TOWARDS THE INTEGRATION OF BERTH ALLOCATION AND CONTAINER STACKING PROBLEMS IN MARITIM CONTAINER TERMINALS

Miguel A. Salido[(a)], Mario Rodriguez-Molins[(b)], Federico Barber[(c)]

[(a), (b), (c)] Institute of Control Systems and Industrial Computing (Technical Univ. of Valencia, Spain)

[(a)]msalido@dsic.upv.es, [(b)]mrodriguez@dsic.upv.es, [(c)]fbarber@dsic.upv.es

ABSTRACT
Services provided by the container terminals are being further optimized due to the competitiveness between the different terminals. In this research, the combined problem of Container Stacking and Berth Allocation in container terminal's management is investigated. On the one hand, each container allocated in the yard should be easily accessible before vessel's arrival (demanded by terminal operators). On the other hand, an immediate berthing is expected for each incoming vessel (by ocean carriers). Thereby, we present an artificial intelligence based-integrated system to relate these problems. Berth Allocation Problem is solved by a metaheuristic algorithm which generates an optimized order of vessels to be moored; and we develop a domain-oriented heuristic planner to give a sequence of movements to allocate containers in the appropriate place according a given berth ordering of vessels. Through these optimized solutions together with the developed system, terminal operators can be assisted to decide the most appropriated solution in each particular scenario.

Keywords: Berth Allocation, Container Reshuffling, planning and scheduling, decision support system.

1. INTRODUCTION

During the last decade, worldwide container transportation has grown considerable, being the Container Terminals one of the key factors within global logistic network. This growth has rise to a more exhaustive analysis to ensure reliability, delivery dates or handling times as well as to increase container throughput from quay to landside and vice versa, etc. (Henesey, 2006). (Stahlbock and Voβ, 2008) provides a survey of the issues which must be optimized in Container Terminals.

Container terminals are mainly interested in reducing the berthing time of vessels. This objective is dependent on different interrelated problems: berth allocation, yard-side operation, storage operation and gatehouse operation. In the literature, these problems are managed independently of others due to their exponential complexity. However, these problems are clearly interrelated so that an optimized solution of one of them may not lead to a good solution in another.

The problems we take into account in this paper are Berth Allocation Problem (BAP) and Container Stacking Problem (CStackP) (see Figure 1). BAP consists of assigning a berthing position, a berthing time and cranes to incoming vessels under several constraints and priorities (length and depth vessels, number of containers, distance from storage yard blocks, etc.). On the other hand, CStackP arises when a vessels berth, since export containers to be loaded in this vessel should be on top of the stacks of the container yard. Thereby, CStackP consists on avoiding unnecessary movements of the yard-crane relocating containers at the time of loading. The relationship among these two problems is very clear since an optimal berth allocation plan may generate a large amount of relocations for export containers given a yard-state. However, a suboptimal berth allocation plan could require fewer movements given the same yard-state.

In this paper, we develop a system to optimize these two problems integration by means of a set of intelligent techniques for solving each one of them in order to achieve a mixed-solution. To this end, we present a heuristically-guided planner for generating a rehandling-free intra-block remarshaling plan for container yards. Furthermore, we introduce a meta-heuristic approach for solving the BAP as an independent problem. Finally, we integrate optimization of both problems. Terminal operator should ultimate decide which solution is the most appropriate one in relation to a multi-objective function: to minimize the waiting times of vessels and to minimize the amount of relocations of containers.

2. A DOMAIN-BASED PLANNER FOR THE CONTAINER STACKING PROBLEM.

In Container Terminals, almost all the operations are related to the container yards, where containers are stacked on top of each other awaiting further transport. A container yard is composed of several blocks, each one of them among 20 or 30 yard-bays (Figure 2). Each yard-bay contains multiple (usually 6) rows (or stacks) and each row has a maximum allowed tier (usually 4 or 5 tiers for full containers).

Loading and offloading containers on the stack is performed by cranes following a 'last-in, first-out' (LIFO) criteria. Containers are stacked in the order they arrive. However, in the loading process of vessels, to access a container which is not at the top of its pile, those above it must be relocated. This remarshaling process is required since the stacking order depends on the order in which ships unload or containers have been stacked. Furthermore, this process also reduces the productivity of cranes and its optimization would minimize the moves required.

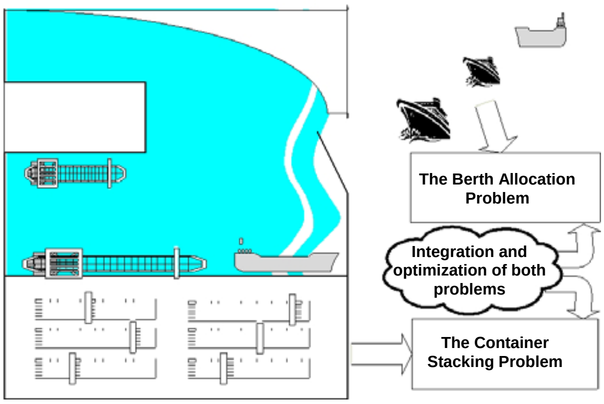


Figure 1. Integrated Remarshaling and Berthing problems in Maritime Terminals.

For safety reasons, it is usually prohibited to move the gantry crane while carrying a container (Lee and Hsu, 2007), therefore these movements only take place in the same yard-bay. In addition, there exist a set of hard/soft constraints regarding container moves or locations where can be stacked, for example, small differences in height of adjacent yard-bays, dangerous containers must be allocated separately by maintaining a minimum distance, etc.

The CStackP is a well known NP-complete combinatorial optimization problem and different approaches has been proposed (Park et al., 2009; Kim and Hong, 2006; etc.). In (Salido et al., 2009), a planning system for remarshaling processes was proposed. This system obtains the optimized plan of reshuffles of containers in order to allocate all selected containers at the top of the stacks, or under another selected containers, in such a way that no reshuffles will be needed to load these outgoing containers.

The proposed planner was specified by means of the standard Planning Domain Definition Language (PDDL, Ghallab, 1998) and it was developed on the well-known domain-independent planner *MetricFF* (Hoffmann, 2003). The developed domain file contains the common features of the problem domain: (i) the domain objects: containers and rows, (ii) the relations among them (propositions), (iii) allowed moves to change the status of the problem (actions), and (i) the goal: each export container must be allocated at the top of the stacks or under other export containers. The problem file describes each particular instance: (i) the initial layout of the containers in the yard (Initial state), (ii) the export containers (goal), and (iii) the function to optimize (minimizing the number of relocation movements).

More details can be seen in (Salido et al., 2009) and as it was anticipated, the Metric-FF-based initial planner was improved by integrating a domain-dependent heuristic (H1) in order to achieve efficiency. (Salido et al., 2009a). H1 computes an estimator of the number of container movements that must be carried out to reach a goal state, which it is used to guide search of solutions.

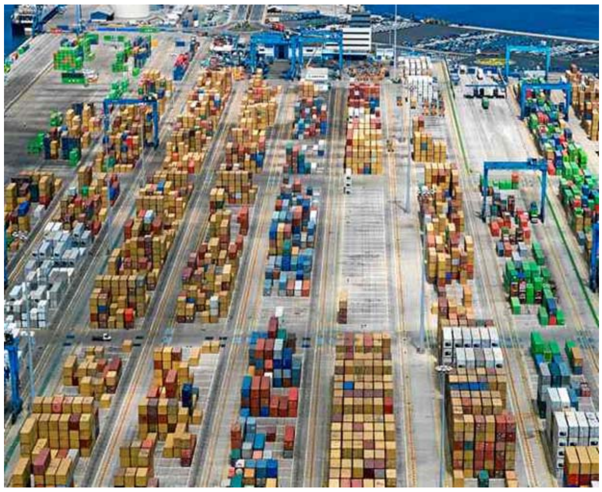


Figure 2. A container yard (left, Puerto de la Luz Terminal).

Moreover, new constrains and optimization criteria have been included in order to take into account real-world requirements:

1. Reducing distance of the goal containers to the cargo side.
2. Increasing the range of the move actions set for the cranes allowing moving a container to 5th tier.
3. Balancing the number of stacked containers within the same bay in order to avoid sinks.

The improved planner can manage a full container yard. The container yard is decomposed in yard-bays, so that the problem is distributed into a set of subproblems. Thus, each yard-bay generates a subproblem. However, containers of different yard-bays must satisfy a set of constraints among them. Therefore, subproblems are sequentially solved, so that each subproblem (yard-bay) takes into account the set of constraints with previously solved subproblems. This decomposition requires taking into account these new added constraints. With these new added constraint and criteria, the developed planner can solve more real-world based problems:

1. Balancing contiguous yard-bays: rows of adjacent yard-bays must be balanced in order to avoid sinks inter yard-bays (CB).
2. Dangerous containers must maintain a minimum security (Euclidian) distance among them (DC).

3. THE BERTH ALLOCATION PROBLEM

The BAP is one of the major problems directly related to productivity in the management of container terminals. Several models are usually considered (Theofanis et al., 2009):

- All vessels to be served are already in the port queue at the time that scheduling begins (static BAP).
- All vessels to be scheduled have not yet arrived but their arrival times are known (dynamic BAP)
- The quay is viewed as a finite set of berths, and each berth is described by fixed-length segments (Discrete BAP).
- Vessels can berth anywhere along the quay (Continuous BAP)

The objective in BAP is to obtain an optimal distribution of the docks and cranes to vessels waiting to berth. Thus, this problem could be considered as a special kind of machine scheduling problem, with specific constrains (length and depth of vessels, ensure a correct order for vessels that exchange containers, assuring departing times, etc.) and optimization criteria (priorities, minimization of waiting and staying times of vessels, satisfaction on order of berthing, minimizing cranes moves, degree of deviation from a pre-determined service priority, etc.).

The First-Come-First-Served (FCFS) rule can be used to obtain an upper bound of the function cost in BAP (Lai and Shih, 1992). On the other hand, several methods have been proposed for solving BAP. Usually, these methods are based on heuristic (Guan and Cheung, 2004) or metaheuristic (Cordeau et al., 2005), (Cheong et al., 2009), etc., approaches. In (Theofanis et al., 2009), a comparative analysis is provided.

Our approach takes into account as a whole the Quay Crane Assignment Problem (QCAP) and the Berth Allocation Problem (BAP) and we use the metaheuristic Greedy Randomized Adaptive Search Procedure, also known as GRASP (Feo and Resende, 1995) in order to obtain optimized solutions efficiently.

Next, let's introduce the notation used in this section: $a(V_i)$ indicates the arrival time of the vessel V_i at port; $m(V_i)$ is the moored time of V_i. At this time, all constraints must hold; $c(V_i)$ is the number of required movements to load and unload containers of V_i; $q(V_i)$ is the number of assigned Quay Cranes (QC) to V_i. The maximum number of assigned QC by vessel depends on its length since a security distance is required. Let's assume that the number of QC does not vary along all the moored time. Thus, the handling time of V_i is given by (where MovsQC is the QC's moves per unit time): $(c(V_i) / (q(V_i) \times MovsQC))$; $d(V_i)$ indicates the departure time of V_i, which depends on $m(V_i)$, $c(V_i)$, and $q(V_i)$; $w(V_i)$ shows the waiting time of V_i from it arrives at port until it moors ($w(V_i) = m(V_i) - a(V_i)$); $l(V_i)$ denotes the length of V_i. There is a distance security between two moored ships: let's assume 5% of their lengths; and, the vessels' priority is $pr(V_i)$.

In order to simplify the problem, let's assume that mooring and unmooring does not consume time and every vessel has a draft lower or equal than the quay. In each case, simultaneous berthing is allowed.

The goal of the BAP is to allocate each vessel according existing constraints and to minimize the total weighted waiting time of vessels:

$$T_w = \sum_i w(V_i)^\gamma \times pr(V_i)$$

The parameter γ ($\gamma \geq 1$) prevents lower priority vessels are systematically delayed. Note that this objective function is different to the classical tardiness concept in scheduling.

3.1. A meta-heuristic method for BAP

BAP can be solved through different methods. We have developed three different methods to solve it; two of them are direct but inefficient solutions. Firstly, we applied the simplest solution, following the FCFS criteria: $\forall i, m(V_i) \leq m(V_{i+1})$. A vessel can be allocated at time t when there is no vessel moored in the berth or there are available quay length and cranes at this time t (Algorithm 1).

```
Data: V: set of ordered incoming vessels; b: state of the berth
Result: Sequence for V
V_last ← ∅;
V_m ← ∅;
foreach V_i ∈ V do
    t ← max(e(V_last), a(V_i));
    inst ← insertVessel(V_i, t, b);
    if !inst then
        T ← d(V_j) | V_j ∈ V_m ∧ d(V_j) > t;
        while t_k ∈ T ∧ !inst do
            inst ← insertVessel(V_i, t_k, b);
        end
    end
    update(b) ;    /* state of the berth b */
    V_last ← V_i;
    V_m ← V_m ∪ V_i;
end
```

Algorithm 1: Allocating vessels using FCFS policy.

```
Data: V_i
Result: V_i could moor
if empty(b) then
    m(V_i) ← a(V_i);
    q(V_i) ← min(maxQC_V, l(V_i)/distQC)
    d(V_i) ← m(V_i) + c(V_i) / (q(V_i) × movsQC);
    return true;
else
    freeQC ← QC(b) − Σ q(V_i) | t ≥ a(V_i) ∧ t < d(V_i);
    freeL ← l(b) − Σ l(V_i) | t ≥ a(V_i) ∧ t < d(V_i);
    if freeQC > 0 ∧ l(V_i) <= freeL then
        q(V_i) ← min(freeQC, l(V_i)/secQC) m(V_i) ← t;
        d(V_i) ← t + c(V_i) / (q(V_i) × movsQC);
        if checkDisponibility(V_i, m(V_i), d(V_i)) then
            return true;
        else
            return false;
        end
    else
        return false;
    end
end
```

Algorithm 2: Function insertVessel. Allocating one vessel in the berth at time t.

We also have implemented a complete search algorithm for obtain the best (optimal) mooring order of vessels: the lowest T_w (lower bound of the function cost). This algorithm uses the *Function insertVessel* (Algorithm 2) to know whether a vessel can be allocated at time t (the required data are: V_i: Vessel for allocating; t: actual time; b: state of the berth at time t).

However, with a complete search, only a limited number of vessels can be taken into account since search space grows exponentially. Therefore, we developed a meta-heuristic GRASP algorithm for berth allocation (Algorithm 3). This is a randomly-biased multistart method to obtain optimized solutions of hard combinatorial problems in a very efficient way. The parameter ρ (0≤ρ≤1) allows tuning of search randomization.

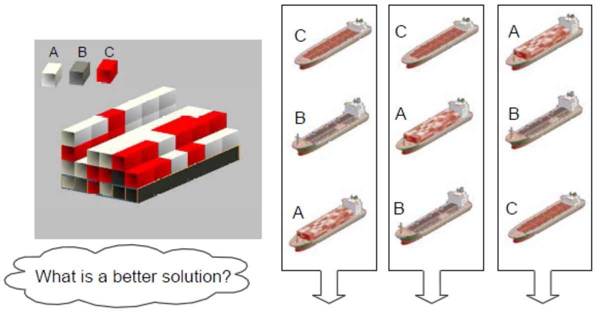
```
Data: ρ factor; E elements; b: state of the berth at t0
Result: Solution s
/* current time at which algorithm is performed                    */
T_assig ← t0;
V ← {V_i};
T_exit ← {};
while V ≠ ∅ do
  /* V_i which can be allocated in available quay                  */
  SV ←
  {V_i}|checkDisponibility(V_i, max(T_assig, a(V_i)), d(V_i)) = True;
  if SV ≠ ∅ then
    T_assig ← first(T_exit) ;        // Time of the next exiting
    vessel
  else
    foreach V_i ∈ SV do
      Allocate(V_i) ;                  // Procedure allocate V_i
      Cost(V_i) ← 0;
      foreach V_j ∈ SV|V_i ≠ V_j ∧
      max(T_assig, a(V_i)), max(T_assig, a(V_i)) + d(V_i) ∩
      max(T_assig, a(V_i)), max(T_a ssig, a(V_i)) + d(V_i) ≠ ∅ do
        Cost(V_i) ← Cost(V_i)+
        pr(V_j)((max(T_assig, a(V_j)) + d(V_j)) −
        max(T_assig, a(V_j)))^γ;
      end
      MaxCost ← max_{V_i∈SV}(Cost(V_i));
      MinCost ← min_{V_i∈SV}(Cost(V_i));
      V_assig ← {V_i ∈ SV|Cost(V_i) ∈
      {MinCost, MinCost + ρ(MaxCost − MinCost)}};
      V_k ← random(V_assig);
      insertVessel(V_k, max(T_assig, a(V_k)), b);
      update(b) ;          // update the state of the berth
      remove(V_k, SV);
      T_exit ← T_exit ∪ d(V_k);
    end
  end
end
;                   // All waiting vessels have been allocated
```

Algorithm 3. Allocating Vessels using GRASP metaheuristic.

4. AN INTEGRATED APPROACH FOR CONTAINER STACKING AND BERTH ALLOCATION PROBLEMS

In previous sections, BAP and CStackP have been studied and solved separately through different intelligent techniques in an efficient way. However, no systems have been developed to relate and optimize both problems in an integrated way. Only some works integrate the BAP with the QCAP, for instance (Giallombardo et al., 2010) which follows to minimize the yard-related house-keeping costs generated by the flows of containers exchanged between vessels. However, there also exists a relationship between the optimization of maritime and terminal-sides operations (BAP, QCAP, container stacking problem, etc.).

Figure 3 shows an example of three berth allocation plans and a block of containers to be loaded in the vessels. Containers of type A, B and C must be loaded in vessels A, B and C, respectively. In the first berth allocation plan the order of vessel is A-B-C, the waiting time for this plan is 205 time units and the number of reshuffles needed to allocate the white containers at the top of the stacks is 110. The second berth allocation plan is B-A-C. In this case the waiting time for this plan is 245 time units and the number of reshuffles is 260. Finally, the third berth allocation plan is C-B-A, the waiting time for this plan is 139 time units and the number of reshuffles is 450. The question is straightforward: what is a better solution? A solution that optimizes the BAP problem could not be the more appropriate for the CStackP (and vice versa).



Waiting Time	205 Time Units	245 Time Units	139 Time Units
N. Reshuffles	110	260	450

Figure 3. Three different plans for the BAP: What is better?

Given a waiting queue of vessels to be allocated and a given state of the containers in the container yard, each solution for the BAP ($SBAP_i$: a feasible sequence of mooring), requires a different number of container's re-locations in the associated CStackP solution ($SCStackP_i$) in order to put on top the containers to be loaded according to the order of berthing. We can associate a cost to each $SBAP_i$ related to the total weighted waiting time of vessels of this berthing order (T_w). Likewise, we can associate a cost to each $SCStackP_i$ as the number of required container relocations. Therefore, we can qualify the optimality of each global solution (Sol_i) of BAP and CStackP as a lineal combination of the quality of each partial solution:

$$Cost(Sol_i)= \alpha*Cost(SBAP_i) + \beta*Cost(SCStackP_i) \quad (1)$$

The best decision will depend on the policy of each maritime terminal (α and β parameters). Thus, by combining the planning and berth allocation solutions (they obtained by the systems developed in Sections 3 and 4), we can assess the cost to the global solution Sol_i.

The applied method is: First, both the BAP and the CStackP data are loaded in the integrated system. Next, the BAP is solved to achieve a solution ($SBAP_i$) based on their constraints and criteria. Then, the CStackP is solved by taken into account the berthing order of vessels obtained in $SBAP_i$. The CStackP planner is applied sequentially for each vessel in $SBAP_i$, according the state of the container yard in each moment. Thus, the optimized remarshaling plan for the berthing order of vessels of $SBAP_i$ is obtained ($SCStackP_i$). After this step, the cost of the global solution (Sol_i) can be calculated by using the previous expression (1). By iterating this integrated process, the operators can obtain a qualification cost of each feasible Sol_i, as well as the best global solution (Sol_i), according the given α and β parameters. A branch and bound method has been also applied in the integrated search of the best global solution (Sol_i), so that the search can be pruned each time the current solution does not improve the best solution found until this moment.

5. EVALUATION.

In this section, we analyze the performance of the algorithms developed in the paper. The experiments were performed on random instances. For the CStackP, containers are randomly distributed in blocks of 20 yard-bays, each one with six stacks of 4 tiers. A random instance of a yard-bay is characterized by the tuple < *n, s* >, where '*n*' is the number of containers and '*s*' (s≤n) is the number of selected containers in the yard-bay. A random instance for the BAP has '*k*' vessels with an arrival exponential distribution with vessel's data randomly fixed (lengths, drafts, moves and priorities).

Table 1: Performance of real-world criteria in CStackPs.

	Metric-FF Planner	H1	CB	DC	CB + DC
Reshuffles	3.98	3.60	5.68	4.30	6.53
Sinks	24.33	32.67	0	33.33	0
Non-Safe Dangerous	15.33	7.67	8.00	0	0

For the developed planning system to solve CStackPs (Section 2), Table 1 shows the performance of the introduced real-world criteria. These experiments were performed on instances < 15, 4 >. The results shown in Table 1 are the average of the best solutions found in 10 seconds and they represent the average number of reshuffles, the average number of sinks generated along the block, and the average number of unsatisfied dangerous containers. It can be observed that H1 outperforms the general purpose Metric-FF-based initial planner in the number of reshuffles and the new introduced criteria (CB, DC) avoid undesired situations.

Table 2: Computing time elapsed (seconds) for BAP.

No. Vessels	Complete search	GRASP
5	< 1	1
10	112	8
12	11830	10
13	57462	12
15	-	15
20	-	30

Table 2 shows the computational times (in seconds) required for solving BAP by using a complete search against the GRASP method with 1000 iterations. As it can be observed, complete search is impracticable from 12 vessels (more than 3 hours). However, the GRASP method takes around 30 seconds to solve a schedule of 20 vessels.

Table 3: Total waiting time elapsed.

No. Vessels	FCFS	CS
5 (separate arrival times)	73	46
10 (separate arrival times)	256	136
5 (closest arrival times)	117	80
10 (closest arrival times)	586	351

Table 3 shows the average waiting times using FCFS and Complete Search (CS) methods described for the BAP, with two different inter-arrival distributions (temporal separation among arriving vessels). Through these data, it is demonstrated that FCFS criteria results a schedule which is far away from the best one (CS).

Using as minimization function the total weighted waiting time (T_w), Figure 4 shows the results given by the FCFS criteria, and the GRASP procedure (with 1000 iterations) respect to the value of ρ. The optimum value is ρ=0,3, which indicates the suitability of the cost function used in the GRASP procedure (Algorithm 3). A total of 20 vessels are allocated, with two different inter-arrival distributions (separate and closest arrival times) among them.

As it was expected, the GRASP procedure obtains a lower T_w than the FCFS criteria. It is also remarkable that using GRASP is more profitable when the inter-arrival distribution of the vessels is closer. It is not possible to know the optimal T_w due to the exponential computational time required by a complete search with 20 vessels.

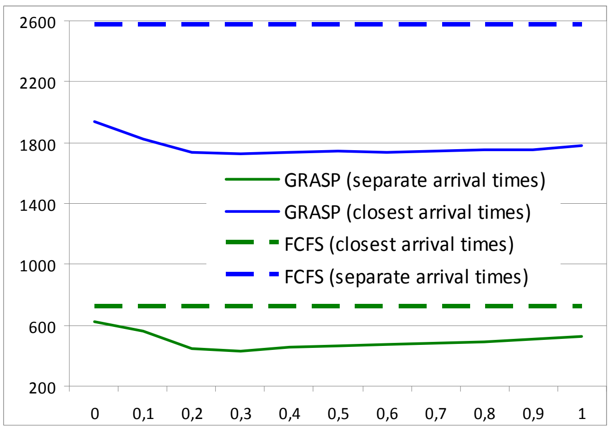


Figure 4: Weighted waiting time (Tw) with FCFS and GRASP procedures.

Finally, Figure 5 shows the combined function cost Cost(Sol_i), introduced in (1) which relates: (i) The normalized total weighted waiting time of vessels, Cost($SBAP_i$), and (ii) the number of its required container relocations, Cost(SCStackP$_i$); for ten different scenarios. In each one of this ten cases, the arrival times and data of vessels, as well as the initial state of the container yard, have been randomly generated. Figure 5 represents the combined function cost, Cost(Sol$_i$) with three different weights of the parameters α and β. We can see that better (or worst) berthing orders can require larger (or smaller) number of container relocations.

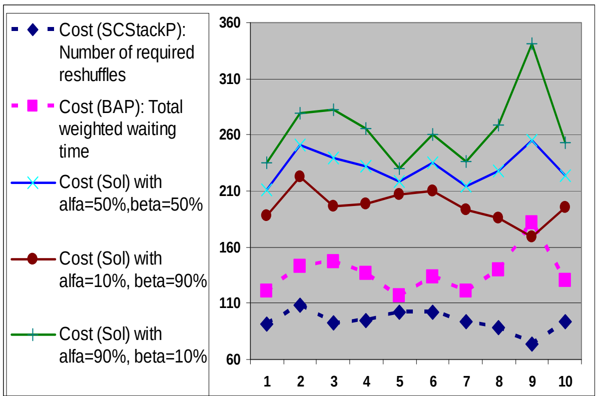


Figure 5. Relating the costs of BAP and CStackP.

6. CONCLUSIONS

This paper presents an efficient way of solving Berth Allocation and Container Stacking Problem, respectively. The former is solved by means of a metaheuristic called GRASP; the latter is solved by an improved planning system used to obtain optimized plans for remarshaling process. Furthermore, an integrated system is studied to provide mixed-solutions for both problems. This system is also oriented to assist to the terminal operators' decision between different feasible alternatives. Several evaluations on randomized scenarios have been performed and we can conclude that a better ordering of vessels does not imply a minimum number of container relocations. As future work, we plan improve GRASP method and adequate the parameters (α, β and γ) to real-world practical decisions and expert knowledge. Then, the developed system, as a computer-based aid system, could assist container terminal's operator to simulate, evaluate and compare different feasible alternatives.

ACKNOWLEDGMENTS

This work has been partially supported by the research projects TIN2007-67943-C02-01 (MEC, Spain-FEDER) and P19/08 (M. Fomento, Spain-FEDER).

REFERENCES

Cheong, C.Y., K.C. Tan and D.K. Liu (2009). Solving the berth allocation problem with service priority via multi- objective optimization. In: *Computational Intell. in Scheduling, 2009. CI-Sched '09. IEEE Symposium on*. pp. 95 –102.

Cordeau, J.F., G. Laporte, P. Legato and L. Moccia (2005). Models and tabu search heuristics for the berth-allocation problem. *Transportation science* 39(4), 526–538.

Feo, T.A. and M.G.C. Resende (1995). Greedy randomized adaptive search procedures. *Journal of Global Optimization* 6(2), 109–133.

Ghallab, M., A. Howe, C. Knoblock, D. McDermott, A. Ram, M. Veloso, D. Weld and D. Wilkins (1998). PDDL - the planning domain definition language. *AIPS-98 Planning Committee*.

Giallombardo, Giovanni, Luigi Moccia, Matteo Salani and Ilaria Vacca (2010). Modeling and solving the tactical berth allocation problem. *Transportation Research Part B: Methodological* 44(2), 232 – 245.

Guan, Y. and R.K. Cheung (2004). The berth allocation problem: models and solution methods. *OR Spectrum* 26(1), 75–92.

Henesey, L. (2006). Overview of Transshipment Operations and Simulation. In: *MedTrade conference, Malta, April*. pp. 6–7.

Hoffmann, Jörg (2003). The metric-ff planning system: translating "ignoring delete lists" to numeric state variables. *J. Artif. Int. Res.* 20(1), 291–341.

Kim, K.H. and G.P. Hong (2006). A heuristic rule for relocating blocks. *Computers & Operations Research* 33(4), 940–954.

Lai, KK and K. Shih (1992). A study of container berth allocation. *Journal of Advanced Transportation* 26(1), 45–60.

Lee, Yusin and Nai-Yun Hsu (2007). An optimization model for the container pre-marshalling problem. *Computers & Operations Research* 34(11), 3295 – 3313.

Park, K., T. Park and K.R. Ryu (2009). Planning for remarshaling in an automated container terminal using cooperative coevolutionary algorithms. In: *Proceedings of the 2009 ACM symposium on Applied Computing*. ACM. pp. 1098–1105.

Salido, M., O. Sapena and F. Barber (2009). The Container Stacking Problem: an Artificial Intelligence Planning-Based Approach. Procc. Of *The Int. Workshop on Harbour, Maritime and Multimodal Logistics Modelling and Simulation* HMS'2009. pp:127-131 (2009)

Salido, M., O. Sapena, M. Rodriguez and F. Barber (2009*a*). A planning tool for minimizing reshuffles in containers terminals. *ICTAI 2009: 21st International Conference on Tools with Artificial Intelligence*.

Stahlbock, R. and S. Voß (2008). Operations research at container terminals: a literature update. *OR Spectrum* 30(1), 1–52.

Theofanis, S., M. Boile and M.M. Golias (2009). Container terminal berth planning. *Transportation Research Record: Journal of the Transportation Research Board* 2100(-1), 22–28.