UNICAMP

UNIVERSIDADE ESTADUAL DE CAMPINAS
Faculdade de Engenharia Elétrica e de Computação

Flávia Barbosa

Recents Advances on The Berth Allocation Problem

Avanços Recentes ao Problema de Alocação de Berços

Campinas
2018

Flávia Barbosa

Recents Advances on The Berth Allocation Problem

Avanços Recentes ao Problema de Alocação de Berços

Tese (Doutorado) apresentada à Faculdade de Engenharia Elétrica e de Computação da Universidade Estadual de Campinas como parte dos requisitos exigidos para a obtenção do título de Doutora em Engenharia Elétrica, na Área de Automação.

PhD Dissertation presented to the School of Electrical and Computer Engineering of the University of Campinas in partial fulfillment of the requirements for the degree of PhD in Electrical Engineering, in Automation area.

Supervisor: Prof. Dr. Akebo Yamakami
Co-supervisor: Prof. Dr. Priscila C. Berbert Rampazzo

ESTE EXEMPLAR CORRESPONDE À VERSÃO FINAL DA TESE (DOUTORADO) DEFENDIDA PELO ALUNO FLÁVIA BARBOSA E ORIENTADA PELO PROF. DR. AKEBO YAMAKAMI.

Campinas
2018

Agência(s) de fomento e nº(s) de processo(s): CNPq, 141225/2014-9

Ficha catalográfica
Universidade Estadual de Campinas
Biblioteca da Área de Engenharia e Arquitetura
Luciana Pietrosanto Milla - CRB 8/8129

B234r

Barbosa, Flávia, 1989-
Recent advances on the berth allocation problem / Flávia Barbosa. – Campinas, SP : [s.n.], 2018.

Orientador: Akebo Yamakami.
Coorientador: Priscila Cristina Berbert Rampazzo.
Tese (doutorado) – Universidade Estadual de Campinas, Faculdade de Engenharia Elétrica e de Computação.

1. Problema de alocação de berços. 2. Meta-heurística. 3. Programação matemática. I. Yamakami, Akebo, 1947-. II. Rampazzo, Priscila Cristina Berbert, 1984-. III. Universidade Estadual de Campinas. Faculdade de Engenharia Elétrica e de Computação. IV. Título.

Informações para Biblioteca Digital

Título em outro idioma: Avanços recentes ao problema de alocação de berços
Palavras-chave em inglês:
Berth allocation problem
Metaheuristics
Mathematical programming
Área de concentração: Automação
Titulação: Doutora em Engenharia Elétrica
Banca examinadora:
Akebo Yamakami [Orientador]
Luiz Leduino Salles Neto
Fabricio Olivetti de França
Antônio Carlos Moretti
Romis Ribeiro de Faissol Attux
Data de defesa: 15-03-2018
Programa de Pós-Graduação: Engenharia Elétrica

COMISSÃO JULGADORA - TESE DE DOUTORADO

Candidato: Flávia Barbosa RA: 083552
Data da Defesa: 15 de março de 2018

Título da Tese: "Recents Advances on The Berth Allocation Problem"
"Avanços Recentes ao Problema de Alocação de Berços"
Prof. Dr. Akebo Yamakami (Presidente, FEEC/UNICAMP)
Prof. Dr. Luiz Leduino Salles Neto (UNIFESP)
Prof. Dr. Fabricio Olivetti de França (UFABC)
Prof Dr. Antônio Carlos Moretti (FCA/UNICAMP)
Prof. Dr. Romis Ribeiro de Faissol Attux (FEEC/UNICAMP)

A ata da defesa com as respectivas assinaturas dos membros da Comissão Julgadora, encontra-se no processo de vida acadêmica do aluno.

To Márcia, Luiz and Rodrigo.

Acknowledgements

I thank my parents, Luiz and Márcia, for the support and love during the challenges.

I thank my little brother, Rodrigo, for always helping me with my computational issues.

I thank my boyfriend, Eduardo, for the affection whenever I needed during this journey.

I thank my friends who always cheered for my success.

I thank my supervisors, Akebo and Priscila for their dedication and contribution.

I thank my former supervisor, Moretti, for his friendship over the years.

I thank Professors José Fernando e Maria Antónia receiving me so well at University of Porto.

Last but not the least, I thank CNPq for providing the financial support for this PhD.

"It is the Lord who goes before you.
He will be with you;
He will not leave or forsake you.
Do not fear or be dismayed."
(Deu 31:8)

Abstract

The intermodal transportation of goods by vessels has increased over the years. In this context, the Berth Allocation Problem (BAP) arises and becomes fundamental to guarantee the efficiency of the maritime terminals, deciding where and when to allocate the vessel over a planning horizon taking into account constraints of time and space. Because the problem is proved NP-hard, this study proposes an exact method and analyzes metaheuristics for tackling the problem. First, considering the BAP as a parallel-machine scheduling problem, an approach for this problem is proposed based on an Evolutionary Metaheuristic, aiming to find several good quality solutions in a single round of the algorithm, considering explicitly the BAP with multiple objectives. A lower bound based on a maximal flow problem was derived in order to evaluate the quality of the solutions. Next, based on a heterogeneous vehicle routing problem with time windows a basic Benders Decomposition algorithm and its variants are reviewed and applied to the BAP. Then, a hybrid optimization procedure based on Genetic Algorithm (GA) and Scatter Search (SS)is developed, and data envelopment analysis (DEA) is adopted to choose the efficient combination of the operators for the algorithm proposed. Because most papers in literature use in their experiments data generated randomly, making comparisons between researches difficult, this thesis proposes a problem generator for the BAP, allowing the generation of appropriate test problems to be commonly used with specific desired properties and under controlled conditions. The data are generated using different parameters and the difficulty of solving the BAP with such data is analyzed through the resolution using the CPLEX. Finally, the instances classified as more difficult are solved through two metaheuristics implemented.

Keywords: berth allocation problem; metaheuristics; mathematical programming; benchmark data instances.

Resumo

O transporte de mercadorias por navios aumentou ao longo dos anos. Neste contexto, o Problema de Alocação de Berços (BAP) surge e torna-se fundamental para garantir a eficiência dos terminais marítimos, ao decidir onde e quando alocar o navio no horizonte de planejamento, levando em consideração restrições de tempo e espaço. Uma vez que o problema foi provado ser NP-hard, este estudo propõe um método exato e uma análise de muitas metaheurásticas para resolvê-lo. Primeiro, considerando o BAP como um problema de sequenciamento de tarefas de máquinas paralelas, uma abordagem é proposta com base em uma Metaheurística Evolutiva, com o objetivo de encontrar várias soluções de boa qualidade em uma única rodada do algoritmo, considerando explicitamente o BAP com múltiplos objetivos. Um limitante inferior baseado em um problema de fluxo máximo foi derivado para avaliar a qualidade das soluções. Em seguida, com base em um problema de roteamento de veículo com janelas de tempo, um algoritmo de decomposição Benders e suas variantes são revisados e aplicados ao BAP. Então, um algoritmo híbrido com base no Algoritmo Genético e Busca Dispersa é desenvolvido e a Análise Envoltória de Dados é adotada para escolher a combinação eficiente de operadores para o algoritmo proposto. Como a maioria dos trabalhos na literatura usa em seus experimentos computacionais dados gerados aleatoriamente, dificultando as comparações entre pesquisas, esta tese também propõe um gerador de dados para o BAP, permitindo que a geração de problemas-teste que sejam comumente usada, padronizando as comparações em trabalhos futuros. Os dados são gerados usando diferentes parâmetros e a dificuldade de resolver o BAP com esses dados é analisada através do CPLEX. Finalmente, as instâncias classificadas como as mais difíceis são resolvidas através de duas metaheurísticas.

Palavras-chaves: problema de alocação de navios; metaheurísticas; programação matemática; gerador de dados.

List of Figures

List of Tables

Contents

1 Introduction

According to the United Nations Conference on Trade and Development ((UNCTAD, 2016)), maritime transport carried over 80% of the volume of global merchandise trade of the world's goods in 2015 for developing countries. However the growing pace in seaborne shipping is the smallest since 2009. The carrying capacity, on the other side, increased by 3.5% to 1.8 billion deadweight tons. Both movements together led to an increase in the available capacity and the freight rates dropped. The freight rates should however decrease even more to attract volume in an increasing pace and this is only possible by a reduction in the operating costs that does not involve major investments. In the ports this can be obtained by minimizing the handling costs which are directly related with the waiting and service times of the vessels.

Berths are a very important resource and a good allocation of vessels to berths entails a reduction in handling costs. This issue has been the subject of research, giving rise to the Berth Allocation Problem (BAP) which can be stated as: where and when to allocate arriving vessels to a berth space over a planning horizon taking into account constraints of time and space, related to the length of the vessels, their arrival times, the number of containers for loading or unloading, the location of the charge stock, time windows, among others. Some assumptions made may be different for each terminal, such as the possibility of waiting for vessels, if several vessels can moor in the same berth, if the vessels arrival time is considered, if the service time is proportional to the size of the vessel, among others. One of the most important characteristics of the problem is whether the berthing space is considered discrete or continuous. It is considered to be discrete if the quay is viewed as a finite set of berths and each berth is described by fixed-length segments or as points and it is considered to be continuous if the vessels can berth anywhere along the quay depending only on the position of other vessels.

According to (MONACO; SAMMARRA, 2007), if the vessels have release dates, the BAP is NP-hard. Therefore, for large instances evolutionary metaheuristics are often recommended to solve the BAP. Metaheuristic is high-level problem-independent algorithmic framework developed specifically to find a solution that is *good enough* in a computing time that is *small enough*. As a result, the computing time required to find a solution for NP-hard problems does not increase as an exponential function of the problem size, when computing exact optimal solutions is computationally intractable. According to (YAGIURA; IBARAKI, 2001), metaheuristics are attractive because they can be developed even if deep mathematical properties of the problem domain are not hand and can still in difficult cases obtain solutions better than those obtained by exact methods and simple heuristics. This thesis aims to study the variations of the BAP, the different methods of resolution, exact or metaheuristic, which can be used to obtain good quality solutions and the influence of the data in solving the problem.

Several models were developed for the BAP based on other problems in the literature such as cutting and packing, scheduling, vehicle routing or generalized set partitioning and, due to the difficulty of the problem, several heuristic approaches are proposed in the literature. Chapter 2 presents a detailed review of the most relevant literature published in the last 10 years. Because the BAP is a combinatorial problem, in Chapter 3 an approach based on an Evolutionary Metaheuristic is proposed. The method works simultaneously with a set of solutions in order to perform exploration and exploitation of the search space, allowing it to find several good quality solutions that can serve as alternatives to a given scenario. The goal is to find this set of solutions in a single round of the algorithm, considering explicitly the BAP with multiple objectives. In Chapter 4 a constructive heuristic with local search is developed in order obtain good solutions for the BAP modeled as a scheduling problem. It is based on the principle that the problem of scheduling can be represented by a maximum flow problem in which preemption in the task handling is allowed. The aim is to verify if it results in an algorithm capable of finding good lower bounds for evolutionary metaheuristics. Benders decomposition is a cutting plane method that has been widely used for solving large-scale mixed integer linear optimization problems, and yet it has never been applied to the BAP. In Chapter 5 the basic Benders Decomposition algorithm and its variants are reviewed and applied to the reformulated BAP. The BAP involves many criteria that can be used to evaluate how good a solution is. For this reason, there are different ways to configure the implemented algorithm and we need a tool to guide the decision on how to use each proposed operator. In Chapter 6 a hybrid optimization procedure is developed based on Genetic Algorithm (GA) and Scatter Search (SS) for the discrete and dynamic BAP (Hybrid Evolutionary Algorithm for the BAP - HEABAP). The data envelopment analysis (DEA) is adopted to choose the efficient combination of the operators for the algorithm proposed. When reviewing these different models it was possible to conclude that there are no benchmark instances available for the BAP and most papers in the literature use in their experiments randomly generated data for that particular paper. There is therefore the need for a problem generator for the BAP problem and a set of controlled test instances that enable the researchers to compare their approaches. To overcome such drawback, in Chapter7 a problem generator is developed for the BAP, allowing the generation of a large number of problem instances with specific desired properties and under controlled conditions. The difficulty of the parameter combinations for the beta distribution are classified and in Chapter 8 two metaheuristics are developed to try to obtain good feasible solutions for the problem in a short computational time. A classical Genetic Algorithm (GA), one of the first metaheuristic proposed in the literature and easily adaptable to any type of problem, is developed and compared with a recent Particle Swarm Algorithm (PSO).

Therefore, the aim of this thesis is to seek recent advances for the BAP, considering different perspectives.

2 Literature Review

This chapter is organized as follows. In Section 2.1 a literature review of the BAP is presented. The models are classified according to the characteristics of berthing space, vessels arrivals, service time, integration with other problems and multiobjective optimization. In Section 2.2 the BAP is formulated based on formulations for other classical problems, and a few comparisons about the models and their complexity are presented. In Section 2.3 the problem classification is summarized and Section 2.4 presents improvements for a generic BAP model considered throughout this thesis.

First, to illustrate the discrete Berth Allocation Problem problem, consider the following numerical example. There are five vessels to be allocated to two berths. Each vessel i has a processing time p_i^k, different for each berth k since it depends on the equipment available for (un)loading, an arrival time a_i and a departure time b_i, as shown in Table 1.

Table 1 – Small instance for the discrete BAP

vessel (i)	processing time at berth 1 (p_i^1)	processing time at berth 2 (p_i^2)	arrival time (a_i)	departure time (b_i)
1	35	40	5	95
2	30	100	0	65
3	25	20	5	55
4	35	30	5	85
5	40	20	20	55

Let x_i^k be the start time for the service of vessel i at berth k. An example of the allocation of the five vessels in the two berths is outlined in Figure 1 and Table 2:

Table 2 – Solution for the instance in Table 1 represented in Figure 1

vessel i (i)	start time of service (x_i^k)	berth (k)	processing time (p_i^k)	waiting time ($x_i^k - a_i$)
1	30	1	35	25
2	0	1	30	0
3	5	2	20	0
4	45	2	30	40
5	25	2	20	5
total			135	70

Table 2 shows that the total time spent to (un)load the five vessels and the total waiting time were, respectively, 135 and 70.

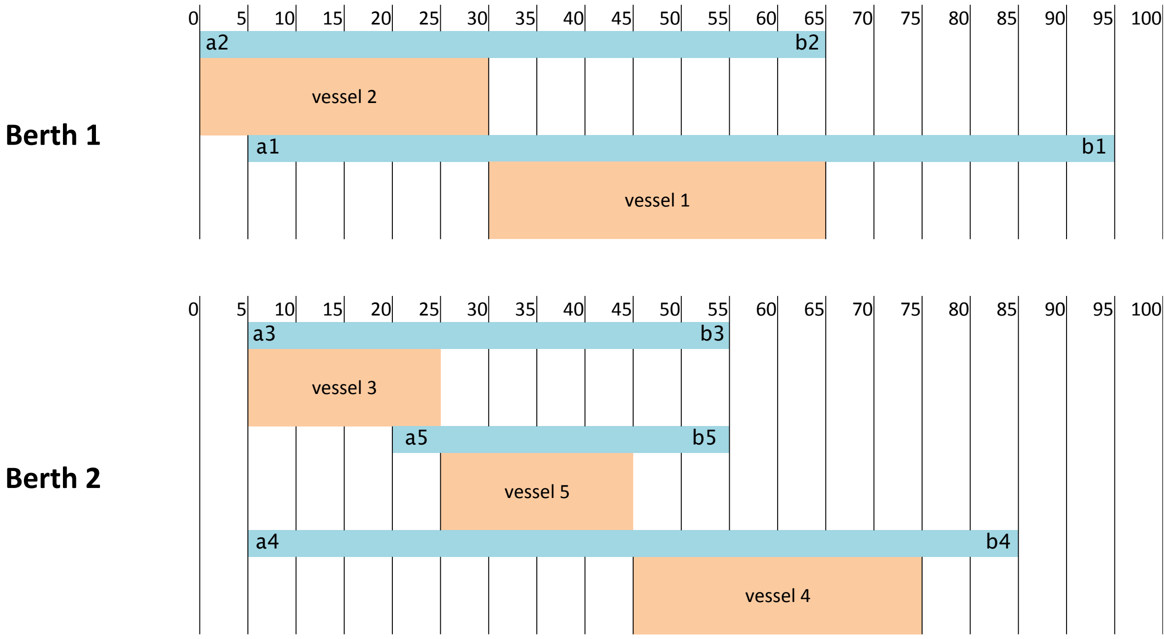


Figure 1 – Example of allocation for the instance in Table 1

2.1 Literature Review on the Berth Allocation

Several approaches for the BAP have been proposed in the literature, addressing problems found in real-world ports. For instance, based on the port of Hong Kong case, (GUAN; CHEUNG, 2004) considered the problem of allocating vessels to berths, allowing multiple vessel mooring per berth and considering the vessels' arrival times, the so-called dynamic problem. The objective was to minimize the total weighted flow time, i.e., the sum of the vessel's waiting and service times, where the weights reflect the relative importance of the vessels. (LI; PANG, 2011) considered the integrated vessel routing and berth assignment problem, for a shipping company operating a fleet of feeder vessels shuttling among various terminals in Hong Kong and the Pearl River Delta, aiming to minimize both the total travel cost of the vessels and the cost incurred in the loading/unloading operations. Some ports in Europe and in China, where there is an intense vessel traffic, are called multi-users terminals: a large number of incoming vessels are simultaneously and dynamically allocated on a long quay, without having a fixed position. The quay is partitioned into several berths and the allocation of vessels to a berth is based on both the characteristics of the vessel and the berth. This type of terminal, according to (IMAI *et al.*, 2005), is widely used in busy container ports, since their productivity heavily depends on the efficient allocation of vessels. (ARANGO *et al.*, 2011) study the berth allocation problem for containerships in the port of Seville, the only inland port in Spain. With several types of cargo, as cereals, scrap metal and cement containers, the traffic increased greatly and it became a bottleneck as there is only one dock for small vessels. It is proposed a simulation-optimization approach aiming minimizing the sum of vessel service time (handling operations time, berth waiting time and logistic operations time) considering truck arrivals, containership arrivals, berth assignment systems, towing vessels, etc.

This variety of characteristics and goals that can be found in practice led to a multitude of approaches. However, these approaches may be classified and grouped according to some common features. In the following sections the most relevant and recent literature on the Berth Allocation Problem (BAP) will be reviewed and organized. Special attention will be given to the data sets used for the computational experiments and validation of each approach.

2.1.1 Berthing space: discrete versus continuous

The berthing space can be considered discrete or continuous. In the discrete BAP, the quay is viewed as a finite set of berths, and at each moment of time only one vessel can be assigned to each berth. A model for this version of the problem was introduced by (CORDEAU *et al.*, 2005). This model has as objective the minimization of the weighted sum of service times and includes constraints related to time-windows for vessel berthing times. An heuristic based on tabu search was developed to solve the discrete version of the BAP, and was then extended to the continuous case. The discrete BAP is also studied in (BUHRKAL *et al.*, 2011) with three different models, each one with a different objective function (minimization of total waiting and handling times, minimization of the weighted sum of vessel service times and minimization of vessels service time). The decision variables determine the assignment of vessels to berths as well as the order by which vessels will be processed in each berth. (BARROS *et al.*, 2011) considered also the discrete form of the problem in tidal bulk port terminals, the so-called Berth Allocation Problem in Tidal Bulk ports with Stock level conditions (BAPTBS). The objective was to minimize the total demurrage[1] incurred, given the tidal conditions and the stock level constraints, considering similarly equipped berth positions. The main assumptions that were made relate to tidal conditions and stock level, as observed in the maritime industrial port complex located in São Luís-MA, Brasil. The proposed model considers the problem as a transportation problem in which the vessels are regarded as origins and the favorable tidal condition as destinations. The test instances were randomly generated, based on real scenarios. In (HANSEN *et al.*, 2008) berths are also considered as a discrete resource and the Minimum Cost Berth Allocation Problem (MCBAP) is proposed with the objective of minimizing vessel waiting and handling costs, as well as earliness premiums and lateness penalties. A Variable Neighborhood Search (VNS) metaheuristic is developed to solve the problem since the model running time, even for a small example, exceeded several hours of computing time. Three sets of instances were used

[1] charges payed to the vessel owner for its delayed operations of loading/unloading.

in the tests: the same as in (HANSEN; OGUZ., 2003), an extended version of instances from (HANSEN; OGUZ., 2003) and a generated set.

In the continuous Berth Allocation Problem vessels can berth anywhere along the quay depending only on the position of other vessels. For this version of the problem, (FROJAN *et al.*, 2015) has recently proposed a mixed integer linear model with multiple quays, including several realistic characteristics, as that a given vessel cannot moor at a given quay for technical or contractual reasons or that the vessels present different adequacies to different quays. The main decision variables are related to the berthing position of a given vessel at the quay to which it is assigned, and vessel relative position variables. The objective function considers the minimization of the waiting and delay costs for each vessel, the vessel-quay assignment cost and the cost associated to the deviation of each vessel form its desired berthing position.

2.1.2 Vessels arrivals: static versus dynamic

Another important characteristic of Berth Allocation Problems concerns the vessel arrival. The problem is classified as static (Static Berth Allocation Problem - SBAP) if all vessels to be serviced are already in the port when scheduling begins. Alternatively, the problem is classified as dynamic (Dynamic Berth Allocation Problem - DBAP), if not all vessels to be scheduled for berthing have arrived at the beginning of the planning horizon, although arrival times are know in advance. According to (MONACO; SAMMARRA, 2007), the computational complexity of the BAP lies on the dynamic arrival process of the vessels. Indeed this problem is NP-hard even if there is a single berth, as it reduces to minimizing the total completion time with release dates on a single machine. On the other hand, the static version of the problem is solvable in polynomial time since it reduces to an assignment problem.

In (IMAI *et al.*, 2001) a model for the static BAP is formulated: binary variables x_{ijk} indicate if vessel j is handled as the kth vessel at berth i, or not. The model was extended to the dynamic BAP, having as objective the minimization of the sum of waiting and handling times for every vessel. A subgradient optimization procedure, based on the Lagrangian relaxation, was developed. (HANSEN; OGUZ., 2003) revisited (IMAI *et al.*, 2001) models and propose a new model for the static BAP where the binary decision variables, as in (IMAI *et al.*, 2001), reflect both the berth assignment and the order by which the vessels are handled, but with the particularity of looking at the vessel sequence from the end to the beginning: $x_{ijk} = 1$ if vessel j is the kth last to be handled at berth i. An extension of this model to the dynamic berth allocation problem is also discussed, and a compact reformulation is proposed, with some further extensions. (HANSEN; OGUZ., 2003) claim to correct an error that (IMAI *et al.*, 2001) models included in the objective function.

(IMAI *et al.*, 2003) incorporated priorities in the dynamic version of the BAP (Priority Berth Allocation Problem - PBAP). The objective function aims the minimization of the sum of each vessel service time weighted by its priority, which in the described application is related to the total handling needs, i.e. vessels with a larger container handling volume have a higher priority. A Lagrangian relaxation formulation is developed to the PBAP, but given the hardness of the relaxed problem (a Quadratic Assignment Problem) and the computational burden that its resolution would imply, a genetic algorithm developed. Later on, in (IMAI *et al.*, 2007), the dynamic berth allocation problem was again addressed at a multi-user container terminal with indented berths for fast handling of mega-containerships. An indented multi-user container terminal is characterized by its capability of fast handling from both sides of large vessels. But if they are small, multiple vessels are permitted to be simultaneously served at a specific berth, which results in the nonlinearity in the formulation. A linear formulation is introduced based on decision variables of berth-vessel-order assignment, and a genetic algorithm is proposed. Also in (SIMRIN; DIABAT, 2015) the dynamic berth allocation problem is formulated as a nonlinear mixed integer program, in which the non-linearity arises in the objective function when trying to model the time during which the terminal remains idle. Again, a genetic algorithm is developed and tested on 6 different instances, considering different numbers of vessels and berths. The dynamic BAP was also solved in (ARANGO *et al.*, 2013), aiming the minimization of the distances traveled by the forklifts and the quay crane, during container loading and unloading operations. A genetic algorithm was integrated in a simulation model, wich is used to test the efficiency of each vessel allocation. The arrival times were taken from Algeciras port database, in Spain, in October 2010.

Finally, (IMAI *et al.*, 2008) looked at a variation of the BAP in which vessels that would normally be served at a multi-user terminal, with a limited capacity, are assigned to an external terminal if their expected waiting time exceeds the time limit. The authors named this problem as the Berth Allocation Problem with an External Terminal (BAPE). Two formulations are proposed, one for the static BAPE and another for the dynamic BAPE. The goal is to find the optimal assignment of vessel-berth-service order, so that the total service time of vessels that are allocated to the external terminal is minimized. A genetic algorithm was developed for the DBAPE and tested for the port of Colombo, Sri Lanka, for 10 days in June 2003.

2.1.3 Handling time: static versus dynamic - integration with the quay crane assignment problem

A vessel should never wait too long to be serviced, as this represents an immobilization cost for the client and an opportunity cost for the port. The duration of a vessel berthing depends on the number of quay cranes allocated to the vessel: as the number of quay cranes allocated to a vessel increases, the duration of vessel berthing decreases. For the discrete BAP this means that the handling time of a vessel may be different for different berths, even if the decision regarding the number of cranes to assign to each vessel is not involved (e.g. (BUHRKAL *et al.*, 2011)).

Hence, the handling time can also be classified as static if the number of cranes that will serve each vessel is fixed, or dynamic if the number of cranes that will work on each vessel is variable and decided together with the berth position and service time. As a result, an integrated Quay Crane Allocation Problem (QCAP) and Berth Allocation Problem (BAP) arises.

According to (VACCA *et al.*, 2013), the problem resulting from the integration of BAP with QCAP is very complex. They implement an exact branch and price algorithm that aims at assigning vessels to berthing positions, performing the scheduling of vessels in each berth and allocating quay cranes (QC) to vessels over a given time horizon, taking into account the quay crane capacity of the terminal. (TURKOGULLARI *et al.*, 2014) also develop an exact solution algorithm - a cutting plane algorithm - to minimize the costs of deviation from the desired berth section, berthing later than the arrival time and departing later than the due time. A binary integer linear program was formulated for the integrated solution of the berth allocation and quay crane assignment.

Being the integrated discrete Berth Allocation and Quay Crane Scheduling Problem (IBAQCSP) NP-hard ((LEE; WANG, 2010)) it is not surprising that heuristic methods have been developed and proposed for this problem. (LEE; WANG, 2010) proposed a genetic algorithm for this problem that was tested on forty random instances, systematically generated. Another genetic algorithm was proposed in (LIANG *et al.*, 2009), combined with a heuristic. The goal is to minimize the sum of handling time, waiting time and delay time for every vessel. (IMAI *et al.*, 2008) also used a genetic algorithm to find an approximate solution for the B&CAP (the BAP formulation from (IMAI *et al.*, 2001) amended with some constraints). The algorithm determines the berth scheduling and the crane scheduling at the same time, with the goal of minimizing the total service time (waiting and handling times). The GA was also used in an hybrid multistage operation approaches developed to facilitate local convergence in (LIANG *et al.*, 2012). The concept of transshipment of vessel to vessel was introduced with the consideration that the total number of quay cranes on berth is fixed. The transshipment is made between two vessels and occurs after the earlier arriving vessel has finished its loading or unloading operations without transshipment. The objective is to minimize the sum of the handling time, waiting time of container vessels on berths (time interval two vessels spend performing the transshipment operation), the delay time of container vessels departure and the waiting time of transshipment. The method is used to solve a case from one of Shanghai container terminal companies in China.

(LALLA-RUIZ *et al.*, 2014) modeled the Tactical Berth Allocation Problem

(TBAP) involved processes: how to determine the berthing position, berthing time and allocation of quay cranes for container vessels arriving to the port over a well-defined time horizon maximizing the sum of the values of the chosen quay crane profiles assigned to all the vessels and, simultaneously, minimizing the yard-related housekeeping cost generated. The proposed problem was solved by a Biased Random Key Genetic Algorithm (BRKGA). (GIALLOMBARDO *et al.*, 2008) present a mixed integer quadratic programming formulation for the Tactical Berth Allocation Problem (TBAP) with quay cranes assignment as well. The problem maximizes the sum of the values of the chosen quay crane assignment profiles over all the vessels and minimizes the yard-related housekeeping costs generated by the flows of containers exchanged between vessels. The formulation has been tested with CPLEX 10.2, which was able to solve some instances at optimality. For others instances it hardly finds a feasible solution and therefore a reformulation based on Dantzig-Wolfe decomposition and column generation, and an incremental approach based on Lagrangian dual, was considered in order to exploit the structure of TBAP and its relation with the BAP formulation. Also for the integrated tactical berth allocation problem (TBAP) and the quay crane assignment problem, (GIALLOMBARDO *et al.*, 2010) proposed a two level heuristic. First the quay crane profiles are assigned for the vessels and next the resulting berth allocation problem is solved for the given quay crane assignment. This procedure is repeated for several quay cranes profiles, which are chosen using the reduced costs arguments of mathematical programming. A tabu search algorithm was developed to solve the BAP aiming to minimize the yard-related transshipment costs.

(GOLIAS *et al.*, 2009a) divided the quay into a number of berths and each berth can service one vessel at a time and assumed that the vessel handling time is proportional to vessel capacity and the assigned berth. It was done in order to minimize the total waiting and delayed departure time for all vessels, reducing indirectly the fuel consumption and emissions produced by the vessels while in idle mode. The resolution approach presented is a genetic algorithm based heuristic. In (THEOFANIS *et al.*, 2007) it was supposed that vessel handling time is berth dependent, because it is related to the time of the landside transfer operations. The problem also considered one long wharf at a multi-user terminal, which was divided into several berths to obtain a set of assignments of vessels to those berths and it was formulated as a linear mixed integer program with the objective of minimizing the total weighted service time of all the vessels (Weighted Berth Allocation Problem - WBAP) and solved with a genetic algorithm based heuristic for medium to large instances.

The integrated berth allocation and quay crane assignments proposed in (CHANG *et al.*, 2010) is based on rolling-horizon approach to minimize the total deviation between the actual and best berthing locations based on each planning horizon, the total penalty for delayed berthing and departure time of vessels and the total energy consumption of quay cranes. A heuristic algorithm is used to reduce the solution dimension and generate feasible solutions for the initialization of the population parallel genetic algorithm. A combination of simulation and optimization technologies is proposed to evaluate the proposed BAP and QCAP strategies. A rolling horizon framework was used as well as in (RAA *et al.*, 2011). The model, that incorporates additional real-life features, assumed that all vessels approaching the berth need to be scheduled at minimum cost, once the each vessel has a desired position to be berth, which is close to the dedicated storage location of the containers that will be (un)loaded. The costs come from penalties for vessel handling delays, deviating from a vessel's preferred berthing location and changes in the number of cranes assigned to a vessel during its service. A hybrid heuristic solution procedure is used to validate the model with a three-month data set from the port of Antwerp. A sensitivity analysis of the available number of quay cranes, quay length and management parameters expressing the trade-offs between cost components was performed to illustrate the model's capabilities to support managerial decision making.

The approach addressed in (YANG *et al.*, 2012) for a multi-user container terminal not only included the berth allocation (BAP) and quay crane assignment (QCAP) problems, but also the interactions between them. The vessel berthing time and departure time obtained in the BAP determine the time window of the corresponding vessel in the QCAP, which updates the vessel handling time and the vessel departure time, and supplies feed-back to the BAP. A a nested loop-based evolutionary algorithm (NLEA) is developed for solving the problem. (MEISEL; BIERWIRTH, 2013) provide a framework, solving jointly not only the berth allocation problem (BAP) and the quay crane assignment problem (QCAP), but also the quay crane scheduling problem (QCSP). Well-known heuristics were used. First, the QCSP is solved for each vessel under a variable number of employed cranes to obtain crane productivity rates. Next, these rates are included in a berth allocation and crane capacity assignment problem (BACCAP) to decide on the berthing position, berthing time, and crane capacity assigned to each vessel. Finally, to generate an overall crane schedule, the QCSP is solved again with respect to the decisions made, establishing time windows for the crane operations (QCSPTW).

2.1.4 Integration with yard management

Yard management thus involves optimal allocation of storage areas for import, export and transshipment containers. If the departure position of a container is far from its yard position, the container must be reallocated before the arrival of the outbound vessel. It means that the favorite berth for each vessel is determined in long-term, inducing container flows inside the yard. The aim o integrating the BAP with yard planning relies on determining if accommodating a costumer request is feasible and how it impacts the whole terminal performance. (PRATAP *et al.*, 2016) simultaneously optimized the stockyard operations and rake schedule for outbound cargo, in conjunction with the arriving vessels and the status of the stockyards at the port. With the goal minimizing the service time of rakes and the delay in unloading time of the vessel due to conflict of stacker/reclaimer, a genetic algorithm approach and a block-based evolutionary algorithm are developed to tackle real-life instances.

The start and end times of vessel operations determine the workload distribution and the deployment of yard equipment. Moreover, berthing locations of vessels determine the storage locations of specific cargo types to specific yard locations. Similarly, the yard assignment of specific cargo types has an impact on the best berthing assignment for vessels berthing at the port. (ROBENEK *et al.*, 2014) combined and solved the berth allocation and the yard assignment for bulk ports as a single large scale optimization problem. A bulk terminal manager faces the challenge of maximizing efficiency both along the quay side and the yard. The operations planning can be divided into tactical level (resource allocation) and operational level (daily and real time decisions). The objective function is to minimize the total service time of all vessels: the sum of total delays and total handling time of vessels berthing at the port. The proposed mixed integer model was decomposed: the master problem is formulated as a set-partitioning problem and subproblems identify columns with negative reduced costs and a metaheuristic approach based on critical-shaking neighborhood search was presented in order to obtain sub-optimal solutions quickly. The test instances are based on a sample of data obtained from the SAQR port, Ras-Al-Khaimah, UAE, the biggest bulk port in the middle east for a time horizon of roughly 10 days from 28th March to 6th April, 2011. (HENDRIKS *et al.*, 2013) present a simultaneous berth allocation and yard planning problem (BAP and YAP, respectively) at tactical level. The objective is to minimize the overall straddle carrier travel distance, finding the berth locations for vessels and assigning storage blocks to containers. A heuristic that alternates between BAP and YAP until no further improvement is possible is proposed. The instances used (cyclic timetable, vessels' load compositions and yard layout) were provided by the terminal operator PSA Antwerp.

2.1.5 Multiobjective optimization approach

According to (CHEONG; TAN, 2008), as port operators try to optimize their operations to obtain a high throughput, there is also a need to account for the satisfaction levels of vessel operators, requiring to minimize concurrently the multiple conflicting objectives. For example, from the point of view of vessels operator, an ideal berthing plan is one where vessels do not have to wait to be berthed and be serviced in the shortest possible time. However, from the point of view of port operators, an ideal berthing plan is one where the makespan is minimal.

Therefore, a multi-objective optimization approach is induced. They optimized the complete schedules with minimum service time and delay in the departure of vessels, subject to a number of temporal and spatial constraints using an ant colony optimization (ACO) incorporated to heuristics in the search process in the form of ant visibility. (IMAI *et al.*, 2007) determine an assignment of calling vessels to berths for cargo handling, with two objectives to minimize: the delay in vessels' departure and the total service time. Two heuristics were proposed based on two existing procedures of the subgradient optimization and genetic algorithm. It was carried out experiments with four berths and 24 calling vessels that arrive randomly at the terminal with an exponential distribution of the average arrival interval of 7 h. (GOLIAS *et al.*, 2009b) formulate the discrete and dynamic berth allocation problem as a multiobjective combinatorial optimization problem. The wharf is divided into a number of berths and the vessels are assigned to a preferential group according to the arrival time. For each group there is an objective function, minimizing the total service time. And there is also an objective function minimizing the total service time of all vessels. A genetic algorithm based heuristic was presented to solve the proposed problem. In (CHEONG *et al.*, 2010) the BAP was solved with a multi-objective evolutionary algorithm (MOEA) by optimizing the berth schedule. The objectives studied were: minimize the makespan, waiting time, and degree of deviation from a predetermined priority schedule, all representing the interests of both port and vessels operators.

2.2 Relationship between the Berth Allocation Problem and other Combinatorial Optimization Problems

There are several approaches to model the BAP. The most relevant ones are presented in more detail in this section.

2.2.1 The Berth Allocation Problem as a Strip Packing Problem

The continuous Berth Allocation problem can be modeled as a strip packing problem.

Other problems in the literature have already been modeled as a packing problem or a cutting stock problem. (TRIGOS; LÓPEZ, 2017) adapt a specialized case of the classical one-dimensional cutting stock problem (1D-CSP) with six main additional features to model and solve planning unit operations with limited resources in the make-to-order industrial environment. The objective is to satisfy demand using the minimum number of manufacturing cycles at the vulcanizing operation during the manufacturing of rubber curved hoses in the automotive industry

Based on the model for the container loading problem proposed by (CHEN *et al.*, 1995), (MARTIN *et al.*, 2015) approached the BAP by mixed integer linear programming interpreting the problem as a special case of two-dimensional cutting stock problem.

Let $\mathcal{N} = \{1, ..., n\}$ be the set of vessels and $\mathcal{M} = \{1, ..., m\}$ the set of berths. For each $i \in \mathcal{N}$, p_i is the processing time and a_i the arrival time of vessel i. The continuous variable x_i indicates the handling start time of vessel i and the integer variable y_i indicates to which berth vessel i was allocated. Other binary variables fix the relative position of two vessels: $l_{ij} = 1$ if vessel i is on the left side of vessel j, and 0 otherwise; $r_{ij} = 1$ if vessel i is on the right side of vessel j, and 0 otherwise; $b_{ij} = 1$ if vessel i is behind vessel j, and 0 otherwise, and $f_{ij} = 1$ if vessel i in front of vessel j, and 0 otherwise. M is a large constant.

Thus, the BAP is formulated as follows:

$$\text{Min } \sum_i (x_i - a_i) \tag{2.1}$$

$$\text{s.t. } x_i + p_i \le x_j + (1 - l_{ij})M \qquad \forall (i,j)|(i < j) \tag{2.2}$$

$$x_j + p_j \le x_i + (1 - r_{ij})M \qquad \forall (i,j)|(i < j) \tag{2.3}$$

$$y_i + 1 \le y_j + (1 - b_{ij})M \qquad \forall (i,j)|(i < j) \tag{2.4}$$

$$y_j + 1 \le y_i + (1 - f_{ij})M \qquad \forall (i,j)|(i < j) \tag{2.5}$$

$$l_{ij} + r_{ij} + b_{ij} + f_{ij} \ge 1 \qquad \forall (i,j)|(i < j) \tag{2.6}$$

$$x_i \ge a_i \qquad \forall i \tag{2.7}$$

$$1 \le y_i \le m \qquad \forall i \tag{2.8}$$

$$l_{ij}, r_{ij}, b_{ij}, f_{ij} \in \{0, 1\} \qquad \forall (i,j) \tag{2.9}$$

$$x_i \ge 0 \qquad \forall i \tag{2.10}$$

$$y_i \ge 0 \text{ and integer} \qquad \forall i \tag{2.11}$$

The objective function (2.1) minimizes the sum of the waiting times of all vessels. Constraints (2.2) and (2.3) ensure that vessels do not overlap in the x axis (time axis) and constraints (2.4) and (2.5) ensure that vessels do not overlap in the y axis (space axis). In constraint (2.6), the check for overlap is necessary only if two vessels are placed in the same berth. Constraint (2.7) does not allow the vessel to be moored before its arrival time. Constraint (2.8) indicates which berth the vessel was allocated. Constraints (2.9), (2.10) and (2.11) specify the nature of the decision variables used.

When approaching the BAP as a strip packing problem, a time-space diagram is used to represent it. (LEE *et al.*, 2010) depicted the solution for the continuous BAP where the allocation plan for each vessel is represented as a rectangle. In this model, the vessels are allowed to berth anywhere along the quay so as to sufficiently utilize the quay resource and vessel shifting is not considered. Two versions of Greedy Randomized Adaptive Search Procedure (GRASP) are developed to search for near optimal solutions, in order to minimize the sum of weighted turnaround time for each incoming vessel. In (DU *et al.*, 2015) the variables are defined in order to quantify the tidal impacts on seaside operations and suggest a cheap and applicable solution to this problem. The objective function minimizes the total departure delay of the vessels. The experiments are performed simulating the one-year seaside operations of a container terminal with typical data in a strong realistic sense generated to keep the problem size realistic and to maintain the validity of the experimental results.

In the time-space diagram presented in (GUAN; CHEUNG, 2004) and (LEE; CHEN, 2009), the horizontal axis represent the time units and the vertical axis represent the berth space. Accordingly, the vessel as a rectangle whose length is the processing time and whose height is the vessel size. In (GUAN; CHEUNG, 2004) multiple vessels are allowed mooring per berth and the vessel arrivals are grouped into batches (similar arrival time) in order to minimize the total weighted flow time. Two formulations are introduced - Relative Position Formulation (space-time diagram) and Position Assignment formulation (space covered by the vessel rectangles) - and a composite heuristic was developed to conduct numerical experiments. (LEE; CHEN, 2009) developed a neighborhood-search based heuristic, treating the quay as a continuous space in order to determine the berthing time and space for each incoming vessel.

In the time-space diagram presented in (GANJI *et al.*, 2010) and (DU *et al.*, 2011), the horizontal axis represents the position along the wharf, while the vertical one represents the time axis. Each vessel is represented by a rectangle, such that the length of the rectangle is the length of the vessel and the height of the rectangle is the duration of its handling time. (GANJI *et al.*, 2010) formulated the continuous BAP as a mixed integer nonlinear programming model, with the objective of minimizing the sum of the service times of all vessels. A genetic algorithm based heuristic is developed to search for a solution for the problem and two test problems, a small and a large-sized problem, were used to test the method. The results from the small test are also compared with the results obtained from the branch and bound algorithm. (DU *et al.*, 2011) also formulated the BAP as a mixed integer nonlinear programming (MINLP) model, whose nonlinear intractability is introduced by the consideration of fuel consumption , which is mainly determined by the sailing speed. The total departure delay of all vessels and the vessel emission is minimized.

(DAI *et al.*, 2008) solved the static version as a rectangle packing problem with arrival time constraints with arrival time constraints. The aim is to minimize the delays faced by vessels, with higher priority vessels receiving the promised level of services. A local search algorithm that employs the concept of sequence pair to define the neighborhood structure is proposed.

2.2.2 The Berth Allocation Problem as a Scheduling Problem

In the discrete case, the BAP can be modeled as an unrelated parallel machine-scheduling problem, where a vessel is treated as a job and a berth as a machine.

Most works on scheduling problem aim in minimizing the maximum completion time, the so-called makespan. (TELLACHE; BOUDHAR, 2017) proved that such problem is NP-hard in the strong sense. It was addressed the problem of scheduling a set of unit-time operations on a two-machine flow shop, subject to the constraints that some conflicting jobs cannot be scheduled simultaneously on different machines. Most scheduling problems aim at minimizing the makespan. (LI *et al.*, 2017) addressed the batch processing machines problem in order to make full use of machine capacity and to improve the processing efficiency. (LABBI *et al.*, 2017) considered the problem of scheduling a set of jobs on two identical and parallel machines with preparation constraints. Each job requires immediately before its execution a set of resources and a non-negligible preparation time, in which the machine is not available for another job. (SOTSKOV; GHOLAMI, 2017) addressed the job-shop problem: given a set of different machines, a set of jobs, consisting of a set of ordered operations and having its own machine route, must be processed on the machines. The classical job-shop problem has the objective of finding a schedule, which is feasible with respect to the resource and precedence constraints. (MASCHIETTO *et al.*, 2017) viewed cranes as parallel machines and trucks as jobs in order to formulate the crane scheduling problem to define the starting time for each loading operation of the pair coil-truck. Two cranes must load a sequence of trucks and are subject to non-interference constraints, as they move on the same track, and each truck has a loading demand. This kind of problem is common in several logistics centers, such as stockyards, depots and warehouses, where cranes or other similar equipment sharing the same rail or road are used for handling cargo. (KOUIDER *et al.*, 2017) considered the job shop scheduling problem with unit-time operations with a set of jobs to be processed on a set of machines. Each job consists of a specific set of operations which expresses a distinct processing route that has already been fixed and known in advance. Each operation has a unit processing time and can be executed by only one machine. Each machine can only handle at most one operation at a time and can be used at most once by each job. The job shop scheduling problem with unit-time operation is realized in many practical scheduling scenarios, as in scheduling lessons or exams at the university, in scheduling games in sport competitions when each game needs the same time, and in scheduling medical procedures in hospitals. (OZTURK *et al.*, 2017) investigated the parallel batch scheduling of unit size jobs with different processing times and release dates on parallel identical machines. Each machine can process multiple jobs simultaneously as long as the capacity constraint is not violated and the jobs processed at the same time constitute a single batch. If each batch can process a single job at a time the problem reduces to a classical scheduling problem in the presence of jobs with release dates, different processing times and parallel machines. Batch processing is encountered in casting, metallurgy, aircraft manufacturing, burn-in operations of integrated circuit and sterilization services of hospitals. (HAN *et al.*, 2016) solved the flow shop scheduling problem with blocking scheduling problem. A set of jobs must be processed on a set of machines and due to the lack of intermediate buffer storage between machines, a job remains in the current machine until the next machine is available for processing.

Besides minimizing the makespan, some problems present multiple objectives. Making an analogy with a scheduling problem, in (ZHU *et al.*, 2017)multitasking scheduling problems with a rate-modifying activity are studied. For this case, jobs denote tasks and machines denote the human worker and the set of jobs, independent and available at time zero, must be processed on a machine that can process only one job at a time. The objective is minimizing makespan, as well as the total completion time, maximum lateness, and due-date assignment related cost by determining when to schedule the rate modifying activity and the optimal task sequence in the presence of multitasking. (THEVENIN *et al.*, 2017) modeled a parallel machine scheduling problem with job incompatibility. Preemption can occur (i.e. a job can be stopped and restarted later), which is usually undesirable in production systems once it increases the throughput time of the jobs and the inventory costs. Therefore, the minimization of multiple objectives is considered, corresponding to the makespan, number of preemptions and summation of the jobs' throughput times.

(XU *et al.*, 2012) considered the Berth Allocation Problem as a parallel-machine scheduling problem, in which the assignment of vessels to berths is limited by water depth and tidal condition. There are n vessels and m berths and the time line $[0, 1)$ is divided into two intervals $[0, T]$ (low water period) and $[T, 1)$ (high-water period). For $i = 1, 2, \ldots, n$, vessel i has a given processing time $p_i > 0$, arrival time $a_i \geq 0$, weight $w_i > 0$, "high-water berth index" H_i, and "low-water berth index" L_i, where $H_i, L_i \in \{1, 2, \ldots\ldots, m\}$. For $k = 1, 2, \ldots, m$, let $z_i^k = 1$ if vessel i is assigned to berth k, and 0 otherwise. Let $I_{ii'}^k = 1$ if vessels i and i' are both assigned to berth k and vessel i is processed before vessel i', and 0 otherwise. Finally, let x_i be the start time of processing of vessel i and M a large constant.

$$\text{Min} \sum_{i=1}^{n} w_i(x_i + p_j - a_i) \tag{2.12}$$

$$\text{s.t.} \sum_{k=1}^{m} z_i^k = 1 \qquad i = 1, 2, \ldots, n \tag{2.13}$$

$$x_i \geq a_i \qquad i = 1, 2, \ldots, n \tag{2.14}$$

$$x_{i'} \geq x_i - M(1 - I_{ii'}^k) \qquad i, i' = 1, 2, \ldots, n \text{ s.t. } i \neq i'; k = 1, 2, \ldots, m \tag{2.15}$$

$$I_{ii'}^k + I_{i'i}^k \leq \frac{1}{2}\left(z_i^k + z_{i'}^k\right) \qquad i, i' = 1, 2, \ldots, n \text{ s.t. } i < i'; k = 1, 2, \ldots, m \tag{2.16}$$

$$I_{ii'}^k + I_{i'i}^k \leq z_i^k + z_{i'}^k - 1 \qquad i, i' = 1, 2, \ldots, n \text{ s.t. } i < i'; k = 1, 2, \ldots, m \tag{2.17}$$

$$z_i^k = 0 \qquad i = 1, 2, \ldots, n; k = 1, 2, \ldots, H_i - 1 \tag{2.18}$$

$$x_i \geq T z_i^k \qquad i = 1, 2, \ldots, n; k = 1, 2, \ldots, L_i - 1 \tag{2.19}$$

$$z_i^k \in \{0, 1\} \qquad i = 1, 2, \ldots, n; k = 1, 2, \ldots, m \tag{2.20}$$

$$I_{ii'}^k \in \{0, 1\} \qquad i, i' = 1, 2, \ldots, n \text{ s.t. } i \neq i' \tag{2.21}$$

The objective function (2.12) minimizes the sum of all vessels completion time. Constraint (2.13) requires each vessel to be assigned to one berth. Constraint (2.14) requires that each vessel can start its processing only after it has arrived at the terminal. Constraint (2.15) states that if vessels i, i' are both assigned to berth k and vessel i is processed before vessel i', then the start time of vessel i' must be no earlier than $x_i + p_i$. Constraints (2.16) and (2.17) ensure that one of $I_{ii'}^k$ and $I_{i'i}^k$ equals 1 if vessels i and i' are both assigned to berth k and that $I_{ii'}^k = I_{i'i}^k = 0$ if one of vessels i and i' is not assigned to berth k. Constraint (2.18) disallows vessel i from being assigned to berths $1, 2, \ldots, H_i - 1$ and Constraint (2.19) disallows vessel i from being processed by berths $1, 2, \ldots, L_i - 1$ during period $[0, T]$. Because the problem is computationally intractable, a simple heuristic solution methods is presented to obtain good solutions to the problem in an efficient time.

According to (SANCHES *et al.*, 2015), many scheduling problems are characterized by the large number of possible solutions. Therefore, an adaptive genetic algorithm is proposed for makespan minimization, once it has been successfully used as a search method to solve this problem due to its capacity of globally exploring the search space and finding good solutions quickly. (MONACO; SAMMARRA, 2007) studied the discrete and dynamic berth allocation problem, dealing with it as a scheduling problem as well. Therefore, they proposed a mixed integer model, which was strengthened by introducing idle time variables that do not depend on the vessel. After, the formulation was improved by defining idle times constraints stronger and last, a tighter version of this constraints was considered. All these three formulations are valid for the BAP, and the last one enjoys the property of a lower number of continuous variables and constraints. The constraints defining the idle time variables play the role of complicating constraints in a Lagrangean relaxation framework: dualizing them, the resulting Lagrangean problem is easy to solve. According to them, if the BAP is thought as a scheduling problem, the problem will be NP-hard independently of the number of berths because vessels have a release date and they are not allowed to berth before the expected arrival time (dynamic case). On the other hand, the static version (the arrival time does not impose a restriction on timing for mooring) of the same problem is solvable in polynomial time since it reduces to an assignment problem.

2.2.3 The Berth Allocation Problem as a Vehicle Routing Problem

Throughout literature, the BAP is modeled as a Vehicle Routing Problem (VRP) as well.

The (VRP) is a combinatorial optimization and integer programming problem which generalizes the well-known traveling salesman problem (TSP). According to (AN; YAN, 2016), modeling the problems as a TSP is advantageous as the presented special structure allows it to be solved by special algorithms.

The VRP has many applications in real world. Considering a special case of the vehicle routing problem where not only each customer has specified delivery time window, but each route has limited time duration - Vehicle Routing Problem with Time Windows and Limited Duration (VRPTW-LD), (KHODABANDEH *et al.*, 2017) modeled the distribution and transportation problem faced by General Electric Appliances & Lighting (GE). The customer locations, which have a pre-specified delivery time window, need carefully be paired together within one truck route and such pairing should not only consider the travel distances between customers but ensure the total demand for customers along the same route does not exceed a truck's capacity. Minimizing the total travel distances and the total number of required trucks will reduce their distribution cost significantly. (MAHVASH *et al.*, 2017) address the three-dimensional loading capacitated vehicle routing problem (3L-CVRP). There is a travelling cost associated to each edge. A fleet of homogenous vehicles is located in the central depot and each vehicle has a maximum weight capacity and a three-dimensional loading space of length, width and height. The demand of each customer is expressed in terms of a set of cuboid items and each item is characterized by length, width, height and fragility status. The 3L-CVRP aims at finding a set of vehicle routes with minimum total traveling cost.

(BUHRKAL *et al.*, 2011) defined the Berth Allocation Problem on a graph $G = (V, A)$ where the set $V = N \cup \{o, d\}$ contains a vertex for each vessel and vertices o and d that represents respectively the origin and destination nodes for any route in the graph. Thereby, the BAP was formulates as a heterogeneous vehicle routing problem with time windows (HVRPTW), in which berths corresponds to vehicles and there is single depot.

The set of arcs is a subset of $V \times V$. Let N be the set of vessels and M the set of berths. Each vessel $i \in N$ has an arrival time a_i, an expected departure time from the port b_i (which implies a time window $[a_i, b_i]$ for vessel i), processing times p_i^k that are dependent on the respective berth $k \in M$ locations and a relative importance v_i. For the origin and destination vertices, the time window depends on the berth k as berths can be available at different times $[s^k, e^k]$). Each binary decision variable l_{ij}^k, $k \in M$, $(i,j) \in A$, takes the value one if vessel j immediately succeeds vessel i at berth k and is zero otherwise. Each continuous variables x_i^k, $i \in V$, $k \in M$, gives the time that vessel i starts being serviced at berth k (if vessel i does not use berth k, $x_i^k = a_i$). The variables x_o^k and x_d^k define the start and end time of activities at berth $k \in M$ respectively. The problem is formulated as follow:

$$\min \sum_{i\in N}\sum_{k\in M} v_i \left(x_i^k + p_i^k \sum_{j\in N\cup\{d\}} l_{ij}^k \right) \tag{2.22}$$

$$\text{s.t.} \sum_{k\in M}\sum_{j\in N\cup\{d\}} l_{ij}^k = 1 \qquad \forall i \in N \tag{2.23}$$

$$\sum_{j\in N\cup\{d\}} l_{oj}^k = 1 \qquad \forall k \in M \tag{2.24}$$

$$\sum_{j\in N\cup\{o\}} l_{id}^k = 1 \qquad \forall k \in M \tag{2.25}$$

$$\sum_{j\in N\cup\{d\}} l_{ij}^k = \sum_{j\in N\cup\{o\}} l_{ji}^k \qquad \forall k \in M, i \in N \tag{2.26}$$

$$x_i^k + p_i^k - x_j^k \le \left(1 - l_{ij}^k\right) M_{ij}^k \qquad \forall k \in M, (i,j) \in A \tag{2.27}$$

$$a_i \le x_i^k \qquad \forall k \in M, i \in N \tag{2.28}$$

$$x_i^k + p_i^k \sum_{j\in N\cup\{d\}} l_{ij}^k \le b_i \qquad \forall k \in M, i \in N \tag{2.29}$$

$$s^k \le x_o^k \qquad \forall k \in M \tag{2.30}$$

$$x_d^k \le e^k \qquad \forall k \in M \tag{2.31}$$

$$l_{ij}^k \in (0,1) \qquad \forall k \in M, (i,j) \in A \tag{2.32}$$

$$x_i^k \ge 0 \qquad \forall k \in M, i \in V \tag{2.33}$$

The objective function (2.22) minimizes the weighted sum of vessel service times. Constraint set (2.23) states that each vessel must be assigned to exactly one berth k, while constraints (2.24) and (2.25) guarantee that for each berth k the degree of origin and destination nodes, respectively, is one. Constraints (2.26) ensure flow conservation for the remaining vertices. Constraints (2.27) guarantee consistency for berthing time and mooring sequence on each berth, and $M_{ij}^k = \max\{b_i + h_i^k - a_j, 0\}$. Constraints (2.28) and (2.29) enforce the time window requirements for each vessel. Constraints (2.30) and (2.31) enforce the berth availability time windows. Finally, constraints (2.32) and (2.33) define the respective domains of the decision variables.

Model (2.22)-(2.33) was solved in (TING *et al.*, 2014a) with a particle swarm optimization (PSO) algorithm.

(CORDEAU *et al.*, 2005) proposed a formulation on multidepot vehicle-routing problem with time windows: the vessels are seen as customers and the berths as depots at which one vehicle is located. There is one vehicle one for each depot and each vehicle starts and ends its tour at its depot, which is divided into an origin vertex and a destination vertex. The vessels are modeled as vertices in a multigraph and the time windows are imposed on every vertex. At the origin and destination vertices, the time windows correspond to the availability period of the corresponding berth. The objective function is the minimization of the weighted sum of the service times. In (OLIVEIRA *et al.*, 2012) the problem was treated as a Vehicle Routing Problem with Time Windows and Multiple Garages (VRPTWMG): the vessels are seen as customers and berths as garages. The objective function minimizes the weighted sum of service time. An application of a hybrid method known as Clustering Search (CS) is proposed, using a Simulated Annealing Algorithm to generate solutions.

2.2.4 The Berth Allocation Problem as a Generalized Set Partitioning Problem

The dynamic BAP has also been modeled as a Generalized Set Partitioning Problem (GSPP).

According to (VOSS; LALLA-RUIZ, 2016), the Set Partitioning Problem (SPP) is a well-known optimization problem because of its complexity and several real-world applications. They presented a reformulation for the Multiple-choice Multidimensional Knapsack Problem, which consists of finding a subset of objects that maximizes the total profit while observing some capacity restrictions, based on the SSP.

(BUHRKAL *et al.*, 2011), (LALLA-RUIZ *et al.*, 2012), (UMANG *et al.*, 2013) and (LALLA-RUIZ; VOSS, 2016) modeled the dynamic BAP (the quay is divided into a finite set of berths to which the vessels can be assigned for loading and unloading purposes) as a Generalized Set Partitioning Problem (GSPP).

Let M be the set of berthing locations, N the set of vessels, Ω the set of columns. Let A and B be two matrices, both containing $|\Omega|$ columns and c_ω the cost of any column ω. $A_{i\omega} = 1$ if column ω represents an assignment of vessel i, and 0 otherwise; $B_{p\omega} = 1$ if position p is contained in the assignment that column ω represents, and 0 otherwise and the rows of B are indexed by the set P, with $|P| = \sum_{k \in M} (e^k - s^k)$. The decision variable is $x_\omega = 1$ if column ω is used in the solution, and 0 otherwise. Thereby, the GSPP formulation for the BAP is presented below:

$$\min \sum_{\omega \in \Omega} c_\omega x_\omega \tag{2.34}$$

$$\text{s.t.} \sum_{\omega \in \Omega} A_{i\omega} x_\omega = 1 \qquad \forall i \in N \tag{2.35}$$

$$\sum_{\omega \in \Omega} B_{p\omega} x_\omega \leq 1 \qquad \forall p \in P \tag{2.36}$$

$$x_\omega \in \{0, 1\} \qquad \forall \omega \in \Omega \tag{2.37}$$

The objective function (2.34) minimizes the vessels service time. Constraints (2.35) guarantee that all vessels are served (generalized upper bound constraints) and constraints (2.36) guarantee that only one vessel can use any berth during each time interval.

(LALLA-RUIZ *et al.*, 2012) created a tabu search with path relinking is created for solving such problem. (UMANG *et al.*, 2013) proposed a squeaky wheel optimization (SWO) meta-heuristic to solve the problem for a small sample of data received from SAQR port, Ras Al Khaimah (RAK), UAE. (LALLA-RUIZ; VOSS, 2016) presented two Partial Optimization Metaheuristic Under Special Intensification Conditions (POPMUSIC) approaches to solve the problem.

2.3 Literature Review Summary

Tables 4 and 4 summarize the classification criteria proposed in this chapter.

paper	space			arrival		service time		integration			base model					approach		data		objective function		
	discrete	continuous	others	static	dynamic	static	dynamic	quay crane	yard management	none	Strip Packing	Scheduling	VRP	GSSP	others	heuristic based	mathematical programming	real port based	generated randomly	minimize service time	minimize makespan	others
(ARANGO *et al.*, 2013)	x				x		x	x							x	x		x				x
(ARANGO *et al.*, 2011)	x				x	x				x					x	x		x		x		
(BARROS *et al.*, 2011)	x				x	x				x					x	x		x		x		x
(BUHRKAL *et al.*, 2011)	x				x	x				x			x	x			x	x		x		
(CHANG *et al.*, 2010)		x			x		x	x							x	x		x				x
(CHEONG; TAN, 2008)		x			x	x				x					x	x			x	x		
(CHEONG *et al.*, 2010)			x		x	x				x					x	x			x		x	
(CORDEAU *et al.*, 2005)	x				x	x				x			x			x		x		x		
(DAI *et al.*, 2008)	x			x		x				x	x						x	x				x
(OLIVEIRA *et al.*, 2012)	x				x	x				x			x			x		x		x		
(DU *et al.*, 2015)		x		x		x				x	x						x		x			x
(DU *et al.*, 2011)		x			x	x				x	x						x	x				x
(FROJAN *et al.*, 2015)		x			x	x				x					x	x		x				x
(GIALLOMBARDO *et al.*, 2008)	x				x		x	x							x		x	x				x
(GIALLOMBARDO *et al.*, 2010)	x				x		x	x							x	x		x				x
(GOLIAS *et al.*, 2009b)	x				x	x				x					x	x			x	x		
(GOLIAS *et al.*, 2009a)	x				x	x				x					x	x			x			x
(GUAN; CHEUNG, 2004)		x			x	x				x	x					x			x			x
(HANSEN; OGUZ., 2003)		x		x	x	x				x					x	x			x	x		
(HANSEN *et al.*, 2008)	x				x	x				x					x	x			x	x		
(HENDRIKS *et al.*, 2013)		x		x		x			x						x	x		x				x
(IMAI *et al.*, 2008)	x				x		x	x							x	x			x	x		
(IMAI *et al.*, 2007)	x				x	x				x					x	x			x	x		
(IMAI *et al.*, 2001)	x				x	x				x					x	x			x	x		
(IMAI *et al.*, 2003)	x				x	x				x					x	x			x	x		
(IMAI *et al.*, 2008)	x	x			x	x				x					x	x		x		x		
(IMAI *et al.*, 2005)		x			x	x				x	x					x			x	x		
(IMAI *et al.*, 2007)	x				x	x				x					x	x			x			x
(LALLA-RUIZ *et al.*, 2014)	x				x	x				x					x	x			x			x
(LALLA-RUIZ *et al.*, 2012)	x				x	x				x				x		x		x		x		
(LALLA-RUIZ; VOSS, 2016)	x				x	x				x				x		x		x		x		
(LEE *et al.*, 2010)		x			x	x				x	x					x			x			x
(LEE; WANG, 2010)	x				x		x	x							x	x			x		x	
(LEE; CHEN, 2009)		x			x	x				x	x					x			x			x
(LI; PANG, 2011)	x				x	x				x					x		x		x			x
(LIANG *et al.*, 2009)	x				x		x	x							x	x		x		x		
(LIANG *et al.*, 2012)	x				x		x	x							x	x		x		x		
(MARTIN *et al.*, 2015)	x				x	x				x	x	x				x			x			x

Table 2 – Classification Summary (cont.)

	space			arrival		service time		integration			base model					approach		data		objective function		
paper	discrete	continuous	others	static	dynamic	static	dynamic	quay crane	yard management	none	Strip Packing	Scheduling	VRP	GSSP	others	heuristic based	mathematical programming	real port based	generated randomly	minimize service time	minimize makespan	others
(MEISEL; BIERWIRTH, 2013)		x			x		x	x							x	x			x	x		
(MONACO; SAMMARRA, 2007)	x				x	x				x					x	x			x		x	
(RAA *et al.*, 2011)		x			x		x	x							x		x	x				x
(ROBENEK *et al.*, 2014)		x			x	x			x						x		x	x		x		
(GANJI *et al.*, 2010)	x				x	x				x	x					x			x	x		
(SIMRIN; DIABAT, 2015)	x				x	x				x					x	x			x	x		
(THEOFANIS *et al.*, 2007)	x				x	x				x					x	x			x	x		
(TING *et al.*, 2014a)	x				x	x				x			x			x		x		x		
(TURKOGULLARI *et al.*, 2014)		x			x		x	x							x		x	x				x
(UMANG *et al.*, 2013)			x		x	x				x				x	x	x	x	x		x		
(VACCA *et al.*, 2013)	x			x			x	x							x		x	x				x
(XU *et al.*, 2012)	x			x	x	x				x		x				x			x		x	
(YANG *et al.*, 2012)		x			x		x	x							x	x			x	x		

Table 4 – Classification Summary

2.4 The Discrete Berth Allocation Model (DBAP)

The model presented in Section 2.2.3 is detailed in this section because it generalizes other models. Some improvements were made by (BUHRKAL *et al.*, 2011) for reducing computation time. First, a class of valid inequalities was formulated to increase the lower bound of the x_i^k variables:

$$s^k l_{oj}^k + \sum_{i \in N} \left(\max \left\{ a_i, s^k \right\} + p_i^k \right) l_{ij}^k \leq x_j^k \;\; \forall k \in M, j \in N$$

On the left hand side at most one of the l variables can be 1 in a feasible solution (constraints (2.23) and (2.26)), hence the inequality is valid no matter which one of the l variables on the left hand side is non-zero.

Second, a variable fixing was proposed. One can fix a variable l_{ij}^k if it's possible to guarantee that an optimal solution exists in which berth k does not use the arc (i, j).

If $p_i^k \geq p_j^k$ and $a_i \geq a_j$, then $l_{ij}^k = 0$. In words, if vessel j arrives before vessel i and has a shorter processing time, then vessel j may not follow vessel i at berth k.

Last, it was highlighted the data used may lead to equivalent solutions once some berths are identical in terms of their availability time window and handling times for all vessels. This problem is tackled by considering berth types instead individual berths: β_k, $k \in M$, represents the number of berths of type k in the problem instance. Constraints (2.24) and (2.25) and the domain of l_{od}^k need to be respectively modified to:

$$\sum_{j \in N \cup \{d\}} l_{oj}^k = \beta_k \;\; \forall k \in M \tag{2.38}$$

$$\sum_{j \in N \cup \{o\}} l_{id}^k = \beta_k \;\; \forall k \in M \tag{2.39}$$

$$l_{od}^k \in \{0, \ldots, \beta_k\} \tag{2.40}$$

3 Multiobjective Algorithm for the BAP

According to (OSYCZKA, 1985), a multiobjective optimization problem is the optimization of a vectorial function whose elements represent each one of the objective functions. The expected solution is composed of a family of solutions considered equal to each other and better than the remainder (partially ordered set of equilibrium points). Along these lines, there are feasible alternatives that do not satisfy any order relation, such as “less than or equal,” impeding the use of the usual concept of optimal solution adopted in mono-objective problems. Multiobjective optimization models better reflects the complex reality of maritime terminals and allows the exploration of a wider range of alternative solutions.

Most studies presented in the literature consider only one objective function: to minimize the costs of the port and the vessel. However, other optimization objectives may emerge, and sometimes it may conflict. In (YANG; WANG, 2010) a bi-objective optimization model was proposed to minimize the turnaround time of vessels and the production cost at the same time. The optimization models and associated techniques have evolved to contemplate the more realistic situation in which multiple objectives compete to solve a given problem (FERREIRA, 1999). (CHEONG *et al.*, 2009) attempt to minimize three objectives that represent the interests of both port and vessels operators: makespan of the port, total waiting time of the vessel and degree of deviation from a predetermined service priority schedule. In this context, to consider a multiobjective optimization may be profitable, where optimal decisions need to be taken in the presence of trade-offs between two or more conflicting objectives. The model proposed by (GOLIAS *et al.*, 2009a) assumes that vessels arriving at the terminal can be assigned to different preferential groups. There will be one objective function for the vessels of each preferential customer and one objective function for all the vessels of different customers that are non-preferential. A Genetic Algorithm heuristic with an integer chromosomal representation is used to exploit in full the characteristics of the problem, four different types of mutation are applied to all the chromosomes at each generation (insert, swap, inversion, and scramble mutation) and a novel multi-population and multi-selection approach was used to find quality solutions for each objective function. In (PRATAP *et al.*, 2015) a problem based on a realistic scenario of a port located in the eastern coast of India is formulated aiming to find the optimal order in which the ships should be allowed to pass through the channel towards the berths. The two objectives are minimizing the waiting time and the deviation from customer priority. A Modified Non-sorting Genetic Algorithm II, aiming to mantain the lateral diversity in the population of the next generation is proposed.

There are several mathematical programming techniques used for solving multiobjective optimization problems based on information that must be known or defined by the decision maker. These techniques tend to generate only one solution in each execution of the algorithm according to (COELLO; ZACATENCO, 2002). Alternatively, the metaheuristics have been widely used in several applications since they are able to obtain a set of good solutions in a single round. Based on the Non-dominated Sorting Genetic Algorithm II (NSGA-II) (DEB *et al.*, 2000; DEB *et al.*, 2002), this chapter proposes the use and adaptation of an Evolutionary Metaheuristic to solve the Berth Allocation Problem considering two objective functions to be simultaneously optimized. The aim of this study is to provide an understanding on how well evolutionary approaches can handle real world scenarios and compete against other operation research approaches. The genetic algorithm was chosen because it is the most classical evolutionary algorithm used in the literature, and its NSGA-II variation showed good results in several applications.

This chapter is organized as follows: the next section presents the mathematical formulation of the BAP and the multiple objectives considered. Section 3.2 introduces the proposed algorithm and describes how it can be applied to the BAP. Section 3.3 presents an application of the techniques in different studied cases; the results are presented in Section 3.4. Finally, Section 3.5 presents comments and conclusions.

3.1 Mathematical Formulation

In (BARBOSA, 2014) proposed a model based on the scheduling in parallel machines is proposed. The model considers the BAP in the dynamic and discrete case, and the processing time does not depend on the berth assignment. The parameters and variables involved in the model of BAP are the ones presented below:

$$
\begin{aligned}
p_i \quad & : \text{processing time of vessel } i \\
e_i \quad & : \text{arrival time of vessel } i \\
z_{ij} \quad & = \begin{cases} 1, & \text{if vessel } i \text{ is in berth } j. \\ 0, & \text{otherwise.} \end{cases} \\
a_{ik} \quad & = \begin{cases} 1, & \text{if vessel } i \text{ is on the left side of vessel } k. \\ 0, & \text{otherwise.} \end{cases} \\
x_i \quad & = \text{ handling starting time of vessel } i.
\end{aligned}
$$

Thus, the following problem is modeled:

$$\min \quad \sum_i (x_i - e_i) \tag{3.1}$$
$$\text{s.a} \quad \sum_j z_{ij} = 1 \ \forall i \tag{3.2}$$
$$z_{ij} + z_{kj} - a_{ik} - a_{ki} \leq 1 \ \forall i \neq k, j \tag{3.3}$$
$$x_i + p_i - (1 - a_{ik})M \leq x_k \ \forall i, k \tag{3.4}$$
$$z_{ij} + z_{kh} + a_{ik} + a_{ki} \leq 2 \ \forall i \neq k, j \neq h \tag{3.5}$$
$$x_i \geq e_i \ \forall i \tag{3.6}$$
$$x_i \geq 0 \ \forall i. \tag{3.7}$$
$$a_{i,k}, z_{i,j} \in \{0, 1\} \ \forall i, k, j \tag{3.8}$$

The objective function (3.1) minimizes the sum of the waiting times of all vessels. Constraint (3.2) ensures that each vessel will be allocated exactly to one berth. Constraint (3.3) shows that if the vessels i and k are in the same berth, then vessel or i is the right side of vessel k or the opposite occurs. The constraint (3.4) ensures that vessels do not overlap each other over time. Constraint (3.5) reinforces that variable $a_{i,k}$ exists only for vessels that are in the same berth. Finally, the constraint (3.6) does not allow the vessels to be moored before its arrival. The last two constraints specify the type of used variables.

3.1.1 Multiple Objectives

Based on (CHEONG *et al.*, 2010) we have selected two different conflicting objectives to the problem, which are described hereafter.

1. Minimizing the waiting time:

$$f_1 = \sum_i (x_i - e_i) \tag{3.9}$$

2. Minimizing the makespan:

$$f_2 = \max\{x_i + p_i\} \tag{3.10}$$

The waiting time (3.9) needs to be minimized from the vessel operator point of view. From the terminal operator point of view the concern is to minimize the makespan (3.10).

In a multiobjective optimization problem, the search space is partially ordained, in a way that two arbitrary solutions are linked to each other in two possible ways: or one of them dominates the other or neither dominates.

Let ω_1 and ω_2 two solutions in the search space of a problem that has m objective functions. Then ω_1 dominates ω_2 (SAWARAGI *et al.*, 1985), if and only if:

$$\forall i \in \{1,2,...,m\}, \qquad f_i(\omega_1) \leq f_i(\omega_2)$$

$$\text{and}$$

$$\exists j \in \{1,2,...,m\}, \qquad f_j(\omega_1) < f_j(\omega_2)$$

In other words, ω_1 is not worse than ω_2 in any of the objectives and is better in at least one of them.

Once the Berth Allocation Problem is formulated as a multiobjective optimization problem, the next section presents the algorithm that treats both objectives simultaneously and explicitly.

3.2 Multiobjective Algorithm: MOBAP

Evolutionary Metaheuristics are complex algorithms that provide adaptive, efficient and robust search engines. According to (KNOWLES *et al.*, 2008), these computational procedures for solving problems arise from the application of heuristic techniques by an iterative process, in which each iteration is called generation. It is based on the following sequence: performing reproduction with genetic inheritance, introduction of random variations, promoting competition and selection of individuals from a given population. In the selection stage, individuals with better fitness have higher chance of being chosen for reproduction. These selected individuals have a predefined probability of passing by the crossover process, in which part of the parents genes are combined to generate new individuals. After performing it, the new individuals can be mutated, with a predefined probability, to maintain a genetically diverse population. It is worth highlighting that the crossover probability is greater than the mutation probability.

We propose an algorithm MOBAP based on NSGA-II (DEB *et al.*, 2002) to solve the multiobjective BAP. The features of the proposed Evolutionary Algorithm have been implemented with the concern tof respecting the characteristics of the BAP problem.

3.2.1 Coding

An individual k is represented by coding a complete and feasible scheduling. The following structures will be used to represent each individual. Figure 2 exemplifies encoding a structure that represents an individual from the population with 2 berths and 10 vessels. A binary matriz represents the vessel/berth allocation, a berth list (LB) represents the vessels sequencing and a vector represents the vessels starting times. The algorithm was implemented in a generic way to take any amount of vessels and berths.

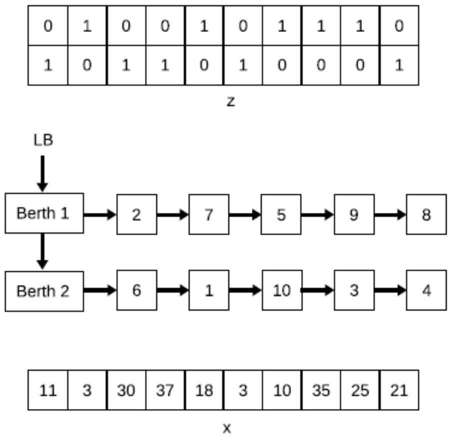


Figure 2 – Individual k.

3.2.2 Population Initialization

The population size P and the number of generations G have been empirically defined after a series of tuning experiments. The initialization of each individual k is done in a random way, that is, for each individual from the population and for each vessel a berth j is drawn randomly to be allocated. Thereby, the element z_{ij} from the representation structure of individual k is initialized with 1 (i.e., vessel i is allocated to berth j). From the initialization of this structure, for each individual k and for each berth a random list is created containing the vessel sequence. It is represented in a structure we named BL.

The initialization of variable x is fulfilled according to the BL structure, and the parameters e_i and p_i are also considered.

All individuals from the population are initialized in order to represent feasible solutions. After the population has been initialized, each individual k, $k = 1, \ldots, P$, is evaluated. That means that the objective functions f_1 and f_2 are calculated for each individual.

3.2.3 Population Evaluation

All individuals from the population are evaluated: for each one the objective values f_1 (sum of waiting times) and f_2 (makespan) according to equations (3.9) and (3.10) presented in Section 3.1.1. All operators have been implemented in order to always

Pseudocódigo 3.1 MOBAP

```
For all (j-berth) of BL list
{
    free = 0;
    For all (i-vessel) of (j-berth)
    {
        If e_i > free
        {
            (individual k).x_i = e_i ;
        }
        Else
        {
            ((individual k).x_i = free ;
        }
        free = (individual k).x_i + p_i;
    }
}
```

create feasible individuals, therefore it is not required to check the constraints during the treatment or evaluation procedure.

3.2.4 Nondominated Sorting Approach

As defined in (DEB *et al.*, 2002), in NSGA-II every individual k is associated with two attributes: $rank_k$ and $distance_k$.

If two solutions are in different nondomination levels (different nondominated frontiers), we prefer the solution k with the lower $rank_k$. Otherwise, if two k_1 and k_2 solutions belong to the same frontier ($rank_{k1} = rank_{k2}$), then we prefer the solution that is located in a less crowded region (that is, higher $distance_k$) (DEB *et al.*, 2002). Figures 3 and 4 illustrate these attributes.

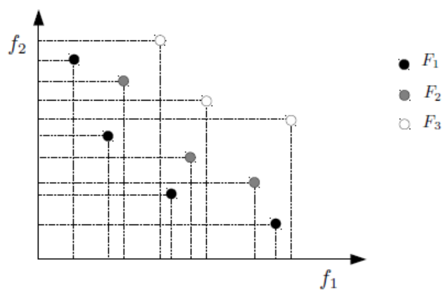


Figure 3 – Nondomination Rank, $rank_k$.

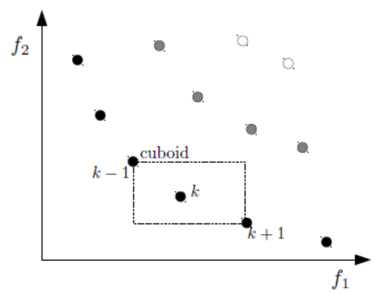


Figure 4 – Crowding distance, $distance$

3.2.5 Main Loop

All individuals from the current population, with $rank_k$ and $distance_k$, form a parent population P_g of P size. Selection, crossover and mutation operators are used to create an offspring population Q_g of $Psize$. A combined population $R_g = P_g \cup Q_g$, of $2P$ size, is sorted according to nondomination rank and crowding distance to choose exactly P population members to the new population P_{g+1} (DEB *et al.*, 2002).

The selection is done through the binary tournament selection algorithm. This algorithm randomly samples two solutions of P_g and compares them according to $rank_k$ and $distance_k$ (as described in Section 3.2.4). The best one is chosen for the following procedures (Figure 5). Pairs of selected solutions are randomly formed. These pairs can go through crossover and mutation to form an offspring population Q_g.

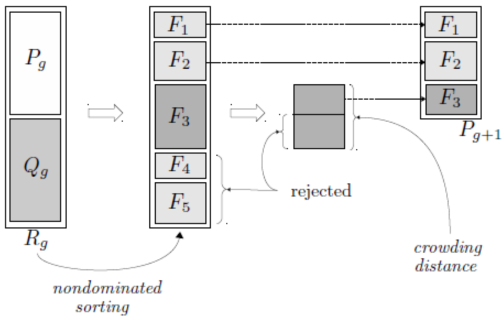


Figure 5 – Selection process for the composition of the next population (DEB *et al.*, 2002).

3.2.6 Crossover

A population of parents is built through binary tournament (two random solutions from the population compete according to $rank_k$ and $distance_k$ criteria, and the best one will belong to the parent population). Pairs of parents from this population are selected to generate pairs of offspring.

The crossover operator is applied to each pair of selected solutions with probability PC, thus generating two offspring as follows.

For 50% of pairs of parents and offspring:

$$
\begin{gathered}
\text{offspring1(berth1)} = \text{parent1(berth1)} \\
\text{offspring1(berth3)} = \text{parent1(berth3)} \\
\vdots \\
\text{offspring2(berth1)} = \text{parent2(berth1)} \\
\text{offspring2(berth3)} = \text{parent2(berth3)} \\
\vdots
\end{gathered}
$$

In other words, the odd berths are copied, and the even berths are completed:

offspring1(berth2) = parent2(berth2), whereas the vessels are not allocated to offspring1

$$\vdots$$

offspring2(berth2) = parent1(berth2), whereas the vessels are not allocated to offspring2

$$\vdots$$

For the other 50%, the crossover is performed in the reverse way:

$$\begin{gathered}
\text{offspring1(berth2)} = \text{parent1(berth2)} \\
\text{offspring1(berth4)} = \text{parent1(berth4)} \\
\vdots \\
\text{offspring2(berth2)} = \text{parent2(berth2)} \\
\text{offspring2(berth4)} = \text{parent2(berth4)} \\
\vdots
\end{gathered}$$

The odd berths are completed following the rules that ensure the feasibility maintenance in the offspring population.

By the end of the crossover, if there still exists vessels not assigned to any berth, they are sequenced according to arrival time (ascending order) and inserted in the berths with the smallest amount of vessels.

Because of this procedure of copying a portion of berths, each offspring will have characteristics inherited from both parents. The reintegration of lost vessels will guarantee the genetic variability necessary in an evolutionary algorithm.

3.2.7 Mutation

The mutation process complements the crossover. It is applied to the offspring and it allows a larger search space to be explored. For each individual from the offspring, according to the probability of mutation, two vessels are randomly selected:

1. if the two vessels are allocated to the same berth, on the solution represented by the offspring only the vessels allocation order is swapped, and the structures x and BL are updated.

2. if the two vessels are allocated to different berths, the berth allocation is swapped, and representation structures z, x and BL are updated.

The offspring population is evaluated by calculating the objectives f_1 and f_2. The operators of classification and agglomeration are evaluated, considering the merge between both the original and the offspring population. The process is repeated with the new population until the total number of generations is attained.

3.3 Experiments

The implementation of the proposed algorithm is based on the adaptation algorithm (DEB *et al.*, 2002). It was written in C Language and executed on an Intel Core i5 1.80GHz Processor, 4Gb RAM. The proposed model was solved with CPLEX. The stopping criteria chosen was computational time, limited in 1 hour.

Computational tests were performed with four BAP instances sizes: 10, 20, 30 and 40 vessels and 2 berths.

For these, as in (BARBOSA, 2014), the vessels processing times were generated by a binomial distribution with 16 trials and $p = 0.5$ and the arrival times were generated based on a uniform distribution in $[0, 25]$.

The binomial distribution is a discrete probability distribution of the number of successes in a sequence of n independent experiments such as yes / no questions, each with a probability of success p. Its probability function is:

$$P(X = k) = \binom{n}{k} p^k (1-p)^{n-k}. \tag{3.11}$$

The uniform distribution is a continuous probability distribution where the probability of generating any point in an interval contained in the sample space is proportional to the size of the interval $[\alpha, \beta]$. Its probability density function is given by:

$$f(x) = \begin{cases} \frac{1}{\beta-\alpha}, & \text{for } \alpha \leq x \leq \beta \\ 0, & \text{otherwise.} \end{cases} \tag{3.12}$$

3.4 Results

The considered mutation and crossover rates have been empirically determined: $PM = 0.1$ and $PC = 0.9$, respectively. The population size was set to $P = 100$ individuals for all instances. The number of generations was $G = 100$ for all instances.

Because of the stochastic nature of evolutionary algorithms it is necessary to perform several test rounds to validate the results. For each study scenario, 10 test rounds were performed.

Tables 5 and 6 present the results from CPLEX for all tests. The results are in the following order:

- column a: minimizing the wainting time (f_1)
- column b: makespan associated with the solution when minimizing the wainting time

- column c: minimizing the makespan (f_2)
- column d: waiting time associated with the solution when minimizing the makespan

For the instances with 10 vessels, the computational time to reach the optimal solution is in seconds on column e.

For the instances with 20, 30 and 40 vessels, the execution time was 1 hour, and optimality was not reached. In this case, the quality of the solution is measured through the GAP, which is a solver output and calculated as:

$$GAP = \frac{UB - LB}{UB} \tag{3.13}$$

where UB is the best upper bound and LB is the best lower bound from CPLEX.

In average, the GAP for these instances was: 52% (20 vessels), 71% (30 vessels) and 79% (40 vessels).

instance	10 vessels					20 vessels			
	a	b	c	d	e	a	b	c	d
1	82	50	47	99	15.94	404	106	77	477
2	51	47	45	56	4.06	434	80	80	517
3	59	42	41	69	3.47	523	89	88	572
4	56	46	45	69	4.63	465	81	80	509
5	55	49	46	67	4.87	450	81	78	542
6	80	52	51	83	25.28	433	80	78	484
7	17	36	35	29	0.48	308	79	78	407
8	77	50	47	83	16.78	367	74	74	420
9	87	45	45	103	20.03	377	81	80	460
10	55	43	42	76	5.53	380	79	77	451

Table 5 – CPLEX Results - 10 and 20 vessels

instance	30 vessels				40 vessels			
	a	b	c	d	a	b	c	d
1	1320	131	129	1547	2091	166	157	2325
2	999	118	119	1263	2356	169	166	2598
3	1168	137	124	1365	2192	172	160	2587
4	1235	134	127	1428	2117	163	161	2608
5	1260	141	127	1394	2276	180	165	2624
6	1200	128	127	1354	2193	175	158	2567
7	1043	121	120	1245	2257	179	165	2622
8	1116	126	123	1326	2111	163	160	2678
9	1169	121	120	1355	2268	170	160	2547
10	976	108	107	1235	2170	168	160	2617

Table 6 – CPLEX Results - 30 and 40 vessels

Table 7 shows the results from the multiobjective algorithm proposed. For each instance 10 test rounds were executed. The average computational time of each round was: 0.436 seconds (10 vessels), 0.613 seconds (20 vessels), 0.726 seconds (30 vessels) and 0.788 seconds (40 vessels). Each round produces a set of solutions. The best extreme solutions among all rounds were chosen for comparison. For example, for instance 1 with 10 vessels, a single solution represents the extreme of both objectives with values $f_1 = 82$ and $f_2 = 47$. On the other hand, for instance 2 with 10 vessels, one point represents the extreme solution that minimized f_1 ($f_1 = 51$ and $f_2 = 46$) and a different point represents the extreme solution that minimized f_2 ($f_1 = 52$ and $f_2 = 45$).

For 10 vessels, the multiobjective algorithm was able to obtain the same minimum values for f_1 e f_2 obtained from CPLEX through a single point as well as two extreme points. It is noteworthy that CPLEX guaranteed optimality of the solution without exceeding the stop criteria, however the multiobjective algorithm achieved the same solutions in a significantly shorter computational time (Table 5, column e). For instances with 20, 30 and 40 vessels, the entries with * on Table 7 refer to solutions in which the multiobjective algorithm obtained better results than CPLEX. The entries with ** refers to solutions in which multiobjective algorithm and CPLEX broke even. It is noteworthy that the computational time spent by the multiobjective algorithm to obtain those solutions is considerably smaller, making this approach more advantageous.

instance	10 vessels		20 vessels		30 vessels		40 vessels	
	waiting time	makespan	waiting time	makespan	waiting time	makespan	waiting time	makespan
1	82	47	399*	77**	1315*	129**	2101	157**
							2093	160
2	51	46	433*	80**	1009	119**	2346*	168
	52	45			1011	118*	2356	166**
3	59	42	520*	88**	1158*	127	2177*	161
	60	41			1160*	124**	2178*	160**
4	56	45	465**	80**	1232*	127**	2111*	162
							2112*	160*
5	55	46	449*	78**	1229*	128	2272*	165**
					1231*	126*	2274*	164*
6	80	51	430*	78**	1203	128	2193**	158**
					1271	127**	2226	157*
7	17	35	311	79	1054	123	2249*	167
			313	78**	1061	119*	2279	164*
8	77	47	365*	74**	1122	123**	2143	161
							2145	160*
9	87	45	377**	80**	1159*	120**	2277	161
							2278	160**
10	55	42	378*	77**	980	108	2182	161
					984	107**	2189	160**

Table 7 – Multiobjective Algorithm Results - Best Extreme Solutions.

To better analyze the obtained solutions, one instance with 10 vessels and one with 40 vessels were chosen. The results will be presented in the next sections.

3.4.1 10 vessels analysis

This instance was chosen once the multiobjective algorithm reached the optimal solutions obtained by CPLEX ($f_1 = 82$ and $f_2 = 47$) in a single extreme point.

When the waiting time was minimized with CPLEX, with $f_1 = 82$, the makespan associated had value $f_2 = 50$. This solution is outlined in Figure 6a.

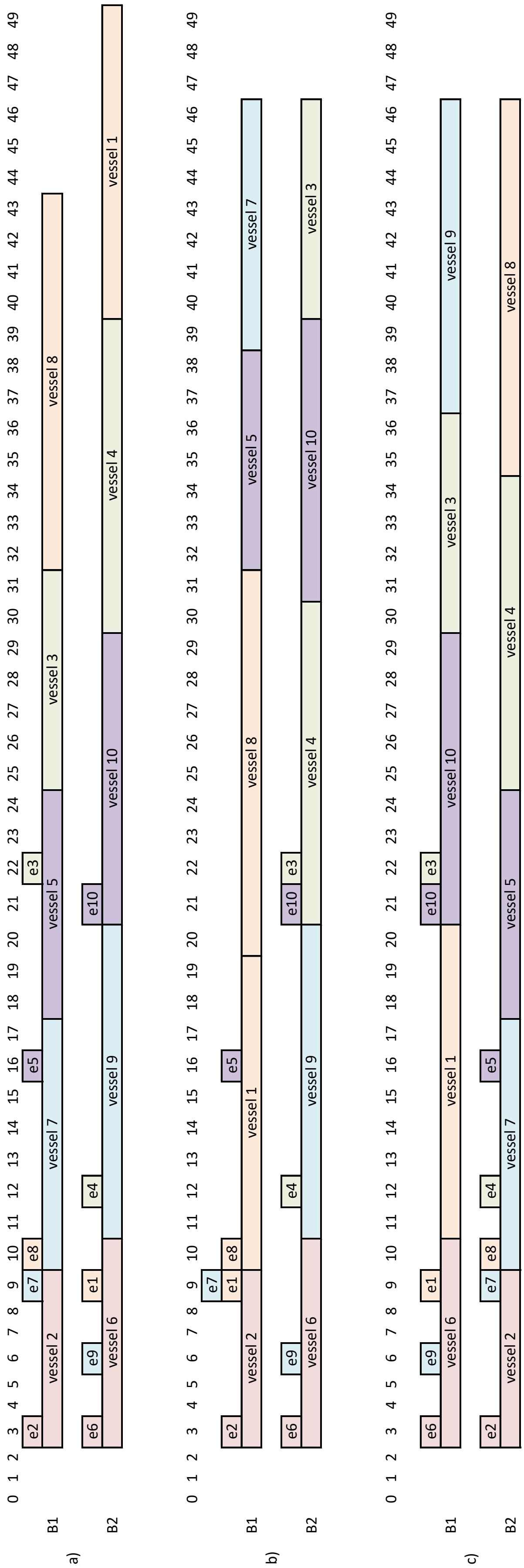


Figure 6 – Results for minimizing the waiting time.

When the makespan was minimized with CPLEX, with $f_2 = 47$, the associated waiting time was $f_1 = 99$, as shown in Figure 6b.

The multiobjective algorithm obtained those same values in the objective space ($f_1 = 82$ e $f_2 = 47$) in 5 out of 10 test rounds, referring to different values (x_i and z_{ij}) in the solution space.

Figure 6c shows the arrangement of the vessels/berths in one of those 5 solutions.

In Figures 6a,b,c it is easy to note that the solution in which CPLEX minimized the waiting time, the associated makespan had the highest value.

For a better understanding of the waiting time obtained by the 3 solutions presented in Figures 6a,b,c, let us analyze the graphics in Figure 7.

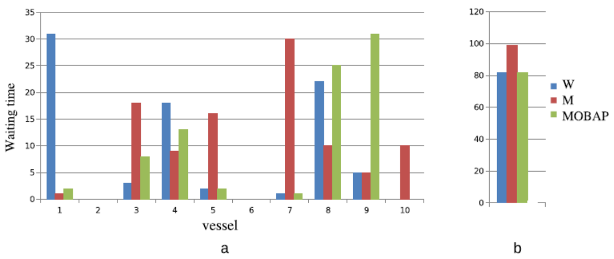


Figure 7 – Waiting time comparison.

The graphic in Figure 7a shows the waiting time results for each vessel in each one of the approaches (minimizing the waiting time with CPLEX (W), minimizing the makespan with CPLEX (M) and the multiobjective algorithm(MOBAP)). It is observed in Figure 7b that when CPLEX minimizes the makespan, the associated waiting time is the worst one obtained. The multiobjective algorithm was able to obtain simultaneously the best result among the ones presented by CPLEX when minimizing the waiting time and the makespan disconnectedly.

Even though the two chosen objectives (minimizing the waiting time and the makespan) do not appear to be conflicting, the multiobjective algorithm proves to be appropriate and advantageous due to this behavior presented by the solutions resulting from the simultaneous optimization of multiple objectives.

3.4.2 40 vessels analysis

For 40 vessels, the instance number 5 was chosen for a more detailed analysis once it presented extreme solutions ($f_1 = 2272$ with $f_2 = 165$ and $f_1 = 2274$ with $f_2 = 164$) that dominate the solutions founded with CPLEX ($f_1 = 2276$ with $f_2 = 165$ highlighted in Figure 10) (Table 7).

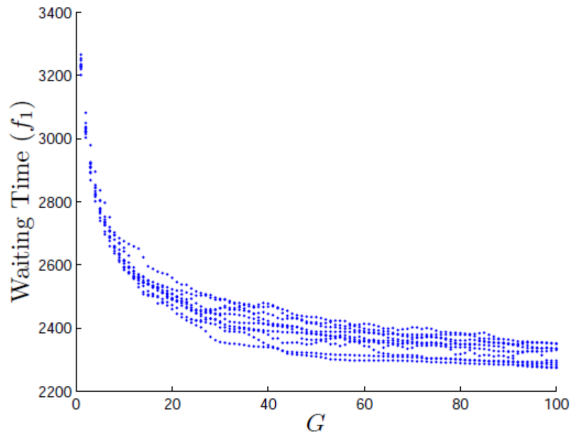


Figure 8 – f_1 average.

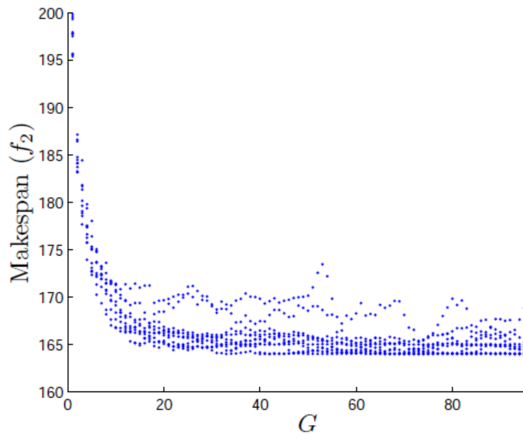


Figure 9 – f_2 average.

Figures 8 and 9 show the evolution of objective functions f_1 and f_2, from the initial generation (random population) to the final generation ($G = 100$) for all the test rounds of the instance number 5 with 40 vessels.

It is easy to confirm the effectiveness of the evolutionary process, which begins with a random solution and derive significant improvements over the generations, a behavior verified in every round. It is important to emphasize that the number of generations could be increased in order to improve the quality of the obtained results. The number of generations $G = 100$ was empirically defined through test rounds: until the solution obtained by the multiobjective algorithm in most instances was better than the one obtained by CPLEX. Higher values for G (number of generations) and P (population size) may lead to better results. Empirically, it is observed that the increase in the number of individuals has more influence on the increase in computational time of execution rather than on the increase in the number of generations.

Figure 10 shows the nondominated frontiers (solutions k with $rank_k = 1$ in the last generation) of all test rounds for this instance. Two non dominated solutions are highlighted among those obtained:

$$f_1 = 2272 \text{ and } f_2 = 165$$
$$f_1 = 2274 \text{ and } f_2 = 164$$

Both solutions are superior in relation to the other solutions, including the ones from CPLEX, and are indifferent to each other, in the sense that they have the same quality

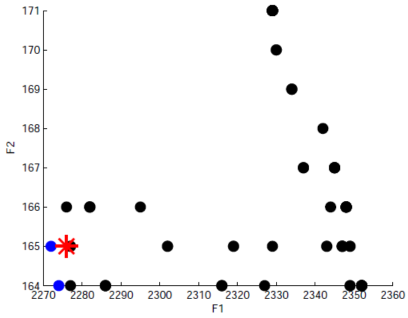


Figure 10 – Nondominated frontiers.

3.5 Conclusion

In this Chapter we proposed the adaptation and application of NSGA-II to the BAP problem. The proposed approach has shown great potential in solving this problem by working with a set of solutions called population, and optimizing each solution in parallel. This characteristic makes it possible to obtain distinct solutions with good quality. Different solutions correspond to alternative planning types obtained in a single round of the algorithm. A decision maker is responsible for the final choice.

The results obtained with the algorithm were compared with the ones presented in (BARBOSA, 2014) towards f_1 (minimizing the sum of waiting times) and amended by the results towards f_2 (minimizing the makespan). For instances with 40 vessels, the multiobjective evolutionary algorithm clearly stood out, both for quality of the solutions obtained and low computational time. Thus, the approach proposed here proved to be competitive and effective for large instances.

Although a latent conflict between the two objective functions chosen was not identified, the multiobjective approach was important to obtain a solution that represents the minimum values for the waiting time and the makespan. This behavior was not observed in the CPLEX results in which each objective was optimized separately. Therefore, the importance of the simultaneous optimization of multiple objectives in the Operational Research is emphasized.

The algorithm showed alternative solutions with good quality and relatively low computational cost. The results obtained encourage the application of the proposed algorithms to more complex subsystems.

4 The BAP as a Maximum Flow Problem: a lower bound approach

It is possible to propose different models for the BAP and to propose different methodologies for its treatment or optimization.

The scale and nature of this problem at large terminals often makes it impossible for the decisions made to be optimal. It is a combinatorial problem of the NP-Hard class, as indicated by(IMAI *et al.*, 2008). Taking advantage of the BAP formulation as a scheduling jobs in parallel machines (Chapter 3), in this chapter we propose two adaptations for a maximal flow algorithm. For this approach, the next step is the proposition of a constructive heuristic with local search to obtain feasible solutions for the BAP. The aim is to verify if the implementation of the constructive heuristic, coupled to the maximum flow algorithm, would result in an algorithm capable of finding good lower bounds for evolutionary metaheuristics.

(KURZ; ASKIN, 2004) created three lower bounds in order to evaluate the heuristics developed for scheduling in flexible flow lines with sequence-dependent setup times to minimize makespan. Such problem is NP-hard, due to the fact that we need not only sequence jobs on machines, we must consider which jobs are to be assigned to the machines. Therefore, we will use the maximal flow algorithm to generate a lower bound to evaluate the multiobjective algorithm proposed for the BAP in Chapter 3

Section 4.1 presents a model for the maximum flow problem. Section 4.2 shows how an arbitrary schedule with preemptions can be transformed into a nonpreemptive schedule without increasing the value of the objective function. Section 4.3 presents a maximal flow algorithm based on finding breakthrough paths with net positive flow between the source and sink nodes. Section 4.4 presents the preliminary results of the exploration of the different formulations and proposals.

4.1 A mathematical model for maximum flow problem

(BRUCKER, 2006) discussed the idea of formulating a maximum flow problem to solve the problem of scheduling jobs in parallel machines presented in Chapter 3.

Consider n jobs with processing times p_i to be scheduled in m identical parallel machines. Each job i has a release time r_i and a due time d_i. The goal is to find a scheduling of these jobs, allowing preemption, so that they are processed within their respective time windows $[r_i, d_i]$ and the maximum lateness $\max_{i=1}^{n} \{C_i - d_i\}$ is minimized, where $C_i = x_i + p_i$. Such a problem is reduced to maximum flow problem in a network.

The time windows $\{r_1, r_2, \ldots, r_n\}$, $\{d_1, d_2, \ldots, d_n\}$ are ordered in a sequence $t_1 < t_2 < \ldots t_r$. Then, define the intervals:

$$I_k := [t_k, t_{k+1}] \tag{4.1}$$

of length

$$T_k = t_{k+1} - t_k. \tag{4.2}$$

We construct a oriented graph with a set (CT) of nodes for the jobs and a set (CI) of nodes for the intervals, and a source node o and a destination node d as shown in Figure 11:

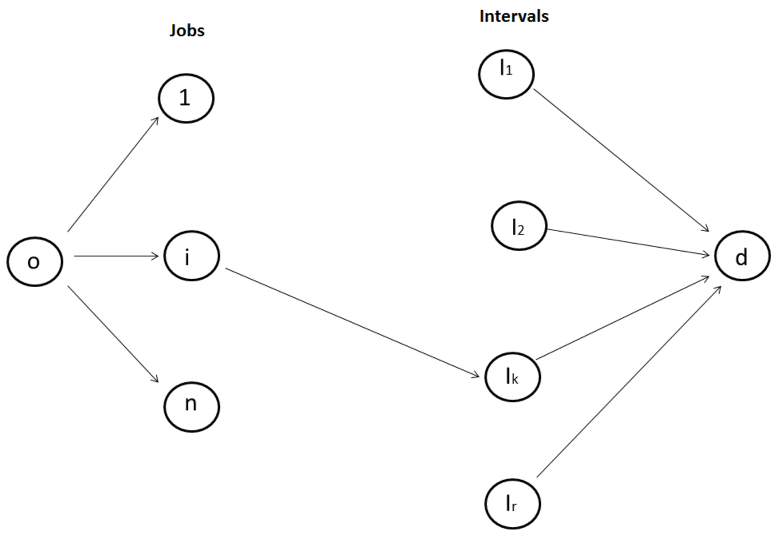


Figure 11 – Flow model.

From o we have an arc to each job vertex i with capacity p_i. From job i to interval I_k there exists if $r_i \leq t_k$ and $d_i \geq t_{k+1}$ (i.e., job i can be processed in interval I_k) with capacity T_k. From each interval vertex I_k we have an arc to d with capacity mT_k.

Let:

- y_i be the flow from node o to job i

- $w_{i,k}$ be the flow from job i to interval I_k, corresponding with the time period in which job i is processed in the time interval I_k

- z_k: the flow from interval I_k to node d

We used parameter f_{ik} to indicate if there exists an arc from job i to interval I_k. In addition, the flow passing through z_k was penalized: P takes larger values for the first intervals, forcing these first intervals to be occupied and the makespan be smaller.

Thus, we formulate the following problem:

$$\max \quad \sum_{i,k}(f_{ik}w_{ik} + y_i + Pz_k) \tag{4.3}$$

$$\text{s.t.} \quad y_i - \sum_k f_{ik}w_{ik} = 0 \ \forall i \tag{4.4}$$

$$\sum_i f_{ik}w_{ik} - z_k = 0 \ \forall k \tag{4.5}$$

$$w_{ik} \leq f_{ik}T_k \ \forall i,k \tag{4.6}$$

$$y_i \leq p_i \ \forall i \tag{4.7}$$

$$z_k \leq mT_k \ \forall k \tag{4.8}$$

$$y_i \geq 0 \ \forall i \tag{4.9}$$

$$z_k \geq 0 \ \forall k \tag{4.10}$$

$$w_{ik} \geq 0 \ \forall i,k. \tag{4.11}$$

The objective function (4.3) maximizes the network flow. (BRUCKER, 2006) states that there exists a schedule respecting all time windows if and only if the maximum flow has the value $\sum_i p_i$. Constraints (4.4) balance the network flow on each node of job i. Constraints (4.5) balance the network flow on each node of interval I_k. Constraints (4.6), (4.7) and (4.8) ensure that the flows will not exceed the arcs capacities.

4.2 Preemption correctness for Maximum Flow Problem

The model presentd in Section 4.1 allows preemption, and thus the solution presented by it must be treated with a constructive heuristic in order to obtain a feasible solution to the original problem (without interruption). Let us illustrate how such a modeling is done for a small scenario with only 5 vessels:

vessel i	p_i	e_i
1	10	9
2	7	3
3	7	22
4	11	12
5	8	16

Table 8 – Data for a 5 vessels sample

from the scheduling model presented in Chapter 3, we have the following solution:

vessel i	start service time	waiting time
1	9	0
2	3	0
3	23	1
4	12	0
5	19	3

Table 9 – Vessel allocation sample

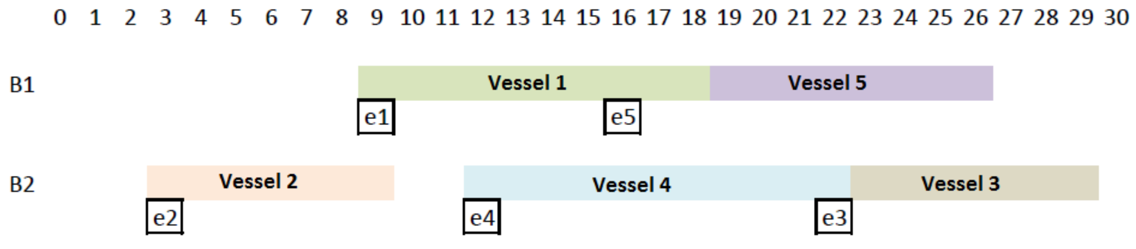


Figure 12 – Scheme for the vessel allocation sample.

Firstly, we need to define the time windows for each vessel. The vessels have an arrival time, however the problem we have worked so far does not restrict the time departure. So, let us set this time limit for each vessel i:

$$d_i = e_{last} + \sum_{k \neq i} p_k$$

where e_{last} is the time that the last vessel arrives at the port.

That is:

$$\begin{aligned}
d_1 &= 22 + (7+7+11+8) = 55 && \text{time window 1: } [9, 55] \\
d_2 &= 22 + (10+7+11+8) = 58 && \text{time window 2: } [3, 58] \\
d_3 &= 22 + (10+7+11+8) = 58 && \text{time window 3: } [22, 58] \\
d_4 &= 22 + (10+7+7+8) = 54 && \text{time window 4: } [12, 54] \\
d_5 &= 22 + (10+7+7+11) = 57 && \text{time window 5: } [16, 57]
\end{aligned}$$

To facilitate the understanding of the problem, consider unit time intervals:

$$I_k = [k, k+1]$$

This modeling with unit intervals increases the size of the graph. On the other hand, modeling with larger size intervals has the disadvantage of not allowing to identify the beginning and the end of processing within the respective interval.

Therefore, there are 58 intervals and:

- vessel 1: Can be serviced from I_9 to I_{55}
- vessel 2: Can be serviced from I_3 to I_{58}
- vessel 3: Can be serviced from I_{22} to I_{58}
- vessel 4: Can be serviced from I_{12} to I_{54}
- vessel 5: Can be serviced from I_{16} to I_{57}

Analyzing the time intervals we see that in the interval I_9 vessels 1 an 2 are allocated, therefore, they are in different berths. The same occurs in the intervals:

- I_{19}, I_{20} and I_{21} for vessels 4 and 5
- I_{22} for vessels 3 and 4
- I_{23}, I_{24}, I_{25}, I_{26} and $I_{27}$for vessels 3 and 5

The processing of vessel 5 was preempted in the interval I_{21}: this scheduling problem modeling in parallel machines allows preemption according to (BRUCKER, 2006). Processing job i at some machine is stopped and either continued at time s on a different machine or at some time $s' > s$ on the same or a different machine. In this case, such a solution is a lower bound for the original problem. From this preempted solution, it is necessary to apply a constructive heuristic to obtain a solution without preemption, processing feasible to the original problem.

For vessel 5, let $5a$ be the first part of its processing and $5b$ the second part. Figure 13 shows the following allocation:

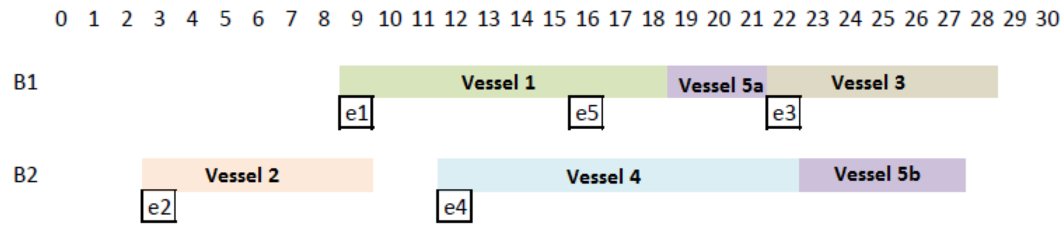


Figure 13 – Allocation of the maximum flow model.

In this case, the vessel 5 has its processing stopped at I_{21} in berth 1, and continued at I_{23} in berth 2. (BRUCKER, 2006) proposes the following correction when this happens: if a vessel i processed on machine k is preempted at time t and continued at time t on a different machine $\bar{k}$, then we may interchange the schedule after time t on k and the schedule after t on $\bar{k}$.

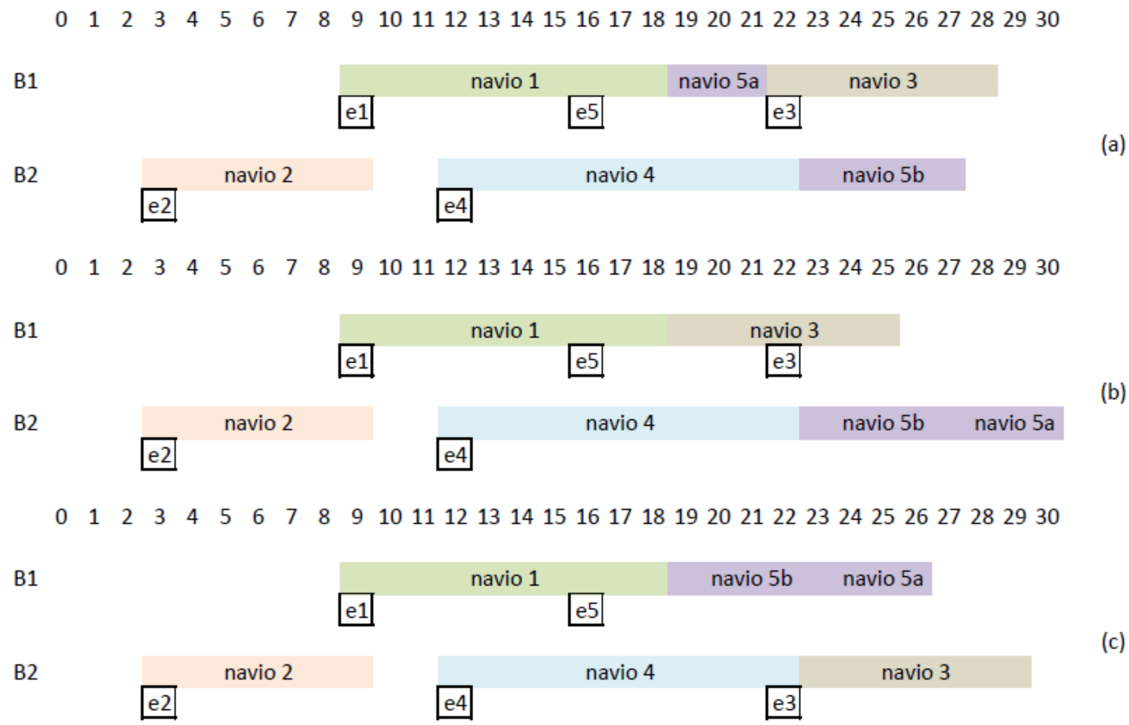


Figure 14 – Preemption correctness.

In Figure 14a vessel 5 is scheduled to be processed in berth 1 from time 19 to time 22 and in berth 2 from time 23 to 28. In Figure 14b vessel 5 is no longer allocated in berth 1 after time 22, but in berth 2 from time 23 to time 31. All vessels after time 22 at 1 moved ahead 3 units of time. Vessels scheduled after time 28 at berth 2 are delayed 3 units of time (in this case there are no vessels being processed after time 28). Then, as can be seen in Figure 14c,the schedule in berth 1 after time 22 are interchanged with the schedule in berth 2 after time 23, considering the vessels time windows. Thus, the solution we obtain is the same as in Figure 12.

4.3 An algorithm for the maximum flow problem

Alternatively to the resolution with CPLEX of the maximum flow model, in order to try to obtain a breakthrough path preemption, we propose an heuristic adapted from the algorithm proposed in (TAHA, 2007). An arc (i, j) has initial capacity $\bar{C_{ij}}$. As portions of these capacities are compromised by passing flow in the arc, the residual capacities (remaining capacities) are updated.If a node j receives flow from node i, it is labeled $[a_i, j]$, where a_j represents the flow from i to j.

1. For all arcs, set the residual capacity equal to the initial capacity: $c_{ij} = \bar{C_{ij}}$.
   Let $a_0 = \infty$ and label source node with $[\infty, -]$.
   Set $i = 0$ (source)

2. Determine S_i (set of unlabeled nodes j that can be reached directly from node i by arcs with positive residuals, $c_{ij} > 0$).
   If $S_i \neq 0$, go to step (3).
   Otherwise, go to step (4).

3. Determine $k \in S_i$ such that:
   If $i \neq 0$, then $c_{ik} = max_{j \in S_i} cij$
   If $i = 0$, then $p_k = max_{j \in S_i} p_j$
   Set $a_k = c_{ik}$ and label node k with $[a_k, j]$.
   If $k = n$, the sink node has been labeled and a breakthrough is found. Go to step (5).
   Otherwise, set $i = k$ go to step (2).

4. (Backtracking)
   If $i = 0$ no breakthrough is possible, go to step (6).
   Otherwise, let r be the node that has been labeled immediately before current node i and remove i from the set of nodes adjacent to r. Set $i = r$ and go to step (2).

5. (Determination of residuals)
   Define the nodes of pth breakthrough path from source node to node n:
   $N_p = (0, k_1, k_2, \ldots, n)$.

The maximum flow along the path is computed as:
$f_p = \min\{a_0, a_{k_1}, a_{k_2}, \ldots, a_n\}$
The residual capacity of each arc along the breakthrough path is decreased:
$c_{ij} = c_{ij} - f_p$
Reinstate any nodes that were removed in step (4). Set $i = 0$ and return to (2) to attempt a new breakthrough path.

6. (Solution)
(a) Given that m breakthrough paths have been determined, the maximum flow in the network is $F = f_1 + \ldots + f_m$
(b) Using the initial and final residuals of arc (i, j), the optimal flow is $x_{ij} = \bar{C}_{ij} - c_{ij}$

Step 3 of the original algorithm (TAHA, 2007) has been modified to restrict that, from the moment a vessel begins to process, the priority is that it finishes being processed before the next one is allocated. Such a modification reduces interruptions, as empirically observed. In step 3, also were analyzed the results by doing $p_k = min_{j \in S_i} p_j$ when $i = 0$. In Tables 10, 11, 12 and 13 we present the results obtained by minimizing the makespan and the respective times that the algorithm took to obtain those values. As a rule, the larger values for the makespan indicate solutions with fewer interruptions (closer to feasibility).

4.4 Computational experiments

In Tables 10, 11, 12 and 13 are presented the values obtained by minimizing the makespan (model in Section 3.1) and respective times that the algorithm took to obtain those values. As a rule, the larger values for the makespan indicate solutions with fewer interruptions (closer to feasibility).

Instance	(TAHA, 2007)		$p_k = max_{j \in S_i} p_j$		$p_k = min_{j \in S_i} p_j$	
	makespan	computational time (s)	makespan	computational time (s)	makespan	computational time (s
1	47	0.051583	48	0.049238	53	0.045458
2	45	0.038818	46	0.032639	47	0.047986
3	41	0.034417	41	0.028553	43	0.033299
4	45	0.040403	46	0.033565	48	0.040767
5	46	0.035522	47	0.040245	50	0.034327
6	51	0.038210	53	0.039768	54	0.042726
7	34	0.015100	35	0.016001	37	0.021011
8	47	0.041774	47	0.041607	49	0.043530
9	45	0.036371	46	0.040963	47	0.041647
10	42	0.031159	42	0.033909	45	0.037252

Table 10 – Flow algorithm for 10 vessels

Instance	(TAHA, 2007)		$p_k = max_{j \in S_i} p_j$		$p_k = min_{j \in S_i} p_j$	
	makespan	computational time (s)	makespan	computational time (s)	makespan	computational time (s)
1	77	0.432904	77	0.428412	79	0.399065
2	80	0.448326	80	0.437686	83	0.396158
3	88	0.547730	88	0.561812	91	0.489982
4	80	0.493009	81	0.497862	85	0.431385
5	78	0.454743	78	0.430506	80	0.387819
6	78	0.366842	79	0.359688	79	0.332487
7	78	0.411342	80	0.401321	81	0.362026
8	74	0.368888	75	0.354042	76	0.327231
9	80	0.450174	81	0.445028	82	0.413670
10	77	0.399154	78	0.397467	81	0.359463

Table 11 – Flow algorithm for 20 vessels

Instance	(TAHA, 2007)		$p_k = max_{j \in S_i} p_j$		$p_k = min_{j \in S_i} p_j$	
	makespan	computational time (s)	makespan	computational time (s)	makespan	computational time (s)
1	129	2.821822	131	2.773858	129	2.430886
2	118	2.248714	119	2.116039	121	1.821490
3	124	2.595374	124	2.572755	125	2.240254
4	127	2.801944	127	2.872677	130	2.427452
5	126	2.857407	128	2.897290	130	2.409831
6	127	2.761277	127	2.727478	128	2.388749
7	119	2.370978	120	2.408710	125	1.987783
8	123	2.464689	123	2.471099	127	2.100020
9	120	2.405109	121	2.384222	120	2.043243
10	107	1.713227	107	1.733995	108	1.501651

Table 12 – Flow algorithm for 30 vessels

Instance	(TAHA, 2007)		$p_k = max_{j \in S_i} p_j$		$p_k = min_{j \in S_i} p_j$	
	makespan	computational time (s)	makespan	computational time (s)	makespan	computational time (s)
1	157	7.153311	157	7.033389	160	5.965235
2	166	8.462691	166	8.177985	170	6.976584
3	160	7.660688	160	7.488536	162	6.242032
4	160	7.093711	162	6.942491	163	7.134168
5	164	8.380155	166	8.052578	165	6.725363
6	157	6.991960	158	6.875074	159	5.876637
7	164	8.277310	164	8.091182	168	6.808482
8	160	7.596789	161	7.459877	163	6.369196
9	160	7.443280	162	7.363338	162	6.218110
10	160	7.314320	160	7.244242	162	6.311438

Table 13 – Flow algorithm for 40 vessels

Table 14 summarizes the comparisons between the lower bound and optimal solution of the problems solved to optimality (instances with 10 vessels); between the lower bound and the best value founded by MOBAP and between the values returned by CPLEX, when CPLEX was given an one hour CPU limit time.

	Instance	Lower bound	Heuristic solution	Best bound	Best integer
10 vessels	1	47	47	40	47
	2	45	45	32	45
	3	41	41	28	41
	4	45	45	43	45
	5	46	46	39	46
	6	51	51	49	51
	7	34	35	33	35
	8	47	47	44	47
	9	45	45	41	45
	10	42	42	38	42
20 vessels	1	77	77	40	77
	2	80	80	41	80
	3	88	88	44	88
	4	80	80	45	80
	5	78	78	40,7826	78
	6	78	78	40,333	78
	7	78	78	51,1538	78
	8	74	74	41	74
	9	80	80	39	80
	10	77	77	38	77
30 vessels	1	129	129	37,7601	130
	2	118	119	36,6434	118
	3	124	124	42	124
	4	127	127	35,7333	127
	5	126	126	36	126
	6	127	127	38	128
	7	119	119	49	119
	8	123	123	36,8668	123
	9	120	120	42	120
	10	107	107	34	108
40 vessels	1	157	157	34,7476	158
	2	166	166	40,333	166
	3	160	160	40	162
	4	160	160	40	163
	5	164	165	38	166
	6	157	158	36	159
	7	164	164	39	166
	8	160	160	42	160
	9	160	160	37	160
	10	160	160	37	160

Table 14 – Lower bound effectiveness

As in (KURZ; ASKIN, 2004), we consider "Loss" as the (makespan - lower bound)/lower bound. In Table 15 reports it.

The makespan are the ones from Table 7 (Chapter 3) and the lower bound are from Tables 10, 11, 12 and 13.

Instance	10 vessels	20 vessels	30 vessels	40 vessels
1	0	0	0	0
	-	-	-	0,01910828
2	0,022222222	0	0,008474576	0,012048193
	0	-	0	0
3	0,024390244	0	0,024193548	0,00625
	0	-	0	0
4	0	0	0	0,0125
	-	-	-	0
5	0	0	0,015873016	0,006097561
	-	-	0	0
6	0	0	0,007874016	0,006369427
	-	-	0	0
7	0,029411765	0,012820513	0,033613445	0,018292683
	-	0	0	0
8	0	0	0	0,00625
	-	-	-	0
9	0	0	0	0,00625
	-	-	-	0
10	0	0	0,009345794	0,00625
	-	-	0	0

Table 15 – “Loss” statics for the MOBAP

It is noted that MOBAP found many solutions very close to the lower bound.

5 A Benders Decomposition Algorithm for the Berth Allocation Problem

Benders Decomposition is a solution method used for solving large-scale mixed integer linear programming problems. It can be described as a divide-and-conquer strategy: in each iteration, new constraints are added to the problem, making it progress towards a solution. The variables of the original problem are divided into two subsets. A first-stage master problem is solved over the first set of variables. Once these variables are fixed, the values for the second set of variables are determined in a second-stage subproblem. The resulting subproblem is a continuous linear program and the standard duality theory can be used to develop cuts.

According to (RAHMANIANI *et al.*, 2017), for more than five decades the Benders Decomposition algorithm has been used to tackle problems in many fields. Computational approaches based on Benders Decomposition to the constrained minimum break problem are proposed in (RASMUSSEN; TRICK, 2007). In (MERCIER *et al.*, 2005) a Benders Decomposition approach for the generalized formulation of the integrated aircraft routing and crew scheduling was implemented. A Benders-like decomposition approach was proposed in (CARAMIA; MARI, 2016) for solving a capacitated facility location problem with two decision makers. Exact solution algorithms based on Benders decomposition are presented in (HUANG; ZHENG, 2015) for the traveling salesman problem with risk constraints. This Chapter develops a Benders Decomposition approach for the Berth Allocation Problem (BAP).

This Chapter is organized as follows. Section 5.1 gives a description of the Benders Decomposition algorithm and its enhancements. Section 5.2 details the Benders Decomposition algorithm applied to the BAP. Section 5.3 reports the results.

5.1 Benders Decomposition

Benders Decomposition is a cutting plane method which reduces the search region by adding linear constraints while preserving the original feasible region.

Suppose a mixed integer linear problem of the form:

$$\min c^T x + f^T y \tag{5.1}$$

$$\text{s.t. } Ax + By \geq b \tag{5.2}$$

$$y \in Y \tag{5.3}$$

$$x \geq 0 \tag{5.4}$$

If (5.1) - (5.4) is an easier optimization problem in x when y is fixed, y are refered as "complicating variables" in (GEOFFRION, 1972).

With y fixed to a feasible integer configuration $\bar{y}$, the resulting model to be solved is given by:

$$\min\ c^t x \tag{5.5}$$

$$\text{s.t. } Ax \geq b - B\bar{y} \tag{5.6}$$

$$x \geq 0 \tag{5.7}$$

with the associate dual problem:

$$\max \quad (b - B\bar{y})^T u \tag{5.8}$$

$$\text{s.t} \quad A^T u \leq c \tag{5.9}$$

$$u \geq 0 \tag{5.10}$$

Defining z as the objective function of (5.5)-(5.7) and $\bar{u}$ as the variable values of the dual problem (5.8)-(5.10), the valid inequality

$$z \geq (b - By)^T \bar{u} \tag{5.11}$$

is a *Benders optimality cut*. In each iteration of the Benders algorithm, a master problem is solved:

$$\min \quad z \tag{5.12}$$

$$\text{s.t} \quad z \geq (b - By)^T \bar{u} \tag{5.13}$$

$$z \in R \tag{5.14}$$

$$y \in Y \tag{5.15}$$

whose solution $\bar{y}$ is the master problem solution and will be used to define the following subproblem (5.5) - (5.7).

If the subproblem (primal problem) is infeasible for a fixed $\bar{y}$, the dual formulation is unbounded. In this case, it is necessary to add a feasibility cut. Let $\bar{\alpha}$ be the extreme ray of the dual formulation. The *Benders feasibility cut*

$$\bar{\alpha}^T (b - By) \leq 0 \tag{5.16}$$

is formulated and added to the master problem in order to eliminate the infeasible solution.

It is noteworthy to mention that the master problem gives a lower bound (LB) and the subproblem gives an upper bound (UB) for the original problem. The procedure iterates until $UB - LB < \epsilon$.

Some different enhancement strategies may be proposed to improve and accelerate the convergence of the Benders Decomposition method, most of them taking into account the special characteristics of each problem. The two most important ones are presented below.

5.1.1 Combinatorial Benders Cut

The Benders Decomposition can also be used as an alternative to the "big-M" approach ("either/or" constraints). (CODATO; FISCHETTI, 2006) proposed and computationally analyzed an automatic problem reformulation for mixed integer linear problems involving logical implications modeled through big-M coefficients:

$$y_{j(i)} = 1 \Rightarrow a_i^T x \geq b_i \tag{5.17}$$

which usually are modeled as:

$$a_i^T x \geq b_i - (1 - y_{j(i)}) * M \tag{5.18}$$

Due to fact that M is a big number, the linear relaxation of the mixed integer linear problem model is poor and the resulting Benders cuts are weak and still depend on the big-M values. Therefore, the classical Benders approach can be viewed as a tool to speed-up the solution of the LP relaxation. The aim of this approach is to remove the model dependency on the big-M coefficients.

The master problem is solved to integrality. If this problem turns out to be infeasible, then the original problem also is. Otherwise, let y^* be an optimal solution. If the subproblem is infeasible for this solution, a Minimal Infeasible Subsystem C is sought, i.e., any inclusion-minimal set of row-indices of system (5.17) such that the linear subsystem:

$$a_i^T x \geq b_i \tag{5.19}$$

has no feasible solution x.

These implication constraints are modeled through the following Combinatorial Benders' (CB) cuts:

$$\sum_{j \in C : y^*_{j(i)} = 0} y_j + \sum_{j \in C : y^*_{j(i)} = 1} (1 - y_j) \geq 1 \tag{5.20}$$

CB cuts of this type are generated in correspondence to a given infeasible solution y^*, and added to the master problem.

5.1.2 Optimality Cut Disaggregation

If the Benders subproblem can be separated into independent subproblems, disaggregated cuts can be obtained in order to accelerate convergence of the Benders Decomposition algorithm. The subproblems are solved in parallel and multiple cuts formed by the dual optimal solutions are added to the Benders master problem simultaneously.

Each subproblem k generates an optimality cut, analogous to (5.11). According to (TANG *et al.*, 2013), these cuts include the same information as the primal Benders cut and restrict the solution space of the master problem in a more accurate-exact way.

5.2 Decomposition approach for the Berth Allocation Problem

There are several models the BAP. For being a more complete and robust model, this Chapter will treat the BAP as in (CORDEAU *et al.*, 2005), (BUHRKAL *et al.*, 2011) and (TING *et al.*, 2014b). The model was detailed in Chapter 2 (Sections 2.2.3 and 2.4). It is a model for the discrete and dynamic berth allocation problem based on a heterogeneous vehicle routing problem with time windows (HVRPTW), in which berths correspond to vehicles and there is a single depot.

According to (MONACO; SAMMARRA, 2007), the computational complexity of the BAP lies in the dynamic arrival process of the vessels. If the vessels have a release date and they are not allowed to berth before the expected arrival time, the problem is NP-hard. On the other hand, if the arrival time does not impose a restriction on timing for mooring, the problem reduces to an assignment problem, solvable in polynomial time. Considering the former case, the following decomposition is suggested. The master problem is an assignment problem and contains only the binary variables l_{ij}^k. The master problem is solved to optimality and the solution is sent to the subproblem. If the subproblem is infeasible, a feasibility cut is generated. If the subproblem is feasible, an optimality check is performed based on the Fundamental Theorem of Duality, as in the traditional Benders Decomposition.

Master problem:

$$\min \sum_{i \in N} \sum_{k \in M} \left(p_i^k \sum_{j \in N \cup \{d\}} l_{ij}^k \right)$$
s.t. (2.23)-(2.26), (2.32)

Subproblem:

$$\min \sum_{i \in N} \sum_{k \in M} \left(x_i^k \right)$$
s.t. (2.27)-(2.31), (2.33), (2.38)

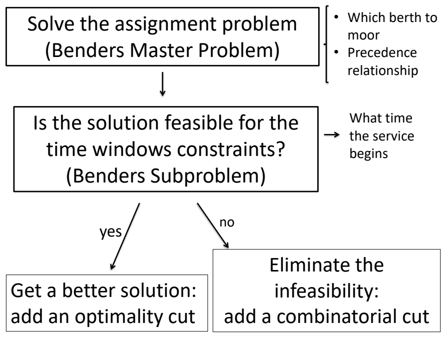


Figure 15 – Decomposition scheme.

5.2.1 Master problem improvement

In order to try to reduce the number of infeasible subproblems, we incorporate in the master problem some information about the subproblem constraints. The following constraints are added to the master problem: if the processing time of all vessels being serviced at berth k is considered, it needs to respect the time window of the berth:

$$\max \left\{ s^k, \min_{i \in N} \{a_i\} \right\} + \sum_{i \in N} \sum_{j \in N \cup \{d\}} p_i^k l_{ij}^k \leq \min \left\{ e^k, \max_{i \in N} \{b_i\} \right\} \ \forall k \in M \tag{5.21}$$

5.2.2 Multi-cut approach

When constraints (2.27) are infeasible, it means that the solution has a subtour. And this does not depend on the berth to cause infeasibility: if a given sequence of vessels is a subtour at berth k, the same sequence is a subtour at any other berth. Therefore, a subtour elimination constraint for all berths may be added:

$$\sum_{i,j \in C : l_{ij}^k = 1} \left(1 - l_{ij}^k \right) \geq 1 \ \forall k \in M \tag{5.22}$$

These constraints are the "The Subtour Formulation" proposed by (PATAKI, 2003) for the Traveling Salesman Problem. If the subproblem infeasibility is in constraints (2.27), the subtour elimination constraints (5.22) are incorporated in the master problem.

5.2.3 Subproblem Disaggregation

After the master problem is solved, the list of vessels allocated to each berth is known, being the only subproblem task to decide the time each vessel must be berthed.

Therefore, it can be separated in $|M|$ disconnected subproblems, each subproblem generating an optimality cut for each berth $k \in M$.

5.3 Computational Results

The implemented Benders Decomposition algorithm follows the strategy presented in (VATSA; JAYASWAL, 2016). It uses an incumbent solution in the branch-and-bound search tree to be passed to the sub-problem for Benders cut generation and the master problem is solved to optimality only once. Callbacks are used to intervene in the branch-and-bound tree search process and add the Benders cuts generated to the master problem as lazy constraints[1]. This method has the advantage that, by using a single search tree, a node is never revisited and a truly superior solution is never overlooked.

All the procedures described in Section 5.2 were implemented in C++ on an Intel Xeon Core processor model E5-W2687 3.10GHz with 128GB RAM and IBM Ilog CPLEX 12.6. CPLEX and the decomposition approach were given an one hour CPU time limit. The set of instances I2 from (CORDEAU *et al.*, 2005) were used in the computational experiments. The results are presented in detail in Table 16. In the columns are the size of the instances: 25 vessels and 5 berths (25x05), 25 vessels and 7 berths (25x07), 25 vessels and 10 berths (25x10), 35 vessels and 7 berths (35x07) and 35 vessels and 10 berths (35x10). Ten instances were tested for each problem size. For the results from CPLEX, it is shown the GAP and the objective function value returned after one hour of execution. From the decomposition approach, it is shown the objective function values for the master problem and for the subproblem returned after one hour of execution. The corresponding GAP is calculated.

Only in 5 instances, out of 50, Benders Decomposition was able to outperform the monolithic model solved by CPLEX. However, interesting enough, these are among the largest instances tested: 35 vessels and 7 berths and 35 vessels and 10 berths.

It is noteworthy that for the BAP model (2.22)-(2.33), the Benders subproblem may present multiple optimal solutions, because the objective function (2.22) is a linear combination of the constraints (2.28) and (2.29). It generates weak optimality cuts, and for this reason the decomposition progresses slowly. Moreover, the dual solutions are degenerate and the Pareto optimal cuts (according to (MAGNANTI; WONG, 1981) if the primal subproblem is degenerate it is possible to select the dual solution that is the closest to the interior of the master problem polyhedron to produce stronger cuts) can not be used to improve Benders Decomposition performance.

[1] constraints unlikely to be violated, and in consequence, applied only as necessary or not before needed

		25x05	25x07	25x10	35x07	35x10
1	GAP:	0.0005	optimal	0.0113	0.0663	0.0577
	Objective:	6559	10088	11998	15012	21308
	master (Lower Bound):	6308	9932	11727	13754	19831
	sub (Upper Bound):	6666	10763	12418	15298	21496
	GAP	0.05370	0.07720	0.05564	0.10092	0.07745
2	GAP:	0.02880	optimal	optimal	0.08232	0.07904
	Objective:	7882	12086	11693	17577	19332
	master (Lower Bound):	7404	11922	11526	15834	17585
	sub (Upper Bound):	8610	12635	12246	17760	18874
	GAP	0.1400	0.0564	0.05879	0.1084	0.0682
3	GAP:	0.0331	optimal	0.0090	0.09692	0.0198
	Objective:	6914	9807	13661	15651	19810
	master (Lower Bound):	6447	9572	13391	13878	19190
	sub (Upper Bound):	7327	10559	14023	15932	20915
	GAP	0.1201	0.0934	0.0450	0.1289	0.0824
4	GAP:	0.0094	0.0012	0.0170	0.0582	0.0645
	Objective:	8843	9984	16696	16247	21843
	master (Lower Bound):	8597	9799	16223	15028	20226
	sub (Upper Bound):	9376	11050	17096	16950	21890
	GAP	0.0830	0.1132	0.0510	0.1133	0.0760
5	GAP:	0.0120	optimal	0.0068	0.1384	0.0408
	Objective:	7598	10763	11897	18538	20108
	master (Lower Bound):	7235	10577	11623	15692	19070
	sub (Upper Bound):	8234	11683	12790	17474	20365
	GAP	0.1213	0.0946	0.0912	0.1019	0.0635
6	GAP:	0.0342	0.0080	0.0032	0.1748	0.0919
	Objective:	7444	12434	14120	17277	19717
	master (Lower Bound):	6856	12137	13941	13968	17645
	sub (Upper Bound):	7908	13378	14575	15725	19570
	GAP	0.1330	0.0927	0.0434	0.1117	0.0983
7	GAP:	0.0009	0.0157	0.0004	0.0658	0.0554
	Objective:	7751	13218	14913	17706	18106
	master (Lower Bound):	7463	12854	14785	16215	16812
	sub (Upper Bound):	8534	13972	15528	17956	18894
	GAP	0.1254	0.08001	0.0478	0.0969	0.1101
8	GAP:	optimal	0.0299	0.0013	0.0744	0.0347
	Objective:	7789	10478	14498	17067	19957
	master (Lower Bound):	7601	10458	14289	15509	18931
	sub (Upper Bound):	8554	11618	14866	17642	20651
	GAP	0.1114	0.0998	0.0388	0.1209	0.0832
9	GAP:	optimal	0.0215	0.0025	0.0675	0.1266
	Objective:	8556	10884	14776	18039	15408
	master (Lower Bound):	8318	9982	14599	16560	13144
	sub (Upper Bound):	9239	11189	15339	18526	14967
	GAP	0.0996	0.1078	0.0482	0.1061	0.1218
10	GAP:	0.0335	0.0226	0.0059	0.0359	0.0454
	Objective:	8579	12580	15150	16700	19973
	master (Lower Bound):	8032	12072	14896	15846	18779
	sub (Upper Bound):	9055	13268	15689	17868	20456
	GAP	0.1129	0.0901	0.0505	0.1131	0.0819

Table 16 – Comparison between CPLEX and Benders Decomposition

5.4 Conclusion

The Benders Decomposition is a cutting plane method described as a divide-and-conquer strategy because in each iteration new constraints are added to the problem. A model for the Berth Allocation Problem as Heterogeneous Vehicle Routing Problem with Time Windows was presented in this Chapter and a Benders Decomposition approach was proposed for the BAP, where several cuts were proposed and implemented. The combinatorial Benders cuts (5.1.1) were applied to reformulate constraints (2.27) and eliminate infeasibility caused by subtours. The cut disaggregation (5.1.2) was used to disaggregate the subproblem, one for each berth.

The computational tests performed indicated that Benders Decomposition may be an interesting approach to solve the BAP. Although being competitive with monolithic model resolution with CPLEX, in general Benders Decomposition does not outperform CPLEX. However, the exception lies on some of the largest instances, indicating that for the most difficult instances Benders Decomposition may have a better performance.

Many real-world systems state change frequently due to unforeseen events. Most of the computational time running scheduling systems is spent in rescheduling, caused by changes in customer priorities, unexpected equipment maintenance, etc. This results in a requirement for frequent re-optimization. For example, when a crane in an automated container terminal malfunctions, a new equipment schedule for the entire port facility must be available within five to ten minutes, otherwise the handling of vessels will be delayed. When such unexpected problems come up, the terminal operator must be ready to change the service system; developing a tool that re-optimizes the system and quickly find a solution to the problem help improve the dynamics of the terminals and consequently their revenue. The results provided by Benders Decomposition in the Berth Allocation Problem open the possibility of incorporating this algorithm in a decision support system to re-optimize solutions whenever unforeseen events occur. Indeed, fixing variables and optimizing the others is inherent to Benders Decompositions algorithms.

6 Hybrid Evolutionary Algorithm for the BAP

Among the several metaheuristics that have already been proposed in the literature to solve the BAP, the genetic algorithm (GA) is the one that stands out most. (GOLDBERG; HOLLAND, 1988) states that genetic algorithms are probabilistic search procedures to work on large spaces involving states that can be represented by strings. It is a metaheuristic inspired by the process of natural selection and is commonly used to generate high-quality solutions to optimization problems hard to solve. A population of individuals (solutions) is evolved toward better solutions. Each candidate solution has a set of chromosomes which can be recombined and mutated to form a new generation. (GOLIAS *et al.*, 2009a) proposed a GA metaheuristic and the measure of performance is used based on each objective functions' satisfactoriness criterion. The vessels are grouped according to the cargo volume, inducing the use of different objective functions, one for each group. (PRATAP *et al.*, 2015) used the non-sorting genetic algorithm (NSGA II) to solve the BAP as a sequencing problem for a realistic scenario of a port located in the eastern coast of India, minimizing the total vessel waiting time and the customer priority.

In some situations, the genetic algorithm may lose the diversity of individuals in the population and converge to a local optimum solution. Therefore, this Chapter develops a hybrid optimization procedure based on Genetic Algorithm (GA) and Scatter Search (SS) for the discrete and dynamic BAP (Hybrid Evolutionary Algorithm for the BAP - HEABAP). Scatter search is an evolutionary optimization procedure. Operating on subset of solutions, the method makes limited use of randomization as a proxy for diversification when searching for a globally optimal solution. Therefore, solutions rapidly tend to the optimum, preserving the diversity required to ensure a global search covering all the solution set and the performance shows of better. (LIU, 2007) developed a hybrid scatter search by incorporating the nearest neighbor rule, threshold accepting and edge recombination crossover into a scatter search conceptual framework to solve the probabilistic traveling salesman problem. (MAENHOUT; VANHOUCKE, 2010) presented a scatter search procedure to solve the airline crew rostering problem. (DEBELS *et al.*, 2006) combined elements from scatter search and a method for the optimization of unconstrained continuous functions that simulates the electromagnetism theory of physics for solving the resource-constrained project scheduling problem. (GONZÁLEZ; ADENSO-DÍAZ, 2006) presented a scatter search metaheuristic to solve the optimum disassembly sequence problem. (RUSSELL; CHIANG, 2006) used a scatter search framework to solve the vehicle routing problem with time windows, once it is a highly constrained problem. (KESKIN; USTER, 2007) developed a scatter search-based metaheuristic approach hybridized with local search and path-relinking routines.

The implementations of the algorithm operators can be done in several ways, according to the coding structure of the solution. This Chapter proposes five initializations, three crossover and three variations for the scatter search parameters, and data envelopment analysis (DEA) is adopted to choose the most efficient combination of the algorithmic operators that will be developed for the hybrid algorithm. Section 6.1 describes the HEABAP. Section 6.2 presents the DEA models for evaluating the relative efficiency of combinations of algorithm operators. Section 6.3 reports the numerical conducted experiments.

6.1 Hybrid Genetic Algorithm

In this section, an overview of the algorithm customized for the BAP described in Chapter 5 is presented.

First, the population size P and the number of generations G need to be empirically defined performing a series of tuning experiments. An individual represents a feasible solution to the problem and all individuals from the population were initialized in order to represent feasible solutions (Section 6.1.1).

Figure 16a exemplifies coding structures x and l that represents an individual from the population for the BAP with 2 berths and 5 vessels. In Figure 16b, an auxiliary structure is represented in the form of linked list (BL) to carry out the operations of initialization, crossover and mutation.

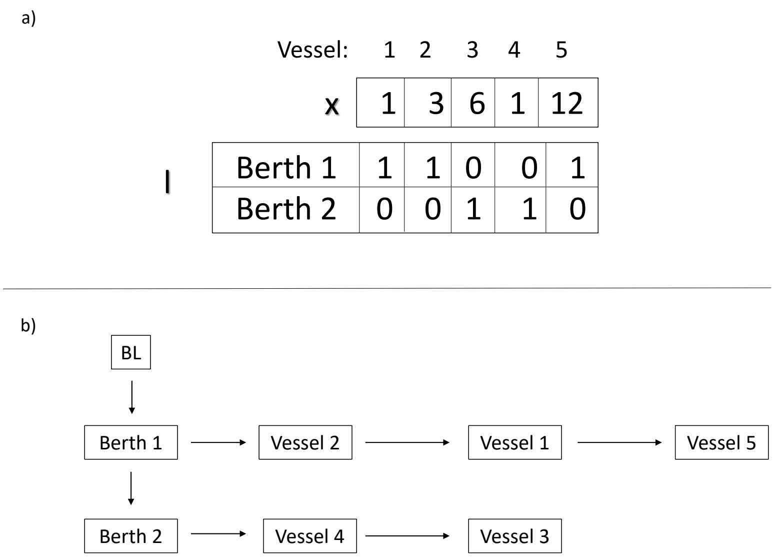


Figure 16 – Representation.

The scatter search method proposed in (LAGUNA; MARTI, 2012) uses strategies for search diversification and intensification, operating on a set of solutions (the so-called *reference set*) by combining these solutions to create new ones. The reference set,

$RefSet$ with size $b = b_1 + b_2 = |RefSet|$, is a collection of b_1 high quality solutions and b_2 diverse solutions of P that are used to generate new solutions through the crossover. In this Chapter, the scatter search was hybridized with the genetic algorithm with the main goal of maintaining the diversity of the population and avoiding premature convergence.

After the population has been empirically initialized it was observed the occurrence of a idle time window for the berth service. For such reason, a treatment was applied to the population in order to improve the generated solutions (Section 6.1.2).

Next, each individual t must be evaluated through the calculation of the objective function value (2.22). The population P is then sorted in ascending order of objective function values. The construction of the initial reference set starts adding to it the first b_1 solutions from initial population P. The structures x of each individual in $P \setminus RefSet$ and each individual in $RefSet$ are compared. From $P \setminus RefSet$, the b_2 most different individuals from the ones in $RefSet$ (in the sense of vessel processing start time) must be selected and copied to $RefSet$. Therefore, the resulting reference set has b_1 high quality solutions and b_2 most diverse solutions.

After the initial reference set is constructed, the set of parents must be generated to be submitted to crossover (Section 6.1.3). The crossover is a problem-specific mechanism, because it is directly related to the solution representation. It considers random pairs of solutions in the $RefSet$ that contain at least a solution that has not been combined in the past. In other words, the procedure does not allow for the same two solutions to be subjected to the crossover more than once. Once a new subset has been updated, the crossover is called, giving raise to the offspring population. The mutation process complements the crossover (Section 6.1.4). It is applied to the offspring population and it allows a larger search space to be explored. A local search procedure (Section 6.1.5), based on the one proposed in (TING *et al.*, 2014b), was also applied to the offspring population. The vessels can be swapped in the same berth and between berths.

Next to the local search, the offspring population is full and the reference set is updated. The update reference set contains the best b solutions in $RefSet \cup$ (*offspring population*) and the individuals are sorted in ascending order of objective function values. If the reference set remains unchanged after the updating procedure, a rebuilding mechanism is performed. It is defined a "change rate" $0 < cr < 1$ to evaluate if $RefSet$ remained unchanged. If the $RefSet$ at the end of generation g contains $cr * b$ individuals equal to those at the end of generation $g - 1$, then $RefSet$ is considered unchanged. The rebuilding mechanism is similar to how the initial $RefSet$ was created. First, a new population P is constructed and the two individuals with the best objective function value are selected. Then, the structure x of each solution in P is compare to best b_1 individuals in the set $RefSet$ amended by this two selected individuals. The solutions in P with a greater number of different positions are added to the $RefSet$, thus replacing the worst $b_2 - 2$ solutions (the $RefSet$ remains b).

More details of the implementations of the operators are described in the following.

6.1.1 Population Inicialization

Initialization was implemented in four ways, which will be described below.

6.1.1.1 Random

All individuals are randomly generated. For each vessel i, a berth is sorted randomly. Then, for each berth the sequence of vessels being serviced is defined randomly.

6.1.1.2 Based on arrival time

In the first part, for each individual t from the population, we chose the m vessels that arrived first to begin service in each one of the m berths, which is select randomly. The remaining vessels are sorted in ascending order of arrival time and then, following such order, a berth is chosen randomly to allocate the vessel.

6.1.1.3 Based on processing time

For each individual, a random a list of vessels (RL) is created. Then, for each vessel i in the list RL, we choose the berth k^* where there is the shortest processing time, i.e.:

$$p_i^{k^*} \leq p_i^k \ \forall k \in M \tag{6.1}$$

Variable x is properly initialized and next we must check for feasibility. If

$$x_i^{k^*} + p_i^{k^*} \leq b_i \tag{6.2}$$

and

$$x_i^{k^*} + p_i^{k^*} \leq e^{k^*} \tag{6.3}$$

vessel i is allocated to berth k^*. Otherwise, we search for the next shortest processing time.

6.1.1.4 Based on berth idle time

For each individual, a random list of vessels (RL) is created. Then, for each vessel i in the list RL, we calculate for each berth k

$$x_i^k + p_i^k \tag{6.4}$$

which is the time that berth k will become available if we allocate vessel i (the so-called *future idle time*). We choose to allocate vessel i in the berth where the shortest future idle time occurs.

6.1.2 Treatment of idleness

To illustrate the proposed treatment, consider the example of (Figure 17a). Suppose that for a given berth $\hat{k}$ the vessels were allocated in the following order (with the respective processing and arrival times) and that the berth time window started at 12.

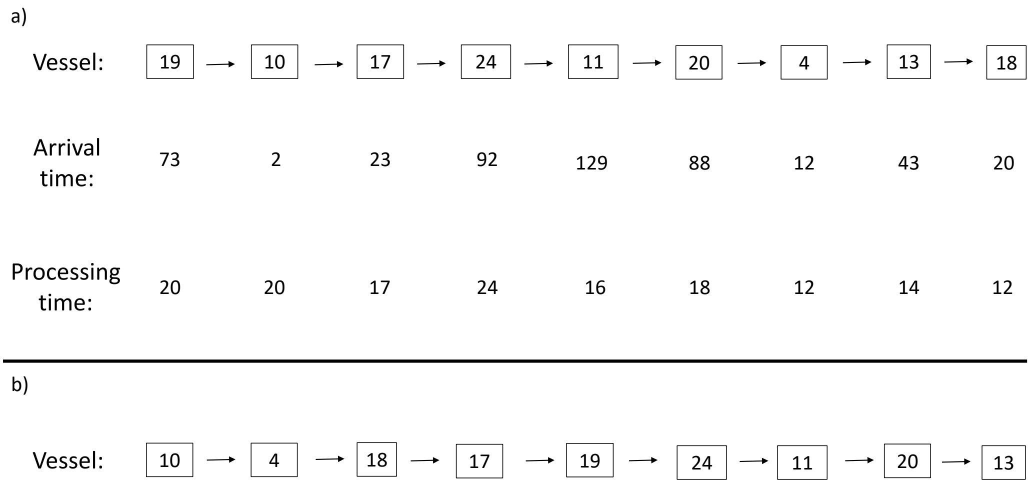


Figure 17 – Treatment of idleness example.

The berth remained idle for 61 units of time, because vessel 19 was the first one to be serviced at time 73, i.e, the idle window is $[s^{\hat{k}}, x_{19}^{\hat{k}}] = [12, 73]$.

Analyzing the arrival times, it is observed that vessels 24, 11 and 20 can not be allocated before vessel 19. The remaining vessels are organized in a increasing order of arrival time and following this order, the possibility of servicing each vessel in the idle time window is checked. When possible, the relocation is performed and the configuration in Figure 17b is obtained. Performing the described process, the idle time window in the beginning of the planning horizon is significantly reduced.

Hereafter, a similar procedure is applied to try to reduce the remaining idle time windows, taking advantage of vessels waiting to be serviced. The vessel with the longest waiting time and the berth with the greatest total idleness are sought. If such vessel is allocated in a different berth, it is relocated in order to be serviced as soon as it arrives to fill the idle time.

6.1.3 Crossover Operators

The crossover was implemented in three ways. Because of the procedure of copying a portion of berths, each offspring will have characteristics inherited from both parents, in addition to ensuring that the offspring population is feasible. The reintegration of lost vessels will guarantee the genetic variability necessary in an evolutionary algorithm.

Before introducing the crossover implementation, we point out that in all crossovers performed, the vessels that were not assigned to any berth must be allocated as described bellow.

First, the time in which each berth becomes available must be calculated, because there may already be some vessels being served (so-called *idle time*). Next, a vessel that was not assigned to any berth and has the shortest arrival time is sought. Let i^* be the index of such vessel. Then, for this vessel, for every berth is calculated:

$$x_{i^*}^k + p_{i^*}^k, \tag{6.5}$$

which represents the time the berth will become available if vessel i^* is allocated (*future idle time*). Let k^* be the index of the berth with the shortest future idle time. If the berth time window is not violated, vessel i^* is allocated in berth k^*. Other wise, the berth with the next shortest future idle time is sought.

This process is repeated until all vessels have been allocated in some berth.

6.1.3.1 Horizontal

For the horizontal crossover, we use the auxiliary structure BL. The even berths are copied, and the odd berths are completed. The odd berths are completed following the rules that ensure the feasibility maintenance in the offspring population (Figure 18).

The crossover is still performed in the reverse way, each with a 50% probability of happening (Figure 19).

6.1.3.2 Vertical

For the vertical crossover, the auxiliary structure BL is used. The first half of the vessels in each berth of parent 1 is copied to the offspring population 1; the first half of the vessels in each berth of parent 2 is copied to the offspring population 2. The second half of the vessels in each berth of the offspring population 1 is completed with the vessels on the second half of the vessels in each berth of the parent 2, provided that it has not been allocated in the previous step; the second half of the vessels in each berth of the offspring population 2 is completed with the vessels on the second half of the vessels

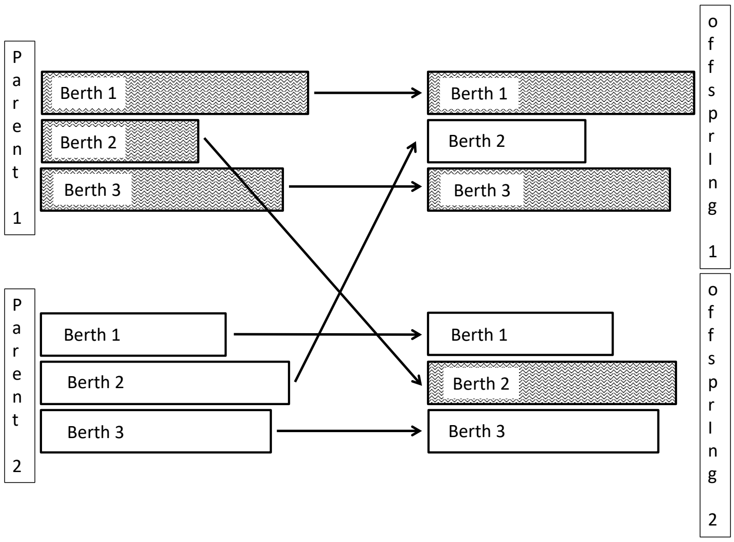


Figure 18 – Crossover 1 representation - horizontally.

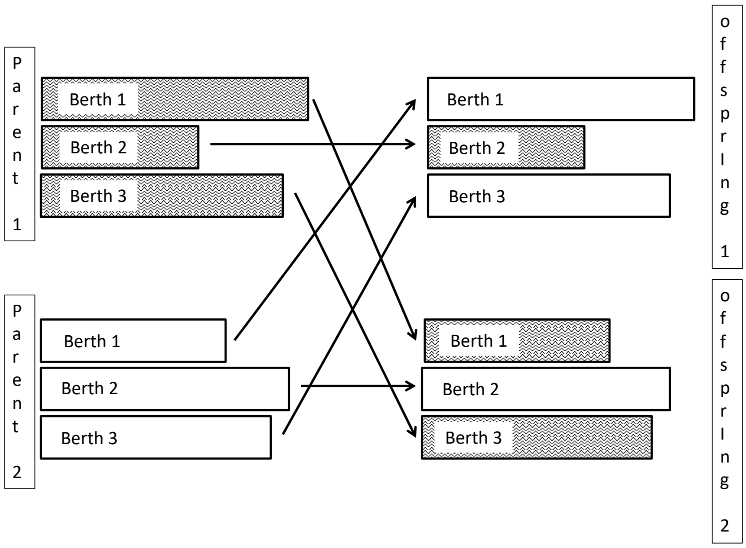


Figure 19 – Crossover 2 representation - horizontally.

in each berth of the parent 1, provided that it has not been allocated in the previous step (Figure 20).

6.1.3.3 Vertical in the structure l

Another vertical crossover was implemented, considering the structure l from the individual codification. The same procedure of Section 6.1.3.2 was applied to the matrix "vessel-berth allocation" (Figure 21).

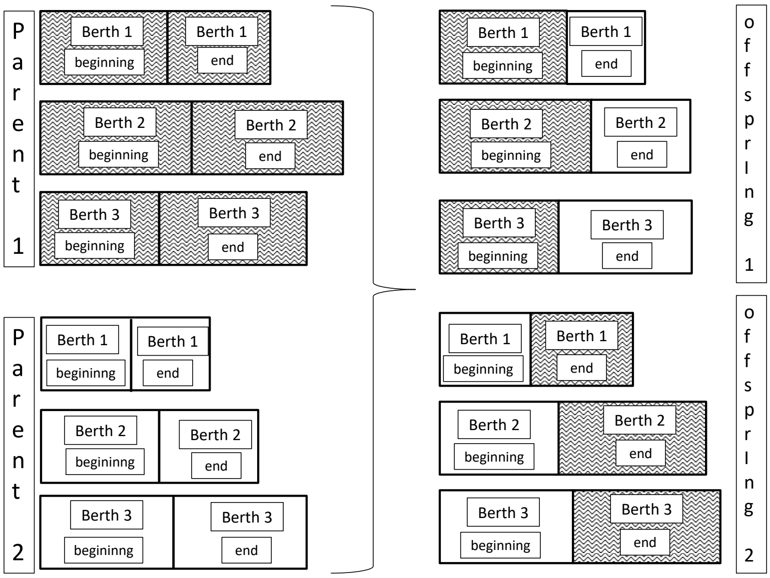


Figure 20 – Crossover 3 representation - vertically.

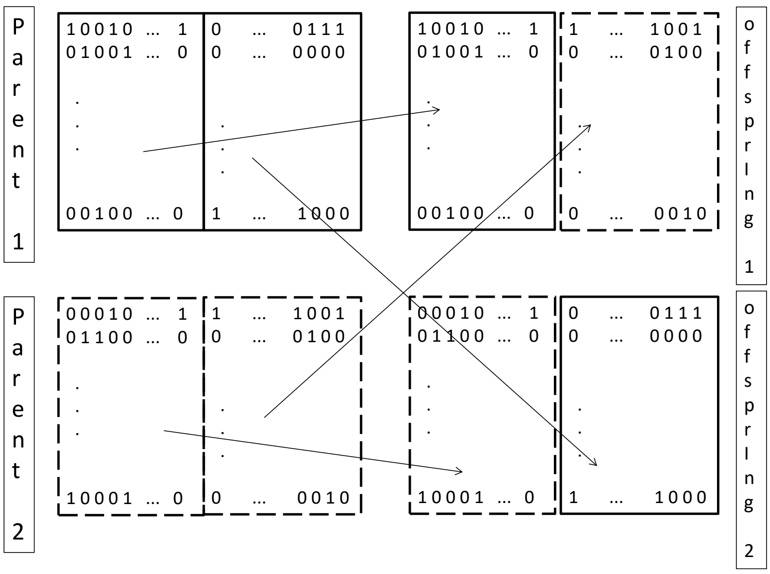


Figure 21 – Crossover 4 representation - vertically.

6.1.4 Mutation Operators

For the mutation, two operators were proposed.

6.1.4.1 Between vessels in the same berth

If the two vessels i and j are allocated to the same berth k, only the vessels allocation order is swapped (Figure 22a). To ensure that the quality of mutated individuals, and consequently of the solution, will not get worse, the mutation process occurs if, and only if:

$$a_i \leq x_j^k \text{ and } a_j \leq x_i^k. \tag{6.6}$$

6.1.4.2 Between vessels in different berths

If the two vessels are allocated to different berths, k' and k'', the berth allocation is swapped (Figure 22b). To ensure that the quality of mutated individuals, and consequently of the solution, will not get worse, besides the condition 6.6, the mutation process occurs if, and only if:

$$x_j^{k'} + p_i^{k'} \leq x_j^{k'} + p_j^{k'} \text{ and } x_i^{k''} + p_j^{k''} \leq x_i^{k''} + p_i^{k''}. \tag{6.7}$$

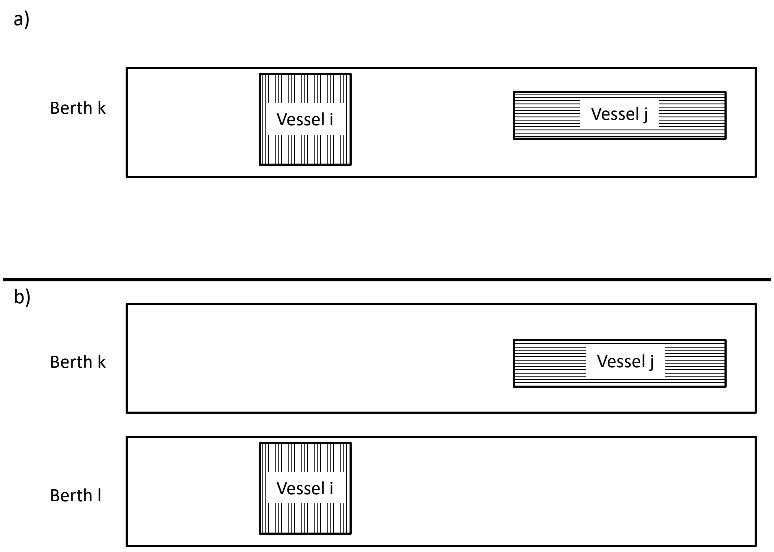


Figure 22 – Mutation representation.

6.1.5 Local search

As in (TING *et al.*, 2014b), to comprehensively explore the neighborhood of the current solution a local search is incorporated to the algorithm.

6.1.5.1 Between vessels in the same berth

Given the sequence of the vessels handled by such berth, it compares all the possible swapping pairs within the same berth and select the best improvement to exchange their values.

6.1.5.2 Between vessels in the different berths

The vessel higher waiting time is selected. For all vessels allocated on a different berth and that have already arrived by the time such selected vessel started being handled, the future idle time is computed if the vessels were swapped. If the swapping does not decrease the future idle time, the structure x and the objective function are recalculated, and the swap that results in the objective function with lower value is performed.

6.2 Data Envelopment Analysis

The BAP involves many criteria that can be used to evaluate how good a solution is, like makespan, waiting time, handling time. Besides, there are different ways to configure the implemented algorithm, we need a tool to guide the decision on how to use each proposed operator. Among the techniques in Multicriteria Decision Making, Data Envelopment Analysis (DEA) has obtained important results in complex situations with multiple and conflicting criteria, which can not be easily analyzed through other approaches, according to (T'KINDT; BILLAUT, 2006). In addition, with Data Envelopment Analysis, it is not necessary for the decision maker to rank or sort the criteria, allowing the comparison of alternatives with heterogeneous characteristics. The DEA models construct a nonparametric and piecewise linear surface involving the data (DEA front), associated with multiple criteria. Once the criteria have been selected and the values corresponding to each alternative solution have been estimated, we apply DEA to analyze the efficiency of each combination.

(COOPER *et al.*, 2011) define the data envelopment analysis (DEA) as a data-oriented approach, used for evaluating the performances of a set of entities called decision-making units (DMUs) which convert multiple inputs into multiple outputs.

Charnes, Cooper and Rhodes used the ratio of outputs to inputs to measure the relative efficiency of a given DMU, building the so called CCR Model. Therefore, the efficiency is expressed based in the conventional *benefit/cost* theory and, for such reason, the model is also known as Constant Returns to Scale (CRS).

Consider a set of n DMUs, represented by solutions, each consuming different amounts of r inputs to produce s outputs. For example, we can consider as input in the DEA model, minimizing the makespan and the waiting time; and as outputs, maximizing berth utilization within the window, throughput within the window ((DAI *et al.*, 2008)), the total number or profit of the vessels processed ((ELIIYI *et al.*, 2008)), among others. Eff_o is the efficiency of DMU o, the one under analysis; v_i and u_j are the weights given, respectively, to inputs i, $i = 1, \ldots, r$, and outputs j, $j = 1, \ldots, s$. The variables x_{ik} are the inputs i and y_{jk} are the outputs j of DMU k, $k = 1, \ldots, n$, and x_{io} are the inputs i and y_{jo} are the outputs j of DMU o.

Turn back to the concept of *benefit/cost*. The efficiency of a DMU o is:

$$Eff_o = \frac{\sum_{j=1}^{s} u_j y_{jo}}{\sum_{i=1}^{r} v_i x_{io}} \tag{6.8}$$

If efficiency is achieved by an equiproportional reduction of inputs and outputs are maintained constant (input orientation - Section 6.2.1), we set

$$\sum_{i=1}^{r} v_i x_{io} = 1 \tag{6.9}$$

and the efficiency becomes

$$Eff_o = \sum_{j=1}^{s} u_j y_{jo} \tag{6.10}$$

When the goal is to increase the outputs without decreasing the inputs (output orientation - Section 6.2.2), we set

$$\sum_{j=1}^{s} u_j y_{jo} = 1 \tag{6.11}$$

and the efficiency becomes

$$Eff_o = \frac{1}{\sum_{i=1}^{r} v_i x_{io}} \tag{6.12}$$

For the remaining DMUs k,

$$\begin{aligned}
Eff_o &\leq 1 && (6.13)\\
\Downarrow & && (6.14)\\
\frac{\sum_{j=1}^{s} u_j y_{jk}}{\sum_{i=1}^{r} v_i x_{ik}} &\leq 1 \ \forall k && (6.15)\\
\Downarrow & && (6.16)\\
\sum_{j=1}^{s} u_j y_{jk} - \sum_{i=1}^{r} v_i x_{ik} &\leq 0 \ \forall k && (6.17)
\end{aligned}$$

6.2.1 Input orientation

The CCR model, input oriented, is described as follows:

$$\max \quad Eff_o = \sum_{j=1}^{s} u_j y_{jo} \tag{6.18}$$

$$\text{s.t} \quad \sum_{i=1}^{r} v_i x_{io} = 1 \tag{6.19}$$

$$\sum_{j=1}^{s} u_j y_{jk} - \sum_{i=1}^{r} v_i x_{ik} \le 0 \ \forall k \tag{6.20}$$

$$v_i, u_j \ge 0 \ \forall i, j \tag{6.21}$$

A DMU is inefficient if the efficiency value given by the optimal value for the problem (6.18)-(6.21) is less than one.

Let $Eff_o = \frac{1}{h_0}$. The dual of problem (6.18)-(6.21) is defined by (6.22)-(6.25):

$$\min \quad h_o \tag{6.22}$$

$$\text{s.t} \quad h_o x_{io} - \sum_{k=1}^{n} x_{ik} \lambda_k \ge 0 \ \forall i \tag{6.23}$$

$$-y_{jo} + \sum_{k=1}^{n} y_{jk} \lambda_k \ge 0 \ \forall j \tag{6.24}$$

$$\lambda_k \ge 0 \ \forall k \tag{6.25}$$

Constraints (6.23) ensures that this reduction in each of the inputs does not exceed the boundary defined by efficient DMUs. Constraints (6.24) ensures that reduction in inputs does not change the current level of DMU outputs.

If the optimal value is equal to one and the slacks of constrains (6.23) and (6.24) are zero (ie, $h_o x_{io} - \sum_{k=1}^{n} x_{ik} \lambda_k = 0$ and $-y_{jo} + \sum_{k=1}^{n} y_{jk} \lambda_k = 0$), then the DMU is efficient.

All efficient points lie on the DEA front. An inefficient DMU ô can be made more efficient by projection onto the front, through proportional reduction of inputs. Multiplying all inputs by the value of $Eff_{\hat{o}}$ (smaller then one), the DMU ô is taken to the efficient front.

6.2.2 Output orientation

The CCR model, output oriented, is described as follows:

$$\min \quad h_o = \frac{1}{Eff_o} = \sum_{i=1}^{r} v_i x_{io} \tag{6.26}$$

$$\text{s.t} \quad \sum_{j=1}^{s} u_j y_{jo} = 1 \tag{6.27}$$

$$\sum_{j=1}^{s} u_j y_{jk} - \sum_{i=1}^{r} v_i x_{ik} \le 0 \ \forall k \tag{6.28}$$

$$v_i, u_j \ge 0 \ \forall i, j \tag{6.29}$$

and its dual problem is defined by (6.30)-(6.33):

$$\max \quad h_o \tag{6.30}$$

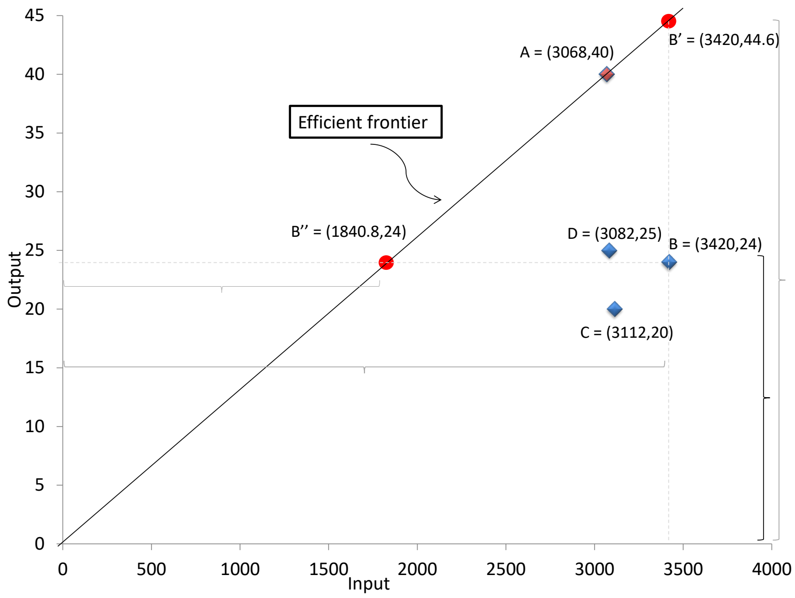
$$\text{s.t} \quad x_{io} + \sum_{k=1}^{n} x_{ik}\lambda_k \geq 0 \;\; \forall i \tag{6.31}$$

$$-h_o y_{jo} + \sum_{k=1}^{n} y_{jk}\lambda_k \geq 0 \;\; \forall j \tag{6.32}$$

$$\lambda_k \geq 0 \;\; \forall k \tag{6.33}$$

For a inefficient DMU ô, $h_{\hat{o}}$ is a number greater than 1, so efficiency is $\frac{1}{h_{\hat{o}}}$. $h_{\hat{o}}$ represents by how much all outputs must be multiplied, keeping inputs constant, for the DMU to reach the efficient front.

According to (COOK; SEIFORD, 2009), the CCR model is referred to as providing a radial projection. The lines that link the inefficient DMUs to the axes allow finding the targets DMUs, which are the points where the lines intercept the front.



Solution	Efficiency
A	1
B	0.538246
C	0.492931
D	0.622161

- B': output orientation
- B": input orientation

$$Eff_A = \frac{1840.8}{3420} = \frac{24}{44.6} = 0.538$$

Figure 23 – CRS projection for one input and one output.

6.2.3 Undesired outputs

As highlighted in (LU; YU, 2012), to determine the inputs and outputs of the DMUs is an important step when applying DEA. They considered the inputs in light of the defined objective function, which is a minimization problem. Therefore, a combination of algorithmic operators is considered efficient if its outputs are minimized. However, as reported earlier in this Section, in the classical DEA the inputs have to be minimized and outputs have to be maximized; that is, the outputs proposed in (LU; YU, 2012) are undesired.

(SCHEEL, 2001) discusses in more detail the undesired outputs. It can be amended by transforming the values of the undesired outputs by a monotone decreasing function. The additive inverse approach incorporates the undesired output u with values $f(u) = -u$. It also can be translated making $f(u) = -u + \beta$. The multiplicative inverse approach incorporates the undesired output using $f(u) = 1/u$. Further, if a DMU is efficient using the multiplicative inverse, then it is efficient as well when the additive inverse is used to incorporate the undesired outputs.

6.3 Numerical experiments

The before mentioned operators were implemented in the following way.

Five combinations of the proposed initializations (Section 6.1.1) were analyzed:

- I1 = random (Section 6.1.1.1),
- I2 = based on arrival time (Section 6.1.1.2),
- I3 = based on processing time (Section 6.1.1.3),
- I4 = based on berth idle time (Section 6.1.1.4),
- I5 = $\frac{1}{4}$I1 + $\frac{1}{4}$I2 + $\frac{1}{4}$I3 + $\frac{1}{4}$I4.

Both idleness treatment was always performed (Section 6.1.2).

Three crossovers were implemented:

- C1: the probability of horizontal crossover (Section 6.1.3.1) is $\mathrm{P}_h = 25\%$, the probability of vertical crossover (Section 6.1.3.2) is $\mathrm{P}_v = 25\%$ and the probability of vertical crossover in structure l (Section 6.1.3.3) is $\mathrm{P}_v(l) = 50\%$,
- C2: the probability of horizontal crossover (Section 6.1.3.1) is $\mathrm{P}_h = 25\%$, the probability of vertical crossover (Section 6.1.3.2) is $\mathrm{P}_v = 50\%$ and the probability of vertical crossover in structure l (Section 6.1.3.3) is $\mathrm{P}_v(l) = 25\%$,
- C3: the probability of horizontal crossover (Section 6.1.3.1) is $\mathrm{P}_h = 50\%$, the probability of vertical crossover (Section 6.1.3.2) is $\mathrm{P}_v = 25\%$ and the probability of vertical crossover in structure l (Section 6.1.3.3) is $\mathrm{P}_v(l) = 25\%$.

Mutation was performed only for vessels in different berths (Section 6.1.4.1) because the local search covers the case of the mutation in the same berth. The local search was always performed, both for vessels in the same berth (Section 6.1.5.1) and for vessels in different berths (Section 6.1.5.2).

For the scatter search, the considered change rate was $cr = 0.95$ and three combinations for b_1 and b_2 ($RefSet$) were analyzed:

- R1: $b_1 = 90$ and $b_2 = 10$,
- R2: $b_1 = 80$ and $b_2 = 20$,
- R3: $b_1 = 70$ and $b_2 = 30$.

Therefore, there were 45 total combinations. Each combination was named by the acronyms of the three constituent operators (initialization, crossover and $RefSet$). The test problems instances were the ones from (CORDEAU *et al.*, 2005). There were five problem classes: 25 vessels and 5 berths; 25 vessels and 7 berths; 25 vessels and 10 berths; 35 vessels and 7 berths; 35 vessels and 10 berths and one instance for each classes was randomly selected. The parameter settings were: population size = 500, number of iterations = 1000 and $RefSet$ size = 100. Each of the 45 combinations was tested 30 times using different seeds and the averages of the inputs and outputs were obtained to fed the model (6.18)-(6.21).

To determine the inputs of the DMUs, take the objective function 2.22:

$$\min \qquad \sum_{i \in N} \sum_{k \in M} v_i \left(\left(x_i^k - a_i \right) + p_i^k \sum_{j \in N \cup \{d\}} l_{ij}^k \right)$$

The service time can be splitted in waiting time:

$$\sum_{i \in N} \sum_{k \in M} v_i \left(x_i^k - a_i \right)$$

and handling time:

$$\sum_{i \in N} \sum_{k \in M} v_i \left(p_i^k \sum_{j \in N \cup \{d\}} l_{ij}^k \right)$$

The computational time is a consequence of the operator combination, and we want the combination with the shortest time to achieve the best values for the objective function. Therefore, it can be considered an input as well to be minimized.

Because the problem has no target to maximize and we are studying the input oriented CCR-model, we consider the outputs as a constant of value one.

The results for the efficiency values for each combination when solving the CCR model for constant output are reported in Table 17.

		25x5	25x7	25x10	35x7	35x10
1	I1 - C1 - R1	0.998461	0.99088	0.996794	0.99718	0.997433
2	I1 - C1 - R2	0.997532	0.989156	0.998056	0.996416	0.995376
3	I1 - C1 - R3	1	0.997491	0.998447	0.997986	1
4	I1 - C2 - R1	0.995843	0.999645	1	0.995562	0.988218
5	I1 - C2 - R2	0.998578	0.994945	0.99665	0.994904	0.993504
6	I1 - C2 - R3	1	1	1	1	1
7	I1 - C3 - R1	0.998927	0.995186	0.992277	0.994288	0.992579
8	I1 - C3 - R2	1	0.996344	1	0.998445	0.992858
9	I1 - C3 - R3	1	0.99787	0.992178	0.993602	1
10	I2 - C1 - R1	0.999968	0.991739	0.990377	0.991989	0.990441
11	I2 - C1 - R2	0.998799	0.992941	0.993665	0.992896	0.992714
12	I2 - C1 - R3	0.995816	0.990554	0.994772	0.991896	0.990964
13	I2 - C2 - R1	0.997988	1	0.990249	0.995623	0.982796
14	I2 - C2 - R2	1	0.994436	0.994267	0.995149	0.987587
15	I2 - C2 - R3	0.999059	0.995676	1	0.991067	0.991342
16	I2 - C3 - R1	0.996938	0.994655	0.987607	0.991377	0.983764
17	I2 - C3 - R2	0.999612	1	0.995134	0.996109	0.984485
18	I2 - C3 - R3	0.998763	1	1	0.991877	0.992665
19	I3 - C1 - R1	0.99544	0.991527	0.994552	0.99716	0.995752
20	I3 - C1 - R2	0.99581	0.997653	0.994269	0.996297	0.997621
21	I3 - C1 - R3	0.999159	0.999744	0.994856	0.995929	0.995399
22	I3 - C2 - R1	0.997313	0.991152	0.998429	0.993162	1
23	I3 - C2 - R2	0.99351	0.99711	0.997144	1	0.999743
24	I3 - C2 - R3	1	0.998415	1	1	1
25	I3 - C3 - R1	0.995627	0.998438	0.99393	0.995847	0.994287
26	I3 - C3 - R2	0.996824	0.997223	0.992808	0.996577	0.994252
27	I3 - C3 - R3	0.99885	0.996044	0.993979	0.996562	0.997816
28	I4 - C1 - R1	0.994353	0.99646	0.99751	0.993413	0.987341
29	I4 - C1 - R2	0.996627	0.995968	0.999362	0.999057	0.988035
30	I4 - C1 - R3	0.993274	0.990306	0.999359	0.999632	1
31	I4 - C2 - R1	0.990994	0.9966	0.999756	0.992831	0.990107
32	I4 - C2 - R2	1	0.995366	0.997699	0.996412	0.992286
33	I4 - C2 - R3	0.99237	0.994845	0.998677	0.999066	0.989117
34	I4 - C3 - R1	0.995843	0.999645	1	0.995562	0.988218
35	I4 - C3 - R2	0.99601	1	0.996126	0.994682	0.988544
36	I4 - C3 - R3	0.992487	0.994392	1	1	0.991558
37	I5 - C1 - R1	0.9944	0.992594	0.994183	0.994512	0.988361
38	I5 - C1 - R2	0.993878	0.998052	1	1	0.99127
39	I5 - C1 - R3	0.997034	0.998172	0.997233	0.996959	0.996132
40	I5 - C2 - R1	0.998993	0.995849	0.992395	0.994366	0.988397
41	I5 - C2 - R2	0.996584	1	1	0.993407	0.995355
42	I5 - C2 - R3	0.997159	0.995186	1	0.998706	0.989067
43	I5 - C3 - R1	0.998192	0.999948	0.991222	0.994745	0.992904
44	I5 - C3 - R2	0.999319	0.993728	0.9923	0.994881	0.992003
45	I5 - C3 - R3	0.995095	0.995864	0.994916	1	0.995693
	average	0.997142867	0.996039978	0.9962484	0.995914022	0.992799644
	standard deviation	0.002399518	0.003050558	0.003318403	0.002611906	0.004624718

Table 17 – Efficiency of the CCR model for constant output

25x5	25x7	25x10	35x7	35x10
I1-C1-R3	I1-C2-R3	I1-C2-R1	I1-C2-R3	I1-C1-R3
I1-C2-R3	I2-C2-R1	I1-C2-R3	I3-C2-R2	I1-C2-R3
I1-C3-R2	I2-C3-R2	I1-C3-R2	I3-C2-R3	I1-C3-R3
I1-C3-R3	I2-C3-R3	I2-C2-R3	I4-C3-R3	I3-C2-R1
I2-C2-R2	I4-C3-R2	I2-C3-R3	I5-C1-R2	I3-C2-R3
I3-C2-R3	I5-C2-R2	I3-C2-R3	I5-C3-R3	I4-C1-R3
I4-C2-R2		I4-C3-R1		
		I4-C3-R3		
		I5-C1-R2		
		I5-C2-R2		
		I5-C2-R3		

Table 18 – Efficient combinations for constant output

The efficient combinations identified by the CCR model for the approach considering three inputs and constant outputs are summarized in Table 18. It is noteworthy that there are many combinations that have shown to be efficient. To refine the analysis, we considered another way of applying the DEA, in which the portions of the objective function as outputs and the time is an input, as proposed in (LU; YU, 2012). Because the outputs should be maximized in a classical input-orientated DEA model, the outputs in this work are undesired. The translated and the multiplicative approach previously presented in Section 6.2.3 and taking into account the order of magnitude of the data, are analyzed to amend such situation. In the translated approach, we used for the waiting time:

$$f(u) = 1000 - u$$

and for the handling time we used:

$$f(u) = 10000 - u$$

For the multiplicative approach we used

$$f(u) = \frac{100}{u}$$

The relative efficiencies obtained by the CCR model of the 45 operator combinations for the translated approach (Trans.) and for the multiplicative approach (Mult.) are reported in Table 19. The efficiency for the multiplicative approach is ranked in Table 20. The efficient combinations for the multiplicative approach are summarized in Table 21.

DMU	Operator combination	25x5		25x7		25x10		35x7		35x10	
		Trans.	Mult.	Trans.	Mult.	Trans.	Mult.	Trans.	Mult.	Trans.	Mult.
1	I1 - C1 - R1	0.805408	0.808961	0.650004	0.646798	0.815762	0.836302	0.782033	0.781385	0.844561	0.850555
2	I1 - C1 - R2	0.812639	0.815397	0.713365	0.708184	0.925519	0.930689	0.897713	0.895673	0.8848	0.889343
3	I1 - C1 - R3	1	1	0.856366	0.856705	0.961282	0.984638	0.9024	0.936892	0.959802	0.96659
4	I1 - C2 - R1	0.827308	0.830723	0.67994	0.680911	0.735031	0.795947	0.810953	0.841581	0.821206	0.819919
5	I1 - C2 - R2	0.918251	0.923315	0.799041	0.798296	0.903552	0.926602	0.886921	0.907161	0.947669	0.948389
6	I1 - C2 - R3	0.983741	0.985636	0.968718	1	0.964283	1	1	1	1	1
7	I1 - C3 - R1	0.749404	0.75064	0.655624	0.6554	0.740841	0.782764	0.795248	0.824201	0.79239	0.794763
8	I1 - C3 - R2	0.975925	0.977804	0.753408	0.753519	0.86699	0.870381	0.874273	0.933372	0.908522	0.907976
9	I1 - C3 - R3	0.957862	0.961113	0.863233	0.862769	0.817657	0.846349	0.935161	0.945187	0.959946	0.967296
10	I2 - C1 - R1	0.624592	0.62931	0.600494	0.598057	0.626775	0.643549	0.72217	0.718258	0.691412	0.692482
11	I2 - C1 - R2	0.604603	0.607099	0.598182	0.594765	0.730968	0.762069	0.830343	0.826421	0.803603	0.805882
12	I2 - C1 - R3	0.803494	0.804598	0.698467	0.69414	0.830905	0.870109	0.799102	0.836801	0.876374	0.874203
13	I2 - C2 - R1	0.669075	0.670485	0.656382	0.685182	0.660157	0.687635	0.722738	0.72118	0.761682	0.757444
14	I2 - C2 - R2	0.669798	0.672996	0.758515	0.756251	0.816167	0.827335	0.806792	0.83037	0.840085	0.838725
15	I2 - C2 - R3	0.854298	0.857669	0.782698	0.783175	0.881129	0.973932	0.812933	0.821302	0.915618	0.923077
16	I2 - C3 - R1	0.646081	0.647621	0.53906	0.538705	0.565035	0.570451	0.625881	0.622136	0.720787	0.716963
17	I2 - C3 - R2	0.762123	0.768306	0.806642	0.809864	0.925598	0.923671	0.945163	0.942295	0.865649	0.860854
18	I2 - C3 - R3	0.796808	0.797903	1	1	1	1	0.984705	0.977591	0.970716	0.96715
19	I3 - C1 - R1	0.681661	0.681224	0.556175	0.554298	0.678386	0.691682	0.665155	0.69423	0.701189	0.705179
20	I3 - C1 - R2	0.640025	0.640668	0.6438	0.645483	0.791411	0.790534	0.770694	0.77945	0.776805	0.780666
21	I3 - C1 - R3	0.849556	0.854241	0.676934	0.67991	0.84934	0.896596	0.797297	0.822198	0.848992	0.852296
22	I3 - C2 - R1	0.693163	0.696221	0.57329	0.570739	0.68458	0.71251	0.704565	0.723593	0.703161	0.710198
23	I3 - C2 - R2	0.868471	0.868945	0.774894	0.776	0.804318	0.805598	0.842298	0.861519	0.867591	0.875462
24	I3 - C2 - R3	0.924789	0.926396	0.793963	0.796038	0.871217	0.923004	0.833149	0.917407	0.911968	0.920314
25	I3 - C3 - R1	0.710381	0.71051	0.582869	0.584564	0.756568	0.775602	0.727951	0.777336	0.74649	0.750006
26	I3 - C3 - R2	0.852797	0.85124	0.698829	0.700169	0.834396	0.88639	0.728948	0.74383	0.842408	0.84602
27	I3 - C3 - R3	0.648843	0.64997	0.605511	0.606226	0.759448	0.781014	0.793648	0.792541	0.82829	0.831207
28	I4 - C1 - R1	0.59646	0.597223	0.645781	0.646768	0.758609	0.760423	0.795307	0.791947	0.784038	0.782768
29	I4 - C1 - R2	0.865572	0.870829	0.741771	0.741154	0.860689	0.863628	0.883686	0.928257	0.877817	0.876624
30	I4 - C1 - R3	0.960059	0.956649	0.758327	0.754734	0.925714	0.927439	0.910198	0.958026	0.982564	1
31	I4 - C2 - R1	0.775533	0.773553	0.635864	0.636584	0.747115	0.750404	0.770999	0.771479	0.798988	0.799073
32	I4 - C2 - R2	0.891731	0.899961	0.757035	0.756106	0.862837	0.864289	0.89215	0.894564	0.902459	0.904554
33	I4 - C2 - R3	0.786814	0.785664	0.605973	0.605827	0.777474	0.779784	0.947283	0.975179	0.963846	0.958657
34	I4 - C3 - R1	0.827308	0.830723	0.67994	0.680911	0.735031	0.795947	0.810953	0.841581	0.821206	0.819919
35	I4 - C3 - R2	0.873126	0.875366	0.715582	0.719082	0.841146	0.843703	0.870852	0.921761	0.85787	0.856682
36	I4 - C3 - R3	0.74216	0.739725	0.746228	0.745352	0.901543	0.904948	0.927622	0.985844	0.920916	0.919472
37	I5 - C1 - R1	0.530996	0.531919	0.379651	0.378405	0.510083	0.50999	0.562606	0.564128	0.502936	0.502791
38	I5 - C1 - R2	0.61087	0.60931	0.52066	0.522021	0.606295	0.610078	0.611455	0.612654	0.620439	0.621856
39	I5 - C1 - R3	0.643357	0.645718	0.54561	0.547088	0.626687	0.634349	0.631495	0.643426	0.66339	0.665044
40	I5 - C2 - R1	0.475612	0.478808	0.445386	0.445628	0.535368	0.563637	0.565659	0.563909	0.570806	0.570657
41	I5 - C2 - R2	0.623769	0.62652	0.510889	0.512981	0.848262	0.933205	0.870968	0.897004	0.886264	0.897604
42	I5 - C2 - R3	0.95336	0.952577	0.767182	0.76624	0.914115	0.957245	0.913169	0.930819	0.950454	0.946789
43	I5 - C3 - R1	0.7615	0.766545	0.607344	0.610283	0.72548	0.723142	0.779855	0.824332	0.785832	0.788434
44	I5 - C3 - R2	0.865662	0.871076	0.719128	0.716769	0.836255	0.858434	0.85956	0.907158	0.868548	0.869139
45	I5 - C3 - R3	0.912304	0.91322	0.789151	0.789346	0.821174	0.853889	0.91017	0.946617	0.945611	0.947198
	average	0.7783842	0.780319489	0.685731244	0.6869206	0.791804267	0.813997489	0.811382689	0.831835444	0.833237778	0.835116
	standard deviation	0.13042397	0.130537502	0.122636407	0.124245319	0.114673045	0.119436027	0.106120636	0.113370828	0.110191101	0.110915572

Table 19 – Efficiency of the CCR model for undesired outputs

DMU	Operator combination	25x5		25x7		25x10		35x7		35x10	
		rank	efficiency	rank	efficiency	rank	efficiency	rank	efficiency	rank	efficiency
1	I1 - C1 - R1	21	0,808961	28	0,646798	23	0,836302	32	0,781385	24	0,850555
2	I1 - C1 - R2	20	0,815397	20	0,708184	7	0,930689	18	0,895673	16	0,889343
3	I1 - C1 - R3	1	1	4	0,856705	3	0,984638	9	0,936892	5	0,96659
4	I1 - C2 - R1	18	0,830723	24	0,680911	26	0,795947	21	0,841581	28	0,819919
5	I1 - C2 - R2	8	0,923315	6	0,798296	9	0,926602	15	0,907161	7	0,948389
6	I1 - C2 - R3	2	0,985636	1	1	1	1	1	1	1	1
7	I1 - C3 - R1	28	0,75064	27	0,6554	29	0,782764	27	0,824201	32	0,794763
8	I1 - C3 - R2	3	0,977804	15	0,753519	15	0,870381	10	0,933372	13	0,907976
9	I1 - C3 - R3	4	0,961113	3	0,862769	21	0,846349	7	0,945187	3	0,967296
10	I2 - C1 - R1	39	0,62931	35	0,598057	40	0,643549	39	0,718258	41	0,692482
11	I2 - C1 - R2	42	0,607099	36	0,594765	33	0,762069	25	0,826421	30	0,805882
12	I2 - C1 - R3	22	0,804598	22	0,69414	16	0,870109	23	0,836801	19	0,874203
13	I2 - C2 - R1	34	0,670485	23	0,685182	39	0,687635	38	0,72118	36	0,757444
14	I2 - C2 - R2	33	0,672996	12	0,756251	24	0,827335	24	0,83037	26	0,838725
15	I2 - C2 - R3	15	0,857669	9	0,783175	4	0,973932	29	0,821302	10	0,923077
16	I2 - C3 - R1	36	0,647621	41	0,538705	43	0,570451	42	0,622136	38	0,716963
17	I2 - C3 - R2	26	0,768306	5	0,809864	10	0,923671	8	0,942295	21	0,860854
18	I2 - C3 - R3	23	0,797903	2	1	2	1	3	0,977591	4	0,96715
19	I3 - C1 - R1	32	0,681224	39	0,554298	38	0,691682	40	0,69423	40	0,705179
20	I3 - C1 - R2	38	0,640668	30	0,645483	28	0,790534	33	0,77945	35	0,780666
21	I3 - C1 - R3	16	0,854241	26	0,67991	13	0,896596	28	0,822198	23	0,852296
22	I3 - C2 - R1	31	0,696221	38	0,570739	37	0,71251	37	0,723593	39	0,710198
23	I3 - C2 - R2	14	0,868945	10	0,776	25	0,805598	20	0,861519	18	0,875462
24	I3 - C2 - R3	7	0,926396	7	0,796038	11	0,923004	14	0,917407	11	0,920314
25	I3 - C3 - R1	30	0,71051	37	0,584564	32	0,775602	34	0,777336	37	0,750006
26	I3 - C3 - R2	17	0,85124	21	0,700169	14	0,88639	36	0,74383	25	0,84602
27	I3 - C3 - R3	35	0,64997	33	0,606226	30	0,781014	30	0,792541	27	0,831207
28	I4 - C1 - R1	43	0,597223	29	0,646768	34	0,760423	31	0,791947	34	0,782768
29	I4 - C1 - R2	13	0,870829	17	0,741154	18	0,863628	12	0,928257	17	0,876624
30	I4 - C1 - R3	5	0,956649	14	0,754734	8	0,927439	5	0,958026	2	1
31	I4 - C2 - R1	25	0,773553	31	0,636584	35	0,750404	35	0,771479	31	0,799073
32	I4 - C2 - R2	10	0,899961	13	0,756106	17	0,864289	19	0,894564	14	0,904554
33	I4 - C2 - R3	24	0,785664	34	0,605827	31	0,779784	4	0,975179	6	0,958657
34	I4 - C3 - R1	19	0,830723	25	0,680911	27	0,795947	22	0,841581	29	0,819919
35	I4 - C3 - R2	11	0,875366	18	0,719082	22	0,843703	13	0,921761	22	0,856682
36	I4 - C3 - R3	29	0,739725	16	0,745352	12	0,904948	2	0,985844	12	0,919472
37	I5 - C1 - R1	44	0,531919	45	0,378405	45	0,50999	44	0,564128	45	0,502791
38	I5 - C1 - R2	41	0,60931	42	0,522021	42	0,610078	43	0,612654	43	0,621856
39	I5 - C1 - R3	37	0,645718	40	0,547088	41	0,634349	41	0,643426	42	0,665044
40	I5 - C2 - R1	45	0,478808	44	0,445628	44	0,563637	45	0,563909	44	0,570657
41	I5 - C2 - R2	40	0,62652	43	0,512981	6	0,933205	17	0,897004	15	0,897604
42	I5 - C2 - R3	6	0,952577	11	0,76624	5	0,957245	11	0,930819	9	0,946789
43	I5 - C3 - R1	27	0,766545	32	0,610283	36	0,723142	26	0,824332	33	0,788434
44	I5 - C3 - R2	12	0,871076	19	0,716769	19	0,858434	16	0,907158	20	0,869139
45	I5 - C3 - R3	9	0,91322	8	0,789346	20	0,853889	6	0,946617	8	0,947198

Table 20 – Rank of the CCR model for the multiplicative approach

25x5	25x7	25x10	35x7	35x10
I1-C1-R3	I1-C2-R3 I2-C3-R3	I1-C2-R3 I2-C3-R3	I1-C2-R3	I1-C2-R3 I4-C1-R3

Table 21 – Efficient combinations for undesired outputs

In this approach, all efficient combinations include the $RefSet$ R3 composed as $b_1 = 70$ and $b_2 = 30$. The random initialization I1 also proved to be more efficient among all the proposed ones.

Comparing Tables 18 and 21, we can see that all the combinations that are efficient for undesired outputs are also efficient for constant output, but the opposite is not valid. Therefore, we can conclude that the approach with undesired outputs refined our quest for the efficient combination to use.

6.3.1 Comparison with the criterion of non-dominance

From Multiobjective Optimization (MO), consider the concepts of the search space partially ordained, in a way that two arbitrary solutions are linked to each other in two possible ways: or one of them dominates the other or neither dominates.

Let ω_1 and ω_2 two solutions (DMUs) in the search space of a problem that has 3 objective functions (f_1 = waiting time, f_2 = handling time and f_3 = computational time). Then ω_1 dominates ω_2 according to (SAWARAGI *et al.*, 1985), if and only if:

$$\begin{gathered}\forall i \in \{1,2,3\}, \qquad f_i(\omega_1) \leq f_i(\omega_2) \\ \text{and} \\ \exists j \in \{1,2,3\}, \qquad f_j(\omega_1) < f_j(\omega_2)\end{gathered}$$

In other words, ω_1 is not worse than ω_2 in any of the objectives and is better in at least one of them (ABRAHAM; JAIN, 2005).

The DMUs that are non-dominated using the criteria previously described are reported in Table 22.

It is noteworthy that all efficient solutions from DEA are non-dominated for MO, but the inverse is not valid. Therefore, the DEA can be used to refine the set of non dominated solutions of MO.

6.3.2 Deviating slightly the parameters

Analyzing Tables 18 and 21 from the combination I1 - C2 - R3, some slight deviations were made in some of the parameters to analyze how the results would be influenced

25x5	25x7	25x10	35x7	35x10
I1-C1-R1	I1-C1-R3	I1-C1-R2	**I1-C2-R3**	I1-C1-R3
I1-C1-R3	I1-C2-R1	I1-C1-R3	I1-C3-R2	I1-C2-R2
I1-C2-R2	**I1-C2-R3**	I1-C2-R1	I3-C2-R2	**I1-C2-R3**
I1-C2-R3	I2-C2-R1	**I1-C2-R3**	I3-C2-R3	I1-C3-R3
I1-C3-R2	I2-C3-R2	I1-C3-R2	I4-C1-R2	I3-C1-R2
I1-C3-R3	**I2-C3-R3**	I2-C2-R3	I4-C1-R3	I3-C1-R3
I2-C1-R1	I3-C1-R3	**I2-C3-R3**	I4-C2-R3	I3-C2-R1
I2-C2-R2	I4-C3-R1	I3-C2-R3	I4-C3-R3	I3-C2-R3
I2-C2-R3	I4-C3-R2	I4-C1-R3	I5-C1-R2	I3-C3-R3
I2-C3-R1	I5-C2-R2	I4-C2-R1	I5-C2-R3	**I4-C1-R3**
I3-C1-R3		I4-C2-R3	I5-C3-R3	I5-C3-R3
I3-C2-R3		I4-C3-R1		
I4-C2-R2		I4-C3-R3		
I5-C3-R1		I5-C1-R2		
I5-C3-R2		I5-C2-R2		
		I5-C2-R3		

Table 22 – Non-dominated combinations

- v1: the probability of horizontal crossover (Section 6.1.3.1) is $\mathrm{P}_h = \frac{1}{7}$, the probability of vertical crossover (Section 6.1.3.2) is $\mathrm{P}_v = \frac{3}{7}$ and the probability of vertical crossover in structure l (Section 6.1.3.3) is $\mathrm{P}_v(l) = \frac{3}{7}$
- v2: the initialization is 0,5*I1 + 0,5*I2
- v3: the initialization is 0,8*I1 + 0,2*I2 and the probability of horizontal crossover (Section 6.1.3.1) is $\mathrm{P}_h = \frac{1}{5}$, the probability of vertical crossover (Section 6.1.3.2) is $\mathrm{P}_v = \frac{2}{5}$ and the probability of vertical crossover in structure l (Section 6.1.3.3) is $\mathrm{P}_v(l) = \frac{2}{5}$
- v4: the initialization is 0,85*I1 + 0,15*I2 and the probability of horizontal crossover (Section 6.1.3.1) is $\mathrm{P}_h = \frac{1}{5}$, the probability of vertical crossover (Section 6.1.3.2) is $\mathrm{P}_v = \frac{2}{5}$ and the probability of vertical crossover in structure l (Section 6.1.3.3) is $\mathrm{P}_v(l) = \frac{2}{5}$
- v5: the initialization is 0,95*I1 + 0,05*I2 and the probability of horizontal crossover (Section 6.1.3.1) is $\mathrm{P}_h = \frac{1}{5}$, the probability of vertical crossover (Section 6.1.3.2) is $\mathrm{P}_v = \frac{2}{5}$ and the probability of vertical crossover in structure l (Section 6.1.3.3) is $\mathrm{P}_v(l) = \frac{2}{5}$

25x5					
CPLEX 12.6	v1	v2	v3	v4	v5
759	760	761	761	759	759
1013	982	986	983	987	976
1039	980	982	980	981	984
721	694	696	697	690	698
973	959	957	958	958	958
1165	1138	1137	1138	1138	1140
872	836	839	851	1150	844
647	630	629	627	628	628
752	754	754	755	755	756
1148	1082	1084	1079	1078	1079

25x7					
CPLEX 12.6	v1	v2	v3	v4	v5
674	659	657	657	657	657
698	659	660	663	662	662
815	812	813	813	808	809
655	655	655	650	656	655
725	726	730	725	729	732
847	796	801	801	801	803
769	748	744	745	741	745
791	780	776	778	775	780
749	750	749	750	751	750
825	826	828	825	825	827

25x10					
CPLEX 12.6	v1	v2	v3	v4	v5
713	715	714	718	716	717
731	729	729	731	729	731
761	762	764	764	764	764
810	820	815	815	814	815
840	840	840	844	842	841
689	692	692	689	689	691
666	669	667	669	668	669
855	855	855	855	855	855
715	716	716	715	713	713
801	801	803	803	801	801

35x7					
CPLEX 12.6	v1	v2	v3	v4	v5
1046	1010	1027	1027	1026	1022
1484	1211	1215	1215	1220	1226
1442	1238	1240	1226	1232	1239
1265	1159	1148	1148	1147	1146
1255	1200	1192	1182	1189	1187
2168	1711	1710	1720	1716	1723
1538	1192	1193	1209	1207	1192
1668	1343	1339	1350	1350	1337
1721	1278	1270	1271	1267	1279
1408	1137	1148	1133	1140	1143

35x10					
CPLEX 12.6	v1	v2	v3	v4	v5
1255	1131	1143	1143	1134	1136
1365	1216	1225	1228	1229	1225
996	950	951	957	947	950
1652	1249	1244	1255	1258	1254
1528	1362	1369	1359	1367	1366
1248	1225	1220	1221	1225	1218
1108	1062	1056	1059	1062	1067
1238	1210	1215	1212	1206	1209
1398	1336	1329	1338	1338	1336
1205	1199	1195	1202	1202	1202

Table 23 – Results of parameter deviations

In table 23 is noted that a small deviation in the proportion of initializations or crossover does not entail significant changes in the value of the objective function.

6.4 Conclusion

As stated previously, the Berth Allocation Problem has shown to be of high resolution complexity, and metaheuristic methods stand out as a faster option to find good solutions. For this reason, this Chapter developed a hybrid method, which combines genetic algorithm with scatter search to solve the problem. One of the main reasons to use a genetic algorithm is that the number of parameters and operators is very large, making the method more robust. The scatter search was used in order to maintain the diversity of solutions.

Because metaheuristic depends on the structure of the problem, there are several ways to implement the genetic algorithm operators. Thus, the initialization, the crossover and the scatter search were implemented in different ways.

The complexity of making the decision based on the empirical analysis increases with the numbers of distinct operators and parameters values and the data envelopment analysis was used to identify the best way to combine those operators in order to obtain the best results for the berth allocation problem. It was possible to measure if the amount of computational time taken compensated the value obtained by the objective function. Three approaches were analyzed and the multiplicative approach proved useful in identifying which operator is more advantageous to use. For future works, we intend to explore such operators to improve the HEABAP performance.

7 Problem generator for the BAP

According to (CHEONG; TAN, 2008), there is no well-established benchmark for the BAP in the literature.

In literature, the Berth Allocation Problem is modeled in different ways. For this reason, there is a lack of appropriate test problems and problem generators to be used by all researchers in their computational experiments. The purpose of developing a data generator is to create benchmark problem instances to allow future work to be broadly and fairly compared.

For the model proposed at Section 2.2.3, the necessary parameters to model the BAP are the following:

- number of vessels: n
- number of berths: m
- vessel i time window: $[a_i, b_i]$
- berth k time window: $[s^k, e^k]$
- processing time of vessel i at berth k: p_i^k
- vessel i relative importance v_i

Due to its extreme versatility, as proposed in (SILVA *et al.*, 2014), the beta distribution is used for the generation of these parameters. The standard beta distribution is a continuous probability distribution with probability density function given by:

$$f(x; \alpha, \beta) = \frac{x^{\alpha-1}(1-x)^{\beta-1}}{\int_0^1 \mu^{\alpha-1}(1-\mu)^{\beta-1}\,\partial\mu},\ 0 \leq x \leq 1,\ \alpha > 0,\ \beta > 0 \tag{7.1}$$

In Figure 24 the shapes the beta distribution assumes for $\alpha = 2$ and $\beta = 2$ (24a), $\alpha = 0.5$ and $\beta = 0.5$ (24b), $\alpha = 2$ and $\beta = 5$ (24c) and $\alpha = 5$ and $\beta = 2$ (24d) are shown.

As in (SILVA *et al.*, 2014), the inverse transform sampling technique was used for generating random numbers from the beta distribution and a uniform pseudo-random generator was implemented to ensure the portability and reproducibility of the test data.

The parameters number of vessels (n) and number of berths (m) are user's input. The berth time window is fixed for every berth and normalized: $s^k = 0$ and e^k is an input.

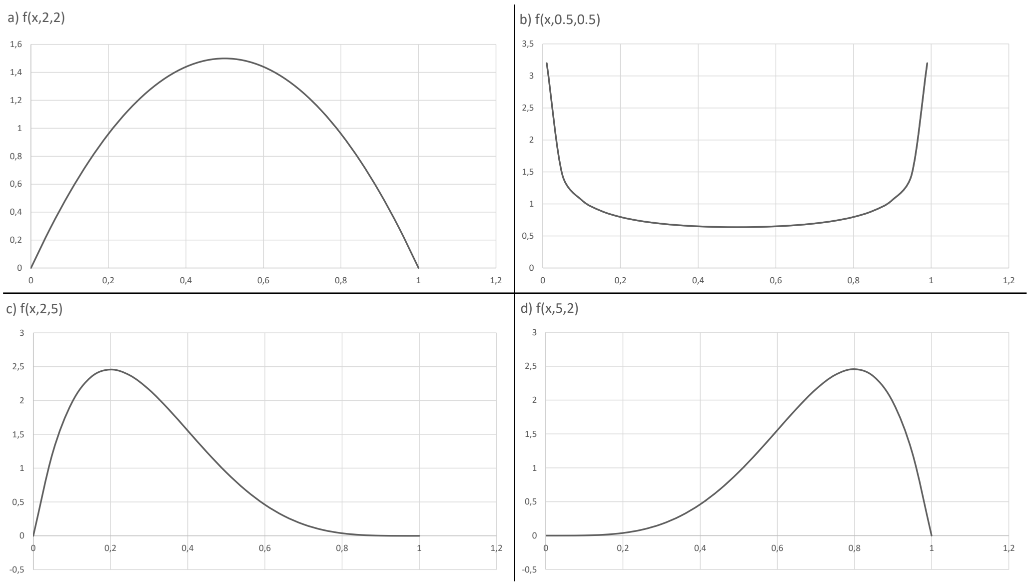


Figure 24 – Beta distributions.

The user has to select one of the four types of beta distributions considered for generating the processing times, the arrival times and the departure times.

For the generation of the values for processing times, first a reference processing time p_ref_i is generated for each vessel in the interval $[1, \frac{m}{n}e^k]$.

If a berth does not have the right equipment to (un)load a vessel, then the vessel can not be moored at that berth. In some cases, there are draft or depth restrictions that forbid a vessel to be moored at a given berth. For these and many others reasons, we need to consider that with a given probability (*prob* - defined as an input by the user) the vessel will not be able to be processed in some berth. A random number *rand* in $(0,1)$ is generated. If $rand \leq prob$, then the processing time of vessel i at berth k is set to $2e^k$, i.e., vessel i can not be moored at berth k. Other wise, p_i^k is generated using the beta distribution $f(x,5,2)$ in the interval $[\frac{1}{2}p\_\text{ref}_i, p\_\text{ref}_i]$.

The arrival times a_i values are generated in the interval $[0, e^k - \max_k p_i^k]$ and the departure times values are generated in the interval $[a_i + \max_k p_i^k, e^k]$.

For the parameters related to the value of relative importance v_i of each vessel i, the user must provide an interval between 0.5 and 1 and a value is then sampled from the Uniform distribution, which is the beta distribution $f(x,1,1)$.

	min	max
p_ref	1	$\frac{m}{n}e^k$
p	$\frac{1}{2}p\_\text{ref}$	p_ref
a	0	$e^k - \max_k p_i^k$
b	$a_i + \max_k p_i^k$	e^k

Table 24 – Data summary

7.1 Computational tests

The computational tests were executed on a personal computer, a Dell Inspiron 14Z with Intel Core I5-3337U 1.80GHz, RAM memory of 6GB and a Solid State Drive of size 240 GB, using CPLEX 12.6. The stopping criteria was computational time, limited to 3600 seconds.

To evaluate the influence of the data on the difficulty of solving the problem with CPLEX, four shapes for the beta distribution are considered: $f(x, 2, 2)$, $f(x, 0.5, 0.5)$, $f(x, 2, 5)$ and $f(x, 5, 2)$ (Figure 24). Because there are 3 parameters (p, a and b), there are 4^3 different combinations of generating the set of parameters for the problem. It was considered $e^k = 100\ \forall k$ and $v_i = 1\ \forall i$. For each combination, 10 instances were generated and solved for all combinations of 20, 30 and 40 vessels with 5, 7 and 10 berths. This totalizes 5760 computational tests solved with CPLEX 12.6. The stopping criteria was computational time, limited in 3600 seconds (one hour).

processing time (p)	arrival time (a)	departure time (b)	solved problems	average comp. time (s)	founded a solution	Average GAP	Unknown Status
$f(x,5,2)$	$f(x,5,2)$	$f(x,2,5)$	10	0,216	0	-	0
$f(x,5,2)$	$f(x,0.5,0.5)$	$f(x,2,5)$	10	0,356	8	-	0
$f(x,0.5,0.5)$	$f(x,0.5,0.5)$	$f(x,2,5)$	10	0,36	9	-	0
$f(x,2,5)$	$f(x,0.5,0.5)$	$f(x,2,5)$	10	0,362	10	-	0
$f(x,2,2)$	$f(x,0.5,0.5)$	$f(x,2,2)$	10	0,424	8	-	0
$f(x,2,5)$	$f(x,0.5,0.5)$	$f(x,0.5,0.5)$	10	0,431	10	-	0
$f(x,5,2)$	$f(x,0.5,0.5)$	$f(x,0.5,0.5)$	10	0,436	8	-	0
$f(x,2,5)$	$f(x,0.5,0.5)$	$f(x,2,2)$	10	0,438	10	-	0
$f(x,2,2)$	$f(x,0.5,0.5)$	$f(x,0.5,0.5)$	10	0,438	8	-	0
$f(x,0.5,0.5)$	$f(x,0.5,0.5)$	$f(x,2,2)$	10	0,438	9	-	0
$f(x,0.5,0.5)$	$f(x,0.5,0.5)$	$f(x,0.5,0.5)$	10	0,453	9	-	0
$f(x,2,2)$	$f(x,0.5,0.5)$	$f(x,2,5)$	10	0,463	8	-	0
$f(x,2,5)$	$f(x,2,2)$	$f(x,2,5)$	10	0,464	10	-	0
$f(x,0.5,0.5)$	$f(x,0.5,0.5)$	$f(x,5,2)$	10	0,472	9	-	0
$f(x,5,2)$	$f(x,0.5,0.5)$	$f(x,2,2)$	10	0,486	8	-	0
$f(x,2,5)$	$f(x,0.5,0.5)$	$f(x,5,2)$	10	0,498	10	-	0
$f(x,2,2)$	$f(x,0.5,0.5)$	$f(x,5,2)$	10	0,514	9	-	0
$f(x,5,2)$	$f(x,0.5,0.5)$	$f(x,5,2)$	10	0,531	8	-	0
$f(x,2,5)$	$f(x,2,2)$	$f(x,0.5,0.5)$	10	0,559	10	-	0
$f(x,2,5)$	$f(x,2,2)$	$f(x,2,2)$	10	0,569	10	-	0
$f(x,2,5)$	$f(x,5,2)$	$f(x,2,5)$	10	0,583	10	-	0
$f(x,2,5)$	$f(x,5,2)$	$f(x,0.5,0.5)$	10	0,607	10	-	0
$f(x,2,2)$	$f(x,2,2)$	$f(x,2,5)$	10	0,637	10	-	0
$f(x,2,5)$	$f(x,2,2)$	$f(x,5,2)$	10	0,684	10	-	0
$f(x,2,5)$	$f(x,5,2)$	$f(x,2,2)$	10	0,782	10	-	0
$f(x,2,2)$	$f(x,2,2)$	$f(x,0.5,0.5)$	10	0,834	10	-	0
$f(x,2,2)$	$f(x,2,2)$	$f(x,2,2)$	10	0,859	10	-	0
$f(x,0.5,0.5)$	$f(x,2,2)$	$f(x,2,5)$	10	0,915	10	-	0
$f(x,2,5)$	$f(x,5,2)$	$f(x,5,2)$	10	0,926	10	-	0
$f(x,2,2)$	$f(x,2,2)$	$f(x,5,2)$	10	0,994	10	-	0
$f(x,0.5,0.5)$	$f(x,2,2)$	$f(x,0.5,0.5)$	10	1,037	10	-	0
$f(x,0.5,0.5)$	$f(x,2,2)$	$f(x,2,2)$	10	1,17	10	-	0
$f(x,2,5)$	$f(x,2,5)$	$f(x,2,2)$	10	1,25	10	-	0
$f(x,0.5,0.5)$	$f(x,2,2)$	$f(x,5,2)$	10	1,291	10	-	0
$f(x,2,2)$	$f(x,5,2)$	$f(x,2,5)$	10	1,775	6	-	0
$f(x,2,5)$	$f(x,2,5)$	$f(x,5,2)$	10	1,776	10	-	0
$f(x,2,5)$	$f(x,2,5)$	$f(x,2,5)$	10	1,856	10	-	0
$f(x,0.5,0.5)$	$f(x,5,2)$	$f(x,2,5)$	10	1,884	8	-	0
$f(x,2,5)$	$f(x,2,5)$	$f(x,0.5,0.5)$	10	2,003	10	-	0
$f(x,0.5,0.5)$	$f(x,5,2)$	$f(x,2,2)$	10	5,286	9	-	0
$f(x,5,2)$	$f(x,5,2)$	$f(x,2,2)$	10	5,673	0	-	0
$f(x,5,2)$	$f(x,2,2)$	$f(x,2,5)$	10	6,297	8	-	0
$f(x,2,2)$	$f(x,5,2)$	$f(x,2,2)$	10	7,699	9	-	0
$f(x,2,2)$	$f(x,5,2)$	$f(x,0.5,0.5)$	10	8,223	8	-	0
$f(x,0.5,0.5)$	$f(x,5,2)$	$f(x,0.5,0.5)$	10	13,075	8	-	0
$f(x,5,2)$	$f(x,2,2)$	$f(x,2,2)$	10	17,788	10	-	0
$f(x,0.5,0.5)$	$f(x,5,2)$	$f(x,5,2)$	10	27,343	9	-	0
$f(x,5,2)$	$f(x,5,2)$	$f(x,0.5,0.5)$	10	29,794	0	-	0
$f(x,2,2)$	$f(x,5,2)$	$f(x,5,2)$	10	35,295	10	-	0
$f(x,5,2)$	$f(x,2,2)$	$f(x,5,2)$	10	46,51	10	-	0
$f(x,5,2)$	$f(x,5,2)$	$f(x,5,2)$	10	74,244	1	-	0
$f(x,0.5,0.5)$	$f(x,2,5)$	$f(x,2,5)$	10	92,344	10	-	0
$f(x,0.5,0.5)$	$f(x,2,5)$	$f(x,0.5,0.5)$	10	103,247	10	-	0
$f(x,5,2)$	$f(x,2,2)$	$f(x,0.5,0.5)$	10	115,738	10	-	0
$f(x,0.5,0.5)$	$f(x,2,5)$	$f(x,2,2)$	9	63,17666667	10	0,00218726	0
$f(x,2,2)$	$f(x,2,5)$	$f(x,2,5)$	9	212,8788889	10	0,0115843	0
$f(x,0.5,0.5)$	$f(x,2,5)$	$f(x,5,2)$	8	5,2975	10	0,00262777	0
$f(x,2,2)$	$f(x,2,5)$	$f(x,0.5,0.5)$	8	466,28625	10	0,006445155	0
$f(x,2,2)$	$f(x,2,5)$	$f(x,2,2)$	7	227,9814286	10	0,00492009	0
$f(x,2,2)$	$f(x,2,5)$	$f(x,5,2)$	6	108,8216667	10	0,00492219	0
$f(x,5,2)$	$f(x,2,5)$	$f(x,2,5)$	4	940,6725	10	0,007966293	0
$f(x,5,2)$	$f(x,2,5)$	$f(x,2,2)$	2	104,11	10	0,010397759	0
$f(x,5,2)$	$f(x,2,5)$	$f(x,0.5,0.5)$	2	107,585	10	0,009646556	0
$f(x,5,2)$	$f(x,2,5)$	$f(x,5,2)$	1	94,94	10	0,010643113	0

Table 25 – Instances tests analysis: 20 vessels and 5 berths

processing time (p)	arrival time (a)	departure time (b)	solved problems	average comp. time (s)	founded a solution	Average GAP	Unknown Status
$f(x,5,2)$	$f(x,5,2)$	$f(x,2,5)$	10	0,37	1	-	0
$f(x,2,5)$	$f(x,0.5,0.5)$	$f(x,2,5)$	10	0,38	10	-	0
$f(x,2,5)$	$f(x,0.5,0.5)$	$f(x,2,2)$	10	0,46	10	-	0
$f(x,2,2)$	$f(x,0.5,0.5)$	$f(x,2,5)$	10	0,47	10	-	0
$f(x,2,5)$	$f(x,2,2)$	$f(x,2,5)$	10	0,48	10	-	0
$f(x,2,5)$	$f(x,0.5,0.5)$	$f(x,0.5,0.5)$	10	0,48	10	-	0
$f(x,2,5)$	$f(x,0.5,0.5)$	$f(x,5,2)$	10	0,50	10	-	0
$f(x,0.5,0.5)$	$f(x,0.5,0.5)$	$f(x,2,5)$	10	0,55	10	-	0
$f(x,2,2)$	$f(x,0.5,0.5)$	$f(x,0.5,0.5)$	10	0,56	10	-	0
$f(x,2,5)$	$f(x,2,2)$	$f(x,0.5,0.5)$	10	0,58	10	-	0
$f(x,2,2)$	$f(x,0.5,0.5)$	$f(x,2,2)$	10	0,59	10	-	0
$f(x,2,2)$	$f(x,5,2)$	$f(x,2,5)$	10	0,62	5	-	0
$f(x,2,2)$	$f(x,0.5,0.5)$	$f(x,5,2)$	10	0,62	10	-	0
$f(x,2,5)$	$f(x,5,2)$	$f(x,2,5)$	10	0,63	10	-	0
$f(x,0.5,0.5)$	$f(x,0.5,0.5)$	$f(x,2,2)$	10	0,64	10	-	0
$f(x,0.5,0.5)$	$f(x,0.5,0.5)$	$f(x,0.5,0.5)$	10	0,65	10	-	0
$f(x,2,5)$	$f(x,2,2)$	$f(x,5,2)$	10	0,66	10	-	0
$f(x,2,5)$	$f(x,2,5)$	$f(x,2,5)$	10	0,66	10	-	0
$f(x,2,5)$	$f(x,2,2)$	$f(x,2,2)$	10	0,69	10	-	0
$f(x,0.5,0.5)$	$f(x,2,2)$	$f(x,2,5)$	10	0,70	10	-	0
$f(x,2,2)$	$f(x,2,2)$	$f(x,2,5)$	10	0,71	10	-	0
$f(x,0.5,0.5)$	$f(x,0.5,0.5)$	$f(x,5,2)$	10	0,76	10	-	0
$f(x,2,5)$	$f(x,2,5)$	$f(x,0.5,0.5)$	10	0,78	10	-	0
$f(x,2,5)$	$f(x,5,2)$	$f(x,0.5,0.5)$	10	0,79	10	-	0
$f(x,2,5)$	$f(x,5,2)$	$f(x,2,2)$	10	0,82	10	-	0
$f(x,2,5)$	$f(x,2,5)$	$f(x,2,2)$	10	0,86	10	-	0
$f(x,5,2)$	$f(x,0.5,0.5)$	$f(x,2,5)$	10	0,88	10	-	0
$f(x,2,5)$	$f(x,5,2)$	$f(x,5,2)$	10	0,93	10	-	0
$f(x,0.5,0.5)$	$f(x,2,2)$	$f(x,0.5,0.5)$	10	0,95	10	-	0
$f(x,0.5,0.5)$	$f(x,5,2)$	$f(x,2,5)$	10	0,95	7	-	0
$f(x,2,5)$	$f(x,2,5)$	$f(x,5,2)$	10	0,96	10	-	0
$f(x,2,2)$	$f(x,2,2)$	$f(x,0.5,0.5)$	10	1,03	10	-	0
$f(x,0.5,0.5)$	$f(x,2,2)$	$f(x,2,2)$	10	1,05	10	-	0
$f(x,5,2)$	$f(x,0.5,0.5)$	$f(x,0.5,0.5)$	10	1,05	10	-	0
$f(x,2,2)$	$f(x,2,2)$	$f(x,2,2)$	10	1,08	10	-	0
$f(x,5,2)$	$f(x,0.5,0.5)$	$f(x,2,2)$	10	1,19	10	-	0
$f(x,5,2)$	$f(x,0.5,0.5)$	$f(x,5,2)$	10	1,19	10	-	0
$f(x,2,2)$	$f(x,2,2)$	$f(x,5,2)$	10	1,22	10	-	0
$f(x,0.5,0.5)$	$f(x,2,2)$	$f(x,5,2)$	10	1,29	10	-	0
$f(x,2,2)$	$f(x,5,2)$	$f(x,2,2)$	10	1,45	9	-	0
$f(x,2,2)$	$f(x,5,2)$	$f(x,0.5,0.5)$	10	1,46	9	-	0
$f(x,2,2)$	$f(x,5,2)$	$f(x,5,2)$	10	2,60	10	-	0
$f(x,5,2)$	$f(x,2,2)$	$f(x,2,5)$	10	4,83	9	-	0
$f(x,5,2)$	$f(x,5,2)$	$f(x,2,2)$	10	7,74	2	-	0
$f(x,0.5,0.5)$	$f(x,5,2)$	$f(x,2,2)$	10	10,14	7	-	0
$f(x,5,2)$	$f(x,5,2)$	$f(x,0.5,0.5)$	10	15,44	4	-	0
$f(x,0.5,0.5)$	$f(x,2,5)$	$f(x,2,5)$	10	20,04	10	-	0
$f(x,0.5,0.5)$	$f(x,5,2)$	$f(x,0.5,0.5)$	10	20,53	8	-	0
$f(x,5,2)$	$f(x,2,2)$	$f(x,0.5,0.5)$	10	28,44	10	-	0
$f(x,0.5,0.5)$	$f(x,2,5)$	$f(x,0.5,0.5)$	10	28,52	10	-	0
$f(x,5,2)$	$f(x,2,2)$	$f(x,2,2)$	10	30,29	10	-	0
$f(x,5,2)$	$f(x,2,2)$	$f(x,5,2)$	10	49,02	10	-	0
$f(x,0.5,0.5)$	$f(x,2,5)$	$f(x,2,2)$	10	69,77	10	-	0
$f(x,0.5,0.5)$	$f(x,2,5)$	$f(x,5,2)$	10	99,51	10	-	0
$f(x,0.5,0.5)$	$f(x,5,2)$	$f(x,5,2)$	9	6,49	9	-	1
$f(x,2,2)$	$f(x,2,5)$	$f(x,0.5,0.5)$	9	6,82	10	0,010	0
$f(x,2,2)$	$f(x,2,5)$	$f(x,2,5)$	9	7,08	10	0,005	0
$f(x,2,2)$	$f(x,2,5)$	$f(x,2,2)$	9	8,24	10	0,008	0
$f(x,5,2)$	$f(x,5,2)$	$f(x,5,2)$	9	13,09	3	-	1
$f(x,2,2)$	$f(x,2,5)$	$f(x,5,2)$	9	15,80	10	0,007	0
$f(x,5,2)$	$f(x,2,5)$	$f(x,2,5)$	6	115,42	8	0,002	1
$f(x,5,2)$	$f(x,2,5)$	$f(x,0.5,0.5)$	5	187,84	10	0,005	0
$f(x,5,2)$	$f(x,2,5)$	$f(x,2,2)$	3	184,59	10	0,007	0
$f(x,5,2)$	$f(x,2,5)$	$f(x,5,2)$	3	192,41	10	0,008	0

Table 26 – Instances tests analysis: 20 vessels and 7 berths

processing time (p)	arrival time (a)	departure time (b)	solved problems	average comp. time (s)	founded a solution	Average GAP	Unknown Status
$f(x,5,2)$	$f(x,5,2)$	$f(x,2,5)$	10	0,431	2	-	0
$f(x,5,2)$	$f(x,0,5,0,5)$	$f(x,2,5)$	10	0,435	10	-	0
$f(x,2,2)$	$f(x,0,5,0,5)$	$f(x,2,5)$	10	0,532	10	-	0
$f(x,2,5)$	$f(x,0,5,0,5)$	$f(x,2,5)$	10	0,572	10	-	0
$f(x,0,5,0,5)$	$f(x,0,5,0,5)$	$f(x,2,5)$	10	0,579	10	-	0
$f(x,5,2)$	$f(x,2,2)$	$f(x,2,5)$	10	0,593333333	8	-	0
$f(x,2,2)$	$f(x,0,5,0,5)$	$f(x,2,2)$	10	0,594	10	-	0
$f(x,5,2)$	$f(x,0,5,0,5)$	$f(x,2,2)$	10	0,63	10	-	0
$f(x,0,5,0,5)$	$f(x,5,2)$	$f(x,2,5)$	10	0,63	8	-	0
$f(x,2,2)$	$f(x,0,5,0,5)$	$f(x,5,2)$	10	0,634	10	-	0
$f(x,5,2)$	$f(x,0,5,0,5)$	$f(x,0,5,0,5)$	10	0,64	10	-	0
$f(x,2,5)$	$f(x,0,5,0,5)$	$f(x,2,2)$	10	0,65	10	-	0
$f(x,2,5)$	$f(x,5,2)$	$f(x,2,5)$	10	0,659	10	-	0
$f(x,2,5)$	$f(x,0,5,0,5)$	$f(x,0,5,0,5)$	10	0,662	10	-	0
$f(x,2,5)$	$f(x,2,2)$	$f(x,2,5)$	10	0,673	10	-	0
$f(x,2,5)$	$f(x,2,2)$	$f(x,2,2)$	10	0,674	10	-	0
$f(x,2,5)$	$f(x,2,2)$	$f(x,0,5,0,5)$	10	0,684	10	-	0
$f(x,2,5)$	$f(x,0,5,0,5)$	$f(x,5,2)$	10	0,704	10	-	0
$f(x,2,2)$	$f(x,2,2)$	$f(x,2,5)$	10	0,708	10	-	0
$f(x,0,5,0,5)$	$f(x,2,2)$	$f(x,2,5)$	10	0,718	10	-	0
$f(x,0,5,0,5)$	$f(x,0,5,0,5)$	$f(x,2,2)$	10	0,741	10	-	0
$f(x,2,2)$	$f(x,0,5,0,5)$	$f(x,0,5,0,5)$	10	0,774	10	-	0
$f(x,0,5,0,5)$	$f(x,0,5,0,5)$	$f(x,0,5,0,5)$	10	0,776	10	-	0
$f(x,5,2)$	$f(x,5,2)$	$f(x,2,2)$	10	0,779	6	-	0
$f(x,2,5)$	$f(x,5,2)$	$f(x,0,5,0,5)$	10	0,796	10	-	0
$f(x,2,5)$	$f(x,2,2)$	$f(x,5,2)$	10	0,82	10	-	0
$f(x,2,5)$	$f(x,5,2)$	$f(x,2,2)$	10	0,828	10	-	0
$f(x,0,5,0,5)$	$f(x,2,2)$	$f(x,0,5,0,5)$	10	0,857	10	-	0
$f(x,2,5)$	$f(x,2,5)$	$f(x,0,5,0,5)$	10	0,858	10	-	0
$f(x,2,2)$	$f(x,2,2)$	$f(x,0,5,0,5)$	10	0,867	10	-	0
$f(x,2,5)$	$f(x,2,5)$	$f(x,2,5)$	10	0,876	10	-	0
$f(x,2,5)$	$f(x,5,2)$	$f(x,5,2)$	10	0,886	10	-	0
$f(x,5,2)$	$f(x,0,5,0,5)$	$f(x,5,2)$	10	0,912	10	-	0
$f(x,0,5,0,5)$	$f(x,0,5,0,5)$	$f(x,5,2)$	10	0,926	10	-	0
$f(x,2,2)$	$f(x,2,2)$	$f(x,2,2)$	10	0,936	10	-	0
$f(x,2,5)$	$f(x,2,5)$	$f(x,2,2)$	10	0,94	10	-	0
$f(x,5,2)$	$f(x,2,5)$	$f(x,2,5)$	10	0,982	9	-	0
$f(x,5,2)$	$f(x,2,2)$	$f(x,0,5,0,5)$	10	1,011	10	-	0
$f(x,0,5,0,5)$	$f(x,2,2)$	$f(x,2,2)$	10	1,017	10	-	0
$f(x,2,2)$	$f(x,5,2)$	$f(x,2,5)$	10	1,023	9	-	0
$f(x,5,2)$	$f(x,2,2)$	$f(x,2,2)$	10	1,076	10	-	0
$f(x,2,5)$	$f(x,2,5)$	$f(x,5,2)$	10	1,095	10	-	0
$f(x,2,2)$	$f(x,2,2)$	$f(x,5,2)$	10	1,108	10	-	0
$f(x,2,2)$	$f(x,5,2)$	$f(x,0,5,0,5)$	10	1,143	9	-	0
$f(x,2,2)$	$f(x,5,2)$	$f(x,2,2)$	10	1,163	9	-	0
$f(x,0,5,0,5)$	$f(x,2,2)$	$f(x,5,2)$	10	1,169	10	-	0
$f(x,0,5,0,5)$	$f(x,5,2)$	$f(x,2,2)$	10	1,297	9	-	0
$f(x,5,2)$	$f(x,2,2)$	$f(x,5,2)$	10	1,395	10	-	0
$f(x,5,2)$	$f(x,5,2)$	$f(x,0,5,0,5)$	10	1,429	5	-	0
$f(x,2,2)$	$f(x,5,2)$	$f(x,5,2)$	10	1,505	10	-	0
$f(x,2,2)$	$f(x,2,5)$	$f(x,2,5)$	10	1,54	10	-	0
$f(x,0,5,0,5)$	$f(x,5,2)$	$f(x,0,5,0,5)$	10	1,635	9	-	0
$f(x,0,5,0,5)$	$f(x,2,5)$	$f(x,0,5,0,5)$	10	1,855	10	-	0
$f(x,2,2)$	$f(x,2,5)$	$f(x,0,5,0,5)$	10	1,888	10	-	0
$f(x,5,2)$	$f(x,5,2)$	$f(x,5,2)$	10	1,9	7	-	0
$f(x,2,2)$	$f(x,2,5)$	$f(x,2,2)$	10	1,903	10	-	0
$f(x,0,5,0,5)$	$f(x,2,5)$	$f(x,2,5)$	10	2,261	10	-	0
$f(x,2,2)$	$f(x,2,5)$	$f(x,5,2)$	10	2,439	10	-	0
$f(x,5,2)$	$f(x,2,5)$	$f(x,2,2)$	10	2,65	10	-	0
$f(x,0,5,0,5)$	$f(x,2,5)$	$f(x,5,2)$	10	3,216	10	-	0
$f(x,0,5,0,5)$	$f(x,2,5)$	$f(x,2,2)$	10	3,66	10	-	0
$f(x,5,2)$	$f(x,2,5)$	$f(x,5,2)$	10	3,809	10	-	0
$f(x,5,2)$	$f(x,2,5)$	$f(x,0,5,0,5)$	10	8,976	10	-	0
$f(x,0,5,0,5)$	$f(x,5,2)$	$f(x,5,2)$	10	12,951	9	-	0

Table 27 – Instances tests analysis: 20 vessels and 10 berths

processing time (p)	arrival time (a)	departure time (b)	solved problems	average comp. time (s)	founded a solution	Average GAP	Unknown Status
$f(x,2,5)$	$f(x,0,5,0,5)$	$f(x,2,5)$	10	0,502	10	-	0
$f(x,2,5)$	$f(x,0,5,0,5)$	$f(x,2,2)$	10	0,649	10	-	0
$f(x,2,5)$	$f(x,0,5,0,5)$	$f(x,0,5,0,5)$	10	0,695	10	-	0
$f(x,2,2)$	$f(x,0,5,0,5)$	$f(x,2,5)$	10	0,73	10	-	0
$f(x,2,5)$	$f(x,2,2)$	$f(x,2,5)$	10	0,769	10	-	0
$f(x,2,5)$	$f(x,0,5,0,5)$	$f(x,5,2)$	10	0,821	10	-	0
$f(x,2,5)$	$f(x,2,2)$	$f(x,0,5,0,5)$	10	0,909	10	-	0
$f(x,2,5)$	$f(x,2,5)$	$f(x,0,5,0,5)$	10	1,017	10	-	0
$f(x,2,5)$	$f(x,2,2)$	$f(x,2,2)$	10	1,083	10	-	0
$f(x,0,5,0,5)$	$f(x,0,5,0,5)$	$f(x,2,5)$	10	1,096	8	-	0
$f(x,2,2)$	$f(x,0,5,0,5)$	$f(x,0,5,0,5)$	10	1,113	10	-	0
$f(x,5,2)$	$f(x,0,5,0,5)$	$f(x,2,5)$	10	1,172	7	-	0
$f(x,2,2)$	$f(x,0,5,0,5)$	$f(x,2,2)$	10	1,177	10	-	0
$f(x,2,5)$	$f(x,2,5)$	$f(x,2,5)$	10	1,182	10	-	0
$f(x,2,5)$	$f(x,5,2)$	$f(x,2,5)$	10	1,202	10	-	0
$f(x,2,5)$	$f(x,2,2)$	$f(x,5,2)$	10	1,308	10	-	0
$f(x,2,2)$	$f(x,0,5,0,5)$	$f(x,5,2)$	10	1,318	10	-	0
$f(x,2,5)$	$f(x,5,2)$	$f(x,0,5,0,5)$	10	1,396	10	-	0
$f(x,5,2)$	$f(x,0,5,0,5)$	$f(x,2,2)$	10	1,439	7	-	0
$f(x,0,5,0,5)$	$f(x,0,5,0,5)$	$f(x,0,5,0,5)$	10	1,468	9	-	0
$f(x,5,2)$	$f(x,0,5,0,5)$	$f(x,0,5,0,5)$	10	1,505	7	-	0
$f(x,2,5)$	$f(x,5,2)$	$f(x,2,2)$	10	1,514	10	-	0
$f(x,2,5)$	$f(x,2,5)$	$f(x,5,2)$	10	1,607	10	-	0
$f(x,2,5)$	$f(x,2,5)$	$f(x,2,2)$	10	1,662	10	-	0
$f(x,5,2)$	$f(x,0,5,0,5)$	$f(x,5,2)$	10	1,692	7	-	0
$f(x,0,5,0,5)$	$f(x,0,5,0,5)$	$f(x,5,2)$	10	1,872	10	-	0
$f(x,0,5,0,5)$	$f(x,0,5,0,5)$	$f(x,2,2)$	10	1,946	9	-	0
$f(x,2,2)$	$f(x,2,2)$	$f(x,2,5)$	10	1,967	10	-	0
$f(x,2,5)$	$f(x,5,2)$	$f(x,5,2)$	10	2,31	10	-	0
$f(x,2,2)$	$f(x,2,2)$	$f(x,2,2)$	10	2,804	10	-	0
$f(x,2,2)$	$f(x,2,2)$	$f(x,0,5,0,5)$	10	3,032	10	-	0
$f(x,2,2)$	$f(x,2,2)$	$f(x,5,2)$	10	4,423	10	-	0
$f(x,0,5,0,5)$	$f(x,2,2)$	$f(x,0,5,0,5)$	10	11,172	10	-	0
$f(x,0,5,0,5)$	$f(x,2,2)$	$f(x,2,5)$	10	11,83	10	-	0
$f(x,0,5,0,5)$	$f(x,2,2)$	$f(x,2,2)$	10	17,288	10	-	0
$f(x,0,5,0,5)$	$f(x,2,2)$	$f(x,5,2)$	10	20,956	10	-	0
$f(x,5,2)$	$f(x,5,2)$	$f(x,2,5)$	10	41,195	0	-	0
$f(x,2,2)$	$f(x,5,2)$	$f(x,2,5)$	10	101,453	8	-	0
$f(x,0,5,0,5)$	$f(x,5,2)$	$f(x,2,5)$	10	273,081	3	-	0
$f(x,2,2)$	$f(x,5,2)$	$f(x,0,5,0,5)$	8	15,51375	8	-	2
$f(x,2,2)$	$f(x,5,2)$	$f(x,2,2)$	8	43,17375	8	-	2
$f(x,2,2)$	$f(x,5,2)$	$f(x,5,2)$	8	60,34625	8	-	2
$f(x,2,2)$	$f(x,2,5)$	$f(x,2,5)$	8	299,295	10	0,002553855	0
$f(x,2,2)$	$f(x,2,5)$	$f(x,0,5,0,5)$	7	38,83428571	10	0,002158607	0
$f(x,5,2)$	$f(x,2,2)$	$f(x,2,5)$	7	52,20857143	8	0,00321897	2
$f(x,2,2)$	$f(x,2,5)$	$f(x,2,2)$	7	71,40571429	10	0,001766614	0
$f(x,0,5,0,5)$	$f(x,2,5)$	$f(x,0,5,0,5)$	7	81,28	10	0,005727813	0
$f(x,2,2)$	$f(x,2,5)$	$f(x,5,2)$	7	93,07428571	10	0,002179654	0
$f(x,0,5,0,5)$	$f(x,2,5)$	$f(x,2,5)$	7	144,9471429	10	0,004873342	0
$f(x,5,2)$	$f(x,2,2)$	$f(x,0,5,0,5)$	7	189,2857143	10	0,01303839	0
$f(x,5,2)$	$f(x,5,2)$	$f(x,2,2)$	7	223,0728571	0	-	3
$f(x,5,2)$	$f(x,2,2)$	$f(x,2,2)$	7	316,1142857	10	0,011449853	0
$f(x,0,5,0,5)$	$f(x,5,2)$	$f(x,2,2)$	7	337,6685714	9	0,00427853	1
$f(x,5,2)$	$f(x,2,2)$	$f(x,5,2)$	7	612,2957143	10	0,009459883	0
$f(x,0,5,0,5)$	$f(x,2,5)$	$f(x,2,2)$	6	520,9383333	10	0,00413608	0
$f(x,0,5,0,5)$	$f(x,5,2)$	$f(x,5,2)$	6	65,77833333	10	0,002837442	0
$f(x,0,5,0,5)$	$f(x,5,2)$	$f(x,0,5,0,5)$	5	56,472	7	0,001824797	2
$f(x,0,5,0,5)$	$f(x,2,5)$	$f(x,5,2)$	5	70,264	10	0,003703622	0
$f(x,5,2)$	$f(x,5,2)$	$f(x,0,5,0,5)$	5	193,824	0	-	5
$f(x,5,2)$	$f(x,5,2)$	$f(x,5,2)$	3	339,2066667	2	0,001640705	5
$f(x,5,2)$	$f(x,2,5)$	$f(x,5,2)$	0	-	10	0,016219735	0
$f(x,5,2)$	$f(x,2,5)$	$f(x,2,2)$	0	-	10	0,016454539	0
$f(x,5,2)$	$f(x,2,5)$	$f(x,0,5,0,5)$	0	-	10	0,019171329	0
$f(x,5,2)$	$f(x,2,5)$	$f(x,2,5)$	0	-	8	0,013717703	2

Table 28 – Instances tests analysis: 30 vessels and 5 berths

processing time (p)	arrival time (a)	departure time (b)	solved problems	average comp. time (s)	founded a solution	Average GAP	Unknown Status
$f(x,2,5)$	$f(x,0.5,0.5)$	$f(x,2,5)$	10	0,89	10	-	0
$f(x,2,5)$	$f(x,0.5,0.5)$	$f(x,0.5,0.5)$	10	1,03	10	-	0
$f(x,2,5)$	$f(x,0.5,0.5)$	$f(x,2,2)$	10	1,19	10	-	0
$f(x,2,5)$	$f(x,2,2)$	$f(x,2,5)$	10	1,22	10	-	0
$f(x,2,5)$	$f(x,0.5,0.5)$	$f(x,5,2)$	10	1,28	10	-	0
$f(x,2,2)$	$f(x,0.5,0.5)$	$f(x,2,5)$	10	1,42	10	-	0
$f(x,2,5)$	$f(x,2,2)$	$f(x,0.5,0.5)$	10	1,42	10	-	0
$f(x,2,2)$	$f(x,0.5,0.5)$	$f(x,0.5,0.5)$	10	1,62	10	-	0
$f(x,2,5)$	$f(x,2,2)$	$f(x,2,2)$	10	1,66	10	-	0
$f(x,2,2)$	$f(x,0.5,0.5)$	$f(x,5,2)$	10	1,8	10	-	0
$f(x,2,2)$	$f(x,0.5,0.5)$	$f(x,2,2)$	10	1,82	10	-	0
$f(x,2,5)$	$f(x,2,5)$	$f(x,0.5,0.5)$	10	1,83	10	-	0
$f(x,2,5)$	$f(x,2,5)$	$f(x,2,5)$	10	1,93	10	-	0
$f(x,2,5)$	$f(x,2,2)$	$f(x,5,2)$	10	2,08	10	-	0
$f(x,2,5)$	$f(x,5,2)$	$f(x,2,5)$	10	2,12	10	-	0
$f(x,2,5)$	$f(x,2,5)$	$f(x,2,2)$	10	2,16	10	-	0
$f(x,2,5)$	$f(x,2,5)$	$f(x,5,2)$	10	2,64	10	-	0
$f(x,2,2)$	$f(x,2,2)$	$f(x,2,5)$	10	2,93	10	-	0
$f(x,2,5)$	$f(x,5,2)$	$f(x,0.5,0.5)$	10	2,95	10	-	0
$f(x,2,5)$	$f(x,5,2)$	$f(x,2,2)$	10	3,16	10	-	0
$f(x,2,2)$	$f(x,2,2)$	$f(x,2,2)$	10	3,26	10	-	0
$f(x,2,5)$	$f(x,5,2)$	$f(x,5,2)$	10	3,69	10	-	0
$f(x,2,2)$	$f(x,2,2)$	$f(x,0.5,0.5)$	10	4,48	10	-	0
$f(x,2,2)$	$f(x,2,2)$	$f(x,5,2)$	10	4,75	10	-	0
$f(x,5,2)$	$f(x,0.5,0.5)$	$f(x,0.5,0.5)$	10	4,77	9	-	0
$f(x,5,2)$	$f(x,0.5,0.5)$	$f(x,2,5)$	10	5,44	9	-	0
$f(x,5,2)$	$f(x,0.5,0.5)$	$f(x,2,2)$	10	5,64	9	-	0
$f(x,0.5,0.5)$	$f(x,0.5,0.5)$	$f(x,2,5)$	10	5,85	10	-	0
$f(x,5,2)$	$f(x,0.5,0.5)$	$f(x,5,2)$	10	6,61	9	-	0
$f(x,0.5,0.5)$	$f(x,0.5,0.5)$	$f(x,0.5,0.5)$	10	6,65	10	-	0
$f(x,0.5,0.5)$	$f(x,0.5,0.5)$	$f(x,2,2)$	10	6,78	10	-	0
$f(x,0.5,0.5)$	$f(x,2,2)$	$f(x,0.5,0.5)$	10	10,88	10	-	0
$f(x,0.5,0.5)$	$f(x,0.5,0.5)$	$f(x,5,2)$	10	13,38	10	-	0
$f(x,0.5,0.5)$	$f(x,2,2)$	$f(x,2,5)$	10	14,45	10	-	0
$f(x,0.5,0.5)$	$f(x,2,2)$	$f(x,2,2)$	10	17,7	10	-	0
$f(x,5,2)$	$f(x,5,2)$	$f(x,2,5)$	10	19,7	0	-	0
$f(x,0.5,0.5)$	$f(x,2,2)$	$f(x,5,2)$	10	21,83	10	-	0
$f(x,0.5,0.5)$	$f(x,5,2)$	$f(x,5,2)$	10	50,81	10	-	0
$f(x,0.5,0.5)$	$f(x,5,2)$	$f(x,2,5)$	10	89,4	9	-	0
$f(x,0.5,0.5)$	$f(x,5,2)$	$f(x,2,2)$	9	14,66	10	0,0013	0
$f(x,0.5,0.5)$	$f(x,5,2)$	$f(x,0.5,0.5)$	9	15,13	10	0,0017	0
$f(x,2,2)$	$f(x,2,5)$	$f(x,2,5)$	9	29,82	10	0,0047	0
$f(x,2,2)$	$f(x,2,5)$	$f(x,2,2)$	9	50,5	10	0,0044	0
$f(x,2,2)$	$f(x,2,5)$	$f(x,5,2)$	9	55,77	10	0,0048	0
$f(x,2,2)$	$f(x,2,5)$	$f(x,0.5,0.5)$	9	62,29	10	0,0053	0
$f(x,2,2)$	$f(x,5,2)$	$f(x,2,5)$	9	125,14	10	0,0009	0
$f(x,0.5,0.5)$	$f(x,2,5)$	$f(x,0.5,0.5)$	9	255,44	10	0,0059	0
$f(x,2,2)$	$f(x,5,2)$	$f(x,0.5,0.5)$	8	47,52	10	0,0013	0
$f(x,2,2)$	$f(x,5,2)$	$f(x,5,2)$	8	65,57	10	0,0018	0
$f(x,2,2)$	$f(x,5,2)$	$f(x,2,2)$	8	70,28	10	0,0014	0
$f(x,0.5,0.5)$	$f(x,2,5)$	$f(x,2,5)$	8	173,54	10	0,0039	0
$f(x,0.5,0.5)$	$f(x,2,5)$	$f(x,5,2)$	8	232,9	10	0,0041	0
$f(x,0.5,0.5)$	$f(x,2,5)$	$f(x,2,2)$	8	280,04	10	0,0034	0
$f(x,5,2)$	$f(x,2,2)$	$f(x,2,5)$	7	240,3	10	0,0034	0
$f(x,5,2)$	$f(x,2,2)$	$f(x,2,2)$	6	184,24	10	0,0027	0
$f(x,5,2)$	$f(x,2,2)$	$f(x,5,2)$	6	231,09	10	0,0032	0
$f(x,5,2)$	$f(x,2,2)$	$f(x,0.5,0.5)$	6	388,25	10	0,0027	0
$f(x,5,2)$	$f(x,5,2)$	$f(x,2,2)$	3	697,45	0	-	7
$f(x,5,2)$	$f(x,5,2)$	$f(x,0.5,0.5)$	1	138,23	0	-	9
$f(x,5,2)$	$f(x,2,5)$	$f(x,5,2)$	0	3600,00	10	0,0151	0
$f(x,5,2)$	$f(x,2,5)$	$f(x,2,2)$	0	3600,00	10	0,0163	0
$f(x,5,2)$	$f(x,2,5)$	$f(x,0.5,0.5)$	0	3600,00	10	0,017	0
$f(x,5,2)$	$f(x,2,5)$	$f(x,2,5)$	0	3600,00	8	0,0166	2
$f(x,5,2)$	$f(x,5,2)$	$f(x,5,2)$	0	3600,00	0	-	10

Table 29 – Instances tests analysis: 30 vessels and 7 berths

processing time (p)	arrival time (a)	departure time (b)	solved problems	average comp. time (s)	founded a solution	Average GAP	Unknown Status
$f(x,2,5)$	$f(x,0,5,0,5)$	$f(x,2,5)$	10	0,874	10	-	0
$f(x,0,5,0,5)$	$f(x,0,5,0,5)$	$f(x,2,5)$	10	1,041	10	-	0
$f(x,2,2)$	$f(x,0,5,0,5)$	$f(x,2,5)$	10	1,056	10	-	0
$f(x,2,5)$	$f(x,0,5,0,5)$	$f(x,0,5,0,5)$	10	1,094	10	-	0
$f(x,0,5,0,5)$	$f(x,0,5,0,5)$	$f(x,0,5,0,5)$	10	1,18	10	-	0
$f(x,2,2)$	$f(x,0,5,0,5)$	$f(x,0,5,0,5)$	10	1,221	10	-	0
$f(x,2,5)$	$f(x,0,5,0,5)$	$f(x,2,2)$	10	1,286	10	-	0
$f(x,5,2)$	$f(x,0,5,0,5)$	$f(x,2,5)$	10	1,362	10	-	0
$f(x,5,2)$	$f(x,0,5,0,5)$	$f(x,0,5,0,5)$	10	1,433	10	-	0
$f(x,2,2)$	$f(x,0,5,0,5)$	$f(x,2,2)$	10	1,463	10	-	0
$f(x,2,2)$	$f(x,0,5,0,5)$	$f(x,5,2)$	10	1,49	10	-	0
$f(x,0,5,0,5)$	$f(x,0,5,0,5)$	$f(x,2,2)$	10	1,547	10	-	0
$f(x,0,5,0,5)$	$f(x,0,5,0,5)$	$f(x,5,2)$	10	1,639	10	-	0
$f(x,2,5)$	$f(x,2,2)$	$f(x,2,5)$	10	1,682	10	-	0
$f(x,2,5)$	$f(x,0,5,0,5)$	$f(x,5,2)$	10	1,732	10	-	0
$f(x,2,5)$	$f(x,5,2)$	$f(x,2,5)$	10	1,828	10	-	0
$f(x,5,2)$	$f(x,0,5,0,5)$	$f(x,2,2)$	10	1,854	10	-	0
$f(x,2,5)$	$f(x,2,2)$	$f(x,0,5,0,5)$	10	1,916	10	-	0
$f(x,5,2)$	$f(x,0,5,0,5)$	$f(x,5,2)$	10	2,003	10	-	0
$f(x,2,5)$	$f(x,2,2)$	$f(x,2,2)$	10	2,103	10	-	0
$f(x,2,5)$	$f(x,5,2)$	$f(x,0,5,0,5)$	10	2,115	10	-	0
$f(x,2,5)$	$f(x,2,5)$	$f(x,2,5)$	10	2,204	10	-	0
$f(x,2,5)$	$f(x,2,5)$	$f(x,0,5,0,5)$	10	2,284	10	-	0
$f(x,2,5)$	$f(x,2,2)$	$f(x,5,2)$	10	2,332	10	-	0
$f(x,2,5)$	$f(x,2,5)$	$f(x,2,2)$	10	2,375	10	-	0
$f(x,2,2)$	$f(x,2,2)$	$f(x,2,5)$	10	2,508	10	-	0
$f(x,2,5)$	$f(x,5,2)$	$f(x,2,2)$	10	2,597	10	-	0
$f(x,2,5)$	$f(x,2,5)$	$f(x,5,2)$	10	3,07	10	-	0
$f(x,2,5)$	$f(x,5,2)$	$f(x,5,2)$	10	3,402	10	-	0
$f(x,2,2)$	$f(x,2,2)$	$f(x,0,5,0,5)$	10	3,886	10	-	0
$f(x,0,5,0,5)$	$f(x,2,2)$	$f(x,2,5)$	10	3,985	10	-	0
$f(x,2,2)$	$f(x,2,2)$	$f(x,2,2)$	10	4,784	10	-	0
$f(x,0,5,0,5)$	$f(x,2,2)$	$f(x,0,5,0,5)$	10	5,435	10	-	0
$f(x,2,2)$	$f(x,2,2)$	$f(x,5,2)$	10	5,713	10	-	0
$f(x,0,5,0,5)$	$f(x,2,2)$	$f(x,2,2)$	10	6,809	10	-	0
$f(x,0,5,0,5)$	$f(x,2,5)$	$f(x,2,5)$	10	8,603	10	-	0
$f(x,2,2)$	$f(x,5,2)$	$f(x,2,5)$	10	9,648	9	-	0
$f(x,0,5,0,5)$	$f(x,2,5)$	$f(x,0,5,0,5)$	10	10,475	10	-	0
$f(x,0,5,0,5)$	$f(x,2,2)$	$f(x,5,2)$	10	11,591	10	-	0
$f(x,0,5,0,5)$	$f(x,2,5)$	$f(x,2,2)$	10	16,245	10	-	0
$f(x,0,5,0,5)$	$f(x,2,5)$	$f(x,5,2)$	10	17,166	10	-	0
$f(x,2,2)$	$f(x,2,5)$	$f(x,2,2)$	10	21,174	10	-	0
$f(x,2,2)$	$f(x,2,5)$	$f(x,2,5)$	10	22,436	10	-	0
$f(x,2,2)$	$f(x,2,5)$	$f(x,5,2)$	10	25,408	10	-	0
$f(x,0,5,0,5)$	$f(x,5,2)$	$f(x,2,5)$	10	29,813	8	-	0
$f(x,2,2)$	$f(x,2,5)$	$f(x,0,5,0,5)$	10	32,062	10	-	0
$f(x,5,2)$	$f(x,5,2)$	$f(x,2,5)$	10	60,972	2	-	0
$f(x,0,5,0,5)$	$f(x,5,2)$	$f(x,0,5,0,5)$	10	82,549	9	-	0
$f(x,0,5,0,5)$	$f(x,5,2)$	$f(x,2,2)$	10	118,409	9	-	0
$f(x,0,5,0,5)$	$f(x,5,2)$	$f(x,5,2)$	10	144,465	10	-	0
$f(x,5,2)$	$f(x,2,2)$	$f(x,0,5,0,5)$	9	4,786666667	10	0,000214812	0
$f(x,5,2)$	$f(x,2,2)$	$f(x,2,2)$	9	5,022222222	10	0,000272662	0
$f(x,5,2)$	$f(x,2,2)$	$f(x,5,2)$	9	7,758888889	10	0,000320907	0
$f(x,2,2)$	$f(x,5,2)$	$f(x,0,5,0,5)$	9	17,18555556	9	-	1
$f(x,2,2)$	$f(x,5,2)$	$f(x,2,2)$	9	19,11111111	9	-	1
$f(x,2,2)$	$f(x,5,2)$	$f(x,5,2)$	9	28,16555556	10	0,00479171	0
$f(x,5,2)$	$f(x,2,2)$	$f(x,2,5)$	9	326,4511111	10	-	0
$f(x,5,2)$	$f(x,5,2)$	$f(x,2,2)$	8	341,71625	3	0,00125925	1
$f(x,5,2)$	$f(x,5,2)$	$f(x,0,5,0,5)$	6	45,61833333	4	0,00124277	2
$f(x,5,2)$	$f(x,5,2)$	$f(x,5,2)$	4	102,6675	4	0,000359706	4
$f(x,5,2)$	$f(x,2,5)$	$f(x,2,5)$	2	461,27	10	0,005041028	0
$f(x,5,2)$	$f(x,2,5)$	$f(x,0,5,0,5)$	2	1358,89	10	0,005203918	0
$f(x,5,2)$	$f(x,2,5)$	$f(x,2,2)$	2	1781,705	10	0,005371266	0
$f(x,5,2)$	$f(x,2,5)$	$f(x,5,2)$	1	53,28	10	0,004540389	0

Table 30 – Instances tests analysis: 30 vessels and 10 berths

processing time (p)	arrival time (a)	departure time (b)	solved problems	average comp. time (s)	founded a solution	Average GAP	Unknown Status
$f(x,0,5,0,5)$	$f(x,0,5,0,5)$	$f(x,2,5)$	10	1,227	6	-	0
$f(x,2,2)$	$f(x,0,5,0,5)$	$f(x,2,5)$	10	1,429	8	-	0
$f(x,2,5)$	$f(x,0,5,0,5)$	$f(x,2,5)$	10	1,62	9	-	0
$f(x,0,5,0,5)$	$f(x,0,5,0,5)$	$f(x,0,5,0,5)$	10	1,711	8	-	0
$f(x,2,5)$	$f(x,0,5,0,5)$	$f(x,2,2)$	10	1,825	9	-	0
$f(x,0,5,0,5)$	$f(x,0,5,0,5)$	$f(x,2,2)$	10	1,843	7	-	0
$f(x,2,2)$	$f(x,0,5,0,5)$	$f(x,2,2)$	10	1,936	8	-	0
$f(x,2,2)$	$f(x,0,5,0,5)$	$f(x,0,5,0,5)$	10	1,984	8	-	0
$f(x,2,5)$	$f(x,2,2)$	$f(x,2,5)$	10	2,069	10	-	0
$f(x,2,5)$	$f(x,0,5,0,5)$	$f(x,0,5,0,5)$	10	2,071	9	-	0
$f(x,0,5,0,5)$	$f(x,0,5,0,5)$	$f(x,5,2)$	10	2,372	9	-	0
$f(x,2,2)$	$f(x,0,5,0,5)$	$f(x,5,2)$	10	2,375	8	-	0
$f(x,2,5)$	$f(x,2,2)$	$f(x,0,5,0,5)$	10	2,527	10	-	0
$f(x,2,5)$	$f(x,0,5,0,5)$	$f(x,5,2)$	10	2,627	9	-	0
$f(x,2,5)$	$f(x,2,2)$	$f(x,2,2)$	10	2,778	10	-	0
$f(x,2,5)$	$f(x,2,2)$	$f(x,5,2)$	10	3,798	10	-	0
$f(x,2,5)$	$f(x,2,5)$	$f(x,2,5)$	10	4,158	10	-	0
$f(x,2,5)$	$f(x,2,5)$	$f(x,0,5,0,5)$	10	4,381	10	-	0
$f(x,2,5)$	$f(x,5,2)$	$f(x,2,5)$	10	4,405	10	-	0
$f(x,2,5)$	$f(x,5,2)$	$f(x,0,5,0,5)$	10	5,277	10	-	0
$f(x,2,5)$	$f(x,2,5)$	$f(x,5,2)$	10	5,373	10	-	0
$f(x,5,2)$	$f(x,0,5,0,5)$	$f(x,5,2)$	10	5,555	4	-	0
$f(x,5,2)$	$f(x,0,5,0,5)$	$f(x,0,5,0,5)$	10	14,54	4	-	0
$f(x,2,5)$	$f(x,5,2)$	$f(x,5,2)$	10	23,888	10	-	0
$f(x,2,2)$	$f(x,2,2)$	$f(x,0,5,0,5)$	10	90,714	10	-	0
$f(x,2,2)$	$f(x,2,2)$	$f(x,2,5)$	10	102,286	10	-	0
$f(x,2,2)$	$f(x,2,2)$	$f(x,2,2)$	10	147,369	10	-	0
$f(x,5,2)$	$f(x,0,5,0,5)$	$f(x,2,5)$	9	1,19	4	-	0
$f(x,2,5)$	$f(x,2,5)$	$f(x,2,2)$	9	4,546666667	10	0,00316727	0
$f(x,2,5)$	$f(x,5,2)$	$f(x,2,2)$	9	8,05	9	-	1
$f(x,0,5,0,5)$	$f(x,2,2)$	$f(x,0,5,0,5)$	9	8,506666667	10	0,00123795	0
$f(x,0,5,0,5)$	$f(x,2,2)$	$f(x,2,5)$	9	9,108888889	10	0,000959358	0
$f(x,0,5,0,5)$	$f(x,2,2)$	$f(x,5,2)$	9	12,79888889	10	0,00116806	0
$f(x,2,2)$	$f(x,2,2)$	$f(x,5,2)$	9	16,45444444	10	0,00015135	0
$f(x,5,2)$	$f(x,0,5,0,5)$	$f(x,2,2)$	9	33,32888889	5	-	0
$f(x,0,5,0,5)$	$f(x,5,2)$	$f(x,2,5)$	9	45,79111111	5	-	1
$f(x,0,5,0,5)$	$f(x,2,2)$	$f(x,2,2)$	8	10,64625	10	0,000795039	0
$f(x,5,2)$	$f(x,2,2)$	$f(x,2,5)$	7	100,2771429	9	0,00266342	1
$f(x,2,2)$	$f(x,2,5)$	$f(x,2,5)$	7	118,1328571	10	0,003995563	0
$f(x,2,2)$	$f(x,2,5)$	$f(x,0,5,0,5)$	7	155,9814286	10	0,00360785	0
$f(x,2,2)$	$f(x,2,5)$	$f(x,5,2)$	7	291,7371429	10	0,00356527	0
$f(x,2,2)$	$f(x,2,5)$	$f(x,2,2)$	7	487,6085714	10	0,004784717	0
$f(x,5,2)$	$f(x,5,2)$	$f(x,2,5)$	6	25,92666667	0	-	4
$f(x,5,2)$	$f(x,2,2)$	$f(x,0,5,0,5)$	6	29,88166667	10	0,006450196	0
$f(x,5,2)$	$f(x,2,2)$	$f(x,2,2)$	6	58,72833333	10	0,003413365	0
$f(x,5,2)$	$f(x,2,2)$	$f(x,5,2)$	6	61,48333333	10	0,003465536	0
$f(x,0,5,0,5)$	$f(x,5,2)$	$f(x,0,5,0,5)$	6	77,51333333	7	0,00297688	3
$f(x,0,5,0,5)$	$f(x,2,5)$	$f(x,0,5,0,5)$	6	395,365	10	0,00770437	0
$f(x,2,2)$	$f(x,5,2)$	$f(x,2,5)$	6	624,3183333	4	0,000866278	3
$f(x,0,5,0,5)$	$f(x,2,5)$	$f(x,2,5)$	6	774,97	10	0,007687028	0
$f(x,0,5,0,5)$	$f(x,5,2)$	$f(x,5,2)$	5	92,488	6	0,00054691	4
$f(x,0,5,0,5)$	$f(x,5,2)$	$f(x,2,2)$	5	142,152	7	0,002404997	3
$f(x,0,5,0,5)$	$f(x,2,5)$	$f(x,2,2)$	5	676,544	10	0,008698172	0
$f(x,0,5,0,5)$	$f(x,2,5)$	$f(x,5,2)$	4	65,02	10	0,004805311	0
$f(x,2,2)$	$f(x,5,2)$	$f(x,2,2)$	3	405,92	6	0,001198743	4
$f(x,2,2)$	$f(x,5,2)$	$f(x,0,5,0,5)$	3	621,0166667	7	0,0015551	3
$f(x,2,2)$	$f(x,5,2)$	$f(x,5,2)$	3	878,61	8	0,001292897	2
$f(x,5,2)$	$f(x,5,2)$	$f(x,2,2)$	2	1881,9	1	-	7
$f(x,5,2)$	$f(x,5,2)$	$f(x,0,5,0,5)$	2	3600	0	-	8
$f(x,5,2)$	$f(x,2,5)$	$f(x,5,2)$	0	-	10	0,462392304	0
$f(x,5,2)$	$f(x,2,5)$	$f(x,2,2)$	0	-	9	0,018403446	1
$f(x,5,2)$	$f(x,2,5)$	$f(x,0,5,0,5)$	0	-	6	0,022475927	4
$f(x,5,2)$	$f(x,2,5)$	$f(x,2,5)$	0	-	5	0,016305238	5
$f(x,5,2)$	$f(x,5,2)$	$f(x,5,2)$	1	3600	1	0,0060103	8

Table 31 – Instances tests analysis: 40 vessels and 5 berths

processing time (p)	arrival time (a)	departure time (b)	solved problems	average comp. time (s)	founded a solution	Average GAP	Unknown Status
$f(x,2,5)$	$f(x,0,5,0,5)$	$f(x,2,5)$	10	1,213	10	-	0
$f(x,2,5)$	$f(x,0,5,0,5)$	$f(x,0,5,0,5)$	10	1,5	10	-	0
$f(x,2,5)$	$f(x,0,5,0,5)$	$f(x,2,2)$	10	1,698	10	-	0
$f(x,2,5)$	$f(x,2,2)$	$f(x,2,5)$	10	1,751	10	-	0
$f(x,2,5)$	$f(x,2,2)$	$f(x,0,5,0,5)$	10	1,898	10	-	0
$f(x,2,5)$	$f(x,0,5,0,5)$	$f(x,5,2)$	10	1,932	10	-	0
$f(x,2,2)$	$f(x,0,5,0,5)$	$f(x,2,5)$	10	2,01	9	-	0
$f(x,2,5)$	$f(x,2,2)$	$f(x,2,2)$	10	2,153	10	-	0
$f(x,2,5)$	$f(x,5,2)$	$f(x,2,5)$	10	2,253	10	-	0
$f(x,2,2)$	$f(x,0,5,0,5)$	$f(x,2,2)$	10	2,306	9	-	0
$f(x,0,5,0,5)$	$f(x,0,5,0,5)$	$f(x,0,5,0,5)$	10	2,45	10	-	0
$f(x,0,5,0,5)$	$f(x,0,5,0,5)$	$f(x,2,2)$	10	2,515	10	-	0
$f(x,0,5,0,5)$	$f(x,0,5,0,5)$	$f(x,5,2)$	10	2,8	10	-	0
$f(x,2,5)$	$f(x,2,2)$	$f(x,5,2)$	10	2,801	10	-	0
$f(x,2,2)$	$f(x,0,5,0,5)$	$f(x,0,5,0,5)$	10	2,846	9	-	0
$f(x,2,2)$	$f(x,0,5,0,5)$	$f(x,5,2)$	10	3,481	9	-	0
$f(x,2,5)$	$f(x,5,2)$	$f(x,2,2)$	10	3,689	10	-	0
$f(x,2,5)$	$f(x,2,5)$	$f(x,0,5,0,5)$	10	3,713	10	-	0
$f(x,2,5)$	$f(x,2,5)$	$f(x,2,5)$	10	4,699	10	-	0
$f(x,2,5)$	$f(x,5,2)$	$f(x,0,5,0,5)$	10	5,259	10	-	0
$f(x,2,2)$	$f(x,2,2)$	$f(x,2,5)$	10	6,337	10	-	0
$f(x,2,5)$	$f(x,5,2)$	$f(x,5,2)$	10	6,956	10	-	0
$f(x,2,5)$	$f(x,2,5)$	$f(x,5,2)$	10	8,776	10	-	0
$f(x,2,2)$	$f(x,2,2)$	$f(x,0,5,0,5)$	10	10,142	10	-	0
$f(x,2,2)$	$f(x,2,2)$	$f(x,5,2)$	10	14,552	10	-	0
$f(x,5,2)$	$f(x,0,5,0,5)$	$f(x,0,5,0,5)$	10	68,464	8	-	0
$f(x,0,5,0,5)$	$f(x,0,5,0,5)$	$f(x,2,5)$	9	1,957777778	10	0,000278529	0
$f(x,0,5,0,5)$	$f(x,2,2)$	$f(x,2,5)$	9	4,032222222	10	0,00110876	0
$f(x,0,5,0,5)$	$f(x,2,2)$	$f(x,0,5,0,5)$	9	5,38	10	0,000641864	0
$f(x,0,5,0,5)$	$f(x,2,2)$	$f(x,2,2)$	9	5,473333333	10	0,00138627	0
$f(x,5,2)$	$f(x,0,5,0,5)$	$f(x,2,5)$	9	6,945555556	6	-	0
$f(x,0,5,0,5)$	$f(x,2,2)$	$f(x,5,2)$	9	8,494444444	10	0,00104295	0
$f(x,5,2)$	$f(x,0,5,0,5)$	$f(x,5,2)$	9	9,844444444	8	0,000646847	0
$f(x,2,5)$	$f(x,2,5)$	$f(x,2,2)$	9	10,12888889	10	0,00136447	0
$f(x,2,2)$	$f(x,2,2)$	$f(x,2,2)$	9	12,50111111	10	0,00117106	0
$f(x,5,2)$	$f(x,0,5,0,5)$	$f(x,2,2)$	8	9,8425	8	0,00033413	0
$f(x,0,5,0,5)$	$f(x,2,5)$	$f(x,0,5,0,5)$	7	45,29714286	10	0,00312722	0
$f(x,0,5,0,5)$	$f(x,2,5)$	$f(x,2,2)$	7	55,47857143	10	0,007159277	0
$f(x,2,2)$	$f(x,2,5)$	$f(x,2,2)$	7	62,74857143	10	0,002958975	0
$f(x,2,2)$	$f(x,2,5)$	$f(x,5,2)$	7	82,88428571	10	0,003218282	0
$f(x,0,5,0,5)$	$f(x,2,5)$	$f(x,5,2)$	7	91,13571429	10	0,003022713	0
$f(x,2,2)$	$f(x,2,5)$	$f(x,2,5)$	7	174,5042857	10	0,003406684	0
$f(x,5,2)$	$f(x,2,2)$	$f(x,0,5,0,5)$	7	218,9385714	10	0,001216485	0
$f(x,5,2)$	$f(x,2,2)$	$f(x,2,2)$	7	411,9828571	10	0,000910956	0
$f(x,5,2)$	$f(x,2,2)$	$f(x,2,5)$	7	465,5771429	10	0,00163916	0
$f(x,2,2)$	$f(x,2,5)$	$f(x,0,5,0,5)$	6	36,71333333	10	0,002330429	0
$f(x,0,5,0,5)$	$f(x,2,5)$	$f(x,2,5)$	6	69,195	10	0,003862196	0
$f(x,2,2)$	$f(x,5,2)$	$f(x,2,5)$	6	281,0366667	7	0,000646937	1
$f(x,5,2)$	$f(x,2,2)$	$f(x,5,2)$	6	368,925	10	0,001100631	0
$f(x,0,5,0,5)$	$f(x,5,2)$	$f(x,2,5)$	5	141,712	8	0,00126865	2
$f(x,0,5,0,5)$	$f(x,5,2)$	$f(x,0,5,0,5)$	5	168,912	9	0,000738735	1
$f(x,2,2)$	$f(x,5,2)$	$f(x,0,5,0,5)$	5	253,836	9	0,001051094	1
$f(x,2,2)$	$f(x,5,2)$	$f(x,2,2)$	5	438,952	9	0,000868284	1
$f(x,5,2)$	$f(x,5,2)$	$f(x,2,5)$	5	576,04	3	0,001430514	3
$f(x,2,2)$	$f(x,5,2)$	$f(x,5,2)$	4	281,0925	9	0,000744472	1
$f(x,0,5,0,5)$	$f(x,5,2)$	$f(x,5,2)$	4	526,655	10	0,000846194	0
$f(x,0,5,0,5)$	$f(x,5,2)$	$f(x,2,2)$	4	594,8775	8	0,000980061	2
$f(x,5,2)$	$f(x,5,2)$	$f(x,0,5,0,5)$	1	2201,84	2	0,00224825	8
$f(x,5,2)$	$f(x,5,2)$	$f(x,5,2)$	1	3600	2	0,001255978	7
$f(x,5,2)$	$f(x,2,5)$	$f(x,2,2)$	0	-	10	0,013184361	0
$f(x,5,2)$	$f(x,2,5)$	$f(x,5,2)$	0	-	10	0,014393717	0
$f(x,5,2)$	$f(x,2,5)$	$f(x,0,5,0,5)$	0	-	9	61,45954436	1
$f(x,5,2)$	$f(x,2,5)$	$f(x,2,5)$	0	-	6	0,009421027	4
$f(x,5,2)$	$f(x,5,2)$	$f(x,2,2)$	0	-	2	0,000109051	8

Table 32 – Instances tests analysis: 40 vessels and 7 berths

processing time (p)	arrival time (a)	departure time (b)	solved problems	average comp. time (s)	founded a solution	Average GAP	Unknown Status
$f(x,2,5)$	$f(x,0,5,0,5)$	$f(x,2,5)$	10	1,699	10	-	0
$f(x,2,5)$	$f(x,0,5,0,5)$	$f(x,0,5,0,5)$	10	2,106	10	-	0
$f(x,2,2)$	$f(x,0,5,0,5)$	$f(x,2,5)$	10	2,139	10	-	0
$f(x,2,5)$	$f(x,0,5,0,5)$	$f(x,2,2)$	10	2,22	10	-	0
$f(x,2,2)$	$f(x,0,5,0,5)$	$f(x,0,5,0,5)$	10	2,383	10	-	0
$f(x,2,5)$	$f(x,0,5,0,5)$	$f(x,5,2)$	10	2,519	10	-	0
$f(x,2,2)$	$f(x,0,5,0,5)$	$f(x,2,2)$	10	2,678	10	-	0
$f(x,2,2)$	$f(x,0,5,0,5)$	$f(x,5,2)$	10	2,878	10	-	0
$f(x,2,5)$	$f(x,2,2)$	$f(x,2,5)$	10	3,04	10	-	0
$f(x,2,5)$	$f(x,5,2)$	$f(x,2,5)$	10	3,112	10	-	0
$f(x,2,5)$	$f(x,2,2)$	$f(x,0,5,0,5)$	10	3,126	10	-	0
$f(x,2,5)$	$f(x,2,2)$	$f(x,2,2)$	10	3,829	10	-	0
$f(x,2,5)$	$f(x,5,2)$	$f(x,0,5,0,5)$	10	4,289	10	-	0
$f(x,2,5)$	$f(x,2,5)$	$f(x,2,5)$	10	4,583	10	-	0
$f(x,2,5)$	$f(x,5,2)$	$f(x,2,2)$	10	4,583	10	-	0
$f(x,0,5,0,5)$	$f(x,0,5,0,5)$	$f(x,2,5)$	10	4,672	10	-	0
$f(x,0,5,0,5)$	$f(x,0,5,0,5)$	$f(x,0,5,0,5)$	10	4,689	10	-	0
$f(x,2,5)$	$f(x,2,2)$	$f(x,5,2)$	10	4,802	10	-	0
$f(x,2,5)$	$f(x,2,5)$	$f(x,0,5,0,5)$	10	5,122	10	-	0
$f(x,2,5)$	$f(x,2,5)$	$f(x,2,2)$	10	5,802	10	-	0
$f(x,5,2)$	$f(x,0,5,0,5)$	$f(x,0,5,0,5)$	10	5,81	9	-	0
$f(x,0,5,0,5)$	$f(x,0,5,0,5)$	$f(x,2,2)$	10	5,987	10	-	0
$f(x,2,5)$	$f(x,2,5)$	$f(x,5,2)$	10	6,154	10	-	0
$f(x,5,2)$	$f(x,0,5,0,5)$	$f(x,2,2)$	10	6,494	9	-	0
$f(x,2,2)$	$f(x,2,2)$	$f(x,2,5)$	10	6,996	10	-	0
$f(x,0,5,0,5)$	$f(x,2,2)$	$f(x,2,5)$	10	7,215	10	-	0
$f(x,0,5,0,5)$	$f(x,0,5,0,5)$	$f(x,5,2)$	10	7,585	10	-	0
$f(x,2,5)$	$f(x,5,2)$	$f(x,5,2)$	10	8,101	10	-	0
$f(x,5,2)$	$f(x,0,5,0,5)$	$f(x,5,2)$	10	8,788	10	-	0
$f(x,0,5,0,5)$	$f(x,2,2)$	$f(x,0,5,0,5)$	10	9,032	10	-	0
$f(x,0,5,0,5)$	$f(x,2,2)$	$f(x,2,2)$	10	10,192	10	-	0
$f(x,2,2)$	$f(x,2,2)$	$f(x,0,5,0,5)$	10	11,693	10	-	0
$f(x,2,2)$	$f(x,2,2)$	$f(x,2,2)$	10	12,62	10	-	0
$f(x,0,5,0,5)$	$f(x,2,2)$	$f(x,5,2)$	10	14,216	10	-	0
$f(x,2,2)$	$f(x,2,2)$	$f(x,5,2)$	10	14,68	10	-	0
$f(x,5,2)$	$f(x,0,5,0,5)$	$f(x,2,5)$	9	5,104444444	9	-	0
$f(x,5,2)$	$f(x,5,2)$	$f(x,2,5)$	8	10,72125	1	0,000259186	1
$f(x,2,2)$	$f(x,5,2)$	$f(x,2,5)$	8	45,075	9	0,000431308	1
$f(x,2,2)$	$f(x,5,2)$	$f(x,0,5,0,5)$	8	74,82	9	0,000400638	1
$f(x,2,2)$	$f(x,5,2)$	$f(x,2,2)$	8	89,75375	9	0,000511237	1
$f(x,2,2)$	$f(x,5,2)$	$f(x,5,2)$	8	119,49125	10	0,000744089	0
$f(x,0,5,0,5)$	$f(x,2,5)$	$f(x,2,2)$	7	177,5271429	10	0,001816381	0
$f(x,0,5,0,5)$	$f(x,5,2)$	$f(x,2,2)$	7	193,92	9	0,000647949	1
$f(x,0,5,0,5)$	$f(x,2,5)$	$f(x,0,5,0,5)$	7	237,5314286	10	0,001427221	0
$f(x,0,5,0,5)$	$f(x,2,5)$	$f(x,5,2)$	7	242,2242857	10	0,002035295	0
$f(x,2,2)$	$f(x,2,5)$	$f(x,2,2)$	7	298,1028571	10	0,001550801	0
$f(x,0,5,0,5)$	$f(x,5,2)$	$f(x,5,2)$	7	311,1714286	9	0,000721911	1
$f(x,2,2)$	$f(x,2,5)$	$f(x,5,2)$	7	332,9942857	10	0,001634901	0
$f(x,2,2)$	$f(x,2,5)$	$f(x,2,5)$	6	13,82	10	0,001221406	0
$f(x,2,2)$	$f(x,2,5)$	$f(x,0,5,0,5)$	6	14,87666667	10	0,001486145	0
$f(x,5,2)$	$f(x,2,2)$	$f(x,0,5,0,5)$	6	36,68666667	10	0,002862428	0
$f(x,0,5,0,5)$	$f(x,5,2)$	$f(x,0,5,0,5)$	6	126,7433333	8	0,000258771	2
$f(x,0,5,0,5)$	$f(x,2,5)$	$f(x,2,5)$	6	144,2633333	9	0,001542333	1
$f(x,5,2)$	$f(x,2,2)$	$f(x,2,5)$	5	17,84	9	0,002798768	1
$f(x,0,5,0,5)$	$f(x,5,2)$	$f(x,2,5)$	5	38,142	8	0,000473617	2
$f(x,5,2)$	$f(x,2,2)$	$f(x,2,2)$	5	58,798	10	0,002071587	0
$f(x,5,2)$	$f(x,2,2)$	$f(x,5,2)$	5	72,802	10	0,002214094	0
$f(x,5,2)$	$f(x,5,2)$	$f(x,0,5,0,5)$	3	2426,73	1	-	7
$f(x,5,2)$	$f(x,5,2)$	$f(x,2,2)$	2	1801,29	1	0,000618614	7
$f(x,5,2)$	$f(x,2,5)$	$f(x,5,2)$	0	-	10	0,006743566	0
$f(x,5,2)$	$f(x,2,5)$	$f(x,2,2)$	0	-	10	0,006909036	0
$f(x,5,2)$	$f(x,2,5)$	$f(x,0,5,0,5)$	0	-	10	0,008360092	0
$f(x,5,2)$	$f(x,2,5)$	$f(x,2,5)$	0	-	9	0,005560449	1
$f(x,5,2)$	$f(x,5,2)$	$f(x,5,2)$	0	-	1	0,000304893	9

Table 33 – Instances tests analysis: 40 vessels and 10 berths

Based on the CPLEX results, the difficulty of the parameters combinations for the beta distribution was classified in Tables 25, 26, 27, 28, 29, 30, 31, 32, 33 as follows. First, we count how many of these 10 instances were completely solved, either by proving the optimality of the solution or by proving that the problem is infeasible. Second, we organized then in non-decreasing order of the average computational time taken by CPLEX. Third, we count how many of the 10 instances CPLEX was able to find a feasible solution. Fourth, for the instances in which a feasible solution was found, but the optimality was not reached, we organized then in a non-decreasing order of average gap. Finally, we count in how many of the 10 instances CPLEX was not able to find a feasible solution within one hour or prove the problem as infeasible.

Analyzing the results, it is easy to see that the HVRPTW model performs better as the *vessels/berths* ratio decreases. According to (BUHRKAL *et al.*, 2011), this happens due to the type of valid inequalities (2.38) introduced. However, the number of variables must be also considered when measuring the difficulty. For example, the instances with 20 vessels and 5 berths and the instances with 40 vessels and 10 berths have the same ratio, 4, but the first one has 20∗20∗5 = 2000 binary variables, while the last one has 40∗40∗10 = 16000 binary variables. Besides, the instances in which the arrival times of all the vessels are concentrated at the beginning of the planning period have shown to be hard for CPLEX. According to (FROJAN *et al.*, 2015), it produces a congestion which appears to be very difficult to manage. For these cases, we will develop two metaheuristics to try to obtain a good feasible solutions for the problem in a short computational time.

7.2 Conclusion

Different models have been proposed for the BAP and it was not found a data benchmark to solve them making comparisons between researches difficult. Most papers in literature use in their experiments randomly generated data. To overcome such drawback, this Chapter proposed a problem generator for the BAP, allowing the generation of appropriate test problems to be commonly used with specific desired properties and under controlled conditions. The data were generated using different parameters and the difficulty of solving the BAP with such data was analyzed through the resolution using the CPLEX. The tables showed that for some data, the CPLEX was able to find the optimal solutions in a few seconds.

8 Evaluating the benchmark data

As stated earlier, the BAP is a NP-hard problem. Therefore, we have developed a Genetic Algorithm in Section 8.1, a classical method that has been one of the first metaheuristic proposed in the literature, easily adaptable to any type of problem in order to evaluated the benchmark data proposed in Chapter 7. The performance of the proposed GA was compared with a recent Particle Swarm Algorithm (PSO) developed by (TING *et al.*, 2014a) and presented in Section 8.2. Both algorithms were implemented with a similar structure (codification, initialization, fitness) to allow comparisons of performance. The results are shown in Section 8.3.

8.1 Genetic Algorithm

Genetic Algorithms are a metaheuristic based in the evolutionary idea of natural selection and genetics. The pioneering work of J. H. Holland in the 1970s, with the publication of his book, *Adaptation in Natural and Artificial Systems*, consolidated the contribution of Genetic Algorithms to operations research ((GOLDBERG, 1989)).

The algorithm can be summarized as follows. The implementation begins by codifying the solution to generate a random population of individuals. (BÄCK *et al.*, 2000b) highlights that the adopted codification can cause individuals to be infeasible. Then these structures are evaluated and assigned a fitness value. It allows to assign to each individual from the search space a value that is used as a measure of performance. In optimization problems, the fitness value incorporates all the aspects present in the objective function. For infeasible individuals, in addition to the objective function, the fitness value also incorporates this information through a penalty cost. Selection is then applied to the current population to create an intermediate population. The crossover and mutation are applied to the intermediate population to create the next population. The crossover generates new individuals through the recombination of characteristics of two or more individuals (inheritance). It is considered the predominant genetic operator, so it is applied with greater probability than the mutation. The mutation create a new individual from a single parent, maintaining the genetic diversity of the population. For such reason, crossover and mutation are considered complementary. The process of going from the current population to the next population constitutes one iteration.

As in (TING *et al.*, 2014a) and (KURZ; ASKIN, 2004), the following solution representation with floating point will be used in this work. To each vessel is assigned a real number between $(0, m)$. The integer part is the berth number to which the vessel is allocated and the fractional part is used to sort the vessels allocated to each berth.

Let $q_k = [q_k(1);\ q_k(2);\ ...\ ;\ q_k(n)]$ be an individual k. For example, for the berth-vessel allocation shown in Figure 1, the representation is shown in Figure 25

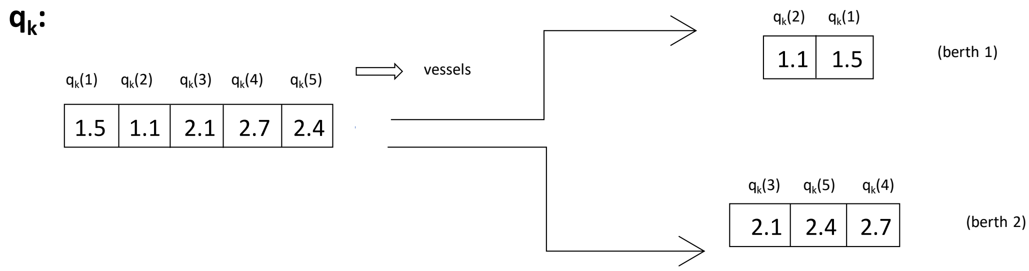


Figure 25 – Solution representation for the allocation in Figure 1

All individuals were randomly generated except one that was based on the *first-come-first-served* heuristic (FCFS). In this case, the vessels were sorted by their arrival times in ascending order and assigned one at a time by choosing the combination of berth and vessel that will finish first, until all vessels have been assigned.

The selection was done through the binary tournament (BÄCK *et al.*, 2000a) in order to built a population of parents. Pairs of parents from this population are selected to generate pairs of offspring through crossover and mutation.

In the crossover, for each pair of parents, a random number rm in the range $[0.3, 0.7]$ is generated. $rm\%$ from the first parent is copied to the new individual and $(1 - rm)\%$ from the second parent is copied to the new individual, giving rise to the first offspring. The second offspring is formed by the complementary portions of both parents.

(MICHALEWICZ, 1996) developed the Non-Uniform Mutation, especially for optimization problems with constraints and floating point coding. It is a dynamic operator designed to improve the search process. The new element resulting from the mutation is, with 50 % probability each:

$$q'_k(i) = \begin{cases} q_k(i) + \Delta(g, ub - q_k(i)) & \text{or} \\ q_k(i) - \Delta(g, q_k(i) - lb) \end{cases} \tag{8.1}$$

where $[lb, ub]$ is $[0, m]$ for the BAP. Function $\Delta(g, y)$ returns a value in the range $[0, y]$ such that the probability of $\Delta(g, y)$ being close to zero increases as g increases $(g = 1, ..., G)$. This property causes this operator to initially explore the search space extensively in the initial iterations, and locally in advanced iterations. (MICHALEWICZ, 1996) proposes the following function:

$$\Delta(g, y) = y.(1 - r^{(\frac{1-g}{G})^b}). \tag{8.2}$$

where r is a random number from [0,1], G is the maximal iteration number and b is a system parameter determining the degree of dependency on iteration number (empirically, we set $b = 3$).

We also incorporate to the GA algorithm local search to improve the solution quality as proposed in (TING *et al.*, 2014a). This technique has two procedures. The vessels can be swapped in the same berth, by comparing all the possible swapping pairs within the same berth and select the best improvement to exchange their fitness values; and between berths, by selecting randomly two vessels in two different berths and selecting the best improvement to exchange.

8.2 Particle Swarm Algorithm

According to (EBERHART; SHI, 2001), the Particle Swarm Optimization - PSO - simulates the movement of organisms in a bird flock. It has been used across a wide range of applications because there are few parameters to adjust. For an optimization problem of n variables, (DU; SWAMY, 2016) defined a swarm of N_P particles. Each particle has its own trajectory, position x_i and velocity v_i, and moves in the search space by successively updating its trajectory. All particles have fitness values that are evaluated by the fitness function to be optimized. The particles are flown through the solution space by following the current optimum particles. The algorithm initializes a group of particles with random positions and then searches for optima by updating iterations. In every iteration, each particle is updated by the particle best *pbest*, denoted x_i^* , $i = 1, ..., N_P$, which is the best solution it has achieved so far. The global best *gbest*, denoted x^g, is also updated, which is the best value obtained so far by any particle in the population.

Because all particles in the swarm learn from *gbest* even if *gbest* is far from the global optimum, particles may easily be attracted to the *gbest* region and get trapped in a local optimum for multimodal problems. In case the *gbest* positions locate on local minimum, other particles in the swarm may also be trapped. If an early solution is suboptimal, the swarm can easily stagnate around it without any pressure to continue exploration.

PSO can locate the region of the optimum faster than other. However, once in this region it progresses slowly due to the fixed velocity stepsize. Linearly decreasing weight PSO effectively balances the global and local search abilities of the swarm by introducing a linearly decreasing inertia weight on the previous velocity of the particle into

$$v_i(t+1) = \alpha v_i(t) + c_1 r_1 [x_i^*(t) - x_i(t)] + c_2 r_2 [x^g(t) - x_i(t)] \tag{8.3}$$

where α is called the inertia weight, and the positive constants c_1 and c_2 are, respectively, cognitive and social parameters decreases from α_{max} to α_{min}.

At iteration $t+1$, the swarm can be updated by

$$x_i(t+1) = x_i(t) + v_i(t+1) \tag{8.4}$$

(EBERHART; SHI, 1998) point out that PSO does not label its operations in the same way as GAs, but analogies exist depending on the implementation of the GA operation. For example, the effect of selection in a GA is to support the survival of the fittest, a concept central to all evolutionary algorithms. PSO does not utilize selection once all particles continue as members of the population for the duration of the run. A particle does not explicitly exchange material with other particles, but its trajectory is influenced by them. The concept of crossover is represented in PSO because each particle is stochastically accelerated toward its own previous best position, as well as toward the global best position or the local best position. It is also apparent in the behavior of particles that appear approximately midway between swarms of particles that are clustering around local best positions, or, occasionally, between successive global best positions. These particles seem to be exploring a region that represents the geometric mean between two promising regions. Mutation allows a GA chromosome to reach any point in the problem space particularly near the end of a run because a number of mutations may be needed to reach a distant point. It may be that a PSO particle cannot reach any point in problem space in one iteration, although this might be possible at the beginning of the run.

8.3 Computational tests

Based on the results from CPLEX, the following instances were chosen to solve with the proposed metaheuristics:

vessels	berths	p	a	b
20	5	$f(x,5,2)$	$f(x,2,5)$	$f(x,2,5)$
30	5	$(x,5,2)$	$f(x,2,5)$	$f(x,2,2)$
30	5	$f(x,5,2)$	$f(x,2,5)$	$f(x,0,5,0,5)$
30	5	$f(x,5,2)$	$f(x,2,5)$	$f(x,5,2)$
30	7	$f(x,5,2)$	$f(x,2,5)$	$f(x,2,5)$
30	7	$f(x,5,2)$	$f(x,2,5)$	$f(x,2,2)$
30	7	$f(x,5,2)$	$f(x,2,5)$	$f(x,5,2)$
40	5	$f(x,5,2)$	$f(x,2,5)$	$f(x,5,2)$
40	7	$f(x,5,2)$	$f(x,2,5)$	$f(x,2,5)$
40	7	$f(x,5,2)$	$f(x,2,5)$	$f(x,0,5,0,5)$
40	7	$f(x,5,2)$	$f(x,2,5)$	$f(x,5,2)$

Table 34 – Instances chosen to test the metaheuristics

The computational tests were executed on a personal computer, a Dell Inspiron 14Z with Intel Core I5-3337U 1.80GHz, RAM memory of 6GB and a Solid State Drive of size 240 GB. The GA and the PSO were implemented in C language.

For both algorithms, the population size was set to 200 and the number of iterations to 500. For the GA, the probability of crossover was 0.9, and individuals who did not pass through the crossover are mutated with a probability of 0.1. For the PSO, the parameter-setting used was as in (TING *et al.*, 2014a): $W = 0.9$, $c1 = c2 = 2$. Due to the stochastic nature of metaheuristics, each instance is run for 30 times and reported the average computational time, the average objective function, the best solution and iteration that the best solution was obtained. Total, the computational tests were performed with 110 instances. The results are shown in Tables 37 and 38.

8.3.1 Comparing the GA with two different encodings

First, the GA developed using the real coding in Section 8.1 was compared with the one developed in Chapter 6. Tables 35 and 36 show the results.

	GA Chapter 6			GA Section 8.1				CPLEX		
	Comp. time	Average objective	Best solution	Comp. time	Average objective	Best solution	Iteration number	Objective Function	Comp. time	FCFS heuristic
20x5 - $p: f(x,5,2)$, $a: f(x,2,5)$, $b: f(x,2,5)$										
1	4,17904	440,92	429	2,44096	442,5	428	80	428	703,56	InFea
2	4,35339	378,933	373	2,44048	376,867	373	27	373	3600	InFea
3	4,45284	359,667	354	2,47887	356,7	354	11	354	77,39	357
4	4,30584	439,733	429	2,43888	434,7	425	72	425	3600	InFea
5	4,52712	328,833	321	2,4408	324,467	321	45	321	37,83	InFea
6	4,56483	425,767	421	2,432	426,033	418	52	419	3600	InFea
7	5,33735	363,333	361	2,44154	365,867	361	46	361	3600	InFea
8	1,95751	186,933	370	2,37677	373,067	371	59	370	2943,91	InFea
9	3,93397	391,633	385	2,39217	394,933	388	40	385	3600	InFea
10	3,70124	411,833	406	2,40779	422,867	416	63	410	3600	InFea
30x5 - $p: f(x,5,2)$, $a: f(x,2,5)$, $b: f(x,2,2)$										
1	8,8354	405,3	390	5,44496	392,433	387	65	390	3600	427
2	9,65109	373,4	360	5,62155	363,867	357	72	354	3600	Infea
3	9,86312	358,533	339	5,36694	341,6	339	76	331	3600	InFea
4	9,62346	405,267	386	5,5594	392,567	385	108	388	3600	442
5	8,43647	437,6	418	5,51518	429,7	418	63	424	3600	InFea
6	8,8245	470,833	454	5,57889	464,8	451	81	459	3600	InFea
7	9,76903	406,9	395	5,47882	397,133	388	108	394	3600	494
8	4,39767	221,567	425	5,46717	423,1	414	150	416	3600	InFea
9	9,09156	485,467	472	5,46193	471,8	460	163	481	3600	InFea
10	9,20963	433,933	416	5,51181	415,633	408	164	415	3600	InFea
30x5 - $p: f(x,5,2)$, $a: f(x,2,5)$, $b: f(x,0,5,0,5)$										
1	8,82407	424,567	402	5,55014	409,667	395	254	405	3600	InFea
2	9,72591	373,1	359	5,67975	368,733	357	82	361	3600	InFea
3	10,4145	355,567	342	5,35032	343,033	338	158	331	3600	InFea
4	9,80861	405,833	387	5,61758	393,733	385	61	388	3600	InFea
5	8,80933	459,267	433	5,61267	443,833	421	113	445	3600	InFea
6	8,98778	483,967	466	5,55913	474,933	458	154	483	3600	InFea
7	10,0954	408,067	397	5,49641	399,133	388	98	400	3600	InFea
8	4,43618	233,233	442	5,51259	446,7	428	135	429	3600	InFea
9	9,45453	498,9	473	6,0647	492,267	472	330	504	3600	InFea
10	9,59904	439,967	418	6,3454	429,867	414	131	427	3600	InFea
30x5 - $p: f(x,5,2)$, $a: f(x,2,5)$, $b: f(x,5,2)$										
1	9,96859	403,833	392	5,36892	390,767	382	43	391	3600	427
2	10,3374	370,167	359	5,58279	361,2	355	128	356	3600	422
3	10,4315	355,967	338	5,29334	341	339	68	331	3600	365
4	10,402	403,267	391	5,49413	392,1	386	163	393	3600	442
5	9,98498	435,467	420	5,44219	426,433	414	139	422	3600	490
6	9,97067	470,7	457	5,48639	462,133	455	257	460	3600	562
7	10,3258	403,9	393	5,43422	394,267	386	120	388	3600	494
8	5,08116	221,6	423	5,42002	427,533	414	65	414	3600	InFea
9	10,2756	480,4	469	5,38488	468,5	461	216	476	3600	539
10	10,3717	427,767	413	5,48115	414,467	406	98	414	3600	497
30x7 - $p: f(x,5,2)$, $a: f(x,2,5)$, $b: f(x,2,5)$										
1	12,1069	604,133	587	4,42206	589,8	584	69	594	3600	InFea
2	10,3655	606,6	585	4,33721	583,633	563	143	-	3600	InFea
3	8,44059	560,933	552	4,31419	560,567	546	139	554	3600	InFea
4	8,34316	637,6	602	4,29948	601,2	586	80	594	3600	InFea
5	8,03397	654,233	624	4,28449	643,867	633	74	632	3600	InFea
6	8,51378	545,6	521	4,41932	534,667	524	68	519	3600	InFea
7	8,18668	592,267	569	4,30246	600,533	567	150	567	3600	InFea
8	4,18853	282,25	540	4,36342	544,967	533	98	538	3600	InFea
9	8,08627	644,767	605	4,35974	618,9	598	249	615	3600	InFea
10	7,95617	716,577	680	4,3448	719,769	689	117	-	3600	InFea
30x7 - $p: f(x,5,2)$, $a: f(x,2,5)$, $b: f(x,2,2)$										
1	8,88333	601,267	586	4,27634	586,033	583	68	580	3600	635
2	8,52508	589,733	573	4,25298	581,833	563	173	579	3600	InFea
3	9,3706	559,7	545	4,21417	547,1	541	84	546	3600	InFea
4	8,51503	611,6	586	4,24585	587,767	577	206	583	3600	InFea
5	8,92464	639,567	627	4,23992	630,5	615	70	630	3600	InFea
6	9,45839	536,9	523	4,31371	523,5	517	119	512	3600	570
7	8,84585	582,633	566	4,20656	572,433	560	197	552	3600	InFea
8	4,67835	277,783	537	4,29258	534,7	526	93	537	3600	InFea
9	7,79943	619,6	600	4,25596	601	594	78	606	3600	InFea
10	9,83773	685,033	667	4,26808	669,667	657	188	678	3600	InFea

Table 35 – GA coding comparison - part I

	GA Chapter 6			GA Section 8.1				CPLEX		
	Comp. time	Average objective	Best solution	Comp. time	Average objective	Best solution	Iteration number	Objective Function	Comp. time	FCFS heuristic
30x7 - $p: f(x,5,2)$, $a: f(x,2,5)$, $b: f(x,5,2)$										
1	9,13121	593,267	581	4,16898	582,467	580	97	579	3600	635
2	9,34076	591,6	577	4,18	581,567	563	182	564	3600	637
3	9,98579	559,367	545	4,1415	545,8	541	93	550	3600	608
4	8,97596	607	586	4,15392	577,9	575	207	575	3600	654
5	9,00258	637,9	622	4,19005	629,533	623	69	615	3600	696
6	9,49101	530,6	518	4,5087	520,833	509	236	509	3600	570
7	9,40395	579,167	554	4,61945	565,933	555	108	560	3600	636
8	4,87362	276,65	534	4,57433	533,3	527	103	536	3600	618
9	9,03852	615,633	602	4,71918	592,2	589	90	598	3600	650
10	9,32982	674,167	659	4,61019	656,9	653	122	656	3600	732
40x5 - $p: f(x,5,2)$, $a: f(x,2,5)$, $b: f(x,5,2)$										
1	18,7147	683,267	665	11,0973	664	652	299	678	3600	842
2	20,7616	467,667	446	10,8854	453,733	448	201	450	3600	542
3	20,0778	399,633	378	10,6708	385,433	369	138	369	3600	489
4	18,392	519,133	498	10,5934	501,1	493	336	505	3600	625
5	19,2863	500,467	481	10,8279	487,8	480	95	482	3600	577
6	19,0954	475,267	448	10,6114	451,733	438	128	428	3600	576
7	19,2899	490,433	463	11,2115	463,867	455	95	472	3600	525
8	9,83619	184,617	342	10,864	347,033	331	156	328	3600	407
9	19,0078	444,933	414	11,5177	420,833	413	282	413	3600	497
10	18,7537	580,5	565	10,8037	566,5	550	212	568	3600	698
40x7 - $p: f(x,5,2)$, $a: f(x,2,5)$, $b: f(x,2,5)$										
_227 1	11,1394	686,133	655	7,52868	652,933	632	229	-	3600	InFea
2	11,9168	523,7	509	7,53006	491,367	466	193	463	3600	559
3	10,976	732,433	707	7,61987	718,464	688	316	-	3600	InFea
4	11,74	595,233	561	7,60239	554,567	546	103	567	3600	InFea
5	11,8577	609,367	583	7,44254	577	564	420	560	3600	InFea
6	11,2875	650,333	617	7,546	586,7	576	143	571	3600	InFea
7	11,8532	567,967	545	7,54718	519,6	508	136	512	3600	InFea
8	5,62577	350,15	668	7,64337	680,267	660	166	-	3600	InFea
9	11,8267	513,9	489	7,41872	479,6	472	157	474	3600	InFea
10	11,5322	669,233	624	7,60144	622,667	604	139	-	3600	InFea
40x7 - $p: f(x,5,2)$, $a: f(x,2,5)$, $b: f(x,0,5,0,5)$										
1	10,3451	673,733	639	7,56658	642,433	615	169	651	3600	InFea
2	11,3547	529,8	501	7,59673	493,6	476	291	469	3600	InFea
3	10,7667	737	707	7,59802	705,767	675	398	-	3600	InFea
4	11,866	605,767	569	7,65005	565,8	552	189	563	3600	InFea
5	12,419	609,433	586	7,474	572,533	559	274	553	3600	InFea
6	10,804	648,567	607	7,55181	593,333	574	223	644	3600	InFea
7	11,9391	571,733	538	7,5683	523,333	508	144	514	3600	InFea
8	5,60101	354,767	659	7,5857	666,3	637	313	671	3600	InFea
9	11,8415	528,233	503	7,49133	490,5	476	147	474	3600	InFea
10	11,4772	690,333	655	7,6489	642,833	624	232	653	3600	InFea
40x7 - $p: f(x,5,2)$, $a: f(x,2,5)$, $b: f(x,5,2)$										
1	14,3951	651,933	618	7,40226	635,667	627	92	614	3600	729
2	13,5733	515,533	483	7,42561	492,933	474	151	474	3600	559
3	12,8502	687,667	671	7,40462	658,733	646	140	663	3600	821
4	13,1228	579,633	560	7,57102	541,3	539	193	545	3600	590
5	12,9593	597,6	570	7,36938	565,3	555	215	558	3600	636
6	12,5648	593,833	570	7,33602	557,333	544	132	550	3600	646
7	12,8794	554,033	512	7,29626	518,733	510	122	512	3600	589
8	6,74939	332,833	634	7,46437	647,567	632	186	639	3600	747
9	12,8804	511,167	495	7,32683	475,367	468	192	476	3600	534
10	13,2078	640,633	624	7,51488	606,167	598	174	608	3600	737

Table 36 – GA coding comparison - part II

Comparing the computational time and the average objective, it is easily noticeable that the GA with real coding is the one that performs better.

8.3.2 Comparing the GA end the PSO

Among the three main methodologies proposed in this Chapter (CPLEX, GA and PSO with real coding), the best solutions obtained are highlighted in bold in the Tables 37 and 38.

	GA				PSO				CPLEX		
	Comp. time	Average obj.	Best solution	Iteration	Comp. time	Average obj.	Best sol.	Iteration	Objective	Comp. time	FCFS heuristic
20 vessels, 5 berths - $p : f(x, 5, 2)$, $a : f(x, 2, 5)$, $b : f(x, 2, 5)$											
1	2.44096	442.5	**428**	80	2.43371	455.45	429	390	**428**	703.56	InFea
2	2.44048	376.867	**373**	27	2.43811	378.2	**373**	312	**373**	3600	InFea
3	2.47887	356.7	**354**	11	2.44245	357	357	0	**354**	77.39	357
4	2.43888	434.7	**425**	72	2.43617	437.333	428	494	**425**	3600	InFea
5	2.4408	324.467	**321**	45	2.49607	338.5	329	300	**321**	37.83	InFea
6	2.432	426.033	**418**	52	2.41964	426.367	423	448	419	3600	InFea
7	2.44154	365.867	**361**	46	2.4342	365.3	364	223	**361**	3600	InFea
8	2.37677	373.067	371	59	2.42829	371	371	206	**370**	2943.91	InFea
9	2.39217	394.933	388	40	2.38351	394.133	389	462	**385**	3600	InFea
10	2.40779	422.867	416	63	2.43171	418.667	411	491	**410**	3600	InFea
30 vessels, 5 berths - $p : f(x, 5, 2)$, $a : 7f(x, 2, 5)$, $b : f(x, 2, 2)$											
1	5.44496	392.433	**387**	65	5.88105	393.167	**387**	316	390	3600	427
2	5.62155	363.867	357	72	5.91886	367.167	363	341	**354**	3600	Infea
3	5.36694	341.6	339	76	5.89994	342.667	341	172	**331**	3600	InFea
4	5.5594	392.567	385	108	5.93147	392.567	**382**	291	388	3600	442
5	5.51518	429.7	**418**	63	5.92249	426.733	421	303	424	3600	InFea
6	5.57889	464.8	**451**	81	6.00652	476.3	458	350	459	3600	InFea
7	5.47882	397.133	**388**	108	5.83806	398.533	**388**	494	394	3600	494
8	5.46717	423.1	**414**	150	5.90615	418.467	**414**	386	416	3600	InFea
9	5.46193	471.8	**460**	163	5.90029	471.7	469	454	481	3600	InFea
10	5.51181	415.633	**408**	164	5.89064	416	**408**	403	415	3600	InFea
30 vessels, 5 berths - $p : f(x, 5, 2)$, $a : f(x, 2, 5)$, $b : f(x, 0.5, 0.5)$											
1	5.55014	409.667	**395**	254	6.06957	410.167	399	434	405	3600	InFea
2	5.67975	368.733	**357**	82	6.07927	368.133	358	127	361	3600	InFea
3	5.35032	343.033	338	158	5.93905	342.533	341	453	**331**	3600	InFea
4	5.61758	393.733	**385**	61	6.05866	396.867	392	234	388	3600	InFea
5	5.61267	443.833	**421**	113	6.04785	438.4	425	449	445	3600	InFea
6	5.55913	474.933	**458**	154	6.06745	487.333	468	270	483	3600	InFea
7	5.49641	399.133	**388**	98	5.90988	403.9	394	405	400	3600	InFea
8	5.51259	446.7	**428**	135	6.07503	439.667	430	184	429	3600	InFea
9	6.0647	492.267	**472**	330	6.08939	494.1	475	431	504	3600	InFea
10	6.3454	429.867	**414**	131	6.03559	432.4	424	368	427	3600	InFea
30 vessels, 5 berths - $p : f(x, 5, 2)$, $a : f(x, 2, 5)$, $b : f(x, 5, 2)$											
1	5.36892	390.767	**382**	43	5.82077	389.833	385	404	391	3600	427
2	5.58279	361.2	**355**	128	5.83069	365.133	363	439	356	3600	422
3	5.29334	341	339	68	5.78876	343	343	94	**331**	3600	365
4	5.49413	392.1	386	163	5.84025	391.867	**382**	418	393	3600	442
5	5.44219	426.433	**414**	139	5.80001	425.267	422	492	422	3600	490
6	5.48639	462.133	**455**	257	5.87109	471.8	455	332	460	3600	562
7	5.43422	394.267	**386**	120	5.74966	399.467	394	172	388	3600	494
8	5.42002	427.533	**414**	65	5.80267	417.733	**414**	381	**414**	3600	InFea
9	5.38488	468.5	**461**	216	5.77338	472.4	469	442	476	3600	539
10	5.48115	414.467	**406**	98	5.81969	415.3	407	336	414	3600	497
30 vessels, 7 berths - $p : f(x, 5, 2)$, $a : f(x, 2, 5)$, $b : f(x, 2, 5)$											
1	4.42206	589.8	**584**	69	4.32267	591.633	588	313	594	3600	InFea
2	4.33721	583.633	**563**	143	4.31193	582.7	567	325	Unknown	3600	InFea
3	4.31419	560.567	**546**	139	4.26428	558.4	548	295	554	3600	InFea
4	4.29948	601.2	**586**	80	4.35129	600.8	590	275	594	3600	InFea
5	4.28449	643.867	633	74	4.21272	647.9	634	451	**632**	3600	InFea
6	4.41932	534.667	524	68	4.21935	535.067	531	473	**519**	3600	InFea
7	4.30246	600.533	567	150	4.21311	590.067	**564**	451	567	3600	InFea
8	4.36342	544.967	**533**	98	4.24737	546.1	537	490	538	3600	InFea
9	4.35974	618.9	**598**	249	4.21599	614.033	**598**	420	615	3600	InFea
10	4.3448	719.769	689	117	4.26803	709.462	**687**	205	Unknown	3600	InFea
30 vessels, 7 berths - $p : f(x, 5, 2)$, $a : f(x, 2, 5)$, $b : f(x, 2, 2)$											
1	4.27634	586.033	583	68	4.17765	588.733	**580**	372	**580**	3600	635
2	4.25298	581.833	563	173	4.16081	580.1	564	452	**579**	3600	InFea
3	4.21417	547.1	541	84	4.14369	546.767	**538**	473	546	3600	InFea
4	4.24585	587.767	**577**	206	4.17538	586.2	**577**	288	583	3600	InFea
5	4.23992	630.5	**615**	70	4.1677	634.133	626	470	630	3600	InFea
6	4.31371	523.5	517	119	4.17379	526.767	524	266	**512**	3600	570
7	4.20656	572.433	560	197	4.17025	568.4	559	461	**552**	3600	InFea
8	4.29258	534.7	**526**	93	4.15368	538.367	532	351	537	3600	InFea
9	4.25596	601	594	78	4.16788	598.6	**593**	198	606	3600	InFea
10	4.26808	669.667	**657**	188	4.21991	678.567	667	495	678	3600	InFea

Table 37 – Comparison results - part I

	GA				PSO				CPLEX		
	Comp. time	Average obj.	Best solution	Iteration	Comp. time	Average obj.	Best sol.	Iteration	Objective	Comp. time	FCFS heuristic
30 vessels, 7 berths - $p: f(x,5,2)$, $a: f(x,2,5)$, $b: f(x,5,2)$											
1	4.16898	582.467	580	97	4.10468	583.067	580	429	**579**	3600	635
2	4.18	581.567	**563**	182	4.1355	582.267	572	466	564	3600	637
3	4.1415	545.8	541	93	4.08757	546.167	**540**	424	550	3600	608
4	4.15392	577.9	575	207	4.11182	577.9	**574**	220	575	3600	654
5	4.19005	629.533	623	69	4.10211	633.2	624	384	**615**	3600	696
6	4.5087	520.833	**509**	236	4.09863	526.167	517	476	**509**	3600	570
7	4.61945	565.933	**555**	108	4.10108	565.467	557	439	560	3600	636
8	4.57433	533.3	**527**	103	4.08356	537.9	533	387	536	3600	618
9	4.71918	592.2	589	90	4.10279	590.033	**588**	156	598	3600	650
10	4.61019	656.9	**653**	122	4.13314	667.4	663	345	656	3600	732
40 vessels, 5 berths - $p: f(x,5,2)$, $a: f(x,2,5)$, $b: f(x,5,2)$											
1	11.0973	664	**652**	299	12.2981	678.567	662	409	678	3600	842
2	10.8854	453.733	448	201	12.5186	449.7	**439**	483	450	3600	542
3	10.6708	385.433	**369**	138	12.466	384.4	374	442	**369**	3600	489
4	10.5934	501.1	493	336	12.4098	498.167	**492**	479	505	3600	625
5	10.8279	487.8	**480**	95	12.6039	494	483	490	482	3600	577
6	10.6114	451.733	438	128	12.6231	453.867	439	489	**428**	3600	576
7	11.2115	463.867	**455**	95	12.5223	470.2	460	424	472	3600	525
8	10.864	347.033	331	156	12.5044	355.733	350	370	**328**	3600	407
9	11.5177	420.833	**413**	282	12.605	420.867	415	405	**413**	3600	497
10	10.8037	566.5	**550**	212	12.7989	574.567	566	452	568	3600	698
40 vessels, 7 berths - $p: f(x,5,2)$, $a: f(x,2,5)$, $b: f(x,2,5)$											
1	7.52868	652.933	**632**	229	7.96548	667.967	648	488	Unknown	3600	InFea
2	7.53006	491.367	466	193	7.99012	495.033	470	500	**463**	3600	559
3	7.61987	718.464	**688**	316	8.04117	732.897	694	476	Unknown	3600	InFea
4	7.60239	554.567	**546**	103	7.93714	554.4	550	492	567	3600	InFea
5	7.44254	577	564	420	7.89156	575.4	563	402	**560**	3600	InFea
6	7.546	586.7	576	143	7.98173	594.533	584	457	**571**	3600	InFea
7	7.54718	519.6	**508**	136	7.88165	527.6	520	461	512	3600	InFea
8	7.64337	680.267	**660**	166	8.03195	686.567	663	448	Unknown	3600	InFea
9	7.41872	479.6	**472**	157	7.95603	487.933	474	323	474	3600	InFea
10	7.60144	622.667	**604**	139	7.9437	627.7	613	499	Unknown	3600	InFea
40 vessels, 7 berths - $p: f(x,5,2)$, $a: f(x,2,5)$, $b: f(x,0.5,0.5)$											
1	7.56658	642.433	**615**	169	8.00369	672.1	639	409	651	3600	InFea
2	7.59673	493.6	476	291	8.09169	506.767	470	491	**469**	3600	InFea
3	7.59802	705.767	**675**	398	8.19223	739.067	694	495	Unknown	3600	InFea
4	7.65005	565.8	**552**	189	7.92617	567.567	554	444	563	3600	InFea
5	7.474	572.533	559	274	7.89535	570.133	554	460	**553**	3600	InFea
6	7.55181	593.333	**574**	223	8.06791	617.033	608	413	644	3600	InFea
7	7.5683	523.333	**508**	144	7.87521	527.2	518	459	514	3600	InFea
8	7.5857	666.3	**637**	313	8.13508	682.267	661	486	671	3600	InFea
9	7.49133	490.5	476	147	8.14429	494.767	480	401	**474**	3600	InFea
10	7.6489	642.833	**624**	232	8.02488	650.867	627	472	653	3600	InFea
40 vessels, 7 berths - $p: f(x,5,2)$, $a: f(x,2,5)$, $b: f(x,5,2)$											
1	7.40226	635.667	627	92	7.80689	633.767	**604**	416	614	3600	729
2	7.42561	492.933	**474**	151	7.83343	501	492	424	**474**	3600	559
3	7.40462	658.733	**646**	140	7.83034	663.433	652	321	663	3600	821
4	7.57102	541.3	539	193	7.82683	541.967	**535**	482	545	3600	590
5	7.36938	565.3	555	215	7.72465	563.4	**553**	436	558	3600	636
6	7.33602	557.333	**544**	132	7.92143	565.3	560	487	550	3600	646
7	7.29626	518.733	**510**	122	7.99787	522.3	517	278	512	3600	589
8	7.46437	647.567	**632**	186	7.783	655.133	646	487	639	3600	747
9	7.32683	475.367	**468**	192	7.79072	476.5	471	435	476	3600	534
10	7.51488	606.167	598	174	7.91903	616.067	**597**	475	608	3600	737

Table 38 – Comparison results - part II

In 71% of the tests, GA outperforms PSO. In 67% of the tests, GA outperforms CPLEX.

In 11% of the tests, GA and CPLEX tie. In 5% of the tests, PSO and CPLEX tie.

In 7 instances, the CPLEX was not able to find a feasible solution to the problem or prove it as infeasible (status Unknown). In 6 out of such 7 instances, the GA found a feasible solution better than the one founded by PSO. In 58% of the tests, the FCFS was infeasible.

It is noteworthy that in Tables 37 and 38, when the computational time for CPLEX is less than 3600s, it means optimality was proven. It happened in 4 instances, and in 3 of then the GA reached the same solution, while PSO approached such optimal solution but did not reach it.

In the implementation of the algorithms, the infeasible solutions were penalized in the objective function (fitness) as follows:

$$\sum_{i,k \in I} x_i^k + p_i^b - b_i \tag{8.5}$$

set of vessel and berths indexes that violate time windows, i.e.,

$$x_i^k + p_i^b > b_i$$

Therefore, this penalty allowed us to observe that, in most instances, the GA was able to reach the feasibility of the solutions faster than the PSO. This behavior is exemplified with the instance 1 for 30 vessels, 5 berths, processing time generated with $f(x, 5, 2)$, arrival time generated with $f(x, 2, 5)$ and departure time generated with $f(x, 2, 2)$. In Figure 26 we can see this progression of the solution over the gener0ations.

In this case, the GA obtained a completely feasible population in iteration 8 with an average objective function of value 706.31. The PSO obtained a completely feasible population in the iteration 156 with average objective function of value 966.145. This justifies the fact that the best solution (with objective function value 357 obtained by both algorithms) has been achieved by the GA in iteration 65 and by the PSO only in iteration 316.

8.4 Conclusion

Developing countries are gaining greater market share in world merchandise trade and it brings job and opportunities, but their ports lack the infrastructure for bigger vessels becoming a bottleneck in the global business operations. Optimizing the

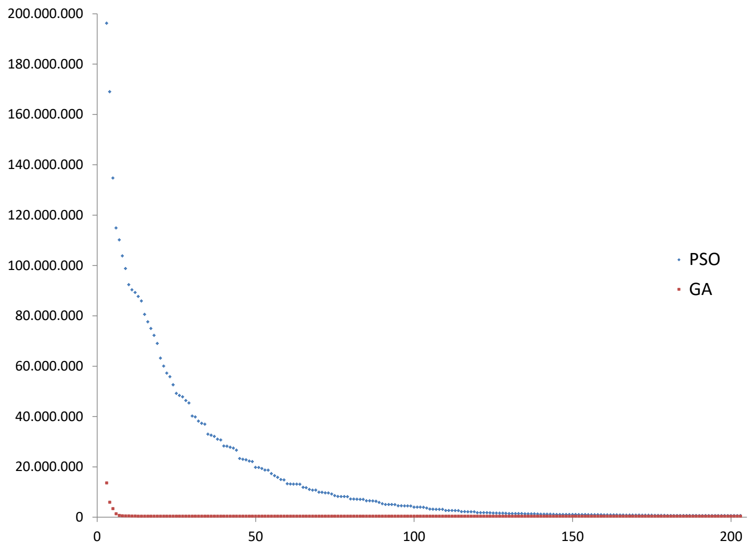


Figure 26 – Progression of the solution over the generations.

shipping lines on routes in Africa, Asia and Latin America means ports in these regions will have to improve performance. Unless they spend heavily on training and human skills development and also in building bigger and more efficient terminals, they may face fewer port calls, less competitive markets and higher shipping costs. The increasing demand for maritime transportation suggests for this ports to search for a logistics to accommodate the incoming vessels minimizing the waiting and service times of vessels and the handling costs, giving raise for the Bert Allocation Problem.

Analyzing the results obtained with the metaheuristics, we noticed that in computational times, GA and PSO were very similar. However, the GA was shown to be faster in the solution quality progression, which can be seen by observing that the best solution was obtained in the smaller iterations. Thus, the GA was more competitive than the PSO. Another interesting point that we can highlight, was the solution of the PSO to the instance 3 for 20 vessels, 5 berths, processing time generated with $f(x, 5, 2)$, arrival time generated with $f(x, 2, 5)$ and departure time generated with $f(x, 2, 5)$. The FCFS heuristic solution is feasible, and we can see that all 30 PSO rounds converged to the FCFS heuristic solution, with average objective function and best solution with the same value, 357. GA has been able to achieve for such instance the solution that was proven by CPLEX to be optimal.

9 Conclusion

This work began in 2012 in a thematic project of Vale Company with Fapesp, when I started my master's degree. In (BARBOSA, 2014) the BAP was modeled as a cutting stock problem and as a scheduling problem. Both models were run in the CPLEX and GLPK and had their results compared to those of a contructive heuristic that was developed and with the FCFS heuristic. The results indicated a difficulty in obtaining the optimal solution, and sometimes even a feasible solution. Thus, based on the scheduling model in parallel machines, this thesis began in Chapter 3 by developing an algorithm based on NSGA-II to solve the multiobjective BAP - MOBAP. Two objectives were considered in this study: minimizing the sum of waiting times and minimizing the makespan. Although a latent conflict between the two objective functions chosen was not identified, the multiobjective approach was important to obtain a solution that represents the minimum values for the waiting time and the makespan. This behavior was not observed in the CPLEX results in which each objective was optimized separately. Therefore, the importance of the simultaneous optimization of multiple objectives in Operation Research is emphasized. For instances with 40 vessels, the multiobjective evolutionary algorithm clearly stood out, both for quality of the solutions obtained and low computational time. Thus, the approach proposed here proved to be competitive and effective for large instances. The scale and nature of this problem at large terminals often makes it impossible for the decisions made to be optimal because it is a combinatorial problem of the NP-Hard classes. Therefore, in Chapter 4 two adaptations for a maximal flow algorithm were proposed based on the scheduling problem to generate a lower bound to evaluate the previous MOBAP algorithm. The "Loss" was calculated as (makespan - lower bound)/lower bound and it was possible to prove that MOBAP found many optimal solutions.

Benders decomposition is a technique in mathematical programming that allows the solution of very large linear programming problems that have a special block structure. Its characteristic is that the best solution to a model is found automatically and for such reason, in Chapter 5 a description of the Benders Decomposition algorithm and its enhancements were given and then the algorithm was applied to the BAP. Results showed that, although being competitive with monolithic model resolution with CPLEX, in general Benders Decomposition does not outperform CPLEX. Compared to mathematical programming, metaheuristics do not guarantee that a globally optimal solution can be found, but they can find a solution that is *good enough* in a computing time that is *small enough*. Due to the inneficiency of the Benders decomposition, the remaining chapters of this thesis invested in the development of metaheuristics to tackle the BAP. In Chapter 6 five initializations, three crossover and three variations for the scatter search parameters were proposed for a hybrid evolutionary algorithm based on Genetic Algorithm (GA) and Scatter Search (SS) for the discrete and dynamic BAP - HEABAP. Data envelopment analysis (DEA) was adopted to choose the most efficient combination of the algorithmic operators and it was possible to measure if the amount of computational time taken by one combination compensated the value obtained by the objective function.

After the extensive literature review that was performed at the beginning of this work, it was observed that there is no well-established benchmark for the BAP in the literature. Therefore, in Chapter 7 with the purpose of creating benchmark problem instances to allow future work to be broadly fairly compared. 5760 computational tests were performed using CPLEX 12.6 and it was possible to classify the difficulty of the parameters combinations for the beta distribution. Based on the results from CPLEX, a Genetic Algorithm was developed and a Particle Swarm Algorithm was implemented to solve the most difficult instances in Chapter 8. It is noteworthy that, for the proposed comparisons, both algorithms were implemented with a similar structure: codification, initialization, fitness. Analyzing the results, we noticed that in computational times, GA and PSO were very similar. However, the GA was shown to be faster in the solution quality progression, because the best solution was obtained a smaller number of iterations. Thus, the GA was more competitive than the PSO.

9.1 Future work

As stated previously, the Berth Allocation Problem has shown to be of high resolution complexity, and metaheuristic methods stand out as a faster option to find good solutions. In this sense, there are still several studies that can be carried out.

In Chapter 3 it is possible to observe patterns in the best solutions obtained by the algorithm. For future research, we want to verify which features the best solutions have in common, and from this proposal, for example, algorithms to fix variable that include such characteristics as previous knowledge regarding the quality of solutions in order to reduce the computational time of the optimization process.

The Genetic Algorithm proposed in Chapter 8 can still be extended to the multiobjective case, and to evaluate the quality of the solutions, the lower bound developed in Chapter 4 can also be extended to the model proposed in the Section 2.2.3 where the processing time of the vessels varies according to the berth in which the vessel is allocated. (BRUCKER, 2006) shows it can be done through the construction of an expanded network.

9.2 Participation in conferences

- "A Benders Decomposition Approach to The Berth Allocation Problem", 8^{th} Industrial Engineering and Management Symposium (IEMS), Porto, Portugal (2017).
- "The Berth Allocation Problem as a Maximum Flow Problem", 28^{th} European Conference on Operational Research, 2016, Poznan, Poland.
- "The Berth Allocation Problem: Case Studies of Metaheuristics and Integer Programming", 27^{th} European Conference on Operational Research, 2015, Glasgow, Scotland.
- "Aspectos Teóricos e Computacionais do Problema de Alocação de Berços em Portos Marítmos", XVII CLAIO - Congresso Latino Ibero Americana de Investigación de Operaciones, 2014, Monterrey, México.

Bibliography

ABRAHAM, A.; JAIN, L. *Evolutionary Multiobjective Optimization: Theoretical Advances and Applications*. 1. Londres: Springer-Verlag, 2005. 1-6 p. Cited on page 95.

AN, Y. W.; YAN, H. S. Lagrangean relaxation approach to joint optimization for production planning and scheduling of synchronous assembly lines. *International Journal of Production Research*, v. 54, n. 22, p. 6718–6735, 2016. Cited on page 34.

ARANGO, C.; CORTÉS, P.; MUÃUZURI, J.; ONIEVA, L. Berth allocation planning in seville inland port by simulation and optimisation. *Advanced Engineering Informatics*, v. 25, n. 3, p. 452 – 461, 2011. ISSN 1474-0346. Special Section: Engineering informatics in port operations and logistics. Cited 2 times on pages 20 and 38.

ARANGO, C.; CORTÉS, P.; ONIEVA, L.; ESCUDERO, A. Simulation-optimization models for the dynamic berth allocation problem. *Computer-Aided Civil and Infrastructure Engineering*, Wiley Online Library, v. 28, n. 10, p. 769–779, 2013. Cited 2 times on pages 23 and 38.

BÄCK, T.; FOGEL, D. B.; MICHALEWICZ, Z. *Evolutionary computation 1: Basic algorithms and operators*. [S.l.]: CRC press, 2000. v. 1. Cited on page 113.

BÄCK, T.; FOGEL, D. B.; MICHALEWICZ, Z. *Evolutionary Computation 2: Advanced Algorithms and Operators*. [S.l.]: CRC press, 2000. v. 1. Cited on page 112.

BARBOSA, F. *Dissertação de Mestrado: O Problema de Alocação de Berços: Aspectos Teóricos e Computacionais*. Dissertação (Instituto de Matemática, Estatística e Computação Científica) — Faculdade de Engenharia Elétrica e de Computação, Universidade Estadual de Campinas, SP, 2014. Cited 4 times on pages 42, 49, 56, and 124.

BARROS, V. H.; COSTA, T. S.; OLIVEIRA, A. C.; LORENA, L. A. Model and heuristic for berth allocation in tidal bulk ports with stock level constraints. *Computers & Industrial Engineering*, v. 60, n. 4, p. 606 – 613, 2011. ISSN 0360-8352. Cited 2 times on pages 21 and 38.

BRUCKER, P. *Scheduling Algorithms*. 5. Osnabruck, Germany: Springer, 2006. Cited 5 times on pages 57, 59, 61, 62, and 125.

BUHRKAL, K.; ZUGLIAN, S.; ROPKE, S.; LARSEN, J.; LUSBY, R. Models for the discrete berth allocation problem: A computational comparison. *Transportation Research Part E: Logistics and Transportation Review*, v. 47, n. 4, p. 461 – 473, 2011. ISSN 1366-5545. Cited 8 times on pages 21, 24, 34, 36, 38, 40, 71, and 111.

CARAMIA, M.; MARI, R. A decomposition approach to solve a bilevel capacitated facility location problem with equity constraints. *Optimization Letters*, v. 10, n. 5, p. 997–1019, Jun 2016. Cited on page 68.

CHANG, D.; JIANG, Z.; YAN, W.; HE, J. Integrating berth allocation and quay crane assignments. *Transportation Research Part E: Logistics and Transportation Review*, v. 46, n. 6, p. 975 – 990, 2010. ISSN 1366-5545. Cited 2 times on pages 25 and 38.

CHEN, C.; LEE, S.-M.; SHEN, Q. An analytical model for the container loading problem. *European Journal of Operational Research*, Elsevier, v. 80, n. 1, p. 68–76, 1995. Cited on page 28.

CHEONG, C. Y.; TAN, K. C. *A Multi-Objective Multi-Colony Ant Algorithm for Solving the Berth Allocation Problem*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008. 333-350 p. Cited 3 times on pages 27, 38, and 99.

CHEONG, C. Y.; TAN, K. C.; LIU, D. K. Solving the berth allocation problem with service priority via multi-objective optimization. In: *Computational Intelligence in Scheduling, 2009. CI-Sched '09. IEEE Symposium on*. [S.l.: s.n.], 2009. p. 95–102. Cited on page 41.

CHEONG, C. Y.; TAN, K. C.; LIU, D. K.; LIN, C. J. Multi-objective and prioritized berth allocation in container ports. *Annals of Operations Research*, v. 180, n. 1, p. 63–103, 2010. ISSN 1572-9338. Cited 3 times on pages 28, 38, and 43.

CODATO, G.; FISCHETTI, M. Combinatorial benders' cuts for mixed integer linear programming. *Operations Research*, v. 54, n. 4, p. 756–766, 2006. Cited on page 70.

COELLO, C. A. C.; ZACATENCO, C. S. P. Evolutionary multiobjective optimization: Past, present and future. *Tutorial-CINVESTAV-IPN, Depto. de Ingenierıa Eléctrica, Sección de Computación*, 2002. Cited on page 42.

COOK, W. D.; SEIFORD, L. M. Data envelopment analysis (dea)–thirty years on. *European journal of operational research*, Elsevier, v. 192, n. 1, p. 1–17, 2009. Cited on page 88.

COOPER, W. W.; SEIFORD, L. M.; ZHU, J. *Handbook on data envelopment analysis*. [S.l.]: Springer Science & Business Media, 2011. v. 164. Cited on page 85.

CORDEAU, J.-F.; LAPORTE, G.; LEGATO, P.; MOCCIA, L. Models and tabu search heuristics for the berth-allocation problem. *Transportation Science*, v. 39, n. 4, p. 526–538, 2005. Cited 6 times on pages 21, 36, 38, 71, 73, and 90.

DAI, J.; LIN, W.; MOORTHY, R.; TEO, C.-P. *Berth Allocation Planning Optimization in Container Terminals*. Boston, MA: Springer US, 2008. 69–104 p. ISBN 978-0-387-75240-2. Cited 3 times on pages 30, 38, and 85.

DEB, K.; AGRAWAL, S.; PRATAB, A.; MEYARIVA, T. A fast elitist non-dominated sorting genetic algorithm for multi-objective optimization: NSGA-II. In: *Proceedings of VI Conference Parallel Problem Solving from Nature*. Paris, France: Springer-Verlag, 2000. v. 1917, p. 849–858. Cited on page 42.

DEB, K.; AGRAWAL, S.; PRATAB, A.; MEYARIVAN, T. A fast and elitist multi-objective genetic algorithm: NSGA-II. *IEEE Trans. Evol. Comput.*, v. 6, n. 2, p. 182–197, 2002. Cited 6 times on pages 10, 42, 44, 46, 47, and 49.

DEBELS, D.; REYCK, B. D.; LEUS, R.; VANHOUCKE, M. A hybrid scatter search/electromagnetism meta-heuristic for project scheduling. *European Journal of Operational Research*, v. 169, n. 2, p. 638 – 653, 2006. ISSN 0377-2217. Feature Cluster on Scatter Search Methods for Optimization. Cited on page 76.

DU, K.-L.; SWAMY, M. N. S. *Particle Swarm Optimization*. Cham: Springer International Publishing, 2016. 153-173 p. Cited on page 114.

DU, Y.; CHEN, Q.; LAM, J. S. L.; XU, Y.; CAO, J. X. Modeling the impacts of tides and the virtual arrival policy in berth allocation. *Transportation Science*, v. 49, n. 4, p. 939–956, 2015. Cited 2 times on pages 29 and 38.

DU, Y.; CHEN, Q.; QUAN, X.; LONG, L.; FUNG, R. Y. Berth allocation considering fuel consumption and vessel emissions. *Transportation Research Part E: Logistics and Transportation Review*, v. 47, n. 6, p. 1021 – 1037, 2011. ISSN 1366-5545. Cited 2 times on pages 30 and 38.

EBERHART; SHI, Y. Particle swarm optimization: developments, applications and resources. In: *Proceedings of the 2001 Congress on Evolutionary Computation (IEEE Cat. No.01TH8546)*. [S.l.: s.n.], 2001. v. 1, p. 81–86 vol. 1. Cited on page 114.

EBERHART, R. C.; SHI, Y. *Comparison between genetic algorithms and particle swarm optimization*. Berlin, Heidelberg: Springer Berlin Heidelberg, 1998. 611–616 p. Cited on page 115.

ELIIYI, D. T.; KORKMAZ, A. G.; ÇIÇEK, A. E. Optimal berth allocation with variable job scheduling. In: *204th EURO Mini Conference Continuous Optimization and Knowledge-Based Technologies*. [S.l.: s.n.], 2008. Cited on page 85.

FERREIRA, P. A. V. *Otimização Multiobjetivo: Teoria e Aplicações*. Tese (Tese de Livre Docência) — Faculdade de Engenharia Elétrica e de Computação, Universidade Estadual de Campinas, SP, 1999. Cited on page 41.

FROJAN, P.; CORRECHER, J. F.; ALVAREZ-VALDES, R.; KOULOURIS, G.; TAMARIT, J. M. The continuous berth allocation problem in a container terminal with multiple quays. *Expert Systems with Applications*, v. 42, n. 21, p. 7356 – 7366, 2015. ISSN 0957-4174. Cited 3 times on pages 22, 38, and 111.

GANJI, S. R. S.; BABAZADEH, A.; ARABSHAHI, N. Analysis of the continuous berth allocation problem in container ports using a genetic algorithm. *Journal of Marine Science and Technology*, v. 15, n. 4, p. 408–416, 2010. Cited 2 times on pages 30 and 39.

GEOFFRION, A. M. Generalized benders decomposition. *Journal of Optimization Theory and Applications*, v. 10, n. 4, p. 237–260, 1972. Cited on page 69.

GIALLOMBARDO, G.; MOCCIA, L.; SALANI, M.; VACCA, I. The tactical berth allocation problem with quay crane assignment and transshipment-related quadratic yard costs. *Proceedings of the European Transport Conference*, p. 1–27, 2008. Cited 2 times on pages 25 and 38.

GIALLOMBARDO, G.; MOCCIA, L.; SALANI, M.; VACCA, I. Modeling and solving the tactical berth allocation problem. *Transportation Research Part B: Methodological*, v. 44, n. 2, p. 232 – 245, 2010. ISSN 0191-2615. Cited 2 times on pages 25 and 38.

GOLDBERG, D. E. *Genetic Algorithms in Search, Optimization and Machine Learning*. 1. Reading, USA: Addison-Wesley, 1989. Cited on page 112.

GOLDBERG, D. E.; HOLLAND, J. H. Genetic algorithms and machine learning. *Machine Learning*, v. 3, n. 2, p. 95–99, Oct 1988. Cited on page 76.

GOLIAS, M. M.; BOILE, M.; THEOFANIS, S. Berth scheduling by customer service differentiation: A multi-objective approach. *Transportation Research Part E: Logistics and Transportation Review*, v. 45, n. 6, p. 878 – 892, 2009. ISSN 1366-5545. Cited 4 times on pages 25, 38, 41, and 76.

GOLIAS, M. M.; BOILE, M.; THEOFANIS, S. Berth scheduling by customer service differentiation: A multi-objective approach. *Transportation Research Part E: Logistics and Transportation Review*, v. 45, n. 6, p. 878 – 892, 2009. ISSN 1366-5545. Cited 2 times on pages 28 and 38.

GONZÁLEZ, B.; ADENSO-DÍAZ, B. A scatter search approach to the optimum disassembly sequence problem. *Computers & Operations Research*, v. 33, n. 6, p. 1776 – 1793, 2006. ISSN 0305-0548. Cited on page 76.

GUAN, Y.; CHEUNG, R. K. The berth allocation problem: models and solution methods. *OR Spectrum*, v. 26, n. 1, p. 75–92, 2004. Cited 3 times on pages 20, 30, and 38.

HAN, Y.; GONG, D.; LI, J.; ZHANG, Y. Solving the blocking flow shop scheduling problem with makespan using a modified fruit fly optimisation algorithm. *International Journal of Production Research*, v. 54, n. 22, p. 6782–6797, 2016. Cited on page 32.

HANSEN, P.; OGUZ., C. A note on formulations of static and dynamic berth allocation problems. *Les Cahiers du GERAD*, v. 30, p. 1–17, 2003. Cited 2 times on pages 22 and 38.

HANSEN, P.; OGUZ, C.; MLADENOVIĆ, N. Variable neighborhood search for minimum cost berth allocation. *European Journal of Operational Research*, v. 191, n. 3, p. 636 – 649, 2008. ISSN 0377-2217. Cited 2 times on pages 21 and 38.

HENDRIKS, M. P. M.; LEFEBER, E.; UDDING, J. T. Simultaneous berth allocation and yard planning at tactical level. *OR Spectrum*, v. 35, n. 2, p. 441–456, 2013. Cited 2 times on pages 27 and 38.

HUANG, Z.; ZHENG, Q. P. Decomposition-based exact algorithms for risk-constrained traveling salesman problems with discrete random arc costs. *Optimization Letters*, v. 9, n. 8, p. 1553–1568, Dec 2015. Cited on page 68.

IMAI, A.; CHEN, H. C.; NISHIMURA, E.; PAPADIMITRIOU, S. The simultaneous berth and quay crane allocation problem. *Transportation Research Part E: Logistics and Transportation Review*, v. 44, n. 5, p. 900 – 920, 2008. ISSN 1366-5545. Cited 3 times on pages 24, 38, and 57.

IMAI, A.; NISHIMURA, E.; HATTORI, M.; PAPADIMITRIOU, S. Berth allocation at indented berths for mega-containerships. *European Journal of Operational Research*, Elsevier, v. 179, n. 2, p. 579–593, 2007. Cited 2 times on pages 23 and 38.

IMAI, A.; NISHIMURA, E.; PAPADIMITRIOU, S. The dynamic berth allocation problem for a container port. *Transportation Research Part B: Methodological*, v. 35, n. 4, p. 401 – 417, 2001. ISSN 0191-2615. Cited 3 times on pages 22, 24, and 38.

IMAI, A.; NISHIMURA, E.; PAPADIMITRIOU, S. Berth allocation with service priority. *Transportation Research Part B: Methodological*, v. 37, n. 5, p. 437 – 457, 2003. ISSN 0191-2615. Cited 2 times on pages 23 and 38.

IMAI, A.; NISHIMURA, E.; PAPADIMITRIOU, S. Berthing ships at a multi-user container terminal with a limited quay capacity. *Transportation Research Part E: Logistics and Transportation Review*, v. 44, n. 1, p. 136 – 151, 2008. ISSN 1366-5545. Cited 2 times on pages 23 and 38.

IMAI, A.; SUN, X.; NISHIMURA, E.; PAPADIMITRIOU, S. Berth allocation in a container port: using a continuous location space approach. *Transportation Research Part B: Methodological*, v. 39, n. 3, p. 199 – 221, 2005. ISSN 0191-2615. Cited 2 times on pages 20 and 38.

IMAI, A.; ZHANG, J.-T.; NISHIMURA, E.; PAPADIMITRIOU, S. The berth allocation problem with service time and delay time objectives. *Maritime Economics & Logistics*, Springer, v. 9, n. 4, p. 269–290, 2007. Cited 2 times on pages 28 and 38.

KESKIN, B. B.; USTER, H. A scatter search-based heuristic to locate capacitated transshipment points. *Computers & Operations Research*, v. 34, n. 10, p. 3112 – 3125, 2007. ISSN 0305-0548. Cited on page 76.

KHODABANDEH, E.; BAI, L.; HERAGU, S. S.; EVANS, G. W.; ELROD, T.; SHIRKNESS, M. Modelling and solution of a large-scale vehicle routing problem at ge appliances & lighting. *International Journal of Production Research*, v. 55, n. 4, p. 1100–1116, 2017. Cited on page 34.

KNOWLES, J.; CORNE, D.; DEB, K. *Multiobjective Problem Solving from Nature: From Concepts to Applications*. 1. Berlin: Springer-Verlag, 2008. Cited on page 44.

KOUIDER, A.; HADDADèNE, H. A.; OURARI, S.; OULAMARA, A. Mixed graph colouring for unit-time scheduling. *International Journal of Production Research*, v. 55, n. 6, p. 1720–1729, 2017. Cited on page 31.

KURZ, M. E.; ASKIN, R. G. Scheduling flexible flow lines with sequence-dependent setup times. *European Journal of Operational Research*, v. 159, n. 1, p. 66 – 82, 2004. ISSN 0377-2217. Cited 3 times on pages 57, 66, and 112.

LABBI, W.; BOUDHAR, M.; OULAMARA, A. Scheduling two identical parallel machines with preparation constraints. *International Journal of Production Research*, v. 55, n. 6, p. 1531–1548, 2017. Cited on page 31.

LAGUNA, M.; MARTI, R. *Scatter search: methodology and implementations in C*. [S.l.]: Springer Science & Business Media, 2012. v. 24. Cited on page 77.

LALLA-RUIZ, E.; GONZáLEZ-VELARDE, J. L.; MELIáN-BATISTA, B.; MORENO-VEGA, J. M. Biased random key genetic algorithm for the tactical berth allocation problem. *Applied Soft Computing*, v. 22, p. 60 – 76, 2014. ISSN 1568-4946. Cited 2 times on pages 24 and 38.

LALLA-RUIZ, E.; MELIÁN-BATISTA, B.; MORENO-VEGA, J. M. Artificial intelligence hybrid heuristic based on tabu search for the dynamic berth allocation problem. *Engineering Applications of Artificial*, v. 25, n. 6, p. 1132 – 1141, 2012. ISSN 0952-1976. Cited 3 times on pages 36, 37, and 38.

LALLA-RUIZ, E.; VOSS, S. Popmusic as a matheuristic for the berth allocation problem. *Annals of Mathematics and Artificial Intelligence*, v. 76, n. 1, p. 173–189, 2016. ISSN 1573-7470. Cited 3 times on pages 36, 37, and 38.

LEE, D.-H.; CHEN, J. H.; CAO, J. X. The continuous berth allocation problem: A greedy randomized adaptive search solution. *Transportation Research Part E: Logistics and Transportation Review*, v. 46, n. 6, p. 1017 – 1029, 2010. ISSN 1366-5545. Cited 2 times on pages 29 and 38.

LEE, D.-H.; WANG, H. Q. Integrated discrete berth allocation and quay crane scheduling in port container terminals. *Engineering Optimization*, v. 42, n. 8, p. 747–761, 2010. Cited 2 times on pages 24 and 38.

LEE, Y.; CHEN, C.-Y. An optimization heuristic for the berth scheduling problem. *European Journal of Operational Research*, v. 196, n. 2, p. 500 – 508, 2009. ISSN 0377-2217. Cited 2 times on pages 30 and 38.

LI, C.-L.; PANG, K.-W. An integrated model for ship routing and berth allocation. *International Journal of Shipping and Transport Logistics*, v. 3, n. 3, p. 245–260, 2011. Cited 2 times on pages 20 and 38.

LI, X.; LI, Y.; WANG, Y. Minimising makespan on a batch processing machine using heuristics improved by an enumeration scheme. *International Journal of Production Research*, v. 55, n. 1, p. 176–186, 2017. Cited on page 31.

LIANG, C.; HUANG, Y.; YANG, Y. A quay crane dynamic scheduling problem by hybrid evolutionary algorithm for berth allocation planning. *Computers and Industrial Engineering*, v. 56, n. 3, p. 1021 – 1028, 2009. ISSN 0360-8352. Intelligent Manufacturing and Logistics. Cited 2 times on pages 24 and 38.

LIANG, C.; HWANG, H.; GEN, M. A berth allocation planning problem with direct transshipment consideration. *Journal of Intelligent Manufacturing*, v. 23, n. 6, p. 2207–2214, 2012. Cited 2 times on pages 24 and 38.

LIU, Y.-H. A hybrid scatter search for the probabilistic traveling salesman problem. *Computers & Operations Research*, v. 34, n. 10, p. 2949 – 2963, 2007. ISSN 0305-0548. Cited on page 76.

LU, C.-C.; YU, V. F. Data envelopment analysis for evaluating the efficiency of genetic algorithms on solving the vehicle routing problem with soft time windows. *Computers & Industrial Engineering*, v. 63, n. 2, p. 520 – 529, 2012. Cited 2 times on pages 88 and 92.

MAENHOUT, B.; VANHOUCKE, M. A hybrid scatter search heuristic for personalized crew rostering in the airline industry. *European Journal of Operational Research*, v. 206, n. 1, p. 155 – 167, 2010. ISSN 0377-2217. Cited on page 76.

MAGNANTI, T. L.; WONG, R. T. Accelerating benders decomposition: Algorithmic enhancement and model selection criteria. *Operations Research*, v. 29, n. 3, p. 464–484, 1981. Cited on page 73.

MAHVASH, B.; AWASTHI, A.; CHAUHAN, S. A column generation based heuristic for the capacitated vehicle routing problem with three-dimensional loading constraints. *International Journal of Production Research*, v. 55, n. 6, p. 1730–1747, 2017. Cited on page 34.

MARTIN, M. P.; CAVALHEIRO, E. M. B.; BARBOSA, F.; FILHO, C. L.; MORETTI, A. C. Métodos exatos e uma abordagem heurística para o problema de alocação de berços. In: . [S.l.: s.n.], 2015. p. 1569–1580. Cited 2 times on pages 28 and 38.

MASCHIETTO, G. N.; OUAZENE, Y.; RAVETTI, M. G.; SOUZA, M. C. de; YALAOUI, F. Crane scheduling problem with non-interference constraints in a steel coil distribution centre. *International Journal of Production Research*, v. 55, n. 6, p. 1607–1622, 2017. Cited on page 31.

MEISEL, F.; BIERWIRTH, C. A framework for integrated berth allocation and crane operations planning in seaport container terminals. *Transportation Science*, v. 47, n. 2, p. 131–147, 2013. Cited 2 times on pages 26 and 39.

MERCIER, A.; CORDEAU, J.-F.; SOUMIS, F. A computational study of benders decomposition for the integrated aircraft routing and crew scheduling problem. *Computers & Operations Research*, v. 32, n. 6, p. 1451 – 1476, 2005. ISSN 0305-0548. Cited on page 68.

MICHALEWICZ, Z. *Genetic Algorithms + Data Structures = Evolution Programs*. 3^{rd}. Berlin, Germany: Springer-Verlag, 1996. Cited on page 113.

MONACO, M. F.; SAMMARRA, M. The berth allocation problem: A strong formulation solved by a lagrangean approach. *Transportation Science*, v. 41, n. 2, p. 265–280, 2007. Cited 5 times on pages 16, 22, 33, 39, and 71.

OLIVEIRA, R. M. de; MAURI, G. R.; LORENA, L. A. N. Clustering search for the berth allocation problem. *Expert Systems with Applications*, v. 39, n. 5, p. 5499 – 5505, 2012. ISSN 0957-4174. Cited 2 times on pages 36 and 38.

OSYCZKA, A. Multicriteria optimization for engineering design. *J. S. Gero (Ed.), Design Optimization*, NewJersey: Academic Press, p. 193–227, 1985. Cited on page 41.

OZTURK, O.; BEGEN, M. A.; ZARIC, G. S. A branch and bound algorithm for scheduling unit size jobs on parallel batching machines to minimize makespan. *International Journal of Production Research*, v. 55, n. 6, p. 1815–1831, 2017. Cited on page 31.

PATAKI, G. Teaching integer programming formulations using the traveling salesman problem. *SIAM REV*, v. 45, p. 116–123, 2003. Cited on page 72.

PRATAP, S.; B, M. K.; SAXENA, D.; TIWARI, M. Integrated scheduling of rake and stockyard management with ship berthing: a block based evolutionary algorithm. *International Journal of Production Research*, v. 54, n. 14, p. 4182–4204, 2016. Cited on page 26.

PRATAP, S.; NAYAK, A.; CHEIKHROUHOU, N.; TIWARI, M. K. Decision support system for discrete robust berth allocation. *IFAC-PapersOnLine*, v. 48, n. 3, p. 875 – 880, 2015. ISSN 2405-8963. Cited 2 times on pages 41 and 76.

RAA, B.; DULLAERT, W.; SCHAEREN, R. V. An enriched model for the integrated berth allocation and quay crane assignment problem. *Expert Systems with Applications*, v. 38, n. 11, p. 14136 – 14147, 2011. ISSN 0957-4174. Cited 2 times on pages 26 and 39.

RAHMANIANI, R.; CRAINIC, T. G.; GENDREAU, M.; REI, W. The benders decomposition algorithm: A literature review. *European Journal of Operational Research*, v. 259, n. 3, p. 801 – 817, 2017. ISSN 0377-2217. Cited on page 68.

RASMUSSEN, R. V.; TRICK, M. A. A Benders approach for the constrained minimum break problem. *European Journal of Operational Research*, v. 177, n. 1, p. 198–213, February 2007. Cited on page 68.

ROBENEK, T.; UMANG, N.; BIERLAIRE, M.; ROPKE, S. A branch-and-price algorithm to solve the integrated berth allocation and yard assignment problem in bulk ports. *European Journal of Operational Research*, v. 235, n. 2, p. 399 – 411, 2014. ISSN 0377-2217. Maritime Logistics. Cited 2 times on pages 27 and 39.

RUSSELL, R. A.; CHIANG, W.-C. Scatter search for the vehicle routing problem with time windows. *European Journal of Operational Research*, v. 169, n. 2, p. 606 – 622, 2006. ISSN 0377-2217. Feature Cluster on Scatter Search Methods for Optimization. Cited on page 76.

SANCHES, D. S.; ROCHA, J. d. S.; CASTOLDI, M. F.; MORANDIN, O.; KATO, E. R. R. An adaptive genetic algorithm for production scheduling on manufacturing systems with simultaneous use of machines and agvs. *Journal of Control, Automation and Electrical Systems*, v. 26, n. 3, p. 225–234, 2015. Cited on page 33.

SAWARAGI, Y.; NAKAYAMA, H.; TANINO, T. *Theory of Multiobjective Optimization*. 1. Orlando, Florida, USA: Academic Press, 1985. Cited 2 times on pages 44 and 95.

SCHEEL, H. Undesirable outputs in efficiency valuations. *European Journal of Operational Research*, v. 132, n. 2, p. 400 – 410, 2001. Data Envelopment Analysis. Cited on page 89.

SILVA, E.; OLIVEIRA, J. F.; WASCHER, G. 2dcpackgen: A problem generator for two-dimensional rectangular cutting and packing problems. *European Journal of Operational Research*, v. 237, n. 3, p. 846 – 856, 2014. ISSN 0377-2217. Cited on page 99.

SIMRIN, A.; DIABAT, A. The dynamic berth allocation problem: A linearized formulation. *RAIRO-Operations Research*, EDP Sciences, v. 49, n. 3, p. 473–494, 2015. Cited 2 times on pages 23 and 39.

SOTSKOV, Y. N.; GHOLAMI, O. Mixed graph model and algorithms for parallel-machine job-shop scheduling problems. *International Journal of Production Research*, v. 55, n. 6, p. 1549–1564, 2017. Cited on page 31.

TAHA, H. A. *Operations Research: an introduction*. 8. New York: Macmillan, 2007. Cited 3 times on pages 63, 64, and 65.

TANG, L.; JIANG, W.; SAHARIDIS, G. K. An improved benders decomposition algorithm for the logistics facility location problem with capacity expansions. *Annals of Operations Research*, Springer, v. 210, n. 1, p. 165–190, 2013. Cited on page 71.

TELLACHE, N. E. H.; BOUDHAR, M. Two-machine flow shop problem with unit-time operations and conflict graph. *International Journal of Production Research*, v. 55, n. 6, p. 1664–1679, 2017. Cited on page 31.

THEOFANIS, S.; BOILE, M.; GOLIAS, M. An optimization based genetic algorithm heuristic for the berth allocation problem. In: *2007 IEEE Congress on Evolutionary Computation*. [S.l.: s.n.], 2007. p. 4439–4445. ISSN 1089-778X. Cited 2 times on pages 25 and 39.

THEVENIN, S.; ZUFFEREY, N.; POTVIN, J.-Y. Makespan minimisation for a parallel machine scheduling problem with preemption and job incompatibility. *International Journal of Production Research*, v. 55, n. 6, p. 1588–1606, 2017. Cited on page 32.

TING, C.-J.; WU, K.-C.; CHOU, H. Particle swarm optimization algorithm for the berth allocation problem. *Expert Systems with Applications*, v. 41, n. 4, Part 1, p. 1543 – 1550, 2014. ISSN 0957-4174. Cited 5 times on pages 36, 39, 112, 114, and 116.

TING, C.-J.; WU, K.-C.; CHOU, H. Particle swarm optimization algorithm for the berth allocation problem. *Expert Systems with Applications*, Elsevier, v. 41, n. 4, p. 1543–1550, 2014. Cited 3 times on pages 71, 78, and 84.

T'KINDT, V.; BILLAUT, J.-C. *Multicriteria scheduling: theory, models and algorithms*. [S.l.]: Springer Science & Business Media, 2006. Cited on page 85.

TRIGOS, F.; LÓPEZ, E. M. A vulcanising decision planning as a particular one-dimensional cutting stock problem with limited part-related tooling in make-to-order industrial environments. *International Journal of Production Research*, v. 55, n. 10, p. 2881–2896, 2017. Cited on page 28.

TURKOGULLARI, Y. B.; TASKIN, Z. C.; ARAS, N.; ALTINEL, I. K. Optimal berth allocation and time-invariant quay crane assignment in container terminals. *European Journal of Operational Research*, v. 235, n. 1, p. 88 – 101, 2014. ISSN 0377-2217. Cited 2 times on pages 24 and 39.

UMANG, N.; BIERLAIRE, M.; VACCA, I. Exact and heuristic methods to solve the berth allocation problem in bulk ports. *Transportation Research Part E: Logistics and Transportation Review*, v. 54, p. 14 – 31, 2013. ISSN 1366-5545. Cited 3 times on pages 36, 37, and 39.

UNCTAD. *United Nations Conference on Trade and Development - Review of maritime transport*. 2016. Disponível em: <http://www.unctad.org>. Cited on page 16.

VACCA, I.; SALANI, M.; BIERLAIRE, M. An exact algorithm for the integrated planning of berth allocation and quay crane assignment. *Transportation Science*, v. 47, n. 2, p. 148–161, 2013. Cited 2 times on pages 24 and 39.

VATSA, A. K.; JAYASWAL, S. A new formulation and benders decomposition for the multi-period maximal covering facility location problem with server uncertainty. *European Journal of Operational Research*, v. 251, n. 2, p. 404 – 418, 2016. Cited on page 73.

VOSS, S.; LALLA-RUIZ, E. A set partitioning reformulation for the multiple-choice multidimensional knapsack problem. *Engineering Optimization*, v. 48, n. 5, p. 831–850, 2016. Cited on page 36.

XU, D.; LI, C.-L.; LEUNG, J. Y.-T. Berth allocation with time-dependent physical limitations on vessels. *European Journal of Operational Research*, v. 216, n. 1, p. 47 – 56, 2012. ISSN 0377-2217. Cited 2 times on pages 32 and 39.

YAGIURA, M.; IBARAKI, T. On metaheuristic algorithms for combinatorial optimization problems. *Systems and Computers in Japan*, John Wiley & Sons, Inc., v. 32, n. 3, p. 33–55, 2001. ISSN 1520-684X. Cited on page 16.

YANG, C.; WANG, N. Notice of retraction a solution method for berth-quay crane allocation problem. In: *Computer Engineering and Technology (ICCET), 2010 2nd International Conference on*. [S.l.: s.n.], 2010. v. 2, p. V2–537–V2–541. Cited on page 41.

YANG, C.; WANG, X.; LI, Z. An optimization approach for coupling problem of berth allocation and quay crane assignment in container terminal. *Computers & Industrial Engineering*, v. 63, n. 1, p. 243 – 253, 2012. ISSN 0360-8352. Cited 2 times on pages 26 and 39.

ZHU, Z.; ZHENG, F.; CHU, C. Multitasking scheduling problems with a rate-modifying activity. *International Journal of Production Research*, v. 55, n. 1, p. 296–312, 2017. Cited on page 32.