Computers and Industrial Engineering Vol. 23, Nos 1–4, pp. 283–286, 1992
Printed in Great Britain.

0360-8352/92 $5.00+0.00

AN OPTIMAL SOLUTION TO A DOCK DOOR ASSIGNMENT PROBLEM

Louis Y. Tsui and Chia-Hao Chang

Department of Industrial and Systems Engineering
University of Michigan-Dearborn
Dearborn, Michigan 48128

ABSTRACT

This paper addresses issues at the shipping/receiving dock of a shipping company, where trucks arrive from vendors to have their shipments unloaded, sorted, and reloaded onto trucks going to the customers. The assignment of dock doors to incoming and outgoing trucks determines the efficiency of the dock operation. A bilinear programming model is proposed to solve the assignment problem.

INTRODUCTION

A large shipping company has several freight yards, located at different cities throughout the country. Inbound trucks arrive at the yard with shipments from venders and/or other freight yards, the shipments are sorted according to their destination, and then loaded onto outbound trucks that make delivery to the customers. The shipments are processed in the shipping/receiving dock, which is rectangular in shape. For management purposes, one side of the dock is designated to the inbound trucks and the opposite side is designated to the outbound trucks. When a full truck comes in, it gets assigned an available receiving door. The truck driver drops the trailer and leaves the yard with another trailer for his next assignment. Each trailer is assigned to a forklift driver, who is responsible for unloading the entire contents and loading them on different empty trailers according to their destination. The empty trailers are parked at the shipping doors, with each door assigned to a particular destination. Once the trailer is full, it is hauled away and replaced by another empty trailer going to the same destination. Therefore, depending on the amount of travel required between the receiving door and the shipping door, the time to process each trailer can vary significantly. No storage is provided at the freight yard, all shipments must be processed the same day they arrive, and they go directly from the inbound trailer to the outbound trailer to minimize the possibility of being damaged or lost. The longer it takes to empty the trailer, the more forklifts are required and the more congested the dock will be. The company has records on the size, weight, origin and destination for each item shipped. Therefore, information can be derived on total weight and volume of items from each origin to each destination. The industrial engineers then use the information to assign dock doors so that the dock can operate smoothly and efficiently. The shipping pattern changes from time to time, which occasionally warrants an adjustment of the door assignment. Because of the information involved and the combinatorial nature of the problem, it is not a trivial task to perform manually. Tsui and Chang [Tsui, et. al. 90] proposed a bilinear programming model and a simple method for a local optimal solution. However, their result depends heavily on the initial solution. This paper proposes an approach for an optimal solution that can be applied directly by the industrial engineers.

THE MODEL

Let there be M receiving doors and N shipping doors at the dock. Let there be I origins and J destinations for the items. Without loss of generality, we can assume that $I \leq M$ and $J \leq N$. If there are more origins than receiving doors, the origins can be redefined to include a larger area so that the above assumption holds. Let $x_{im} = 1$ if origin i is assigned to receiving door m, $x_{im} = 0$

otherwise. Let $y_{jn} = 1$ if destination j is assigned to shipping door n, $y_{jn} = 0$ otherwise. Let d_{mn} be the distance between receiving door m and shipping door n, and let the number of forklift trips required to move the items originated from i to destination j be denoted as w_{ij}. The objective is to find an assignment of receiving doors to the origins and shipping doors to the destinations, so that the total distance traveled by the forklifts is minimized. The problem can be formulated as below.

$$\text{Minimize } f(X,Y) = \sum_i \sum_j \sum_m \sum_n w_{ij} d_{mn} x_{im} y_{jn}$$

subject to

$$\sum_m x_{im} = 1 \text{ for } i = 1,2...I \quad (1)$$

$$\sum_i x_{im} = 1 \text{ for } m = 1,2...M \quad (2)$$

(P)

$$\sum_n y_{jn} = 1 \text{ for } j = 1,2...J \quad (3)$$

$$\sum_j y_{jn} = 1 \text{ for } n = 1,2...N \quad (4)$$

$$x_{im} = 0 \text{ or } 1 \text{ for all } i, m$$

$$y_{jn} = 0 \text{ or } 1 \text{ for all } j, n$$

Since there are w_{ij} forklift trips from origin i to destination j, if origin i is assigned to door m and destination j is assigned to door n, the total distance traveled will be $w_{ij}d_{mn}$, hence, the objective function sums up the total distance traveled from all receiving doors to all shipping doors according to the door assignment X and Y. Constraint set (1) guarantees that each origin is assigned only one receiving door, constraint set (2) guarantees that each receiving door is assigned to only one origin. Constraint set (3) guarantees that each destination gets assigned a shipping door, and constraint set (4) restricts such that each shipping door is assigned to only one destination.

Although different methods are available for solving bilinear problems [Thieu, 88], [Sherali and Shetty, 80], [Vaish and Shetty, 77], [Gallo and Ulkucu, 77], their constraints are in a form of $X \varepsilon D$, $Y \varepsilon E$, and X and Y are real. Whereas in this case, X and Y are 0-1 integer variables and both D and E have the property of an assignment problem instead of a more general form. It is much easier to solve the assignment problem than to solve a general linear programming problem; thus, it is preferred not to add any additional cutting plane constraint to problem (P). The branch and bound method proposed by Al-Khayyal et. al. [Al-Khayyal and Falk, 83] cannot be directly applied because the variables in this problem are 0-1 integer variables. We propose the following.

It is easily seen that problem (P) is equivalent to the following :

$$\text{Minimize } g(X)$$

subject to

$$\sum_m x_{im} = 1 \text{ for } i = 1,2...I \quad (1)$$

(GP)

$$\sum_i x_{im} = 1 \text{ for } m = 1,2...M \quad (2)$$

$$x_{im} = 0 \text{ or } 1 \text{ for all } i, m$$

$$\text{where } g(X) = \text{Min} \sum_j \sum_n c(X,j,n)\, y_{jn}$$

subject to

$$\sum_n y_{jn} = 1 \text{ for } j = 1,2...J \quad (3)$$

$$\sum_j y_{jn} = 1 \text{ for } n = 1,2...N \quad (4)$$

$$y_{jn} = 0 \text{ or } 1 \text{ for all } n, j$$

$$\text{where } c(X,j,n) = \sum_i \sum_m w_{ij} d_{mn} x_{im}.$$

$$\text{Let } v = \text{Min} \sum_j \sum_n c(j,n)\, y_{jn}$$

subject to

$$\sum_n y_{jn} = 1 \text{ for } j = 1,2...J \quad (3)$$

(CP)

$$\sum_j y_{jn} = 1 \text{ for } n = 1,2...N \quad (4)$$

$$y_{jn} = 0 \text{ or } 1 \text{ for all } n, j$$

where $c(j,n) = \text{Min} \sum_i \sum_m w_{ij} d_{mn} x_{im}$

subject to

$$\sum_m x_{im} = 1 \text{ for } i = 1,2...I \quad (1)$$

$$\sum_i x_{im} = 1 \text{ for } m = 1,2...M \quad (2)$$

$$x_{im} = 0 \text{ or } 1 \text{ for all } i, m$$

Since $c(j,n) \leq c(X,j,n)$ for all X, j, and n, we have $v \leq g(X)$ for any X. Therefore, v is a lower bound for problem P.

The Algorithm

We are now ready to state the branch and bound algorithm.

1. Calculate c(j,n) for each j,n pair. Solve problem (CP) for v and Y. Use the resulting Y as the initial starting point to find a local optimal solution as the initial incumbent according to Tsui's method [Tsui et. al. 90]. If the lower bound v equals the incumbent, we have found the optimal solution, stop. Otherwise, go to step 2.

2. Calculate $\sum_j w_{ij}$ for all i and sort the results in descending order so that $\sum_j w_{b1,j} \geq \sum_j w_{b2,j} \geq \sum_j w_{b3,j} \cdots\cdots \geq \sum_j w_{bl,j}$. Treat the original problem as a candidate problem, go to step 3.

3. Select a candidate problem with the most number of imposing additional constraints in the form of $x_{ij}=1$. If the list is empty, stop, the incumbent is the optimal solution. If there are more than one problem to choose from, select the one with the smallest lower bound. Let k be the number of the additional constraints, then we have $x_{b1m1}=1$, $x_{b2m2}=1$, ..., $x_{bkmk}=1$. Let B={b1, b2, ..., bk} and C={i; $1 \leq i \leq I$, and $i \notin B$}. Let D={m1, m2, ..., mv} and E ={m; $1 \leq m \leq M$, and $m \notin D$}. For each m in E, create a candidate problem with the additional constraint $x_{bk'm}=1$ where k'=k+1. Delete the original candidate problem.

4. Calculate the lower bound for these candidate problems by solving the problem

(CP) where $c(j,n) = c1(j,n) + \sum_i w_{bi,j} d_{mi,n}$

and $c1(j,n) = \text{Min} \sum_i \sum_m w_{ij} d_{mn} X_{im}$

subject to

$$\sum_i X_{im} = 1 \text{ for } i \in C \quad (1)$$

$$\sum_m X_{im} = 1 \text{ for } m \in E \quad (2)$$

$$X_{im} = 0 \text{ or } 1 \text{ for all } i \in C \text{ and } m \in E$$

5. If the lower bound represents a feasible solution that is less than the incumbent, update the incumbent. If the lower bound is greater than or equal to the incumbent, discard the candidate problem. Otherwise, add the candidate problem to the list. Go to step 3.

COMPUTATIONAL EXPERIENCE

The algorithm is implemented in C and tested on a Sun IPC with 8M RAM and 25 MHz clock. The W matrix is generated with random numbers uniformly distributed between 0 and 9; the D matrix is also generated randomly with numbers uniformly distributed between 50 and 100. Many runs and efforts are made to speed up the convergence of the algorithm. It turns out that if c1(j,n) in step 3 is estimated by the sum of row minimums instead of the actual minimum, a better CPU time can be obtained. It is also observed that the incumbent improves frequently at the early stage, then few or no changes are made, and finally the last portion of the CPU time is spent proving that the incumbent is optimal. Problems of different sizes are run, two problems are solved for each size combination, and the computational results are given in Table 1.

Table 1. Computational Results

Size of X I x M	Size of Y J x N	Average Run Time in seconds	% of time improving incumbent
5x6	5x6	5	70
5x6	7x8	6	35
5x6	9x10	9	46
6x7	6x7	13	35
6x7	8x9	23	46
6x7	10x11	50	43
7x8	7x8	84	26
7x8	9x10	109	19
7x8	11x12	344	31
8x9	8x9	969	31
8x9	10x11	3512	77
8x9	12x13	5526	11
9x10	11x12	25285	12

CONCLUSION

There is a trade-off between the tightness of the lower bound and the number of candidate problems examined. When the lower bound is as high as possible, we have less candidate problems to examine because they get pruned at an earlier stage. When the lower bound is not as high, more candidate problems are examined. However, it only pays if the time spent calculating the lower bound outweighs the time spent examining the candidate problems. It is apparent that the CPU time increases dramatically as the size of the problem increases. From the computational experience, it also appears that the optimal solutions are obtained at an early stage, and the rest of the time is spent proving that the incumbent is optimal. For a large problem that dictates we must terminate the algorithm prematurely, we are confident that the incumbent will be reasonably close to optimal. However, further research is needed to prune the candidate problems at some early stage, so that larger problems can be solved within a reasonable amount of time.

REFERENCES

Tsui, L.Y., C.H. Chang, "A Microcomputer Based Tool For Assigning Dock Doors In Freight Yards," Computers & Industrial Engineering, Vol 19, 1990.

Thieu, T.V., "A Note On The Solution Of Bilinear Programming Problems By Reduction to Concave Minimization," Mathematical Programming, 41, 1988.

Al-Khayyal F.A., J.E. Falk, "Jointly Constrained Bicovex Programming," Mathematics of Operations Research, Vol 8, No.2, May, 1983.

Sherali H.D., C.M. Shetty, "A Finitely Convergent Algorithm for Bilinear Programming Problems Using Polar Cuts And Disjunctive Face Cuts," Mathematical Programming, 19, 1980.

Gallo G., Ulkulu A., "Bilinear Programming: An Exact Algorithm," Mathematical Programming, Vol 12, 1977.

Vaish H., C.M. Shetty, "A Cutting Plane Algorithm For The Bilinear Programming Problem," Naval Research Logistics Quarterly, Vol 24, 1977.