OPTIMIZING YARD ASSIGNMENT AT AN AUTOMOTIVE TRANSSHIPMENT TERMINAL

Luigi Moccia[a, b], Gregorio Sorrentino[c]

[a] Istituto di Calcolo e Reti ad Alte Prestazioni, Consiglio Nazionale delle Ricerche, Via P. Bucci 41C, 87036 Rende (CS) - Italy
[b] Dipartimento di Elettronica, Informatica e Sistemistica, Università della Calabria, Via P. Bucci 41C, 87036 Rende (CS) - Italy
[c] Dipartimento di Elettronica, Informatica e Sistemistica, Università della Calabria, Via P. Bucci 41C, 87036 Rende (CS) - Italy

[a] luigi.moccia@icar.cnr.it, [c] gsorrentino@deis.unical.it

ABSTRACT
We present an optimization model for the yard allocation process in an automotive transshipment terminal. The cars arrive and depart by ships in large batches and the yard planners have to dynamically assign incoming cars to parking rows. The integer linear programming model has been implemented in a commercial solver (CPLEX). We will present at the conference the model, the CPLEX computational results, and a new meta-heuristic.

Keywords: automotive transshipment terminal, yard management, integer linear programming

1. INTRODUCTION

We discuss an operational problem arising in an automotive transshipment terminal. Maritime automotive transportation is developing along lines similar to container transportation where the hub and spoke arrangement is widely adopted (Mattfeld, 2006). Deep-sea vessels operate between a limited number of transshipment terminals (hubs). Smaller vessels (feeders) link the hubs with the other ports (spokes). This network topology results in the consolidation of capacity along the routes linking the transshipment ports and in the growth of their importance. Deep-sea car carriers have a capacity up to 6000 vehicles while ships deployed on the short-sea segment attain a capacity up to 1000 vehicles. Therefore, automotive transshipment terminals manage large flows of incoming and outgoing cars. Unlike containers, cars are "fragile" objects that require careful and consequently labour intensive handling. For example, cars can not be stacked and this results in larger yards with respect to container terminals. The yard management process is the heart of a transshipment terminal. The cars arrive and depart by ships in large batches and the yard planners have to dynamically assign incoming cars to parking rows. Once assigned to parking rows, the cars are not relocated inside the yard, i.e. their initial yard position is not modified during their duration of stay. The reason is that a re-handling process will augment the risk of damage which has to be kept at the lowest possible level. This "no-relocation" rule, combined with the low density yard, augments the importance of optimal yard assignment. In fact, the total traveled distance becomes a critical issue in such a low density yard. The transport of a car from the quay to the parking slot is performed by a driver. The drivers are grouped in teams and they are assisted by a mini-bus that brings back the drivers to the quay when unloading (to the yard in the loading case). This driver gathering process is another relevant operational problem. In the following we will indicate as a group a set of cars that arrive and depart by the same pair of vessels, and are of the same type (car model and brand). In order to facilitate the yard management and the driver gathering process, a group is allocated to a set of adjacent parking rows, Figure 1. The number of required parking rows depends upon the car length, and upon the row length since the rows have variable lengths in the yard. Yard managers prefer not to share a row between different groups. Therefore, partially empty rows are possible.

We present an optimization model for the yard allocation process. The model has been implemented in a commercial integer linear programming solver (CPLEX).We will present at the conference the model, the CPLEX computational results, and a new meta-heuristic. The meta-heuristic is inspired by the well-known Greedy Randomized Adaptive Search Procedure (GRASP), (Resende and Ribeiro, 2003), and it is guided by the principle of maximizing the rotation index of the most favorable yard positions (Goetschalkx and Ratliff, 1990).

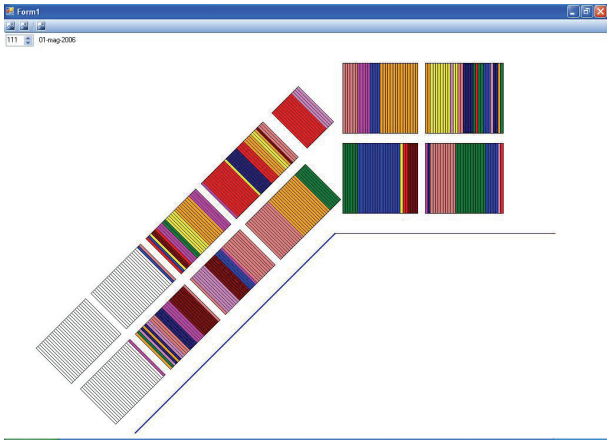

Figure 1: Example of yard allocation

2. MATHEMATICAL MODEL

We assume a discretized rolling time horizon, and we define as a time step a fraction of a work shift. Let t express the time step index, $t \in T = \{1,\ldots, |T|\}$, i.e. the time horizon consists in $|T|$ time steps. The set of groups to allocate during the time horizon is indicated by $K = \{1,\ldots, |K|\}$, and $R = \{1,\ldots, |R|\}$ is the set of parking rows. The group related data are:

- η^k, number of cars of the group k;
- v_r^k, maximum number of cars of the group k that can fill the row r;
- a^k, arrival time;
- b^k, departure time;
- o^k, quay unloading position;
- d^k, quay loading position;
- c_a^k, largest admissible unloading handling time, i.e. transport from the quay unloading positions to the assigned parking rows;
- c_b^k, largest admissible loading handling time, i.e. transport from the parking rows to the the quay loading positions.

Rows are numbered in the filling direction, i.e. if row r is filled before row s then $r < s$. The row ordering is such that if rows r and $s, r < s,$ are adjacent then $s = r+1$. We will consider later in this section the case of an "ending-row" arising when a given row r does not have an adjacent row in the filling direction. In the following we assume that it always exists an adjacent row. For each group k we have to find a set of free adjacent rows of sufficient capacity. Since we consider parking rows of variable length, the number of required rows is variable as well. Let r be the first row in the filling direction assigned to group k. Then the last row will be $r + q_r^k$, where q_r^k is the smallest positive integer value satisfying the following inequality:

$$\sum_{\alpha=0}^{q_r^k-1} v_{r+\alpha}^k \geq \eta^k$$

The q_r^k value expresses the number of rows needed by the group k when the group first row is r, i.e. the group would occupy the row interval I_r^k defined as $I_r^k = \{r, r+1,\ldots, r+q_r^k-1\}$. Analogously, we define as ϑ_s^k the number of rows that the group k would require if s is the *last* row of the group, i.e. ϑ_s^k is the smallest positive integer value satisfying the following inequality:

$$\sum_{\alpha=0}^{\vartheta_s^k-1} v_{s-\alpha}^k \geq \eta^k$$

Consequently, we have the row interval $\{s-\vartheta_s^k+1, s-\vartheta_s^k+2,\ldots, s\}$ which is equivalent to I_r^k if $r = s-\vartheta_s^k+1$. Since the q_r^k and ϑ_s^k values are related to the filling direction, we indicate them as "forward row request", and "backward row request", respectively.

Thanks to this notation we can indicate an assignment of a group to the yard as an assignment to a first row. The "ending-row" case is now treaded by considering non-admissible an assignment of a group k to a first row r such that $q_r^k > 1$ and the set $\{r, r+1,\ldots, r+q_r^k-2\}$ contains an ending-row. Let $\overline{R} \subset R$ be the set of ending-rows, we define as *R(k)* the subset of R such that we do not have an intermediate ending-row for any assignment of k to $r \in R(k)$ i.e. $R(k) = \{r \in R : I_r^k \setminus \{r+q_r^{k-1}\} \cap \overline{R} = \varnothing\}$. Our decision variables are:

- x_r^k if the first row of the group k is r, i.e. the group occupies the row set $\{r, r+1,\ldots, r+q_r^k-1\}$.

Quick Time è un decompressore sono necessari per visualizzare quest'immagine.

Since we want to minimize the total handling time, we define as c_{vz} the time distance between $v \in R \cup O$ and $z \in R \cup D$, where the set O represents the unloading positions $O = \bigcup_{k \in K} \{o^k\}$, and, similarly we indicate by D the set of loading positions, $D = \bigcup_{k \in K} \{o^k\}$. The time distances are derived by the terminal operational database. Thus, these values incorporate set-up times and reflect real average speeds between positions. Our decision variables induce cost coefficients defined as follows:

- $c^k_{o^k r}$, unloading handling time for the group k when the first assigned row is r: $c^k_{o^k r} = \sum_{\alpha=0}^{q^k-2} c_{o^k, r+\alpha} \times v^k_{r+\alpha} + c_{o^k, r+q^k-1} \times (\eta^k - \sum_{\alpha=0}^{q^k-2} v^k_{r+\alpha})$
- $c^k_{rd^k}$, loading handling time for the group k when the first row is r: $c^k_{rd^k} = \sum_{\alpha=0}^{q^k-2} c_{r+\alpha, d^k} \times v^k_{r+\alpha} + c_{r+q^k-1, d^k} \times (\eta^k - \sum_{\alpha=0}^{q^k-2} v^k_{r+\alpha})$

We express by $H_t, \forall t \in T$, the largest desired handling per time step. We use the H_t values to avoid, if possible, handling peaks. The model can be solved iteratively by using, at the first iteration, arbitrarily large H_t values. Then, if the planners prefer to smooth the resulting handling peaks of this first solution, the model is solved by imposing the desired H_t values. The process is iterated until a feasible and satisfying solution has been found.

The following sets are defined for notational compactness:

- $T(k) = \{t \in T : a^k \le t \le b^k\}, \forall k \in K$, the set $T(k)$ represents the duration of stay of group k;
- $K(t) = \{k \in K : t \in T(k)\}, \forall t \in T$, groups that at the time step t are into the terminal;
- $K_a(t) = \{k \in K : t = a^k\}, \forall t \in T$, groups that at the time step t arrive at the terminal;
- $K_b(t) = \{k \in K : t = b^k\}, \forall t \in T$, groups that at the time step t leave the terminal.

We can now formulate our problem, in the following indicated as Adjacent Row Dynamic Assignment Problem (ARDAP):

$$\min \sum_{k \in K} \sum_{r \in R(k)} (c^k_{o^k r} + c^k_{rd^k}) y^k_r \quad (1)$$

subject to

$$\sum_{r \in R(k)} y^k_r = 1 \qquad \forall k \in K \quad (2)$$

$$\sum_{k \in K(t)} (y^k_r + \sum_{\alpha=1}^{\theta^k_r - 1} y^k_{r-\alpha}) \le 1 \qquad \forall r \in R, \forall t \in T \quad (3)$$

$$\sum_{r \in R(k)} c^k_{o^k r} y^k_r \le c^k_a \qquad \forall k \in K \quad (4)$$

$$\sum_{r \in R(k)} c^k_{rd^k} y^k_r \le c^k_b \qquad \forall k \in K \quad (5)$$

$$\sum_{k \in K_a(t)} \sum_{r \in R(k)} c^k_{o^k r} y^k_r + \sum_{k \in K_b(t)} \sum_{r \in R(k)} c^k_{rd^k} y^k_r \le H_t \qquad \forall t \in T \quad (6)$$

$$y^k_r \in \{0,1\} \qquad \forall k \in K, \forall r \in R(k) \quad (7)$$

The objective function (1) minimizes the handling times. The constraints (2) state that every group k must be allocated to one and only one admissible row r, since r must belong to $R(k)$. The feasibility of the assignment is guaranteed by constraints (3): a row r at a time step t is occupied if a group $k \in K(t)$ is allocated to r as its first row, $y^k_r = 1$, or if its first row belongs to the interval $\{r - \theta^k_r + 1, r - \theta^k_r + 2, ..., r - 1\}$. Constraints (4) and (5) model loading and unloading priorities, respectively. The c^k_a (equivalently c^k_b) coefficient can be set to a smaller value to ensure that the group k is assigned to rows closer to the unloading (loading) quay position. This results in user controlled parameters to specify group priorities, since closer rows mean faster handling times. Observe that constraints (4) and (5) could be eliminated incorporating them in the definition of the admissible row assignment set $R(k)$. However, these constraints highlight the degree of available intervention and we prefer to maintain them. Constraints (6) limit the maximum handling for each time step. The model dimensions are as follows: $|K| \times |R|$ binary variables and $3|K| + |T| \times (|R| + 1)$ constraints. In term of computational complexity we prove that ARDAP is strongly NP-hard.

Theorem 1 ARDAP is strongly NP-hard.

Proof- We will prove this by showing that the *Generalized Assignment Problem* (GAP) is a particular case of the ARDAP. In the GAP (Martello and Toth, 1992) we have to find a minimum cost (or equivalently a maximum profit) assignment of a set of weighted items to a set of knapsacks. Let N be the set of items, $N = \{1, ..., n\}$, and M the set of knapsacks, $M = \{1, ..., m\}$. We indicate by c_{ij} the assignment cost of the item i to the knapsack j, by w_{ij} the weight of the item i when assigned to the knapsack j, and by w_j the capacity of the knapsack j. An equivalent ARDAP instance could be defined as follows:

- an item i corresponds to a group k and vice versa, i.e $K = N$, and in the following we equivalently refer to items or groups;
- the ARDAP time horizon consists of only one time step, i.e. $|T| = 1$, and all the groups defined above arrive and leave the terminal at this time step, i.e. $K(1) = K$;

- the set of rows has cardinality equal to the sum of the knapsacks capacities, $|R| = \sum_{j \in M} w_j$;
- the set of ending rows has cardinality equal to the number of knapsacks, $|\overline{R}| = m$;
- we define a partition of the set R in m subsets $S_j, j \in M$: $S_j = \{r_j, ..., s_j\}$ where $r_j = \sum_{l=1}^{j-1} w_l + 1$ and $s_j = r_j + w_j - 1$, i.e. $|S_j| = w_j$, and the row s_j is an ending-row, $s_j \in \overline{R}$;
- the group forward row request q_r^k is constant for the row belonging to a given subset S_j and it is equal to the corresponding weight of the item, i.e $q_r^k = w_{kj}, \forall r \in S_j, j \in M$; similarly, the group backward row request θ_s^k is equal to the weight of the item in each subset S_j;
- the group to row assignment cost $c_{o^k r}^k + c_{r d^k}^k$ is constant for the row belonging to a given subset S_j and it is equal to the corresponding cost of the item, i.e $c_{kj}, \forall r \in S_j, j \in M$;
- the right-hand side coefficients of constraints (4) - (6) are set to arbitrarily large values.

It is immediate to see that the procedure outlined above constructs an ARDAP instance equivalent to the GAP one. An optimal solution for this ARDAP instance could be polynomially transformed in an optimal solution for the GAP. Therefore, if it exists a pseudo-polynomial algorithm A for the ARDAP, then A would solve the GAP as well. Since the GAP is known to be strongly NP-hard, the result follows.

ACKNOWLEDGMENTS

This work was conducted when the first author was on a temporary leave at the Università della Calabria. The research has been supported by the MUR (Italy) under the project AUTOMA - AUTOmobile logistic Management.

REFERENCES

Goetschalkx, M. and Ratliff, D. H. (1990). Shared storage policies based on the duration stay of unit loads. *Management Science*, 36(9):1120–1132.

Martello, S. and Toth, P. (1992). Generalized assignment problems. In *Algorithms and Computation*. Springer-Verlag.

Mattfeld, D. C. (2006). *The Management of Transshipment Terminals, Decision Support for Terminal Operations in Finished Vehicle Supply Chains*. Springer.

Resende, M. and Ribeiro, C. (2003). Greedy randomized adaptive search procedures. In Glover, F. W. and Kochenberger, G. A., editors, *Handbook of Metaheuristics*, volume 57 of *International Series in Operations Research & Management Science*. Springer.