A GRASP-based search technique for scheduling travel-optimized warehouses in a logistics company

Christian Pérez, Miguel A. Salido

Universitat Politécnica de Valencia

cripeber@upv.es, msalido@dsic.upv.es

Overview

Introduction

Problem specification

Solving techniques
- Ordering heuristics
- Improvements

Evaluation

Conclusions and future work

Introduction

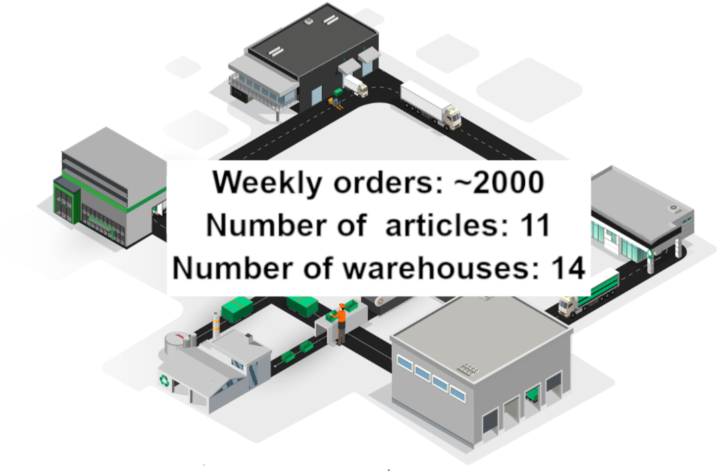

Weekly orders: ~2000
Number of articles: 11
Number of warehouses: 14

Figure: Weekly data management by the company.

Introduction

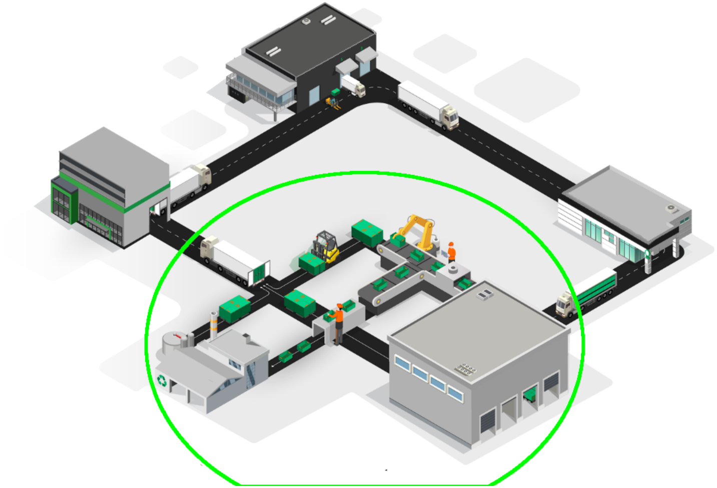


Figure: Diagram of the inner operation of the company.

Introduction

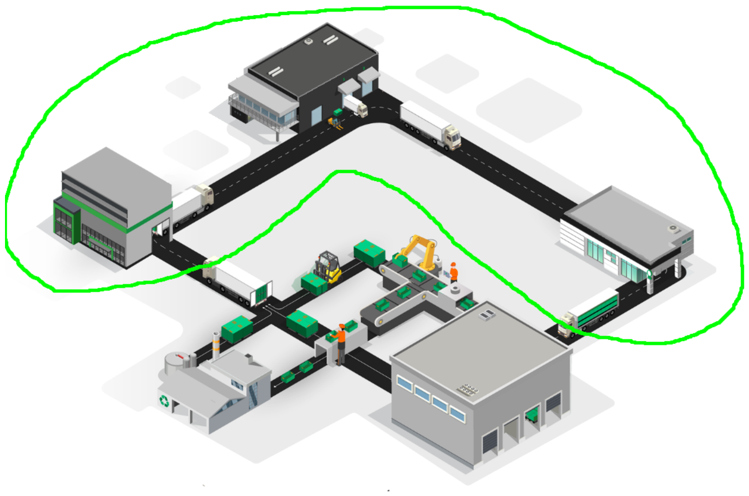


Figure: Diagram of the outter operation of the company.

Problem specification

- $DS = \{O, W, P\}$ is a triple with the weekly data of the company.
- $O = \{o_1, o_2, \ldots, o_N\}$ is a set of orders.
  - Each o_i is composed by $(D_{o_i}, Ir_{o_i}, Ware_{o_i})$
  - D_{o_i} is the delivery date.
  - Ir_{o_i} is a set of articles to be loaded.
  - $Ware_{o_i}$ is a set of posible warehouses.
    - Each $Ware_{o_i j}$ is composed by $(ava_{o_i j}, pr_{o_i j}, dl_{o_i j})$.

Problem specification

- $W = \{w_1, w_2, w_3, \ldots w_M\}$ is a set of warehouses
  - Each warehouse w_i is a bidirection matrix.
  - The rows refers to an article.
  - The column refers to a day of week.
  - $w_{ijk} \in \mathbb{N}$
- $P = \{p_1, p_2, p_3, \ldots p_P\}$ is a set of prices

Objective function

- $f \longleftarrow \alpha * Ct + (1 - \alpha) * \sum Cs$
- Ct : Variable that contains the transport cost of a warehouse order tuple.
- Cs : List containing the stock cost of each of the items in the order for a specific warehouse.
- α : [0 - 1]

Solving techniques

The Greedy algorithm provided by the company

- Control of the distance from the warehouse to the customer and other convenient factors.
- The zone of influence or geographical area is a higher priority in the assignment.
- Restrict the warehouses loading by the type of articles contained in the order.
- The quantity of articles is not constant throughout the year.

Solving techniques

GRASP

```
1: input: Orders $O$, Stocks matrix $W$, Article price vector $P$, Size of LCR list $n$
2: output: Optimized solution $s^*$
3: $s \longleftarrow [\quad]$
4: $LCR \longleftarrow$ LCRordening$(O, n)$
5: while $|s| \neq |LCR|$ do:
6:     $j \longleftarrow 0$
7:     $lcr \longleftarrow [\quad]$
8:     while $j < |LCR|$ do:
9:         $lcr \longleftarrow lcr \cup$ LocalSearch$(LCR_j, \alpha, |LCR|)$
10:        $j \longleftarrow j + 1$
11:    end while
12:    $s \longleftarrow s \cup lcr*$
13:    $LCR \longleftarrow (LCR \setminus lcr*) \cup LCR_{n+1}$
14: end while
15: return $s$
```

Solving techniques

Local Search

```
1: *input:* A order $o$, List of item price $P$, Alpha value for objective function $\alpha$, Length of LCR list $nLCR$
2: *output:* Index of warehouse chosen $j$, Minimum cost of warehouse chosen $Q*$
3: $Q \longleftarrow [\quad]$
4: $I \longleftarrow Ir_o, M \longleftarrow W_o$
5: For $j$ in $\{0, \dots, |M|\}$:
6:     $Cs \longleftarrow [\quad]$
7:     $Ct \longleftarrow \textbf{getTransportCost}(M_j)$
8:     For $i$ in $\{0, \dots, |I|\}$:
9:         $Cs \longleftarrow Cs \cup P_i * \sum_{k=D_o}^{|wst_{ij}|} wst_{ijk} \mid wst_{ijk} < 0$
10:    end for
11:    $Q_j \longleftarrow \alpha * Ct + (1 - \alpha) * \sum Cs$
12:    $Q \longleftarrow Q \cup Q_j$
13:    $\alpha \longleftarrow \textbf{AlphaScheduler}(Cs)$
14: end for
15: return $\left(min(Q), Q_{min(Q)}\right)$
```

Ordering heuristics

GRASP

```
1: *input:* Orders $O$, Stocks matrix $W$, Article price vector $P$, Size of
   LCR list $n$
2: *output:* Optimized solution $s^*$
3: $s \longleftarrow [\ \ ]$
4: $LCR \longleftarrow$ **LCRordening**$(O)$
5: while $|s| \neq |LCR|$ do:
6:    $j \longleftarrow 0$
7:    $lcr \longleftarrow [\ \ ]$
8:    while $j < |LCR|$ do:
9:       $lcr \longleftarrow lcr \cup$ **LocalSearch**$(LCR_j, \alpha, |LCR|)$
10:      $j \longleftarrow j + 1$
11:   end while
12:   $s \longleftarrow s \cup lcr*$
13:   $LCR \longleftarrow (LCR \setminus lcr*) \cup o_{n+1}$
14: end while
15: return $s$
```

Ordering heuristics

LCR ordering

The ordering of the LCR list depending on a number of characteristics of the orders:

- Number of available warehouses (Model 1)
- Number of warehouses and articles (Model 2)
- Number of warehouses and delay (Model 3)
- Number of warehouses and standarized dealy and articles (Model 4)
- Number of weight warehouses and delay (Model 5)
- Early dealy and number of articles (Model 6)

Improvements

Shuffle

This method randomize the LCR list from the criteria used in LCR ordening.

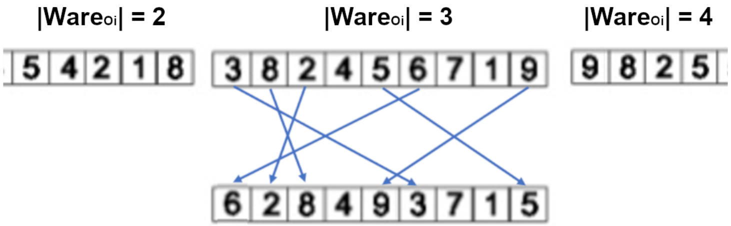


Figure: Example of randomize shuffle method

Improvements

Local Search

```
1: input: A order $o$, List of item price $P$, Alpha value for objective
   function $\alpha$, Length of LCR list $nLCR$
2: output: Index of warehouse chosen $j$, Minimum cost of warehouse
   chosen $Q*$
3: $Q \longleftarrow [\ \ ]$
4: $I \longleftarrow Ir_o, M \longleftarrow W_o$
5: For $j$ in $\{0, \ldots, |M|\}$:
6:     $Cs \longleftarrow [\ \ ]$
7:     $Ct \longleftarrow$ getTransportCost$(M_j)$
8:     For $i$ in $\{0, \ldots, |I|\}$:
9:         $Cs \longleftarrow Cs \cup P_i * \sum_{k=D_o}^{|wst_{ij}|} wst_{ijk} \mid wst_{ijk} < 0$
10:    end for
11:    $Q_j \longleftarrow \alpha * Ct + (1 - \alpha) * \sum Cs$
12:    $Q \longleftarrow Q \cup Q_j$
13:    $\alpha \longleftarrow$ AlphaScheduler$(Cs)$
14: end for
15: return $\left(min(Q), Q_{min(Q)}\right)$
```

Improvements

Alpha scheduler

It is observed that alpha value polarize fitness function during the execution tests.
Increases or decreases the alpha variable regardless of whether or not the assigned order is in a warehouse with negative stock.

```
$F_\alpha \longleftarrow (1-\alpha)/nLCR$

**if** $\sum Cs \leqslant 0$ **then**:
    $\alpha \longleftarrow \alpha + F_\alpha$
**else:**
    $\alpha \longleftarrow \alpha - F_\alpha$
**end if**
```

Evaluation

Study cases

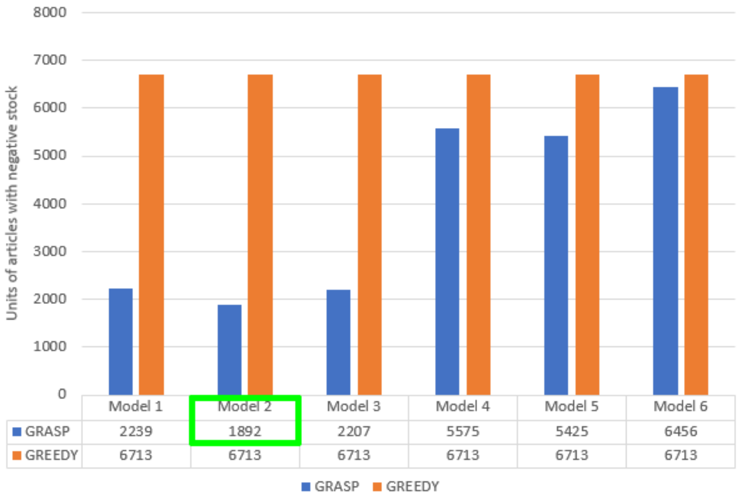
We manage three cases provided by the company:

- **Test case:** dataset prepared by the company to test the algorithm. This data set is used to tune the parameters of the GRASP algorithm.
- **Study cases:** dataset with orders for the first and second week of July

Complexity

- **Orders:** $\simeq 2000$
- **Possible warehouses in o_i:** [1 - 14]
- **Articles in o_i:** [1 - 11]

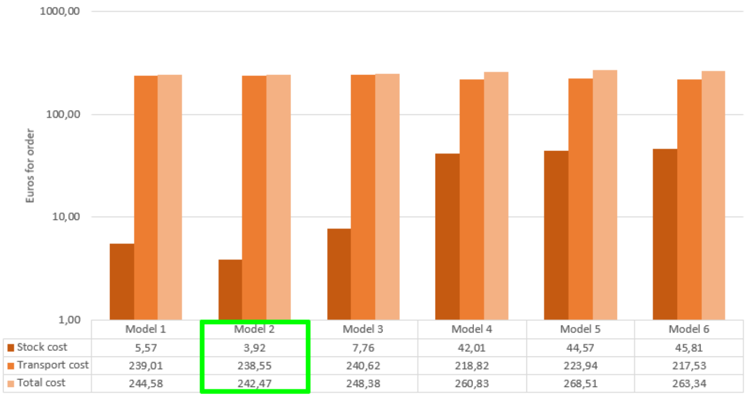
Evaluation



	Model 1	Model 2	Model 3	Model 4	Model 5	Model 6
■ GRASP	2239	1892	2207	5575	5425	6456
■ GREEDY	6713	6713	6713	6713	6713	6713

Figure: Dispersion of negative stock for each model.

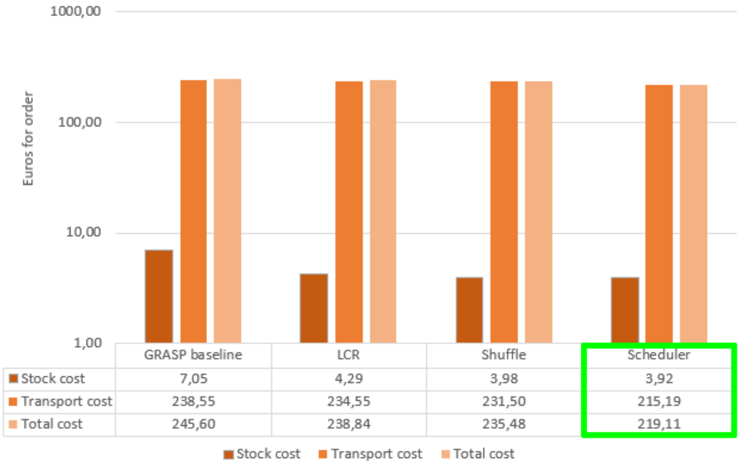
Evaluation



	Model 1	Model 2	Model 3	Model 4	Model 5	Model 6
■ Stock cost	5,57	3,92	7,76	42,01	44,57	45,81
■ Transport cost	239,01	238,55	240,62	218,82	223,94	217,53
■ Total cost	244,58	242,47	248,38	260,83	268,51	263,34

Figure: Cost comparison for each model.

Evaluation



	GRASP baseline	LCR	Shuffle	Scheduler
Stock cost	7,05	4,29	3,98	3,92
Transport cost	238,55	234,55	231,50	215,19
Total cost	245,60	238,84	235,48	219,11

Figure: Different costs for each improvement.

Evaluation

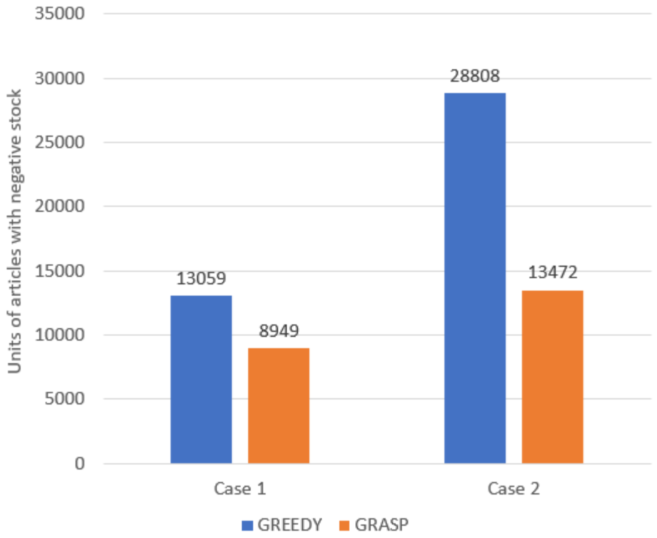
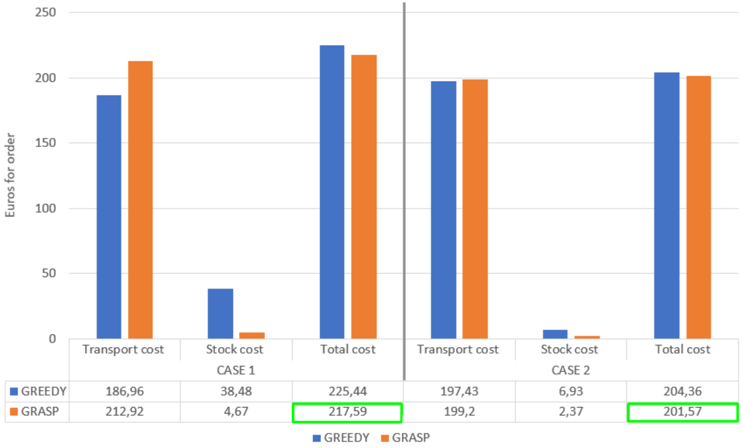


Figure: Dispersion of negative stock for case study.

Evaluation



	CASE 1			CASE 2		
	Transport cost	Stock cost	Total cost	Transport cost	Stock cost	Total cost
🟦 GREEDY	186,96	38,48	225,44	197,43	6,93	204,36
🟧 GRASP	212,92	4,67	217,59	199,2	2,37	201,57

Figure: Different costs for each case study.

Conclusions and future work

Conclusions

- Logistic companies need to optimize their strategies to be more efficient.
- This paper shows that two problems can be combined: the warehouse stock management and routing problem.
- GRASP-based metaheuristic improve the negative average stock by up to 82%.
- Current tools cannot find a solution in a reasonable time.
- The proposed developed algorithm obtains a optimized solution in less than 1.5 minutes.

Future work

- Create a data pool from the grasp-based metaheuristic solutions.
- Develop a genetic algorithm where the initial population is fed with solutions obtained by GRASP.

A GRASP-based search technique for scheduling travel-optimized warehouses in a logistics company

Christian Pérez, Miguel A. Salido

Universitat Politécnica de Valencia

cripeber@upv.es, msalido@dsic.upv.es