Truck Dock Assignment Problem with Time Windows and Capacity Constraint in Transshipment Network Through Crossdocks

Andrew Lim[1,2], Hong Ma[1], and Zhaowei Miao[1,2]

[1] Dept of Industrial Engineering and Logistics Management,
Hong Kong Univ of Science and Technology, Clear Water Bay, Kowloon, Hong Kong
[2] School of Computer Science & Engineering,
South China University of Technology, Guang Dong, P.R. China

Abstract. In this paper, we consider the over-constrained truck dock assignment problem with time windows and capacity constraint in transshipment network through crossdocks where the number of trucks exceeds the number of docks available, the capacity of the crossdock is limited, and where the objective is to minimize the total shipping distances. The problem is first formulated as an Integer Programming (IP) model, and then we propose a Tabu Search (TS) and a Genetic algorithms (GA) that utilize the IP constraints. Computational results are provided, showing that the heuristics perform better than the CPLEX Solver in both small-scale and large-scale test sets. Therefore, we conclude that the heuristic search approaches are efficient for the truck dock assignment problem.

Keywords: Heuristic search, crossdocks, dock assignment.

Areas: Heuristics, industrial applications of AI.

1 Introduction

Dock assignment for trucks is one of key activities at crossdocks. Trucks are assigned to docks for the duration of a time period during which the cargo and trucks are processed. Dock availability and times of arrivals/departures (as given by an estimated time of arrival/departure or ETA/ETD for each truck) can change during the course of the planning horizon due to operational contingencies (for example, delays, traffic control). A familiar scene at crossdocks these days is when arriving trucks are waiting for process, sometimes for a long time, before finally proceeding to their docks, because the gate is occupied by another truck. So they need to schedule those docks well in order to increase the utilization and achieve better performance of the transshipment network. Firstly, good dock assignment can help crossdock increase the utilization by reducing dock delays. Secondly, good dock assignment can minimize distances (times) cargo are required to transfer from dock to dock. Because of the large number of freight and the dynamic nature of the problem, scheduling has become more difficult. This has made it more important for crossdock operators to use docks in the best possible way.

We consider the truck dock assignment in transshipment network through crossdocks. But in classical models, where transshipment is studied in the context of network flow [1]. One such model where transshipment becomes an important factor is in crossdocking which has become synonymous with rapid consolidation and processing. Tsui and Chang used a bilinear program of assigning trailers to doors, where the objective was to minimize weighted distances between incoming and outgoing trailers [8]. Recently, a study by Bartholdi and Gue examined minimizing labor costs in freight terminals by properly assigning incoming and outgoing trailers to doors [2]. Although previous cross-docking studies have considered intra-terminal factors such as types of congestion that impact costs, they do not address actual dock assignments to arriving vehicles when considering the time window of trucks and capacity of crossdocks.

Also our problem is similar in some ways to the problem of gate assignments in airports, for which some analytical work exists. For example, the basic gate assignment problem is a quadratic assignment problem and shown to be NP-hard [7]. Since the gate assignment problem is NP-hard, various heuristic approaches have been used by researchers and work has focused on the over-constrained airport gate assignment, where there is an excess of flights over gates [3, 6]. The objective there was to minimize the number of flights without any gate assigned (i.e. those left on the ramp) and the total walking distance.

In this paper, we consider the over-constrained truck dock assignment problem with time windows and capacity constraint in transshipment network through crossdocks where the number of trucks exceeds the number of docks available and the capacity of the crossdock is limited, and where the objectives are to minimize the total shipping distances. The problem is formulated as an IP problem. The air gate assignment problem is **NP**-hard, then our problem is also **NP**-hard because the air gate assignment problem is a special case of our problem. We use both a Tabu Search and a GA algorithms to solve the problem. Computational results are provided , showing that our heuristics work well in all the test sets.

This work is organized as follows: in the next section, we give an IP model. Tabu search and GA algorithms are developed in Section 3 and section 4. We provide computational results in Section 5. In Section 6, we summarize our findings.

2 Problem Description and Formulation

In this section, we provide an IP model for the over-constrained truck dock assignment problem with time windows and capacity constraint which attempts to assign trucks within its time window to docks to minimize shipping distances between docks. We have capacity constraint that the total number of pallets inside the crossdock cannot exceed the capacity of crossdock. Also note that in real world, cargo containers are huge in size, and one pallet usually carries exactly one cargo container at one time. Therefore, terms 'pallet', 'cargo', and cargo container' refer to the same transportation unit in the transshipment network throughout the paper. The following notations are used:

N: set of trucks arriving at and/or departing from the crossdock;
M: set of docks available in the crossdock;
n: total number of trucks, that is $|N|$, where $|N|$ denotes the cardinality of N;
m: total number of docks, that is $|M|$;
a_i: arrival time of truck i $(1 \leq i \leq n)$;
d_i: departure time of truck i $(1 \leq i \leq n)$;
$w_{k,l}$: shipping distance for pallets from dock k to dock l $(1 \leq k, l \leq m)$;
$f_{i,j}$: number of pallets transferring from truck i to truck j $(1 \leq i, j \leq n;)$
C: capacity of crossdock, i.e. the maximum number of pallets the crossdock can hold at a time.

In addition, we use another dock, dock $m+1$, which is rent from others for temporary usage when there is not enough capacity left inside the crossdock, or when all the docks are occupied. New arriving truck should go to this dock to load and unload cargoes, but it will incur a much higher cost (distance is longer). This dock can be used by many trucks simultaneously with infinite capacity. We regard it as a reasonable remedy to make the problem always feasible. Let $w_{k,m+1}$ $(w_{m+1,k})$ be shipping distance from dock k $(m+1)$ to dock $m+1$ (k) $(1 \leq k \leq m)$. Figure 1 illustrates an outline of the crossdock and major elements of our problem.

The following auxiliary variables are used:

$x_{i,j} \in \{0,1\}$: 1 iff truck i departs no later than truck j arrives; 0 otherwise.

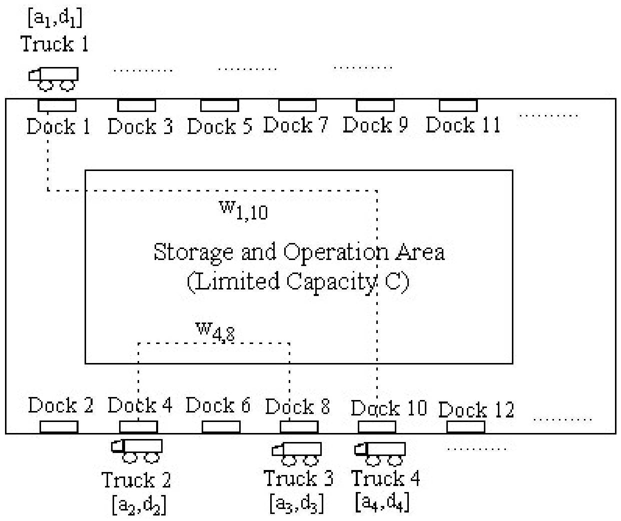

Fig. 1. Truck Dock Assignment Problem with Operation Time Constraint

The decision variables are as follows:

$y_{i,k} \in \{0, 1\}$: 1 if truck i is assigned to dock k, 0 otherwise. $(1 \leq i \leq n$, $1 \leq k \leq m+1)$. $y_{i,k} = y_{j,k} = 1$ implies that $a_i > d_j$ or $a_j > d_i$ $(1 \leq i, j \leq n)$;

$z_{ijkl} \in \{0, 1\}$: 1 iff truck i is assigned to dock k and truck j is assigned to dock l $(1 \leq i, j \leq n, 1 \leq k, l \leq m+1)$.

Before we give out the model, in order to make the problem and data reasonable, some remarks of are made as follows:

1) $f_{i,j} \geq 0$, iff $d_j \geq a_i$ $(1 \leq i, j \leq n)$, otherwise $f_{i,j} = 0$ which means truck i will transfer some cargo to truck j iff truck j departs no earlier than truck i arrives;
2) $a_i < d_i$ $(1 \leq i \leq n)$ which means for each truck, the arrival time should strictly smaller than its departure time;
3) $n > m$ which satisfies the over-constrained condition;
4) capacity C is defined as follows: when truck i comes, it consumes units of capacity equal to $\sum_{k=1}^{m} \sum_{l=1}^{m} \sum_{j=1}^{n} f_{i,j} y_{i,k} y_{j,l}$. On the other hand, when truck j leaves, $\sum_{k=1}^{m} \sum_{l=1}^{m} \sum_{i=1}^{n} f_{i,j} y_{i,k} y_{j,l}$ units of capacity are released.
5) sort all the a_i's and b_i's in an increasing order, and let t_r $(r = 1, 2..., 2n)$ correspond to these $2n$ numbers such that $t_1 \leq t_2 \leq ... \leq t_{2n}$. Use this notation, we can easily formulate the set of capacity constraints.

Our objective is to minimize the total shipping distance of transferring cargo between docks inside the crossdock. The IP model is as follows:

$$\min \sum_{k=1}^{m+1} \sum_{l=1}^{m+1} \sum_{i=1}^{n} \sum_{j=1}^{n} f_{i,j} w_{k,l} z_{ijkl}$$

s.t.

$$\sum_{k=1}^{m+1} y_{i,k} = 1 (1 \leq i \leq n) \tag{1}$$

$$z_{ijkl} \leq y_{i,k} (1 \leq i, j \leq n, 1 \leq k, l \leq m+1) \tag{2}$$

$$z_{ijkl} \leq y_{j,l} (1 \leq i, j \leq n, 1 \leq k, l \leq m+1) \tag{3}$$

$$y_{i,k} + y_{j,l} - 1 \leq z_{ijkl} (1 \leq i, j \leq n, 1 \leq k, l \leq m+1) \tag{4}$$

$$x_{i,j} + x_{j,i} \geq z_{ijkk} (1 \leq i, j \leq n, i \neq j, 1 \leq k \leq m) \tag{5}$$

$$\sum_{k=1}^{m} \sum_{l=1}^{m} \sum_{i \in \{i:\ a_i \leq t_r\}} \sum_{j=1}^{n} f_{i,j} z_{ijkl} - \sum_{k=1}^{m} \sum_{l=1}^{m} \sum_{i=1}^{n} \sum_{j \in \{j:\ d_j \leq t_r\}} f_{i,j} z_{ijkl} \leq C (1 \leq r \leq 2n) \tag{6}$$

$$y_{i,k} \in \{0,1\}, y_{j,l} \in \{0,1\}, z_{ijkl} \in \{0,1\}(1 \leq i,j \leq n, 1 \leq k,l \leq m+1) \tag{7}$$

Constraints (1) ensures that each truck must be assigned to exactly one dock. Constraints (2)-(4) jointly define the variable z which represent the logic relationship among $y_{i,k}$, $y_{j,l}$ and z_{ijkl}. Constraint (5) specifies that one dock cannot be occupied by two different trucks simultaneously. Finally, constraint (6) is capacity constraint which means that for each time point t_r, the total number of pallets inside the crossdock cannot exceed the capacity C.

3 Tabu Search

Tabu search (TS) is a heuristic search procedure that proceeds iteratively from one solution to another by moves in a neighborhood space with the assistance of adaptive memory [4]. We next describe TS memory and the framework used for the problem.

3.1 TS Memory

The solution is represented by a sequences A, which has length n (n is the number of trucks). The sequence A represents the dock assignment. For example, consider an instance with m docks and n trucks. The solution is then a sequence $(s_1, s_2, ..., s_n)$, which means that Truck 1 is assigned to dock (gate) s_1, Truck 2 is assigned to dock s_2, …, Truck n is assigned to dock s_n ($0 \leq s_i \leq m$, $1 \leq i \leq n$). If the truck i is unassigned to any of the docks, which is possible when all the docks are occupied, we give s_i the value of 0. If the solution is feasible, the dock assignment is then uniquely determined by the sequence of $(s_1, s_2, ..., s_n)$. The representation is depicted in Fig. 2. In the TS memory we implement, only the assignment information is captured so that only the move that has the identical neighborhood exchange move to the assignments will be forbidden.

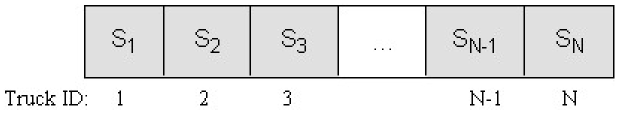

Fig. 2. The Solution Representation

3.2 Neighborhood Search

We use the a modified neighborhood search approach from Ding [3], which consists of three moves:

The Insert Move: Move a single truck to a dock gate other than the one it currently assigns. It is depicted in Fig. 3.

The Interval Exchange Move: Exchange two truck intervals in the current assignment. A truck interval is a group of consecutive trucks assigned to one dock gate. The move is depicted in Fig. 4.

The Rented Dock Exchange Move: Exchange a truck which is assigned to the rented dock with a truck that is assigned to a general dock gate.

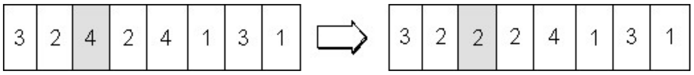

Fig. 3. The Insert Move

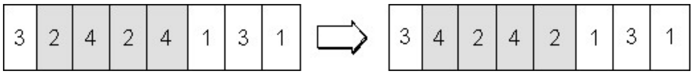

Fig. 4. The Interval Exchange Move

3.3 TS Framework

The TS algorithm can be described by the following steps:

1. Generate an initial solution x_{init} randomly, set $x_{curr} \leftarrow x_{init}$;
2. Generate a set of neighborhood solutions $N(x_{curr})$ of x_{curr} by the Insert Move and Interval Exchange Move;
3. The solution $x^{'} \in N(x_{curr})$ with the least cost and satisfying either one of the two conditions (1) and (2) and must satisfying condition (3) will be selected: (1) it is not forbidden (i.e. the assignment is not identical to any assignments of recent tabu tenure moves); (2) The cost of $x^{'}$ is better than the current best cost (aspiration criterion); (3) The occupied capacity of $x^{'}$ at boundary time points of all time windows is less than the capacity of crossdock C.
4. Set $x_{curr} \leftarrow x^{'}$; update the TS memory;
5. If the termination conditions are satisfied, stop; otherwise jump to step 2;

When we generate the neighborhood solutions, we randomly choose three types of moves with equal probability. There are two termination conditions: either the best solution cannot be improved within a certain number of iterations, or the maximum number of iterations has been reached.

4 Genetic Algorithm

Genetic algorithms (GA) have become a well-known meta-heuristic approach for difficult combinatorial optimization problems [5]. In this second approach to the dock assignment problem, we found it suitable to solve it. We first discuss some essential components of GA, including solution representation and crossover operators, and then outline the framework of our GA.

4.1 Solution Representation

The chromosome is an important component in GA and has great influence on the algorithm output. In the basic GA, a chromosome is usually encoded as a sequence and represents a solution. Similar to the TS solution representation (see Fig. 2), we here represent the dock assignment solution by a chromosome sequence that defines an assignment of trucks to the docks. During the population initialization process, if the randomly generated chromosome is infeasible, we just drop the infeasible sequence and generate a new one.

4.2 Crossover and Mutation

In our problem, chromosomes are not permutation sequences such as in the Travelling Salesman Problem. Hence,well-known crossover operators, such as Partially Mapped Crossover and Cycle Crossover, cannot be used. We implemented two crossover operators: One-Point Crossover and Two-Point Crossover.

In the One-Point Crossover, one random crossover point is selected. The first part of the first parent is attached with the second part of the second parent to make the first offspring. The second offspring is built from the first part of the second parent and the second part of the first parent. The following is an example of One-Point Crossover operator (the crossover point is denoted by |):

Parent1: (3 2 4 2 4 | 1 3 1), Parent2: (2 1 2 4 3 | 2 2 4)
Offspring1: (3 2 4 2 4 | 2 2 4), Offspring2: (2 1 2 4 3 | 1 3 1)

In the Two-Point Crossover, two random crossover points are selected for one crossover operation. The chromosomal materials are swapped between two cut points to produce offsprings. This is illustrated in the following example:

Parent1: (3 2 | 4 2 4 | 1 3 1), Parent2: (2 1 | 2 4 3 | 2 2 4)
Offspring1: (3 2 | 2 4 3 | 1 3 1), Offspring2: (2 1 | 4 2 4 | 2 2 4)

In the proposed GA, we use both of the above-mentioned cross over operators with equal probability. We chose the 'Swap Mutation' as our mutation operator, which selects two positions at random and swaps the values at those positions. For example, the following mutation swaps the values at position 3 and position 6: (3 2 4 2 4 1 3 1) $\rightarrow$ (3 2 1 2 4 4 3 1). For simplicity, we do not apply any repair function to the infeasible offsprings. Therefore, if an offspring is infeasible by violating constraints in the problem, we simply remove it from the GA population base.

4.3 GA Framework

With these components of GA, we now outline GA as follows . In this algorithm, #pop, #crossover, #iter and p_1 are parameters which are specified within experiments.

```
**Initialize** Pop with size $\sharp pop$
**for** $iter \leftarrow 1$ to $\sharp iter$ **do**
  **for** $off \leftarrow 1$ to $\sharp crossover$ **do**
    Randomly select ParentA and ParentB
    Crossover ParentA and ParentB to produce OffspringA and OffspringB
  **end for**
  **for** each new-produced individual indv **do**
    mutate indv with probability $p_1$
  **end for**
```

```
    evaluate each individual
    select the best $\sharp pop$ individuals from all the individuals
    update current best solution
  **end for**
```

5 Experimental Results and Analysis

All the algorithms were coded in Java and tested on a P4 2.4GHz PC with 512M RAM. As comparison, we use ILOG CPLEX 8.0 to the IP formulation presented in Section 2. The test generation process, parameter settings of various methods, and detailed computational results are presented in the following subsections.

5.1 Test Data and Experiment Setup

We chose a representative layout of a crossdock to have two parallel sets of terminals, where docks are symmetrically located in the two terminals shown in Fig. 5. We set the distance between two adjacent docks within one terminal (e.g., dock 1 and dock 3) to be 1 unit and the distance between two parallel docks in different terminals (e.g., dock 1 and dock 2) to be 3 units. To simplify the problem, we assumed that forklift can only walk 'horizontally' or 'vertically', i.e., if one forklift wants to transfer one pallet from dock 3 to dock 2, his walking distance is 1+3=4 units. This is similar to the so-called Manhattan metric. In addition, we set the distance between rented dock and any of docks inside the crossdock to be 10 in order to make it undesirable to use.

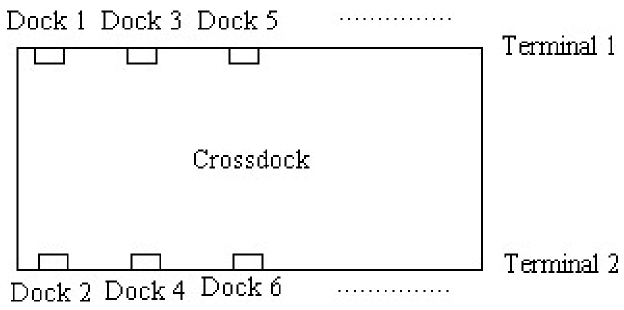

Fig. 5. Crossdock topology

The start points of truck time window $a_i (1 \leq i \leq n)$ are uniformly generated in the interval $[1, \frac{n*70}{m}]$. The end points b_i are generated as $b_i = a_i + [45, 74]$. The number of transferring pallets $f_{i,j}$ is randomly generated in the interval [6, 60] if $d_j \geq a_i$ (0 otherwise). In TS, Maximum number of iterations is 10^6 and each time 100 neighbors are generated with a tabu tenure = 10. The algorithm is to terminate if the best solution was not improved within 10^4 iterations. In GA, we specify $\sharp iter = 10^4$, $\sharp pop = 300$, and $\sharp crossover = 500$. The mutation probability p_1 is taken to be 0.2. The maximum iteration is 10^5 and the termination condition was when the best solution did not improve within 500 iterations.

5.2 The Results

We designed two categories of test sets. The detailed results are presented in Table 1–2 respectively. The first row of each table denotes the instance ID or group ID. The second row contains the sizes, where $n \times m$ means that there are n trucks and m docks. The rest of the rows provide the results of various methods proposed in this paper. Each result cell contains two values. The value on the top provides the result, whereas the value at the bottom provides the computational time in seconds.

1. Small-scale Test Sets
Small-scale test sets are generated with the size ($n \times m$) ranging from 12×4 to 18×7. We see that GA performs extremely well for this group of test sets, as it obtains the best objective values for all test sets. Solution quality of TS are slightly worse than GA while the runtime of TS is considerably faster. At last, CPLEX gets the optimal solutions only for the three smallest test sets within the time limit of 2 hours. For all other 7 test sets that CPLEX exceeds the time limit, the solution quality is generally worse than the two meta-heuristic approaches.

2. Large-scale Test Sets
Large-scale test sets ranging from 20×6 to 35×9 are used here. We see from Table 2. that the performance of TS surpasses GA in both solution quality and runtime, but in general, both meta-heuristics have provided quite good solutions. beating GA in 8 of the 10 test sets. We also note the solution time of GA grows very quickly as the problem size expands, which indicates that GA does not scale quite well for the problem.

As a whole, we can conclude from the experiment that both two proposed meta-heuristics are effective approaches for the dock assignment problem. If a minimal runtime of solving medium-to-large-scale instances is required, TS is then more preferred than GA.

Table 1. Results: Small-scale Test Sets

Size	12 X 4	12 X 5	14 X 4	14 X 5	16 X 6	16 X 7	18 X 6	18 X 7
CPlex	$\underline{7523}$	$\underline{8914}$	$\underline{12117}$	3680	22984	13049	14712	14276
Time(s)	1035	5460	3166	≥ 7200	≥ 7200	≥ 7200	≥ 7200	≥ 7200
TS	$\underline{7523}$	8917	12200	3578	21017	12528	13760	13818
Time(s)	0.55	0.58	0.56	0.58	0.66	0.64	0.72	0.78
GA	$\underline{7523}$	$\underline{8914}$	$\underline{12117}$	$\underline{3464}$	$\underline{20953}$	$\underline{12489}$	$\underline{13635}$	$\underline{12986}$
Time(s)	1.89	1.75	2.53	2.61	2.91	4.13	4.81	5.58

Table 2. Results: Large-scale Test Sets

Size	20 X 6	20 X 7	25 X 6	25 X 7	30 X 8	30 X 9	35 X 8	35 X 9
CPlex	62537	64682	101683	112648	146595	134762	148654	145725
Time(s)	≥ 7200	≥ 7200	≥ 7200	≥ 7200	≥ 7200	≥ 7200	≥ 7200	≥ 7200
TS	$\underline{40777}$	$\underline{49219}$	$\underline{68610}$	$\underline{76418}$	99442	$\underline{97981}$	$\underline{105312}$	116147
Time(s)	1.06	1.02	1.31	1.28	1.84	1.81	2.28	2.34
GA	42405	60610	68892	78112	$\underline{95918}$	98355	105768	$\underline{111341}$
Time(s)	5.36	1.95	9.95	7.20	17.83	15.80	51.11	59.00

6 Conclusions and Future Work

In this paper, we consider the over-constrained truck dock assignment problem with time windows and capacity constraint in transshipment network through crossdocks where the number of trucks exceed the number of docks available and the capacity of the crossdock is limited, and where the objectives are to minimize the total shipping distances. The problem is formulated as an IP model. We then propose a Tabu Search (TS) and a Genetic algorithms (GA) that utilize the IP constraints. Experiments were conducted using a range of test data sets that reflect realistic scenarios. The heuristic search algorithms are compared with the CPLEX solver, showing they obtain better results within shorter running times. In the future, we think that although this problem considers crossdock optimization problem, it can also be applied to the air gate assignment problem in airport to schedule the flights well in order to achieve better performance. The other direction of research is to study a realistic model which take the operational time, instead of the distance, of each pallet into consideration. Therefore, the arrival/departure time windows of the trucks can be well related to the inner-crossdock operations.

References

[1] J.E. Aronson. A survey on dynamic network flows. *Annals of Operations Research*, 20:1–66, 1989.

[2] J. Bartholdi and K. Gue. Reducing labor costs in an ltl cross-docking terminal. *Operations Research*, 48:823–832, 2000.

[3] H. Ding, A. Lim, B. Rodrigues, and Y. Zhu. The over-constrained airport gate assignment problem. *Computers and Operational Research*, 32:1867–1880, 2005.

[4] F. Glover and M. Laguna. *Tabu Search.* Kluwer Academic Publishers, Dordrecht, The Netherlands, 1998.

[5] John H. Holland. *Adaptation in Natural and Artifical Systems.* University of Michigan Press, Ann Arbor, MI, 1975.

[6] A. Lim, B. Rodrigues, and Y. Zhu. Airport gate scheduling with time windows. *Artificial Intelligence Review*, 24:5–31, 2005.

[7] T. Obata. The quadratic assignment problem: Evaluation of exact and heuristic algorithms. Technical report, 2000.

[8] L. Tsui and C. Chang. Optimal solution to dock door assignment problem. *Computers and Indutrial Engineering*, 23:283–286, 1992.