reposition the funnel in the above deck arrangements in the general arrangement of the vessel.

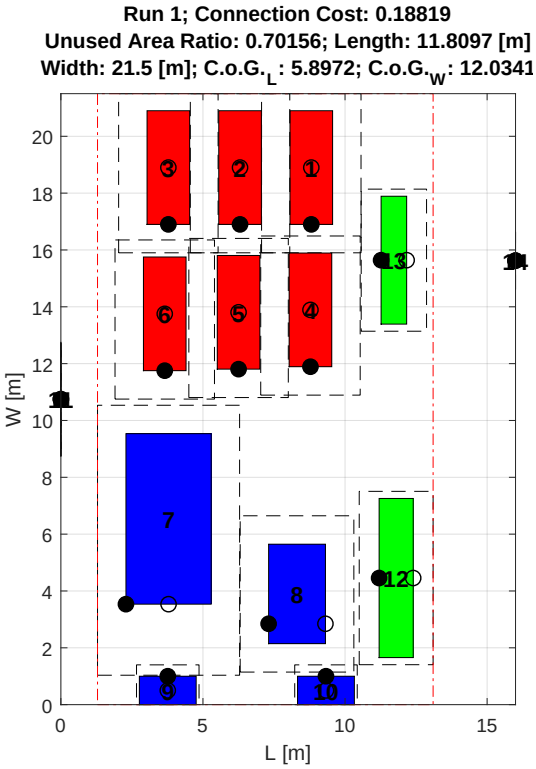


Figure 6.10: Example Pareto front arrangements for scenario 4.

The process of iteratively adapting the constraints to create a specific arrangement inspired by the initial use of the tool, highlights an interesting improvement.

As the tool generates arrangements, even if they are unrealistic, they can bring about new arrangement ideas for the designer, who may like to slightly modify an interesting generated arrangement to make it more realistic. If an additional feature of the tool would enable the designer to quickly, manually create the desired modified arrangement and plot its connection cost value and unused area ratio in the scatter plot, the modified arrangement can be judged in relation to the solutions the tool generated itself. This could be a very useful addition to this tool for the designer, supporting him in analysing and evaluating various realistic design scenarios in an early design phase.

6.5. Conclusions on the Application of the Tool

The tool was used to investigate 4 scenarios for the machinery space arrangement of a TSHD vessel in an early concept design stage. The use of the tool allowed the designer to, in limited time, evaluate a diesel-direct driven dredge- and jet pump arrangement and three arrangements with electrical-driven dredge and jet pumps with a varying amount of generator sets.

The scenario analysis leads to various insights for the arrangement options for the machinery spaces. Depending on the context of the overall design process, analysing and evaluating the options could lead to a choice for one configuration option. Alternatively, the analysis can also support the choice to keep the options open and postpone the decision for a configuration to next iteration in a later phase of the design: the scenario analysis determines the minimum dimensions for the machinery space to enable that all relevant scenarios can fit in the assigned area. If during a future design iteration of the general arrangement of the complete vessel, adjacent spaces would need to be enlarged and the length of the considered machinery spaces would need to be shortened, the scenario analysis with the tool in the early stage of the design showed already that it can be possible to shorten the machinery spaces to 17 frames (11.9m), if a specific generator set configuration and arrangement are applied.

The tool has shown to be useful for a designer early in the concept design stage where it is important to not limit the design freedom, by enabling the designer to generate, analyse and evaluate a large number of options for machinery space arrangements in a limited period of time.

During the testing of the tool in the design process of the trailing suction hopper dredger, it has shown to have difficulties to generate logically ordered arrangements when the number of components increases without the addition of more constraints. However, these ‘chaotic’ arrangements may still provide (new) insight and stimulus for the designer to think ‘out of the box’ to develop unusual but good arrangements for the machinery spaces.

Designing such an unusual new arrangement based on an interesting ‘chaotic’ one as generated by the tool, can be realized by adding constraints in the tool, but this is often quite cumbersome. A useful addition to the tool for the designer would be to allow for a quick, manual alteration of a generated arrangement, in order to check the feasibility of the new arrangement and how its scores on the two objectives relative to the arrangements generated by the tool.

7

Conclusions and Recommendations

In this final chapter conclusions are drawn from the research and recommendations are presented for further research.

7.1. Conclusions

The research started in chapter 2 with an investigation into the design process at Vuyk, how the machinery spaces fit in to this design process and what Vuyk considers good machinery space design. The chapter continued with a literature study on machinery space design and concept exploration. The chapter concluded with a set of design requirements that are important for a tool for early design machinery space arrangement.

The research started in chapter 2 with an investigation into the design process at Vuyk, concluding how the design of the arrangement of machinery spaces fits in to thegeneral design process and what Vuyk considers a good machinery space design. Based on this investigation and an additional literature study on machinery space design and on methods of 'concept exploration' in design processes, a set of design requirements is concluded that are important for a tool for early design machinery space arrangement. Accurately estimating the space needed for the arrangement of the components in the machinery spaces early in the design stages is one of the main objectives.

This should be ensured by exploring the solution space of technically feasible results and presenting the arranged layouts in a visual way, indicating for each generated arrangement the reference values for the Unused Area Ratio and for the Connection Cost.

This research concluded also that other main design drivers for the arrangement of a ship machinery space are: the placement of accesses, escapes, transport routes and walkways, taking into account sufficient space for maintenance around the machinery and proper engine room ventilation.

Investigations in chapter 3 on the best way to model the machinery arrangement problem, led to the method for developing solutions according the Facility Layout Problem. This method was developed to generate facility layouts for manufacturing applications that best satisfy predefined goals, and was considered to be most promising to deal with the arrangement of specialized components in ship machinery spaces.

The method complies to the most relevant identified design requirements for the tool (section 2.4), such as the handling of relationships between the components, incorporating multiple objectives and handling multiple floors. It also has predefined ways to deal with several of the identified design drivers, such as the space for maintenance.

After investigating various resolution approaches for the Facility Layout Problem the decision was made to use a Particle Swarm Optimization algorithm.

During the model testing in chapter 5 several shortcomings of the defined model were discovered. It appears that the formulation in which the overlap constraint is enforced with a penalty function, imposes severe difficulties for the particle swarm algorithm, as the penalty function was unable to be properly tuned. The balance where the particles can explore the solution space broadly while also being able to distinguish between feasible and infeasible solutions was not found. It is concluded that this particular constrained enforcement method in unsuitable for optimally solving the Facility Layout Problem with a Particle Swarm Optimization algorithm.

Investigation and development of several mutations for adapting the algorithm, resulted in the generation of more diverse arrangement solutions by the tool:

- By implementing a mutation inspired by a local search heuristic, to better allow the algorithm to optimize the generated arrangements, the model could be improved, and also by implementing a mutation which is akin to a repair algorithm, to allow for more diverse exploration.; both these mutations are considered promising for further development.
- With the addition of the grouping constraint the model can be forced to generate solutions which are less chaotic, but this comes at a cost of a loss of solution diversity; this is a useful addition if the designer already has good ideas of the kind of layouts that are interesting to explore and it adds value to the iterative nature of the design process.
- The developed pathing filter enables the model to better account for the identified design drivers;it succeeds in filtering out technically infeasible solutions that were not identified before.
- When the model is able to escape the local minima that are an issue due to the penalty function, the designer input of the relationship matrix that guides the connection cost objective performs well in generating arrangements with a logical layout.

The application of the tool in chapter 6 shows the usefulness of the tool to the designer.

A large number of options for machinery space arrangements could be generated, analysed and evaluated in a limited period of time, providing the designer with the desired design freedom early in the concept design stage of a trailing suction hopper dredger. The application of the tool leads to various insights for the arrangement options for the machinery spaces.

Depending on the context of the overall design process, analysing and evaluating these options could lead to a choice for one configuration option.

Alternatively, it is concluded that the analysis can also support the choice to keep the options open and postpone the decision for a configuration to next iteration in a later phase of the design: the scenario analysis determines the minimum dimensions for the machinery space for all relevant scenarios. These arrangement scenario's can also be considered in a later stage of the design, supporting the designer to find new configurations if for example the machinery space needs to be shortened during a future design iteration of the general arrangement of the complete vessel.

This research project was guided by the formulated research questions, leading to the above mentioned conclusions. These show that the research objective has been reached as it was stated in section 1.2:

> *Automatically generate arrangements of machinery spaces in an early design phase and quantify their performance in the overall design perspective of the vessel, in order to support the designer in selecting a feasible arrangement for a machinery space in this early design phase.*

7.2. Recommendations for Further Research

The investigation in chapter 2 showed that only a limited amount of literature on machinery space design exists. This design is mainly informed by the experience of the designers, and more research to capture this experience is recommended.

In order to improve the performance of the design tool, it is recommended to implement the enforcement of the overlap constraints with a repair function, instead of a penalty function. This would allow the particle swarm optimization algorithm to explore the solution space of feasible solutions without getting trapped in a local minimum, improving the solution diversity and thus the concept exploration capabilities of the model.

The developed pathing filter has currently been applied after the tool has generated all the arrangements. By integrating this filter within the algorithm iterations, it could be used to steer the concept exploration away from infeasible regions which are currently not recognized by the algorithm.

A useful addition to the tool for the designer would be to allow for a quick, (limited) manual alteration of the position of components in a generated arrangement, in order to check the feasibility of the new arrangement and how its scores on the two objectives relative to the arrangements generated by the tool.

Various design requirements were identified which were not incorporated into the model due to the scope of the research. The usefulness of the tool can be extended by developing the model to a Multiple-Floor Facility Layout Problem, to account for multiple decks in the connected machinery spaces that are often designed in vessels. Likewise, being able to handle hull shapes would allow for a more wide application of the tool.

Developing a database of reference components usable for the input of the tool could make the tool more accessible and quick to use for a designer.

The selection of machinery is currently left to the user of the tool. Assisting the user in this step makes the tool more accessible to less experienced designers. de Vos (2018) developed an Automatic Topology Generation tool which could be used to aid the designer in choosing the input used for the tool for machinery space arrangement.

Bibliography

A. Ahmadi, M. S. Pishvaee, and M. R. A. Jokar. A survey on multi-floor facility layout problems. *Computers & Industrial Engineering*, 107:158–170, may 2017. doi: 10.1016/j.cie.2017.03.015.

D. Andrews. Marine requirements elucidation and the nature of preliminary ship design. 2011.

J. Babicz. *Wartsila Encyclopedia of Ship Technology*. Biuro Okretowe, 2015. ISBN 978-952-93-5535-8. URL https://www.amazon.com/Wartsila-Encyclopedia-Ship-Technology-Babicz/dp/9529355351.

A. Brownlee and J. R. Woodward. Why we fell out of love with algorithms inspired by nature. The Conversation, June 2015. URL https://theconversation.com/why-we-fell-out-of-love-with-algorithms-inspired-by-nature-42718.

P. de Vos. *On early-stage design of vital distribution systems on board ships*. PhD thesis, Delft University of Technology, 2018.

A. Drira, H. Pierreval, and S. Hajri-Gabouj. Facility layout problems: A survey. *Annual Reviews in Control*, 31(2):255–267, jan 2007. doi: 10.1016/j.arcontrol.2007.04.001.

E. Duchateau. *Interactive Evolutionary Concept Exploration in Preliminary Ship Design*. PhD thesis, Delft University of Technology, 2016.

H. Dyckhoff. A typology of cutting and packing problems. *European Journal of Operational Research*, 44(2):145–159, jan 1990. doi: 10.1016/0377-2217(90)90350-k.

G. W. Evans†, M. R. Wilhelm†, and W. Kariwiwski. A layout design heuristic employing the theory of fuzzy sets. *International Journal of Production Research*, 25(10):1431–1450, oct 1987. doi: 10.1080/00207548708919924.

G. M. Fadel and M. M. Wiecek. Packing optimization of free-form objects in engineering design. In *Optimized Packings with Applications*, pages 37–66. Springer International Publishing, 2015. doi: 10.1007/978-3-319-18899-7_3.

J. W. Gillespie. *A Network Science Approach to Understanding and Generating Ship Arrangements in Early-Stage Design*. PhD thesis, University of Michigan, 2012. URL http://hdl.handle.net/2027.42/96126.

S. Helwig. *Particle Swarms for Constrained Optimization*. doctoralthesis, Friedrich-Alexander-Universität Erlangen-Nürnberg (FAU), 2010.

S. M. K. Heris. Particle swarm optimization (pso) in matlab — video tutorial. http://yarpiz.com/440/ytea101-particle-swarm-optimization-pso-in-matlab-video-tutorial, May 2016.

J. H. Holland. Adaptation in natural and artificial systems. an introductory analysis with application to biology, control, and artificial intelligence. *Ann Arbor, MI: University of Michigan Press*, pages 439–444, 1975.

Y. Hu. Dimension prediction of marine system equipment based on first principles. Master's thesis, Delft University of Technology, Jan. 2016.

J. Kennedy and R. Eberhart. Particle swarm optimization. In *Proceedings of ICNN'95 - International Conference on Neural Networks*. IEEE, 1995. doi: 10.1109/icnn.1995.488968.

S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi. Optimization by simulated annealing. *Science*, 220 (4598):671–680, may 1983. doi: 10.1126/science.220.4598.671.

H. Klein Woud and D. Stapersma. *Design of Propulsion and Electric Power Generation Systems*. Institute of Marine Engineers, 2002. ISBN 1-902536-47-9.

K.-Y. Lee, M.-I. Roh, and H.-S. Jeong. An improved genetic algorithm for multi-floor facility layout problems having inner structure walls and passages. *Computers & Operations Research*, 32(4): 879–899, apr 2005. doi: 10.1016/j.cor.2003.09.004.

Q. Liu. A rectilinear pipe routing algorithm: Manhattan visibility graph. *International Journal of Computer Integrated Manufacturing*, pages 1–10, apr 2015. doi: 10.1080/0951192x.2015.1033019.

V. Martínez-Cagigal. Multi-objective particle swarm optimization (mopso). https://nl.mathworks.com/matlabcentral/fileexchange/62074-multi-objective-particle-swarm-optimization-mopso, Dec. 2017. version 1.3.0.0.

D. N. Mavris and D. A. DeLaurentis. Methodology for examining the simultaneous impact of requirements, vehicle characteristics, and technologies on military aircraft design, 2000.

S. Moran. *Process Plant Layout*. Butterworth-Heinemann, 2017. ISBN 978-0-12-803355-5.

Pagmo Development Team. Particle swarm optimization (pso). https://esa.github.io/pagmo2/docs/cpp/algorithms/pso.html, 2017. Date Accessed: 20-08-2018.

Psychonaut. Manhattan distance. https://commons.wikimedia.org/wiki/File:Manhattan_distance.svg, Apr. 2006. Date Accessed: 26-09-2018.

S. S. Rao. *Engineering optimization: theory and practice*. John Wiley & Sons, 2009.

D. Stapersma and P. d. Vos. Dimension prediction models of ship system components based on first principles. In *Proceedings 12th International Marine Design Conference 2015*, pages 391–405, Tokyo, Japan, 2015. JSNAOE.

E. Tupper. *Introduction to Naval Architecture, Fifth Edition*. Butterworth-Heinemann, fifth edition edition, 2013. ISBN 978-0-08-098237-3.

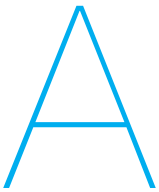
K. van den Berg. Vuyk engineering rotterdam general presentation, Jan. 2018.

B. J. Van Oers. *A Packing Approach for the Early Stage Design of Service Vessels*. PhD thesis, Delft University of Technology, 2011.

B.-C. Wu, G.-S. Young, W. Schmidt, and K. Choppella. Applying fuzzy functions and sequential coordination to optimization of machinery arrangement and pipe routing. *Naval Engineers Journal*, 110 (6):43–54, nov 1998. doi: 10.1111/j.1559-3584.1998.tb02964.x.

Y. Wu and E. Appleton. The optimisation of block layout and aisle structure by a genetic algorithm. *Computers & Industrial Engineering*, 41(4):371–387, feb 2002. doi: 10.1016/s0360-8352(01)00063-8.

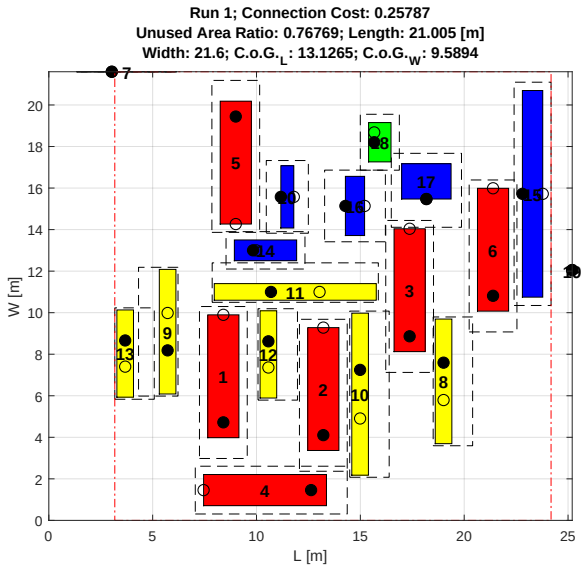
Yarpiz. Facility layout design and location allocation in matlab. http://yarpiz.com/378/ypap109-layout-design, Sept. 2015. Date Accessed: 15-05-2018.



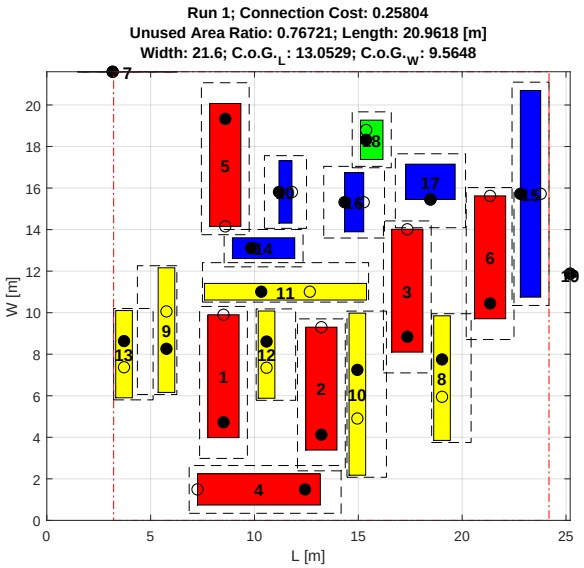
Appendix: Results of the Model Testing and Adaptations

A.1. Setup

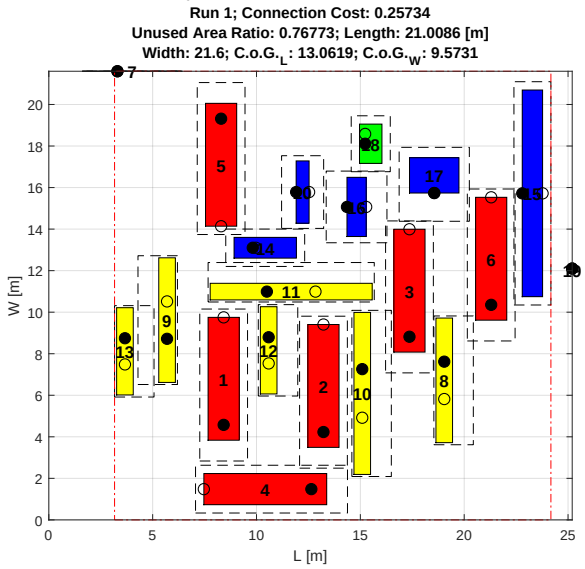
A.1.1. Arrangements of The Initial Model Runs



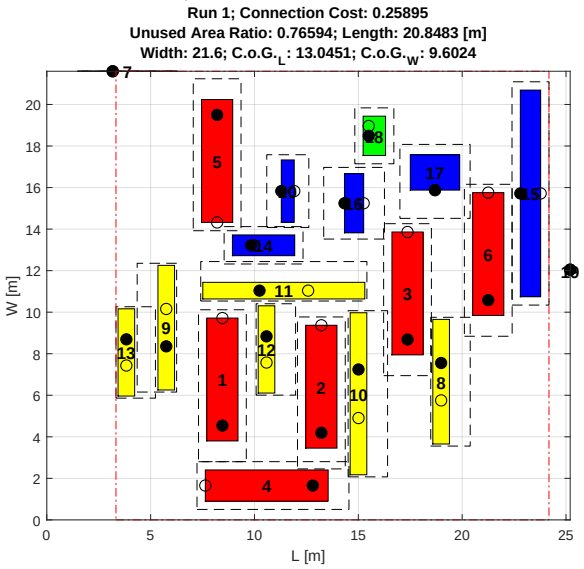
(a) Pareto front layout 1



(b) Pareto front layout 2

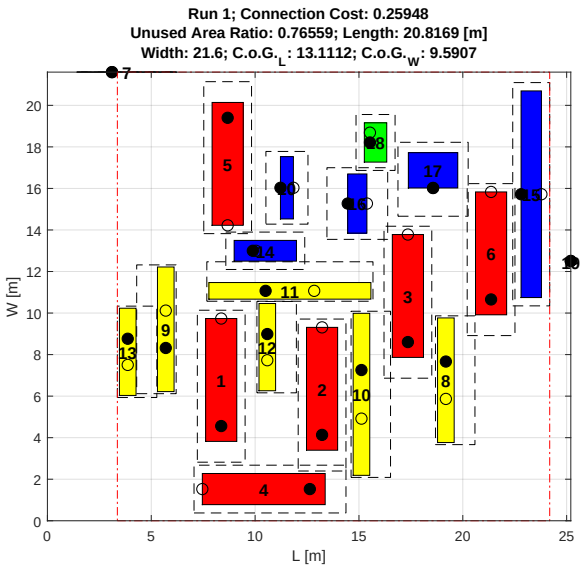


(c) Pareto front layout 3

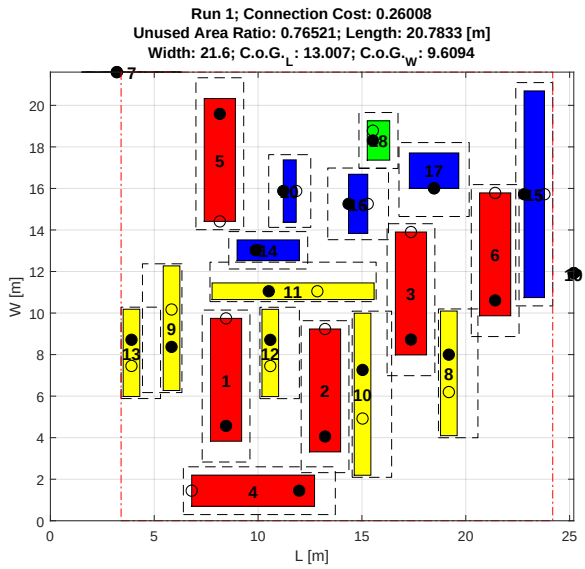


(d) Pareto front layout 4

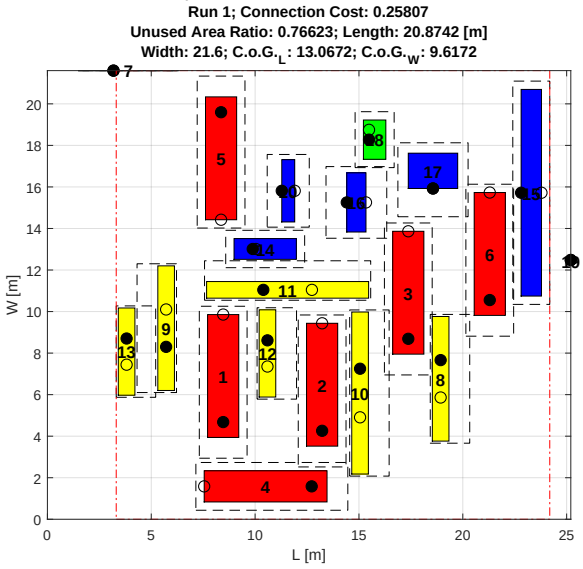
Figure A.1: Layout 1 to 4 of the Pareto front of figure 5.2



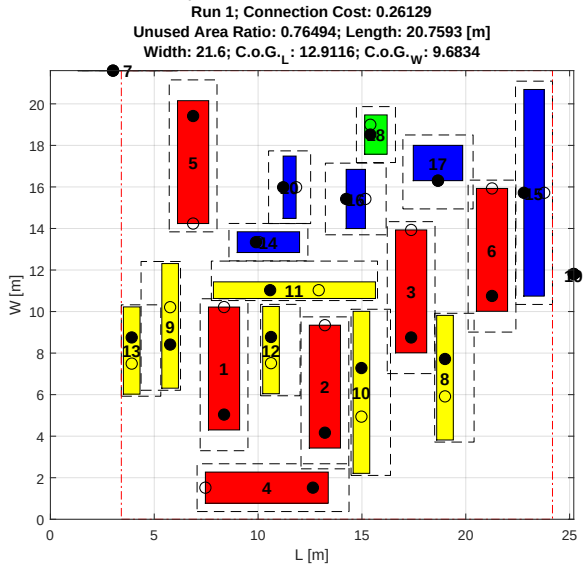
(a) Pareto front layout 5



(b) Pareto front layout 6

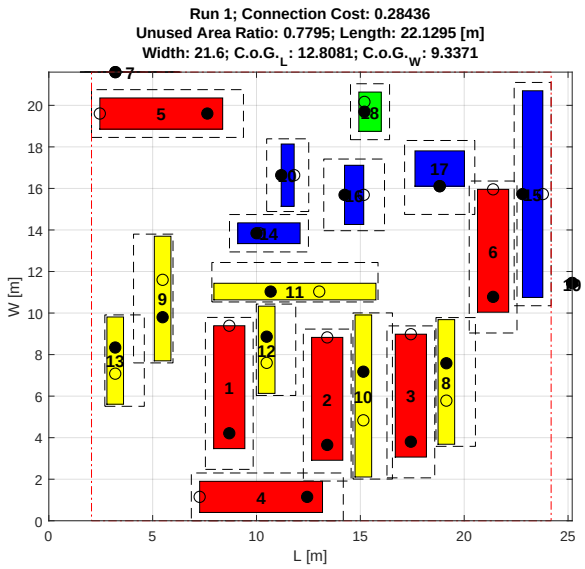


(c) Pareto front layout 7

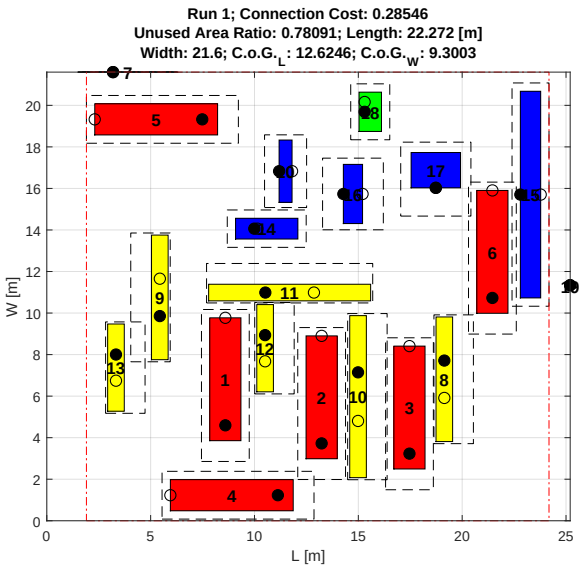


(d) Pareto front layout 8

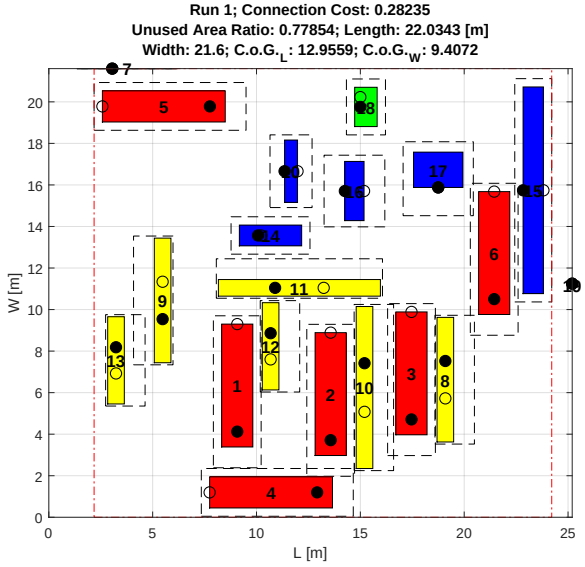
Figure A.2: Layout 5 to 8 of the Pareto front of figure 5.2



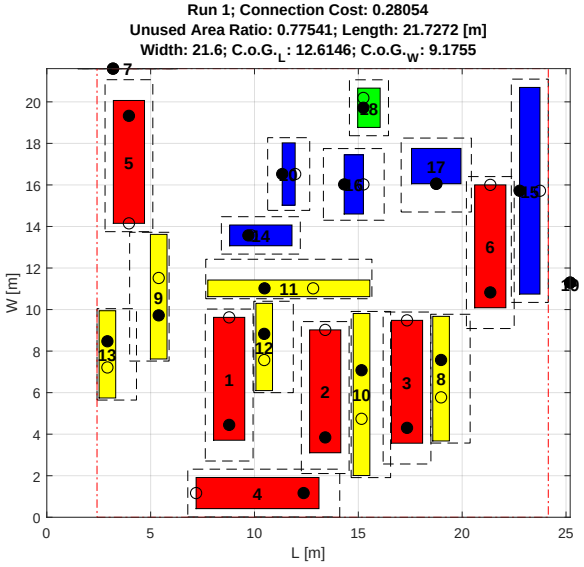
(a) First found solution



(b) Second found solution



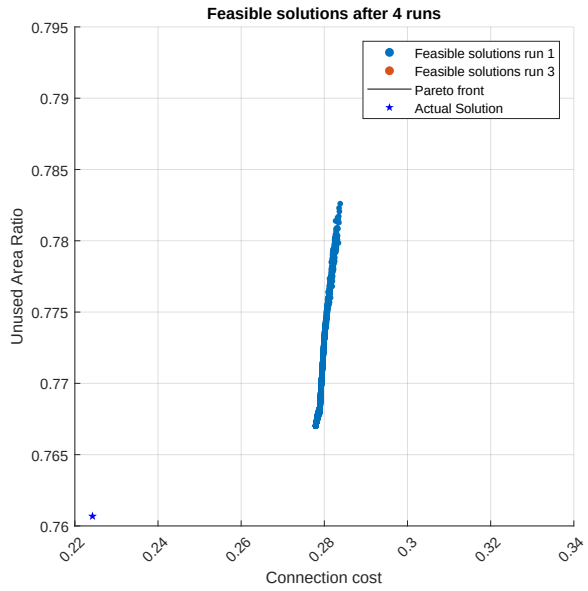
(c) Third found solution



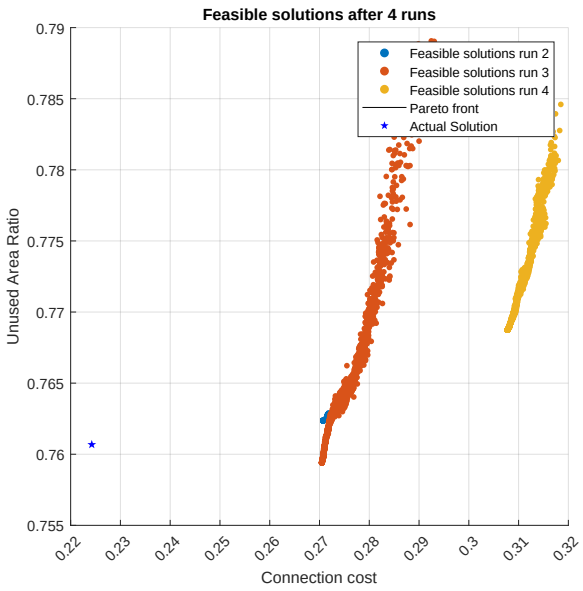
(d) Fourth found solution

Figure A.3: Initial found solutions of the run correspondng to the Pareto front in figure 5.2

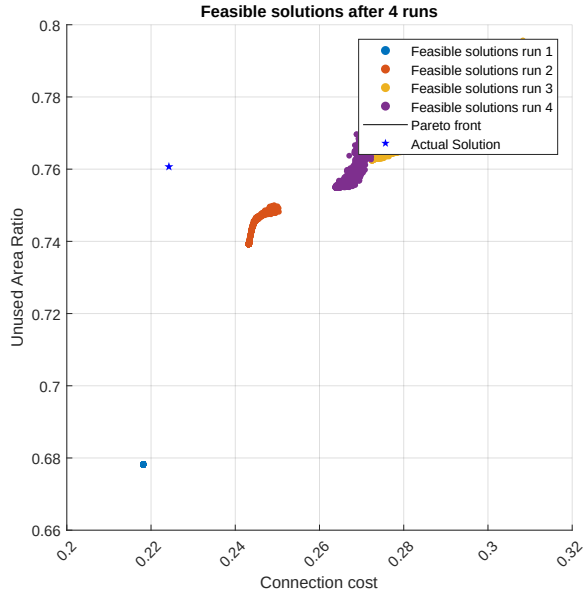
A.2. Investigating the Effect of the PSO Parameters



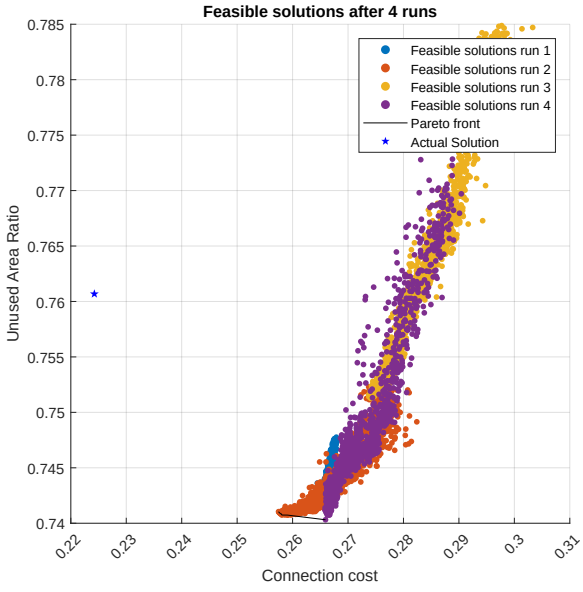
(a) Run $w = 0.45$



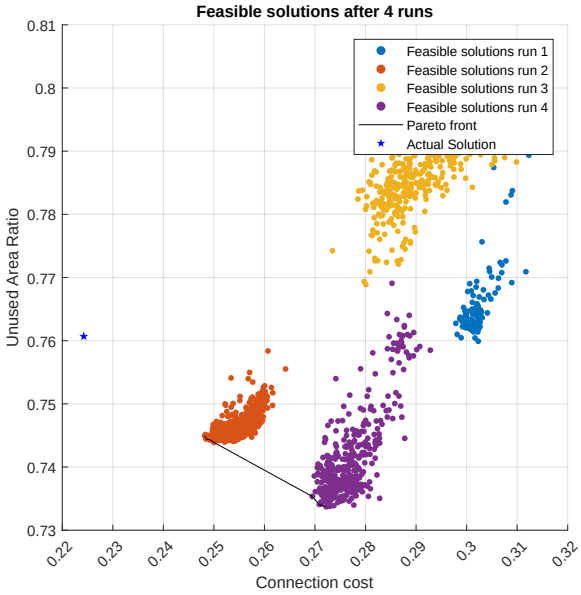
(b) Run $w = 0.50$



(c) Run $w = 0.55$

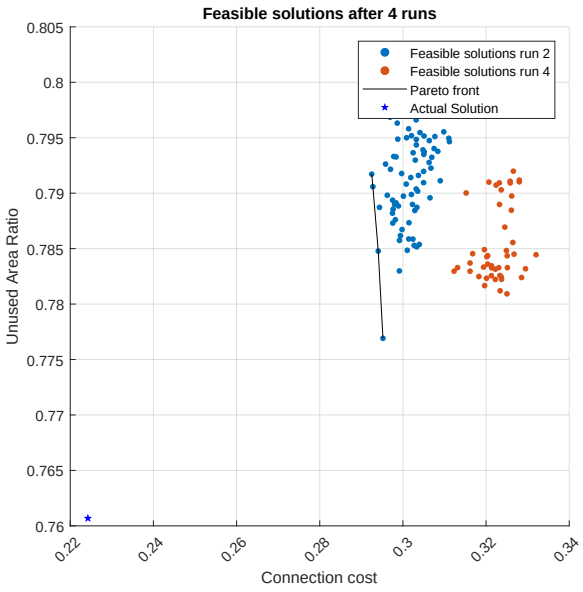


(d) Run $w = 0.60$

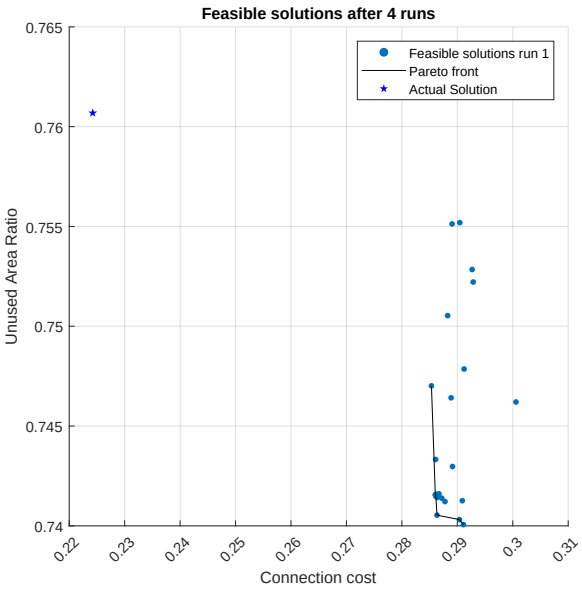


(e) Run $w = 0.65$

Figure A.4: Runs with varying w, and $c_1(w)$ and $c_2(w)$

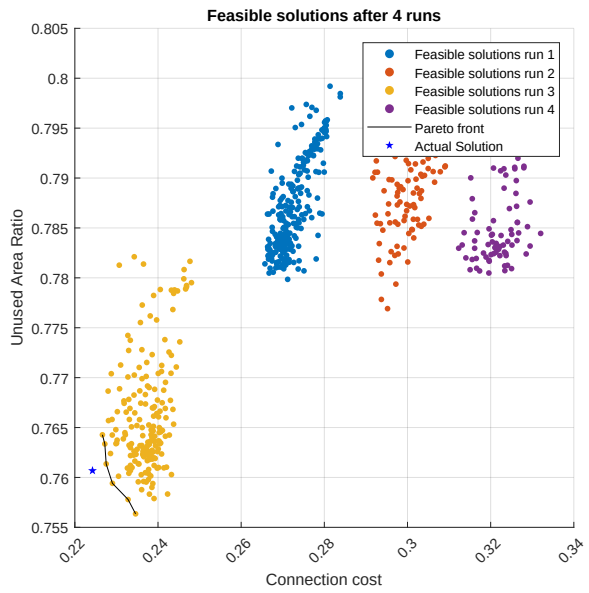


(a) Run $w = 0.70$

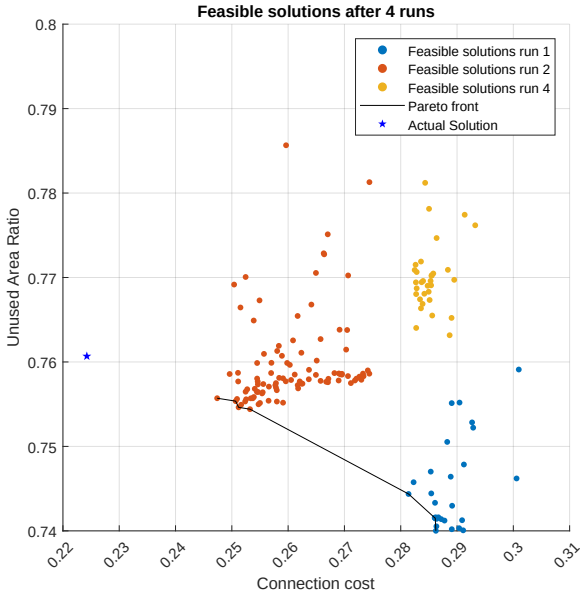


(b) Run $w = 0.75$

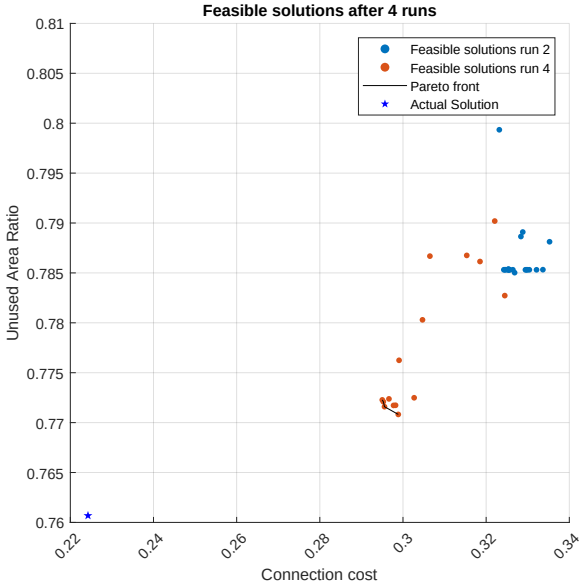
Figure A.5: Runs with varying w, and $c_1(w)$ and $c_2(w)$



(a) Run $w = 0.70$

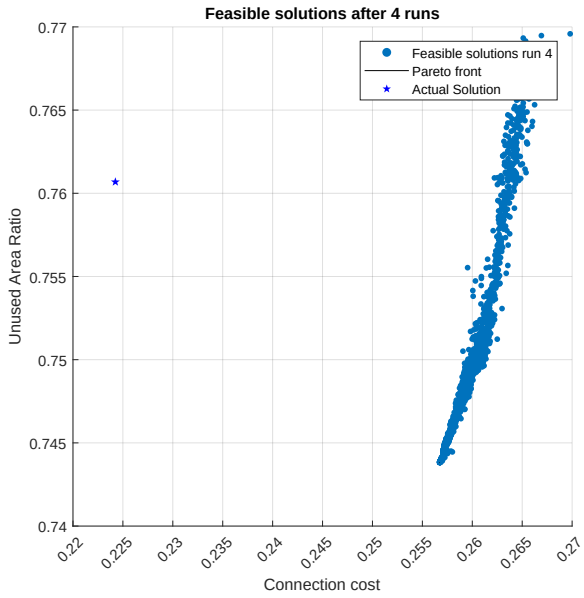


(b) Run $w = 0.75$

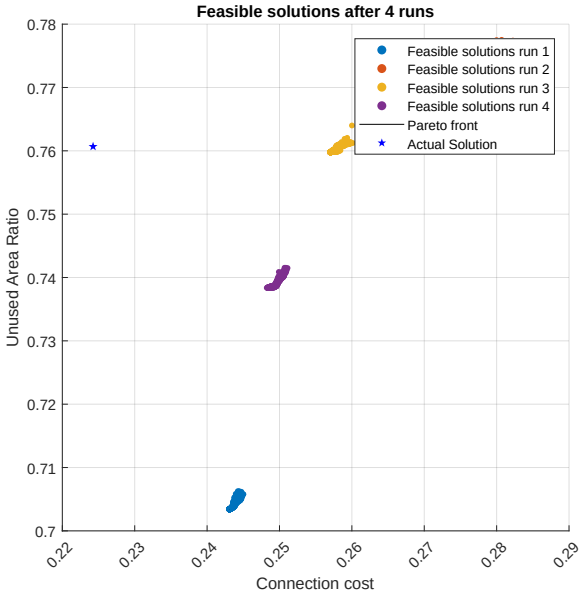


(c) Run $w = 0.80$

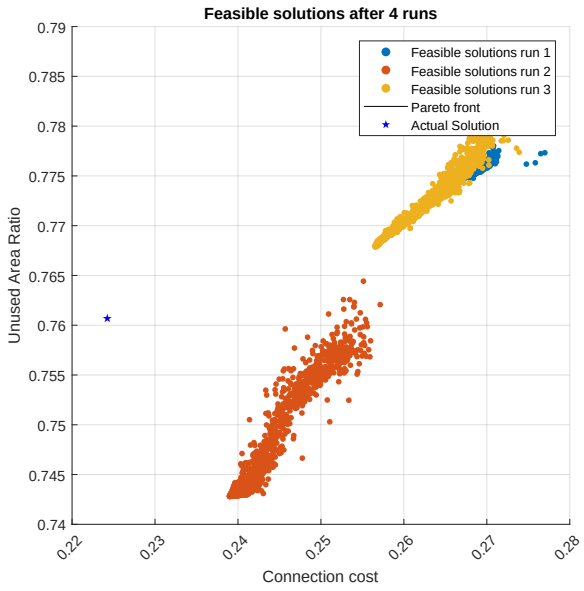
Figure A.6: Runs with varying w, and $c_1(w)$ and $c_2(w)$ with 500 iterations



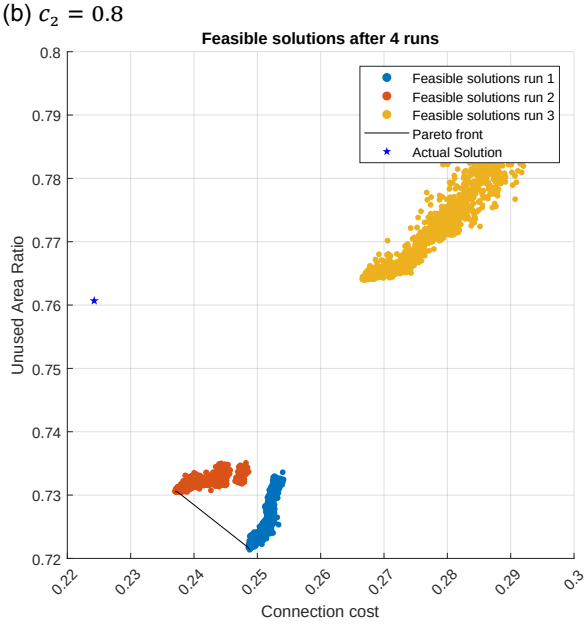
(a) $c_2 = 0.6$



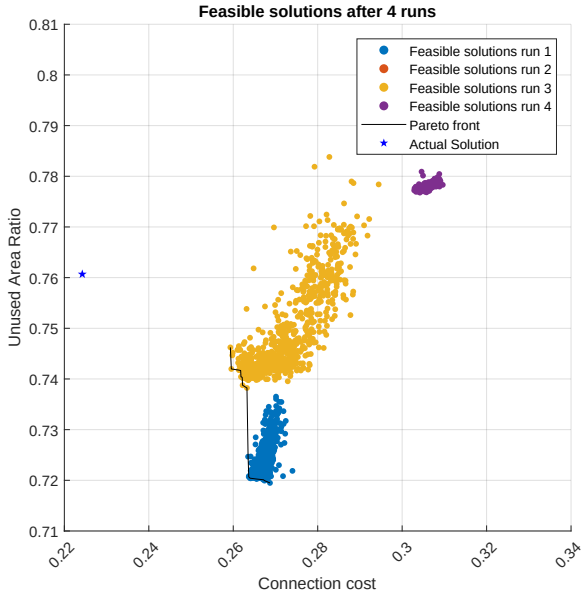
(b) $c_2 = 0.8$



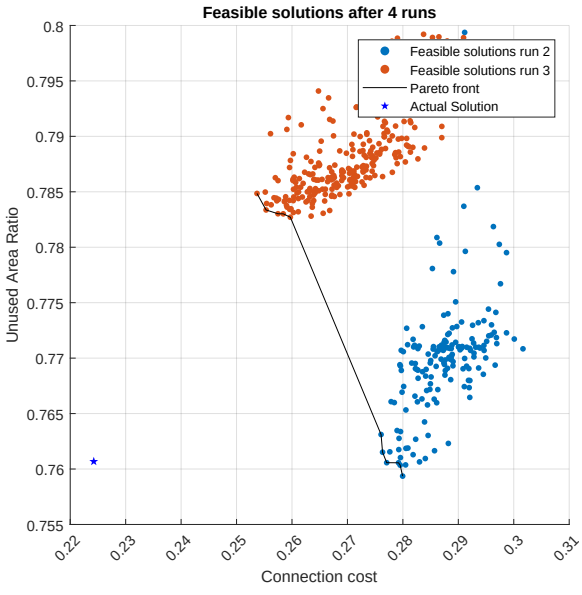
(c) $c_2 = 1.0$



(d) $c_2 = 1.2$

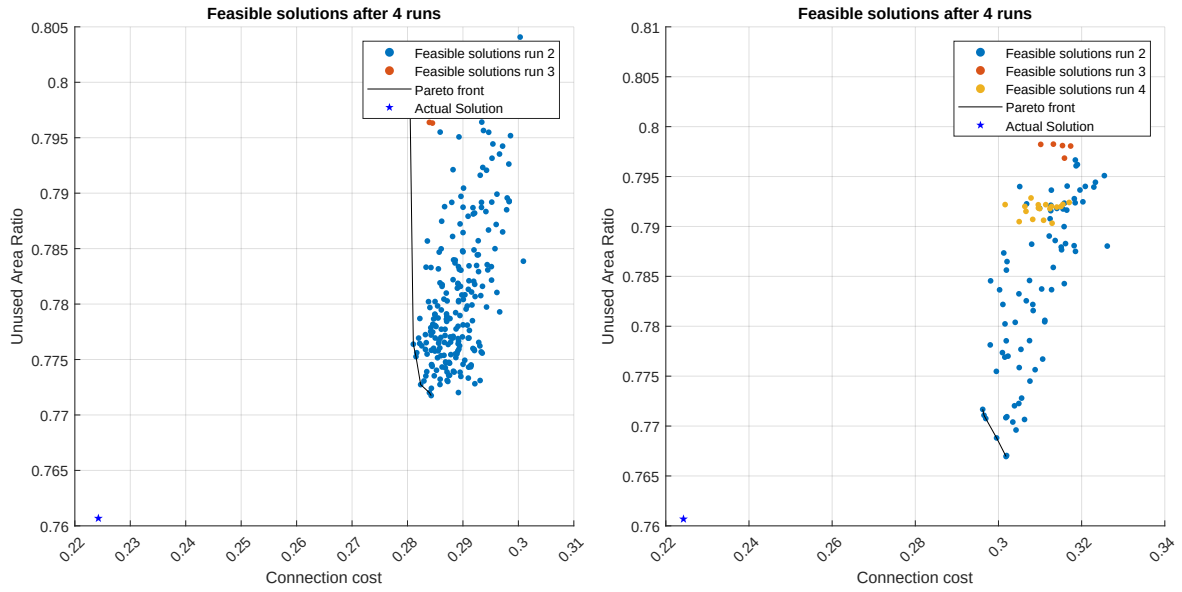


(e) $c_2 = 1.4$



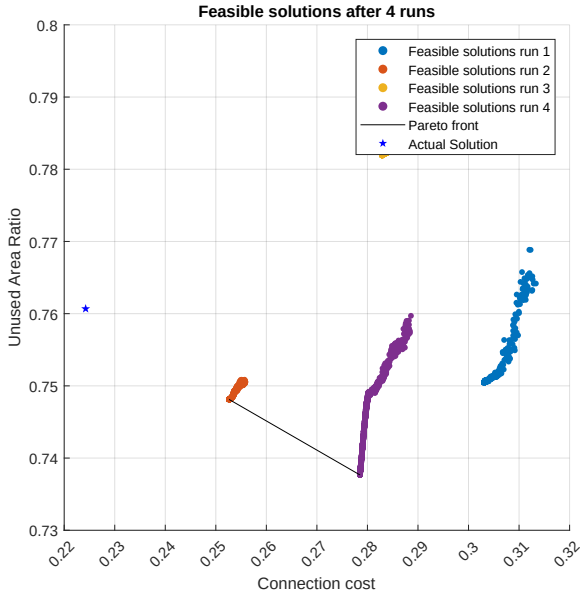
(f) $c_2 = 1.6$

Figure A.7: Runs with constant w, and $c_1(w)$, and varying c_2

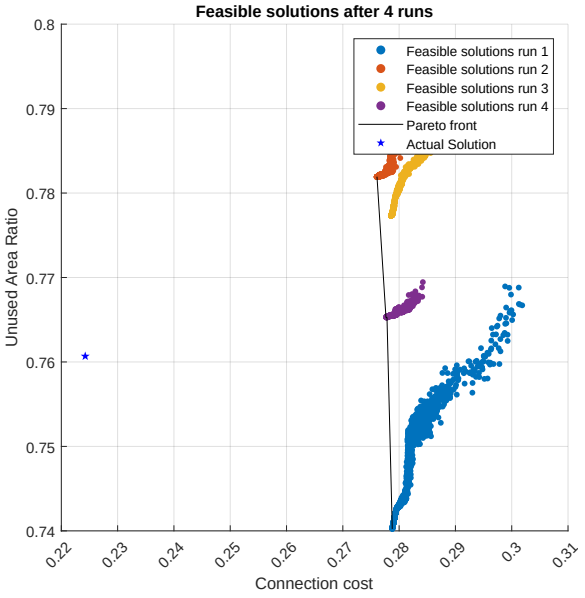


(a) $c_2 = 1.8$ (b) $c_2 = 2.0$

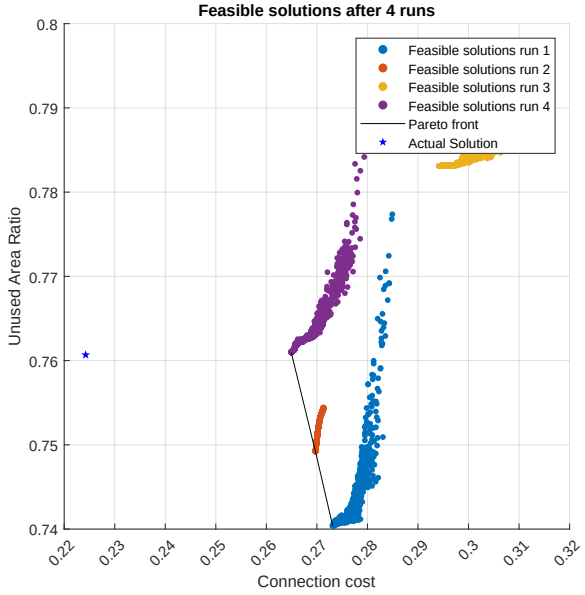
Figure A.8: Runs with constant w, and $c_1(w)$, and varying c_2



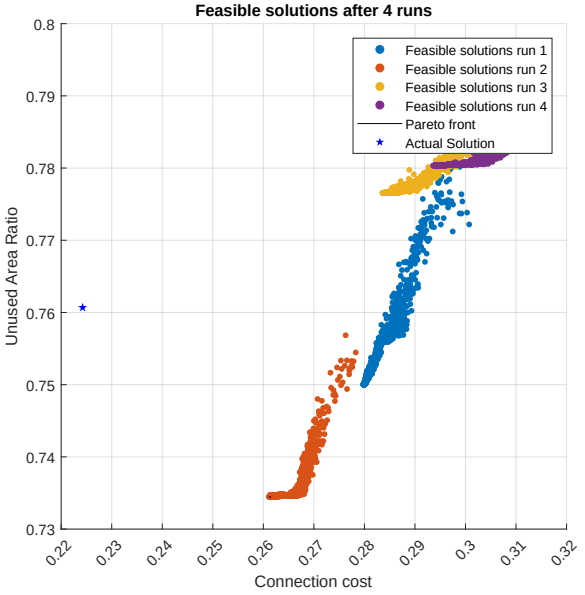
(a) $c_1 = 0.0$



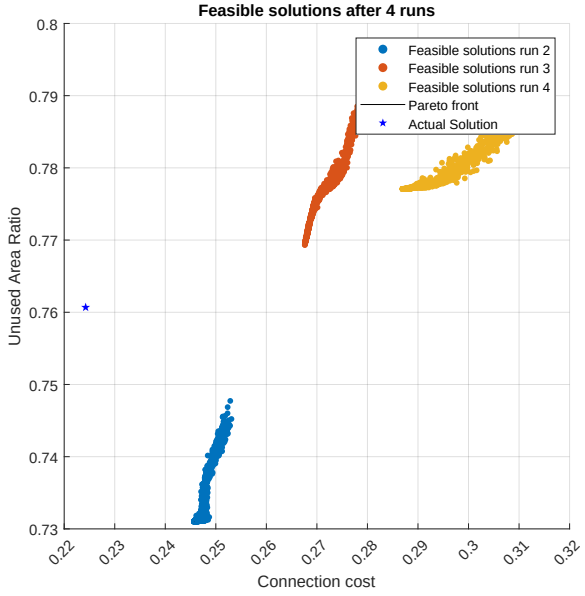
(b) $c_2 = 0.2$



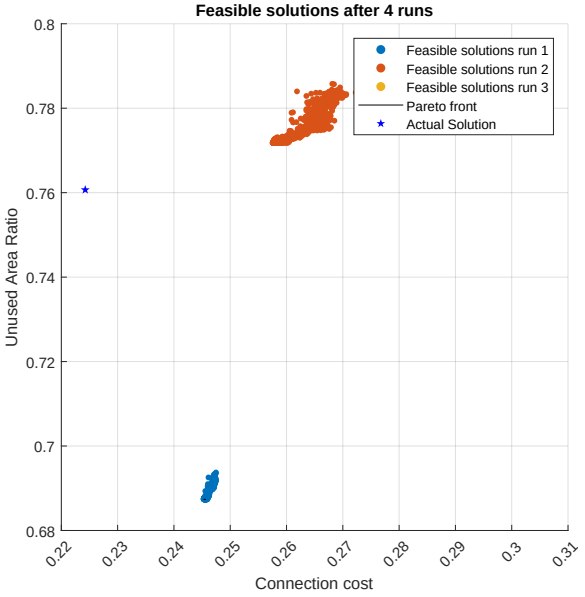
(c) $c_1 = 0.4$



(d) $c_2 = 0.6$

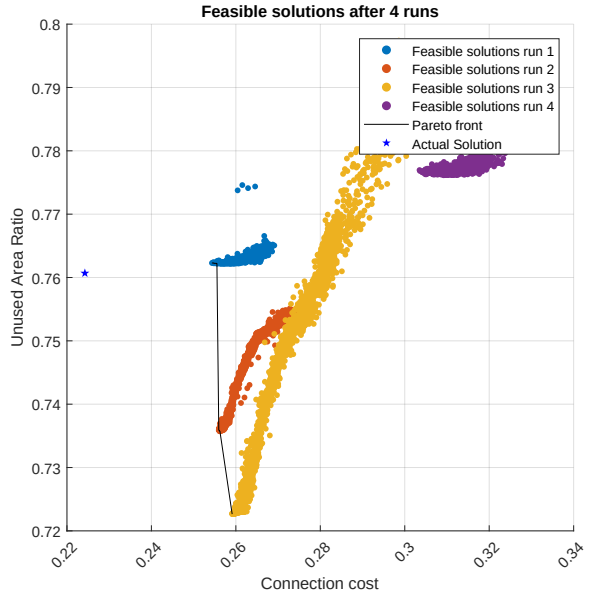


(e) $c_1 = 0.8$

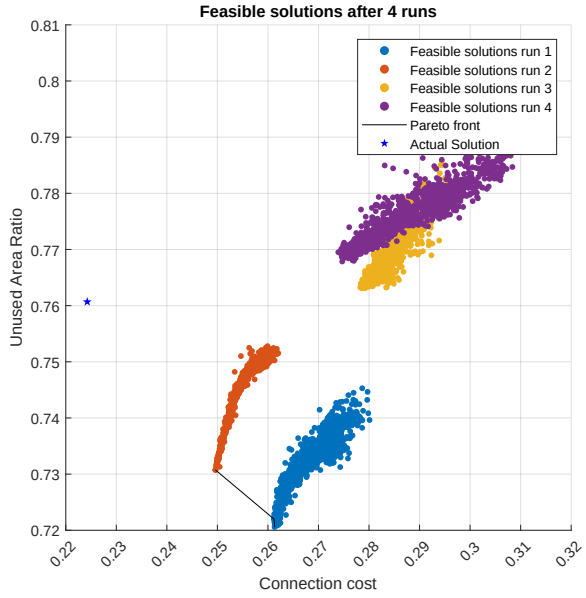


(f) $c_2 = 1.0$

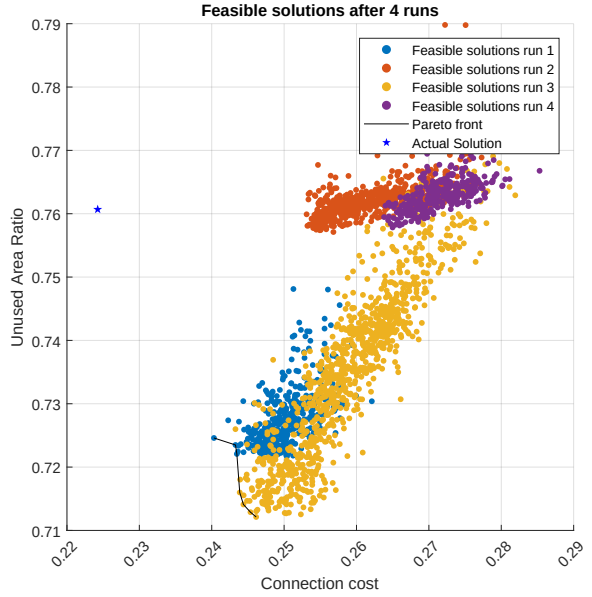
Figure A.9: Runs with constant w, and $c_2(w)$, and varying c_1



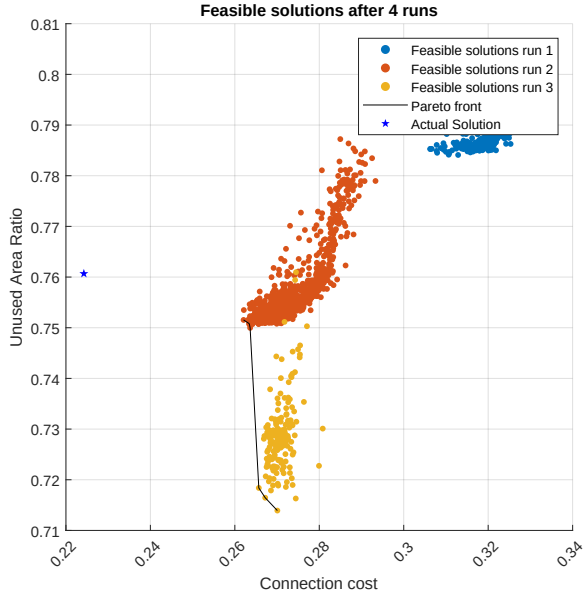
(a) $c_1 = 1.2$



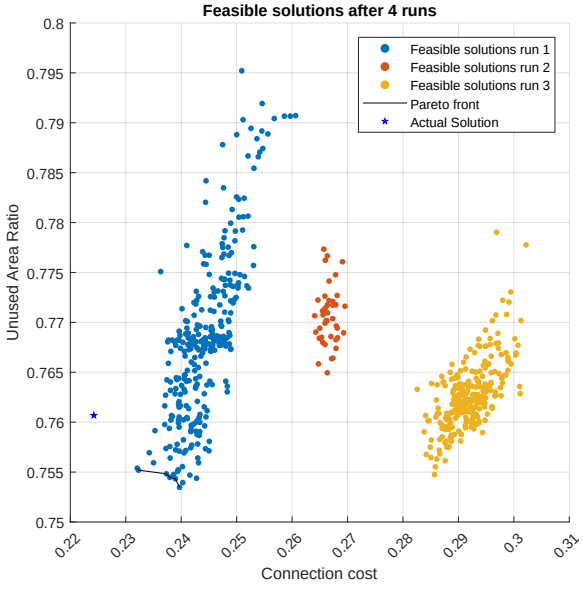
(b) $c_2 = 1.4$



(c) $c_1 = 1.6$

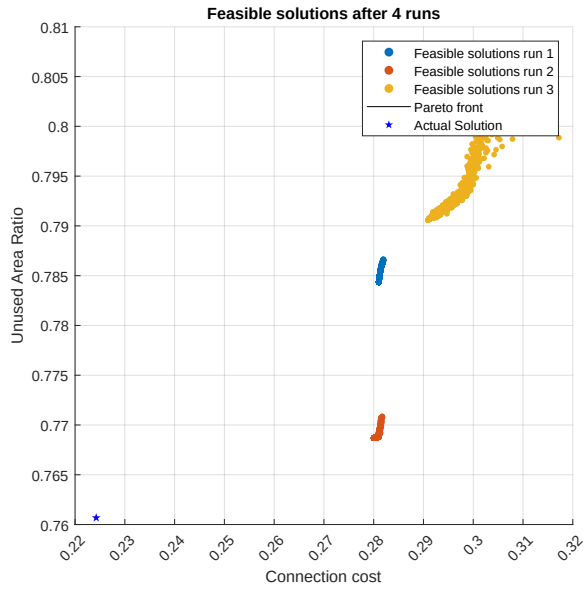


(d) $c_2 = 1.8$

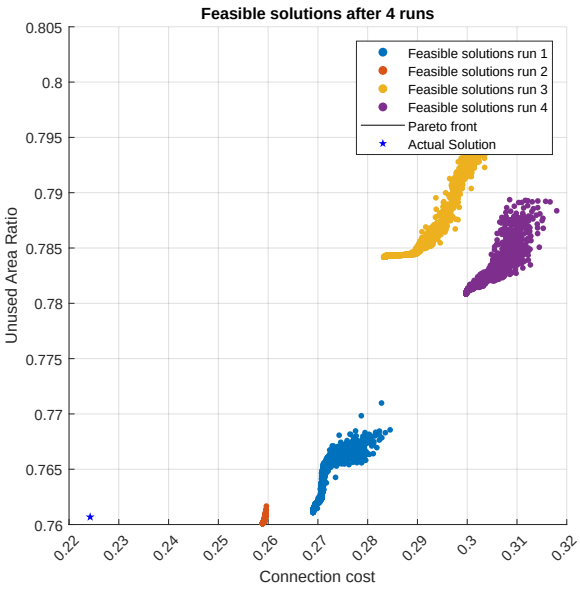


(e) $c_1 = 2.0$

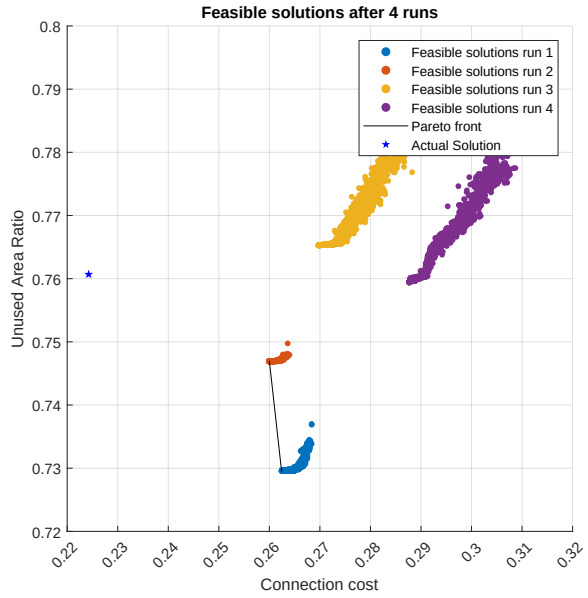
Figure A.10: Runs with constant w, and $c_2(w)$, and varying c_1



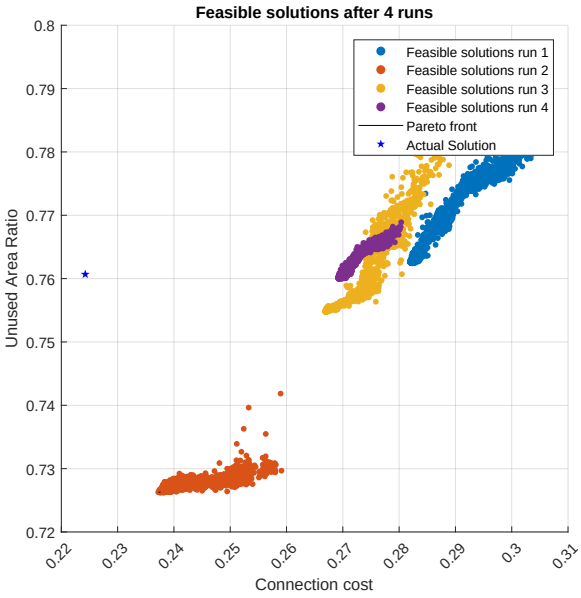
(a) $w = 0.40$



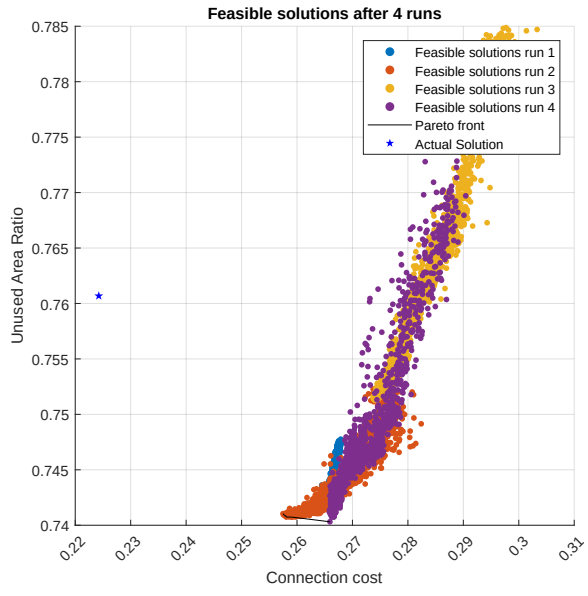
(b) $w = 0.45$



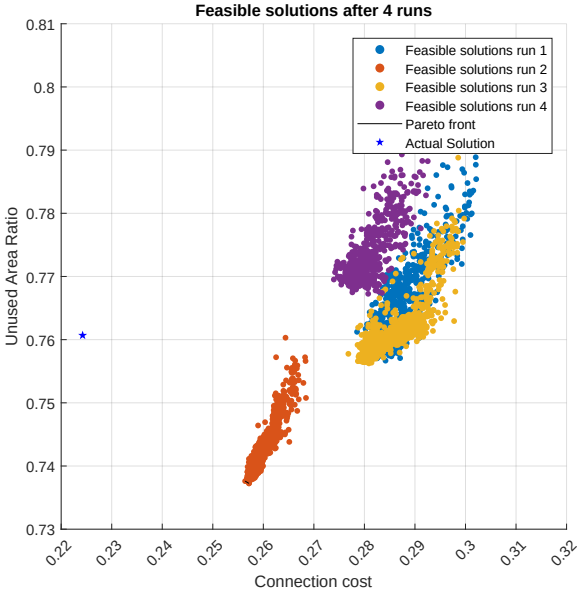
(c) $w = 0.50$



(d) $w = 0.55$

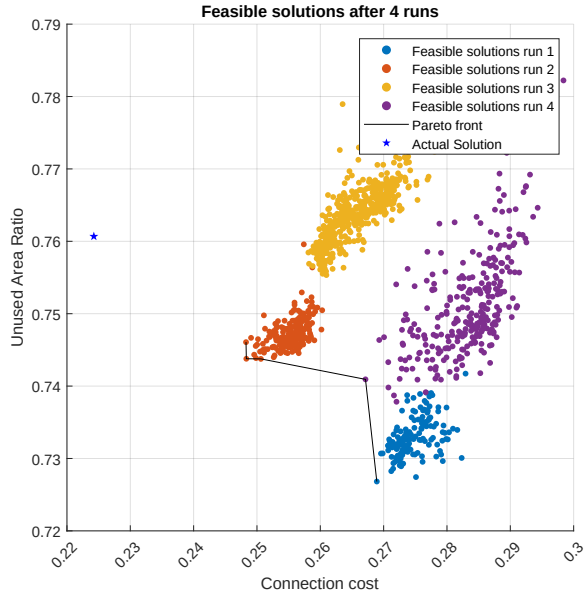


(e) $w = 0.60$

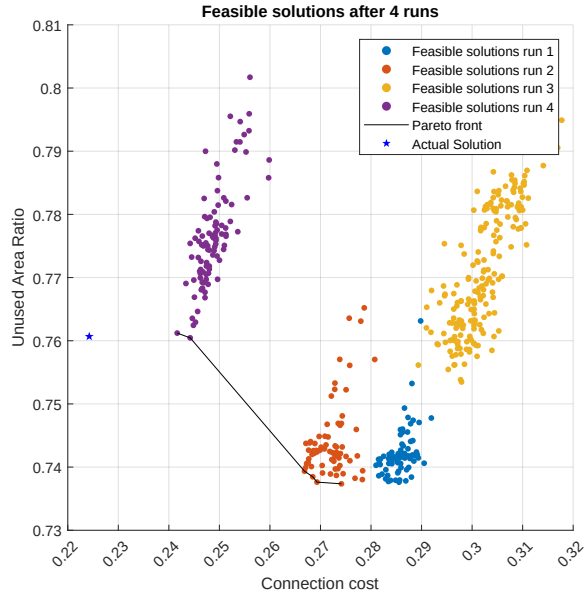


(f) $w = 0.65$

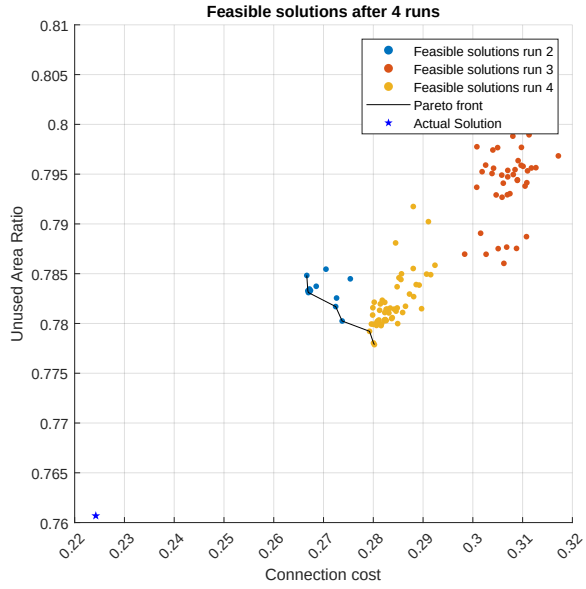
Figure A.11: Runs with constant c_1, and c_2, and varying w



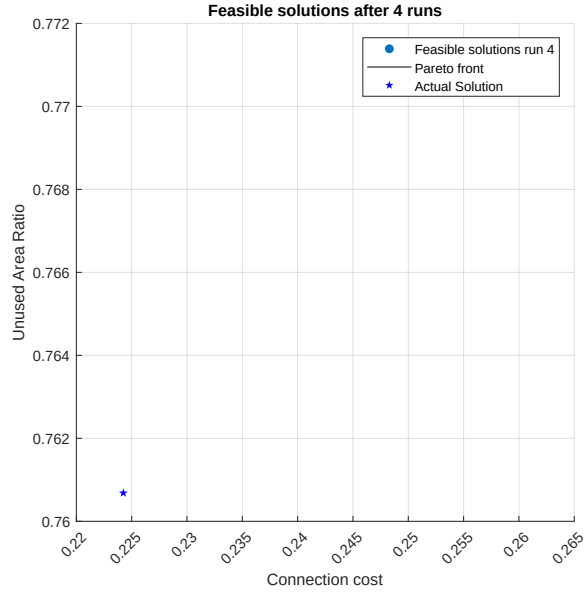
(a) $w = 0.70$



(b) $w = 0.75$



(c) $w = 0.80$

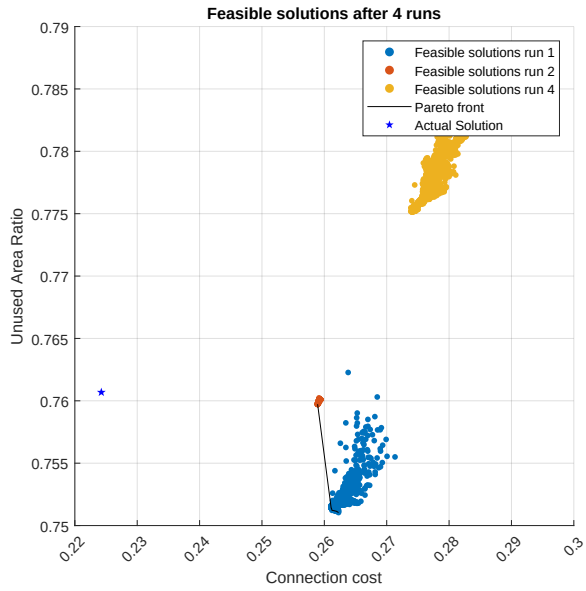


(d) $w = 0.85$

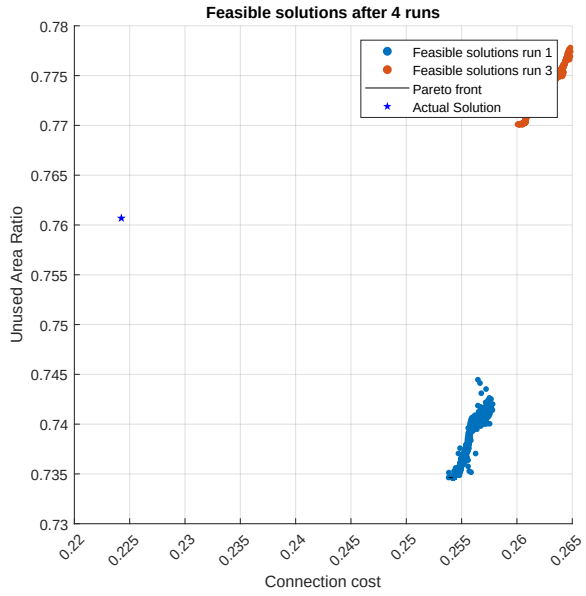
Figure A.12: Runs with constant c_1, and c_2, and varying w

A.3. Mutations

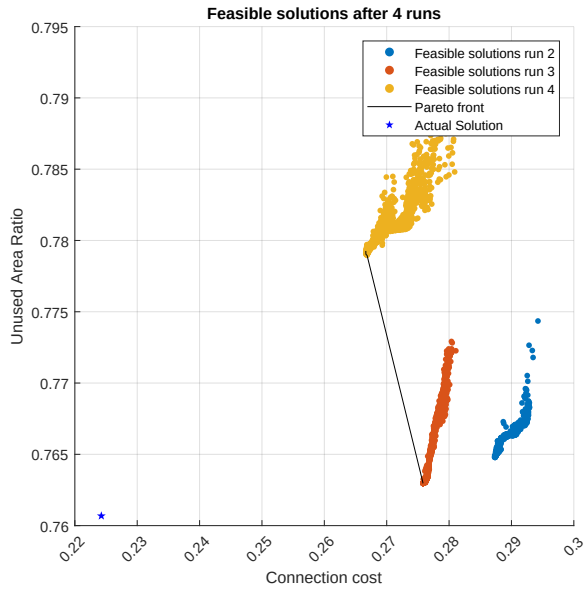
A.3.1. Mutation: Replace



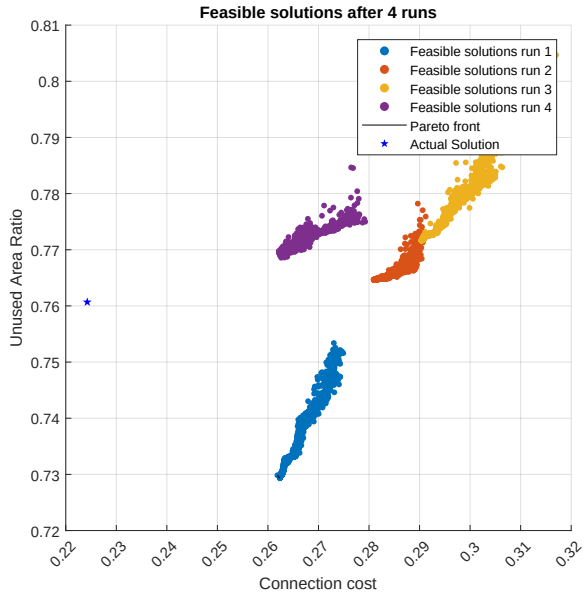
(a) $w = 0.40, c_1(w) = c_2(w)$



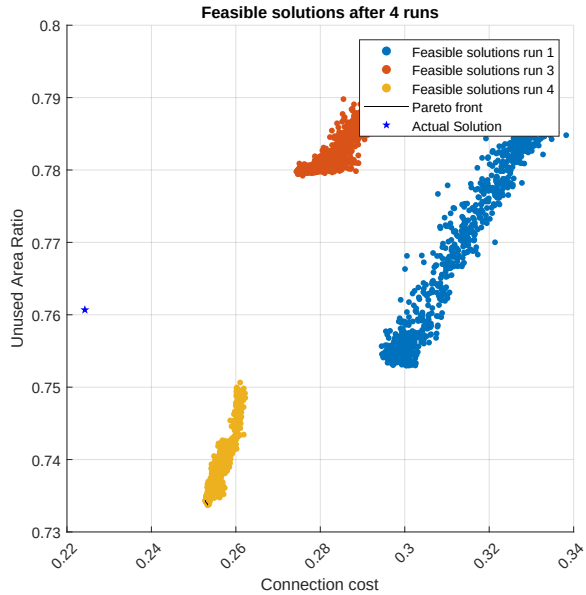
(b) $w = 0.45, c_1(w) = c_2(w)$



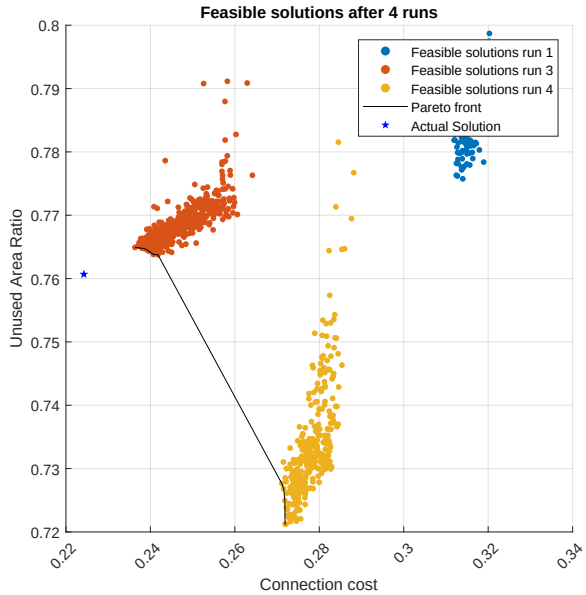
(c) $w = 0.50, c_1(w) = c_2(w)$



(d) $w = 0.55, c_1(w) = c_2(w)$

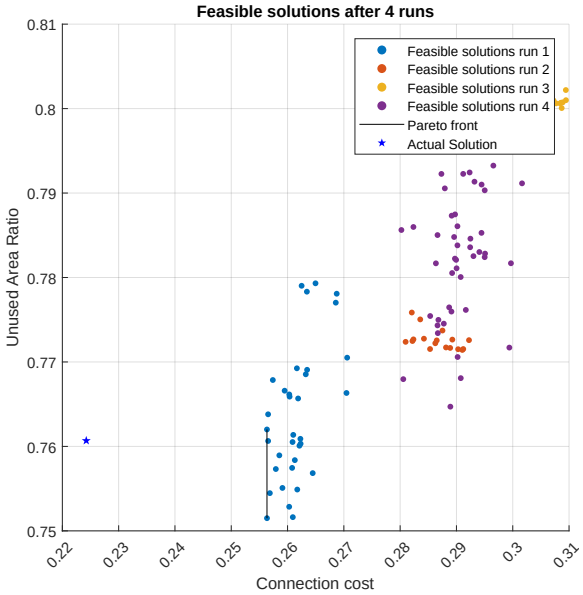


(e) $w = 0.60, c_1(w) = c_2(w)$

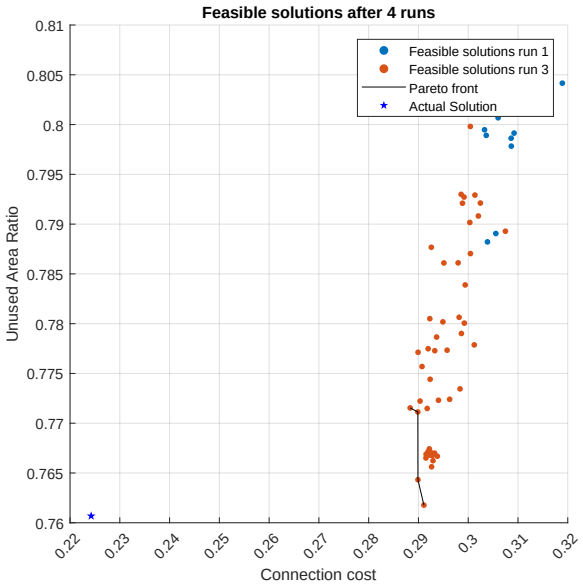


(f) $w = 0.65, c_1(w) = c_2(w)$

Figure A.13: Scatter plot of solutions found with mutation: Replace, a mutation percentage of 20% while varying w and $c_1(w) = c_2(w)$

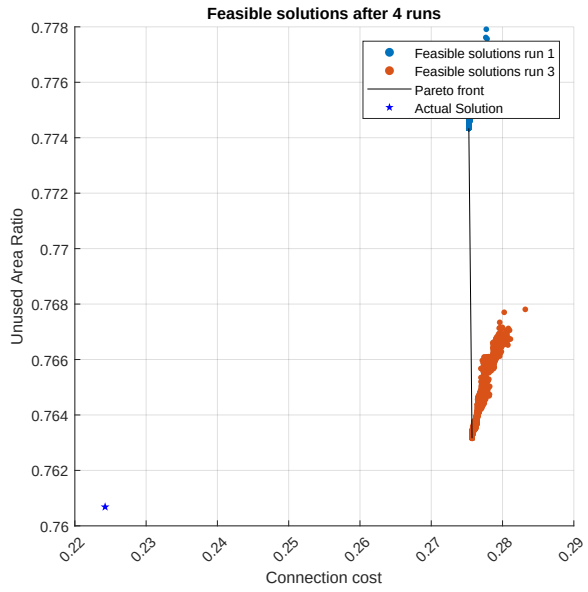


(a) $w = 0.70, c_1(w) = c_2(w)$

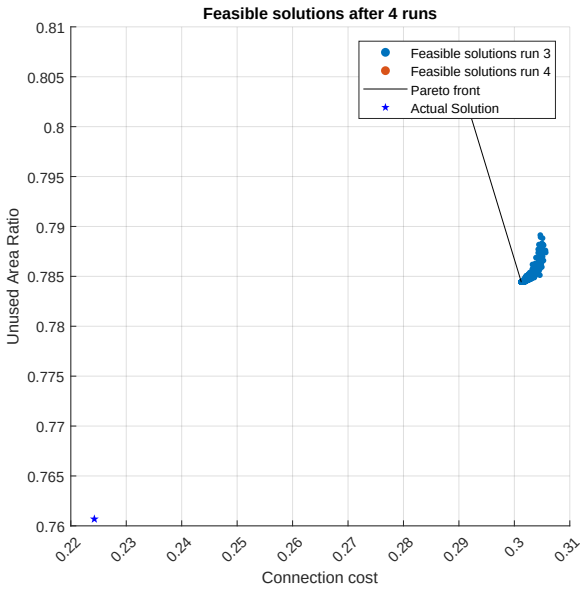


(b) $w = 0.75, c_1(w) = c_2(w)$

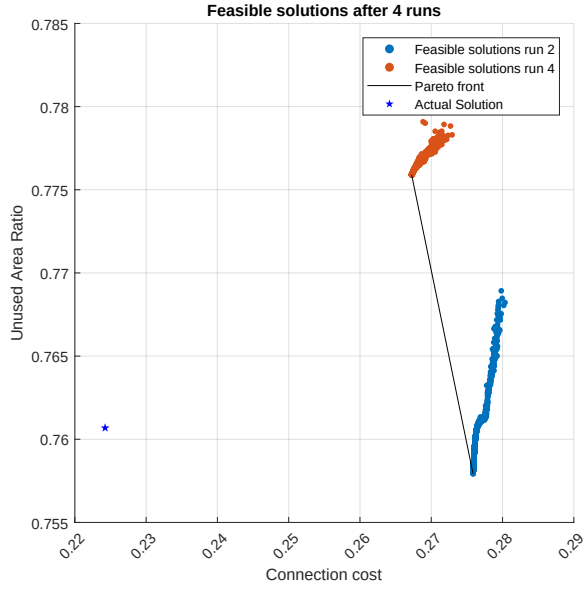
Figure A.14: Scatter plot of solutions found with mutation: Replace, a mutation percentage of 20% while varying w and $c_1(w) = c_2(w)$



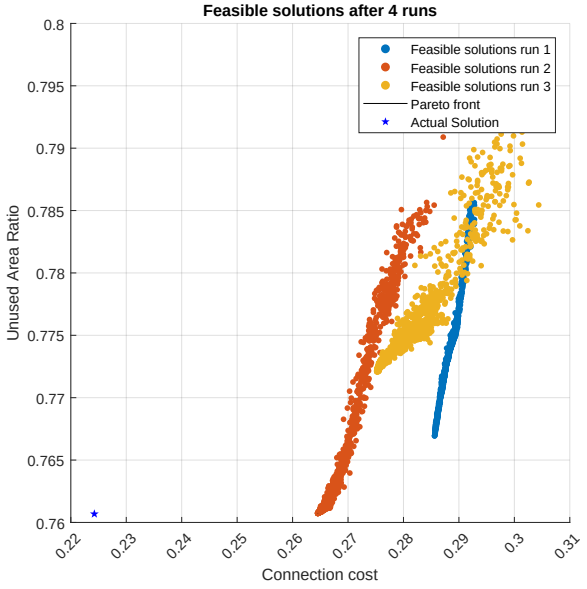
(a) $w = 0.40, c_1(w) = c_2(w)$



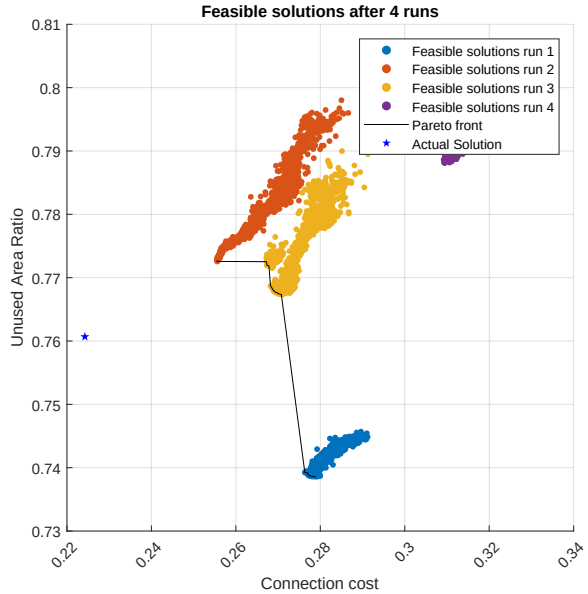
(b) $w = 0.45, c_1(w) = c_2(w)$



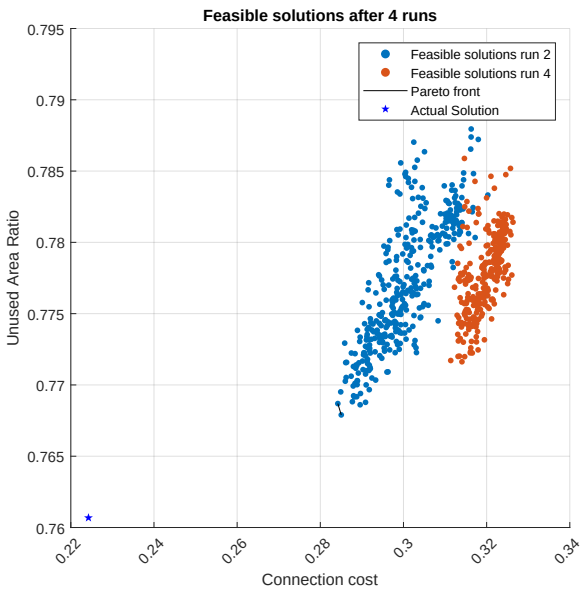
(c) $w = 0.50, c_1(w) = c_2(w)$



(d) $w = 0.55, c_1(w) = c_2(w)$

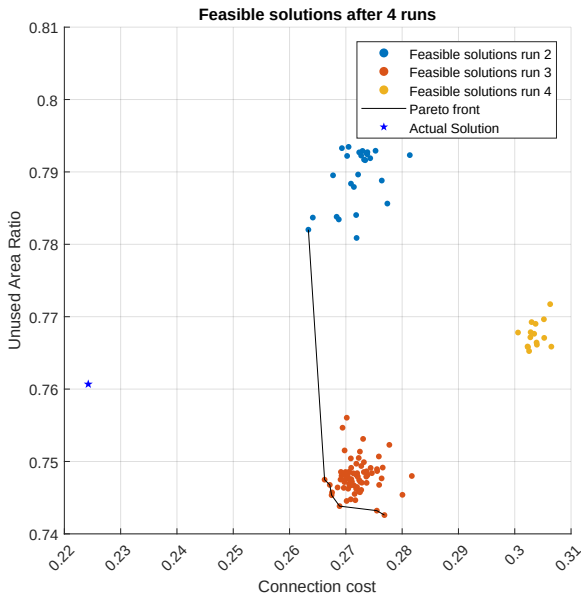


(e) $w = 0.60, c_1(w) = c_2(w)$

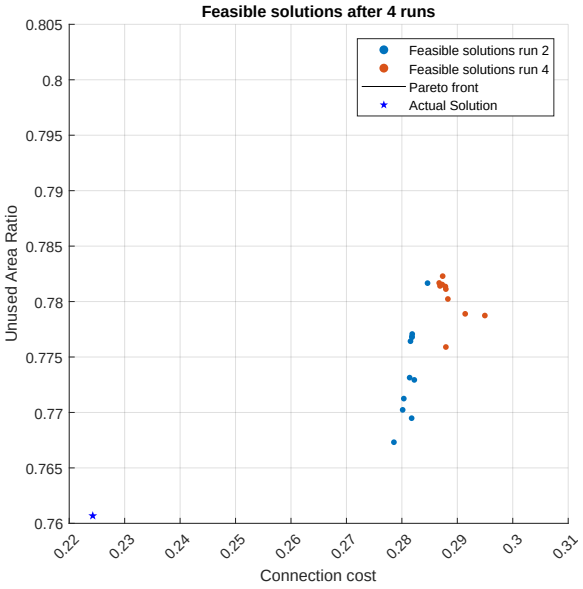


(f) $w = 0.65, c_1(w) = c_2(w)$

Figure A.15: Scatter plot of solutions found with mutation: Replace, a mutation percentage of 40% while varying w and $c_1(w) = c_2(w)$



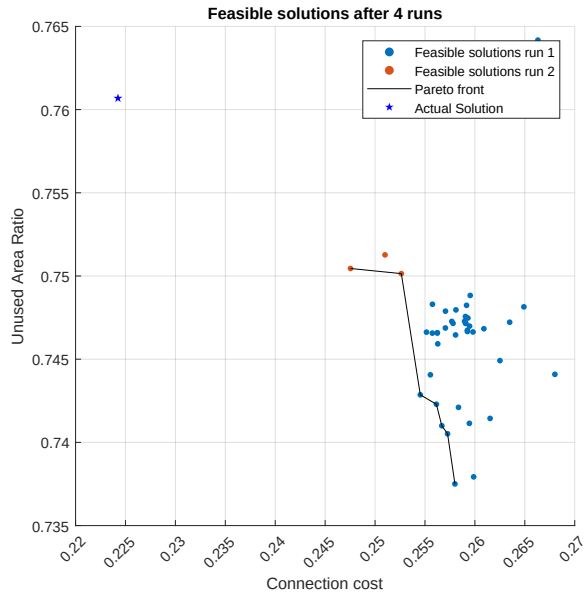
(a) $w = 0.70, c_1(w) = c_2(w)$



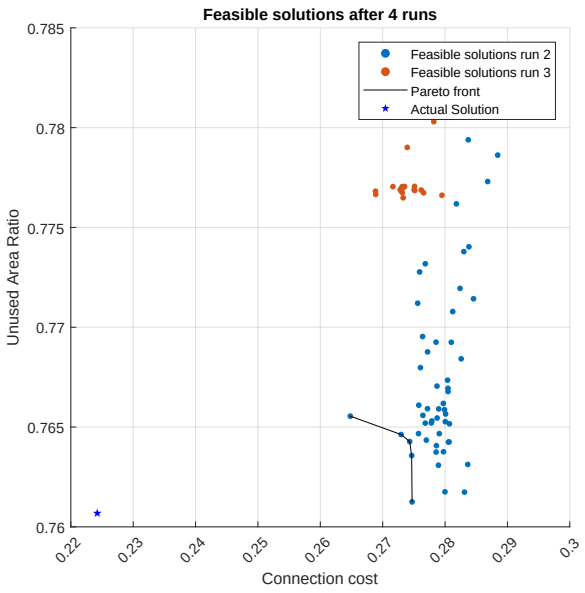
(b) $w = 0.75, c_1(w) = c_2(w)$

Figure A.16: Scatter plot of solutions found with mutation: Replace, a mutation percentage of 40% while varying w and $c_1(w) = c_2(w)$

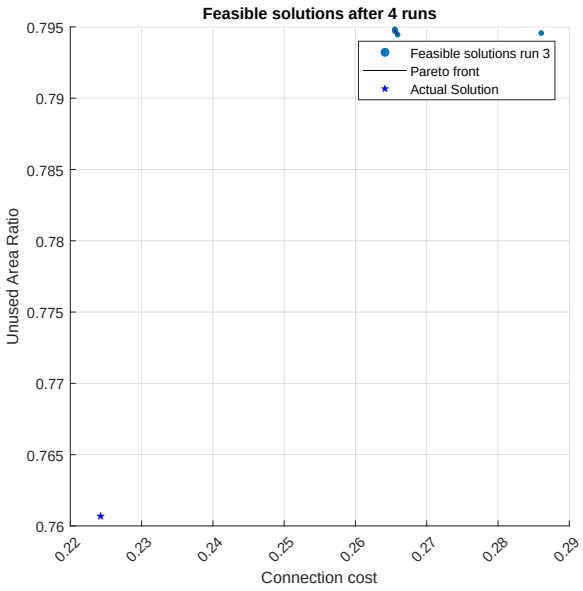
A.3.2. Mutation: Swap



(a) $w = 0.45, c_1(w) = c_2(w)$

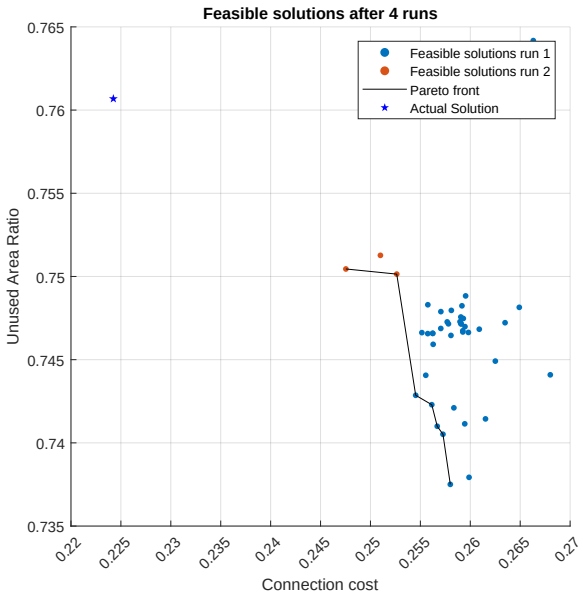


(b) $w = 0.50, c_1(w) = c_2(w)$

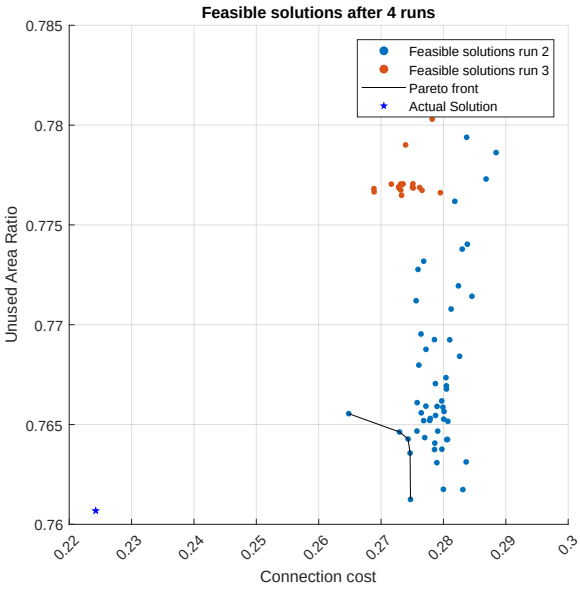


(c) $w = 0.60, c_1(w) = c_2(w)$

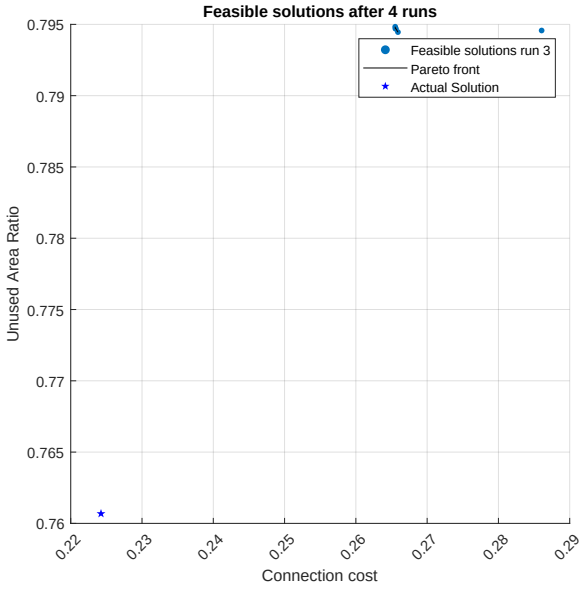
Figure A.17: Scatter plot of solutions found with mutation: Swap while varying w and $c_1(w) = c_2(w)$



(a) $w = 0.45, c_1(w) = c_2(w)$



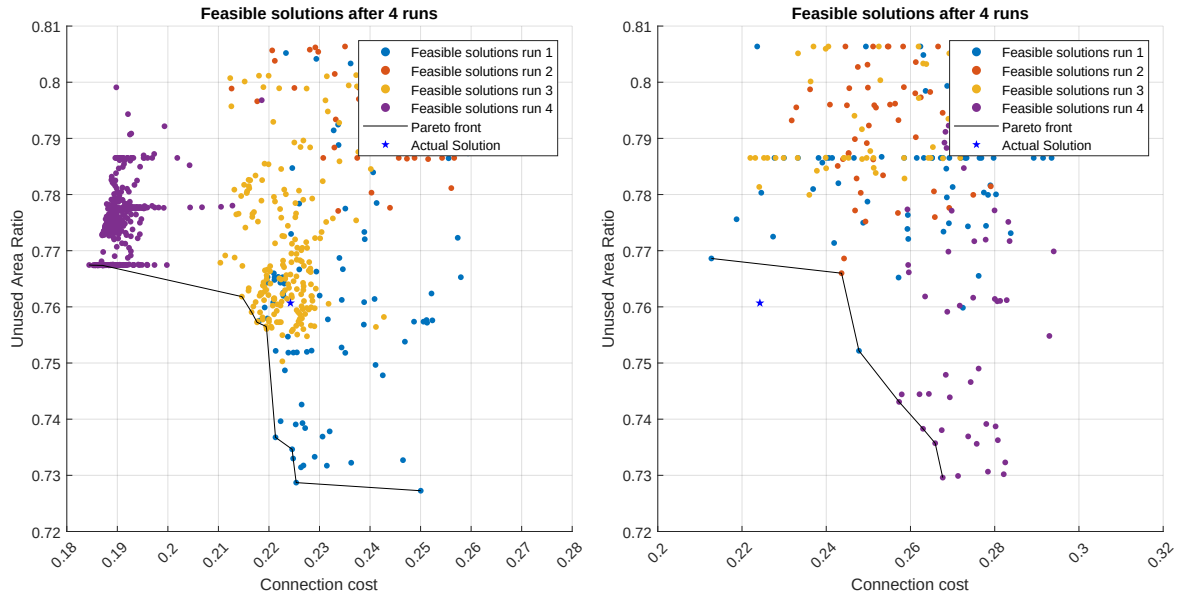
(b) $w = 0.50, c_1(w) = c_2(w)$



(c) $w = 0.60, c_1(w) = c_2(w)$

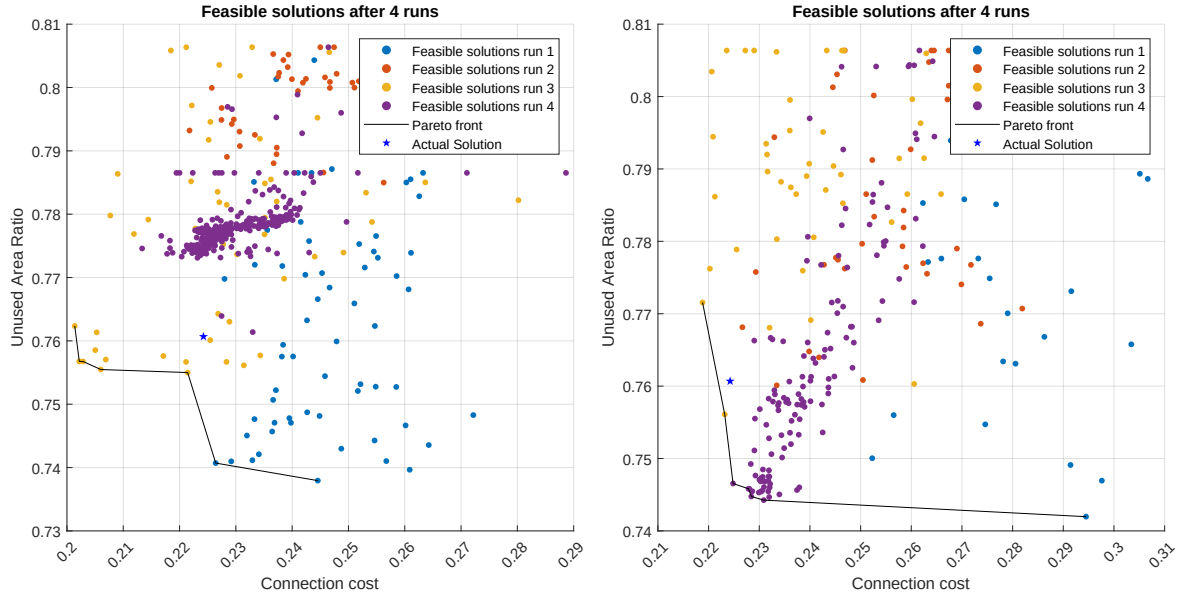
Figure A.18: Scatter plot of solutions found with mutation: Swap while varying w and $c_1(w) = c_2(w)$

A.3.3. Mutation: Improve Nondominated Particles



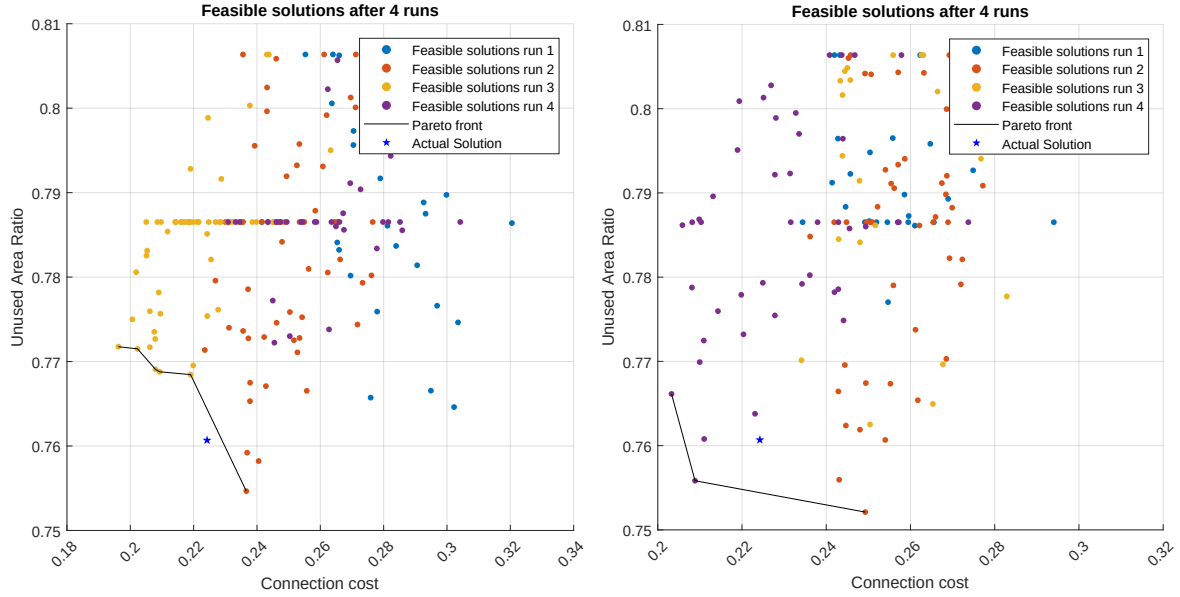
(a) $w = 0.40, c_1(w) = c_2(w)$

(b) $w = 0.45, c_1(w) = c_2(w)$



(c) $w = 0.50, c_1(w) = c_2(w)$

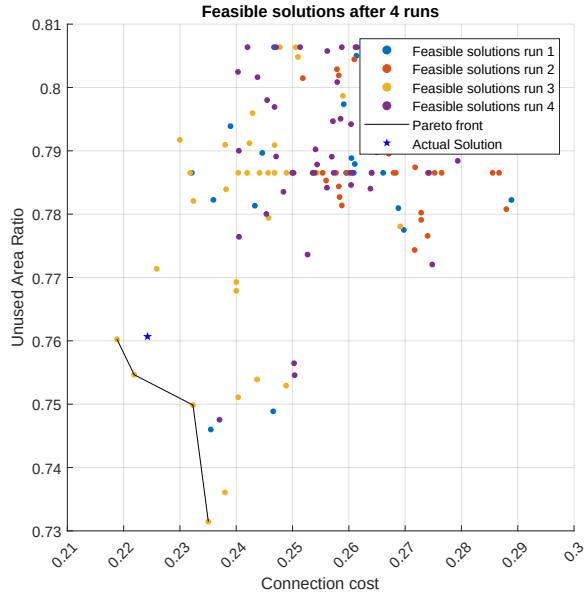
(d) $w = 0.55, c_1(w) = c_2(w)$



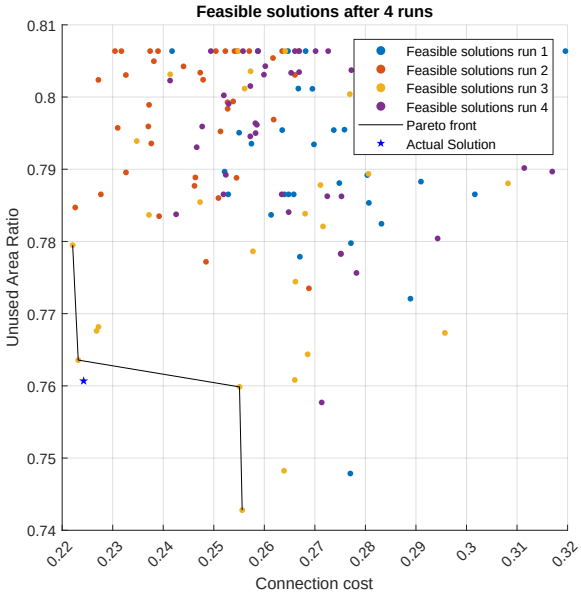
(e) $w = 0.60, c_1(w) = c_2(w)$

(f) $w = 0.65, c_1(w) = c_2(w)$

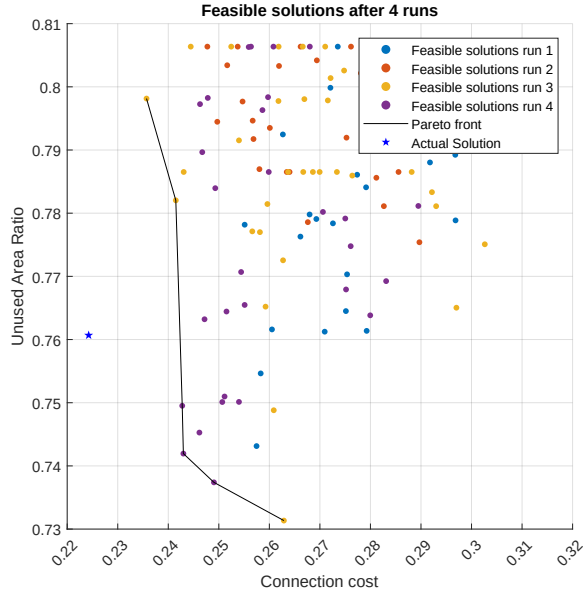
Figure A.19: Scatter plot of solutions found with mutation: Improve Nondominated Particles while varying w and $c_1(w) = c_2(w)$.



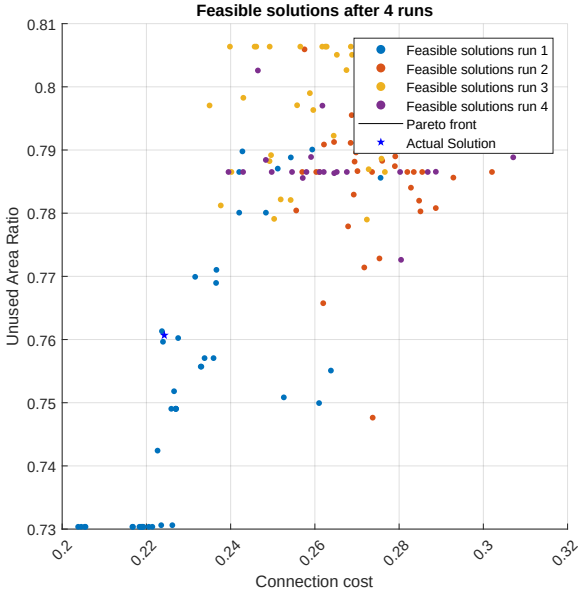
(a) $w = 0.70, c_1(w) = c_2(w)$



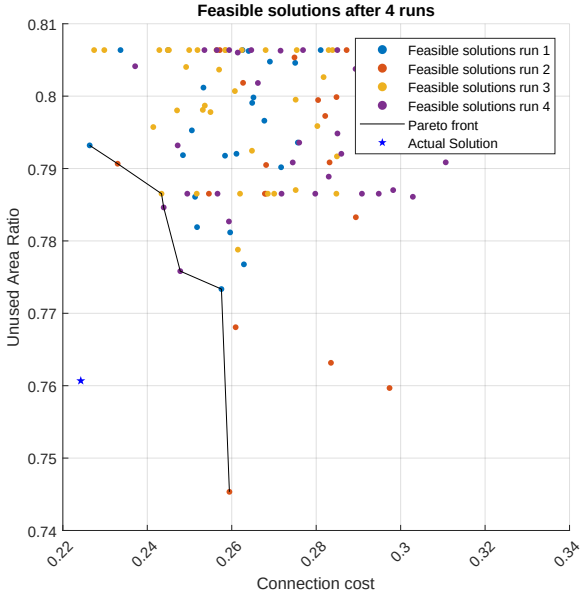
(b) $w = 0.75, c_1(w) = c_2(w)$



(c) $w = 0.80, c_1(w) = c_2(w)$



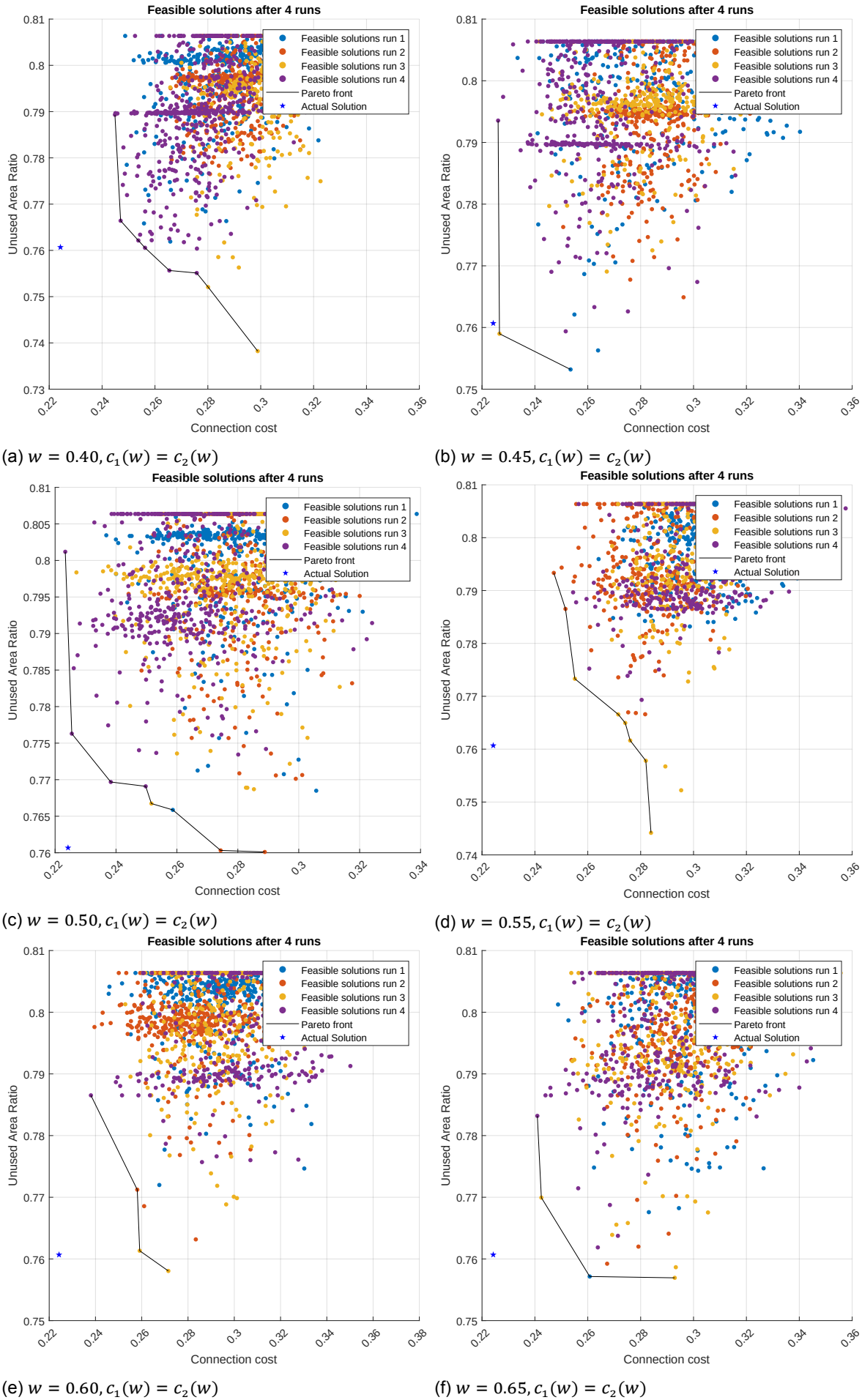
(d) $w = 0.85, c_1(w) = c_2(w)$



(e) $w = 0.90, c_1(w) = c_2(w)$

Figure A.20: Scatter plot of solutions found with mutation: Improve Nondominated Particles while varying w and $c_1(w) = c_2(w)$

A.3.4. Mutation: Improve Infeasible Particles



(a) $w = 0.40, c_1(w) = c_2(w)$

(b) $w = 0.45, c_1(w) = c_2(w)$

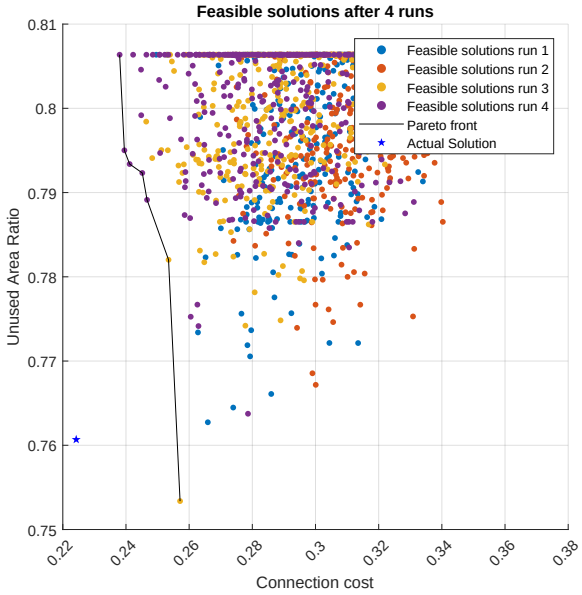
(c) $w = 0.50, c_1(w) = c_2(w)$

(d) $w = 0.55, c_1(w) = c_2(w)$

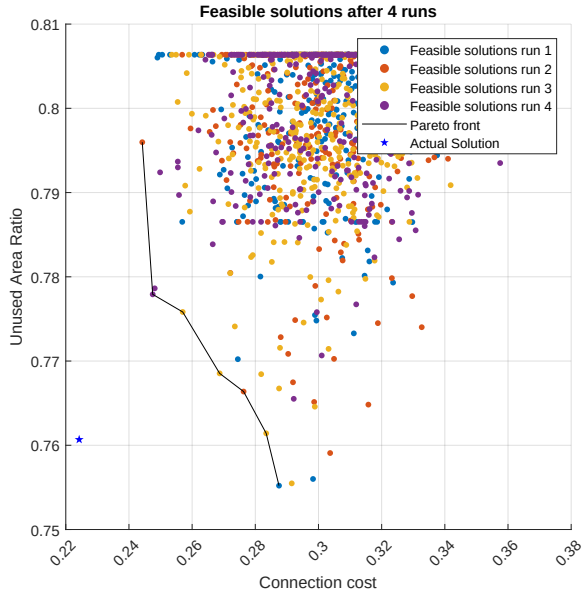
(e) $w = 0.60, c_1(w) = c_2(w)$

(f) $w = 0.65, c_1(w) = c_2(w)$

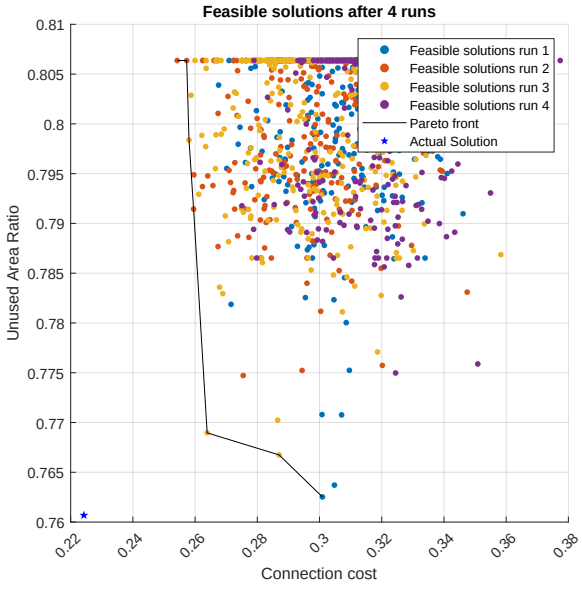
Figure A.21: Scatter plot of solutions found with mutation: Improve Infeasible Particles while varying w and $c_1(w) = c_2(w)$



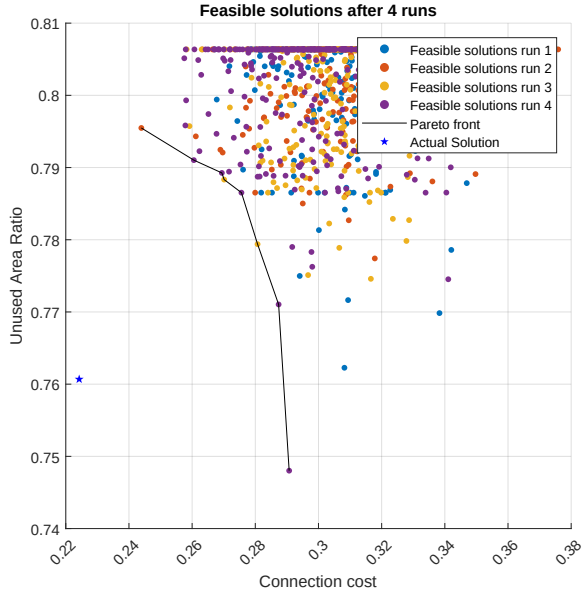
(a) $w = 0.70, c_1(w) = c_2(w)$



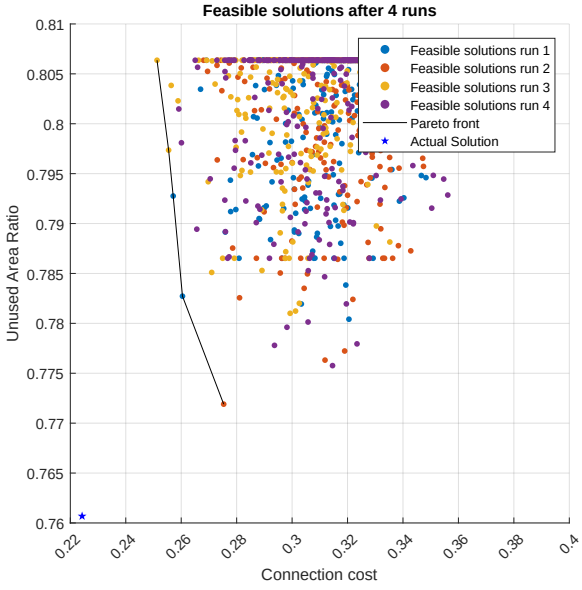
(b) $w = 0.75, c_1(w) = c_2(w)$



(c) $w = 0.80, c_1(w) = c_2(w)$



(d) $w = 0.85, c_1(w) = c_2(w)$



(e) $w = 0.90, c_1(w) = c_2(w)$

Figure A.22: Scatter plot of solutions found with mutation: Improve Infeasible Particles while varying w and $c_1(w) = c_2(w)$