한국항만학회 '98 추계학술대회논문집, pp. 315~322

論 文
B3-2

A Neighborhood Beam Search Algorithm for Routing Yard-Side Equipment in Port Container Terminals

K. Y. Kim* · K. H. Kim**

컨테이너 터미널에서 야드장비의 경로결정을 위한 이웃에 대한 빔 탐색 방식

김기영 · 김갑환

key words : Container terminal (컨테이너 터미널), Loading (적하), Heuristic (발견적 기법), Beam search (빔 탐색)

Abstract

It is discussed how to route yard-side equipment during the loading operation in port container terminals. The number of containers to be picked up at each yard-bay, as well as the route of a yard-side equipment (for example, transfer crane or straddle carrier) in a yard, are determined. The objective of the problem is to minimize the total container handling time in the yard. An encoding method to represent nodes in the search space is introduced utilizing inherent properties of the optimal solution by which the search space is greatly reduced. A beam search algorithm is suggested. A numerical experiment is carried out to compare the performance of the beam search algorithm with those of other approaches.

1. Introduction

The efficiency of the loading operation is highly dependent on the travel time of the yard-side equipment (YSE) such as a transfer crane or a straddle carrier. The travel time of the YSE depends significantly on both the visiting sequence of the YSE and the number of containers to be picked up at each visiting yard-bay (which we call pick-up schedule). Thus, in this paper, the problem to determine the pick-up schedule is addressed (which we call the

---

* Institute of Asia-Pacific Studies, Waseda University, Japan

** CIIPMS, Department of Industrial Engineering, Pusan National University,

YSE routing problem). Based on a special solution property derived from the formulation, a neighborhood bean search algorithm is developed. An attempt is made to minimize the container handling time of a YSE by optimally determining the yard-bay sequence that the YSE visits and the number of containers for the YSE to pick up at each visiting yard-bay, simultaneously.

Constraints to be satisfied are (a) the pick-up schedule must satisfy the requirements of the work schedule of the corresponding quay crane, and (b) the total number of containers of each group picked up at each yard-bay must be equal to the initial number of containers of the corresponding group stored at the yard-bay. Note that in Tables 1 and 2, the total number of containers of a specific group in the work schedule is equal to the total initial number of containers of the group in a yard.

Table 1. An example of a work schedule

B(h) = the set of the yard-bay numbers where containers of group *h* exist

c_{hj} = the initial number of containers of group *h* stacked at yard-bay *j*

r_t = the number of containers to pick up during partial-tour *t*

g_t = container group number of containers that must be picked up during partial-tour *t*

Table 2. The initial number of containers in each yard-bay in the example

yard-bay no. / container group	3	5	7	8	12	14	18	22	24
A			3			5			4
B	3				2				
C		2					5		
D				3				3	

Table 3. Travel distances between yard-bays

(unit : m)

to / from	3	5	7	8	12	14	18	22	24
3		2	4	5	9	11	15	19	21
5	2		2	3	7	9	13	17	19
7	4	2		1	5	7	11	15	17
8	5	3	1		4	6	10	14	16
12	9	7	5	4		2	6	10	12
14	11	9	7	6	2		4	8	10
18	15	13	11	10	6	4		4	6
22	19	17	15	14	10	8	4		2
24	21	19	17	16	12	10	6	2	

A "partial-tour" of a YSE means a visiting sequence of yard-bays during a YSE picks up all the containers onto a cluster of cells in the containership. In Table 1, the first partial-tour of the YSE corresponds to a visiting sequence of yard-bays for the YSE to pick up 5 containers of group A. A complete tour for the YSE can be obtained by connecting partial-tours.

The following notations are used to formulate the YSE routing problem:

x_j^t = the number of containers to be picked up at yard-bay j during partial tour t that is a decision variable

S(h) = the set of partial-tour numbers whose corresponding container group number is *h*

In the following, a useful property will be suggested in order to obtain the satisfying solution efficiently. The set of constraints (a) and (b) in the previous page may be expressed as follows::

(Constraint-subset k)

$$\sum_{j\in B(k)} x_j^t = r_t \quad, t \in S(k), \qquad \text{.......................................(1)}$$

$$\sum_{t\in S(k)} x_j^t = c_{kj} \quad, j \in B(k) \qquad \text{.......................................(2)}$$

Constraint (1) and (2) are of the same form as the transportation problem for which the basic feasible solution may be easily obtained. Let $|S(k)| = m_k$ and $|B(k)| = n_k$. Then, we can obtain a solution to constraint-subset k by setting $m_k n_k - (m_k + n_k - 1)$ variables to be zero among $m_k n_k$ variables. We call the solution "a basic solution to constraint-subset k." It is known that considering only the basic feasible solutions to constraint (1) and (2) is sufficient to obtain the optimal solution (Kim, 1998).

Once x_j^t for all j and t are given, the remaining problem is to determine the route of a YSE in a way of minimizing the total travel distance of the yard equipment.

2. Encoding Solutions

A method to represent the nodes in the search space is described in this section. Based on the property that one of basic feasible solution to constraint-subset k can be an optimal solution, only those basic feasible solutions will be considered as candidates for the number of containers to pick up at each yard-bay. Thus, it is necessary to devise an efficient node representation method that maps only basic feasible solutions in the search space onto the set of symbolic strings.

The symbolic string consists of several sections, each of which corresponds to a basic feasible solution to constraint subset k. A section is divided again into three sectors, each of which we call yard-bay sector, partial-tour sector, and sequencing sector, respectively. Yard-bay sector and sequencing sector consist of the yard-bay identification numbers which have containers of the corresponding group, while partial-tour sector is composed of the partial-tour numbers for loading containers of the corresponding group. The strings in yard-bay sector and partial-tour sector determine basic feasible solution to the corresponding constraint subset, while sequencing sector tells us the travel sequence during each of partial-tours.

In the example of Fig. 1, there are 4 sections in the string, each of which corresponds to a specific container group. In section A, the first three slots constitute yard-bay sector and the next two slots constitute partial-tour sector and the last three slots constitute sequencing sector. Once a string is given, the visiting sequence of yard-bays and the number of containers to pick up at each yard-bay can be uniquely determined as follows:

(step 1) Suppose the section for container group A is given as shown in Fig. 1. Then, a basic feasible solution to constraint-subset *k* can be found, which are related to container group A, by a method to get an initial basic feasible solution in the transportation problem. In this paper, the number of containers to be picked up at each yard-bay is determined by the North-western corner rule. Fig. 2 illustrates how to get basic feasible solutions from the string shown in Fig. 1.

(Step 2) The basic feasible solution obtained in (step 1) tells us how many containers should be picked up at each yard-bay during a specific partial-tour. Then, following the relative sequence of yard-bays in sequencing sector, the visiting sequence of yard-bays during a specific partial-tour can be determined. In the example of

Fig.s 1 and 2, for partial-tour 1, the visiting sequence becomes (1) yard-bay 24 (2) yard-bay 14 because the substring in the sequencing sector for group A indicates that the higher priority is given to the yard-bay 24 than 14. Finally, the pick-up schedule is obtained as in Table 4.

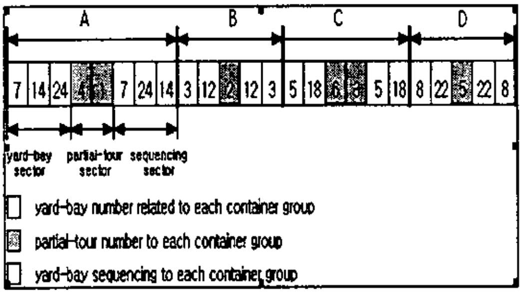

Fig. 1. An illustration of encoding which leads to a basic feasible solution

demand / supply	PT 4	PT 1	sum
YB 7	3		3
YB 14	4	1	5
YB 24		4	4
sum	7	5	12

(a) a BFS for container group A

demand / supply	PT 2	sum
YB 3	3	3
YB 12	2	2
sum	5	5

(b) a BFS for container grou

demand / supply	PT 3	PT 6	sum
YB 5	2		2
YB 18	1	4	5
sum	3	4	7

(c) a BFS for container group C

demand / supply	PT 5	sum
YB 8	3	3
YB 22	3	3
sum	6	6

(d) a BFS for container grou

PT: partial tour
YB: yard-bay
BFS: basic fesible solutio

Fig. 2. Basic feasible solutions decoded from a chromosome

Table 4. The final pick-up schedule for the example

Sequence (partial-tour number)	1		2		3		4		5		6
visiting yard-bay	24	14	12	3	5	18	7	14	22	8	18
number of container	4	1	2	3	2	1	3	4	3	3	4

3. A Neighborhood Beam Search Algorithm

Fig. 4 gives an example of the search tree. In the beam search, we search for a subset of all possible nodes starting from a root node in which the most three promising nodes are selected at each level. Consequently, the number of nodes remains manageable, even if there is a great deal of branches at each level and the search level is deep. Whenever the beam search is used, there are only w (beamwidth) nodes under consideration at any level, rather than the exponentially explosive number of nodes.

We can begin with a feasible and complete solution and search for neighborhoods to improve the solution, which we call the neighborhood beam search algorithm. For the convenience of the explanation, the work schedule shown in Table 1 and the yard-map that consists of Tables 2 and 3 continue to be used.

The following terminology are used to describe the neighborhood beam search algorithm for routing a single yard-side equipment:

Node	: a string that specifies a basic feasible solution and a sequence of yard-bays that a YSE visits.
Root node	: the node of level zero which is a complete string
Stage	: the depth of the search from level zero
Edge	: an arc connecting a node to an immediate lower-stage node
Beam node	: a node from which branches to lower-level nodes are created
Beamwidth	: the maximum number of beam nodes allowed at each stage

Branch factor	: the maximum number of nodes which may be created from a beam node

In the neighborhood beam search algorithm, newly connected nodes from the root node or a beam node are generated systematically by exchanging two characters each other within each sector in the string. By exchanging every combination of two characters in the same sector as in the swap process, all neighbor nodes of a beam node can be obtained as follows: First, each container group (one of four container groups in root node in Fig. 5) in a sting is sequentially selected. Within the selected container group, each sector (one of twelve sectors) is selected. Then, every combination of two elements in the selected sector are exchanged each other. Note this method is similar to the swap mutation operator, while the only difference is that the neighbor node is generated systematically. All beam nodes in a level are explored, all newly connected nodes from all beam nodes are evaluated and a specified number of the most promising nodes are selected. The beam nodes are selected from the whole set of nodes in the level based on the value of the total travel distance.

Figures 4 and 5 illustrate the neighborhood beam search algorithm. At level 1, fourteen newly connected nodes are generated from a root node by exchange two characters each other within each sector of the root string. Next, the most three promising nodes are selected at level 1, that is, node 6, node 7 and node 9. These nodes become new beam nodes at level 1. At level 2, newly connected nodes from beam nodes at level 1 are obtained and are evaluated again. At level 2, node 35, node 41 and node 42 become new beam nodes.

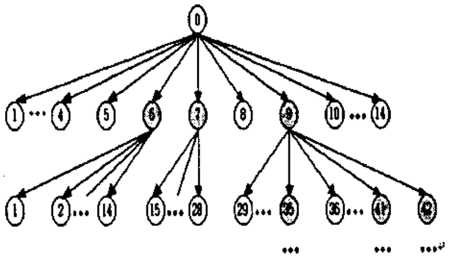

Fig. 4. An illustration of a search tree

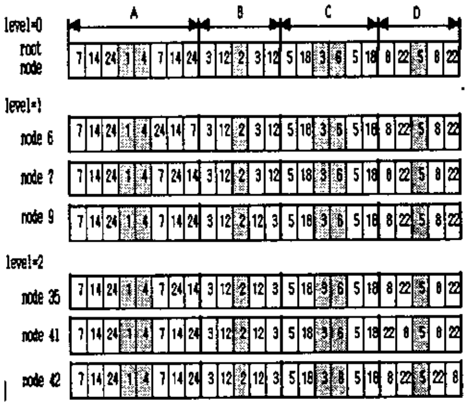

Fig. 5. The representation of nodes in the search space

4. Performance Comparisons

4.1 The Test Data and the Experimental Design

The experiment objective was to evaluate the relative performances among the suggested heuristic search algorithms. Ten small-sized and ten large-sized problems were solved.

Small-Sized Problems: Containers were scattered over ten yard-bays on a line and the travel distance was used as performance measure. The locations of ten yard-bays were generated randomly over a block that consists of twenty-five yard-bays. Forty-eight containers of four container groups were distributed over ten yard-bays

Table 5. The results of the comparison among algorithms for the yard-side equipment routing problem

	Measure	Mixed integer programming	Enumeration Based on DP	GA P=400, G=100 c=0.9,m=0.05	Neighborhood beam search (w=7)
Small-Sized Problem (10 yard-bays)	Travel Distance	100%	100%	102.1%	101.2%
	Run time (seconds)	16 hours for a problem with 3 yard-bays	5	53.2	2
Large-Sized Problem (30 yard-bays)	Travel Distance	x	x	119 % of the result by the beam search	100%
	Run time (seconds)	x	More than 3.35x1036 nodes in the search spaces	63	70.4

(P = population size, G = the number of generation, w = beamwidth x = no problem is solved)

randomly. Work schedule was also generated by allocating a randomly-chosen number of containers to a randomly selected partial-tour repeatedly until the allocated number of containers consumes the total number of containers of the corresponding group specified in yard-bays. Setup time for a move between yard-bays was assumed to be zero and the travel time per one-yard-bay-distance was assumed to be 1 time unit.

Large-Sized Problems: The number of yard-bays was thirty and the rectilinear distance was used as the distance measure. The locations of thirty yard-bays were generated randomly over a plane of size 30x40 and 243 containers of six container groups were distributed over thirty yard-bays randomly. The other conditions were the same as the small-sized problems.

In the neighborhood beam search, the branch factor was set to a sufficient large number though pilot runs so that all newly connected nodes for the test problem could be sprouted. The depth of search (the maximum number of level) was also set to a sufficient large number so that the search could progress to such a level that no further neighborhood might be obtained. As the number of levels increased, it was expected that the objective function value converged to a near optimal value. The beamwidth was set to 7 through pilot runs for which run time of the beam search became close to that of the genetic algorithm.

4.2 The Results of the Experiments

The pilot run was performed for the large-sized problem using the neighborhood beam search algorithm. Fig. 6 shows the quality of the solution improves as the level of the beam search algorithm increases. The result of the beam search algorithm was not sensitive to the beamwidth.

Run time of the beam search algorithm becomes equal to the one of the GA when the beamwidth becomes seven. The travel time of the neighborhood beam search algorithm with the beamwidth of seven (584) is lower than the results by the GA with the same computational time (622-712).

Table 5 summarizes the results of the comparison among algorithms for the yard-side equipment routing problem. In the neighborhood beam search algorithm, the branch factor and the depth of search were set to be a sufficient large number. The beam-width was set to be seven (w=7).

For small-sized problems, all solution techniques were compared with each other. For the mixed integer programming formulation, a problem with only three yard-bays was exceptionally solved. It took 16 hours. Nevertheless, it is interesting that the enumeration based on DP solves some small-sized problems optimally within a few seconds. Thus, for small-sized problems, the result of enumeration based on DP was the reference point for the comparison. The comparison results showed there is no significant travel distance difference between the genetic algorithm and the neighborhood beam search. However, it was noticeable that the neighborhood beam search outperformed the genetic algorithm a little in both the quality of the solution and run time.

Although both mixed integer programming and enumeration based on DP guarantee the optimal solution, it was almost impossible to get the optimal solution for large-sized problems because of the excessive computational time. Thus, the result of the neighborhood beam search algorithm was used as the reference point and which is the reason why the test was omitted. The comparison results for large-sized problems in Table 5 showed that the neighborhood beam search worked better than the genetic algorithm for the yard-side equipment routing problem.

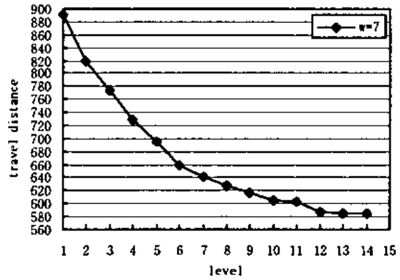

Fig. 6. Travel distance obtained by pilot run of the beam search algorithm

5. Summary and Conclusions

A beam search algorithm with a reduced solution set was introduced to determine the load sequence of export containers in a port container terminal. An attempt was made to determine the number of containers to be picked up at each yard-bay as well as the travel route of the yard-side equipment. The objective is to minimize the total container handling time. A representation method of nodes in the search space was developed which utilizes inherent properties of the optimal solution. A numerical experiment was carried out to compare the performances of the heuristic algorithms. By the experimental test, it was shown that although there is no significant difference between the genetic algorithm and the neighborhood beam search, it is noticeable that the neighborhood beam search outperforms the genetic algorithm a little.

In this paper, the case of single quay crane and yard-side equipment (straddle carrier or transfer crane) is considered. But, multiple cranes are commonly used for the loading operation of modern large-sized containerships. Thus, an algorithm is needed to solve the load scheduling for multiple cranes, which is a promising topic for the further researches.

요 약 문

본 연구는 컨테이너 터미널에서 선적 작업 시 야드 장비의 운행 경로 결정을 어떻게 할 것인지를 다루고 있다. 각 야드 베이에서 선적해야 할 컨테이너 수와 야드 장비의 경로를 결정하는 문제이다. 문제의 목적 함수는 야드에서 총 컨테이너 취급 시간을 최소화하는 것이다. 탐색 공간에서 노드를 표현할 암호화 방법에 최적해의 속성을 이용하여 탐색공간을 크게 감소시켰다. 빔 탐색 방식을 제안하였다. 다른 연구들과 빔 탐색 방식의 성능을 비교하기 위하여 실험을 하였다.

References

1) Kim, K. Y. (1998) "Mathematical Models and Search Methods for Containership Load Planning in Port Container Terminals", Ph.D. Dissertation, Pusan National University.

2) Cho, D. W. (1982) "Development of a Methodology for Containership Load Planning", Ph.D. Dissertation, Oregon State University.

3) Gifford, L. A. (1981) "Containership Load Planning Heuristic0 for a Transtainer based Container Port". M.Sc. Thesis, Oregon State University.

4) Change, Y. L., Matsuo, H., and Sullivan, R. S. (1989) "A Bottleneck-based Beam Search for Job Scheduling in a Flexible Manufacturing System", International Journal of Production Research. Vol.27, No.11, pp.1949-1961.

5) Ow, P. S. and Morton, T. E. (1988) "Filtered Beam Search in Scheduling", International Journal of Production Research, Vol.26, No.1, pp.35-62.