

A Hybrid GRASP-VNS for Ship Routing and Scheduling Problem with Discretized Time Windows

Jesica de Armas^a, Eduardo Lalla-Ruiz^a, Christopher Expósito-Izquierdo^a,
Dario Landa-Silva^b, Belén Melián-Batista^a

^a*University of La Laguna, Dept. of Computer and Systems Engineering, 38271, Spain*

^b*School of Computer Science, The University of Nottingham, UK*

Abstract

This paper addresses the Ship Routing and Scheduling Problem with Discretized Time Windows. Being one of the most relevant and challenging problems faced by decision makers from shipping companies, this tramp shipping problem lies in determining the set of contracts that should be served by each ship and the time windows that ships should use to serve each contract, with the aim of minimizing total costs. The use of discretized time windows allows for the consideration of a broad variety of features and practical constraints in a simple way. In order to solve this problem we propose a hybridization of a Greedy Randomized Adaptive Search Procedure and a Variable Neighborhood Search, which improves previous heuristics results found in literature and requires very short computational time. Moreover, this algorithm is able to achieve the optimal results for many instances, demonstrating its good performance.

Keywords: Ship Routing and Scheduling Problem, Tramp Shipping, GRASP, Variable Neighbourhood Search

1. Introduction

The most important mode of transport for international trade is seaborne shipping. An estimated 80 per cent of world trade is carried by sea [43]. It means that, compared to other modes of freight transportation, ships are far superior for moving large volumes over long distances. Due to the increasing

Email addresses: jdearmas@ull.es (Jesica de Armas), elalla@ull.es (Eduardo Lalla-Ruiz), cexposit@ull.es (Christopher Expósito-Izquierdo), dario.landasilva@nottingham.ac.uk (Dario Landa-Silva), mbmelian@ull.es (Belén Melián-Batista)

16 development of economies and globalization, the international trade is con-
17 tinuously rising, and as a consequent the sector of maritime transport has
18 also shown an enormous growth.

19 Container ships require a huge capital investment and very high daily
20 operating costs. Investment in a ship may range in the millions and operating
21 costs in the thousands dollars a day. There are three main types of costs in
22 seaborne shipping: capital and depreciation, running, and operating costs.
23 Capital and depreciation costs are related to the loss of ships market value
24 respect to the initial investment. Running costs are usually fixed and include
25 maintenance, insurance, crew salaries, and overhead costs, among others.
26 Operating costs are directly related to ships daily operations, and include
27 fuel consumption, port and customs expenses, tolls at canals, etc. These
28 latter costs depend on characteristics like travel distance, navigation speed,
29 and maritime routes. Therefore, capital and depreciation costs, and running
30 costs are not usually expenses that can be subject to optimization as a result
31 of improvements in routing, but the operating costs can be optimized through
32 better routing as it is the aim in this work. Accordingly, good scheduling is
33 of economical essence in this increasingly competitive area.

34 In this regard, Gatica and Miranda [15] focus on optimally solving a ship
35 routing and scheduling problem with a heterogeneous tramp fleet. They
36 propose a network-based model in which discretized time windows for pick-
37 ing and delivering cargoes are defined. Discretized time windows are just
38 time instants in which these picking and delivering cargoes can be carried
39 out. This allows to consider a broad variety of features and practical con-
40 straints by simply adding/deleting arcs or modifying the corresponding cost
41 parameters, which has the advantage of preserving the network structure. In
42 particular, they consider problems in which navigation speed can be used to
43 control fuel consumption, which may have a significant impact in the quality
44 of the solution, since fuel consumption follows an approximately cubic func-
45 tion of speed [37]. They solved the model by means of the general-purpose
46 solver, CPLEX¹. Numerical results show that the model presents a much
47 better trade-off between solution quality and computing time than a similar
48 constant-speed continuous model. Recently, in order to obtain quality results
49 for real-life-sized problems in less computational time, Castillo-Villar et al.
50 [7] develop a Variable Neighborhood Search (VNS) algorithm to solve this

¹<http://www-01.ibm.com/software/commerce/optimization/cplex-optimizer/>

51 problem. This VNS is very simple, since it is just a list of neighborhoods
52 sequentially explored. Results reported in that work show that the VNS
53 provides solutions with a gap between 6% and 7% to the optimal solutions.

54 Due to the fact that this is an operational problem (*i.e.*, it has to be solved
55 daily) and may be integrated with other related problems (Berth Allocation
56 Problem [4], Container Stowage Problem [3], etc.), it is important to solve
57 it quickly and provide results of the highest possible quality. Motivated by
58 that, this work presents a hybridization of a Greedy Randomized Adaptive
59 Search Procedure (GRASP) with a more complex and elaborate VNS to solve
60 the same problem with the aim of reducing the gap and obtaining results in
61 less computational time.

62 The remainder of this article is organized as follows. At first, Section 2
63 reviews research efforts from earlier studies that are related to the current
64 study. Then, in Section 3 the formulation of the problem is shortly out-
65 lined. In Section 4 the main features of the hybrid GRASP-VNS method
66 is described. Experimental tests are performed and results are discussed in
67 Section 5. Finally, in Section 6 the main conclusions extracted in this work
68 and some future work lines are summarized.

69 **2. Literature Review**

70 In the literature, the first works about ship routing arose in the 70s [1, 2],
71 but the first survey appeared more than ten years later [36]. Recent reviews
72 on ship routing problems can be found in works by Christiansen et al. [8, 9],
73 where literature contributions are classified.

74 The majority of papers about ship routing and scheduling problems focus
75 on the development of Mixed Integer Programming (MIP) models or heuris-
76 tics/metaheuristics methods to solve them. Fox and Herden [13] describe a
77 MIP model to schedule ships from ammonia processing plants to eight ports
78 in Australia. The objective is to minimize freight, discharge, and inventory
79 holding costs while taking into account the inventory, minimum discharge
80 tonnage, and ship capacity constraints. Moreover, using a MIP model, an
81 inventory routing problem with multiple products was analyzed by Ronen
82 [38] for liquid bulk oil cargo. Sherali et al. [40] present an aggregate MIP
83 model for the problem of transporting refined-oil products from three ports
84 in Kuwait to customers located in Europe, North America, and Japan. The
85 model considers the existence of alternative maritime routes. The authors
86 develop a reformulation of the MIP model and a set of valid inequalities that

87 allow them to design several algorithmic solution strategies.

88 In relation to heuristics and metaheuristics, Korsvik et al. [20] use a Tabu
89 Search algorithm which allows infeasible solutions with respect to ship ca-
90 pacity and time. Other works in the literature introduce specific concepts
91 in ship routing problems. Romero et al. [35] propose a GRASP and discuss
92 aspects related to data gathering and updating, which are particularly diffi-
93 cult in the context of ship routing. Lin and Liu [22] combine ship allocation,
94 routing and freight assignment in a particular kind of ship routing. Kosmas
95 and Vlachos [21] consider a cost function that depends on the wind speed
96 and its direction, as well as on the wave height and its direction, and solve
97 the problem using a Simulated Annealing algorithm.

98 Moreover, in the related literature, depending on the operation mode,
99 three kinds of ship routing problems can be distinguished: liner, industrial,
100 and tramp shipping. Liners [19, 25, 42] operate according to an agreed
101 itinerary and schedule similar to a bus line. In industrial shipping, the cargo
102 owner or shipper controls the ships. Industrial operators strive to minimize
103 the costs of shipping their cargoes. Tramp fleets engage in contracts to trans-
104 port specified (usually large) volumes of cargo between two ports within a
105 period of time. They engage in contracts to make one or several trips, each
106 trip having specified origin and destination ports and time windows for pick-
107 ing and delivering the cargo. Tramp is usually the operation mode selected
108 to transport liquid and dry commodities, or cargo involving a large number
109 of units. During the last few decades there has been a shift from industrial
110 to tramp shipping [8, 9].

111 Particularly, the literature on tramp shipping problems is quite sparse
112 and only a few papers tackle such problems. The reason for this lack of re-
113 search interest in this shipping sector is attributed to the historic existence
114 of a large number of small tramp shipping companies operating in the mar-
115 ket. However, more recently, increased demand and the tendency of larger
116 companies to outsource shipping of their cargoes has led to the growth of
117 small companies, the growth of the associated scheduling problems, and a
118 corresponding increase interest from researchers in this type of problems. A
119 tramp routing and scheduling problem was solved by Brønmo et al. [5], where
120 a multistart Local Search heuristic is developed. The proposed unified Tabu
121 Search heuristic by Korsvik et al. [20] also solves the specific tramp ship-
122 ping. In contrast to the procedure followed by Brønmo et al. [5], Malliappi
123 et al. [23] present a VNS heuristic, and the results show that this procedure
124 outperforms the previous heuristics. Norstad et al. [29] address this prob-

125 lem considering speed optimization and develop a multistart Local Search
126 heuristic to solve it

127 Additionally, as introduced above, Gatica and Miranda [15] develop a
128 network-based model for the Ship Routing and Scheduling Problem with Dis-
129 cretized Time Windows with a heterogeneous tramp fleet. The objective is
130 to minimize the total operating cost of serving a set of trip cargo contracts,
131 considering time window constraints at both the origin and destination of
132 cargoes. A distinctive aspect of their methodology is that time windows for
133 picking and delivering cargoes are discretized. This leaves room for including
134 a broad variety of features and practical constraints, such as navigation speed
135 to control fuel consumption. More specifically, they assume that pick-up and
136 delivery may start only at a finite set of time instances within the correspond-
137 ing time windows. In general (*i.e.* urban) vehicle routing, the only known
138 application of time discretization is for modelling time-dependent travel times
139 (time to traverse an arc depends on the time instance the travel starts). That
140 approach is followed, for example, by Ichoua et al. [18], Woensel et al. [44],
141 and Donati et al. [10]. Gatica and Miranda [15] demonstrate that numerical
142 results considering discretized time windows presents a much better trade-
143 off between solution quality and computational time than a similar constant
144 speed continuous model. Recently, Castillo-Villar et al. [7] developed a VNS
145 algorithm to solve this specific tramp shipping problem with discretized time
146 windows, obtaining quality results in less computational time than the nec-
147 essary for the initial MIP model implemented in CPLEX.

148 The main contribution of this paper is to propose a hybrid GRASP-
149 VNS algorithm that improves upon the results from Castillo-Villar et al. [7].
150 We have developed a more elaborated VNS with a more complex structure
151 for better exploration of the search space, which jointly with the proposed
152 GRASP as start method, allows to obtain a better performance. The im-
153 provement is on achieving smaller gap values than those reported by Castillo-
154 Villar et al. [7]. Our technique achieves results of higher quality in short
155 computation times. In this particular problem, the quality of results is very
156 important, since, as stated before, minimizing operating cost is of high rele-
157 vance in the competitive area of ship routing. Moreover, the faster the results
158 are obtained, the more agility will have the rest of processes that depend on
159 the pickup or delivery of ships cargoes, so that it is another challenge to
160 achieve.

161 3. Ship Routing and Scheduling Problem with Discretized Time 162 Windows

163 This section presents the description of the Ship Routing and Scheduling
164 Problem with Discretized Time Windows (hereinafter SRSPDTW). Firstly,
165 Section 3.1 introduces the details of the discretized modelling approach and
166 the characteristics of the problem. Secondly, Section 3.2 presents the math-
167 ematical model proposed by Gatica and Miranda [15] and used by Castillo-
168 Villar et al. [7], which is considered in this paper.

169 3.1. Problem Description

170 We consider the routing and scheduling problem for tramp shipping which
171 is composed of: (i) a fleet of ships; (ii) a set of cargo contracts that need to
172 be served; (iii) a set of time instants or discretized time windows at which
173 each contract can be served; and (iv) a set of links or arcs between time
174 instants of different contracts for each ship. Each of these arcs represents a
175 ship serving a contract at a time instant and then serving another contract
176 at a different time instant, all this with an associated cost. Each contract
177 is a single trip from one port to another, picking-up and delivering a cargo.
178 A contract must be served at one of the possible time instants that are also
179 called nodes. Therefore, the problem here presented consists of deciding the
180 set of contracts to be served by each ship and the chosen time instants, *i.e.*
181 selecting a set of arcs, with the aim of serving all contracts while minimizing
182 total relevant costs.

183 The fleet of ships is heterogeneous due to differences in capacity, speeds,
184 fuel consumption, etc. Although each ship can serve only one contract at a
185 time, there are incompatibilities between cargoes and ships or between ships
186 and ports, so that not all ships can serve all contracts. Furthermore, two
187 contracts may be incompatible with each other, *i.e.*, the corresponding trips
188 cannot be done consecutively by the same ship, unless a time delay (*e.g.* for
189 cleaning) or a third trip is placed between them.

190 It is important to notice that given a sequence of contracts to be served
191 by a single ship, an empty trip must take place from the delivery port of
192 each contract to the origin of the next contract in the sequence, unless these
193 two ports coincide. These empty trips represent a significant portion of total
194 avoidable cost and they are taken into account in this problem. Relevant
195 costs are mainly operating costs, but may also include other kinds of costs
196 as long as they can be associated with individual trips.

197 One of the most important costs corresponds to the fuel consumption
 198 expenses. Since fuel consumption depends on navigation speed, controlling
 199 the speed impacts not only on the travel time, but also on the travel costs.
 200 In this work, a network-based model is used, and it allows for the consid-
 201 eration of navigation speed and a broad variety of features and practical
 202 constraints by simply adding/deleting arcs between contracts or modifying
 203 the corresponding cost parameters, which has the advantage of preserving
 204 the network structure. This flexibility arises from the discretization of the
 205 time windows, which allows for both, the feasibility (existence) and the cost
 206 parameter of each potential arc, to be determined outside the model.

207 3.2. Mathematical Formulation

208 As stated above, the discretized modeling approach used in Gatica and
 209 Miranda [15] and Castillo-Villar et al. [7] has been adopted. In order to make
 210 this work self-contained, the model is explained below.

211 The SRSPDTW can be defined as follows. Let $G = (V, A)$ be a directed
 212 graph, where V is the node set and A is the arc set. Each node $i \in V$ represent
 213 a time instant and the contract associated with that node is represented by
 214 $n(i)$. On the other hand, each contract $n(i)$ has a set of associate nodes
 215 $D_{n(i)}$, *i.e.*, the set of possible starting times for trip $n(i)$. In SRSPDTW, the
 216 ships are indexed by means of $k = 1, 2, \dots, B$, where B is the number of
 217 available ships. Each $arc(i, j, k)$ represents the service of contracts $n(i)$ and
 218 $n(j)$ consecutively by ship k . The arc is included in the network if both, the
 219 trips and the ship, are compatible, and if it is feasible for ship k to begin
 220 service of contract $n(i)$ at the time instance represented by node i , make the
 221 empty trip from the destination port of contract $n(i)$ to the origin of contract
 222 $n(j)$, and be available to begin service of contract $n(j)$ at the time instance
 223 associated with node j .

224 For each arc, the cost parameter c_{ijk} represents the total minimal cost
 225 when the ship delivers contract $n(i)$ immediately followed by contract $n(j)$.
 226 To complete the network, a fictitious node 0 is created to represent the source
 227 and destination of all ships (ports that can be different). For each ship k and
 228 node i , if contract $n(i)$ is compatible with ship k , both an $arc(0, i, k)$ and an
 229 $arc(i, 0, k)$ also exist. Cost c_{0ik} is calculated based on the real initial position
 230 of ship k , and cost c_{i0k} represents the cost incurred if ship k serves contract
 231 $n(i)$ and must go to a final destination port.

232 The mathematical formulation of the problem from Gatica and Miranda
 233 [15] is as follows:

$$\text{minimize } \sum_{(i,j,k) \in A} c_{ijk} \cdot x_{ijk} \quad (1)$$

234

$$\text{s.t. : } \sum_{i \in V / (0,i,k) \in A} x_{0ik} \leq 1 \quad k = 1, 2, \dots, B \quad (2)$$

235

$$\sum_{(i,j,k) \in A / j \in D_n} x_{ijk} = 1 \quad n = 1, 2, \dots, N \quad (3)$$

$$\sum_{i \in V / (i,j,k) \in A} x_{ijk} = \sum_{l \in V / (j,l,k) \in A} x_{jlk} \quad j \in V, \quad k = 1, 2, \dots, B \quad (4)$$

$$x_{ijk} \in \{0, 1\} \quad (i, j, k) \in A \quad (5)$$

236

237

238

239

where N is the number of contracts to be served, V is the set of nodes in the network, D_n is the set of nodes associated with contract n (i.e., set of possible starting times for trip n), A is the set of arcs in the network, c_{ijk} is the cost of $\text{arc}(i, j, k)$, and:

$$x_{ijk} = \begin{cases} 1 & \text{if } \text{arc}(i, j, k) \text{ is part of the solution} \\ 0 & \text{otherwise} \end{cases} \quad (6)$$

240

241

242

243

244

Selecting $\text{arc}(i, j, k)$ as part of the solution ($x_{ijk} = 1$) implies that ship k will serve contract $n(i)$ and will serve contract $n(j)$ immediately afterwards. Selecting $\text{arc}(0, i, k)$ implies that $n(i)$ is the first contract to be served by ship k , and selecting $\text{arc}(i, 0, k)$ implies that $n(i)$ is the last contract to be served by ship k .

245

246

247

248

249

250

251

252

253

254

255

256

The objective function (1) represents the total solution cost. Constraints (2) ensure that each ship is employed in at most one route. A route is defined as a sequence of contracts to be served. Constraints (3) ensure that, for each contract n , exactly one arc entering set D_n is selected, establishing that each contract must be served exactly once, by exactly one ship, which begins service at exactly one of the nodes or time instants in the discretized time window for cargo pick up. For nodes different to the central fictitious node, constraints (4) state that if an entering arc is selected, a leaving arc must also be selected and that both arcs must be associated with the same ship. For the fictitious node, these constraints (4) state that if a leaving arc associated with ship k is selected, then an entering arc associated with the same ship must also be selected (i.e., if a ship exits the node), and then it

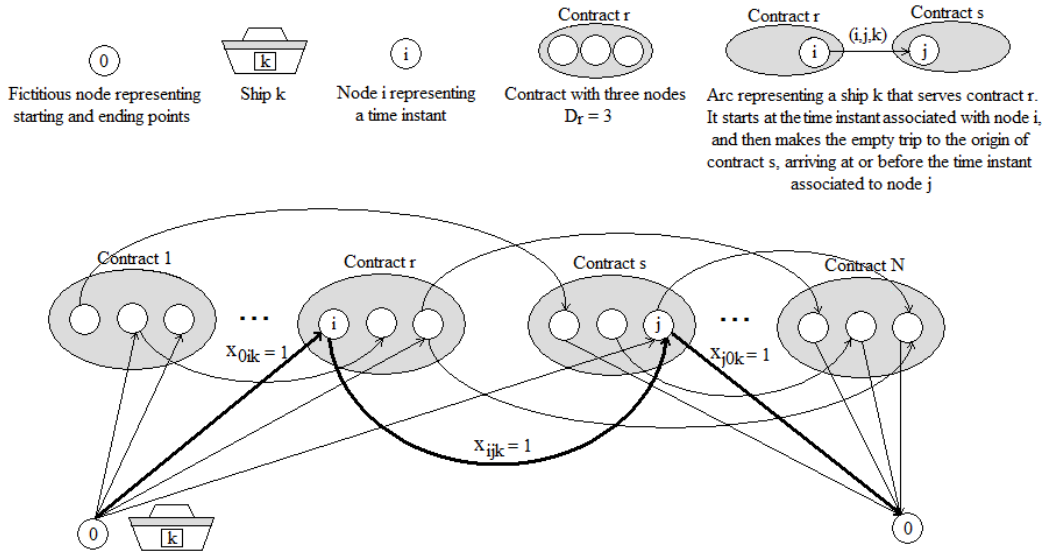


Figure 1: Example of a partial graph of the model for a ship k

257 must return to it. Arcs leaving the fictitious node represent the ships that
 258 are, in fact, used in the solution.

259 Figure 1 shows an illustrative partial graph of the model, where the nodes
 260 of the network represent discrete and feasible starting times for each contract.
 261 The ovals group all nodes related to the same contract. Notation is on the
 262 top of the figure. As stated before, all ships are supposed to start and end
 263 at a fictitious node 0. Some arcs for a single ship k are drawn, based on the
 264 feasible trips that can be selected. The chosen route is marked with bold
 265 lines. If ship k can serve contract r at time instant of node i , the $arc(0, i, k)$
 266 will be selected and x_{0ik} will be equal to 1. Then, if the ship k can serve
 267 contract r and go to serve contract s at time instant of node j , the $arc(i, j, k)$
 268 will be selected and x_{ijk} will be equal to 1. Finally, if ship k does not serve
 269 more contracts, it is supposed to go to the fictitious node 0, so that $arc(j, 0, k)$
 270 is used and x_{j0k} will be equal to 1.

271 4. Hybrid GRASP-VNS Methodology

272 On the one hand, Greedy Randomized Adaptive Search Procedure (GRASP)
 273 is a metaheuristic algorithm commonly applied to combinatorial optimization
 274 problems, and consists of iterations made up from successive constructions

275 of a greedy randomized solution and subsequent iterative improvements of
276 it. The greedy randomized solutions are generated by adding elements to the
277 problem solution set from a list of elements ranked by a greedy function ac-
278 cording to the quality of the solution they will achieve. To obtain variability
279 in the candidate set of greedy solutions, well-ranked candidate elements are
280 often placed in a Restricted Candidate List (also known as RCL), and chosen
281 at random when building up the solution. GRASP was first introduced in
282 Feo and Resende [12], and some survey papers are Feo and Resende [11] and
283 Resende and Ribeiro [34].

284 On the other hand, VNS, proposed by Mladenović and Hansen [26], is
285 another metaheuristic for solving combinatorial optimization problems. VNS
286 systematically changes different neighborhoods within a local search, unlike
287 many metaheuristics where only a single neighborhood is employed. The
288 basic idea is that a local optimum defined by one neighborhood structure is
289 not necessary the local optimum of another neighborhood structure, thus the
290 search can systematically traverse different search spaces which are defined
291 by different neighborhood structures. This makes the search much more
292 flexible within the solution space of the problem, and potentially leads to
293 better solutions which are difficult to obtain by using single-neighborhood-
294 based local search algorithms. Many extensions of VNS have been made,
295 mainly to be able to solve large problem instances [16, 17, 24, 28, 30].

296 This paper proposes the use of a hybrid GRASP-VNS algorithm providing
297 a solution to the SRSPDTW. Results are obtained in less computational
298 time than previous approaches [7, 15], and solutions are of high quality, as
299 it is shown in Section 5. Both aspects are specially important when dealing
300 with large-scale instances. The proposed hybrid algorithm incorporates two
301 powerful features, the effective constructive and improving ability of GRASP
302 and the flexibility of VNS to explore different search spaces for the problem.

303 It is important to notice that a solution to the problem consists of a
304 route for each ship, so that a route is defined as the set of contracts to be
305 performed by the corresponding ship as well as the chosen discretized time
306 windows. The general algorithm (Algorithm 1) proposed in order to obtain
307 these kind of solutions is based on the repetition (L times) of two main
308 steps: the construction of an initial feasible solution using a GRASP, and
309 the improving of this solution applying a VNS algorithm.

Algorithm 1: General Algorithm

```
// Initialization.
1 Initialize  $BestSol \leftarrow \emptyset$ .
2 while (the stopping condition is not met (L is not reached)) do
3   Generate an initial solution  $s$  using GRASP algorithm.
   // VNS.
4   while (the stopping condition is not met (M is not reached)) do
5     (1) Set  $k \leftarrow 1$ ;
6     (2) Repeat the following steps (a), (b), and (c) until  $k = k_{max}$ :
7       (a) Shaking. Generate a point  $s'$  at random from the  $k^{th}$ 
           neighborhood of  $s$ :
8       (b) Local Search.
9         while (improvement is achieved) do
10           $s'' \leftarrow \text{swapInter}(s')$ ;
11           $s'' \leftarrow \text{improveRoutes}(s'')$ ;
12         while (improvement is achieved) do
13           $s'' \leftarrow \text{relocation}(s')$ ;
14           $s'' \leftarrow \text{improveRoutes}(s'')$ ;
15         while (improvement is achieved) do
16           $s'' \leftarrow \text{2-opt}(s')$ ;
17           $s'' \leftarrow \text{improveRoutes}(s'')$ ;
18         while (improvement is achieved) do
19           $s'' \leftarrow \text{relocationChanging}(s')$ ;
20           $s'' \leftarrow \text{improveRoutes}(s'')$ ;
21       (c) Move or not. If this local optimum is better than the
           incumbent, move there ( $s \leftarrow s''$ ), and continue the search
           ( $k \leftarrow 1$ ); otherwise, set  $k \leftarrow k + 1$ .
22   Update  $BestSol$ .
```

311 4.1. GRASP for an Initial Feasible Solution

312 In order to obtain an initial solution, a GRASP has been developed. This
313 algorithm operates as follows. Firstly, a list composed of ships, contracts,
314 and costs is created, as shown in Figure 2. If the problem is composed of B
315 ships, the first B contracts are assigned to each ship with their corresponding
316 costs, as long as the ships can go to perform the contracts. This cost is
317 the least possible cost so that each ship performs the contract using a time
318 node, and taking into account that ships are supposed to be in fictitious
319 node 0 at the beginning. The list is sorted in non-increasing order of cost,

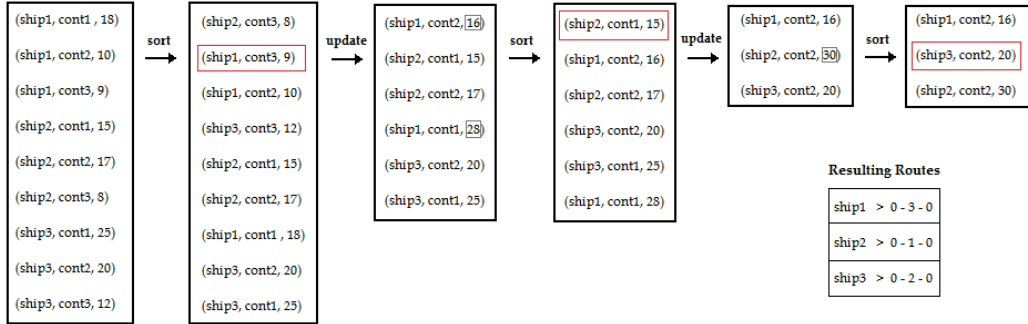


Figure 2: GRASP behaviour example

320 and the chosen element is randomly selected from the RCL. The RCL is
 321 formed by the first three elements of the sorted list, parameter that has been
 322 adjusted in order to obtain variable quality solutions. Once the element is
 323 chosen, the corresponding contract is assigned to the ship. Then, the element
 324 is deleted from the list, as every other element that contains the selected
 325 contract. Moreover, for every element containing the selected ship, the cost
 326 is updated taking into account that the ship is previously performing the
 327 selected contract. This process is repeated until the list is empty, point at
 328 which a new list is created with the following B contracts. When there are
 329 no more contracts, the process finishes.

330 At this point, we have the route that each ship will perform, but it could
 331 happen that some contracts have not been assigned due to arc restrictions.
 332 In that case, a new process starts, trying to insert these contracts at some
 333 point within the routes. If that is not possible, *swapInter* and *relocation*
 334 movements (which are explained in next section) are tried repeatedly seeking
 335 to achieve the new contract insertion. After carrying out a certain number
 336 of iterations of these movements without achieving the insertion, the process
 337 is suspended, as it can happen that it is not possible to obtain an initial
 338 feasible solution.

339 Figure 2 shows an example where the problem is composed of three ships
 340 and three contracts. The first three contracts are assigned to each ship
 341 supposing that they are at the fictitious node at the beginning. Then, the
 342 list is sorted by cost and one of the elements of the RCL is selected. This
 343 selection is depicted in the figure using a framing red rectangle. The list is
 344 updated deleting the elements with the selected contract and changing the
 345 cost corresponding to the selected ship, since now it is performing the selected

346 contract. This process continues until the list is empty, so that the first three
347 contracts are just in a ship route. Similarly, the next three contracts will be
348 assigned to the three ships repeating the process until there are no more
349 contracts.

350 4.2. Variable Neighborhood Search Algorithm

351 As shown in Algorithm 1, unlike the VNS composed of a list of neighbor-
352 hoods sequentially explored in Castillo-Villar et al. [7], the VNS algorithm
353 applied in this work is composed of three phases: shaking, local search, and
354 move decision. At the beginning, for M times, variable k is set to 1 (line 5),
355 and then the iteration of the three phases starts.

356 In the shaking phase a solution is randomly generated applying the corre-
357 sponding neighborhood structure, *i.e.*, the k^{th} neighborhood structure (line
358 7), with k_{max} representing the total number of neighborhood structures. The
359 sequence of neighborhood structures has been chosen following the ideas
360 described by Repoussis et al. [32] which provided high quality results for
361 a vehicle routing problem that presents many similarities with our prob-
362 lem. The sequence is defined as follows: *CROSS*, $2-opt^*$, *relocation*,
363 *relocationChanging*, and *swapInter*. This sequential selection is applied
364 based on cardinality, which implies moving from relatively poor to richer
365 neighborhood structures, and significantly increases the possibilities of find-
366 ing higher quality solutions. The neighborhood structures *GENI* and *Or* –
367 *opt* used by Repoussis et al. [32] have been discarded because they are
368 not applicable to this particular kind of routes. On the other hand, the
369 *relocationChanging* structure, similar to *relocation*, has been added to the
370 sequence. Each operator works randomly, so that the corresponding operator
371 in each iteration of the VNS is performed a limited number of times in order
372 to try obtaining a feasible solution. If it is not possible, the VNS proceeds
373 to next iteration increasing k . The way they work is the following:

- 374 • The *CROSS* operator [41] selects a subsequence of contracts from a
375 route, other subsequence of contracts from other route, and exchanges
376 both subsequences ($O(P^2n^2)$ being P the maximum length of the sub-
377 sequences).
- 378 • The $2-opt^*$ operator [31] chooses two routes and exchange the last part
379 of both routes after two selected point, one from each route ($O(n^2)$).

- 380 • The *relocation* operator [6] deletes a contract from a route and inserts
381 it into another route ($O(n^2)$).
- 382 • The *relocationChanging* operator is a modification of the *relocation*
383 one, where the nodes from contracts between the new one is going to
384 be inserted can change to another node belonging to these contracts,
385 in order to accommodate the new one ($O(n^2)$).
- 386 • The *swapInter* operator selects a contract from a route, other contract
387 from other route, and swaps them ($O(n^2)$).

388 In the local search phase (lines 8-20), four different neighborhood struc-
389 tures are sequentially applied at each iteration: *swapInter*, *relocation*, $2 -$
390 *opt**, and *relocationChanging*. These structures are similar to the ones ap-
391 plied in the shaking phase, but instead of working randomly, they search the
392 movement that involves the highest reduction of cost, *i.e.*, the best solution
393 of the neighborhood. This way, each structure is applied until no improve-
394 ment is achieved (lines 9-20). An improvement method is always applied
395 after performing a neighborhood movement. This method explores all feasi-
396 ble combinations of arcs that connect two contracts in the route of a ship,
397 selecting the pair of arcs with lowest cost. It means that this method tries
398 to find an improvement of the solutions based on an analysis of the time
399 windows of each contract, respecting the contracts already assigned to the
400 route of a ship.

401 The order of neighborhood structures exploration in the local search phase
402 has been established by means of the following study. Firstly, each structure
403 has been individually applied in the local search phase of the VNS, in order
404 to asses its contribution during the search process. A representative subset
405 of instances has been used in this analysis,

406 A subset of representative instances - one instance of each group of 15
407 instances explained in Section 5 - has been used in this analysis. In Fig-
408 ure 3 the neighborhood structures *swapInter*, *relocation*, $2 - Opt^*$, and
409 *relocationChanging* are identified by $N1$, $N2$, $N3$, and $N4$, respectively.
410 The first graph shows that the $N4$ provides the lowest average value of the
411 minimization objective function when used individually. However, applying
412 this neighborhood structure is computationally expensive, so that obtaining
413 results involves large times. For this reason, using $N4$ as first or second
414 neighborhood structure has been discarded. Thereupon, secondly, each com-
415 bination of two structures without $N4$ has been applied as shown in the

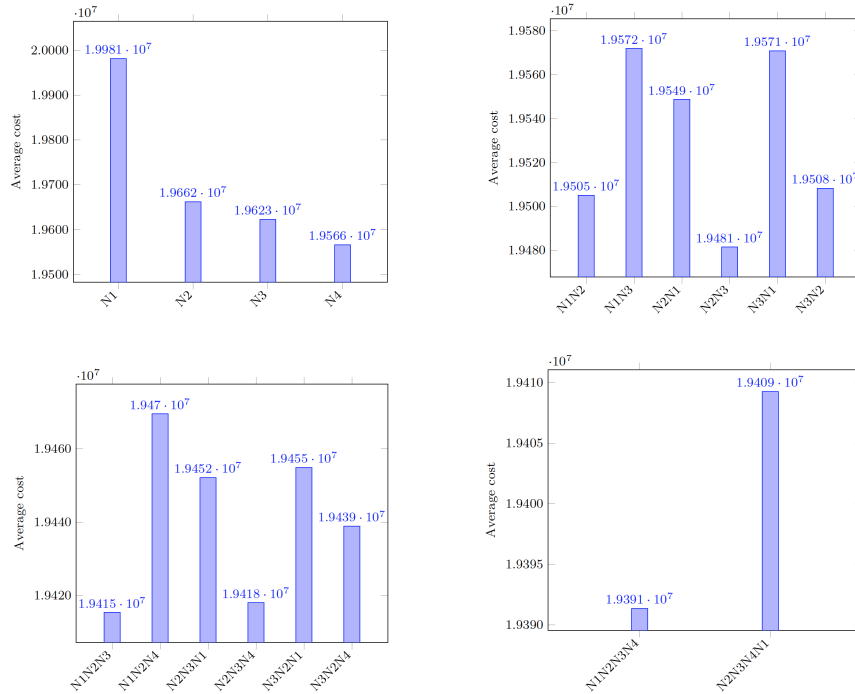


Figure 3: Evaluation of the combination of neighborhood structures

416 second graph, and the three combinations which involve better results have
 417 been selected ($N1N2$, $N2N3$, $N3N2$). Then, each combination of three
 418 structures starting from these better ones has been applied as shown in the
 419 third graph, and the two combinations which involve better results have
 420 been selected ($N1N2N3$, $N2N3N4$). As last step, every combination of four
 421 structures starting from these better ones has been applied as shown in the
 422 fourth graph, and the best one has been selected ($N1N2N3N4$).

423 Finally, in the move decision phase (line 21) the new solution is compared
 424 to the initial one, and if the new one is better, then the solution is updated
 425 and the search starts again setting k to 1. Otherwise, k is increasing by 1
 426 and the next neighborhood in the shaking phase is used.

427 5. Computational Experiments

428 This section is devoted to analyze the performance of the hybrid GRASP-
 429 VNS algorithm introduced in Section 4 for solving the SRSPDTW. Results

430 produced by the proposed algorithm have been compared to exact solutions
431 and previous results reported in the literature [7]. For more clarity, here-
432 inafter the whole heuristic algorithm proposed by Castillo-Villar et al. [7] is
433 referenced as CVH, and its greedy start method is referenced as Greedy. Our
434 algorithm has been implemented using Java Standard Edition 7 and compu-
435 tational experiments have been performed using a 3.00 GHz Intel Core i-5
436 processor with 6 GB of RAM running under Ubuntu 12.10.

437 The set of instances used in this work is the same set generated by
438 Castillo-Villar et al. [7]. There are eighteen groups of instances, each on
439 considering a different combination of ships, time window nodes, and con-
440 tracts. The set of discrete time windows (*i.e.* number of possible starting
441 times for each contract) consists of 3, 6, or 15 nodes. The number of ships
442 varies over the values of 4, 5, 7, and 9. The number of contracts varies over
443 the values of 30, 40, and 50. Each group contains 15 different instances. In
444 total, the benchmark is composed of $3 \cdot 4 \cdot 3 \cdot 15 = 540$ instances.

445 In order to obtain the best results using the proposed GRASP-VNS al-
446 gorithm (Algorithm 1), a parameter setting experimental study has been
447 conducted. Applying the Friedman test [14], M has been fixed to 10, L
448 to 10, and k_{max} to 5. Moreover, the maximum number of times that each
449 neighborhood structure in the shaking phase is tried until a feasible move-
450 ment is obtained has been fixed to 30. This is because not all movements
451 corresponding to a neighborhood are feasible due to time windows.

452 With the aim of demonstrating not only the benefits of our whole pro-
453 posal, GRASP-VNS, but also the benefits of both sides, GRASP and VNS,
454 firstly, 20 executions have been made for each instance using our VNS to-
455 gether with the Greedy, based on prioritizing the earliest due date contracts
456 and seeking to assign each contract at the earliest possible instant to a ship.
457 This combination is referenced as Greedy-VNS. The average results have
458 been compared to the ones provided by Castillo-Villar et al. [7], where stop-
459 ping rules consider a limit time of 7,200 seconds for CPLEX (sometimes its
460 solution does not correspond to an optimal solution) and a maximum number
461 of iterations without improvement in the solution for their whole heuristic
462 method CVH. These results, produced by CPLEX and the heuristic method,
463 have been kindly provided by the authors. This way, it is possible to compare
464 the performance of our VNS algorithm with the performance of the VNS by
465 Castillo-Villar et al. [7]. Secondly, we have made 20 executions using our
466 hybrid GRASP-VNS proposal (see Section 4) in order to show the improve-
467 ments provided by our VNS, the improvements provided by the GRASP

468 regarding the Greedy, and the general improvements of the hybrid GRASP-
 469 VNS regarding the CVH. We have use the same formula than Castillo-Villar
 470 et al. (2014) to calculate the gap values:

$$gap = \frac{Z - CPLEX}{CPLEX} \cdot 100 \quad (7)$$

471 where Z corresponds to the value obtained by the corresponding heuristic.
 472 Positive gaps are obtained when CPLEX finds better solutions.

473 Tables 1, 2, and 3 show a summary of all results obtained for these in-
 474 stances with 30, 40, and 50 contracts, respectively. Each instance type, con-
 475 sisting of 15 instances, is indicated according to its combination of contracts
 476 (*Cn.*), ships (*Sh.*), and nodes (*Nd.*) in columns 1, 2, and 3. The next four
 477 columns are related to solutions obtained using CPLEX. Column 4 is the the
 478 number of instances from which an optimal solution is found (*Opt. found*).
 479 Column 5 is the number of instances for which an optimal solution is not
 480 found, but a feasible solution is found (*Only feas. sol.*). Column 6 is the
 481 number of instances for which no solution is found (*Sol. not found*). Finally,
 482 column 7 is the average time spent to obtain a solution (*Avg. time(s.)*). It is
 483 important to note that sometimes CPLEX is not able to achieve the optimal
 484 solution within the limit time, but it might provide a feasible one or not at
 485 all. Thus, the value of this column corresponds to the average of the time
 486 needed to obtain the optimal or a feasible solution for the 15 instances of
 487 each group, so that if there is not solution for an instance this instance is not
 488 taken into account to calculate the average. This last consideration is always
 489 keeping in mind to calculate the average values in this work.

490 The next three columns present the CVH results: the number of in-
 491 stances for which solution is not found (*Sol. not found*), the average time
 492 needed by the algorithm to provide a solution taking into account that it
 493 is executed considering a maximum number of iterations without any im-
 494 provement in the solution as stopping criterion (*Avg. time(s.)*), and the gap
 495 between CPLEX and CVH objective function values (*Gap₁(%)*). Results of
 496 Greedy together with our VNS algorithm, *i.e.* Greedy-VNS, are shown in
 497 next five columns: the number of instances for which feasible solution is not
 498 found by the algorithm (*Sol. not found*), the average number of executions
 499 (of the 20 executions made for each instance) for which an optimal objective
 500 value is reached (*Avg. opt found*), the average time needed to provide a solu-
 501 tion considering that the algorithm finishes when loops finish, and the loops
 502 are controlled by fix parameters L and M (*Avg. time(s.)*), the gap between

503 CPLEX and Greedy-VNS objective function values ($Gap_2(\%)$), and the gap
504 between CVH and Greedy-VNS objective function values ($Gap_3(\%)$). Finally,
505 results for the GRASP-VNS algorithm proposed here are shown in the six
506 right-most columns. Column *Sol. not found* gives the number of instances
507 for which a solution is not found by the algorithm. Column *Avg. opt found*
508 shows the average number of executions for which the optimal objective value
509 is reached. Column *Avg. time(s.)* shows the average time spent by the al-
510 gorithm to obtain solutions. Columns $Gap_4(\%)$, $Gap_5(\%)$, and $Gap_6(\%)$
511 present the gap between CPLEX and GRASP-VNS objective function val-
512 ues, the gap between the CVH and GRASP-VNS objective function values,
513 and the gap between Greedy-VNS and GRASP-VNS objective function val-
514 ues, respectively.

515 Table 1 shows results for instances of 30 contracts corresponding to the
516 smallest size instances. In terms of the quality of solutions, using our VNS
517 instead of the VNS by Castillo-Villar et al. [7], *i.e.* the Greedy-VNS algo-
518 rithm, the gap with regard to CPLEX solutions is considerably reduced from
519 5.17 to 0.27% on average, and with regard to CVH, solutions are improved
520 on an average of 4.63%. This behavior is repeated with the larger instances
521 as shown in next two tables. This way, we have demonstrated the better
522 performance of our VNS. Additionally, if GRASP replaces Greedy obtaining
523 the GRASP-VNS proposal of this paper, then the results are even better,
524 so that the gap goes from 0.27 to 0.18% regarding CPLEX, and from -4.63
525 to -4.70% regarding CVH. Once again, this behavior is repeated with the
526 larger instances as shown in next two tables. Although the improvement
527 introduced by GRASP could seem not very high, an important advantage
528 of using it is that this is able to find more feasible solutions than the other
529 proposals, as shows column 15 (*Sol. not found*) of each table, and another
530 remarkable aspect is that more than half of the times (11.46 out of 20) that
531 the GRASP-VNS is executed using these instances, an optimal solution is
532 found, demonstrating the robustness of the algorithm. This ratio decreases
533 when the number of contracts increases due to instances complexity, as can
534 be seen in the following tables. However, the approach provided by Castillo-
535 Villar et al. [7] did not produce optimal solutions according to their paper.
536 In regard to computation time, results of GRASP-VNS are far better than
537 CPLEX and CVH. It is noteworthy that although execution machines are
538 different, the magnitude of values is not only due to the difference between
539 machines, but also because of the efficiency of the proposed algorithm.

540 Instances of 40 contracts are medium size instances and results for them

541 are shown in Table 2. For this set of instances CPLEX and CVH need
542 between 4 and 8 minutes on average to obtain solutions, whereas GRASP-
543 VNS requires about 20 seconds on average. The average gap value between
544 CPLEX solutions and GRASP-VNS solutions increase a bit due to the mag-
545 nitude of instances, to a value of 0.46%. However, the gap between CVH
546 and GRASP-VNS is even better than the gap obtained with 30-contract in-
547 stances (-6.06%). This demonstrates that the performance of CVH worsens
548 when the complexity of instances increase, while this is not the case for the
549 proposed GRASP-VNS algorithm.

550 Table 3 shows instances of 50 contracts corresponding to the largest size
551 instances. The behaviour of the GRASP-VNS algorithm is very similar to the
552 one corresponding to 40-contract instances. One more time, the average gap
553 values in respect to CPLEX is low, 0.58%, and CVH results are improved on
554 an average of 5.14%. Notice that average computation times for these largest
555 instances are shorter than the average times for 40-contract instances, but
556 it is due to a particular 40-contract instance with 5 ships and 15 nodes that
557 consumes particularly longer computation time.

558 An important point that can be highlighted from Tables 1, 2, and 3 is
559 that our GRASP-VNS algorithm always finds a solution if CPLEX has found
560 a solution, and even sometimes GRASP-VNS is able to find a solution when
561 CPLEX has not found a feasible one, as can be seen for 50-contract instances.
562 In contrast, CVH always solves less number of instances than CPLEX.

Cn. Sh. Nd.	CPLEX			CVH			Greedy - VNS			GRASP - VNS								
	Opt. found	Only feas. sol.	Sol. not found	Avg. time (s.)	Sol. not found	Avg. time (s.)	Gap1 (%)	Sol. not found	Avg. opt. found	Avg. time (s.)	Gap2 (%)	Gap3 (%)	Sol. not found	Avg. opt. found	Avg. time (s.)	Gap4 (%)	Gap5 (%)	Gap6 (%)
30 4 3	13	0	2	1.00	3	4.83	4.79	3	10.58	0.55	0.22	-4.32	2	11.84	0.61	0.12	-4.41	-0.10
30 4 6	13	0	2	4.84	2	15.92	5.36	2	7.38	1.17	0.36	-4.71	2	9.92	1.33	0.21	-4.84	-0.14
30 4 15	13	0	2	171.84	2	93.15	5.53	2	5.46	6.42	0.42	-4.81	2	9.38	7.26	0.22	-4.99	-0.19
30 5 3	15	0	0	1.13	0	7.06	4.34	0	13.00	0.60	0.22	-3.94	0	13.80	0.62	0.18	-3.97	-0.03
30 5 6	15	0	0	12.73	0	26.53	5.49	0	11.33	1.30	0.20	-4.98	0	12.53	1.31	0.17	-4.99	-0.02
30 5 15	15	0	0	140.60	0	151.40	5.49	0	10.33	7.18	0.19	-4.99	0	11.26	7.34	0.20	-4.97	0.01
	14.00	0.00	1.00	55.36	1.17	49.82	5.17	1.17	9.68	2.87	0.27	-4.63	1.00	11.46	3.08	0.18	-4.70	-0.08

Table 1: Summary of results for instances with 30 contracts

Cn. Sh. Nd.	CPLEX			CVH			Greedy - VNS			GRASP - VNS								
	Opt. found	Only feas. sol.	Sol. not found	Avg. time (s.)	Sol. not found	Avg. time (s.)	Gap1 (%)	Sol. not found	Avg. opt. found	Avg. time (s.)	Gap2 (%)	Gap3 (%)	Sol. not found	Avg. opt. found	Avg. time (s.)	Gap4 (%)	Gap5 (%)	Gap6 (%)
40 5 3	14	0	1	3.85	6	21.55	5.00	6	5.67	0.82	0.50	-4.24	1	5.53	2.63	0.35	-4.39	-0.16
40 5 6	15	0	0	42.00	3	82.50	8.11	3	4.75	2.09	0.67	-6.76	0	3.13	11.72	0.61	-6.97	-0.23
40 5 15	15	0	0	719.40	3	442.58	7.49	3	1.42	14.08	0.72	-6.20	0	3.53	87.50	0.54	-6.40	-0.21
40 7 3	15	0	0	8.66	0	33.93	6.27	0	1.93	69.03	0.40	-5.47	0	1.73	1.00	0.34	-5.52	-0.05
40 7 6	15	0	0	447.53	0	137.13	7.38	0	2.13	2.61	0.49	-6.37	0	3.20	2.64	0.41	-6.44	-0.08
40 7 15	12	3	0	1870.86	0	645.53	7.76	0	2.07	16.63	0.54	-6.63	0	1.66	16.97	0.49	-6.66	-0.04
	14.33	0.50	0.17	515.38	2.00	227.20	7.00	2.00	2.99	17.54	0.55	-5.95	0.17	3.13	20.41	0.46	-6.06	-0.13

Table 2: Summary of results for instances with 40 contracts

Cn.	Sh.	Nd.	CPLX			CVH			Greedy - VNS			GRASP - VNS								
			Opt. found	Only feas. sol.	Sol. not found	Avg. time (s.)	Sol. not found	Avg. time (s.)	Gap1 (%)	Sol. not found	Avg. time (s.)	Gap2 (%)	Gap3 (%)	Sol. not found	Avg. opt. found	Avg. time (s.)	Gap4 (%)	Gap5 (%)	Gap6 (%)	
50	7	3	14	0	1	18.71	2	109.30	5.95	2	0.69	1.41	0.76	-4.85	1	0.66	1.61	0.58	-5.02	-0.18
50	7	6	13	0	2	183.53	4	385.45	5.35	4	0.00	4.35	0.70	-4.39	1	1.53	4.82	0.57	-4.46	-0.08
50	7	15	7	7	1	4291.85	2	1551.38	5.87	2	0.14	30.67	0.67	-4.87	1	0.36	34.14	0.64	-4.91	-0.04
50	9	3	15	0	0	23.80	0	134.06	6.69	0	1.00	1.60	0.59	-5.68	0	0.73	1.61	0.54	-5.71	-0.04
50	9	6	15	0	0	371.53	0	558.40	6.53	0	0.73	4.70	0.86	-5.30	0	0.86	4.70	0.51	-5.59	-0.31
50	9	15	7	8	0	4450.20	0	2795.53	6.09	0	0.13	32.71	0.94	-4.83	0	0.06	33.09	0.61	-5.13	-0.30
			11.83	2.50	0.67	1556.60	1.33	922.35	6.08	1.33	0.45	12.57	0.75	-4.99	0.50	0.70	13.33	0.58	-5.14	-0.16

Table 3: Summary of results for instances with 50 contracts

Cn.	Sh.	Nd.	<i>Greedy</i>		<i>GRASP</i>	
			Gap (%)	Avg. time (s.)	Gap (%)	Avg. time (s.)
30	4	3	44.29	0.10	23.59	0.20
30	4	6	50.56	0.12	27.81	0.31
30	4	15	54.62	0.25	34.75	1.07
30	5	3	54.19	0.10	23.80	0.13
30	5	6	54.56	0.13	26.67	0.19
30	5	15	55.60	0.28	30.57	0.44
			52.30	0.16	27.87	0.39

Table 4: Summary of Greedy and GRASP results for instances with 30 contracts

Cn.	Sh.	Nd.	<i>Greedy</i>		<i>GRASP</i>	
			Gap (%)	Avg. time (s.)	Gap (%)	Avg. time (s.)
40	5	3	59.54	0.11	28.20	1.67
40	5	6	66.01	0.17	33.09	7.01
40	5	15	59.35	0.41	40.01	8.03
40	7	3	66.01	0.12	25.84	0.17
40	7	6	58.75	0.19	30.34	0.27
40	7	15	59.35	0.54	34.21	0.70
			61.50	0.26	31.95	2.98

Table 5: Summary of Greedy and GRASP results for instances with 40 contracts

Cn.	Sh.	Nd.	<i>Greedy</i>		<i>GRASP</i>	
			Gap (%)	Avg. time (s.)	Gap (%)	Avg. time (s.)
50	7	3	67.63	0.13	31.70	0.36
50	7	6	73.10	0.23	36.97	0.85
50	7	15	75.02	0.76	43.74	4.66
50	9	3	67.76	0.14	27.96	0.20
50	9	6	68.60	0.27	30.28	0.36
50	9	15	69.48	0.97	33.49	1.35
			70.26	0.42	34.02	1.30

Table 6: Summary of Greedy and GRASP results for instances with 50 contracts

563 In order to better clarify the impact of the GRASP on the solution of
564 the problem, Tables 4, 5 and 6 provide a comparison of gaps - regarding the
565 CPLEX solutions - when applying the Greedy and the GRASP individually.
566 As can be seen, the GRASP always provides much more quality results than
567 the Greedy in very short time. Therefore, it is reflected once again the
568 improvement made by the GRASP.

<i>Gaps/Ships</i>	4	5	7	9
Gap ₄	0.18	0.34	0.51	0.55
Gap ₅	-4.75	-5.28	-5.50	-5.48

Table 7: Gaps per number of ships

<i>Gaps/Nodes</i>	3	6	15
Gap ₄	0.35	0.41	0.45
Gap ₅	-4.84	-5.55	-5.51

Table 8: Gaps per number of nodes in contracts

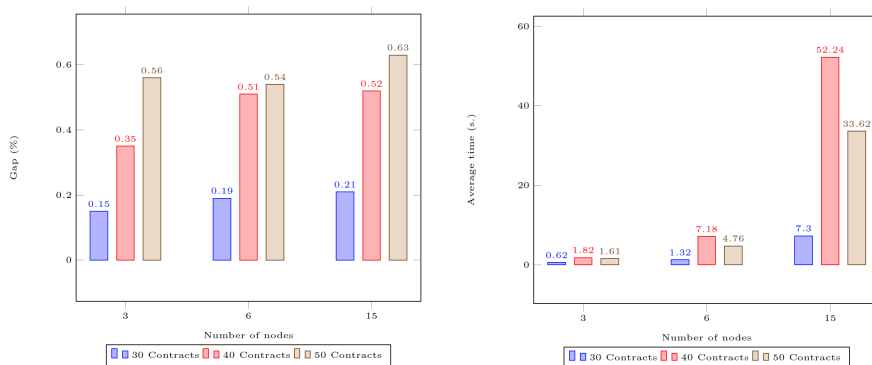


Figure 4: Average time and gaps per number of nodes

569 From previous tables, it can be deduced that the number of contracts af-
570 fects the solutions quality, since the problem complexity increases. In order
571 to know how other instances features influence the quality, Tables 7 and 8
572 report gaps classified according to the number of ships and the number of
573 nodes per contract. As before, Gap₄(%) and Gap₅(%) present the gap be-
574 tween CPLEX and GRASP-VNS objective function values, and between the
575 CVH and GRASP-VNS objective function values, respectively. In Table 7,
576 it can be seen that an increment in the number of ships slightly increases the
577 gap between CPLEX and GRASP-VNS solutions due to the rise in instances
578 complexity. Nevertheless, solutions provided by CVH are continuously im-
579 proved by the GRASP-VNS algorithm, giving an indication that the quality
580 of CVH solutions is clearly much more influenced by the increasing complex-
581 ity. In Table 8, it can be seen that the increment in the number of nodes
582 also results in an increase of the gap between CPLEX and GRASP-VNS so-
583 lutions. Moreover, once again, CVH solutions are widely improved by the

584 proposed GRASP-VNS algorithm, as shown by the negative gaps.

585 With the aim of analysing the behaviour of the GRASP-VNS algorithm
586 when time windows are more discretized, Gap_4 from Table 8 has been split
587 by number of contracts requested in the instances, obtaining the first chart
588 of Figure 4. This way, it can be seen that the highest gap is always presented
589 for 50-contract instances and the lowest gap for the 30-contract instances
590 as expected due to the rise in complexity. A slightly tendency of the gap
591 to increase appears for each number of contracts when the number of nodes
592 increases. The same comparison has been made taking into account time
593 instead of gap. However, in this case it is evident that 40-contract instances
594 present more difficulties to be solved than the other ones, since times are
595 always the highest when solving these instances. Additionally, the sharp
596 increase of time going from 6 to 15 nodes is quite clear, which means that
597 the more discretization is used, the higher will increase the time.

598 6. Conclusions and Further Research

599 In this paper, a hybrid GRASP-VNS algorithm for solving a SRSPDTW
600 has been proposed. This problem belongs to the tramp shipping category,
601 which is increasingly present in the field of maritime cargo transport. The
602 objective considered is to minimize the total cost of serving a set of trip
603 cargo contracts, discretized time windows for picking and delivering cargoes.
604 This allows for a broad variety of features and practical constraints, such as
605 navigation speed to control fuel consumption. Moreover, previous works in
606 literature demonstrated that numerical results considering discretized time
607 windows presents a much better trade-off between solution quality and com-
608 puting time than a similar constant speed continuous model. Even taking
609 into account discretized time windows, using exact algorithm to obtain the
610 optimal results involves large computational times. The hybrid GRASP-VNS
611 algorithm proposed here achieves high-quality solutions in less computational
612 time, and it has been demonstrated that both parts of the algorithm, GRASP
613 and VNS, contribute to this good behaviour.

614 It is noticeable from the computational experiments that results obtained
615 do not only improve previous approximated solutions in the literature, but
616 they are much closer to the optimal ones, presenting an average gap between
617 0.18 and 0.58 %. Actually, optimal solutions are obtained for many instances.
618 Additionally, this GRASP-VNS algorithm finds solutions even when CPLEX
619 is not able to find a feasible one in two hours.

620 On the other hand, an analysis of the proposed hybrid algorithm be-
621 haviour was conducted in order to understand how the number of time win-
622 dows influences the quality of results. It has been shown that the quality of
623 solutions is slightly affected by the level of discretization, so that the more
624 number of nodes, the higher the gap in respect to optimal solutions but still
625 within an acceptable level. However, the computational time shows a sharp
626 increase when the number of nodes goes from 6 up to 15. This means that
627 although the quality of solutions is acceptable, when the number of nodes
628 increases, the computational effort rises quickly. Hence, implementing the
629 right degree of discretization of the problems instances in hand is a key as-
630 pect when solving this problem.

631 On the basis of the contributions presented in this paper, the next stage
632 of the research will be focused on the analysis of how the consideration of the
633 Container Stowage Problem impacts in the selection of contract nodes. The
634 more time containers spend in maritime terminal, the more money should
635 be paid, so this cost should be taken into account. Another open line for
636 future research is applying the proposed hybrid GRASP-VNS algorithm to
637 other tramp shipping or even to other kind of ship routing problems.

638 7. Acknowledgments

639 This work has been partially funded by the European Regional Develop-
640 ment Fund, the Spanish Ministry of Economy and Competitiveness (project
641 TIN2012-32608) and the Tri-continental Campus of Excellence. Eduardo
642 Lalla-Ruiz and Christopher Expósito-Izquierdo thank the Canary Govern-
643 ment the financial support they receive through their doctoral grants.

644 References

- 645 [1] Appelgren, L.H., 1969. A column generation algorithm for a ship
646 scheduling problem. *Transportation Science* 3, 53–68.
- 647 [2] Appelgren, L.H., 1971. Integer programming methods for a vessel
648 scheduling problem. *Transportation Science* 5, 64–78.
- 649 [3] Avriel, M., Penn, M., Shpirer, N., 2000. Container ship stowage problem:
650 complexity and connection to the coloring of circle graphs. *Discrete*
651 *Applied Mathematics* 103, 271 – 279.

- 652 [4] Bierwirth, C., Meisel, F., 2010. A survey of berth allocation and quay
653 crane scheduling problems in container terminals. *European Journal of*
654 *Operational Research* 202, 615–627.
- 655 [5] Brønmo, G., Christiansen, M., Fagerholt, K., Nygreen, B., 2007. A
656 multi-start local search heuristic for ship scheduling—a computational
657 study. *Computers and Operations Research* 34, 900–917.
- 658 [6] Cassani, L., Righini, G., 2004. Heuristic algorithms for the tsp with rear-
659 loading, in: 35th Annual Conference of the Italian Operations Research
660 Society (AIRO XXXV), Lecce, Italy.
- 661 [7] Castillo-Villar, K., González-Ramírez, R., Miranda González, P., Smith,
662 N., 2014. A heuristic procedure for a ship routing and scheduling prob-
663 lem with variable speed and discretized time windows. *Mathematical*
664 *Problems in Engineering* 2014, 24–36.
- 665 [8] Christiansen, M., Fagerholt, K., Nygreen, B., Ronen, D., 2013. Ship
666 routing and scheduling in the new millennium. *European Journal of*
667 *Operational Research* 228, 467–483.
- 668 [9] Christiansen, M., Fagerholt, K., Ronen, D., 2004. Ship routing and
669 scheduling: Status and perspectives. *Transportation Science* 38, 1–18.
- 670 [10] Donati, A.V., Montemanni, R., Casagrande, N., Rizzoli, A.E., Gam-
671 bardella, L.M., 2008. Time dependent vehicle routing problem with a
672 multi ant colony system. *European Journal of Operational Research*
673 185, 1174 – 1191.
- 674 [11] Feo, T., Resende, M., 1995. Greedy randomized adaptive
675 search procedures. *Journal of Global Optimization* 6, 109–133.
676 doi:10.1007/BF01096763.
- 677 [12] Feo, T.A., Resende, M.G., 1989. A probabilistic heuristic for a compu-
678 tationally difficult set covering problem. *Operations Research Letters* 8,
679 67 – 71.
- 680 [13] Fox, M., Herden, D., 1999. Ship scheduling of fertilizer products. *ORI*
681 12, 21–28.

- 682 [14] Friedman, M., 1937. The use of ranks to avoid the assumption of nor-
683 mality implicit in the analysis of variance. *Journal of the American*
684 *Statistical Association* 32, 675–701.
- 685 [15] Gatica, R., Miranda, P., 2011. Special issue on latin-american research:
686 A time based discretization approach for ship routing and scheduling
687 with variable speed. *Networks and Spatial Economics* 11, 465–485.
- 688 [16] Hansen, P., Mladenovic, N., Brimberg, J., Moreno Pérez, J., 2010.
689 *Handbook of Metaheuristics*. Springer. chapter Variable Neighborhood
690 Search. pp. 61–86.
- 691 [17] Hoeller, H., Melian, B., Voss, S., 2008. Applying the pilot method to
692 improve VNS and GRASP metaheuristics for the design of SDH/WDM
693 networks. *European Journal of Operational Research* 191, 691–704.
- 694 [18] Ichoua, S., Gendreau, M., Potvin, J.Y., 2003. Vehicle dispatching with
695 time-dependent travel times. *European Journal of Operational Research*
696 144, 379 – 396.
- 697 [19] Kjeldsen, K., 2011. Classification of ship routing and scheduling prob-
698 lems in liner shipping. *INFOR* 49, 139–152.
- 699 [20] Korsvik, J., Fagerholt, K., Laporte, G., 2010. A tabu search heuristic
700 for ship routing and scheduling. *Journal of the Operational Research*
701 *Society* 61, 594–603.
- 702 [21] Kosmas, O., Vlachos, D., 2012. Simulated annealing for optimal ship
703 routing. *Computers and Operations Research* 39, 576–581.
- 704 [22] Lin, D.Y., Liu, H.Y., 2011. Combined ship allocation, routing and
705 freight assignment in tramp shipping. *Transportation Research Part*
706 *E: Logistics and Transportation Review* 47, 414–431.
- 707 [23] Malliappi, F., Bennell, J., Potts, C., 2011. A variable neighborhood
708 search heuristic for tramp ship scheduling. *Lecture Notes in Computer*
709 *Science (including subseries Lecture Notes in Artificial Intelligence and*
710 *Lecture Notes in Bioinformatics)* 6971 LNCS, 273–285.
- 711 [24] Melian, B., 2006. Using memory to improve the VNS metaheuristic for
712 the design of SDH/WDM networks, in: Almeida, F and Aguilera, MJB

- 713 and Blum, C and Vega, JMM and Perez, MP and Roli, A and Sampels,
714 M (Ed.), Hybrid Metaheuristics, Proceedings, pp. 82–93. 3rd Interna-
715 tional Workshop on Hybrid Metaheuristics, Gran Canaria, SPAIN, OCT
716 13-14, 2006.
- 717 [25] Meng, Q., Wang, S., Andersson, H., Thun, K., 2014. Containership
718 routing and scheduling in liner shipping: Overview and future research
719 directions. *Transportation Science* 48, 265–280.
- 720 [26] Mladenović, N., Hansen, P., 1997. Variable neighborhood search. *Com-
721 puters & Operations Research* 24, 1097–1100.
- 722 [27] Moon, I., Qiu, Z., Wang, J., 2014. A combined tramp ship routing,
723 fleet deployment, and network design problem. *Maritime Policy and
724 Management* 42, 68–91.
- 725 [28] Moreno-Vega, J.M., Melian, B., 2008. Introduction to the special issue
726 on variable neighborhood search. *Journal of Heuristics* 14, 403–404.
- 727 [29] Norstad, I., Fagerholt, K., Laporte, G., 2011. Tramp ship routing and
728 scheduling with speed optimization. *Transportation Research Part C:
729 Emerging Technologies* 19, 853–865.
- 730 [30] P. Hansen, N.M., Perez, J.M., 2008. Variable neighborhood search.
731 *European Journal of Operational Research* 191, 593–595.
- 732 [31] Potvin, J., Rousseau, J., 1995. *J Oper Res Soc* 46, 1433–1446.
- 733 [32] Repoussis, P.P., Paraskevopoulos, D.C., Tarantilis, C.D., Ioannou, G.,
734 2006. A reactive greedy randomized variable neighborhood tabu search
735 for the vehicle routing problem with time windows, in: Almeida, F
736 and Aguilera, MJB and Blum, C and Vega, JMM and Perez, MP and
737 Roli, A and Sampels, M (Ed.), Hybrid Metaheuristics, Proceedings, pp.
738 124–138. 3rd International Workshop on Hybrid Metaheuristics, Gran
739 Canaria, Spain, Oct 13-14, 2006.
- 740 [33] Resende, M., Ribeiro, C., 2002. Greedy Randomized Adaptive Search
741 Procedures.
- 742 [34] Resende, M., Ribeiro, C., 2003. Greedy randomized adaptive search
743 procedures, in: Glover, F., Kochenberger, G. (Eds.), *Handbook of Meta-
744 heuristics*. Kluwer Academic Publishers, pp. 219–249.

- 745 [35] Romero, G., Durn, G., Marenco, J., Weintraub, A., 2013. An approach
746 for efficient ship routing. *International Transactions in Operational Re-*
747 *search* 20, 767–794.
- 748 [36] Ronen, D., 1983. Cargo ships routing and scheduling: Survey of models
749 and problems. *European Journal of Operational Research* 12, 119–126.
- 750 [37] Ronen, D., 1993. Ship scheduling: The last decade. *European Journal*
751 *of Operational Research* 71, 325 – 333.
- 752 [38] Ronen, D., 2002. Marine inventory routing: shipments planning. *J Oper*
753 *Res Soc* 53, 108–11.
- 754 [39] Santhanakrishnan, S., Narendran, T., Ganesh, K., Anbuudayasankar,
755 S., 2012. Comparison of meta-heuristics for container ship routing prob-
756 lem. *International Journal of Services and Operations Management* 12,
757 348–367.
- 758 [40] Sherali, H., Al-Yakoob, S., Hassan, M., 1999. Fleet management models
759 and algorithms for an oil-tanker routing and scheduling problem. *IIE*
760 *Transactions* 31, 395–406.
- 761 [41] Taillard, É., Badeau, P., Gendreau, M., Guertin, F., Potvin, J.Y., 1997.
762 A tabu search heuristic for the vehicle routing problem with soft time
763 windows. *Transportation science* 31, 170–186.
- 764 [42] Tran, N., Haasis, H.D., 2013. Literature survey of network optimization
765 in container liner shipping. *Flexible Services and Manufacturing Journal*,
766 1–41.
- 767 [43] UNCTAD, 2013. Review of maritime transport.
- 768 [44] Woensel, T.V., Kerbache, L., Peremans, H., Vandaele, N., 2008. Vehicle
769 routing with dynamic travel times: A queueing approach. *European*
770 *Journal of Operational Research* 186, 990 – 1007.