ERASMUS UNIVERSITEIT ROTTERDAM
ERASMUS SCHOOL OF ECONOMICS

Bachelor Thesis

Econometrics and Operations Research
Quantitive Logistics

Heuristics for a berth allocation problem with additional constraints on the movement of the quay cranes

Name student: Emma Boelen
Student ID number: 431336

Supervisor: N. Milovanovic
Second assessor: T.A.B. Dollevoet

7th July 2019

Abstract

This thesis explores the berth allocation and crane assignment problem, with in particularly the heuristics and meta-heuristics construction heuristic, local refinements, squeaky wheel optimisation, tabu search and simulated annealing. The speed and cost are examined and it is found that the tabu search is the best performing meta-heuristic. Further, the movement of the quay cranes are restricted along a rail so that the different quay cranes cannot pass each other. A scenario where maintenance needs to be done on a quay crane is also explored.

Table of content

1 Introduction **1**

2 Literature review **2**

3 Model **3**

4 Research methodology **4**
4.1 Construction heuristic with local refinements 4
4.2 Squeaky Wheel Optimisation 6
4.3 Tabu Search 7
4.4 Simulated Annealing 8

5 Quay Crane Movement **8**
5.1 Quay Crane Movement Restriction 8
5.2 Quay Crane Maintenance 9

6 Results **10**
6.1 BACAP Heuristics and Meta-heuristics 10
6.2 Quay Cranes Movement Restricted 13

7 Conclusion **19**

1 Introduction

Exporting and importing goods through a shipping container is currently the cheapest and least fuel consuming method per unit weight when transporting goods. Due to growing economies, globalisation and the corresponding increase in demand for products, in combination with a limitation on the growth of ports, it is of great importance that the logistics in a port are correct and efficient. This paper focuses on a discrete dynamic berth allocation and quay crane assignment problem (BACAP) and the methodology presented in Meisel and Bierwirth (2007). Firstly the implementation of a construction heuristic with local refinements is done followed by two meta-heuristics: a squeaky wheel optimisation (SWO) and a tabu search (TS). Furthermore, another meta-heuristics is explored, namely simulated annealing (SA) from Kim and Moon (2001). Finally an extension is looked into by adding a constraint on how the quay cranes (QCs) can move along the quay. Further extended by the problem that maintenance on the QCs need to be scheduled. A comparison of all heuristics is made and a conclusion is drawn on which heuristic is of the highest quality.

Container shipping contribute for a large part to global emission of pollution. Having a good logistics methodology resulting in the least amount of fuel emitted is of great importance to issues that extent further than minimizing the monetary cost. Finding such a solution is therefore especially important in the current environment, where minimizing fuel consumption is becoming an increasingly important topic.

A BACAP looks at the way a set of vessels which arrive at a port should be positioned and how the QCs should be assigned to the vessels in the best way so the service costs are minimized. This problem has been much explored in recent literature. Different heuristics have been applied to different variations on this problem. The main problem investigated is the variation where the vessels need to be assigned to a static continuous or a static discrete quay. It is commonly assumed that the QCs can move freely and no further costs or constraints are implemented on the movement of the QCs. In this paper a comparison of different heuristics and meta-heuristics for the BACAP problem taken from different papers is made. The comparison is done on the basis of speed and cost. Further an adjustment is made on the assumption that the QCs can move freely. The movement of the QCs is then limited dependent on the position of the QC such that the QCs can only move along a rail and cannot pass each other anymore.

The paper outline is as follows. In Section 2 a literature review is done on papers which previously explored the BACAP problem. In Section 3 we define the variables that will be used throughout the paper and the objective function is defined. Section 4 explains the methodology of the different heuristic and meta-heuristic methods that will be explored for the BACAP. Several pseudo codes are also given to explain the general outline of the code that can also be found in the appendix. Section 5 looks into the problem where a restriction is made on the movement of the QCs along a rail and the additional problem where maintenance needs to be done on one of the QCs. The results of the problems explained in Section 4 and 5 are presented in Section 6.1 and 6.2 respectively. A conclusion is drawn in Section 7, we also present possible future research here.

2 Literature review

Berth allocation and crane assignment problems can be categorized as either static or dynamic and either discrete, continuous or hybrid. A static BAP has either no arrival times for the incoming vessels or has arrival times, but they are not seen as hard constraints and can be deviated from. A dynamic BAP, however, has given arrival times that cannot be deviated from. A discrete BAP are such that the quay is divided into subdivision called berths, where a vessel can be positioned along several berths, but a maximum of one vessel can be placed at one berth at a specific time. Whereas in a continuous BAP, the vessels can be placed at any arbitrary position along the quay. Hybrid problems also have berths which are subdivisions of the quay, but the berths are also flexible on how many vessels can occupy one berth at one time.

BAP is first seen in Lai and Shih (1992) and has since received a lot of attention in the literature, especially over the past few years. Imai et al. (1999), Imai et al. (2003a) and Golias et al. (2007) all look at the discrete dynamic case. A linear MIP problem, lagrangian relaxation and genetic algorithm are applied to minimize the waiting and handling times. Imai et al. (1999), who applies a lagrangian relaxation, also allows for the priority list to deviate from the first come first serve principle to optimise the problem, but does not consider the preferred berthing locations with respect to the position of the containers. Therefore their results overlook this significant aspect making the approach in this paper more realistic. Imai et al. (2003a) also considers different priority lists, but takes the size of the vessels as the basis to create the priority list.

Imai et al. (2003b) and Liu et al. (2005) look at the continuous dynamic case. Simulated annealing, a heuristic decomposition approach and another heuristic which places vessels in a feasible position according to a first-come-first-serve basis are used to minimize the service times or costs. Extra care is taken when positioning the QCs in Liu et al. (2005) such that a safe distance between the QCs is kept and they do not cross each other, since they are positioned along the same rail track, and time for the QC to travel between positions is also considered.

Nishimura et al. (2001) applies a genetic algorithm to solve a dynamic BAP with hybrid spacing. They also take the water-depth into consideration, where the depth of the water along the berths is not constant, resulting in some vessels to be restricted in where they can be positioned.

3 Model

The following variables are defined and used throughout the paper, as defined by Meisel and Bierwirth (2007), in Table 1.

Variables	Description
Input data	
V	set of vessels to be served, $V = \{1, 2, ..., n\}$
Q	number of available QCs
L	number of 10 meter berth segments (length of quay)
H	planning horizon
T	set of h-1 periods, $T = \{0, 1, ..., H - 1\}$
l_i	length of vessel $i \in V$ given as a number of 10 meter segments
b_i^0	desired berthing position of vessel $i \in V$
m_i	crane capacity demand of vessel $i \in V$ given as a number of QC-hours
r_i^{min}	minimum number of QCs needed to serve vessel $i \in V$ simultanously
r_i^{max}	maximum number of QCs allowed to serve vessel $i \in V$ simultanously
R_i	feasible range of QCs assignable to vessel $i \in V$. $R_i = \{r_i^{min}, r_i^{max}\}$
ETA_i	expected time of arrival of vessel $i \in V$
EST_i	earliest starting time if journey of vessel $i \in V$ is speeded up, $EST_i \leq ETA_i$
EFT_i	expected finishing time of vessel $i \in V$
LFT_i	latest finishing time for vessel $i \in V$ without additional penalty cost arising
c_i^1, c_i^2, c_i^4	service cost rates for vessel $i \in V$ given in units of USD 1000 per hour
c^4	operation cost rate given in unit of USD 1000 per QC-hour
α	interference exponent
β	berth deviation factor
Decision variables	
b_i	integer, berthing position of vessel i
s_i	integer, time when the handling of vessel i starts (berthing time)
e_i	integer, time when the handling of vessel i finishes (finishing time)
r_{itq}	binary, set to 1 if exactly q QCs are assigned to vessel i at time t, $q \in R_i$, 0 otherwise
ΔETA_i	integer, required speed up of the vessel i to reach its berthing time, $\Delta ETA_i = (ETA_i - s_i)$
ΔEFT_i	integer, tardiness of vessel i, $\Delta EFT_i = (e_i - EFT_i)^+$
u_i	binary, set to 1 if the finishing time of vessel i exceeds LFT_i, 0 otherwise

Table 1
Definitions of input data and decision variables

The formula that is optimized throughout this paper is the cost function defined in Equation 1.

$$\text{minimize } Z = \sum_{i \in V} (c_i^1 \cdot \Delta ETA_i + v_i^2 \cdot \Delta EFT_i + c_i^3 \cdot u_i + c^4 \cdot \sum_{t \in T} \sum_{q \in R_i} q \cdot r_{itq}) \quad (1)$$

The data used throughout this paper are of similar type as those used in Meisel and Bierwirth (2007). The data consists of 20 instances of 10, 20 and 30 vessels; 60 instances in total. For each instance l_i, m_i, ETA_i, EST_i, LFT_i and b_i^0 is given for all vessels. The remaining data, including L, H, Q, c_i^1, c_i^2, c_i^3, c^4, r_i^{min} and r_i^{max} can be found in Meisel and Bierwirth (2007).

4 Research methodology

A good quality heuristic does not only give a solution that is close to the optimal solution, but does this in a reasonable time frame. This is of great importance because the larger instances that are looked at in this paper involves assigning thirty vessels. But realistically, a quay can be much larger and the number of vessels that are incoming can then also be much larger. Therefore, the speed of the heuristic is also key to allow the heuristic to be applied to larger instances in real life.

Three different meta-heuristics are implemented and compared for speed and accuracy. Firstly the construction heuristic with local refinements is defined. The construction heuristic with local refinements is the basis for all the meta-heuristics, where the meta-heuristics adjust the priority list to try to get a lower total service cost. The BACAP explored with the different meta-heuristics assume that a constant number of QCs are available throughout the problem, and that the movement of a QC is instant and is not restricted by the position of other QCs. This assumption is further discussed and explored in Section 5.

4.1 Construction heuristic with local refinements

The construction heuristic (CH) goes through all vessels in the order of the priority list. Per vessel it goes through all the possible arrival times starting from the ETA. For the given arrival time it firstly looks at b_i^0, if this position is not feasible due to a lack of QCs, then the code continues to the next s_i position in the following order: $[ETA_i, ETA_i + 1, ETA_i - 1, ..., EST_i, ..., H]$. When the position is not feasible due to vessels overlapping, another b_i position is explored in the following order: $[b_i^0, b_i^0 + 1, b_i^0 - 1, ..., 0, ..., L - l_i]$. When a feasible solution is found, firstly the cost for this solution is calculated and compared to the current best solution. Regardless if it is a new best solution, the code continues to the next s_i position. This procedure is done for every vessel until all vessels are assigned, or when a vessel cannot be assigned at any position, making the instance unfeasible. Algorithm 1 is used as pseudo code.

```
for vessel_i ∈ vessels do
    i = prioritylist(vessel_i);
    initialize Z_i* = ∞;
    for s_i = [ETA_i, ETA_i + 1, ETA_i − 1..., EST_i, ..., H] do
        for b_i = [b_i^0, b_i^0 + 1, b_i^0 − 1, ..., 0, ..., L − l_i] do
            if enough QCs available then
                if not overlapping with other vessels then
                    if new best solution then
                        set Z_i* = Z_i(s_i*, b_i*) = Z_i(s_i, b_i);
                    end
                    continue to next s_i
                end
            else
                continue to next s_i
            end
        end
    end
end
```

Algorithm 1: Construction heuristic pseudo code

Local refinements (LR) makes sure that one vessel that is relatively large does not occupy a lot of QCs leaving no QCs for later vessels. Starting at the first vessel in the priority list, p_1, a limit is set on how many QCs can be assigned, $r_{p1}^{|v|}$ within the R_{p1} range. By lowering the amount of QCs that are assigned to this vessel, more QCs will be available for later vessels. Per $r_{p1}^{|v|}$ a CH is done with no further adjustments or constraints to the other vessels and the total service cost is calculated. The service costs are compared and the space-time graph of the $r_{p1}^{|v|}$ corresponding to the lowest total service cost is selected. Vessel p_1 is removed from this space-time graph and assigned again without the additional constraint on r_i^{max}. This is the final position of this vessel. This process is repeated for the next vessel, but with the previous vessel already set in place. The final vessel, p_n can be placed with a normal CH algorithm without further constraints. Algorithm 2 shows the pseudo code for the local refinements. In this algorithm, the function $Construction\_Heuristic\_All(STgraph)$ means that all remaining vessels are assigned using CH. While the function $Construction\_Heuristic(STgraph, i)$ means that only vessel i is assigned using the construction heuristic.

for $vessel_i \in vessels$ **do**
  p_i = *prioritylist*($vessel_i$);
  initialize Space-Time graph (STgraph);
  rmaxoriginal = rmax;
  for $r_{pi}^{|v|} \in R_{pi}$ **do**
    set $rmax = r_{pi}^{|v|}$;
    $Z_i = Construction\_Heuristic\_All(STgraph)$
  end
  $r_{pi,min}^{|v|} = r_{pi}^{|v|}$ *of min* Z_i
  if $r_{pi,min}^{|v|} = rmaxoriginal$ **then**
    set rmax = rmaxoriginal;
    STgraph = Construction_Heuristic(STgraph, i);
  else
    remove $vessel_i$ *from STgraph, where* $rmax = r_{pi}^{|v|}$
    set rmax = rmaxoriginal;
    STgraph = Construction_Heuristic(STgraph, i);
  end
end

Algorithm 2: Construction heuristic with local refinement pseudo code

4.2 Squeaky Wheel Optimisation

Squeaky wheel optimisation (SWO), first seen in Clements et al. (1997), is a mix of the double-back optimisation heuristic and a genetic algorithm. The initial solution for this method is a construction heuristic with local refinement based on a first come first serve (FCFS) priority list. The individual redundant costs are calculated per vessel, where the redundant cost is defined as how much a vessel contributes to the costs minus the minimum cost that is inevitable for this vessel. Two consecutive vessels switch positions in the priority list if the vessel lower in the priority list has a higher redundant cost. In other words the vessel with the highest redundant cost will be served earlier in the next iteration. In total $Q - 1$ swaps can be made within one iteration. With the new priority list a LR is done again, and this new solution is accepted as the new best solution regardless of whether the total service costs are lower or not. This is to escape local minima. If a priority list is found that has already been used before is then the solutions will be trapped in a cycle and no new solutions will be found. To avoid this, the CH is applied in this iteration instead of the LR. The LR is applied again when a priority list is found that has not been seen before. The SWO has two stopping criteria, namely when there are 10 consecutive iterations with no improvement on the best solution found, or if after an iteration is completed, the code is found to have run for more than one hour. The pseudo code is shwon in in Algorithm 3.

Calculate lowerbound cost (LB) per vessel.
Obtain initial solution S with the cost per vessel (CPV) using FCFS.
$Redundant\_Cost_i = CPV_i - LB_i$;
while *stopping condition not met* **do**
  for *vesseli = 1, ..., vessels-1* **do**
    i = prioritylist(vesseli);
    j = prioritylist(vesseli+1);
    if $Redundant\_Cost_i < Redundant\_Cost_j$ **then**
      prioritylist(vesseli) = j;
      prioritylist(vesseli+1) = i;
    end
  end
  Obtain solution and cost per vessel with new priority list
end

Algorithm 3: Pseudo Code SWO

As can be seen in the pseudo code, the SWO uses the priority space and the solution space to try to get a better solution. The algorithm looks at the solution of the last iteration, the solution space, and uses this information to adjust the priority list, the priority space.

4.3 Tabu Search

The tabu search (TS) also explores the option of different priority lists. A tabu search looks at the whole neighborhood of a solution and looks which swap within the neighbourhood is the greatest-decent, starting at an initial feasible solution. TS remembers the last several iterations by adding those swaps in a tabu list. Any swap that is in the tabu list will not be explored when looking at the neighbourhood, therefore cannot be selected again for the next iteration. A swap will stay in the tabu list for a set number of iterations. In relation to the BACAP problem, the tabu search starts with a construction heuristic with local refinement based on a FCFS priority list as the initial solution. The TS then looks at all the neighbourhood priority lists. A swap within the neighbourhood of a BACAP is defined as a priority list that can be constructed by switching the position of two vessels in the list. The TS goes through all swaps by applying a CH to the priority list, if a swap is found that has a total service cost lower than the current best solution, this swap is chosen. If no better solution is found within the entire neighbourhood, then the swap that had the lowest cost is chosen. A LR is done on the priority list corresponding to the chosen swap. This priority list will also be the basis for the next iteration where the neighbourhood of this priority list is explored and is also added to the tabu list. Because the TS always continues to the next neighbourhood regardless of whether the costs are actually lower or not and makes sure it does not circle back to a previously explored priority list by using the tabu list, it is a good method to escape local minima and works well for NP-hard problems according to Bland and Dawson (1991). The stopping criteria is the same as for SWO, when after 10 consecutive iteration no improvement is made on the best solution or when after an iteration the code is found to be running for longer than one hour.

4.4 Simulated Annealing

Kim and Moon (2001) suggest a meta-heuristic algorithm to solve the BACAP problem. Simulated annealing (SA) schedules the temperature using several control parameters. An initial temperature T_0, cooling rate r where $0 < r < 1$ and a temperature length R. An uphill move is defined as when comparing the current solution S with a solution S' and the cost(S') is larger than cost(S). In other words, moving from S to S' results in an increase in costs. While a downhill move is defined as when moving from S to S' results in a decrease in costs. SA allows uphill movements, but uses the assigned temperature parameters to limit how big an uphill movement can be. This is also described in the pseudo code, shown in Algorithm 4, which can also be found in Kim and Moon (2001).

Step 1. Obtain an initial solution S. Let $T = T_0$
Step 2. Repeat the following steps until one of the stopping conditions becomes true.
  Step 2.1. Perform the following loop R times.
    Step 2.1.1. Pick a random neighbor S' of S.
    Step 2.1.2. Let $\delta = cost(S') - cost(S)$. The cost is evaluated by Equation 1.
    Step 2.1.3. If $\delta < 0$ (downhill move), set $S = S$'.
    Step 2.1.4. If $\delta \geq 0$ (uphill move), generate random number, x, from the interval, $(0, 1)$; If $x < exp(-\delta/T)$, then set $S = S'$
  Step 2.2. Set $T = rT$ (Reduce the temperature).

Algorithm 4: Simulated annealing pseudo code, taken from Kim and Moon (2001)

For the BACAP the initial solution is found using the construction heuristic with local refinement using the FCFS priority list. There are two stopping conditions, either when the temperature goes below a certain constant temperature, or when the time limit of one hour is reached.

5 Quay Crane Movement

5.1 Quay Crane Movement Restriction

One of the assumptions made so far is that the QCs can be positioned anywhere along the quay. However, looking at the problem from a more realistic perspective a constraint where the movement of the QCs is restricted is added. All the QCs are now assumed to be placed along a rail that spans horizontally along the quay. Therefore the QCs can move freely along this axis, but cannot pass each other. This complicates the BACAP because now not every QC can be assigned to every arriving vessel anymore, only if the QC can reach the vessel without having to pass another QC. Any incoming vessel now needs to be placed in a position where there is not only space available, but also has QCs available that can reach the vessel. The construction heuristic is used as a basis when extending the problem, all QCs are assigned a position along the quay and are set to busy or free using a binary variable. Vessel i is assigned position p_i where p_i is integer. To make sure the QCs do not pass each other, Equation 2 must be satisfied if there are n vessels and H is the planning horizon.

$$0 \leq p_1 < ... < p_i < p_{i+1} < ... < p_n < H \tag{2}$$

When a vessel that requires four QCs needs to be assigned whilst ten QCs are available, the additional six QCs are distributed equally to the left and the right of the vessel, to try to increase the chances that other vessels can easily be placed at either side. If the distance between the vessel and the vessel directly next to it is smaller than the minimum length of all the vessels that still need to be assigned, that is Equations 3 and 4 do not hold, there is no point on leaving QCs on this side of the vessel. Therefore if one of the equations is not satisfied, all the QCs are moved to the side with the most space.

$$p_i - p_{i-1} + s_{i-1} < l_j, \forall j \in V \tag{3}$$

$$p_{i+1} - p_i + s_i < l_j, \forall j \in V \tag{4}$$

Taking further care that if there are more QCs than spaces available one side of the vessel, then the additional QCs are also moved to the other side of the vessel to ensure that no overlapping of QCs take place and no two QCs are assigned to the same position.

5.2 Quay Crane Maintenance

A further exploration is the scenario that maintenance needs to be done on a QC, resulting in this QC having to stand still for a set time frame on the rail. Firstly, the time when this QC is not available is chosen. This is done by solving the movement restricted problem, explained in Section 5.1, first. The space-time graph constructed is used to calculate when the least amount of berths are occupied during every possible time period that the crane could be stopped. This is done by using a root mean square method (RMS). Once the time is chosen, all possible positions where the QC can be stopped is explored by calculating the corresponding CH. The different position are compared and the position which results in the lowest total service cost is chosen as the position where the QC can best be stopped.

6 Results

6.1 BACAP Heuristics and Meta-heuristics

All instances consist of vessels which are one of the following three types;feeder, medium or jumbo. These types are assigned to a vessels with a probability of 60% for feeder vessels, 30% for medium vessels and 10% for jumbo vessels. Meisel and Bierwirth (2007) provides the general characteristics of these specific vessels, which are provided in Table 2. Every specific instance gives the following data for every vessel: the type, length, crane demand, EST, ETA, LFT and desired berthing position. Furthermore, for the quay we know that the length is 1000 meters, divided into 100 berthing positions of ten meter each. There are ten QCs available over the time span of seven days (168 hours), c_4 is USD 100, β is 0.01 and α is 0.9. Furthermore, for the SA the following parameters are chosen; initial temperature $T = 40$ degrees, cooling rate $CR = 0.65$, iterations before cooling $R = 10$ and minimum temperature is 1 degrees. These numbers are based on Kim and Moon (2001) where these parameters are found to be the best for their BACAP instances.

Class	r_i^{min}	r_i^{max}	c_i^1	c_i^2	c_i^3
Feeder	1	2	1	2	3
Medium	2	4	2	2	6
Jumbo	4	6	3	3	9

Table 2
General characteristics per vessel type. Costs are given in USD 1000. Table is mentioned in Meisel and Bierwirth (2007).

The results of the heuristics and meta-heuristics with no constraints on the movement of the QCs can be found in Table 3 for the instance with ten vessels, Table 4 for the instances with 20 vessels and Table 5 for the instances with 30 vessels. Per instance a lower bound (LB) is calculated. The LB is the sum of the minimum cost the vessels can incur if they are positioned at their preferred berthing location and arriving at their ETA. Therefore not having any additional penalty costs, giving the lowest possible cost the vessels can incur as a lower bound. Then for the five (meta-)heuristics, namely the construction heuristic (CH), local refinement (LR), squeaky wheel optimization (SWO), tabu search (TS) and simulated annealing (SA), the following data is collected: the lowest total service cost found (Z), the running time (Time) and the relative error (RE) with regards to the lower bound calculated by $RE = \frac{Z-LB}{LB}$.

	CH			LR			SWO			TS			SA			
Instance	Z	Time	RE	Z	Time	RE	Z	Time	RE	Z	Time	RE	Z	Time	RE	LB
0	**44.1**	0.2	0.3	**44.1**	2.9	0.3	**44.1**	86.1	0.3	**44.1**	203.9	0.3	**44.1**	546.3	0.3	33.8
1	**11.0**	0.1	0.0	**11.0**	1.1	0.0	**11.0**	22.0	0.0	**11.0**	78.9	0.0	**11.0**	138.6	0.0	11.0
2	44.3	0.2	0.7	44.3	2.8	0.7	34.6	68.4	0.4	**33.5**	157.1	0.3	**33.5**	376.1	0.3	25.6
3	**36.4**	0.2	0.3	**36.4**	2.9	0.3	**36.4**	52.1	0.3	**36.4**	175.9	0.3	**36.4**	460.5	0.3	28.1
4	29.1	0.2	0.5	29.1	2.4	0.5	26.3	63.3	0.4	**23.8**	121.1	0.2	**23.8**	321.6	0.2	19.1
5	58.1	0.2	0.8	58.1	2.2	0.8	51.6	52.4	0.6	51.6	135.9	0.6	**49.6**	443.8	0.5	32.4
6	**27.0**	0.2	0.2	**27.0**	2.2	0.2	**27.0**	42.2	0.2	**27.0**	115.8	0.2	**27.0**	338.0	0.2	22.1
7	56.6	0.2	0.7	56.6	3.3	0.7	55.1	76.3	0.6	**50.5**	183.9	0.5	51.8	479.5	0.5	33.8
8	44.9	0.2	0.7	44.9	2.3	0.7	34.2	52.6	0.3	**32.1**	160.9	0.2	**32.1**	342.4	0.2	26.0
9	**21.4**	0.1	0.1	**21.4**	1.5	0.1	**21.4**	29.1	0.1	**21.4**	92.2	0.1	**21.4**	271.8	0.1	20.3
10	**12.4**	0.1	0.0	**12.4**	1.3	0.0	**12.4**	34.8	0.0	**12.4**	72.9	0.0	**12.4**	167.8	0.0	12.4
11	53.9	0.2	1.1	53.9	3.8	1.1	45.6	101.2	0.8	**42.9**	260.9	0.7	**42.9**	491.6	0.7	25.9
12	33.4	0.2	0.3	33.4	2.3	0.3	33.4	36.8	0.3	**31.4**	115.0	0.2	**31.4**	340.7	0.2	26.1
13	41.2	0.2	0.2	41.2	3.2	0.2	**39.0**	75.8	0.1	**39.0**	143.4	0.1	**39.0**	495.7	0.1	35.4
14	18.8	0.1	0.4	18.8	1.4	0.4	**15.4**	29.9	0.2	**15.4**	91.5	0.2	**15.4**	216.3	0.2	13.1
15	35.9	0.2	0.5	35.9	2.4	0.5	**34.9**	45.6	0.4	**34.9**	122.8	0.4	**34.9**	328.0	0.4	24.4
16	**26.1**	0.1	0.1	**26.1**	2.0	0.1	**26.1**	37.8	0.1	26.1	98.7	0.1	**26.1**	318.1	0.1	22.8
17	36.4	0.2	0.2	36.4	2.7	0.2	**31.5**	60.6	0.1	**31.5**	197.0	0.1	**31.5**	396.4	0.1	29.2
18	**33.7**	0.2	0.1	**33.7**	3.2	0.1	**33.7**	102.7	0.1	**33.7**	128.3	0.1	**33.7**	547.7	0.1	29.8
19	**35.0**	0.2	0.3	**35.0**	2.6	0.3	**35.0**	48.4	0.3	**35.0**	166.9	0.3	**35.0**	428.7	0.3	26.7
Average	35.0	0.2	0.4	35.0	2.4	0.4	32.4	55.9	0.3	31.7	141.2	0.2	31.7	372.5	0.2	24.9

Table 3
Results for all instances with 10 vessels. Time is given in seconds and total service cost (Z) is given in USD 1000. For every instance, the lowest total service cost is put in bold.

	CH			LR			SWO			TS			SA			
Instance	Z	Time	RE	Z	Time	RE	Z	Time	RE	Z	Time	RE	Z	Time	RE	LB
0	44.3	0.6	0.1	44.3	9.0	0.1	**43.0**	293.5	0.1	**43.0**	1159.8	0.1	43.2	1638.7	0.1	39.4
1	51.9	0.4	0.1	51.9	12.0	0.1	**51.1**	527.0	0.1	**51.1**	1981.2	0.1	**51.1**	2457.8	0.1	47.2
2	54.2	0.4	0.3	54.2	12.3	0.3	47.6	287.6	0.1	**47.5**	1441.1	0.1	49.0	2984.7	0.1	42.8
3	34.7	0.3	0.1	34.7	8.6	0.1	33.7	284.4	0.0	**33.5**	956.1	0.0	**33.5**	1526.2	0.0	32.6
4	87.6	0.5	0.7	80.6	14.0	0.6	74.3	294.6	0.5	**70.6**	2063.3	0.4	71.2	2917.1	0.4	51.2
5	79.6	0.5	0.6	79.6	16.1	0.6	**65.8**	794.8	0.3	66.3	1790.2	0.3	67.3	2512.5	0.3	50.8
6	52.4	0.4	0.1	52.4	11.1	0.1	50.3	429.9	0.1	**50.2**	1122.7	0.1	**50.2**	1962.7	0.1	47.7
7	168.9	0.5	1.5	168.9	15.3	1.5	168.9	336.8	1.5	163.2	1523.4	1.4	**161.6**	3451.1	1.4	68.0
8	119.3	0.5	0.9	119.3	15.4	0.9	111.5	798.6	0.8	**96.2**	3769.9	0.5	97.0	2755.6	0.5	63.2
9	69.8	0.4	0.4	69.8	9.8	0.4	62.6	464.9	0.2	**55.7**	1443.9	0.1	58.0	2672.2	0.1	50.7
10	44.2	0.3	0.3	44.2	8.2	0.3	38.9	336.4	0.2	**36.0**	1071.4	0.1	**36.0**	1371.9	0.1	33.5
11	49.5	0.6	0.2	49.5	10.8	0.2	47.2	276.0	0.1	**46.9**	1217.6	0.1	47.2	1658.9	0.1	41.5
12	**54.7**	0.5	0.1	**54.7**	12.7	0.1	**54.7**	321.0	0.1	**54.7**	1157.4	0.1	**54.7**	2535.0	0.1	48.5
13	95.7	0.5	1.0	91.6	15.4	0.9	88.9	545.0	0.8	**70.4**	1911.4	0.4	**70.4**	2515.8	0.4	48.9
14	125.3	0.4	1.0	117.3	13.1	0.9	117.3	433.2	0.9	111.4	1959.1	0.8	**104.8**	2644.7	0.7	63.3
15	79.6	0.5	0.6	79.6	16.1	0.6	**65.8**	809.0	0.3	66.3	1684.6	0.3	68.8	2991.6	0.3	51.2
16	54.0	0.3	0.1	54.0	9.5	0.1	53.9	501.9	0.1	**53.8**	1513.0	0.1	54.0	2135.5	0.1	49.0
17	46.7	0.4	0.1	46.7	9.6	0.1	45.8	269.5	0.1	**44.4**	1458.9	0.0	**44.4**	1968.3	0.0	43.3
18	60.3	0.4	0.3	60.3	11.4	0.3	60.3	304.9	0.3	**54.4**	2338.1	0.1	**54.4**	2424.5	0.1	47.4
19	86.2	0.4	0.8	86.2	14.0	0.8	86.1	399.8	0.8	74.9	2265.8	0.6	**71.7**	2723.4	0.5	47.3
Average	72.9	0.4	0.5	72.0	12.2	0.4	68.4	435.5	0.4	64.5	1691.5	0.3	64.4	2392.4	0.3	48.4

Table 4
Results for all instances with 20 vessels. Time is given in seconds and total service cost (Z) is given in USD 1000. For every instance, the lowest total service cost is put in bold.

	CH			LR			SWO			TS			SA			
Instance	Z	Time	RE	Z	Time	RE	Z	Time	RE	Z	Time	RE	Z	Time	RE	LB
0	72.6	0.5	0.4	72.6	19.1	0.4	72.6	803.8	0.4	**64.2**	3652.9	0.2	70.0	3618.4	0.3	53.4
1	288.3	0.9	2.5	271.7	37.8	2.3	271.7	715.3	2.3	**147.4**	3909.8	0.8	173.0	3663.2	1.1	83.0
2	86.3	0.6	0.6	86.3	19.3	0.6	85.6	475.0	0.6	**75.1**	3202.2	0.4	78.8	3629.1	0.5	53.7
3	237.4	0.9	2.0	225.2	36.1	1.9	225.2	642.7	1.9	**165.7**	3821.3	1.1	199.5	3632.4	1.5	78.4
4	183.3	0.7	1.4	163.6	29.7	1.2	133.9	2641.5	0.8	**109.1**	3643.8	0.4	132.2	3635.3	0.8	75.3
5	234.2	0.8	2.0	220.4	25.9	1.8	211.5	873.4	1.7	196.8	3966.2	1.5	**195.0**	3677.3	1.5	79.3
6	236.9	0.9	2.4	207.3	30.8	1.9	184.8	796.0	1.6	**162.2**	3768.1	1.3	182.6	3639.1	1.6	70.7
7	255.3	0.7	2.5	244.8	26.5	2.3	159.0	2060.1	1.2	**150.6**	4145.0	1.0	164.1	3603.5	1.2	73.6
8	-	-	-	336.4	59.1	2.7	316.2	694.6	2.5	**280.2**	3974.0	2.1	301.2	3662.7	2.3	91.1
9	56.0	0.6	0.2	56.0	18.2	0.2	53.2	963.4	0.2	**51.7**	2952.6	0.1	52.4	3618.3	0.2	45.1
10	271.5	0.9	2.3	271.5	34.5	2.3	237.2	2711.4	1.9	**194.2**	3773.2	1.4	217.8	3632.1	1.7	81.3
11	425.3	1.1	4.5	389.5	55.4	4.1	**296.3**	2211.7	2.8	299.3	3820.1	2.9	326.6	3645.0	3.2	77.1
12	195.2	0.7	1.5	184.0	29.0	1.3	171.6	837.0	1.2	**124.2**	3683.1	0.6	131.3	3618.0	0.7	78.7
13	-	-	-	-	-	-	-	-	-	315.9	1659.9	2.0	**330.9**	3644.9	2.2	104.9
14	96.3	0.6	0.5	96.3	20.8	0.5	93.4	827.5	0.5	**90.0**	3789.7	0.4	93.3	3638.7	0.5	63.8
15	165.2	0.9	1.3	133.3	35.5	0.9	**110.9**	2116.8	0.6	118.0	3761.4	0.7	120.8	3657.1	0.7	71.3
16	205.7	0.8	1.8	198.6	32.1	1.7	173.2	2318.5	1.3	**150.9**	3948.1	1.0	172.0	3643.1	1.3	74.4
17	197.2	0.7	1.8	197.2	29.7	1.8	197.2	725.7	1.8	**154.5**	4197.3	1.2	159.6	3669.2	1.3	70.5
18	247.2	0.7	2.2	215.4	34.2	1.8	202.0	739.9	1.6	**184.4**	3825.9	1.4	192.6	3631.5	1.5	76.3
19	80.5	0.5	0.5	80.5	21.0	0.5	80.5	523.3	0.5	**73.5**	3891.3	0.4	80.5	3603.1	0.5	53.6
Average	196.4	0.7	1.7	192.1	31.3	1.6	172.4	1246.2	1.3	155.4	3669.3	1.0	168.7	3638.1	1.2	72.8

Table 5
Results for all instances with 30 vessels. Time is given in seconds and total service cost (Z) is given in USD 1000. For every instance, the lowest total service cost is put in bold.

Two instances were found to be unfeasible when the construction heuristic with local refinements and the squeaky wheel optimisation were applied, namely instance eight and 13 for 30 vessels, as can be seen in Table 5. Overall, the tabu search did work best out of the (meta-)heuristic methods. Although SWO reached a stopping criteria much faster, with an average time of 1246.2 seconds (roughly 21 minutes) for all instances of 30 vessels compared to an average time of 3669.3 seconds (roughly 61 minutes) for TS and 3638.1 seconds (roughly 61 minutes) for SA. It was found that SWO performed less good compared to the TS. Although TS ran the longest as it reached the one hour limit frequently, it did perform the best. The lowest total service cost are put in bold in Table 3, 4 and 5. When comparing which (meta-)heuristic found the lowest total service cost, TS found the lowest cost in 18 instances for 10 vessels, although it should be noted that for 9 of these instances this service cost was also already found at the construction heuristic. For the instances with 20 vessels TS found the lowest value for 15 instances, and was only outperformed three times by SA and once by SWO for the instances of 30 vessels. Therefore TS clearly performed the best for these BACAPs. Especially considering that the time constraint stopped the code the majority of the time. Therefore if a port has enough time, if for example they only need the results later on in the day or the next day, it could be possible to increase the time limit and let the TS try to get an even better solution.

6.2 Quay Cranes Movement Restricted

When the QCs are attached along a rail and can therefore not pass each other, the total service cost do, as expected, increase. Only one instance, namely instance 2 for 10 vessels had a slight decrease in cost. The results for the instances with 10 and 20 vessels can be found in Table 6. The results for the instances with 30 vessels can be found in Table 7 together with the results for when a QC is stopped for maintenance.

	n = 10					n = 20				
	Normal		Extended			Normal		Extended		
Instance	Z	time	Z	time	Increase in Z	Z	time	Z	time	Increase in Z
0	44.1	0.2	58.0	0.6	0.32	44.3	0.6	61.1	0.8	0.38
1	11.0	0.1	13.0	0.2	0.18	51.9	0.4	103.2	0.9	0.99
2	44.3	0.2	43.2	0.4	-0.02	54.2	0.4	69.0	0.7	0.27
3	36.4	0.2	40.2	0.4	0.10	34.7	0.3	54.1	0.9	0.56
4	29.1	0.2	32.1	0.4	0.10	87.6	0.5	95.0	1.2	0.08
5	58.1	0.2	77.2	0.4	0.33	79.6	0.5	150.4	0.8	0.89
6	27.0	0.2	27.0	0.4	0.00	52.4	0.4	65.3	1.3	0.25
7	56.6	0.2	59.5	0.6	0.05	168.9	0.5	298.3	1.1	0.77
8	44.9	0.2	50.6	0.5	0.13	119.3	0.5	197.8	0.9	0.66
9	21.4	0.1	21.4	0.3	0.00	69.8	0.4	90.4	0.7	0.30
10	12.4	0.1	12.4	0.2	0.00	44.2	0.3	54.6	0.7	0.24
11	53.9	0.2	60.1	0.5	0.12	49.5	0.6	79.7	0.9	0.61
12	33.4	0.2	37.4	0.4	0.12	54.7	0.5	87.2	0.8	0.59
13	41.2	0.2	54.4	0.6	0.32	95.7	0.5	134.0	0.8	0.40
14	18.8	0.1	18.8	0.2	0.00	125.3	0.4	170.8	1.2	0.36
15	35.9	0.2	35.9	0.4	0.00	79.6	0.5	164.7	0.7	1.07
16	26.1	0.1	26.1	0.3	0.00	54.0	0.3	72.5	0.7	0.34
17	36.4	0.2	36.4	0.4	0.00	46.7	0.4	69.7	0.8	0.49
18	33.7	0.2	39.5	0.4	0.17	60.3	0.4	79.8	0.8	0.32
19	35.0	0.2	38.8	0.4	0.11	86.2	0.4	136.5	0.7	0.58
Average	35.0	0.2	39.1	0.4	0.10	72.9	0.4	111.7	0.9	0.51

Table 6
The results with constrained QC movement (Extended) compared to the CH results when the QC could move anywhere (Normal, these results are also found in Table 3 and 4). The total service cost (Z) is given in 1000 USD and time is given in seconds. Increase in Z is the percentage increase of Z from Normal to Extended.

	Normal	Extended	QC1	QC2	QC3	QC4	QC5	QC6	QC7	QC8	QC9	QC10
n30i0, RMS = 6												
Z	72.6	97.5	92.8	97.5	97.5	97.5	98.5	98.5	98.5	99.5	99.5	98.2
time	0.5	1.2	104.8	93.1	88.9	90.7	89.5	90.3	89.1	93.3	96.0	101.0
position	-	-	[49]	[23-30]	[23-30]	[23-30]	[36]	[36]	[36]	[7-30, 41-98]	[41-99]	[27-35]
n30i1, RMS = 9												
Z	288.3	406.6	406.5	406.4	395.7	394.8	387.9	398.2	39.2	394.7	394.7	395.8
time	0.9	1.9	172.3	163.4	158.5	157.3	154.9	156.0	153.8	153.0	153.7	160.0
position	-	-	[1-7]	[2-7]	[3-5]	[23-25]	[65]	[80-83]	[26-31]	[83]	[83]	[65-84]
n30i2, RMS = 1												
Z	86.3	104.8	102.1	102.1	102.1	102.1	105.1	110.2	104.8	104.8	104.8	104.8
time	0.6	1.0	104.8	93.1	88.9	90.7	89.5	90.3	89.1	93.3	96.0	101.0
position	-	-	[16]	[16]	[16]	[16]	[16-18]	[15-19]	[54-97]	[54-98]	[54-99]	[54-100]
n30i3, RMS = 19												
Z	237.4	529.2	505.4	518.4	489.3	451.9	451.9	451.9	470.3	461.4	513.9	531.8
time	0.9	2.1	195.2	189.7	173.3	177.2	172.0	172.8	180.2	180.8	191.6	198.2
position	-	-	[1-8]	[2-8]	[27]	[69]	[69]	[69]	[68]	[70]	[76, 87-90]	[70]
n30i4, RMS = 14												
Z	183.3	196.3	196.3	196.3	196.3	196.3	196.3	196.3	196.3	196.3	196.3	196.3
time	0.7	1.3	125.4	117.2	112.7	113.2	108.0	108.0	108.2	107.5	111.2	115.0
position	-	-	[1-18]	[2-18]	[3-18]	[4-1]	[11-18, 44-66]	[44-66]	[44-66, 81-97]	[44-66, 81-98]	[44-66, 80-99]	[81-100]
n30i5, RMS = 6												
Z	234.2	281.6	281.6	28.6	284.6	284.6	281.4	281.5	281.6	281.6	281.6	281.6
time	0.8	1.5	132.0	128.3	128.9	124.8	134.4	131.7	120.0	125.4	132.9	198.2
position	-	-	[1]	[2-27]	[3-27]	[4-27, 49-94]	[5-27, 49-95]	[6-27, 49-96]	[7-28, 49-97]	[49-98]	[49-99]	[49-100]
n30i6, RMS = 15												
Z	236.9	287.9	287.9	287.9	287.9	283.4	281.9	278.8	275.9	276.1	275.7	276.7
time	0.9	1.5	154.9	147.9	124.2	124.7	124.0	123.3	120.7	121.1	123.8	130.0
position	-	-	[1-8]	[2-8]	[3-8]	[31]	[42]	[39]	[29-30]	[29-30]	[29-30]	[30]
n30i7, RMS = 3												
Z	255.3	377.5	377.5	377.5	377.5	377.5	377.5	377.5	377.5	377.5	377.5	377.5
time	0.7	1.5	145.7	141.0	133.8	134.4	134.2	132.8	137.8	136.5	142.9	147.3
position	-	-	[1-91]	[2-92]	[3-93]	[4-94]	[5-95]	[6-96]	[7-97]	[8-98]	[9-99]	[10-100]
n30i8												
Z	-	-	-	-	-	-	-	-	-	-	-	-
time	-	-	-	-	-	-	-	-	-	-	-	-
position	-	-	-	-	-	-	-	-	-	-	-	-
n30i9, RMS = 4												
Z	56.0	68.9	68.9	68.9	68.9	68.9	68.9	68.9	68.9	68.9	68.9	68.9
time	0.6	1.1	108.0	101.0	100.5	99.7	99.7	99.6	100.5	101.2	99.8	110.1
position	-	-	[1-28]	[2-28]	[21-28]	[21-28]	[56-65]	[56-65]	[56-65]	[56-65]	[82-85, 94-99]	[94-100]
n30i10, RMS = 2												
Z	271.5	365.3	341.3	341.2	39.7	339.9	338.0	338.0	347.3	341.2	341.2	341.3
time	0.9	1.8	173.1	170.6	152.2	156.3	153.5	156.9	159.5	159.9	164.5	167.7
position	-	-	[1-19]	[2-18]	[45]	[39-43]	[45]	[45]	[68-69]	[69-73]	[92-99]	[91-100]
n30i11, RMS = 12												
Z	425.3	505.7	505.7	505.7	505.7	505.7	505.7	505.7	505.7	505.7	505.7	505.7
time	1.1	2.0	191.2	187.8	179.8	178.3	179.5	180.8	181.9	181.4	191.0	194.0
position	-	-	[1-91]	[2-92]	[3-93]	[4-94]	[5-95]	[6-96]	[7-97]	[8-98]	[9-99]	[10-100]
n30i12, RMS = 14												
Z	195.2	262.5	262.4	262.4	266.5	266.4	262.0	257.3	253.0	262.0	262.4	262.4
time	0.7	1.5	153.4	142.3	136.8	136.2	131.4	129.8	134.2	133.0	138.3	153.1
position	-	-	[1-7]	[2-7]	[8-10, 20-21]	[8-10]	[53]	[47-48]	[48]	[48]	[54-99]	[54-100]
n30i13												
Z	-	-	-	-	-	-	-	-	-	-	-	-
time	-	-	-	-	-	-	-	-	-	-	-	-
position	-	-	-	-	-	-	-	-	-	-	-	-
n30i14, RMS = 19												
Z	96.3	160.1	160.1	160.1	159.1	159.1	159.1	159.1	159.1	159.1	159.1	159.1
time	0.6	1.3	133.7	121.3	117.7	115.7	115.2	115.4	120.2	120.9	123.3	130.8
position	-	-	[1-41]	[2-41]	[46-47]	[46-47]	[46-47]	[46-47]	[46-47]	[46-47]	[46-47]	[46-47]
n30i15												

	Normal	Extended	QC1	QC2	QC3	QC4	QC5	QC6	QC7	QC8	QC9	QC10
Z	165.2	-	-	-	-	-	-	-	-	-	-	-
time	0.9	-	-	-	-	-	-	-	-	-	-	-
position	-	-	-	-	-	-	-	-	-	-	-	-
n30i16, RMS = 7												
Z	205.7	274.5	274.5	274.5	274.5	274.5	274.5	274.5	274.5	274.5	274.5	274.5
time	0.8	1.6	155.1	145.8	146.4	144.4	143.0	138.4	137.9	141.4	143.9	148.7
position	-	-	[1-91]	[2-92]	[3-93]	[4-94]	[5-95]	[6-96]	[7-97]	[8-98]	[9-99]	[10-100]
n30i17, RMS = 4												
Z	197.2	280.0	266.9	266.9	266.1	264.1	262.2	262.2	269.0	266.1	257.1	252.4
time	0.7	1.6	149.6	147.5	147.3	145.7	144.1	143.1	145.0	145.0	140.9	146.8
position	-	-	[43-44]	[43-44]	[42]	[42]	[46]	[46]	[42]	[66-75]	[30]	[31]
n30i18, RMS = 5												
Z	247.2	272.9	272.9	272.9	272.9	272.8	272.9	272.9	272.8	284.6	284.6	284.6
time	0.7	1.8	162.6	154.3	153.8	153.5	150.0	150.0	151.1	155.6	161.8	164.6
position	-	-	[20-27]	[20-27]	[20-27]	[49-67]	[49-67, 69]	[49-67, 69]	[7-27]	[49-98]	[49-99]	[49-100]
n30i19, RMS = 16												
Z	80.5	106.0	106.0	105.9	106.0	106.0	106.0	106.0	106.0	106.0	106.0	105.9
time	0.5	1.1	117.3	103.9	102.1	101.8	102.5	101.4	102.4	100.0	105.8	113.9
position	-	-	[1-8]	[19-92]	[3-8, 14-93]	[4-8, 14-94]	[5-9, 14-95]	[6-8, 15-96]	[7-8, 16-97]	[8-17-98]	[18-99]	[10-18]

Table 7
The results when the QCs are constrained in their movement (Extended) is compared to the CH results when the QC could move anywhere also found in Table 3 and 4. The total service cost (Z) is given in USD 1000 and time is given in seconds.

For the movement restricted results in Table 6, the increase in Z is also shown. This represent the increase in total service cost with respect to the CH instance when there was no restriction on the movements on the QCs. It can be seen that the total service cost increase consistently, with an average of 10% for the instance with 10 vessels and an average of 51% for the instances of 20 vessels.

For the problem when a QC needs to be stopped for maintenance, the time that the QC is stopped is based upon when the least amount of vessels are at the quay during a five hour frame according to RMS. It does happen that no vessels are at the quay for a full five hour period, therefore the QC would be stopped during this period, having no further impact on the assignment of the vessels. Therefore only the instances with 30 vessels are explored for this extension, because the instances with 10 and 20 vessels will have a high chance of having a 5 hour period with no vessels at the quay. Leading to results that do not contribute to exploring how stopping a QC will influence the vessels. The QC needs to stand still for a five hour period within a specific day so maintenance can be done, day four is chosen (so from hour 72 to hour 95) so that there are no issues with vessels still needing to come in, or leave before the end of the period like you would have on day one and seven. To calculate day four for all 30 vessel instances is also consistent and allows for better comparison.

It was found that setting a QC still for maintenance actually decreased the cost for quite a few instances. Although usually adding more constraints leads to higher cost, it is not too surprising. When a QC stands still it might move a larger vessels that takes up a lot of space and thereby increasing the cost slightly, but creating space for other vessels which will then decrease their cost.

One example of this can be seen in Figure 1 and 2, this is instance three of 30 vessels when QC four needs maintenance on position 69 represented by the black box. Figure 1 firstly shows the assignment of the vessels with no QC standing still for maintenance, leading to a total service cost of USD 529,200. Figure 2 then shows how the vessels shift with the QC standing still. The vessels in dark grey represent all vessels that are placed at a different position, only the vessels from hour 75 onwards are shown, as previous to this point in time there are no differences in assignment of vessels. With the movement of vessel one, vessel eight arrives earlier reducing a very high penalty cost, decreasing the cost for vessel eight from USD 41,500 to USD 33,400. This has a further domino effect on more vessels being able to decrease their cost, decreasing the total service cost to USD 451,900. A total reduction of USD 77,300. The SWO or TS would also likely have allowed vessels with such a high redundant cost to be assigned earlier, so applying a meta-heuristic to this problem would have likely also decreased the cost. But due to the large amount of instances that needed to be calculated, this was not possible due to time constraints. If the specific QC that needs maintenance is known beforehand, which would be the case in real life, then a meta-heuristic can be applied to find an even better solution in a reasonable time frame.

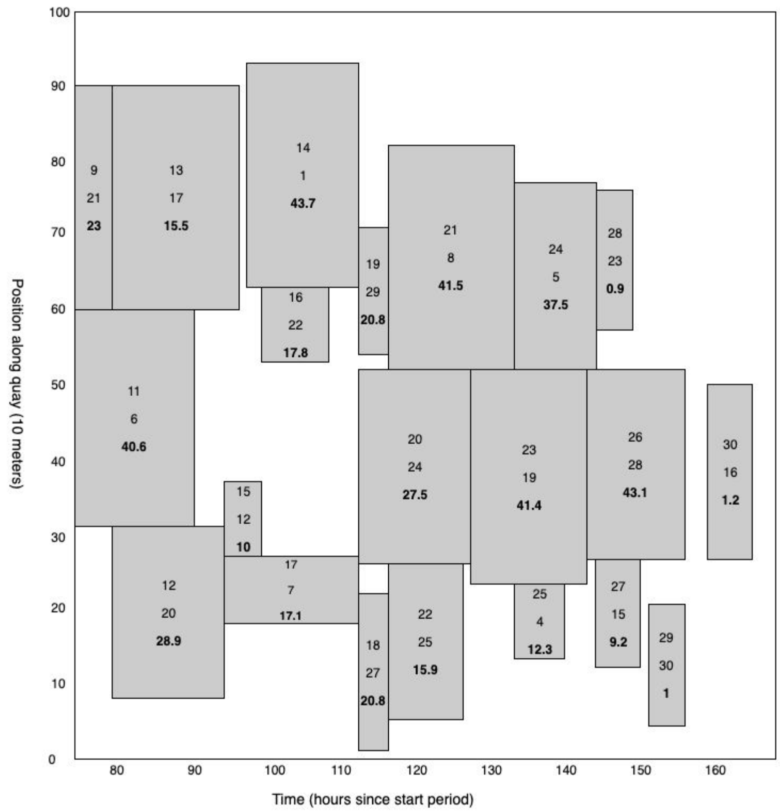

Figure 1
The assignment of the vessels of instance three with 30 vessels with movement restriction on the QCs. Each box is a vessel, shown in the space-time diagram. The first number in the box is the order of arrival therefore the order of placement, the second number is the vessel number and the third number is the service cost that this vessel contributes in USD 1000.

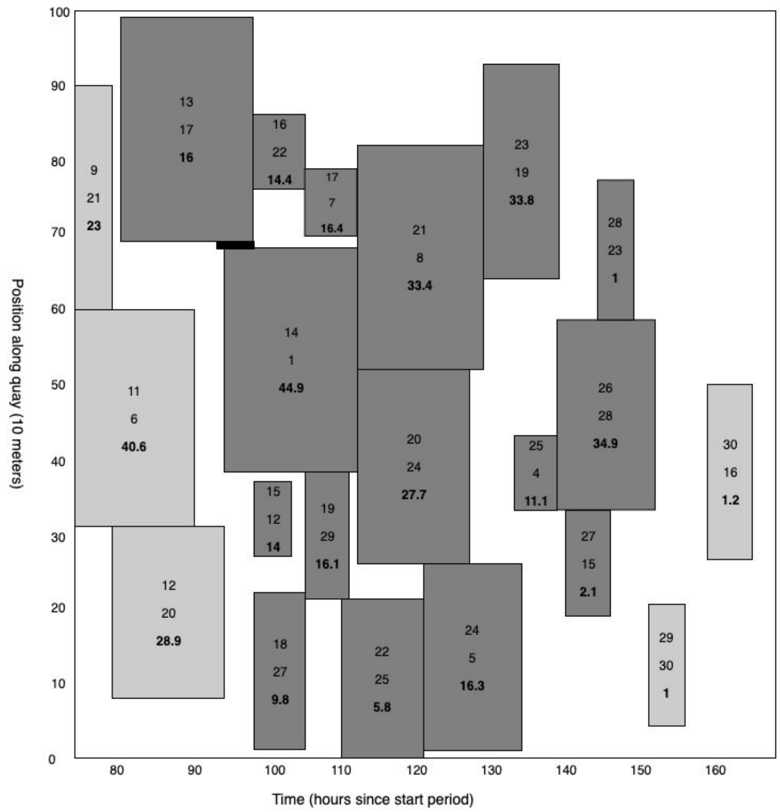

Figure 2
The assignment of the vessels shown in the space-time diagram is of instance three with 30 vessels, with QC four stopped for maintenance for five hours at position 69. The first number in the vessels is the order of arrival of the vessel therefore the order of placement, the second number is the vessel number and the third number is the service cost that this vessel contributes in USD 1000.

7 Conclusion

This paper explores different heuristics and meta-heuristics to assess which method is best at realistically assigning vessels in a BACAP. The position of the vessels when assigning them is explored, where a position should be found as close that is as possible to its preferred berthing location and its estimated time of arrival. But also the quay cranes are a limited resource on the quay and need to be assigned efficiently. The heuristics explored are a construction heuristic with local refinement, while the meta-heuristics explored are the squeaky wheel optimisation, tabu search and simulated annealing. It is found that the tabu search, although taking longer than the other methods, gives the best results. Furthermore the extension when the quay cranes are restricted in its movement is explored. It is found that the total service cost increases with 10% on average for the instances with 10 vessels, and with an average of 51% for the instances with 20 vessels. Additionally, the scenario when a quay crane needs to be set still for a five hour period for maintenance is explored. For all of the ten quay cranes it is calculated what the optimal position is to get the lowest total service cost and it was found that frequently setting a quay crane still allows shifts in the vessel positions that decreases the total service cost.

Future research can be done on looking at the possibility to apply a meta-heuristic to the crane movement restricted instances, to see how well the meta-heuristics would improve the result further. Also further research can be done on the methodology on how the QC not assigned to a vessel could be distributed to the left and the right of the vessel, so more incoming vessels can be positioned at preferred positions.

References

Bland, J. A. and Dawson, G. P. (1991). Tabu search and design optimization. *Computer-aided design.*

Clements, D. P., Crawford, J. M., Joslin, D. E., Nemhauser, G. L., Puttlitz, M. E., and Savelsbergh, M. W. P. (1997). Heuristic optimization: A hybrid ai/or approach.

Golias, M., Boile, M., and Theofanis, S. (2007). The berth allocation problem: a formulation reflecting time window service deadlines. Technical report.

Imai, A., Nishimura, E., and Papadimitriou, S. (1999). The dynamic berth allocation problem for a container port. *Transportation Research Part B: Methodological.*

Imai, A., Nishimura, E., and Papadimitriou, S. (2003a). Berth allocation with service priority. *Transportation Research Part B: Methodological.*

Imai, A., Sun, X., Nishimura, E., and Papadimitriou, S. (2003b). Berth allocation in a container port: using a continuous location space approach. *Transportation Research Part B: Methodological.*

Kim, K. H. and Moon, K. C. (2001). Berth scheduling by simulated annealing. *Transportation Research Part B: Methodological.*

Lai, K. and Shih, K. (1992). A study of container berth allocation. *Journal of Advanced Transportation.*

Liu, J., Wan, Y., and Wang, L. (2005). Quay crane scheduling at container terminals to minimize the maximum relative tardiness of vessel departures.

Meisel, F. and Bierwirth, C. (2007). Heuristic for the integration of crane productivity in the berth allocation problem. *Transportation Research Part E: Logistics and Transportation Review.*

Nishimura, E., Imai, A., and Papadimitriou, S. (2001). Berth allocation planning in the public berth system by genetic algorithms. *European Journal of Operational Research.*

Appendix

Table 8 shows all the functions used to generate all the results in this paper. Along with these functions, there are also 61 matrices. The matrix 'general_char.mat' is the matrix that is can also be found in Table 2. These are the general characteristics of the vessels once the type is known. Furthermore, the matrices of the 60 instances are also included. This is because the letters indicating the type is replaced with the following number; feeder = 1, medium = 2, jumbo = 3.

The four functions that should be called in the command window to run the code are:

Load_all_Heuristics
Load_CH_extension
Load_CH_extfreeQC
min_cost

At the beginning of these four codes there are three lines where the instance that needs to be run can be indicated. These need to be adjusted to the instance that you would want to run.

To further clarify where some functions are called. Figure 3 summarizes which functions are called within which functions.

Function	Description
assign_number	Used in the tabu search. Assigns a number to a priority list, the number is then added to the tabu list so to make it easier to check whether a priority list is in the tabu list.
check_QC_available	Used in construction_heuristic_extension. Checks how many QCs can reach the incoming vessel it it berths at position bi.
Construction_Heuristic	Assigns the vessels according to the construction heuristic.
Construction_Heuristic_extension	Assigns the vessels according to the construction heuristic, but restricts the QCs to only move along the rail.
Load_all_Heuristics	Calls the 5 (meta-)heuristics. Gives the total service cost and the calculation time for all 5 methods.
Load_CH_extension	For all 10 QCs, sets the QC still at all possible 91 position it can stand still for maintenance. Then calls the Construction_Heuristic_extension to calculate the total service time.
Load_CH_extfreeQC	Calls the Construction_Heuristic_extension to calculate the total service cost and running time for when the QCs can restricted in their movement along the rail.
Load_CH_LR	Calls Construction_Heuristic and Local_Refinements to calculate their total service cost and running time for a FCFS priority list
Load_instance	Creates the x-matrix that contains all the information of the vessels in the instance.
Local_Refinements	Assigns the vessels according to the construction heuristic with local refinements
makebi	Used in CH, LR and CH_extension, create a vector that goes through all the bi positions in the order $[b_i^0, b_i^0 + 1, b_i^0 - 1, ..., 0, ..., L - l_i]$
makePLmatrix	Used in the tabu search. Makes a matrix of all priority lists that are in the neighbourhood.
makesi	Used in CH, LR and CH_extension, create a vector that goes through all the si positions in the order $[ETA_i, ETA_i + 1, ETA_i - 1, ..., EST_i, ..., H]$
min_cost	Calculates the lower bound for the instance that is indicated in the code
MoveAndAssignCranes	Used in Construction_heuristic_extension. Once the position is chosen where the vessel will berth, this code will assign the QCs to the vessel and move the other QCs so that they are evenly spread next to the vessel.
QC_Assignment	Used in CH and LR. For a given berthing position and arrival time, checks if the vessel can be placed (is feasible) and if so checks the cost.
QC_Assignment_extension	Used in Construction_Heuristic_extension. For a given berthing position and arrival time, checks if the vessels can be placed (is feasible) considering the restriction of movement of the QCs along the rail. If feasible, it also calculates the service cost.
Simulated_Annealing	Assigns the vessels according to the simulated annealing meta-heuristic
Squeaky_Wheel_Optimization	Assigns the vessels according the the squeaky wheel optimisation meta-heuristic.
Tabu_Search	Assigns the vessels according the the tabu search meta-heuristic.

Table 8
A summary of the content of the functions.

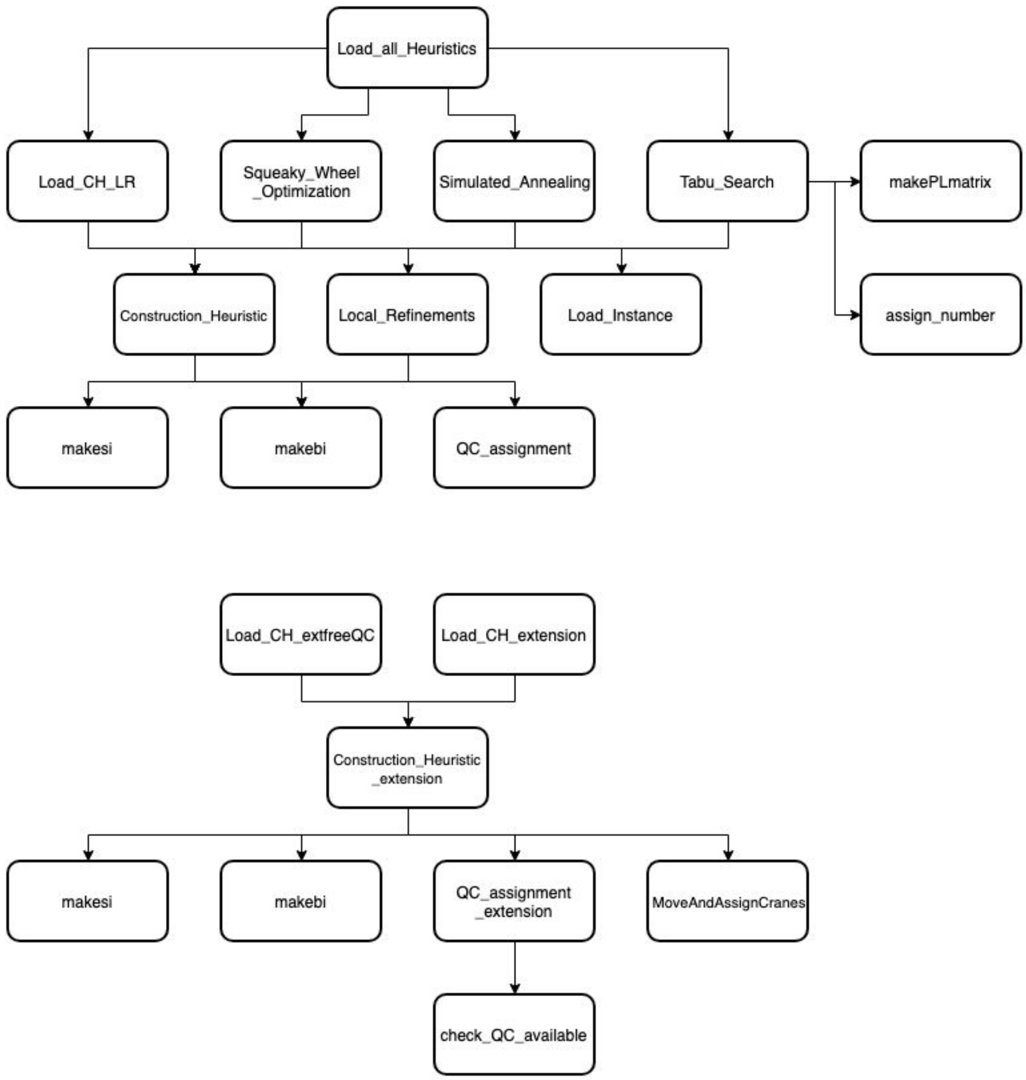

Figure 3
A flowchart of which functions are called within which functions.