Nature Inspired Metaheuristics for Optimizing Problems at a Container Terminal

Gulić, Marko; Maglić, Livia; Valčić, Sanjin

Source / Izvornik: **Pomorstvo, 2018, 32, 10 - 20**

Journal article, Published version
Rad u časopisu, Objavljena verzija rada (izdavačev PDF)

https://doi.org/10.31217/p.32.1.16

Permanent link / Trajna poveznica: https://urn.nsk.hr/urn:nbn:hr:187:733584

Rights / Prava: In copyright / Zaštićeno autorskim pravom.

Download date / Datum preuzimanja: **2024-07-17**

Repository / Repozitorij:

Repository of the University of Rijeka, Faculty of Maritime Studies - FMSRI Repository

Multidisciplinary
SCIENTIFIC JOURNAL OF
MARITIME RESEARCH

University of Rijeka
Faculty of Maritime
Studies Rijeka

Multidisciplinarni
znanstveni časopis
POMORSTVO

Nature Inspired Metaheuristics for Optimizing Problems at a Container Terminal

Marko Gulić, Livia Maglić, Sanjin Valčić

University of Rijeka, Faculty of Maritime Studies, Studentska 2, 51000 Rijeka, Croatia, e-mail: mgulic@pfri.hr

ABSTRACT

Nowadays, maritime transport is the backbone of the international trade of goods. Therefore, seaports play a very important role in global transport. The use of containers is significantly represented in the maritime transport. Considering the increased number of container shipments in the global transport, seaport container terminals have to be adapted to a new situation and provide the best possible service of container transfer by reducing the transfer cost and the container transit time. Therefore, there is a need for optimization of the whole container transport process within the terminal. The logistic problems of the container terminals have become very complex and logistics experts cannot manually adjust the operations of terminal processes that will optimize the usage of resources. Hence, to achieve further improvements of terminal logistics, there is a need to introduce scientific methods such as metaheuristics that will enable better and optimized use of the terminal resources in an automated way. There is a large number of research papers that have successfully proposed the solutions of optimizing the container logistic problems with well-known metaheuristics inspired by the nature. However, there is a continuous emergence of new nature inspired metaheuristics today, like artificial bee colony algorithm, firefly algorithm and bat algorithm, that outperform the well-known metaheuristics considering the most popular optimization problems like travel salesman problem. Considering these results of comparing algorithms, we assume that better results of optimization of container terminal logistic problems can be achieved by introducing these new nature inspired metaheuristics. In this paper we have described and classified the main subsystems of the container terminal and its logistic problems that need to be optimized. We have also presented a review of new nature inspired metaheuristics (bee, firefly and bat algorithm) that could be used in the optimization of these problems within the terminal.

ARTICLE INFO

Review article
Received 4 March 2018
Accepted 3 May 2018

Key words:
Container terminal
Optimization problems
Nature inspired metaheuristics
Bee algorithm
Firefly algorithm
Bat algorithm

1 Introduction

Today, maritime transport is the backbone of the global trade of goods. It is generally accepted that more than 90% of global trade is carried by sea [1]. Consequently, seaports have become the most important junction of global trade. Considering the modes of transportation by sea, in 2016, more than 15% of the overall maritime transportation refers to the transport of goods by containers [2], therefore, a container transport plays an essential role in the world trade. According to [3], containers are large boxes used for the transport of goods from one destination to another. Compared to conventional bulk unit, the use of containers has several advantages, namely less product packaging, less damaging and higher productivity. An increasing number of container shipments leads to higher demands on the seaport container terminals, container logistics, and management, as well as on technical equipment [4]. Therefore, container transport requires almost perfect coordination between all participants, working resources and processes within the entire container transport system [5]. Hence, the largest seaport container terminals have to be adapted to new changes and technologies. Transit time, reliability of delivery times and cost are the main factors influencing mode choice in the container transport sector [6]. There are many logistical problems (berth allocation, stowage planning, crane split, transport optimization etc.) within container terminals that need to be optimized and mutually synchronized in order to improve the complete container transport. If only one process is not synchronized with other processes within the container terminal, this could lead to a delay in the transfer of the container between the two transport means as well as in an increase of the transport cost of a single container. Considering the business activity of the container terminal, the main interest is to provide the best service to shipping companies in order to reduce the waiting time of transfer means (vessels, trains and trucks) when loading/unloading containers and thus to increase the total container traffic within the terminal. According to [6], the main optimization problems at the intermodal container terminal are: stacking containers in the stacking area (the part of a storage yard), allocating resources in the terminal and scheduling transport mean (especially vessel) loading and unloading operations. Today, an effective and efficient use of information technology as well as appropriate optimization methods play a key role when resolving these optimization problems [4]. The optimization of the container transfer within a container terminal is a non-deterministic polynomial hard (NP-hard) problem. If a problem is NP-hard, it means that the best solution for certain problem cannot be determined as the size of this problem increases [6]. Hence, it is impossible to determine an optimized solution of the transfer within the container terminal manually. Therefore, the procedure of resolving NP-hard problems in order to get a sufficiently good solution must be automated. Thus, the metaheuristics can be used. In [7], the authors have defined the term metaheuristic in computer science and mathematical optimization:

> *In computer science and mathematical optimization, a metaheuristic is a higher-level procedure or heuristic designed to find, generate, or select a heuristic (partial search algorithm) that may provide a sufficiently good solution to an optimization problem, especially with incomplete or imperfect information or limited computation capacity.*

Hence, there is a large number of research papers that have proposed the solutions of optimizing the terminal logistic problems with metaheuristics. The majority of these researches are based on the well-known metaheuristics inspired by nature [6, 8-29], like genetic algorithm (inspired by the process of natural selection) [30], ant colony optimization algorithm (inspired by the behaviour of ants seeking a path between their colony and a source of food) [31] and particle swarm optimization (inspired by the social behaviour of bird flocks) [32]. Nature inspired metaheuristics mimic different natural systems and processes using mathematical models and algorithms [33]. Using these nature inspired algorithms, very good results of optimizing some logistic problems within the container terminal are obtained. While genetic algorithm, ant colony optimization algorithm, and particle swarm optimization are well-known optimization metaheuristics, there is a continuous emergence of new nature inspired metaheuristics that could be used in the optimization of container terminal problems like artificial bee colony algorithm (based on the intelligent behaviour of honey bee swarm) [34], firefly algorithm (inspired by the flashing behaviour of fireflies) [35, 36], bat algorithm (based on the echolocation which is important to hunt prey as well as to perceive distances and the environment) [37] etc. According to the results presented in [38-44] in which the old (genetic algorithm, ant colony algorithm and particle swarm algorithm) and the new (bee, firefly and bat algorithm) nature inspired metaheuristics are compared, the new metaheuristics inspired by nature outperform the old algorithms in solving various optimization problems. Considering these results of comparing algorithms, we assume that better results of optimization of terminal logistic problems can be achieved by introducing these new nature inspired metaheuristics. Therefore, artificial bee colony algorithm, firefly algorithm and bat algorithm will be presented in this paper as potential solutions for better optimization of logistic problems within a container terminal.

In this paper, we have described and classified the main subsystems of the container terminal and its logistics problems that need to be optimized. We have also presented a review of new nature inspired metaheuristics that could be used in the optimization of these logistics problems within the terminal. This paper is organized as follows: in section 2, the container terminal and the main optimization problems within the subsystems of container terminals are described; in section 3, new nature inspired metaheuristics that could be used to resolve the optimization of logistic problems are presented; and, finally, the conclusion is given in section 4.

2 Optimization of Logistic Problems within the Seaport Container Terminal

In general terms, a seaport container terminal can be described as an open system of material flow with two external interfaces of which one is the quayside where containers are loaded and unloaded on/off vessel, and the other is the landside (hinterland) where the same process is done but there are trucks and trains instead of vessels on/off which the containers are loaded or unloaded [4]. On the transfer from the quayside to the landside, the containers are placed in a storage yard thus facilitating the whole transfer process of containers. In Figure 1, operation areas of a seaport container terminal and flow of container transports can be seen.

When arriving at the port, a container vessel is assigned to a berth where the containers are unloaded by quay cranes. Next, the containers are transferred from quayside to the terminal yard (usually to the stacking area) by special vehicles (terminal tractor with trailer, multi-trailer, shuttle carrier, automated guided vehicles etc.). The containers are stacked side by side and on the top of each other forming blocks. The blocks are served by stacking cranes like rail mounted gantry cranes, rubber-tiered gantries, overhead bridge cranes etc. Each contain-

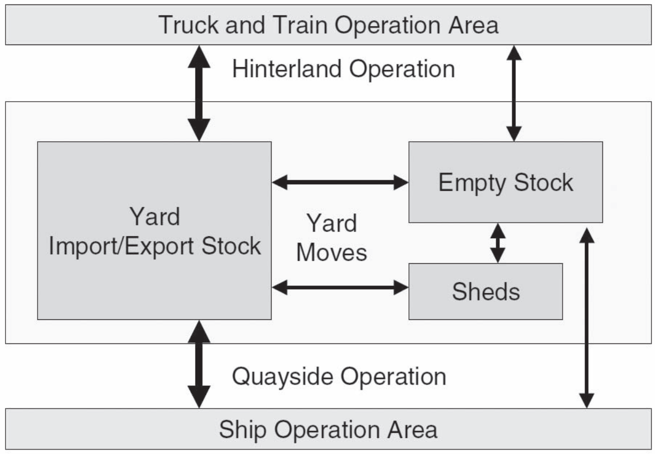

Figure 1 Operation Areas of a Seaport Container Terminal and Flow of Container Transports

Source: [4]

er in the stacking area can be stored for a certain period waiting for the transport means (truck or train) in which it has to be loaded. The transfer from the storage yard to the particular transport means is made by special terminal vehicles like straddle carriers. This process can also be performed in reverse order, to unload containers off trucks and train, transfer through the container terminal and load on board a vessel. According to [4], the stacking blocks, vessels, trains, and trucks belong to the category 'stock'. Each stock is defined as a source that can store containers. There is no difference between these various stocks and the only difference is in capacity and complexity. The usage of these stocks has to be optimally used in order to decrease the container transfer time through the terminal and the cost of this container transfer as well as to maximally and fully utilize the usage of seaport container terminal resources.

Therefore, there is a need for the optimization of the whole container transfer process within the terminal. The logistics of container terminals, especially large terminals, has become very complex and logistics experts cannot manually adjust the operations of terminal processes that will optimize the usage of resources. Hence, to achieve the further improvements of terminal logistics, there is a need to introduce scientific methods like optimization algorithms that will enable better utilization of the terminal resources in an automated way.

Furthermore, the transfer of containers through the terminal is dynamic and demands online optimization. There are various problems that can interfere the optimized schedule of operations within the container terminal like damaged container, container documentation delay, container delay, technical problems of a crane or a vehicle, a modification in the plan of loading containers on board the vessel, bad weather conditions (e.g. strong wind) etc. Therefore, if the optimized schedule of operations is interfered for just one process within the terminal, the optimization algorithm shall be capable to adapt quickly to the new situation and find the optimal solution for that situation minimizing a delay time and the unexpected expenses. As stated earlier, all processes of a container transfer within the container terminal need to be optimized and mutually synchronized. If only one process is not synchronized with the others, the container transfer through the terminal may be delayed causing an increase of transport cost.

In this section, we have presented specific problems within the container terminal that should be optimized for a better utilization of seaport resources and also for the reduction of the transit time of each container through the container terminal.

2.1 The Ship Planning Problem

The ship planning problem can be divided into three main logistic sub problems: the berth allocation, the stowage planning and the crane split. This is a very complex logistic problem and there is a great challenge how to synchronize all these sub problems to optimize the whole logistic problem.

2.1.1 Berth Allocation

Before entering a port, each vessel must have an assigned berth on which it will stay during the container loading/unloading process. Several objectives of optimized berth allocation exist: allocating the berth for a certain vessel as close as possible to the stacking area where the containers for loading are located, capability to utilize all berths by allocating them at the same time in order to maximally use the complete berth facilities as well as to minimize the waiting times of all vessels before each vessel berthing and capability to minimize the berthing time of all vessels using the optimal assignment of seaport resources. Minimizing the waiting and berthing times affects the reduction of the complete time of a vessel stay in the port which is one of the greatest challenges in optimizing logistic problems of the container terminal. It may occur that one vessel is delayed so, due to this unscheduled situation, the whole plan of allocating berths has to be changed in order to establish a new optimal plan of berth allocation. Hence, there is a need for an automatic and optimized berth allocation plan which can be adjusted to new changes because creating a new modified optimized berth allocation plan is a very complex problem and cannot be made manually by an expert, especially when creating a plan for one of the world largest seaports where a large number of vessels arrives every day.

According to [45-49], the berth allocation problem is known to be NP-hard problem when a number of vessels, for which an allocation berth has to be done, is growing and the exact methods cannot find a satisfactory solution. Therefore, as stated earlier, to resolve a NP-hard problem, metaheuristics can be used. The well-known metaheuristics inspired by nature, like genetic algorithm [30], ant colony optimization algorithm [31] and particle swarm optimization [32] have been successfully used in solving this problem for a large number of vessels. The research in [8-13] has used these metaheuristics to solve different models of berth allocation problem automatically. As the well-known metaheuristics inspired by the nature are successfully used in the optimization, we assume that the new metaheuristics, like bee, firefly or bat algorithm, should be used for berth allocation optimization in order to improve the optimization results according to the excellent performances of these new metaheuristics obtained in [38-44]. These metaheuristics will be presented in Section 3 in details.

2.1.2 Stowage Planning

Stowage planning is the most important logistic problem of ship planning. Stowage planning is divided in two-step processes. First, the shipping line operator creates a stowage plan of the vessel for all seaports in which the vessel is staying. In the stowage plan, the position of every transported container has to be defined. Usually, a stowage position of each container is defined according to the category (type, length or weight of container, discharge port etc.) to which it belongs. Then, each container of a certain category has a defined stow position inside the part of stowage where its category loads. Considering the shipping line interest of minimizing the time of loading/unloading process, the optimization objective of stowage planning is minimization of shifts during loading/unloading containers. The containers are stacked on the top of each other and the stowage plan need to be created in a way that, when unloading containers, there is a minimal number of unnecessary reshuffles of containers that have to be moved in order to get and unload certain container. Thus, the loading/unloading process is faster and the vessel port time in a seaport is reduced. Furthermore, it is necessary that, when creating a stowage plan, the strength and stability of the vessel must be satisfied. Therefore, stowage planning is a very complex problem which consists of optimizing the stacking of containers to minimize the port time when a vessel stays at a port while, at the same time, satisfying the vessel parameters such as stability.

After an employee of the shipping company has created a stowage plan, this plan is sent to the terminal operator. This plan serves as the starting point of the terminal ship planner. There are various objectives of optimization here but the most important ones are the stacking reshuffles (similar to the vessel stowage reshuffles) and distance between the assigned berth of the vessel and the position of containers within the stacking area. The smaller the distance between containers and vessel is, the faster is the loading process of these containers. The similar process is when the unloading is done. Considering the unnecessary reshuffles, the containers have to be stacked in the stacking area in a way that on the top of the container blocks are those containers that must be loaded first. Otherwise, the number of reshuffles increases as well as the cost and the waiting time. As stated earlier, all processes within the terminal that have to be optimized are interconnected. Hence, without an optimized stacking of the container in the stacking area, an optimized container loading on board the vessel cannot be done. Optimization of stacking area will be described in detail later in section 3.2.

According to the research in [50], the stowage planning is NP-hard problem therefore using a metaheuristics to resolve such a problem is welcome in the stowage planning logistic problem. The authors in [14, 15] have used the genetic algorithm (nature inspired metaheuristic method) to generate stowage plan automatically. The initial computational results have shown effective sub-optimal solutions of this problem.

2.1.3 Crane Split

The crane split is the allocation of quay cranes to individual vessels that are in the seaport. This logistic problem is very important because if small number of quay cranes is allocated for a certain vessel, the loading/unloading process may be slow and the port time can be increased. On the contrary, allocating too many quay cranes to certain vessel can reduce maximal utilization of these cranes and there may be a delay during the loading/unloading process of a vessel for which a small number of cranes is allocated. It can be seen that inaccurate crane split affects the berth allocation problem too (longer vessel port time causes longer operations of a particular berth causing the delay of other vessels berthing), therefore these two logistic problems are strongly related. Thus, the objective for optimization is the allocation of an optimal number of quay cranes for serving every vessel in the seaport minimizing the vessels' stays in the port and maximizing the economic utilization of each crane. The problem of quay cranes allocation is called quay crane assignment problem (QCAP). Beside the QCAP, a scheduled plan of operations has to be determined for each assigned quay crane. This problem is called quay crane scheduling problem (QCSP). Furthermore, there are additional parameters that have to be fulfilled like safety distance between adjacent cranes which limits the movement of cranes. These additional parameters for operations and cranes lead to a variety of different models for crane split scheduling. The authors in [51] have proven that the crane split is NP-hard problem. The manual method for solving crane split problem has to be replaced with a method which determines a good solution automatically. Using metaheuristics, the resolving of the crane split problem can be automated. The authors in [16-19] have used genetic algorithm for solving crane split problems. Besides the genetic algorithm, which is the most popular metaheuristic inspired by nature, there are two research papers [20, 21] in which the authors have used particle swarm optimization algorithm inspired by nature to resolve this crane split problem. As the well-known metaheuristics inspired by nature have been successfully used in the crane split optimization, we assume that the usage of new nature inspired metaheuristics, like bee, firefly or bat algorithm, in the optimization of the crane split problems could improve the optimization results. Therefore, in future researches, it will be interesting to introduce these new algorithms in solving crane split problems.

2.2 Storage and Stacking Logistics

The optimization of the storage and stacking logistic problems is increasingly in the focus of research over the years. Considering the growth of container traffic in the world, there is an increasing number of containers that have to be stacked in seaports. Therefore, the space within the stacking area is becoming a part of the seaport which has to be optimized in order to manage as many containers as possible. According to [6], an average of 80% of the containers is transferred to the stacking area before transferring from the container terminal. As stated earlier, containers are stacked side by side and on top of each other forming blocks. Therefore, it often occurs that one or more containers block the retrieval of a particular container which has to be transferred according to the defined schedule. Hence, the containers must be stacked in a way that, when retrieving containers, there is a minimal number of additional movements of containers that stay in the stacking area. These additional unproductive movements are called reshuffles. Reshuffling consumes time and slows down the transhipment time between the stacking area and other areas of the container terminal. Furthermore, this delay causes the delay of loading/unloading vessels and also increases the port time of a vessel. Besides the optimization of additional container movements, the usage of stacking area resources needs to be optimized. Stacking crane performs a container movement. Usually, it needs to optimize the retrieving of more containers within the stacking area in a way to minimize the retrieval time, the operational cost and a number of crane operations. Hence, the operation schedule of each stacking crane, which minimizes the cost and the number of operations for a certain task, has to be defined. Minimizing the number of crane operations on one task, can result with a possibility to maximize the utility of this crane on other tasks within the stacking area.

The well-known optimization problems within the stacking area are post-stacking problems like Blocks Relocation Problem (BRP), the ReMarshalling Problem (RMP) and the PreMarshalling Problem (PMP) [52]. In RMP and PMP the containers are reshuffled within the bay to minimize the number of future unproductive moves. The number of containers in the bay remains constant while performing RMP and PMP. The BRP problem consists of reshuffles and retrieval of containers from a stacking area to a destination vessel. Thus, retrieving and reshuffling operations might be carried out in parallel for the BRP. All the containers have to be retrieved with a minimal number of reshuffles.

Since these optimization problems described above are NP-hard, it is difficult to optimize the container scheduling problems [6]. Thus, the metaheuristics should be used in order to resolve these problems. The genetic algorithm, which is the metaheuristic inspired by nature, has been used in [6, 22-26, 53] to resolve the various stacking problems. Considering these researches listed above, we assume that it would be good to introduce new nature inspired metaheuristics in order to improve the results obtained by using genetic algorithm.

2.3 Transport Optimization

At a container terminal, a transport can be divided into the horizontal and the vertical transport (stacking cranes). Furthermore, the horizontal transport consists of the quayside and the landside transport. The quayside transport includes a transport between the stacking area and the vessels, while the landside transport includes a transport between the stacking area and the trucks or trains and vice versa. There are different transport vehicles that are used for the horizontal transport like trucks, multi-trailers, manned or automated straddle carriers, automated guided vehicles (AGVs) etc.

When loading/unloading containers on board/off board the vessel, containers have to be transported from/to the stacking area. The quayside transport problem consists of synchronization and optimization related to quay cranes, transport vehicles and stacking cranes. Optimization of container transfer between the quay and the stacking area takes into account the synchronization with the quay and the stacking cranes. Therefore, these additional requirements make the optimization problem difficult. There are two primary modes of transport at the quayside: the single-cycle mode, where the vehicles serve only one quay crane and the dual-cycle mode, where the vehicles serve multiple quay cranes simultaneously. In the single-cycle mode, it is hard to optimize the container transport. Eventually, the distance between the quay cranes and the containers that have to be transferred could be reduced by stacking the containers within the stacking area as close as possible to the quay crane, but this optimization problem does not belong to the transport vehicles optimization but to the storage and stacking logistics described in Section 3.2. In the dual-cycle mode, there is a possibility to optimize the problem of quayside transport by creating an optimized transport schedule for each transport vehicle in order to minimize the transfer time and the empty distance for the vehicle in order to increase the vehicle productivity. The landside transport is similar to the quayside transport, but instead of vessels, there are trucks or trains. Considering the optimization problem when the containers are loaded from the stacking area to the trains, it is the similar procedure as is loading containers on board the vessels. Here, the most important thing is increasing the productivity of transport vehicles (which operate between the stacking area and the railhead area) by defining the optimized work schedule for every vehicle in order to minimize the transfer time and the empty distance of the vehicles. Furthermore, the waiting times of cranes (in stacking area and railhead area) have to be reduced as much as possible. It is easy to see that the vertical transport (stacking cranes) is related to the horizontal transport and these transports have to be synchronized in order to achieve high-quality optimization of logistic problems within these transports. The authors in [27-29] have used the genetic algorithm for resolving specific problems within the horizontal and vertical transport. We assume that the introduction of the new nature inspired metaheuristics could also improve this optimization problem considering the performances of these metaheuristics. In the next section, these new metaheuristics are presented.

3 New Metaheuristics Inspired by Nature

As stated earlier, the optimization of the container transfer within the container terminal is a NP-hard problem which cannot be resolved manually in an optimizing way. Therefore, a metaheuristic, which is a higher-level procedure that may provide a sufficiently good solution to an optimization problem, can be used to resolve the optimization problem of the transfer within the container terminal. In Section 2 the logistic problems within the container terminal, which have to be optimized, have been presented. Moreover, there is a large number of research papers that have proposed very good solutions of optimizing the container terminal logistic problems with well-known metaheuristics inspired by nature, like genetic algorithm, ant colony optimization algorithm and particle swarm optimization. Nowadays, there is a continuous emergence of new metaheuristics inspired by nature. In this paper, artificial bee colony (ABC) algorithm [34], firefly algorithm [35, 36] and bat algorithm [37] will be presented in details as the potential solution for resolving the optimization problems at the seaport container terminal. We assume that these three metaheuristics could improve the optimization of the terminal logistic problems because these metaheuristics show better optimization results on well-known optimization problems than the old nature inspired metaheuristics (genetic algorithm, ant colony algorithm and particle swarm algorithm), in view of the research in [38-44].

In [38], the ABC algorithm is compared with the genetic algorithm and the particle swarm algorithm. These algorithms are used for optimizing multivariable functions (Griewank, Rastrigin, Rosenbrock, Ackley and Schwefel functions). The results showed that ABC outperforms the other algorithms.

Bat algorithm, genetic algorithm and particle swarm algorithm have been compared in [39] on the problem of neural network training which is a complex task of great importance in the supervised learning field of research. According to the results of neural network training, bat algorithm shows better time performance and better quality of solution than the other two algorithms.

Travels Salesman Problem (TSP) is a well-known NP-hard problem in which a salesman wants to visit all cities and return to the start city by minimizing travel time. In [40], the author has compared the particle swarm optimization algorithm, ABC algorithm, bat algorithm and ant colony optimization algorithm given the produced results when solving a TSP problem. Bat algorithm outperforms the other algorithms. ABC algorithm produces similar results as the well-known particle swarm optimization algorithm while the well-known ant colony optimization algorithm produces the worst results. In [44], the authors have proposed an algorithm which is the combination of the ABC and genetic algorithm. This proposed algorithm outperforms the 'ordinary' genetic algorithm in resolving a TSP problem.

Furthermore, the genetic algorithm and the ABC algorithm are compared according to the results performed on the Capacitated Vehicle Routing Problem (CVRP) [41] and the Job Shop Scheduling problem (JSSP) [42]. The CVRP problem is an optimization problem of the goods distribution to different customers by vehicles of the same capacity. The goal is to minimize the total cost of distribution. JSSP problem is an optimization problem in which ideal jobs are assigned to resources minimizing the makespan. The results have shown that the ABC algorithm provides better results for the CVRP and JSSP problems than the genetic algorithm.

In [43], the firefly algorithm has been compared with the genetic algorithm, ant colony optimization algorithm and particle swarm optimization on the TSP problem. The results have shown that the firefly algorithm outperforms other three algorithms in resolving the TSP problem.

This indicates that, considering the very good results of optimizing the terminal logistic problems by using old metaheuristics, there is a high probability that new metaheuristics will also behave well when using these metaheuristics in the optimization of the terminal problems as these new metaheuristics outperform the old metaheuristics in resolving different well-known optimization problems. Therefore, we will present the ABC algorithm, firefly algorithm and bat algorithm in this section as the potential solutions for the optimization of the container terminal logistic problems.

3.1 The Artificial Bee Colony Algorithm

The artificial bee colony (ABC) algorithm [34] is the most popular metaheuristic inspired by the behaviour of bees. There are many customized versions of this algorithm that solve different continuous, discrete and combinatorial problems. The ABC algorithm is based on the bees' foraging. This process consists of food sources, scout bees, onlooker bees and employed bees.

The bees evaluate the interest of particular food source according to the many criteria like distance from the hive, quantity of food etc. It is important to reduce the energy cost of delivering as much food as possible to the hive.

$\mathcal{N}$ food sources are chosen
while *the stop criterion is not met* **do**
  Employed bees go out of the hive to exploit food sources
  Onlooker bees go out of the hive and split based on the interest of food sources to exploit them
  Employed bees eventually abandon some food sources
  Scout bees eventually search for new food sources
end

Figure 2 Main Steps of the ABC Algorithm

Source [54]

This is an optimization task. Thus, the bee colony always exploits the best food source at the moment. When another food source seems to be better, the bee colony adapts and starts to exploit this food source. An employed bee exploits some food source and brings food to the hive. After storing this food, the employed bee decides if it will dance on the dance floor to provide the information about the 'interest' of the food source. The dance floor is an integral part of each hive and it attracts onlooker bees (searching for a good food source) that are not exploiting any food source currently. If the employed bee considers that the exploited food source is of interest for the colony, it dances on the floor to give the information about the food source. After the dance of the employed bee, onlooker bees that are interested in exploiting this food source, use provided information and may go to the food source. The more interesting the food source is, the more onlooker bees leave the hive and go to this food source. The information about good food sources provides minimizing (optimizing) the energy cost of bee colony while searching for food. If the employed bee considers that the exploited food source is no more interesting for the colony, it does not dance on the floor. It may become an onlooker bee or scout bee. The scout bee tries to discover a new interesting food source. If it discovers an interesting food source, it becomes an employed bee, exploits the food source and returns to the hive in order to attract other onlooker bees with its dancing on the dance floor.

In Figure 2, the main steps of the artificial bee colony algorithm can be seen. The food sources are actually individual areas of the entire solution space of the particular optimization problem. There are *N* food sources defined randomly at the beginning of the algorithm. The more interesting the food source, the better the solution found in this food source. Therefore, there is a higher probability that the onlooker bee will choose the most interesting food source, i.e. the algorithm will search for a better solution in the area of a solution space with the best solution found so far. Hence, more onlooker bees that become employed bees will exploit this food source i.e. the algorithm will search for solutions much more in this area of solution space.

To fully exploit the food source, the employed bees look for the food in the entire neighbourhood of the detected food source. Considering the ABC algorithm, the initially found solution in some area of the solution space has to be mutated in order to discover solutions in the neighbourhood of this initial solution. Each solution consists of fragments where each fragment holds some value that determines this solution. When mutating the solution, the algorithm changes the value only of one fragment in order to get a completely new solution which is in the solution space neighbourhood of the initial mutated solution. In this way, the similar solutions are compared in order to find the best solution among this area of solution space (food source).

When there is no more food in the particular food source i.e. no more better solutions are found in the neighbourhood for a certain period of searching, the employed bees return to the hive and become onlooker bees (wait for some more interesting food area) or scout bees (search randomly for some new food source). Scout bees which randomly discover new food sources, provide the possibility of discovering better food sources (solution areas) in order to minimize the energy cost of collecting food i.e. the procedure that randomly discovers better solution within the solution space, finds new solutions in the solution space of a particular optimization problem that can be better than the solutions found before. Thus, when one food source is fully exploited (i.e. there is no more better solution for the optimization problem), this food source (area of solution space) is changed with new food source (area of the solution space) discovered randomly by some scout bee.

The algorithm always remembers the best solution obtained after each iteration. The algorithm ends when the number of iterations is finished or a sufficiently good solution is found.

3.2 The Firefly Algorithm

The firefly algorithm [35, 36] is based on the ability of fireflies to emit light in order to attract a mate. Attracting a mate by emitting light depends on the intensity of the light and the distance between potential mates. Considering two fireflies at a short distance, the one firefly emits light with weaker intensity moves toward the firefly which emits light with stronger intensity. Thus, a firefly with the

Objective function $f(\boldsymbol{x})$, $\boldsymbol{x} = (x_1, ..., x_d)^T$
Generate initial population of fireflies $\boldsymbol{x}_i$ $(i = 1, 2, ..., n)$
Light intensity I_i *at* $\boldsymbol{x}_i$ *is determined by* $f(\boldsymbol{x}_i)$
Define light absorption coefficient γ
while *(t <MaxGeneration)*
for $i = 1 : n$ *all n fireflies*
  for $j = 1 : n$ *all n fireflies (inner loop)*
  if $(I_i < I_j)$, *Move firefly i towards j;* **end if**
    Vary attractiveness with distance r via $\exp[-\gamma r]$
    Evaluate new solutions and update light intensity
  end for j
end for i
Rank the fireflies and find the current global best $\boldsymbol{g}_*$
end while
Postprocess results and visualization

Figure 3 Pseudo Code of the Firefly Algorithm

Source [36]

strongest light is the most attractive mate. As stated above, the distance between two fireflies also plays an important role in attracting potential mates. The smaller the distance between two fireflies is, the greater is the possibility of attraction between these fireflies. In the firefly algorithm, fireflies are moving in the solution space. The current position of each firefly represent a solution (determined by the objective function) of the optimization problem. The better the solution, the stronger the light intensity that firefly emits. Thus, a firefly with a worse solution moves toward the firefly with a better solution. In this way, a solution space is searching in order to find the best possible solution of the optimization problem. In Figure 3 the pseudo code of the firefly algorithm is presented.

First, the objective function has to be defined. Next, the algorithm generates the initial population of fireflies (solutions). The fireflies are randomly distributed within the space area, i.e. the solutions are randomly defined within the solution space. For each firefly, the light intensity is determined, i.e. for each randomly defined solution, the quality of solution is calculated with the objective function. Considering the light intensity of the fireflies, a light absorption coefficient has to be defined in order to setup the rule of convergence between the fireflies i.e. the solutions. As stated above, besides the light intensity, the distance between fireflies also affects the convergence between the fireflies. In each iteration, each solution (firefly) is compared with all defined solutions (fireflies) according the value of the objective function (light intensity). If the light intensity of the currently comparable firefly is less strong than the light intensity of a certain firefly in the solution space (i.e. if the value of the currently comparable solution is smaller than the value of a certain defined solution), than this firefly with a less strong light intensity moves towards the firefly with a stronger light intensity (i.e. the solution moves toward the better solution). The movement of the solution (firefly) within the solution space, actually represents the change of the coordinates of this solution i.e. it represents a new solution with the coordinates that are closer to the coordinates of solution with better goodness of the objective function. This change of the coordinates is defined by the light absorption coefficient and the distance between the solutions. In this way, the algorithm searches for better solution within the solution space. If a solution has the highest value of the objective function (i.e. if a firefly has the strongest light intensity), the algorithm changes randomly its coordinates in order to find some new solution with a better value of the objective function (i.e. the firefly moves randomly in order to find better mate). The best result obtained in each iteration is saved within the algorithm. The algorithm is stopped when the maximum number of iterations is reached or the goodness of the solution is good enough to solve requested optimization problem.

3.3 The Bat Algorithm

The bat algorithm [37] is based on the echolocation behaviour of bats. Using the echolocation, a bat moves (flies) in the complete darkness, detects the prey and avoids obstacles on its way. After emitting a very loud sound pulse, the bat listens to the echo that bounces back and creates a 'picture' of their environment and recognizes the prey within this environment. When the bat detects the prey, the pulse rate accelerates in order to determine the prey as precise as possible. The algorithm for resolving an optimization problem is based on this hunting strategy described above. The algorithm is searching for the good solution within the solution space. When the algorithm detects a good solution, then it starts to search the neighbourhood of this solution in order to find an even better solution of the defined optimization problem. In this way, a solution space is searched in order to find the best possible solution of the optimization problem. In Figure 4 the pseudo code of the bat algorithm is presented.

First, the objective function that solves the optimization problem has to be defined. Next, the algorithm randomly generates the initial bat population. Each bat represents a solution within the solution space. At the beginning, the velocity of each bat is set to 0, i.e. the bats start to fly in the first iteration. Moreover, the frequency of each bat has to be defined. The frequency affects the velocity of the bat. Higher the frequency, greater the velocity of the bat i.e. the bat moves faster through the air in order to detect the prey. A greater velocity means that the algorithm searches for new solutions within the solution space much farther away than the current solution of the bat is. Furthermore, the loudness and the pulse rate have to be defined for each bat. At the beginning, the loudness of each bat is high because there is no good solution found so far. Therefore, searching for the good solution is performed on a large area of the solution space i.e. the bat has not found the prey so far and therefore it emits a very loud sound pulse in order to detect the prey in a large area. Unlike the loudness, the pulse rate is very low at the beginning because the bat just starts to detect

```
Objective function f(x),   x = (x_1, ..., x_d)^T
Initialize the bat population x_i (i = 1, 2, ..., n) and v_i
Define pulse frequency f_i at x_i
Initialize pulse rates r_i and the loudness A_i
while (t <Max number of iterations)
Generate new solutions by adjusting frequency,
and updating velocities and locations/solutions
    if  (rand > r_i)
    Select a solution among the best solutions
    Generate a local solution around the selected best solution
    end if
    Generate a new solution by flying randomly
    if  (rand < A_i  &  f(x_i) < f(x_*))
    Accept the new solutions
    Increase r_i and reduce A_i
    end if
Rank the bats and find the current best x_*
end while
Postprocess results and visualization
```

Figure 4 Pseudo Code of the Bat Algorithm

Source [37]

the prey i.e. the algorithm starts to find solutions within the solution space. In this moment, there is no good solution found and therefore the algorithm randomly searches for a solution within the solution space. In each iteration, every bat is flying randomly to detect a prey (i.e. the algorithm is trying to find better solutions by randomly changing the parameters of each existing solution) or is flying in the direction of the bat that has detected the best possible prey so far (i.e. the algorithm is trying to find better solutions by changing the parameters of each existing solution to make the existing solutions closer to the best solution found so far within the solution space). When a pulse rate of a bat is low (no prey in sight i.e. no good solution found by this bat within the algorithm), there is a greater possibility that the bat is flying towards the bat that has detected the best prey so far. On the contrary, when a pulse rate of a bat is high (a prey is in sight i.e. a good solution has been already found by this bat within the algorithm), there is a greater possibility that the bat is flying around its prey (searching for the better solution within the solution space in the neighbourhood of already found solution). When a bat detects a better prey than detected before (algorithm finds better solution), then the loudness is reducing and the pulse rate is increasing. These parameters ensure the future searching of new solutions around this good solution found by this bat. The algorithm is stopped when the maximum number of iterations is reached or the goodness of the solution is good enough to solve the requested optimization problem.

4 Conclusion

As stated at the beginning, a maritime trade is the backbone of the global trade of goods. Therefore, seaports have become the most important intersection of global trade. A container transport plays an essential role in the global trade. Given the increasing number of container shipments in the world, there is a need to improve the entire container transport system. Thus, the container transport requires almost perfect coordination between all participants, working resources and processes. As a part of the container transport system, the seaport container terminals have to be adapted to this increased number of container shipments by maximally utilising the seaport resources in order to optimize the terminal logistic problems as well as to speed up the container transfer through the terminal. There are many logistic problems (berth allocation, stowage planning, crane split, transport optimization etc.) within the container terminal that need to be optimized and mutually synchronized to improve the container transfer through the terminal. The logistics of container terminals have become increasingly complex and therefore experts can no longer manually optimize the usage of terminal resources. Hence, there is a need to introduce scientific methods like optimization algorithms that will enable better utilization of the terminal resources in an automated way. A large number of research papers propose the solutions for optimizing the container terminal logistic problems with well-known metaheuristics inspired by nature like genetic algorithm, ant colony optimization algorithm and particle swarm optimization. Using these nature inspired algorithms has led to very good results in the optimization of terminal problems. There is a large number of new nature inspired metaheuristics that could be used in the optimization of the container terminal logistic problems. Artificial bee colony algorithm, firefly algorithm and bat algorithm have been presented in this paper. These three new metaheuristics outperform the old metaheuristics in solving well-known optimization problems like travel salesman problem. Therefore, we assume that better results of optimization of container terminal problems can be achieved by introducing these new nature inspired metaheuristics. In the future, we are going to introduce these new metaheuristics and test them while resolving logistic problems within the container terminal.

References

[1] Imo.org. (2017). International Maritime Organization. [online] Available at: http://www.imo.org/en/Pages/Default.aspx [Accessed 12 Dec. 2017].

[2] Unctad.org. (2017). UNCTAD (United Nations Conference on Trade and Development) - Review of Maritime Transport 2017. [online] Available at: http://unctad.org/en/PublicationsLibrary/rmt2017_en.pdf [Accessed 22 Dec. 2017].

[3] Agerschou, H. (2009). Planning and design of ports and marine terminals. London: Thomas Telford Ltd. p. 27.

[4] Voß, S., Stahlbock, R. and Steenken, D. (2004). Container terminal operation and operations research - a classification and literature review. OR Spectrum, 26(1), pp. 3-49.

[5] Dundović, Č. and Hess, S. (2007). Unutarnji transport i skladištenje. Rijeka: Pomorski fakultet, p. 72.

[6] Kozan, E. and Preston, P. (1999). Genetic algorithms to schedule container transfers at multimodal terminals. International Transactions in Operational Research, 6(3), pp. 311-329.

[7] Bianchi, L., Dorigo, M., Gambardella, L. and Gutjahr, W. (2008). A survey on metaheuristics for stochastic combinatorial optimization. Natural Computing, 8(2), pp. 239-287.

[8] Ting, C., Wu, K. and Chou, H. (2014). Particle swarm optimization algorithm for the berth allocation problem. Expert Systems with Applications, 41(4), pp. 1543-1550.

[9] Nishimura, E., Imai, A. and Papadimitriou, S. (2001). Berth allocation planning in the public berth system by genetic algorithms. European Journal of Operational Research, 131(2), pp. 282-292.

[10] Imai, A., Nishimura, E. and Papadimitriou, S. (2001). The dynamic berth allocation problem for a container port. Transportation Research Part B: Methodological, 35(4), pp. 401-417.

[11] Imai, A., Nishimura, E., Hattori, M. and Papadimitriou, S. (2007). Berth allocation at indented berths for mega-containerships. European Journal of Operational Research, 179(2), pp. 579-593.

[12] Tong C.J., Lau H.C. and Lim A. (1999). Ant Colony Optimization for the Ship Berthing Problem. In: P.S. Thiagarajan and R. Yap ed. Lecture Notes in Computer Science, 1742. Berlin: Springer, pp. 359-370.

[13] Cheong C.Y. and Tan K.C. (2008). A Multi-Objective Multi-Colony Ant Algorithm for Solving the Berth Allocation Problem. In: Y. Liu, A. Sun, H.T. Loh, W.F. Lu and EP. Lim ed. Advances of Computational Intelligence in Industrial Systems. Studies in Computational Intelligence, 116. Springer, Berlin, Heidelberg, pp. 333–350.

[14] Wilson, I., Roach, P. and Ware, J. (2001). Container stowage pre-planning: using search to generate solutions, a case study. Knowledge-Based Systems, 14(3-4), pp. 137-145.

[15] Roach P. A, Wilson I. D. (2002). A genetic algorithm approach to strategic stowage planning for container-ships. Working paper, University of Glamorgan, UK.

[16] Shin, K. and Lee, T. (2012). Container loading and unloading scheduling for a Mobile Harbor system: a global and local search method. Flexible Services and Manufacturing Journal, 25(4), pp. 557-575.

[17] Wang, Y., Chen, Y. and Wang, K. (2009). A case study of genetic algorithms for quay crane scheduling. Opportunities and Challenges for Next-Generation Applied Intelligence. 214, pp. 119–125.

[18] Hakam, M., Solvang, W. and Hammervoll, T. (2012). A genetic algorithm approach for quay crane scheduling with non-interference constraints at Narvik container terminal. International Journal of Logistics Research and Applications, 15(4), pp. 269-281.

[19] Diabat, A. and Theodorou, E. (2014). An Integrated Quay Crane Assignment and Scheduling Problem. Computers & Industrial Engineering, 73, pp. 115-123.

[20] Wang, S., Zheng, J., Zheng, K., Guo, J. and Liu, X. (2012). Multi Resource Scheduling Problem Based on an Improved Discrete Particle Swarm Optimization. Physics Procedia, 25, pp. 576-582.

[21] Tang, L., Zhao, J. and Liu, J. (2014). Modeling and solution of the joint quay crane and truck scheduling problem. European Journal of Operational Research, 236(3), pp. 978-990.

[22] Preston, P. and Kozan, E. (2001). An approach to determine storage locations of containers at seaport terminals. Computers & Operations Research, 28(10), pp. 983-995.

[23] Bazzazi, M., Safaei, N. and Javadian, N. (2009). A genetic algorithm to solve the storage space allocation problem in a container terminal. Computers & Industrial Engineering, 56(1), pp. 44-52.

[24] Kozan, E. and Preston, P. (2006). Mathematical modelling of container transfers and storage locations at seaport terminals. OR Spectrum, 28(4), pp. 519-537.

[25] Park, T., Kim, J. and Ryu, K. R. (2010). Iterative replanning using genetic algorithms for remarshaling in a container terminal. In: Proceedings of the 10th IASTED international Conference on Artificial Intelligence and Applications. Innsbruck: ACTA press, pp. 22-28.

[26] Gheith, M., Eltawil, A. and Harraz, N. (2015). Solving the container pre-marshalling problem using variable length genetic algorithms. Engineering Optimization, 48(4), pp. 687-705.

[27] Hartmann, S. (2004). A general framework for scheduling equipment and manpower at container terminals. OR Spectrum, 26(1), pp. 51-74.

[28] Skinner, B., Yuan, S., Huang, S., Liu, D., Cai, B., Dissanayake, G., Lau, H., Bott, A. and Pagac, D. (2013). Optimisation for job scheduling at automated container terminals using genetic algorithm. Computers & Industrial Engineering, 64(1), pp. 511-523.

[29] Jerald, J., Asokan, P., Saravanan, R. and Rani, A. (2006). Simultaneous scheduling of parts and automated guided vehicles in an FMS environment using adaptive genetic algorithm. The International Journal of Advanced Manufacturing Technology, 29(5-6), pp. 584-589.

[30] Mitchell, M. (1998). An introduction to genetic algorithms. Cambridge, MA: MIT press.

[31] Colorini, A., Dorigo, M. and Maniezzo, V. (1991). Distributed Optimization by Ant Colonies. In: the 1st European Conference on Artificial Life. Paris: Elsevier Publishing, pp. 134-142.

[32] Kennedy, J. and Eberhart, R. (1995). Particle Swarm Optimization. In: Proceedings of IEEE International Conference on Neural Networks IV. Piscataway: IEEE, pp. 1942-1948.

[33] Feng, X., Lau, F. C. and Gao, D. (2009). A new bio-inspired approach to the traveling salesman problem. In: International Conference on Complex Sciences. Shanghai: Springer, pp. 1310-1321.

[34] Karaboga, D., Gorkemli, B., Ozturk, C. and Karaboga, N. (2012). A comprehensive survey: artificial bee colony (ABC) algorithm and applications. Artificial Intelligence Review, 42(1), pp. 21-57.

[35] Yang, XS. (2010). Firefly Algorithm, Lévy Flights and Global Optimization. In: M. Bramer, R. Ellis and M. Petridis ed. Research and Development in Intelligent Systems XXVI. London: Springer, pp. 209-218.

[36] Yang, X.S. (2010). Nature-Inspired Metaheuristic Algorithms, 2nd edition. Frome: Luniver Press.

[37] Yang, X. S. (2010). A new metaheuristic bat-inspired algorithm. Studies in Computational Intelligence. 284, pp. 65-74.

[38] Karaboga, D. and Basturk, B. (2007). A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm. Journal of Global Optimization, 39(3), pp. 459-471.

[39] Khan, K. and Sahai, A. (2012). A Comparison of BA, GA, PSO, BP and LM for Training Feed forward Neural Networks in e-Learning Context. International Journal of Intelligent Systems and Applications, 4(7), pp. 23-29.

[40] Almufti, S. (2017). Using Swarm Intelligence for solving NPHard Problems. Academic Journal of Nawroz University, 6(3), pp. 46-50.

[41] Pinninghoff M., Alvarez G. and Contreras R. Bees and Genetic Algorithms: A Comparison on a Classic Problem. In: XIII Encontro Nacional de Inteligencia Artificial e Computacional, Recife: BDBComp, pp. 265-276.

[42] Muthiah A. and Rajkumar R. (2014). A Comparison of Artificial Bee Colony algorithm and Genetic Algorithm to Minimize the Makespan for Job Shop Scheduling. Procedia Engineering, 97, pp. 1745-1754.

[43] Ariyaratne, M. K. A. and Fernando, T. G. I. (2014). A Comparative Study on Nature Inspired Algorithms with Firefly Algorithm. International Journal of Engineering and Technology. 4(10), pp. 611-617.

[44] Nagpure, H. and Raja, R. (2012). RBGCA-Bee Genetic Colony Algorithm for Travelling Salesman Problem. International Journal of Computer Science and Information Technologies, 3(6), pp. 5384-5389.

[45] Lim, A. (1998). The berth planning problem. Operations Research Letters, 22(2), pp. 105-110.

[46] Hansen, P. and Oguz, C. (2003) A note on formulations of static and dynamic berth allocation problems. Les Cahiers du GERAD, 30, pp. 1-17.

[47] Imai, A., Sun, X., Nishimura, E. and Papadimitriou, S. (2005). Berth allocation in a container port: using a continuous location space approach. Transportation Research Part B: Methodological, 39(3), pp. 199-221.

[48] Cordeau, J., Laporte, G., Legato, P. and Moccia, L. (2005). Models and Tabu Search Heuristics for the Berth-Allocation Problem. Transportation Science, 39(4), pp. 526-538.

[49] Vis, I. and de Koster, R. (2003). Transhipment of containers at a container terminal: An overview. European Journal of Operational Research, 147(1), pp. 1-16.

[50] Avriel, M. and Penn, M. (1993). Exact and approximate solutions of the container ship stowage problem. Computers & Industrial Engineering, 25(1-4), pp. 271-274.

[51] Pinedo, M. (2002). Scheduling – Theory Algorithms and Systems. 3rd edition. New York: Springer, Cham.

[52] Caserta, M., Schwarze, S. and Voß, S. (2011). Container rehandling at maritime container terminals. Handbook of terminal planning. 49, pp. 247-269.

[53] Maglić L. (2016). Optimizacija raspodjele kontejnera na slagalištu lučkoga kontejnerskog terminala. Doktorska disertacija. Sveučilište u Rijeci, Pomorski fakultet u Rijeci.

[54] Aupetit S. and Slimane M. (2016). Nature Inspires New Algorithms. In: P. Siarry P ed. Metaheuristics. Cham: Springer, pp. 263-286.