A Constraint Satisfaction Approach for Master Bay Plans

D. Ambrosino, A. Sciomachen
Università degli studi di Genova - Facoltà di Economia -
Via Vivaldi 2 – 16126 Genova –Italy
E-mail:ambrosin/sciomach@economia.unige.it

Abstract: In this paper we deal with the so-called Master Bay Plan (MBP) Problem, that is the problem of stowing a given set of containers of different types into the available cells of a containership, while not violating some structural and operational constraints. We analyse in details the above constraints and present a Constraint Satisfaction Programming (CSP) approach for defining this combinatorial problem and finding good feasible solutions, with respect to some optimisation criteria.

1. Introduction and problem definition

The stowage of a containership is one of the problem that has to be solved daily by any company which manages a terminal container [1,7]. This problem, usually denoted Master Bay Plan (MBP), consists in determining where to stow a given set of containers of different types among a set of available locations (cells) of a containership in such a way to satisfy some structural and operational constraints related to both the containers and the ship. In particular, in looking for a solution of the problem we have to take into a proper account, besides the structural characteristics of the ship, such as type, size and capacity:

- the size, shape, weight and volume of the freight to be loaded (e.g. cars, pallets and containers);
- the material handling system used for the loading / unloading operations (e.g. forklifts and ship-to shore cranes),
- specific location requirements imposed by the shipping companies;
- the equilibrium and stability of the ship during its journey and after

some unloading operations;
- the destination of the freight;
- the quality, quantity and type of the freight to be loaded at the next harbours.

It can be easily noted that MBP is a NP-hard combinatorial problem; the well known Assignment Problem (AP) and Knapsack Problem (KP) [5], where the knapsack is the ship and the objects to be loaded are the containers, that are both included in the definition of MBP, give an idea of its complexity even if all the other constraints are not considered.

Presently, the stowage of containers is usually performed by the Captain few hours before the ship arrival and it is evaluated with respect to some stability rules also for allowing further containers to be loaded at the very last moment. The main aim of this work is to derive and implement some rules that may allow to determine good stowage plans. Such rules can be thought as a prototype of a Decision Support System (DSS) that can help Captains in this activity giving a good distribution of the containers in the ship in a short computational time. Unfortunately there is no, at least to the Authors' knowledge, any computational approach proposed in the recent literature for dealing with MBP. DSSs, heuristics, mainly based on relaxation of Mixed Integer Programming (MIP) models, and stochastic models have been suggested as the most suitable approaches for solving problems that have only some commonalties with MBP (see [2,3,4] among others).

We present a CSP approach that is mainly based on heuristic procedures that rely on a specific knowledge of the problem and on the experience of the Captains of the ships.

In Section 2 we present the main characteristics of a containership. A CSP formulation of the problem is proposed in Section 3, while more details about the constraints are given in Section 4. Finally, in Section 5 we present the algorithmic stucture of our resolution approach.

2. The basic structure of a containership

When solving MBP of particular interest are the constraints related to the structure of the ship, its type and the size of the hold and upper deck. We consider here two types of containerships, namely Ro-Ro (Roll on-Roll off), which load / unload containers through the shutter located either at bow or stern of the ship, and Lo-Lo (Lift on-Lift off), which load / unload containers from the top (by using forklifts).

To give an idea of how a stowage takes place, let us consider the basic

structure of a ship (see Figure 1) and its cross section; it consists of a given number of cells where the containers are located, having variable size depending on the ship (the most common one is 8 feet in height, 8 feet in largeness and 20 feet in depth). Each cell is addressed by the following identifiers:

bay, that gives its position related to the cross section of the ship (counted from bow to stern);

row, that gives its position related to the vertical section of the corresponding bay (counted from the centre to outside);

tier, that gives its position related to the horizontal section of the corresponding bay (counted from the bottom to the top of the ship).

A Ro-Ro ship has stern ramps through which trailers can stow the containers and a forklift that allows to bring the containers from a truck and moving them to their assigned cells. The holds are reached by mean of stern ramps located in a cross way on each bay. They have some special angular connections for helping the stowage of the containers to be located in the upper deck. Note that the capacity of the upper deck is usually greater than the capacity of the hold in both directions.

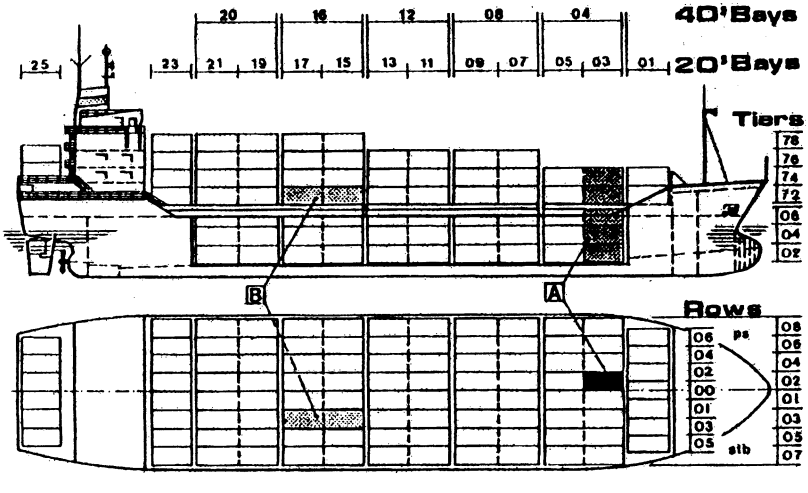

Figure 1: Horizontal and cross sections of a standard containership.

3. The proposed CSP formulation

A CSP problem is generally defined by a set $X = \{x_1, x_2, ..., x_n\}$ of n variables, their associated domain $D_1, D_2, ..., D_n$, and a set $C = \{c_1, c_2, ..., c_p\}$ of p constraints. Each constraint c_j, $j = 1, ..., p$, involves a specific set $X_j \subseteq X$ of argument variables and specifies which values of the variables belonging to X_j are compatibles. A constraint may be specified in a great variety of forms, such as logical expressions, algebraic equations or inequalities. A given variable can take only a finite number of candidate values. The goal is to find an assignment to all variables which does not violate any of the p constraints.

In the present case, let us denote a cell of a ship by x_{ijk}, where indices i, j and k give, respectively, its bay, row and tier position. x_{ijk} hence is our variable of the problem taking value in a finite domain D_{ijk} of integer numbers that identify a specific subset of containers; in particular, $D_{ijk}[dest]$, $D_{ijk}[type]$, $D_{ijk}[size]$ and $D_{ijk}[weight]$ denote, respectively, the set of destinations, types, sizes and weights of the containers that can be possibly stowed in cell x_{ijk}.

In our CSP formulation we also have to specify sets {Bay}, split into {odd-Bay} and {even-Bay}, {Tier} and {Row} that identify the cells' numbers in the corresponding section. In what follows we will denote by {A} \ {B} the set of available indices for {A} but not {B}.

4. The constraints of the problem

As we have already mentioned, solving MBP means to take into account both the constraints related to the particular ship under consideration previously analysed and those of the containers, here below analysed.

Size of the containers. We consider here the standard sizes of a container, namely 20 and 40 feet in length with a section of 8'x8'. Containers of 40' require two contiguous bow-stern locations of 20 feet each that, for security reasons, have usually to be located in odd bays (for instance bay02); consequently those cells of the same row and tier belonging to bay01 and bay03 are not anymore available for stowing container of 20'. Moreover, we assume, as it is always required, that they can be stowed only in the third tier of the hold (tier06) or in any other tier in the upper deck and cannot be located under cells where containers of 20' are already stowed or over empty cells. Referring to

Figure 1 and using the notation given in the previous Section, we can then express these constraints as:

if $D_{ijk}[size] == 40'$ ***then*** $D_{i-1jk}[size] = 40'$ *or* $D_{i+1jk}[size] = 40'$; (1)
i in {odd-Bay}\{22,24}; j in {Row} \ {05,06}; k in {Tier}\ {02,04}.

if $D_{ijk}[size] == 40'$ ***then*** $D_{i-1jk+2}[size]$!= 20' && $D_{i+1jk+2}[size]$!= 20'; (2)
i in {odd-Bay}; j in {Row}; k in {Tier}\ {02,04,06,82}.

if $D_{ijk}[size] == 40'$ ***then*** $D_{i-1jk-2}[size]$!= ∅ && $D_{i+1jk-2}[size]$!= ∅; (3)
case hold: i in {odd-Bay} \ {22,24}; j in {Row}\ {05,06}; k= 6;
case upper deck: i in {odd-Bay}; j in {Row}; k in {Tier}\ {02,04,06,82}

Containers of 20', instead, can be stowed in any cell but not over cells with containers of 40'; more formally conditions 4) and the asymmetric of 2) hold:

if $D_{ijk}[size] == 20'$ ***then*** $D_{ijk-2}[size]$!= ∅; (4)
case hold: i in {even-Bay}\ {21,23,25}; j in {Row} \ {05,06};
k in{Tier} \ {02,82,84,86};
case upper deck: i in {even-Bay}; j in{Row}; k in{Tier}\{02,04,06,82}.

Type of the containers. Let us have only three types of containers, namely standard, carreageable and containers for dangerous loads. Note that this last type of containers cannot be stowed in the upper deck and in adjacent locations, while carreageable containers are considered as containers of 40' and then conditions 1) – 3) hold allowing both odd and even bays. Moreover they must be located in the first tier in the upper deck (tier02) and all the locations over them must be free, that is:

if $D_{ijk}[type] ==$ carreageble ***then*** $D_{ijk+2}[type] =$ ∅ && $D_{ijk+4}[type] =$ ∅;
i in { Bay} \ {25}; j in {Row}; k = {82}. (5)

Weight of the containers. The standard weight of an empty container ranges from 2 to 3,5 tons, while the maximum weight of a full container to be stowed into a ship ranges from 20-32 and 30-48 tons. Usually, as far as the unloading operation is concerned, containers are put on the yard into different stacks on the basis of their size, destination and class, that is derived according to their weights and allows to establish a priori those containers that have to be located in the hold and those that have to be located in the upper deck of the ship.

Note that the weight constraint related to each container is strictly dependent on the presence in the ship of shutters that enable the entry from the hold, since they cannot support more than a given weight. In particular, conditions 6) – 7) must be usually verified, saying that the weight of a stack of 3 containers of 20' and 40' cannot be greater than 45 and 66 tons, respectively.

if $D_{ij82}[type] = D_{ij84}[type] = D_{ij86}[type] = = 20'$ (6)
then $D_{ij82}[weight] + D_{ij84}[weight] + D_{ij86}[weight] \leq 45$ tons;
i in {even-Bay} \ {21,23,25,27,29}; j in {Row}.

if $D_{ij82}[type] = D_{ij84}[type] = D_{ij86}[type] = = 40'$ (7)
then $D_{ij82}[weight] + D_{ij84}[weight] + D_{ij86}[weight] \leq 66$ tons;
i in {odd-Bay}\ {21,23,25,27,29}; j in {Row}.

Moreover, the weight of a container located in a tier cannot be greater than the weight of the container located on the next tier having the same row and bay, that is:

$D_{ijk}[weight] \leq D_{ijk-2}[weight]$ (8)
i in {Bay} \ {21,23,25}; j in{Row} \ {05,06}; k in {Tier}.

Destination of the containers. A part from the more general rule for which we have to load first those containers having as destination the final stop of the ship and for the last those containers that have to be unloaded first (9), it is necessary to consider that in each odd bay there are stern ramps that allow to entry in the hold from the upper deck and hence we cannot load in the hold containers having a greater unload priority than those located in the upper deck closed to the shutters. Then we have condition (10)

$D_{ijk+1}[dest] \leq D_{ijk}[dest]$; (9)
i in {Bay} ; j in {Row}; k in {Tier}\ {06,86}.

$D_{ij1k1}[dest] \leq D_{ij2k2}[dest]$; (10)
case left shutter: i in {odd-Bay} \ {22,24}; $j1$ in {02,04,06}; $j2$ in {02,04}; $k1$ in {82}; $k2$ in {06};
case central shutter: i in {odd-Bay} \ {22,24}; $j1$ in {01,02}; $j2$ in {Row}; $k1$ in {82}; $k2$ in {06};
case right shutter: i in {odd-Bay}\ {23,25}; $j1$ in {02,04,06}; $j2$ in {02,04}; $k1$ in {82}; $k2$ in {06}.

Distribution of the containers. Such constraints are related to a proper weight distribution that is the basic condition for a good stowage. Note that it is possible to check the stability of the ship by using some mathematical expressions only when the ship is completely loaded. However, there are some rules, mainly derived from the experience and the knowledge of the load, that help in establishing the most suitable way for stowing the containers in such a way that there will be a well balanced distribution of the weights. In practice, the most heavy containers are located in the hold, while the others are located in the upper deck.

After the loading operations we have to verify the cross and vertical equilibrium conditions, that is the weight on the right side of the ship, including rows 01-03 of the hold and rows 01-03-05 of the upper deck, must be equal, within a given tolerance value, to the weight on the left side of the ship, including rows 02-04 of the hold and rows 02-04-06 of the upper deck, while the weight on each tier must be greater than the weight on the tier immediately over it (condition 11).

$$\sum_{ij} D_{ij06}\,[weight] \leq \sum_{ij04}[weight] \leq \sum_{ij02}[weight] \qquad (11)$$

case hold: i in {Bay} \ {22,24}; j in {Row};

$$\sum_{ij} D_{ij86}\,[weight] \leq \sum_{ij84}[weight] \leq \sum_{ij} D_{ij82}\,[weight]$$

case upper deck: i in {Bay}; j in {Row}.

Finally, note that a ship must be loaded in such a way to be able to travel independently on the weather conditions and that the stability constraint must be satisfied also after some possible unloads at intermediate destinations. In fact, if some containers are moved from only one side of the ship we would have an unbalanced load on the other side of the ship.

A part from the specific constraints given above, we also have to give the basic constraints of this combinatorial problem. First of all we have to ask that a container is assigned to only one cell and that each cell can store at most one container, that are the constraints of AP. Then we have to consider the knapsack constraint, that is to force to total weight of the container to be stowed in the ship to be not greater that the maximum capacity Q of the ship, that is:

$$\sum_{ijk}\,[weight] \leq Q \qquad (12)$$

5. The resolution approach

To find a solution of the present problem means to assign values to all variables that satisfy the constraints. The search for a solution stops when all variables are instantiated, that is when their domain is reduced to only one value. Unfortunately, general approaches for solving CSPs are not efficient from a computational point of view [6]. Efforts for making more practical the process of finding a CSP solution are usually devoted to the development of heuristics tailored for a specific application. We propose a CSP approach consisting of two main phases, namely domain reduction and value generation. In the reduction process the constraints are used for tightening the feasible region of the problem throughout the deletion of those values that do not allow any solution; this reduction implies temporary assignments to variables that enable in turn, by making active the related constraints, the reduction of the current search space.

The domain reduction is here performed by removing from each variable domain D_{ijk} those values that are not compatible with the type, size and weight of the containers to be possibly stowed in the corresponding cell. Note that as soon as a constraint is analysed the domain of all the involved variables are reduced and consequently the new values of the restricted domain are analysed for feasibility. This «*constraints propagation*» mechanism plays a crucial role in improving the computational performance of a CSP algorithm and in the determination of a solution since it allows a fast reduction of the variables' domain deleting from them those values that are not globally consistent and that cannot belong to any feasible solution of the problem. If the domain of a variable becomes empty then the current partial solution cannot originate any feasible solution and a backtrack is required. Analogously, if the domain of a variable is reduced to only one value then the variable is immediately instantiated. To be more precise, let x_{ijk} be the last variable for which a consistent value in D_{ijk} has been found and let x'_{ijk} be the next selected variable. The first value in D'_{ijk} such that constraints are satisfied is assigned to x'_{ijk}; if this assignment is not consistent a backtrack is performed, looking for a new value for x_{ijk} only among possible consistent values instead of proceeding in an enumerative way.

The constraint propagation mechanism stops whenever there is not any other possibility of reducing variables' domain and all the constraints are satisfied.

For obtaining a solution we then perform the «*generation phase*»,

where we have to select a variable, to choose a value in its domain and to temporary assign such value to it. If the value satisfies all constraints the variable is considered to be instantiated and a new variable is chosen for instantiation, otherwise a backtrack is performed.

To find an efficient ordering rule for selecting variables for their instantiation is an essential point. The idea is to select first the variable that would minimise the number of backtracks. In the present case our first ordering rule has been to select first the variable having the smallest domain, but it did not result to be a good strategy. In fact, using this selection criterion the variables related to the hold located in the first and second tiers were selected first since they allow only the stowage of containers of 20'; then, after instantiating them in a bow-stern order always a backtrack were required when the cells located in the upper deck had to be considered. Finally, we decided to select first those variables involved in the greater number of constraints, considering first the weights constraints 6)-8), 11).

In looking for feasible solutions we analysed a number of alternatives that could result or not into the desired solution, possible the optimal one, when, besides the satisfability of the constraints, we had also to maximise the total number of containers to be loaded and minimise the number of shifts, that is the number of containers that it necessary to move for loading / unloading other containers. In fact, shifts strongly affect both the time and costs of the handling operations. It is worth mentioning that the costs that are considered as our objective function are directly proportional to the number of harbours to be visited and the number of bays of the ship while inversely proportional to the tiers; in fact, the minimisation of the empty moves of the containers can be obtained by stowing the containers having priority of destination into bays located closed to the exit and into higher tiers that are more easily reachable.

The combined used of a backtracking algorithm and the Branch & Bound in the optimisation phase has allowed us to find good feasible solutions in a reasonable computational time (within 10 minutes of CPU time on a IBM PC 486 for istances with 350 containers of 2 types and 3 destinations).

Note that the backtracking algorithm uses the same criterion than Branch & Bound, that is the problem is split into different sub-problems while generating a search tree, where nodes are partial solutions of the problem, and adding some bounds for reducing as much as possible the number of nodes to be analysed. The difference between the two combined approaches used in our work is that the backtracking tree is

explored in its depth while in the Branch & Bound tree we analyse the nodes at the same level.

The above approach has been implemented in LPA Prolog under Windows 3.1 with some procedures related to the definition of domain D_{ijk} written in C language (more details about implementation aspects can be found in a forthcoming paper)

As a concluding remark we can say that the development of a DSS based on the proposed CSP approach for solving MBP can be considered very promising also in understanding the problem and in its definition phase.

References

[1] Atkins W.H.: *Modern Marine Terminal Operations and Management*, Boyle, Oakland, 1991.

[2] Bischoff E.E., Mariott M.D.: A comparative evaluation of heuristics for container loading, *European Journal of Operational Research* 44 (1990) 267-276.

[3] Crainic T.G., Gendreau M., Dejax P.: Dynamic and stochastic modesl for the allocation of empty containers, *Operations Research* 41 (1993) 102-126.

[4] Hee K.M, Wijbrands R.J.: Decision support system for container terminal planning, *European Journal of Operational Research* 34 (1988) 262-272.

[5] Martello S., Toth P.: *Knapsack Problems*, John Wiley, 1992.

[6] Nadel B.: Constraint Satisfaction Algorithms, *Computational Intelligence*, 5, 1989.

[7] Thomas B.J.: *Management of port maintenance: a review of current problems and practices,* H.M.S.O, London, 1989.