Available online at www.sciencedirect.com

ScienceDirect

Transportation Research Procedia 52 (2021) 35–42

Transportation Research Procedia

www.elsevier.com/locate/procedia

23rd EURO Working Group on Transportation Meeting, EWGT 2020, 16-18 September 2020, Paphos, Cyprus

Hybrid Metaheuristic Approach to Solve the Problem of Containers Reshuffling in an Inland Terminal

Chiara Colombaroni*, Gaetano Fusco, Natalia Isaenko, Dario Molinari

University of Rome La Sapienza

Abstract

Abstract— The paper deals with the problem of minimizing the reshuffling of containers in an inland intermodal terminal. The problem is tackled according to a hybrid approach that combines a preliminary selection of heuristics and a genetic algorithm. The heuristics are used to determine the initial population for the genetic algorithm, which aims to optimize the locations of the containers to store in the yard in order to minimize the operational costs. A simulation model computes the costs related to storage and pick-up operations in the yard bay.
The proposed optimization method has been calibrated by selecting the optimal parameters of the genetic algorithm in a toy case and has been tested on a theoretical example of realistic size. Results highlighted that the use of a suitable heuristic to generate the initial population outperforms the genetic algorithm, initialized with a random solution, by 20%.

Keywords: Inland Freight Terminal Optimization; Reshuffling; Heuristics; Genetic Algorithm; Simulation

1. Introduction

Containerization is an efficient way to transport many kinds of goods, as it allows stocking and transporting them in the same container without handling them individually; it also reduces paperwork and damages to cargos and speeds up transfers between modes of transport. Many operations can be fully automated, especially at maritime ports, where standard containers are handled, and large freight volumes make capital investments profitable. Different conditions hold at inland terminals, where most operations are carried out by human operators with little help from automation, because of many non-standard intermodal transport units to handle (inland containers and swap bodies) and smaller freight volumes to shift between smaller inland vehicles (trains and trucks).

Usually, the containers are temporarily stored in the storage yard at container terminals. A combination of

* Corresponding author. Tel.: +39-06-44585-128.
E-mail address: chiara.colombaroni@uniroma1.it

10.1016/j.trpro.2021.01.006

container demand increase and storage yard capacity scarcity creates complex operational challenges for storage yard managers. Block stacking is used to minimize the surface required to store containers, at the expense of not having all containers available for direct retrieval. To retrieve one of the lower containers, other containers that are on top of it may need to be relocated first. In general, two types of container retrieval operations can be identified (Murty et al., 2005):

- a productive move: a container is moved directly from its storage location to a truck or train waiting to pick it up;
- an unproductive or reshuffling move: when it is necessary to pick up a container to retrieve another one stored underneath it in the same stack.

Unproductive moves heavily affect operational costs of the inland terminal because of fuel consumption and wear of both crane tires and yard pavement, as well as because of the time spent to perform them.

An extensive literature has been produced in the last two decades on container relocations at maritime terminals. Container relocation is an NP-hard problem, which has been tackled by introducing different heuristic algorithms (Zhang 2000; Murty et al. 2005; Wu and Ting et al., 2010; Ting and Wu, 2017). A broad review of the specific problem of container reshuffling was made by Caserta et al. (2011) and, more recently, by Tang et al. (2015).

Jovanovic et al. (2017) suggest using several heuristics, generate a large number of solutions, combine them for different stages of the algorithm, and choose the best found. They argue that simple randomization supply many solutions that do not have good properties and apply a deterministic approach that tries to generate only solutions that satisfy some desirable properties. Very few authors dealt with the handling of Intermodal Transport Units (ITU) in inland terminals, where the problem is complicated because of the different types of ITU and different types of cranes (quay cranes, reach stackers, forklifts). Carrese and Tatarelli (2011) determined a feasible sequence of stacking operations that minimize the costs associated with the storage of the ITU considering spatial and operative constraints. They proposed to solve a combined problem of container reshuffling and path optimization: a heuristic solving procedure based on a double string genetic algorithm integrated by an algorithm able to estimate the number of reshuffles.

In this paper, we adopt a hybrid approach that combines a preliminary selection of heuristics and a metaheuristic genetic algorithm to process their outcomes. The stochastic approach allows exploring a wide region of the solution space, which can be hard to investigate in case of large-scale problems. Moreover, the evolutionary approach enables a regular update of the solution as a new train to be unloaded arrives. In such a way, it is not necessary to restart a new problem from scratch, but the new instance is added to the current solution that represents the current state of the yard bay.

2. Methodology framework

The solution approach combines deterministic heuristics and a genetic algorithm that takes them as initial solutions and carries out an optimization every time a train arrives at the terminal and determines the optimal positions of the containers to unload. The genetic algorithm finds an optimal location to store the containers arrived at the terminal to minimize the number of actual and expected moves. The objective function also considers both the costs for reshuffling the blocking containers and the expected costs for the future retrievals of the containers depending on their expected departure time. The initial population generated by the algorithm is produced either randomly or by following one of three proposed heuristics:

1. Consider only the free positions in the bay that are on a stack in which no container is expected to be reshuffled in the future (that is, a stack where a container has a subsequent departure time of a container under it);
2. Consider only the free positions in the bay that are on a stack in which no containers are expected to leave before the container to position;
3. Consider only the lowest free positions;

The Genetic algorithms are stochastic heuristic methods that demonstrated to be quite effective in solving non-convex discrete optimization problems. However, they also exhibit poor convergence properties and become inefficient for large size problems. The goal of using the heuristics in the initial population is to improve the convergence of the algorithm.

The logic of the algorithm is depicted in the flow chart of Figure 1. Here the solid lines represent the physical process of the container handling problem while the dashed arrows represent the actual updating flow of the algorithm, that starts from the input data and updates the yard bay by carrying out the two processes of train loading (green lines) and unloading (red lines). It is assumed that the operations of containers loading (from the storage area to the trains)

and unloading (from the train to the storage area) do not co-occur.

Thus, the model simulates the arrival of one train at a time at the inland terminal.

The block labeled as 1 in the figure depicts the inputs of the problem, which are, more specifically:

- the layout of the container yard, described by the number of stacks, tiers, and rows;
- characteristics of the containers: length, resistance, weight, expected departure time;
- number and characteristics of cranes;
- the list of containers to retrieve from every train arrived and for every train departing.

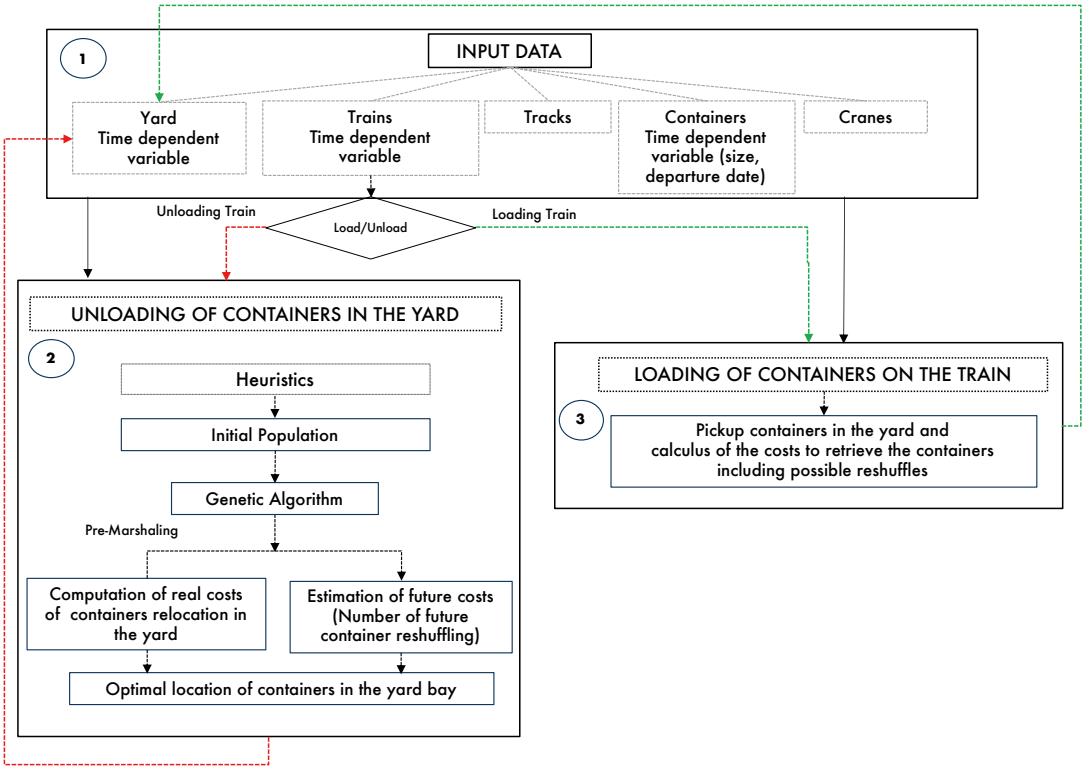

Figure 1 Flow chart of the solution method.

The block labeled as 2 in the figure describes the optimization process that is applied to every train to unload and is composed of the following steps:

1. Read the list of containers to unload from a train arrived and store in the yard;
2. The genetic algorithm generates the initial population either randomly or by following a heuristics. The element of the population is composed of the container to unload and the position where unload it;
3. For each container, identify the blocking containers, if any;
4. Apply the genetic operators based on expected departure time and height of stack;
5. Check the constraints compliance to ensure the solution feasibility;
6. Compare the different solutions and choose the one that minimizes the expected total costs for handling the containers unloaded (that is, the costs for storing them in the yard and the expected costs to pick them at the scheduled time for their departure).
7. Update the positions of the containers in the yard.
8. Go back to step 1 if a new train to unload arrives.

The block labeled as 3 describes the process of loading a departing train. Every time a train has to be loaded, the costs of the operations to retrieve the containers are computed.

At the end of each simulation step, the bay layout is updated.

3. Formulation of the problem

The problem consists of the minimization of the costs related to unproductive moves triggered by retrieving containers from a storage yard. The following hypotheses are introduced:

- all containers at the yard-bay have the same size;
- only one crane is used to handle the containers;
- the initial configuration of the container bay and the expected time to retrieve each container are known in advance.

The objective function (or fitness function in genetic algorithm jargon) entails the total operating costs associated with storage operations, i.e., the costs required to store and to reshuffle the containers, determined according to a feasible sequence of stacking operations. It is formulated as a linear combination of the total costs of the reshuffles and the total cost of crane movements:

$$C = Min\left(\sum_{u=1}^{U}\sum_{f=1}^{F}\sum_{r=1}^{R}\sum_{l=1}^{L} c_{frl}^{u} x_{frl}^{u} + \sum_{i\epsilon I}\sum_{j\epsilon J} g_{ij} x_{ij}\right) \quad (1)$$

where:

- c_{frl}^{u} represents the unit cost of one reshuffle carried out by a crane to stack the container u in the cell identified by line f, row r, and level l;
- g_{ij} represents the unit movement cost of a crane to pick up and deliver one container in the storage bay;
- x_{ij}represents the link between nodes I and J of the graph in the inland terminal;
- x_{frl}^{u} and x_{ij} are decision variables that can only take values 0 or 1, and specifically:
- $x_{frl}^{u} = 1$ if the container u is assigned to the cell identified by line f, row r, and level l;
- $x_{ij} = 1$ if the link ij if the crane task is to move from node i to node j.

The method not only computes the actual costs of stacking and retrieving a container u if a blocking container currently obstructs the assigned position, but aims at also predicting the future moves if any needed to reshuffle the container u if it blocks a container that is expected to be picked up before it. Of course, the current expected cost of containers unloading can be modified by other successive trains to unload. According to Colombaroni et al. (2019), the reshuffling costs c_{frl}^{u} are composed of four possible terms, whose computation is specified in the mentioned paper.

4. Solving procedure

A. Genome definition

The genome is a straightforward representation $\{ f, r, l \}$ of the cells in the yard bay to which the characteristics of the containers are associated. Any possible genome makeup defines an individual in the population and identifies a possible solution for the container storage problem. Any time a new train arrives and is unloaded or leaves the terminal and is loaded, the yard bay configuration is changed, and the genetic code is modified accordingly.

B. Initial population

The first step of the genetic algorithm is to generate the initial population, formed by a predefined number of possible solutions, each of which is characterized by different genome makeup. The initial population is generated by following one of three heuristics that ensure each container to be assigned to a position complying with the constraints. These heuristics are used to find an initial population that could give a better solution at the end of the optimization, by restricting the set of feasible solutions when generating it. In particular, the three heuristics work in the following way:

1. The first heuristics excludes from the feasible set all the free positions that are on a stack in which there will be future reshufflings, meaning that there is a container with a lower departure time under a container with a later departure time. For example, in figure 2a, the first two stacks have a container with a higher departure time than that of the container under it. The free positions on the top of those stacks are excluded from the set of feasible positions, and the position on top of the stack 3 is considered.
2. The second heuristics excludes all the free positions on the stacks that have at least a container with a earlier departure time than the container to position. For example, in figure 2b, since the container to unload departs on day 4, the only available position is on the top of stack 3, where all the containers depart after day 4.
3. The third heuristics considers only the lowest free positions. For example, in figure 2c, the lowest free position is on level 2 of the stack 3, which is the only feasible position.

If there are no sufficient free positions that comply with the heuristics, random free positions are added.

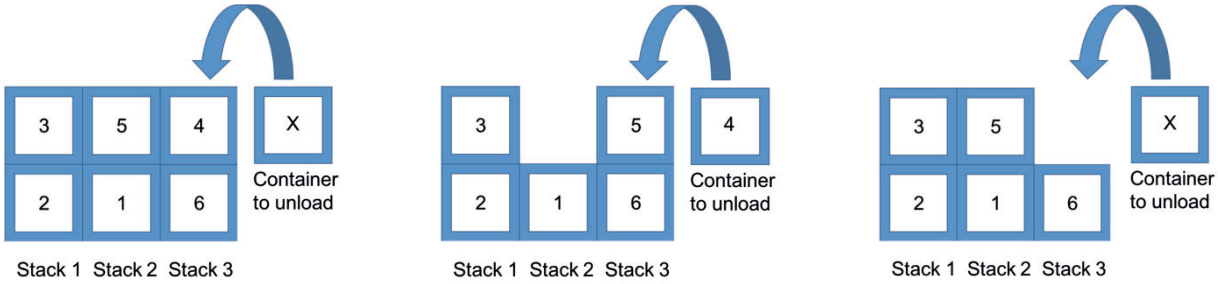

Figure 2 (a) Representation of the first heuristics. (b) Representation of the second heuristics (c) Representation of the third heuristics

C. Cross-over

The Cross-over operator combines the genetic patrimony of a pair of individuals to generate a pair of new individuals. The cross-over rule applied to generate the genome makeup of the children randomly chooses two cross-over points for each pair of parents; one child receives all genes within these points from the first parent and all genes out these cross-over points from the second parent; the second child receives the complementary genes. The parents are accepted for the cross-over with a given probability and, if accepted, the children undergo a verification aimed at avoiding unfeasible solutions.

D. Mutation

A mutation operator applies, with a given probability, random alterations to one or more genes of any individual of the population and assigns a new feasible cell in the yard bay to the corresponding container. After a given number of iterations without any improvement of the objective function, the mutation probability is increased linearly up to a maximum value. As an improvement of the objective function is obtained, the mutation probability is reset to its initial value.

E. Elitism

The elitism feature consists of preserving a fraction of the best solutions for the new generation.

F. Fitness evaluation

Fitness coincides with the objective function. In order to evaluate the fitness, the reshuffles' costs and the crane paths are computed. The reshuffles' costs are obtained by applying the function (1), subject to constraints that account for the retrieval times of the containers, and the limits on the climbing power of the cranes.

5. Application to a test case

5.1 Assumptions

The optimization procedure has been applied in a simulated inland terminal where the handling operations of containers are carried out by reach stackers. In the simulation, all containers arrive at the terminal and depart from it by rail. This assumption simplifies the implementation of the simulation without affecting the nature of the problem since it does not alter the processing of the containers, which are bunched to be loaded onto a train. It is assumed that the arrival times of all trains and the departure times of all containers are known in advance.

We tested the optimization in two different cases: a Toy-bay and a Real-life sized bay.

The inland terminal in the toy case is composed of a 3-level yard bay having a capacity of 72 containers, one rail where trains arrive and depart, and an operational area where the reach stackers move from the train to the bay and vice versa. The yard bay track is aligned with the first line of the bay, from which the reach stacker operates to stack or retrieve the containers. The yard bay has 72 possible positions that are accessible only from one side and allow containers to be stacked up to 3 levels. A fraction of 49% of the positions is initially occupied. Each train carries eight 20-feet long containers. The terminal of the real-life sized case if composed by a 3-level yard bay with a capacity of 408 containers and 50% of the bay positions occupied. Each train carries 40 20-feet long containers. Arriving and departing trains alternate at the terminal. The time interval between successive trains is long enough to allow loading

or unloading operations not to affect each other. Since the solution procedure aims at minimizing the costs for containers reshuffles at the terminal by predicting future retrievals, testing its effectiveness requires simulating the entire process of storage and retrieval of all the containers.

The tests aim at assessing the performances of the genetic algorithm with each of three heuristics and consist of 5 simulations of the entire storage-retrieval process of the bay. An initial population with all random individuals is performed for comparison. Each simulation run includes five arriving alternated with five departing trains and followed by five (six in the real case) departing trains that empty the yard bay. In the simulation, actual costs instead of predicted ones are computed.

Table 1 Values of genetic algorithm operators for the real-sized case

Values of genetic algorithm operators	
Initial Population	100 individuals
Generations	100 iterations
Elitism factor	0.1 (30%)
Cross-over factor	1 (100% probability to cross-over)
Maximum number of iterations with no improvement	10
Mutation factor min	0.2
Mutation factor max	0.8
Mutation factor step	0.1

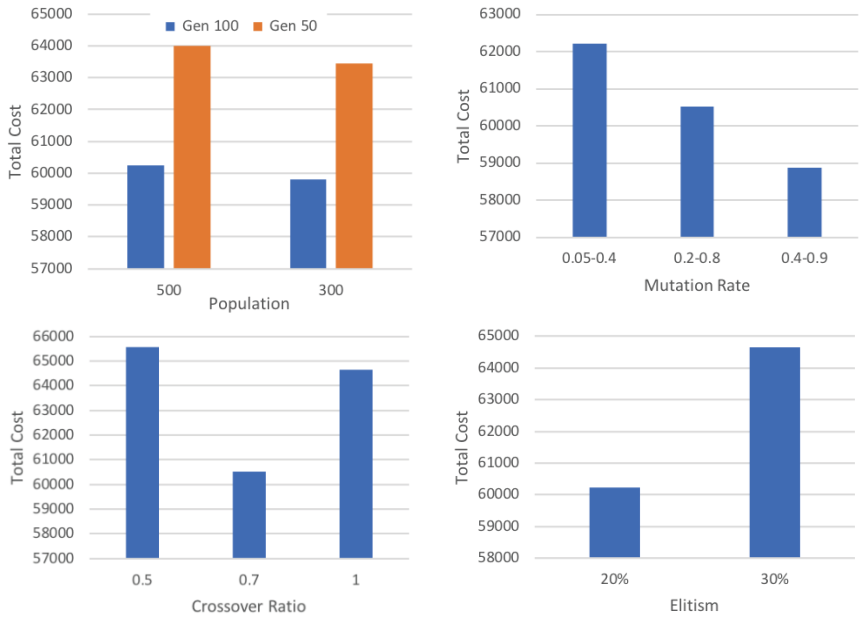

Figure 3 Histograms representing the sensitivity analysis.

5.3 Sensitivity Analysis of the Genetic Parameters

Finding the best parameters for a genetic algorithm is not an easy task. Those parameters can vary a lot between one application and another, which means that there is not a fixed set of them that you can blindly use. In order to find the optimal setting for our genetic algorithm, we performed a set of tests by simulating the optimization procedure while modifying the most relevant parameters and recording the results. This procedure has been carried out for both the random approach and the heuristic ones. In particular, the parameters involved in the tests are: Population dimension, number of generations, Elitism, Cross-over factor, Mutation factors. The results for the toy bay are obtained through an average of over five simulations of the optimization and are shown in Figure 3. For compactness, only the results for the heuristics 2 are reported, since the behavior of the other approaches are similar to this one. As shown in the histograms, for a small bay, there is no improvement with the increase of the population, since the

possible solutions are much less. Reducing the number of generations from 100 down to 50 has a detrimental effect in both cases as expected since the optimization does not have enough time to converge to a satisfying value. Contrary to other applications of genetic algorithms, for this problem, the best mutation factor is pretty high. Indeed, with a mutation factor that starts from 0.4 and goes up to 0.9 when the fitness remains constant, the optimization gets a push towards the global optimum. For what concerns the cross-over factor, the best values for the current application is around the 70% of probability of generating the offspring, while the elitism factor around 20% seems to be better thanks to the fact that the optimization does not focus too much on the good solutions that we already have, but tries more new solutions which will help to find the best one.

5.2 ***Results***

The average trend of costs incurred for container handling during the unloading phases for different simulations carried out in the real case is shown in Fig.4 (left). In this case, the first heuristics seem to perform better with a 3% gain over the random case, followed by the third heuristics and the second heuristics with a loss of 23% and 51%. The interesting part can be seen in the right part of Fig. 4: the approaches that had higher costs when unloading the containers, now have the smallest loading costs, with the heuristics 2 that has a 34% gain over the random case. This result suggests that spending time on placing optimally the containers gives a significant advantage for their future retrieval. We can see that the cost of the heuristics 1, which was the "best" in unloading, has a loading cost which is the double of the heuristics 2 (3% more than the random approach), which was the "worst" in the unloading phase. Fig.5 shows the total costs of both loading and unloading operations for the four approaches for the different simulations carried out. In terms of total costs, also here, the second heuristics outperforms the others with a total gain of 20% over the random case. This result highlights that the higher costs incurred in the unloading phase are more than compensated by the smaller costs obtained thanks to the better positioning in the loading phase. All the percentual differences compared to the random case can also be found in the table below.

The right part of Fig.5 shows the behavior of the fitness for the heuristics 2 for each of the five unloaded trains for all the 100 generations. It is possible to observe that, after some generations, the value of the fitness function for every train remains constant and then restarts decreasing, thanks to the gradual increase of the mutation probability, which helps the algorithm to escape from local minima. This outcome is evident, for example, for the 5th train, from the 42nd to the 67th generation.

Table 2 Comparison between the actual costs of the random approach and the heuristic approaches

	Unloading Cost (Difference %)	Loading Cost (Difference %)	Total Loading/Unloading Cost (Difference %)
Random	76,200 (0%)	401,400 (0%)	477,600 (0%)
Heuristics 1	73,800 (-3%)	413,200 (3%)	487,000 (2%)
Heuristics 2	115,400 (51%)	265,000 (-34%)	380,400 (-20%)
Heuristics 3	93,400 (23%)	354,600 (-12%)	448,000 (-6%)

6. Conclusions

In this paper, a solution procedure for the optimization of the reshuffles of containers at an inland terminal has been presented together to a sensitivity analysis. The innovation of the procedure is the introduction of three heuristics that help the generation of a better first population, which improves the convergence of the optimization. The solution procedure uses costs that are expected to be incurred in the future when the unloaded containers will be retrieved. The algorithms have been tested in a simulation of a real-sized bay by computing the costs incurred during the entire loading and unloading process of a yard bay.

Results highlighted that the use of a suitable heuristic to generate the initial population outperforms the genetic algorithm, initialized with a random solution, by 20%.

The algorithm can be applied in a dynamic context. It is, in fact, straightforward to update the list of containers and their features with new information received in real-time, such as arrivals of new containers that have to be stored in the stocking yard; a new list of containers to retrieve; update the expected container departure times.

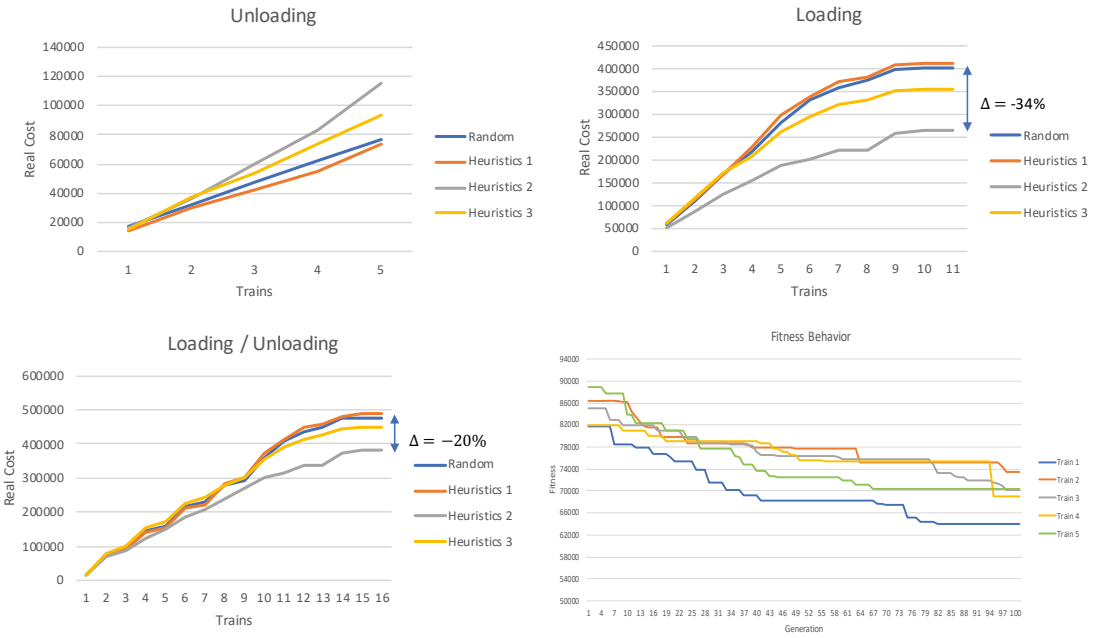

Figure 4 Total actual costs for container reshuffles incurred over five simulation runs in the phases of unloading (up left), loading (up right), loading and unloading (bottom left) corresponding to the implementation of the four approaches and Fitness behavior for the heuristics 2 over 100 generations for each of the five unloading trains (bottom right).

References

1. Murty, K. G., Liu, J., Wan, Y. W., & Linn, R. (2005). A decision support system for operations in a container terminal. Decision Support Systems, 39(3), 309-332.
2. Caserta, M., Schwarze, S., & Voß, S. (2011). Container rehandling at maritime container terminals. In Handbook of terminal planning, 247-269. Springer New York.
3. Tang, L., Jiang, W., Liu, J. & Dong, Y. (2015). Research into container reshuffling and stacking problems in container terminal yards, IIE Transactions, 47:7, 751-766.
4. Colombaroni, C., Fusco, G., & Isaenko, N. (2019). Optimization and simulation approach of containers handling operations at intermodal terminals. Advances in Transportation Studies.Vol.3, 93-106.
5. Carlo, H. J., Vis, I. F., & Roodbergen, K. J. (2014). Storage yard operations in container terminals-Literature overview, trends, and research directions. European Journal of Operational Research, 235(2), 412-430.
6. Gharehgozli, A., Mileski, J.P. & Duru O. (2017) Heuristic estimation of container stacking and reshuffling operations under the containership delay factor and mega-ship challenge. Maritime Policy & Management, 44:3, 373-391.
7. Lin, D.Y. & Chiang, C.W. (2017) The Storage Space Allocation Problem at a Container Terminal, Maritime Policy & Management, 44:6, 685-704.
8. Güven, C. & Eliiyi, D.T. (2019) Modelling and optimisation of online container stacking with operational constraints, Maritime Policy & Management, 46:2, 201-216.
9. Carrese, S., & Tatarelli, L. (2011). Optimizing the stacking of the Intermodal Transport Units in an inland terminal: an heuristic procedure. Proc.Social Behav. Sci., 20, 994-1003.
10. Pap, E., Bojanic, G., Ralevic, N., Georgijevic, M., & Bojanic, V. (2012). Crane scheduling method for train reloading at inland intermodal container terminal. In IEEE 10th Jubilee Int. Symp. Intelligent Systems and Informatics (SISY), 189-192.
11. Wu, K. C., & Ting, C. J. (2010, September). A beam search algorithm for minimizing reshuffle operations at container yards. In Proceedings of the international conference on logistics and maritime systems, 15-17.
12. C. Zhang. Resource Planning in Container Storage Yard. PhD thesis, Hong Kong University of Science and Technology, 2000.
13. R. Jovanovic, M. Tuba, and S. Voß. A multi-heuristic approach for solving the premarshalling problem. Central European Journal of Operations Research, 25(1):1–28, 2017.
14. Ting, Ching-Jung & Wu, Kun-Chih, 2017. "Optimizing container relocation operations at container yards with beam search," Transportation Research Part E: Logistics and Transportation Review, Elsevier, vol. 103(C), pages 17-31.