Title
Reducing Ship Turn-Around Time Using Double-Cycling

Authors
Goodchild, A. V.
Daganzo, C. F.

Publication Date
2004-05-01

Institute of Transportation Studies
University of California at Berkeley

Reducing Ship Turn-Around Time Using Double-Cycling

A. V. Goodchild and C. F. Daganzo

RESEARCH REPORT
UCB-ITS-RR-2004-4

May 2004
ISSN 0192 4095

Reducing Ship Turn-Around Time Using Double-Cycling

A. V. Goodchild (*)
Department of Civil and Environmental Engineering, University of California at Berkeley
416 G McLaughlin Hall, University of California at Berkeley, Berkeley, CA 94720
phone: (510) 525-7897, fax: (510) 643-8919
anne_g@uclink.berkeley.edu

C. F. Daganzo
Department of Civil and Environmental Engineering, University of California at Berkeley
416 A McLaughlin Hall, University of California at Berkeley, Berkeley, CA 94720
phone: (510) 642-3853, fax: (510) 643-8919
daganzo@ce.berkeley.edu

Document length – 4703 words + 11 figures
Submission date – August 1, 2003

*- corresponding author

ABSTRACT
Double cycling improves efficiency by unloading and loading a ship simultaneously; using wasted crane moves to transport containers. This paper demonstrates that double cycling can reduce ship turn-around time through this efficiency improvement. The paper describes the nature of the double cycling problem and, for a given loading-plan, quantifies the benefits using a greedy algorithm. The relationship between the benefits of double cycling, and the problem parameters is analyzed using simple formulas and a simulation program. This paper demonstrates that double cycling can create significant efficiency gains, and should benefit any party interested in improving port operations, and reducing the cost of container shipping.

1 INTRODUCTION

To date much research has focused on minimizing container ship turn-around time or the amount of time a ship spends (unproductive) in port. The research has focused on several aspects of the problem including optimizing ship loading-plans, the use of landside equipment, and berth scheduling (e.g. (*1*), (*2*), (*3*)). In addition to this academic research, there is much unpublished work in the industry and commercial software is readily available to optimize loading-plans (*4*),(*5*). To date, double cycling has not been addressed in the academic literature. This paper fills that gap by analyzing the opportunity to reduce ship turn-around time using double cycling.

Ship turn-around time is dominated by the time necessary to unload and load containers. When unloading and loading a ship, most cranes spend only half of their moves carrying a container. During unloading, the crane is empty when moving to the ship (FIGURE 1a). During loading, the crane is empty when returning to the dock. Double cycling is the practice of using these "empty" moves to carry a container, thus making the crane more productive, and reducing turn-around time (FIGURE 1b).

With current single cycling or status quo methods, the number of moves necessary to turn-around the ship is fixed, and does not depend on the order in which the crane operates on the ship's columns. With double cycling, however, the number of moves depends on the order of operations. Therefore, the problem of double cycling is one of scheduling jobs, or finding the order in which to operate on the columns that minimizes ship turn-around time.

In this paper a move refers to one crane operation, from the ship to shore, or the shore to ship, with or without a container. A cycle is two consecutive moves. This research assumes a crane starts and ends on shore, so a complete unloading and loading operation involves an even number of moves and a whole number of cycles.

The results of this work are generic in nature and will yield useful insights into the potential for double cycling to reduce turn-around time. The problem has been simplified greatly in this analysis because attempting to model every detail of the problem would cause the results to be less revealing of the big picture. Specific problems may need to be addressed on an individual basis, but the results presented here will provide insight into the potential of double cycling to reduce ship turn-around time.

This paper uses a greedy strategy to analyze double cycling's benefits. At first the analysis is done on a simplified version of the problem, but it is then extended to consider many real-world complications. The analysis allows us to comment on the relationships between ship design, loading-plans, flexibility, size of hatch-coverings, and double cycling opportunities.

Section 2 describes the nature of the problem and provides a simple example for explanatory purposes. Section 3 develops bounds on the number of cycles necessary to turn-around a ship using double cycling. Section 4 details the results provided by a

problem-specific simulation program. Extensions to the basic problem are covered in sections 5 and 6; fixed ordering, and deck hatches.

2 NATURE OF THE PROBLEM

The layout of containers on a ship can be modeled as a 3-dimensional matrix. Containers are stacked on top of one another, and arranged in rows along the length of the ship. FIGURE 2 gives a top view and side view of such a ship. It is assumed that dockside containers for loading are ready when required, and that containers being unloaded can be easily removed from the immediate area. In the initial analysis it will be assumed that ships lack hatch-coverings, or doors on the deck that separate above deck and below deck storage. This assumption will be revisited in section 6 of this paper.

Consider the case where a large container ship arrives in a port with a set of containers on board to be unloaded and a loading-plan for containers to be loaded. The loading-plan indicates the placement of containers on the ship. Given are u_c and l_c, the number of containers to be unloaded and loaded, respectively, in each column, labeled c. FIGURE 3 is an example problem that will be used throughout the paper. Note that if any "rehandles" are necessary, these are included in the total number of imports and exports. A rehandle is a container than must be moved to access containers below it, but will then be stowed again before the ship departs. Thus, in FIGURE 3, $u_A = 3$ and $l_A = 2$.

The set of column labels, c, is called S. An ordering of these columns can be described by a permutation function, Π. A permutation is a one-to-one correspondence between $i \in \{1,\ldots,N\}$ and S such that $\Pi(i) = c$, or $i = \Pi^{-1}(c)$.

Turn-around time, which is assumed to be proportional to the number of cycles, will be represented by the variable w. We shall ignore for the moment the delays caused by moving the crane sideways during each cycle and will return to this issue later. The general method for unloading and loading the ship with double-cycling is described below:

- Choose a permutation, Π. Unload all containers in the first column, then unload all containers in the second column, proceed in this fashion until all columns have been unloaded.
- Loading can begin as soon as at least one column has been unloaded or is empty. Loading can proceed in any unloaded or empty column until loading operations are complete. If there is no column available, the loading operations must wait. This is called blocking delay.

2.1 Example

Consider the ship of FIGURE 3. FIGURE 4(a) is a queueing diagram for a single cycle operation where columns are handled in the order {A,B,C,D} both for loading and unloading. Note that loading operations must wait until time $w = 10$, when unloading is finished. If double cycling was allowed we could have started loading as early as $w = 4$, using the same sequence, {A,B,C,D} for unloading and loading. The result it shown in

FIGURE 4(b). This requires 14 cycles. The same number of cycles is obviously obtained if we start loading each column as early as possible, as in FIGURE 4(c). This introduces some blocking but does not change the completion time, which is still 14 cycles. With double-cycling, however, the completion time can depend on the sequence, as illustrated in FIGURE 4(c). In this case loading operations can be started as early as $w = 3$ without any blocking delay, and the completion time is $w = 13$.

An objective of this paper is to determine loading and unloading sequences that minimize the completion time. Notice that in the example the same sequences have been used for loading and unloading containers. For the most part, this will be assumed throughout the paper. The assumption is reasonable because is ensures that the number of lateral moves a crane must make does not get excessively large since the crane would be working on nearby columns most of the time. Further, it will be shown that this extra restriction does not change the optimum turn-around time significantly for large problems.

3 BOUNDS ON THE NUMBER OF CYCLES

A lower bound is now developed for the number of cycles necessary to unload and load a ship using the status quo and double cycling techniques. It is <u>not</u> assumed in the development of this bound that the loading and unloading permutations, (Π and Π‘) are the same. Define:

$$Y = \sum_{i=1}^{N} u_{\Pi'(i)} = \sum_{c \in S} u_c \ ,$$

$$\Lambda = \sum_{i=1}^{N} l_{\Pi(i)} = \sum_{c \in S} l_c \ .$$

Recall from FIGURE 4(a) that using singe cycling, the number of cycles necessary to turn a ship around is given by:

$$Y + \Lambda \ . \tag{1}$$

This is intuitive; one cycle for each container. Notice this is invariant to the ordering.

For double cycling, with a specific loading permutation, Π, and unloading permutation, Π', the number of cycles must be at least $\Lambda + u_{\Pi'(1)}$ which satisfies:

$$\Lambda + u_{\Pi'(1)} \geq \Lambda + \min_c(u_c) \ . \tag{2}$$

Then, the right hand side of (2) is a general lower bound for double cycling. It is also a lower bound if we force the same permutation for loads and unloads. In other words, if we define w_1^* as the number of cycles with 1 optimal permutation for both loading and

unloading and w_2^* as the number of cycles with 2 optimal permutations (1 optimal for unloading and 1 optimal for loading), we can write:

$$w_1^* \geq w_2^* \geq \Lambda + \min_c(u_c). \quad (3)$$

It is proven below that if $\Lambda \geq Y$, a tight upper bound on the number of cycles is $\Lambda + \max_c(u_c)$. This is true even if we use the same permutation for loads and unloads. Thus,

$$w_1^* \leq \Lambda + \max_c(u_c) \quad (if\ \Lambda \geq Y). \quad (4)$$

This tight upper bound is obtained by bounding from above the cycles of a sub-optimal solution where the loading and unloading sequences are given by the same "greedy" permutation, G. This greedy permutation is obtained by ordering the columns in ascending order of the variable d_c where:

$$d_c = l_c - u_c \qquad (if\ \Lambda \geq Y), \quad (5a)$$
$$d_c = u_c - l_c \qquad (if\ \Lambda < Y). \quad (5b)$$

The rationale for (5a), which we are about to use, is that we want the unloading operations to run ahead of the loading operations as much as possible. Associated with this strategy, we can define cumulative curves of columns completed, $\mathcal{U}(w)$ and $\mathcal{L}(w)$, as in the top and bottom parts of FIGURE 5. Define as well the horizontal steps of these curves, $U_i = u_{G(i)}$ and $L_i = l_{G(i)}$; see FIGURE 5. Finally, we define $\mathcal{L}^{-1}(i)$ and $\mathcal{U}^{1}(i)$ for integer i to be at the left of each step (as shown by the solid dots of FIGURE 5 and the notations along the bottom of the abscissas axis). Note that $\mathcal{U}(Y) = \mathcal{L}(\Lambda) = N$. We are now in a position to prove (4).

Proof of (4):
Define w_G as the number of cycles with G. Since $w_G \geq w_1^*$ it suffices to show that $\Lambda + \max_c(u_c) \geq w_G$. Consider a horizontal shift, L, to curve $\mathcal{L}$, that will ensure curve $\mathcal{L}$ is to the right of curve $\mathcal{U}$ (as in the dotted curve in the top of FIGURE 5), i.e. that $\mathcal{L}^{-1}(i\text{-}1) + L \geq \mathcal{U}^{1}(i)$ for $i = 1,...N$. Obviously the least possible delay, L_0, is $\max_i(\mathcal{U}^{1}(i) - \mathcal{L}^{-1}(i\text{-}1)) = \max_i(U_i + (\mathcal{U}^{1}(i\text{-}1) - \mathcal{L}^{-1}(i\text{-}1))$. Since $\Lambda \geq Y$, the greedy strategy satisfies $(\mathcal{U}^{1}(i\text{-}1) - \mathcal{L}^{-1}(i\text{-}1)) \leq 0\ \forall\ i$. Hence $L_0 \leq \max_i(U_i)$. Therefore $\max_i(U_i) + \Lambda$ is an upper bound to the number of cycles. QED.

In summary, we have shown that if $\Lambda \geq Y$, then

$$\Lambda + \min_c(u_c) \leq w_2^* \leq w_1^* \leq w_G \leq \Lambda + \max_c(u_c). \quad (6a)$$

It will now be shown that if $Y > \Lambda$ then the symmetric result obtained by interchanging load and unloads is obtained, i.e.:

$$\mathrm{Y} + \min_c(l_c) \le w_2^* \le w_1^* \le w_G \le \mathrm{Y} + \max_c(l_c), \tag{6b}$$

where G in this case is defined by (5b). Hence, if we define

$u' = \min_c(u_c), l' = \min_c(l_c), u^* = \max_c(u_c), \textit{and } l^* = \max_c(l_c)$, we can write:

$$\max(\Lambda, \mathrm{Y}) + \min(u', l') \le w_2^* \le w_1^* \le w_G \le \max(\Lambda, \mathrm{Y}) + \max(u^*, l^*). \tag{7}$$

Clearly, for ships where $(\Lambda,\mathrm{Y}) >> (u^*, l^*)$, the greedy strategy is very close to both the upper and lower bound since all the members of (7) are relatively close to $\max(\Lambda,\mathrm{Y})$. In this sense, the greedy strategy is asymptotically optimal. How quickly optimality is approached will be demonstrated in the next section.

Proof of (6b):
That (6b) is true should be obvious by symmetry. If one were to record the process of unloading and loading the ship, and then play this recording in reverse, the reversed movie would display a sequence of operations with the same total time for a problem in which the role of imports and exports is switched. Thus, for every problem instance with exports greater than imports, there is a dual instance where imports are greater than exports. FIGURE 6(b) shows the reverse movie of FIGURE 6(a). Note that the greedy strategy of FIGURE 6(a) continues to be greedy in FIGURE 6(b) (with loads and unloads reversed), and that everything said up to this point, including the bounds, continues to hold with time running backward. Note that in the case of FIGURE 6(b) the greedy strategy (with time running backward) implies a reverse ordering of $\{d_c\}$, as specified in (5b), thus (6b) holds. QED

4 SIMULATION

A simulation program was developed that includes a meta-heuristic optimization module to further understand how the greedy algorithm would perform relative to the upper and lower bounds, and to the meta-heuristic benchmark. It uses simulated annealing to search for the minimum turn-around time. The same ordering was assumed for imports and exports. The program provides turn-around times (in number of moves) using the status quo, the greedy algorithm, simulated annealing, and a fixed ordering strategy (the relevance of this result is discussed in section 5).

The simulator can generate data for many different ship designs and market conditions (imports and exports, origins and destinations, and a range of realistic N's). Values of u_c and l_c are generated by sampling from a beta distribution, and altering the parameters p and q. Shallow, deep, large, and small container ships can be modeled by altering the number of columns, and the maximum column height (h).

The probability function of the beta distribution is $P(x) = \frac{(1-x)^{q-1}x^{p-1}}{B(p,q)}$ where B is the beta function $B(p,q) = \frac{(p-1)!(q-1)!}{(p+q-1)!}$. The mean and variance of the beta distribution are $\mu = \frac{p}{p+q}$, $\sigma^2 = \frac{pq}{(p+q)^2(p+q+1)}$. If we define each draw from the beta distribution as x. The number of containers for import or export in one column is then $[xh]^+$. Parameters p, q, and h are set separately for imports and exports. Notice some columns will have 0 containers.

Each point in FIGURE 7,FIGURE **8**, and FIGURE 10 are the result of one simulation run. These examples show a large range of ship sizes (4 to 600 or 1000 columns). Ships of less than 10 columns can be considered as individual rows or individual rows of hatches of a larger ship. As expected, the simulated annealing and greedy turn-around time provide significant savings over single cycling.

FIGURE 7(a) shows the percentage difference between the simulated annealing turn-around time and the upper and lower bounds for varying number of columns. Notice the simulated annealing turn-around time is very close to the lower bound and is only significantly different from the lower bound for very small ships. Recall that the lower bound is not necessarily achievable.

FIGURE 7(b) shows the percentage difference between the *greedy* strategy and the upper and lower bounds. It shows a very similar distribution of data to FIGURE 7(a). For ships with many columns, the upper and lower bounds are both good estimates for the total turn-around time using the greedy strategy.

Although the greedy strategy does not always provide the shortest turn-around time, the results from the simulation show the difference between the greedy strategy turn-around time and the benchmark is small (FIGURE 8(a)), and is negligible even for medium sized ships.

FIGURE 8(b) shows using the greedy strategy can reduce turn-around time by as much as 50% over the status quo. The largest savings occur when the number of unloads and loads are similar and column heights are consistent. For small ships with very large or very small load/unload ratios and/or very inconsistent column heights, the savings are less. Notice, however, that the performance improves even for these “lopsided” ships as the number of columns increases.

5 FIXED ORDERING

The greedy algorithm may suggest a sequence of operations that requires an impractical set of moves around the ship. As an example, see FIGURE 9(a) which shows the number of each step in the sequence. Operating using this sequence requires double cycling between one column and another many columns away. It may not be possible to operate

on the ship in this order due to crew availability, or impractical due to the time necessary to move the crane laterally along the ship. Therefore we consider here operating on a fixed sequence of columns, determined on the basis of proximity (FIGURE 9(b)).

FIGURE 10 shows the results of the simulation for the fixed ordering algorithm as compared to the greedy algorithm and the simple approximation. For large ships (over 200 columns of containers) the greedy algorithm, restricted ordering, and simple approximation are very close. The reduction in turn-around time due to double cycling is robust to restrictions on the sequence of operations.

6 HATCHES

Throughout the analysis presented above, it has been assumed that ships lack hatch coverings. This is a distinct simplification of reality; as most container ships currently have hatch coverings, as shown in FIGURE 11.

Hatches may change the nature of the problem already addressed, because the columns previously considered are no longer independent. To access the containers below a hatch all containers must be unloaded from above the hatch, and before loading containers atop a hatch all columns of containers below the hatch must be loaded. This section considers how the introduction of hatches will impact turn-around time when using double cycling.

The following algorithm (the greedy hatch strategy) will be considered. As in the hatchless case, assume that each column on the ship is given an initial label. To carry out the strategy it is necessary to:

1. Order the *hatches* using a greedy strategy. This is accomplished through the same method as for the hatchless case if the hatches are treated as columns, considering only the containers atop the hatches.
2. Order the *columns* within each hatch using a greedy strategy, considering only the containers below deck.

The algorithm is then carried out as follows:

1. Apply the greedy strategy to the containers above deck, treating hatches as columns, pausing each time all containers above hatch *i* have been removed.
2. During the *i*th pause, unload and load the containers below the *i*th hatch using double cycling.

Of course this method may not provide the shortest turn-around time, but the value of its simplicity is that parts of the unloading and loading process are equivalent to the hatchless ship problem already addressed. Each piece below a hatch is equivalent to a hatchless ship, and the containers above a hatch to a column of a hatchless ship. These relationships will allow us to the use the analysis of the hatchless ship to develop bounds for the hatched case. First it is necessary to define some notation.

h – hatch label
S_h – the set of columns for hatch h
N_h – the number of columns of hatch h
H - the set of hatches
F – the number of hatches

$\overline{u}_{hc}$ - the number of containers to unload below hatch h in column $c \in S_h$
$\underline{u}_{hc}$ - the number of containers to unload above hatch h in column $c \in S_h$
$\overline{l}_{hc}$ - the number of containers to load below hatch h in column $c \in S_h$
$\underline{l}_{hc}$ - the number of containers to load above hatch h in column $c \in S_h$

$\overline{u}_h = \sum_{c \in S_h} \overline{u}_{hc}$ - containers to unload below hatch h
$\underline{u}_h = \sum_{c \in S_h} \underline{u}_{hc}$ - containers to unload above hatch h
$\overline{l}_h = \sum_{c \in S_h} \overline{l}_{hc}$ - containers to load below hatch h
$\underline{l}_h = \sum_{c \in S_h} \underline{l}_{hc}$ - containers to load above hatch h

$\overline{Y} = \sum_{h \in H} \sum_{c \in S_h} \overline{u}_{hc}$ - containers for unloading below deck
$\underline{Y} = \sum_{h \in H} \sum_{c \in S_h} \underline{u}_{hc}$ - containers for unloading above deck
$\overline{\Lambda} = \sum_{h \in H} \sum_{c \in S_h} \overline{l}_{hc}$ - containers for loading below deck
$\underline{\Lambda} = \sum_{h \in H} \sum_{c \in S_h} \underline{l}_{hc}$ - containers for loading above deck

w_A = the number of cycles above deck
w_B = the number of cycles below deck
$w_{B,h}$ = the number of cycles below hatch h

Also define the greedy permutation for hatches;
$P(i) = h$; $P^{-1}(h) = i$ for $i = 1...F$; $h \in H$

and the greedy permutation for containers below hatch, $h \in H$:
$\Pi(i \mid h) = c \in S_h$; $\Pi^{-1}(c|h) = i$ for $c \in S_h$, $\{i = 1...N_h\}$

Note that:
$Y = \overline{Y} + \underline{Y}$ - containers for unloading
$\Lambda = \overline{\Lambda} + \underline{\Lambda}$ - containers for loading

It will now be shown that an upper bound on the optimum number of cycles is:

$$\sum_{h\in H} \max\{\bar{u}_h, \bar{l}_h\} + \max\{\underline{\Lambda}, \underline{Y}\} + \max_h\{\underline{u}_h, \underline{l}_h\} + \max_{c\in S_h}\{\bar{u}_{hc}, \bar{l}_{hc}\}. \quad (8)$$

Proof of (8):

We know from (7) that w_A = number of cycles for a ship of F columns with data given by $\{\underline{u}_h,\underline{l}_h\} \le \max\{\underline{\Lambda},\underline{Y}\} + \max_h\{\underline{u}_h,\underline{l}_h\}$. Likewise,

$w_B = \sum_{h\in H} w_{B,h} = \sum_{h\in H}$ number of cycles for a ship of N_h columns with data given by $\{\bar{u}_{hc}, \bar{l}_{hc}\} \le \sum_{h\in H} \max\{\bar{u}_h, \bar{l}_h\} + \max_{c\in S_h}\{\bar{u}_{hc}, \bar{l}_{hc}\}$.

Obviously then, the total number of cycles with the algorithm satisfies $w_A + w_B \le \sum_{h\in H} \max\{\bar{u}_h, \bar{l}_h\} + \max\{\underline{\Lambda}, \underline{Y}\} + \max_h\{\underline{u}_h, \underline{l}_h\} + \max_{c\in S_h}\{\bar{u}_{hc}, \bar{l}_{hc}\}$. QED

As in the hatchless case, a lower bound can be determined by assuming there is no blocking during double cycling.

$$\text{Lower bound} = \sum_{h\in H} \max\{\bar{u}_h, \bar{l}_h\} + \max\{\underline{\Lambda}, \underline{Y}\} + \min_h\{\underline{u}_h, \underline{l}_h\} + \min_{c\in S_h}\{\bar{u}_{hc}, \bar{l}_{hc}\}. \quad (9)$$

We have found that the gap between the upper bound and the lower bound is small for large ships, unless the distribution of containers is very unusual. Thus, the simple algorithm is reasonably efficient.

Currently many ship operators are reluctant to operate on more than one hatch in one cycle. Therefore, it may not be feasible to unload atop one hatch, while loading atop another as in the greedy hatch strategy. If cranes can only operate on one hatch at a time, the results from the hatchless case are still applicable but there will be no double cycling above deck. This should but the benefits of double-cycling roughly in half.

7 DISCUSSION

The analysis in this paper has been presented from the perspective of ship turn-around time, but we have ignored the time necessary to make lateral moves along the ship. One could include a penalty for lateral crane moves in the calculation of ship turn-around time, but given the slow nature of lateral crane moves, it is not realistic to think that cranes will be double cycling between one row and another. Therefore, the analysis in this paper is most appropriate for application to just a single row on the ship. Turn-around time can then be considered as the sum of the turn-around time for each row, plus the time necessary to move the crane laterally between one row and the next.

In typical port operations, there is uncertainty as to which containers will be present when the ship is loaded. Although ports will develop a loading-plan, it needs to be flexible to changes that occur just prior to the unloading and loading. For example, a container may not pass security checks, or a shipper may want specific containers expedited. These last minute changes are usually handled on a case-by-case basis. The greedy algorithms are very well suited to this need for flexibility. First, there will be many "free" moves that can be used to load containers onto the ship without increasing turn-around time. There will typically be as many free moves as the difference in imports and exports. Second, the sequence is easy to determine and can be quickly recalculated. Third, the sequence determined by the greedy algorithm is not unique; there are many sequences which will provide the same turn-around time, so there is some opportunity to re-order the sequence if necessary.

As has been shown, the greedy algorithm can significantly reduce the number of cycles necessary to unload and load all relevant containers. This reduction in the number of cycles will translate into a reduction in ship turn-around time. A reduction in turn-around time implies a cost savings for the ship operator, because more of the ship's time can be spent in a productive manner. The extent to which a reduction in the number of cycles affects turn-around time, and the extent to which a reduction in turn-around time affects a ship operating company's profitability will depend on the specifics of the port and ship operator. The reduction in the number of cycles gives an indication of the significant opportunities double cycling provides for cost reduction.

8 CONCLUSIONS

The analysis proves that double cycling presents an exciting and significant opportunity to reduce ship turn-around time. The benefits of double cycling are significant for both hatched and hatchless ships, and are robust to constraints on the sequence of operations. A general approach has been used to study the nature of the problem rather than a detailed approach that would specify the magnitude of the savings in specific cases. With this general approach the results are applicable to a wide range of situations including large or small ships, those with many or few port visits, and a varying number of cranes operating per ship. The method requires only a small number of parameters to understand the magnitude of savings double cycling provides.

9 ACKNOWLEDGEMENT

This research was supported in part by the University of California, Berkeley, Transportation Center.

10 REFERENCES

1. Daganzo, C.F., "The Crane Scheduling Problem", Transportation Research Part B, 23 (3), 159-175 (1989)

2. Imai, A., Nishimura, E., Papadimitriou, S., “Berth Allocation with Service Priority”, Transportation Research Part B, Vol. 37B, no. 5, p. 437-457, June 2003.

3. Kim, K.H., Kim, H.B., **“**The Optimal Sizing of the Storage Space and Handling Facilities for Import Containers”, Transportation Research Part B, Vol. 36B, no. 9, p. 821-835, Nov. 2002.

4. Yau, V., APL, personal conversations 2002-2003

5. Ward, T., JWD/Liftech, personal conversations 2002-2003

11 LIST OF TABLES AND FIGURES

FIGURE 1 (a) Unloading using single cycling **(b)** Unloading and loading with double cycling.

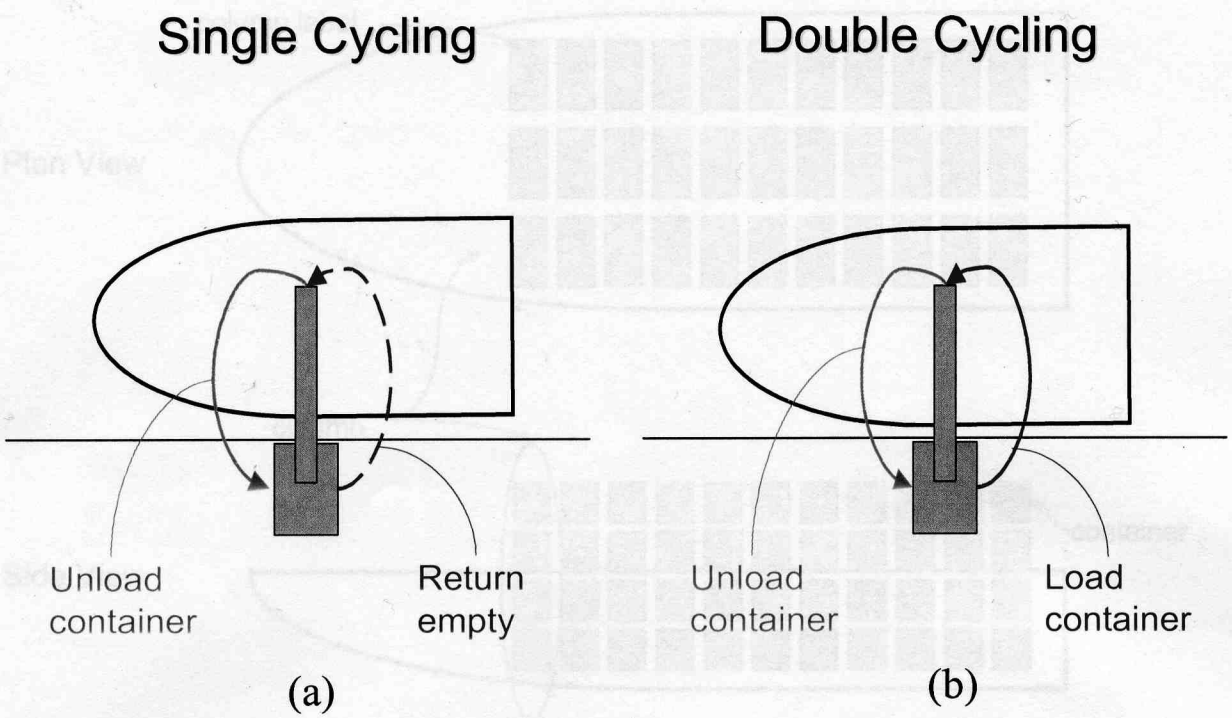


FIGURE 2 Plan and side views of a container ship.

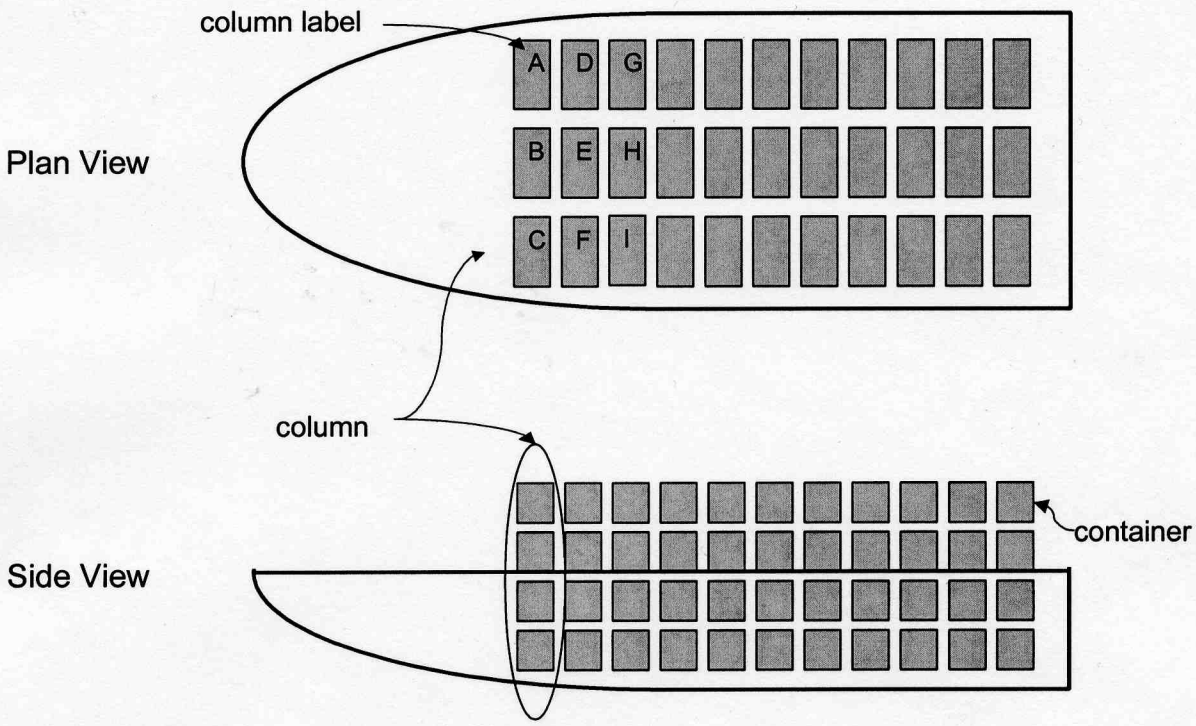


FIGURE 3 Import loading pattern and export loading-plan.

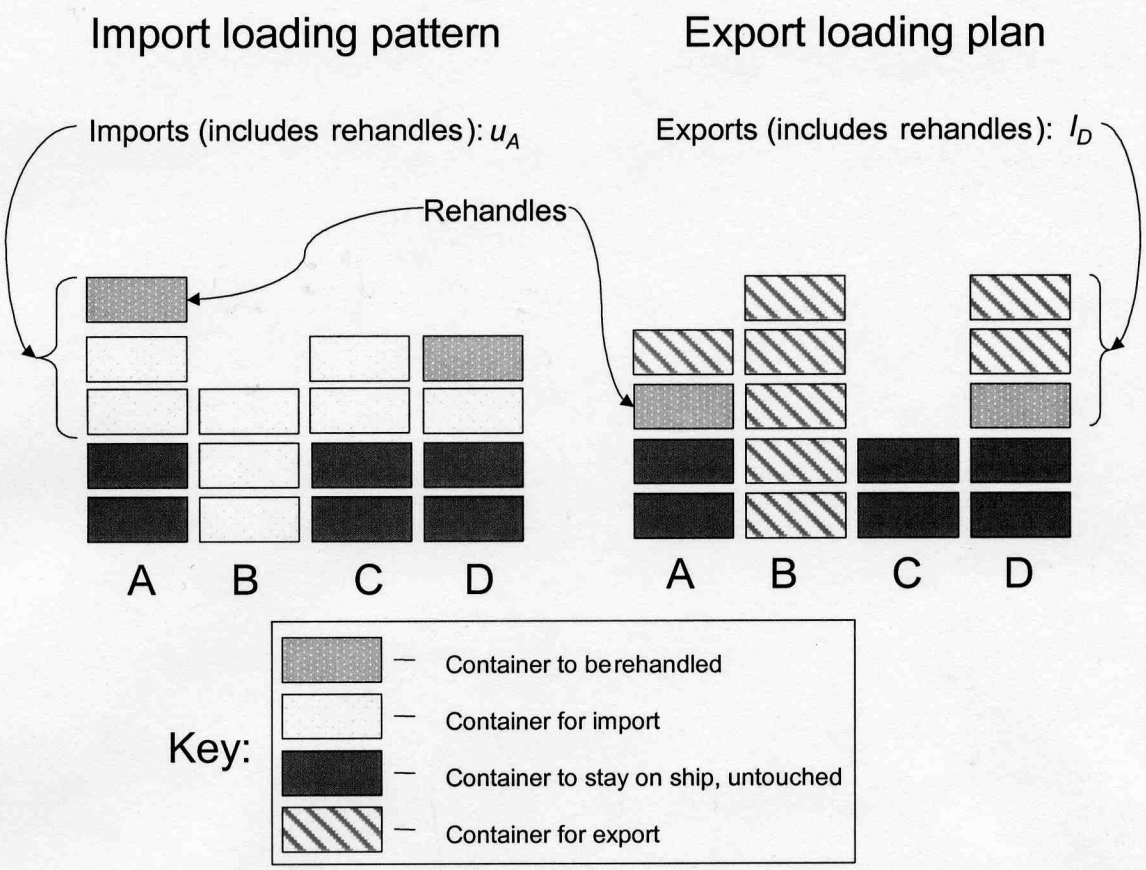


FIGURE 4(a) Turn-around time using the status quo method and ordering $S = \{A,B,C,D\}$, 20 cycles **(b)** A double cycling sequence $S = \{A,B,C,D\}$ starting unloading as soon as possible without blocking delay. Turn-around time is 14 cycles. **(c)** A double cycling sequence $S = \{A,B,C,D\}$ with blocking delay. Turn-around time is 14 cycles. **(d)** A double cycling sequence $S = \{B,A,C,D\}$ without blocking delay. Turn-around time is 13 cycles.

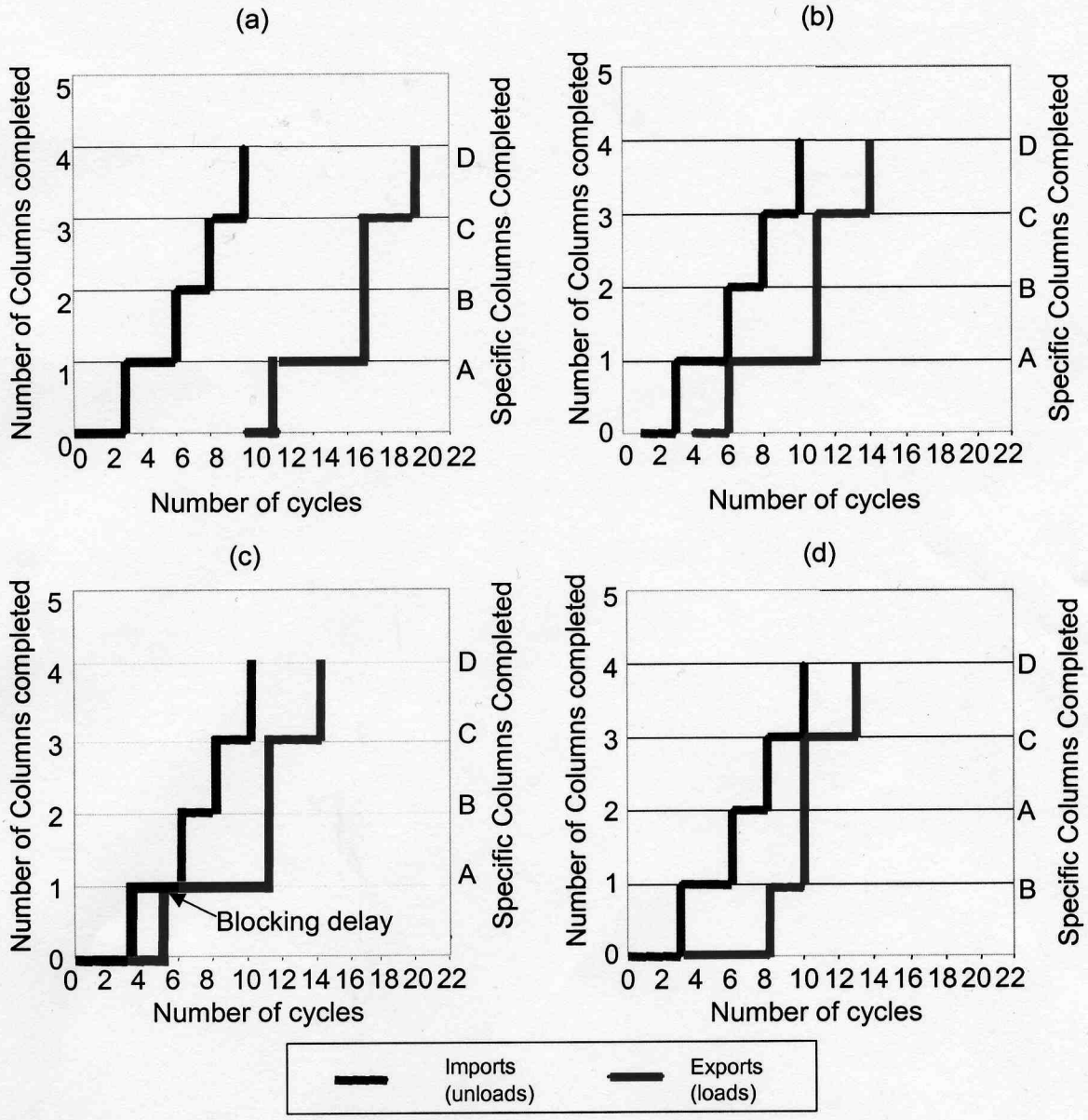
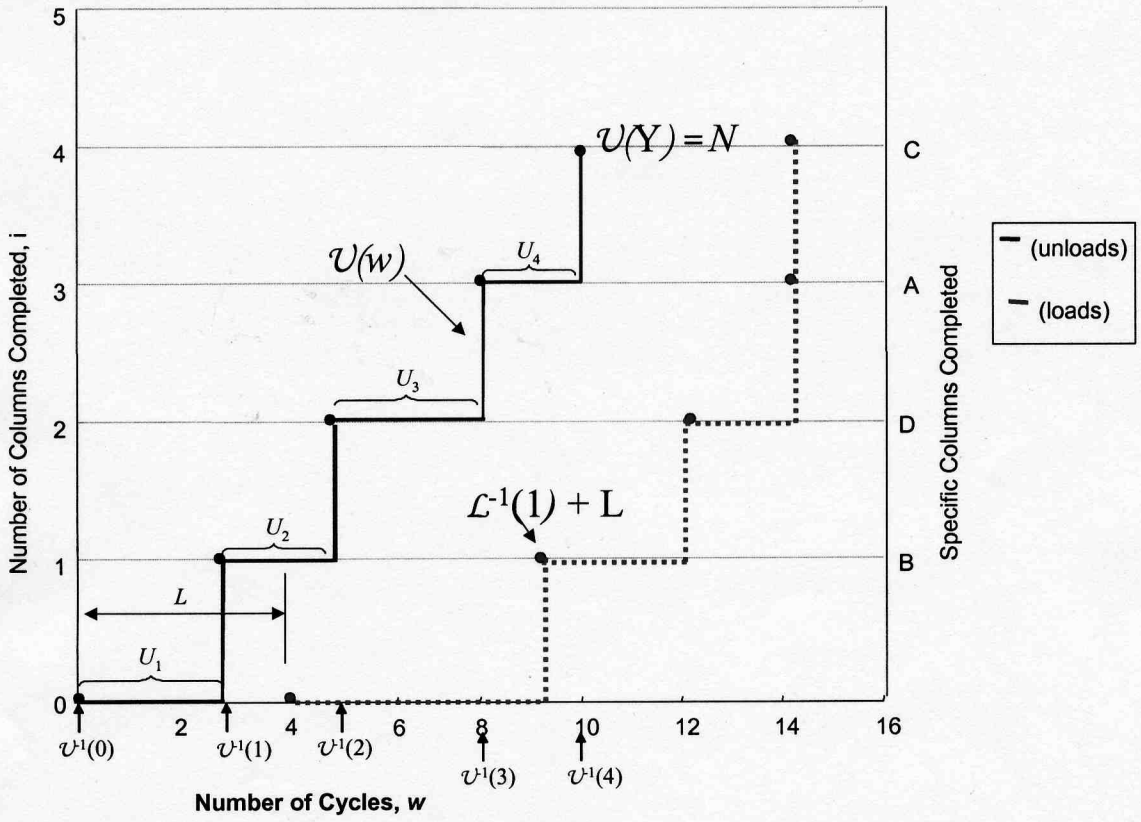


FIGURE 5 The curves $\mathcal{U}$ and $\mathcal{L}$ of the greedy strategy.



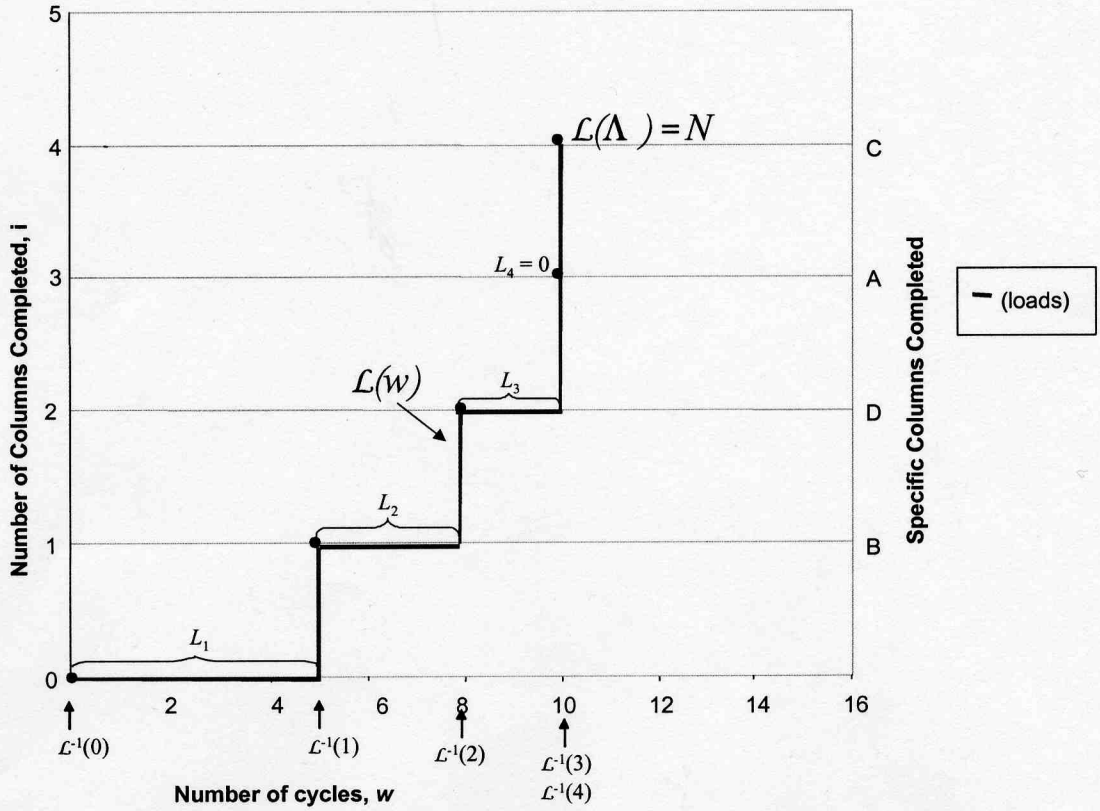


FIGURE 6(a) Definition of the greedy strategy **(b)** Rotated version of part (a).

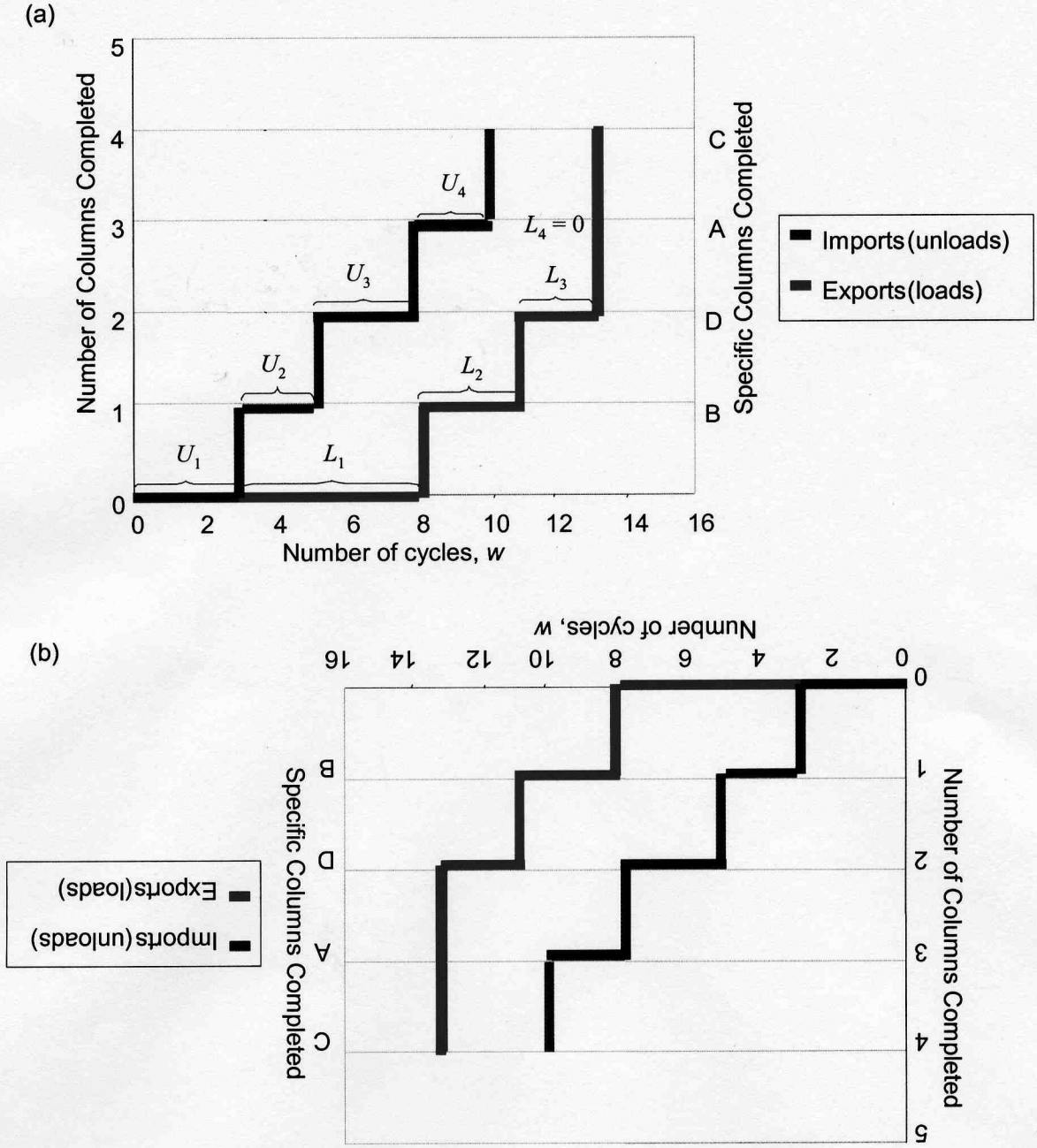
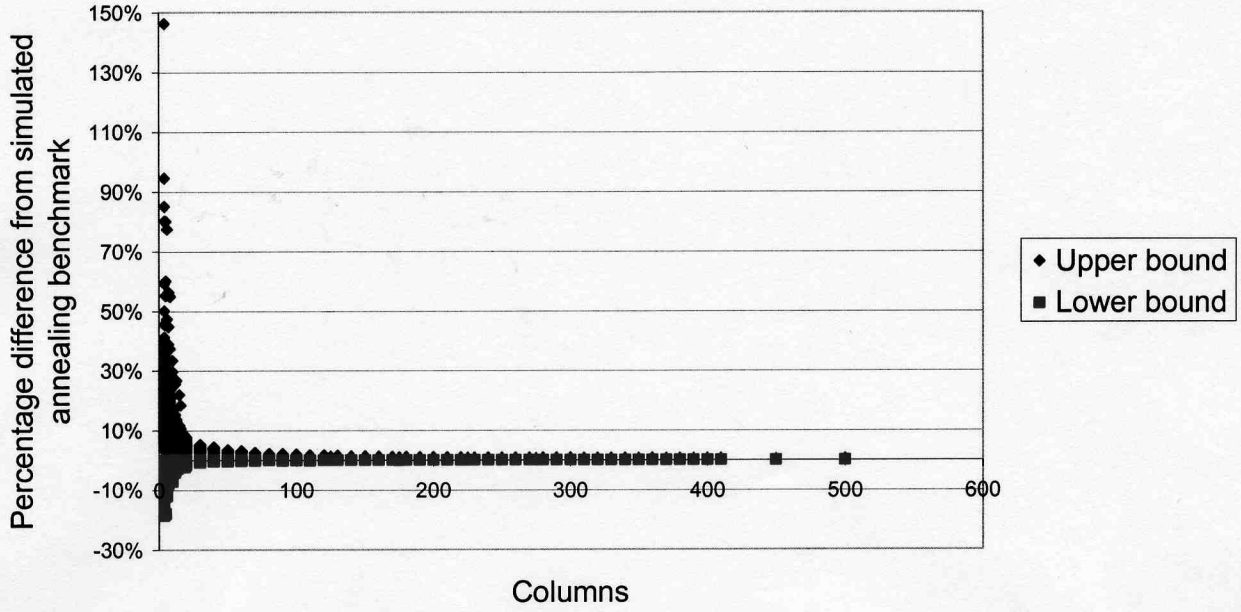


FIGURE 7(a) Percentage difference between upper bound, lower bound and simulated annealing turn-around time **(b)** Percentage difference between greedy strategy turn-around time and the upper and lower bounds.

(a)



(b)

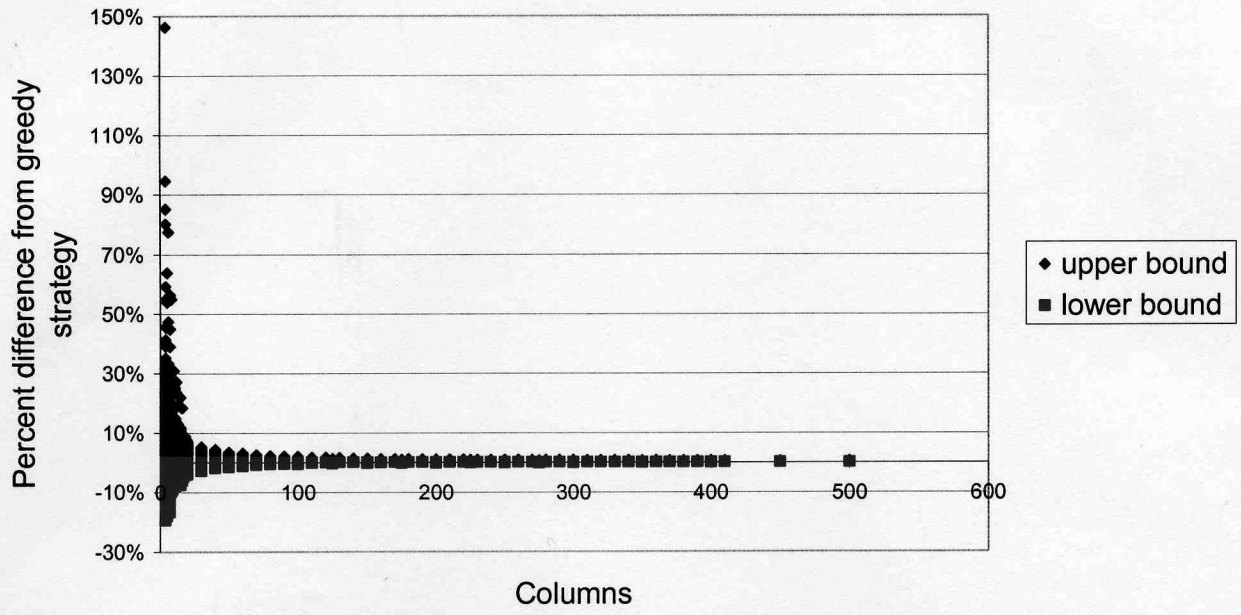
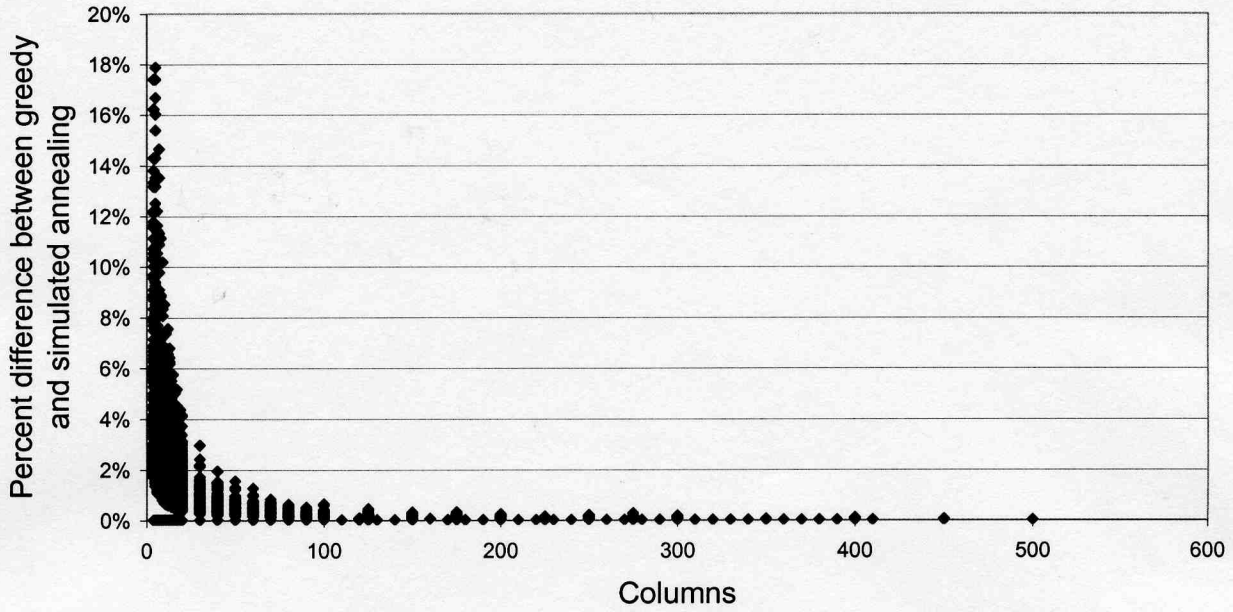


FIGURE 8 (a) Percent difference between the greedy and the simulated annealing turn-around time for varying numbers of columns **(b)** Percentage difference between the greedy strategy and the status quo turn-around time.

(a)



(b)

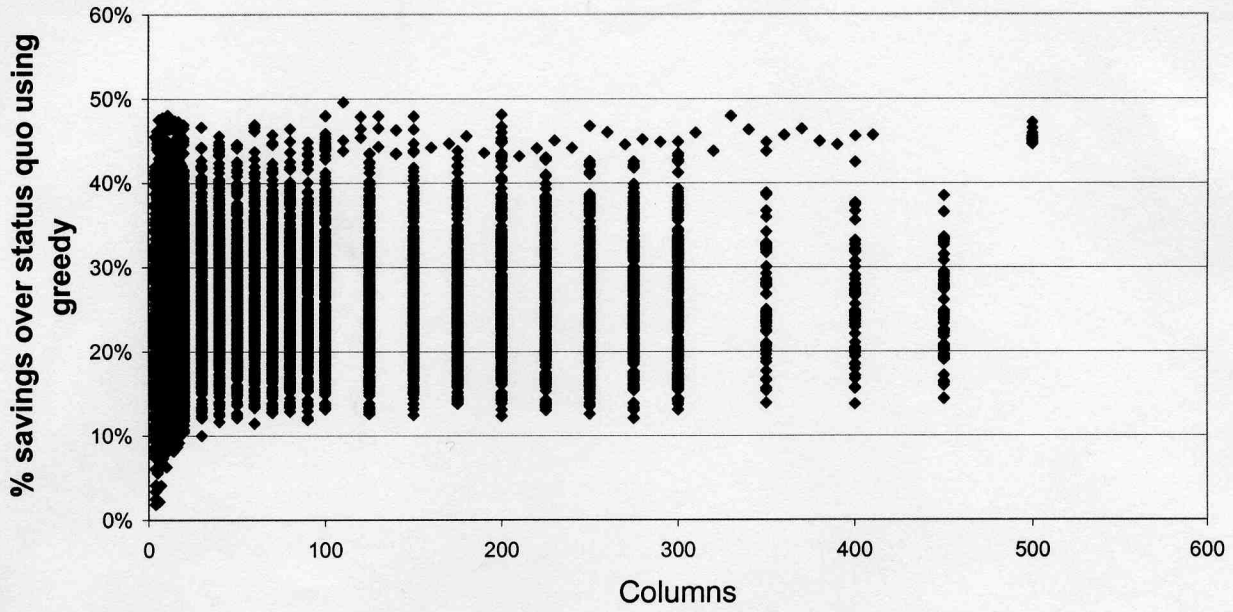


FIGURE 9(a) Sequence of operations as suggested by the greedy algorithm. **(b)** Sequence of operations with restricted ordering.

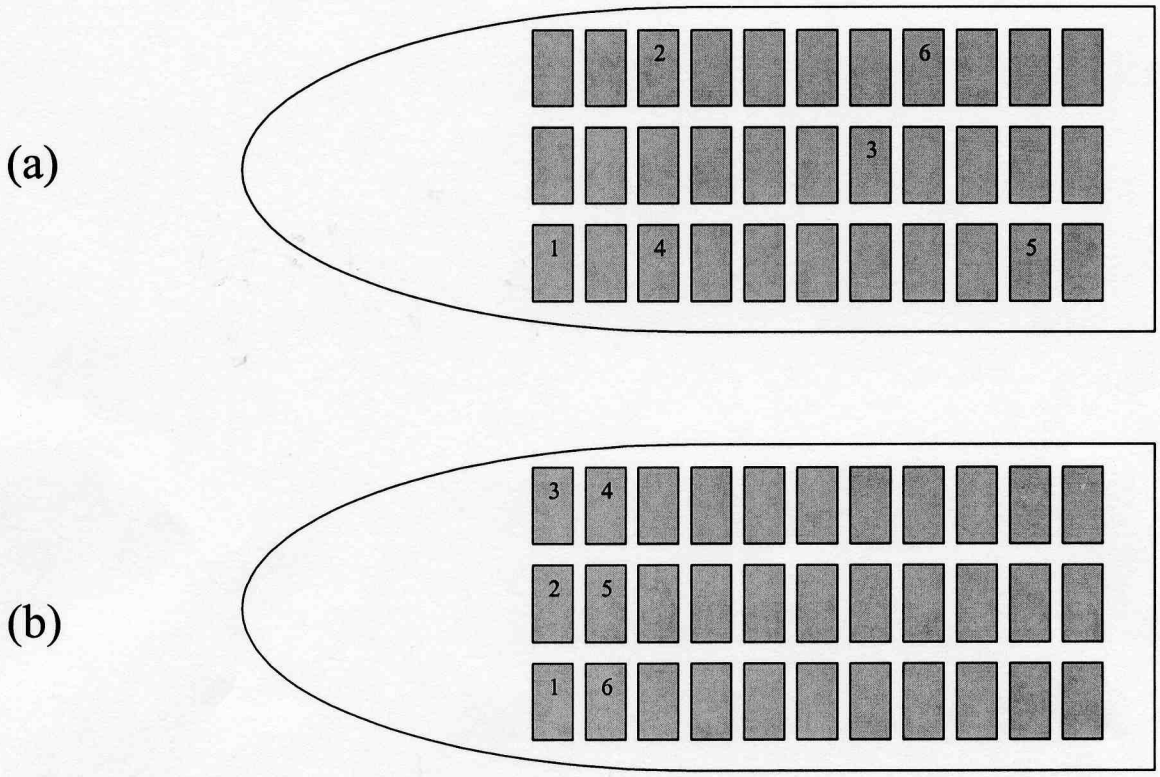


FIGURE 10 Percentage savings over the status quo for the restricted and greedy algorithms, and the approximation based on the expected number of moves.

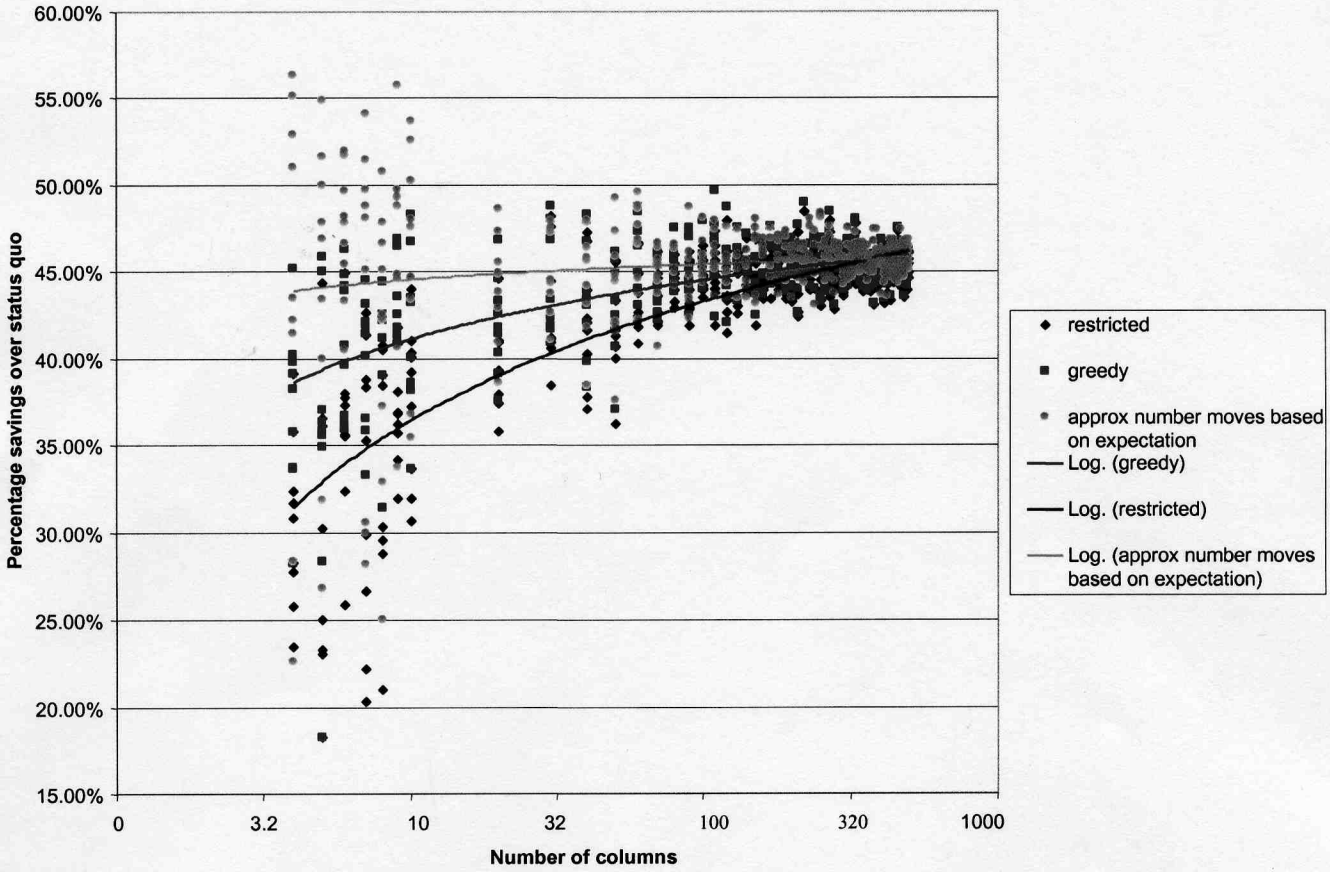
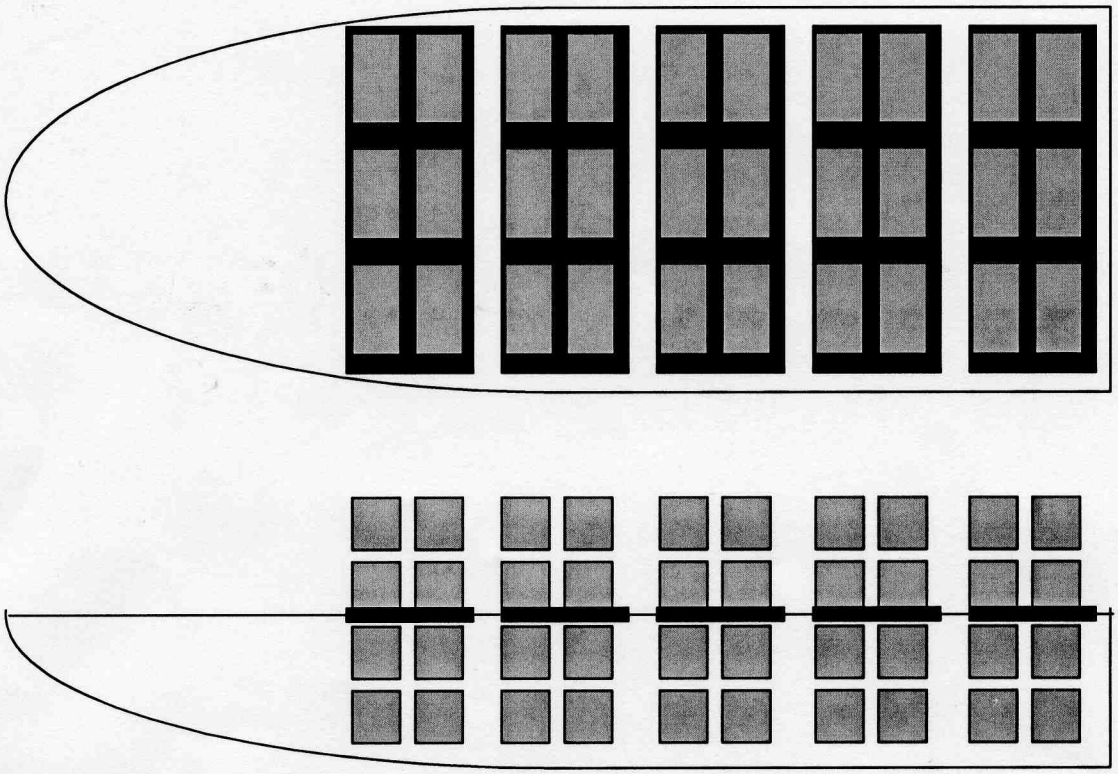


FIGURE 11 (a) Plan view and side view of a ship with hatch coverings **(b)** A simple hatched ship.

(a)



(b)