

Positioning of goods in a cross-docking environment

Iris F.A. Vis

Vrije Universiteit Amsterdam, Faculty of Economics and Business Administration, De Boelelaan 1105, Room 3A-31, 1081 HV Amsterdam, The Netherlands

Kees Jan Roodbergen

RSM Erasmus University, P.O. box 1738, 3000 DR Rotterdam, The Netherlands

Please refer to this article as:

Vis, I.F.A., Roodbergen, K.J. (2008), Positioning of goods in a cross-docking environment. *Computers & Industrial Engineering* 54(3), 677–689.

Abstract

Cross docking is one of the options to reduce lead times and inventories and to improve customer response time in supply chains. Cross-docking centres are dynamic environments where products arrive, are regrouped, and leave the same day. In this paper we focus on the process of short-term storage of unit-loads in a cross-docking environment. The goal is to determine temporary storage locations for incoming unit loads such that the travel distances of the forklift trucks with these unit loads are minimised. We model this problem as a novel application of the minimum cost flow problem and show the applicability of the model for different types of layouts and priorities. We demonstrate both the efficiency and effectiveness of the method in the operational and design phase at cross docking environments by applying it to practice-oriented examples. Furthermore, we show that the approach is superior to a commonly used heuristic method.

Keywords: logistics; distribution; cross docking; short-term storage assignment; networks and graphs; practice of OR

Introduction

Warehouses are under a continuous pressure to decrease cost and to reduce the duration of stay of the products. In some cases it is possible to almost eliminate storage by using cross docking. Cross docking basically means that products are unloaded from a truck and subsequently loaded into other trucks. Loads are divided over the loading trucks based on their destination. Unit loads (e.g. a pallet with products) can be stored on the floor between unloading and loading for a few hours to wait for a truck to arrive. However, products are not stored for longer periods. This way, cross docking enables very short lead times for products.

For a successful implementation of cross docking, attention could be paid at least to the following factors (see Moore and Roy, 1998 and Schaffer, 1997). Firstly, the supply must be very reliable with short lead times, because there is no stock available to buffer between supply and demand. Products should be delivered at the right time, in the right quantity and of the right quality. Otherwise, trucks will travel too late or half-empty to customers. Secondly, information should be available through the entire supply chain. Arrival times, departure times and destinations have to be available in time such that the planning of trucks can be done in advance. Thirdly, to ensure that products arrive and depart on time, trucks should arrive on time. Transport should be reliable to prevent delays in delivery to customers. Furthermore, products with high demands and products with highly predictable demands are more suitable (see Richardson, 1999). Though it must be noted that transport volume itself can be increased by consolidating multiple orders (see Klineciewicz and Rosenwein, 1997).

Except for the external factors mentioned, cross docking can only be successful if the design of the cross-docking centre and the operating policies for all processes are executed efficiently. Potential advantages of cross docking are, for example, reduction of inventories

and associated costs, shorter lead times, improvement of service to customers, improvement of relations with suppliers and the possibility to take real-time decisions. Schaffer (1998) discusses in detail how cross docking can improve efficiency.

In a typical situation, the cross docking centre consists of a number of dock doors where trucks can load or unload. These dock doors can be located along any of the walls. Trucks arrive according to a schedule to load or unload. First trucks arrive that need to be unloaded. Each arriving unit load, which is not immediately transferred to another truck, has to be assigned to a storage location. General-purpose forklift trucks generally perform transport between trucks and storage locations. When (almost) all incoming loads have been received, trucks start arriving for loading. If a truck arrives for loading, all pallets for this destination are moved from their storage locations to this truck.

Figure 1 gives an illustration of a possible layout of a cross-docking centre.

INSERT FIGURE 1

Cross docking is an interesting subject, both from a research perspective as well as from a practical point of view. Many organisations are gradually turning to the concept of cross docking and are searching for efficient methods to regulate the process. Most cross-docking operations currently are run using simple heuristics. Only little research has been performed on the subject of cross docking. Even though the number of different operations in a cross-docking centre is small compared to a full-service warehouse, the planning issues that do exist, are of a complex nature.

For example, an important planning problem in cross-docking centres concerns the dock door assignment. The issue is to assign incoming and departing trucks to dock doors such that the operations inside the facility can be performed as efficiently as possible. Tsui and Chang (1990) present a heuristic to assign trucks to dock doors such that the total travel distance in the facility is minimised. A branch-and-bound procedure for this problem is described in Tsui and Chang (1992).

Gue (1995) constructed an LP-model to determine the material flow in a facility depending on a parameter that determines the influence of the supervisor on the assignment of incoming trailers to dock doors. A near-optimal assignment of destinations to unloading dock doors is determined with a local search method that uses 2-interchange to alter trailer assignments to dock doors, and calculating material handling requirements in each step using the LP-model. Bartholdi and Gue (2000) use a simulated annealing approach to interchange designations of dock doors (i.e. which trailers load at which doors). The objective is to minimise workers travel time and waiting time due to congestion.

In warehouses, storage of goods is common. Within warehouses products are stored, contrary to cross-docking centres, for a longer period of time. There exist numerous ways to assign products to storage locations in warehouses. For example, the random storage policy assigns every incoming load to a location that is selected randomly from all eligible empty locations with equal probability. Furthermore, several storage policies have been developed that take product turnover into account. This means that products with the highest sales rates are located at the easiest accessible locations. For more detailed information on storage assignment in warehouses, refer to Van den Berg (1999). The dynamics in cross docking are, however, very different from warehousing. There is, for example, no need to distinguish products by demand frequency, because each unit load arrives and leaves the same day, without any further processing. If we assume that for any unit load the unloading dock door and the loading dock door is known, which is not unusual, then we can determine the best storage location based on the full distance travelled in the entire facility. This in contrast with

the retrieval processes in warehouses, where often only the distance within one specified area is minimised (see e.g. Ratliff and Rosenthal, 1983).

In this paper we seek to determine short-term storage locations such that the total travel distances in the entire facility are minimised. In the next section we describe the problem and assumptions and give the “row-based storage assignment algorithm” which can be used to solve the problem in polynomial time. Other approaches of the problem and related solution approaches are discussed thereafter. An example of the “row based storage assignment algorithm” is presented. Finally, we discuss the results of various numerical experiments to demonstrate the efficiency and performance of our algorithm. At the end, conclusions are given.

Model and Algorithm

At the cross-docking centre we observe loads arriving at i dock doors at one side of the centre and leaving at j dock doors at the opposite side (some of our initial assumptions will be relaxed in the next section). Trucks bring and take away loads from the centre. The storage area is divided into several parallel rows. Via these rows vehicles or workers can travel to transport loads from unloading doors to loading doors. Unit loads are stored on the ground or in racks where they can be accessed directly.

At first sight, the mentioned characteristics seem to match those of a transshipment problem (see, e.g., Hillier and Lieberman, 2001). In a transshipment problem, a network arises consisting of factories as origins (compare to our unloading dock doors), warehouses as intermediate nodes (compare to our storage locations), and retailers as destinations (compare to our loading dock doors). The goal of the transshipment problem is to find a minimum-cost solution that satisfies all demand at all destinations. The main difference with our problem is that in the transshipment problem loads are interchangeable. That is, demand of a destination may be met by loads from any of the origins, whichever results in the lowest costs. In a typical cross-docking setting, however, incoming loads are unique in the sense that their final destination is given and cannot be changed. In this section, we will present a network representation that takes this additional restriction into account.

We make the following assumptions:

1. From the dock door assignment it is known for each truck at which door it arrives. Several authors have addressed this issue; see for example Tsui and Chang (1990, 1992), Gue (1999) and Bartholdi and Gue (2000).
2. Available capacity of each storage location and as a result the capacity of each row is known. This information can be easily obtained from the information system.
3. At each unloading dock door the exact aggregated quantity of unit loads designated for each of the loading dock doors is known. This is a direct consequence of assumption 1 combined with the requirement in cross docking that a good information system – preferably with EDI – is present (see for example Schaffer, 1997).
4. Total capacity of the facility is sufficient to store all incoming freight.

The objective of this section is to develop a model and polynomial time “row based storage assignment” algorithm to solve the problem of determining storage locations for arriving unit loads at a cross-docking centre such that the total travel distance within the facility is minimised. The algorithm can be used multiple times a day in a so called *block* assignment approach. We define a block here as a time period of predefined length. At the start of a time block, the algorithm assigns all loads that will arrive during that block, to empty storage locations. As time passes of the current time block, previously stored loads

might be retrieved from their storage locations for loading. As a result, these locations – in addition to those that remained empty – can be used in the assignment procedure at the start of the next block. As a result, storage locations can be used multiple times per day.

For each unit load a certain distance has to be travelled from the unloading dock door, via the storage location, to the loading dock door. For each combination of an unloading and a loading dock door, there exists a minimum distance path between the two doors. To tranship a unit load from its unloading to its loading door at least this minimum distance has to be travelled. During the transport through row(s) located on the shortest path numerous storage locations will be passed. If the unit load would be stored at such a location, then no extra distance would have to be travelled with the load to store it as compared to direct transfer. If none of the locations in row(s) on the shortest path is free, then a storage location in another row needs to be chosen. Consequently, an extra distance has to be travelled to store the unit load. We interpret this extra distance as costs to be made to store the unit load. Clearly, these costs are considered zero in the case that a storage location on the shortest path between the two dock doors is used.

The following parameters of the layout of the cross-docking centre need to be collected before the algorithm can be applied. Usually, this information is collected once and only needs to be updated if there is a change in the physical appearance of the facility.

- N Number of unloading dock doors in the cross-docking centre.
- M Number of loading dock doors in the cross-docking centre.
- R Number of storage rows in storage area in the cross-docking centre.
- l Length of each storage row
- ρ Width of each storage row
- z_k Storage capacity of row k ($1 \leq k \leq R$). The value of this parameter is related to the length of each row, the size of each storage location and the number of locations at each side of the row.
- D_{ij} Length of the shortest paths between each pair of unloading door i ($1 \leq i \leq N$) and loading dock door j ($1 \leq j \leq M$).
- C_{kij} Costs to store a unit load in row k ($1 \leq k \leq R$) if the unit load is transhipped from door i ($1 \leq i \leq N$) to door j ($1 \leq j \leq M$). These costs are zero if row k is on the shortest path from door i to door j .

Before each calculation cycle, the following information (parameter) needs to be available:

- x_{ij} number of unit loads with origin at door i ($1 \leq i \leq N$) and destination at door j ($1 \leq j \leq M$). The sum of all x_{ij} will be denoted as X .

The solution of the algorithm needs to indicate for each combination of unloading and loading dock doors how many loads are stored in each of the storage rows. The goal is to minimise the total distance to be travelled to store and retrieve all loads. The variable that can be changed to achieve the best solution is defined as follows.

- $y_{v_{ij}w_k}$ number of unit loads ($1 \leq y_{v_{ij}w_k} \leq x_{ij}$) with origin at door i ($1 \leq i \leq N$) and destination at door j ($1 \leq j \leq M$) that are stored in row k .

To explain the notation for this variable, we first have to define our network. We construct a directed network G where each flow of unit loads from an unloading dock door i ($1 \leq i \leq N$) to a loading dock door j ($1 \leq j \leq M$) is represented by a node v_{ij} . For example, in the case that 3 unloading and 3 loading dock doors exist, 9 nodes represent the various flows.

Namely, a node representing the flow of unit loads from unloading dock door 1 to loading dock door 1, a node representing the flow of unit loads from unloading dock door 1 to loading dock door 2 and so on.

For each node v_{ij} the total number of unit loads to be shipped from i to j is known and equals x_{ij} . To include these quantities in the directed network, we introduce a source node S . Directed arcs are added from source node S to the nodes v_{ij} for all i and j . The lower bound and upper bound of the flow $y_{Sv_{ij}}$ on each directed arc (S, v_{ij}) equal x_{ij} .

Each storage row k ($1 \leq k \leq R$) is represented by a node w_k . The storage capacity of row k is limited to z_k . To incorporate these capacities into the directed network, we introduce a sink node T . Directed arcs are added from each node w_k to the sink node T . The number of unit loads stored in row k equals the flow $y_{w_k T}$ on the arc (w_k, T) . The lower bound of the flow on each of these arcs equals zero. The upper bound of each directed arc (w_k, T) equals z_k . Consequently, the maximum number of unit loads assigned to a certain row equals the maximum capacity of the row. Note that also this network formulation does not meet the requirements for a transshipment problem since not every arc in this network can carry any desired amount of flow which is a strict requirement for a transshipment problem (see Hillier and Lieberman, 2001).

The objective is to assign unit loads to storage locations such that travel distances with loaded vehicles are minimised. By connecting nodes v_{ij} with nodes w_k we can derive an assignment of unit loads arriving at door i and leaving at door j to a storage row k . Thus, we add directed arcs (v_{ij}, w_k) for all i, j, k to the network. As explained, extra travel distances may occur if unit loads with a known origin and destination are assigned to a storage row that are not on the shortest path from i to j . These costs C_{kij} are assigned to the directed arcs (v_{ij}, w_k) for all i, j, k . The costs of all other arcs equal zero.

Finally, we add a directed arc (S, T) to ensure that the flow leaving from the source S equals the flow arriving in the sink node T . The lower and upper bound of this flow y_{TS} equal the total number of unit loads X to be transhipped from the unloading dock doors to the loading dock doors. In all nodes the incoming flow equals the outgoing flow.

Thus, we have defined a directed network $G = (V, A)$ with

$$\begin{aligned} V &= \{v_{ij} \mid 1 \leq i \leq N \text{ and } 1 \leq j \leq M\} \\ &\cup \{w_k \mid 1 \leq k \leq R\} \\ &\cup \{S, T\} \end{aligned}$$

$$\begin{aligned} A &= \{(v_{ij}, w_k) \mid v_{ij}, w_k \in V\} \\ &\cup \{(S, v_{ij}) \mid v_{ij} \in V\} \\ &\cup \{(w_k, T) \mid w_k \in V\} \\ &\cup \{(S, T)\} \end{aligned}$$

With the lower and upper bounds

$$\begin{aligned} x_{ij} &\leq y_{Sv_{ij}} \leq x_{ij} \\ 0 &\leq y_{w_k T} \leq z_k \\ X &\leq y_{TS} \leq X \end{aligned}$$

The objective is to find flows $y_{v_{ij}w_k}$ from nodes v_{ij} to nodes w_k in the directed network such that the total costs are minimised and the capacity constraints are met. We can use any minimum cost flow algorithm to obtain the values of $y_{v_{ij}w_k}$. For example, the ‘‘enhanced capacity scaling algorithm’’ can be used. This algorithm searches for arcs (i, j) with large

flows and then merges nodes i and j in a single node and continues from there with its computations. For a more detailed explanation of this method and various other minimum cost flow algorithms, see Ahuja *et al.* (1993). The mentioned algorithm has a time complexity of $O((m \log n)(m + n \log n))$, where n is the number of nodes and m the number of arcs in the network.

Summarising, our “row based storage assignment algorithm”, which can be used to determine storage locations for arriving unit loads at a cross-docking centre such that total travel distances are minimised, consists of the following steps.

- Step 1: Collect information on the layout of the cross-docking centre, including the number of unloading (N) and loading (M) dock doors, the number of storage rows (R) and the length (l) and width (ρ) of each storage row.
- Step 2: Create a matrix D_{ij} that contains the length of the shortest paths from each unloading dock door i to each loading dock door j .
- Step 3: Determine the length of every path a load might take from any unloading door via any row to any loading door. Organise and recalculate these distances to obtain a two-dimensional cost matrix C_{kij} containing “extra distances”, calculated as the actual distance from unloading door i to loading door j via storage row k minus D_{ij} (the distance of the shortest path from i to j). Each matrix row corresponds to a storage row and each column corresponds to a combination of an unloading and loading dock door.
- Step 4: Select a time period T (i.e., block) for which all incoming loads X need to be assigned to empty storage locations.
- Step 5: Determine the number of unit loads x_{ij} to be transhipped from each unloading dock door i to each loading dock door j during the next time period T .
- Step 6: Create network G using the data from steps 1 – 5.
- Step 7: Apply any minimum cost flow algorithm to obtain an optimal storage assignment solution.
- Step 8:
 - If there are any time periods T left, then continue with step 5.
 - If not, continue with step 9.
- Step 9: Done.

In the sections “illustrative example” and “numerical experiments” we will show how the algorithm can be used. In the next section, we will describe some variations on our model and algorithm.

Simplification and extensions of the algorithm

Simplification for symmetric layouts

For cross-docking centres with a symmetric layout we can speed up the “row based storage assignment algorithm”. In a symmetric layout the number of loading dock doors equals the number of unloading dock doors. Furthermore, the rows used for temporarily storage are positioned exactly between the loading and unloading dock doors. As a consequence, the total number of storage rows R equals the number of unloading dock doors and equals the number of loading dock doors. For an illustration of a symmetric layout, refer to Figure 2. In this specific situation the storage costs for the flow of unit loads from door i to door j equal the storage costs for the flow of unit loads from door j to door i . As a result, it is possible to use one node v_{ij} for flows between door i to door j in both directions. Instead of using R^2 nodes to represent the flow of unit loads from one dock door to another we now just need $\frac{1}{2} R*(R+1)$ nodes.

The lower and upper capacity of the flow on the arcs (s, v_{ij}) will be equal to $x_{ij} + x_{ji}$. The rest of the solution procedure remains the same. The example in the next section illustrates this reduction of the number of nodes in the network.

Zone-based storage

In the “row-based storage assignment algorithm” the costs for each storage location in a row are equal. Time pressure is, however, often higher for loading trucks than for unloading. Therefore, it might be useful to try and locate the unit loads as close to the loading doors as possible. To this end, we divide each row into p zones. We replace the nodes, which represent rows by a series of nodes representing the zones. Since we are now only interested in the distance to the loading dock, we define the costs C_{kij} as the distance from zone k ($1 \leq k \leq p \cdot R$) to loading dock door j . Furthermore, we need to redefine z_k to be the storage capacity of zone k ($1 \leq k \leq p \cdot R$).

Location-based storage

We can take the adaptation we made for zone-based storage one step further by taking into account the individual storage locations. This means that p will now represent the number of locations in a row. The advantage is that we can more precisely determine the most appropriate storage location for each load. The downside is that the number of nodes in the network increases, which will increase computation times.

Other layouts of the cross-docking centre

Until so far we have studied cross-docking centres with a so-called I-layout (see Figure 1), where loading and unloading occurs on opposite sides of the building. Other layouts may also be used in practice. For example, a U-layout with all dock doors on the same side or a layout where loading and unloading doors are intermingled. In the original algorithm we defined the costs C_{kij} as the excess travel time compared to the shortest path. If the loading and unloading doors are on the same wall, however, the shortest path does not pass any storage locations. Therefore, the excess travel time cannot be easily defined. Furthermore, if loading and unloading can occur at the same side of the building, we will need to know the exact storage location within the row, because each location of a row might result in a different travel times. To solve this problem, we just use the adapted version for location-based storage, which already identifies individual storage locations. If loading has priority over unloading, then no changes have to be made. If loading and unloading are seen as equally important, the costs C_{kij} can be redefined as the actual distance to go from dock door i to dock door j via storage location k .

Illustrative example

To illustrate the application of our algorithm for a symmetric layout, we discuss the following illustrative example. We consider a cross-docking centre with a symmetric layout. There are three unloading dock doors ($N = 3$) and three loading dock doors ($M = 3$). Between each pair of opposite doors a row is positioned for temporary storage of unit loads. Consequently, the value of R equals three. This layout with distances (in metres) is illustrated in Figure 2.

INSERT FIGURE 2

With the data presented in Figure 2 we construct the following matrix D_{ij} containing the length of the shortest paths between each pair (i, j) for $1 \leq i \leq 3$ and $1 \leq j \leq 3$:

$$D_{ij} = \begin{pmatrix} 50 & 60 & 70 \\ 60 & 50 & 60 \\ 70 & 60 & 50 \end{pmatrix}$$

In the case that unit loads are not stored in the row which can be traversed on the shortest route from door i to door j extra distances need to be travelled. These costs C_{kij} are presented in the following matrix. The rows of the matrix indicate the rows ($k = 1, 2, 3$) in which the storage can occur. Each column represents an unloading door in combination with a loading door. Namely, unloading door 1 combined with loading door 1, unloading door 1 combined with loading door 2, ..., unloading door 3 combined with loading door 3. The elements in the matrix C_{kij} represent the extra costs in the case that unit loads are transhipped from door i ($1 \leq i \leq 3$) to door j ($1 \leq j \leq 3$) via temporarily storage in row k ($1 \leq k \leq 3$).

$$C_{kij} = \begin{pmatrix} 0 & 0 & 0 & 0 & 20 & 20 & 0 & 20 & 40 \\ 20 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 20 \\ 40 & 20 & 0 & 20 & 20 & 0 & 0 & 0 & 0 \end{pmatrix}$$

Each row k ($1 \leq k \leq 3$) has a certain (remaining) storage capacity z_k . These capacities are presented in the following matrix:

$$z_k = \begin{pmatrix} 75 \\ 80 \\ 150 \end{pmatrix}$$

For each unloading dock door i it is known which quantity of unit loads need to be transhipped to each of the loading dock doors j . These quantities x_{ij} are given in the following matrix:

$$x_{ij} = \begin{pmatrix} 80 & 10 & 10 \\ 20 & 70 & 10 \\ 5 & 45 & 50 \end{pmatrix}$$

From matrix x_{ij} it can be concluded that in total 300 unit loads need to be transhipped. Consequently, the value of X equals 300. With these data we construct the directed network G . From the proposed simplification in the previous section, we know that we can use one node to represent the flow of unit loads from door i to door j and from door j to door i . For example, we can use the node v_{12} to represent the flow from unloading door 1 to loading door 2 and for the flow from unloading door 2 to loading door 1. Clearly, we can decrease the total number of nodes in the directed network from 9 to 6.

The costs remain the same on the directed arcs. However, the quantities to be transhipped, which are indicated in x_{ij} need to be transformed. For example, x_{12} becomes $x_{12} + x_{21} = 10 + 20 = 30$. The new matrix x'_{ij} is:

$$x'_{ij} = \begin{pmatrix} 80 & 30 & 15 \\ - & 70 & 55 \\ - & - & 50 \end{pmatrix}$$

Figure 3 presents the directed network G related to the data given.

INSERT FIGURE 3

The objective is to assign the unit loads to storage locations such that the total travel distance is minimised. Therefore, we determine a flow in the directed network such that the

extra distances to be travelled are minimised. By applying a minimum cost flow algorithm the following solution is obtained (all other variables have value zero):

$$\begin{aligned}
 y_{v_{11}w_1} &= 75 & y_{v_{13}w_3} &= 15 \\
 y_{v_{11}w_3} &= 5 & y_{v_{22}w_2} &= 70 \\
 y_{v_{12}w_2} &= 10 & y_{v_{23}w_3} &= 55 \\
 y_{v_{12}w_3} &= 20 & y_{v_{33}w_3} &= 50
 \end{aligned}$$

From these values it can be concluded that almost all unit loads can be stored at a storage location along the shortest path from their origin to their destination. However, some of the unit loads need to be stored in a row, which is not traversed on the shortest route from the origin to the destination of the unit load. As a result, an extra distance of 600 metres needs to be travelled to tranship all unit loads from their origin to their destination via a storage location.

Numerical Experiments

The objective of this section is to demonstrate the efficiency and performance of our row-based storage assignment algorithm in both the operational and design phase for various cross-docking environments. To this end, we compare optimal results of our approach with the results from a heuristic method that is commonly used in practice, namely “the nearest location first heuristic”. The logic of this heuristic is based on a situation in which the employees are allowed to select storage locations themselves, which typically results in a load to be stored at the empty location nearest to the load’s origin. Both the algorithm and the heuristic have been implemented in Delphi.

For both methods, we calculated average total travel distances in various experiments. In this way, we can easily compare both methods and indicate actual benefits in total workload instead of comparing the absolute values of extra travel distances for loads that have not been stored on their shortest path. To obtain valid estimates of the average total travel times, the number of replications for each experiment has been determined (see Law and Kelton, 2000). To obtain a relative error smaller than 1% with a probability of 95%, a replication size of 2000 appeared to be sufficient for all experiments in this paper.

First, we consider a set of representative layouts for the cross-docking centre, containing both non-symmetric and symmetric layouts. Thereafter, we perform additional experiments for a fixed layout while varying the values of some of the input parameters. In this way, we can study the influence of parameters on the solutions of both the heuristic and optimal algorithm and their mutual differences. At the end, we demonstrate how our algorithm can be used in designing a cross-docking centre.

Comparison for a set of layouts

As mentioned by Gue (1999), cross-docking centres usually range between 6 and 200 dock doors. In the following experiments, we compare the performance of our algorithm and the nearest location first heuristic for terminals ranging from small (50 doors) to large (200 doors). Specifically, we vary the values of N (i.e., the number of unloading dock doors), M (i.e., the number of loading dock doors) and R (i.e. number of storage rows) between 25 and 100 as indicated in Table 1. For reasons of symmetry no instances with $L \geq M$ need to be included.

The length of each row equals 25 metres. Loads can be stored on both sides of each row at one metre wide locations of unit capacity (see Figure 1). As a result, the capacity of each row equals 50 loads. The width of each row equals 6 metres to allow sufficient space for both storage locations and for travelling through the row. Typically, larger cross-docking

centres have more arriving loads than smaller centres. To incorporate this in our experiments in a way that still allows mutual comparisons, we fix the average number of arriving loads per dock. Note that this only fixes the average per door across replications; individual observations are allowed to vary. Moreover, arriving trucks are in practice typically not evenly distributed over all incoming dock doors. Usually, the doors in the middle of the centre are the busiest (see Bartholdi and Gue, 2000). To emulate this effect, we assume that 75% of the loads arrive at the middle one third of the doors. In the next section, we will vary the value of this parameter to demonstrate its effect on the results of the optimal algorithm and the heuristic.

Table 1 presents average total travel distances in metres for both the heuristic and the optimal algorithm. Approximate calculation times for the optimal algorithm on 1.7 GHz computer vary for a single replication between 0.05 seconds for the smallest layout up to 18 seconds for a layout with 200 dock doors. In practice, only a single run is required and as a result these computation times demonstrate that the algorithm can easily be applied.

INSERT TABLE 1

From the results in Table 1 we can conclude that our algorithm outperforms the heuristic method for both symmetric and non-symmetric layouts. If the cross-docking environment becomes larger the differences between the optimal and heuristic approaches even increase from 27% to 33% for symmetric layouts. For non-symmetric layouts smaller differences can be noticed, however, the results still demonstrate the superiority of the optimal method over the heuristic method. Considering the limited computation times and the results in Table 1, it can be concluded that the algorithm is both efficient and effective.

Varying the arrival pattern of incoming loads

In the next experiment, we vary the percentage of the loads arriving at the middle one third of the doors from 33% (which results in an evenly distribution of loads over all doors) up to 99%. The experiment has been performed for a small symmetric cross-docking environment with 50 dock doors, 25 storage rows and 1000 arriving loads. The values of the other parameters remain as before. Figure 4 shows the resulting average total travel distances in metres for both methods.

INSERT FIGURE 4

The difference between the optimal method and the heuristic method varies from 8% to 39% if we vary the percentage of arriving loads at the middle dock doors from 33% to 99%. In the fairly unusual situation that loads are exactly evenly distributed over the various doors, the difference between the optimal and heuristic method is at its smallest. This can be explained as follows. Both the optimal and heuristic method try to use space in the aisle(s) directly opposite the dock door where a load arrives. As long as fewer loads arrive at a door than storage capacity is available in the opposite row, essentially no performance differences occur between the algorithm and the heuristic. If more loads than the nearest aisle's capacity arrive at a door, which is a common situation in practice, the nearest row will be full and then the heuristic dramatically fails in finding the most efficient locations compared to the optimal method. As a result, the difference between the two methods increases up to almost 40%. Furthermore, from this experiment it appears that our algorithm is less sensitive to dock door assignment policy changes than the heuristic.

Varying available storage capacity

Within cross-docking environments the available storage capacity may fluctuate from completely occupied to almost empty. In the next experiment, we will examine in which way both the optimal and heuristic method respond to a varying amount of available storage locations. We define f as the available storage capacity divided by the number of incoming loads. We study a cross docking environment with a total of 50 dock doors and 25 storage rows each with a length of 60 metres. The values of the other input parameters are the same as in the previous experiment. We vary the value of f between 1 (100% occupancy) and 2 (twice as much storage capacity as needed) with increments of 0.1. Figure 5 presents the percentage difference in performance between the optimal and heuristic method.

INSERT FIGURE 5

From the results in Figure 5 it can be concluded that the performance of the heuristic method slightly improves compared to the optimal method if the amount of available storage capacity increases. However, the difference between both methods is still 14% even when there is twice as much capacity available as there are loads arriving. Quite likely, less storage space may be available at times. In that case, differences up to 33% can occur.

Designing a cross-docking centre

So far, we have demonstrated our algorithm in the operational phase by assigning incoming loads to storage locations. However, the algorithm can also be used in the design phase of a cross-docking centre. Namely, given a required storage capacity, number of unloading and loading dock doors and expected number of arriving loads, the algorithm can be used to determine an optimal number of storage rows and their length.

As a demonstration, we study the design of a cross docking centre that needs 25 unloading and 25 loading dock doors. The required storage capacity equals 2000 loads, so total length of all rows together must be 1000 metres. We vary the number of rows from 25 to 60. The length of each row clearly depends on the number of rows. For example, if we have 50 rows, each row equals 20 metres to cover the total required storage capacity of 1000 metres. Actually, each metre represents a storage location and as a result, we need to round the length of each row to an integer number. Therefore, it can occur that adding an extra row cannot result in a decrease of the length of each of the rows. For example, if we have 51 rows each row still needs to be 20 metres. Arriving loads are assumed to be on average evenly distributed over the various dock doors and with an average of 50 loads per door. Figure 6 presents average total travel distances for a varying number of storage rows by performing 2000 replications for each configuration.

INSERT FIGURE 6

An optimal number of rows can be found by searching for the minimum average total travel distance among all instances. While gradually increasing the number of rows from 25 to 42 and decreasing the related row length from 40 to 24 metres the average total travel distances decrease. An optimal number of rows in this setting equals 42. For higher values, the total travel distances start to increase again.

One interesting feature of Figure 6 are the flat sections that appear in the curve on the right. This is due to the effect that the addition of a single row is insufficient to allow a decrease in the row length. Thus, the additional row will increase capacity. This extra capacity is, however, added at the far end of the facility and not needed. Hence it will not be used. Only once sufficiently extra rows are added to allow for a decrease in row length for all

rows, an effect on travel distances will be observed. It can be concluded that the algorithm is also useful as part of a design procedure.

Conclusions

This paper describes a method that is capable of determining storage locations for loads in a cross-docking centre, such that the travel time is minimised for the vehicles that transport loads. The method consists of a network formulation that incorporates loading and unloading dock door locations, travel distances and available storage space in the facility. Solutions can be obtained in polynomial time. The method can also easily be adapted to accommodate other situations. We showed adaptations that enable the applicability of the model for situations with different layouts and for situations where loading incurs a higher time pressure than unloading. Experiments illustrate the effectiveness of the method both in the operational and design phase. Differences up to about 40% with a commonly used heuristic method have been found.

References

- Ahuja, R.K., T.L. Magnanti & J.B. Orlin. (1993). *Network Flows, Theory, Algorithms, and Applications*. New Jersey: Prentice Hall.
- Bartholdi, J.J. & K.R. Gue. (2000). Reducing labor costs in an LTL crossdocking terminal. *Operations Research*, 48(6), 823-832.
- Gue, K.R. (1995). *Freight terminal layout and operations*. Ph.D. thesis. Atlanta, GA: Georgia Institute of Technology.
- Gue, K.R. (1999). The effects of trailer scheduling on the layout of freight terminals. *Transportation Science*, 33(4), 419-428.
- Hillier, F.S. & G.J. Lieberman. (2001), *Introduction to Operations Research*, 7th edition. McGraw-Hill, Boston.
- Klincewicz, J.G. & M.B. Rosenwein. (1997). Planning and consolidating shipments from a warehouse. *Journal of the Operational Research Society*, 48(3), 241-246.
- Law, A.M. & W.D. Kelton. (2000). *Simulation Modeling and Analysis*, third edition. McGraw-Hill, New York.
- Moore, T. & C. Roy. (1998). Manage inventory in a real-time environment. *Transportation & Distribution*, July, 68-73.
- Ratliff, H.D. & A.S. Rosenthal. (1983). Orderpicking in a rectangular warehouse: a solvable case of the traveling salesman problem. *Operations Research*, 31(3), 507-521.
- Richardson, H.L. (1999). Cross docking: information flow saves space. *Transportation & Distribution*, 40(11), 51-54.
- Schaffer, B. (1997). Implementing a successful crossdocking operation. *IIE Solutions*, 29(10), 34-36.
- Schaffer, B. (1998). Cross docking can increase efficiency. *Automatic I.D. News*, 14(8), 34-37.
- Tsui, L.Y. & C.H. Chang. (1990). A microcomputer based decision support tool for assigning dock doors in freight yards. *Computers & Industrial Engineering*. 19(1-4), 309-312
- Tsui, L.Y. & C.H. Chang. (1992). An optimal solution to a dock door assignment problem. *Computers & Industrial Engineering*, 23(1-4), 283-286.
- Van den Berg, J. (1999). A literature survey on planning and control of warehousing systems. *IIE Transactions*, 31, 751-762.

FIGURES
Figure 1:

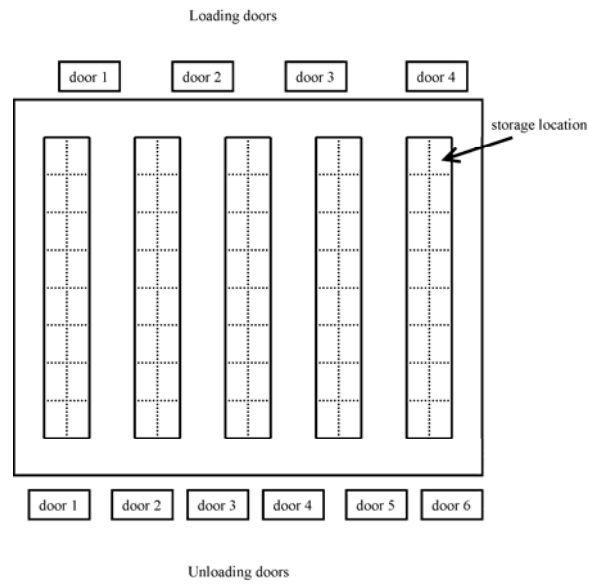


Figure 2:

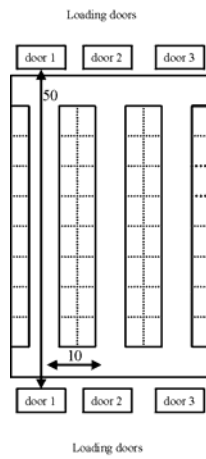


Figure 3:

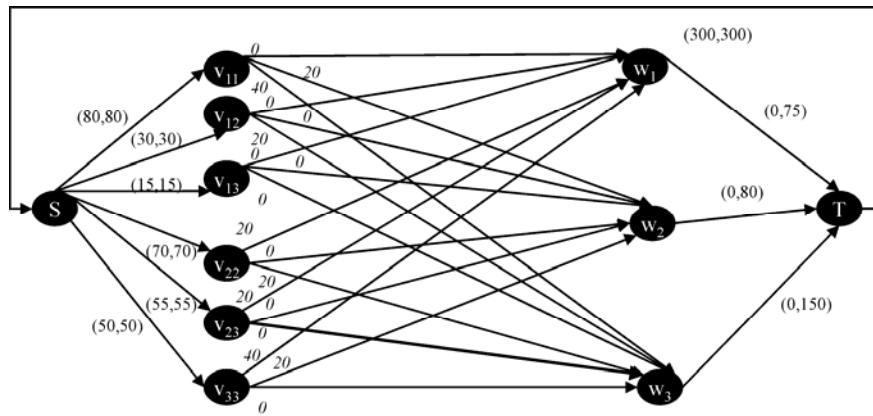


Figure 4:

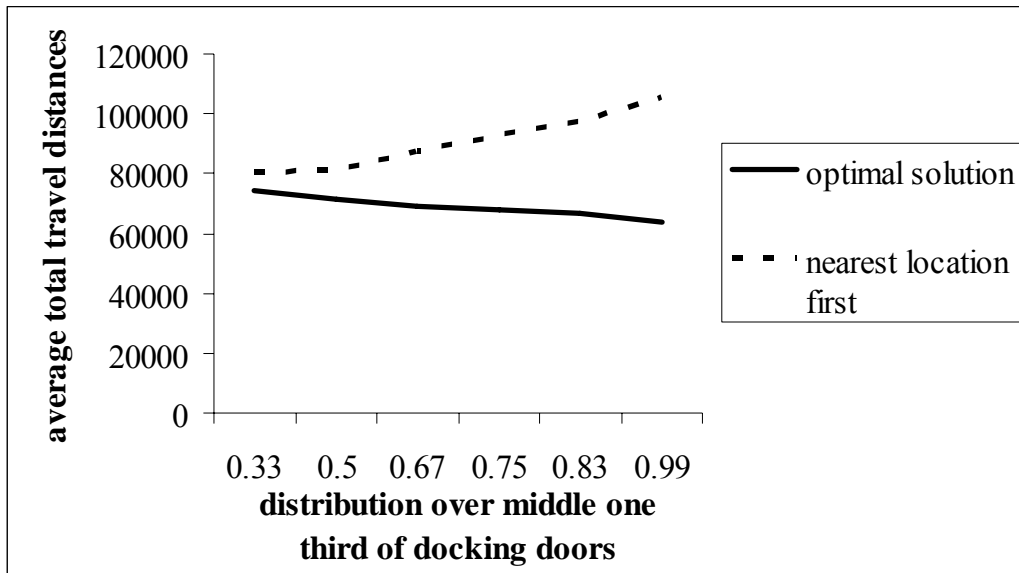


Figure 5:

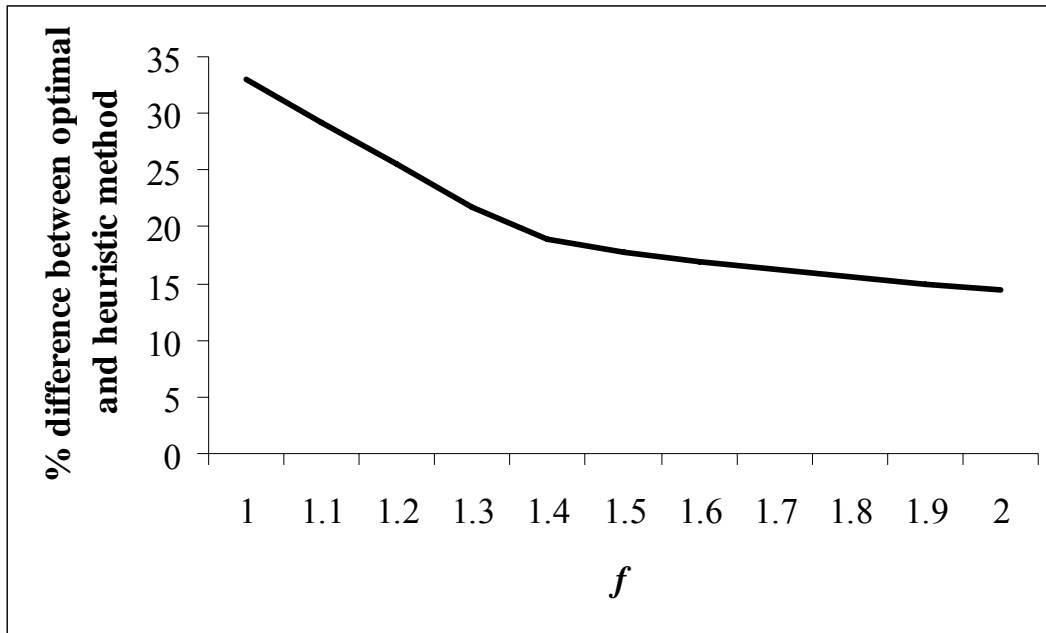


Figure 6:

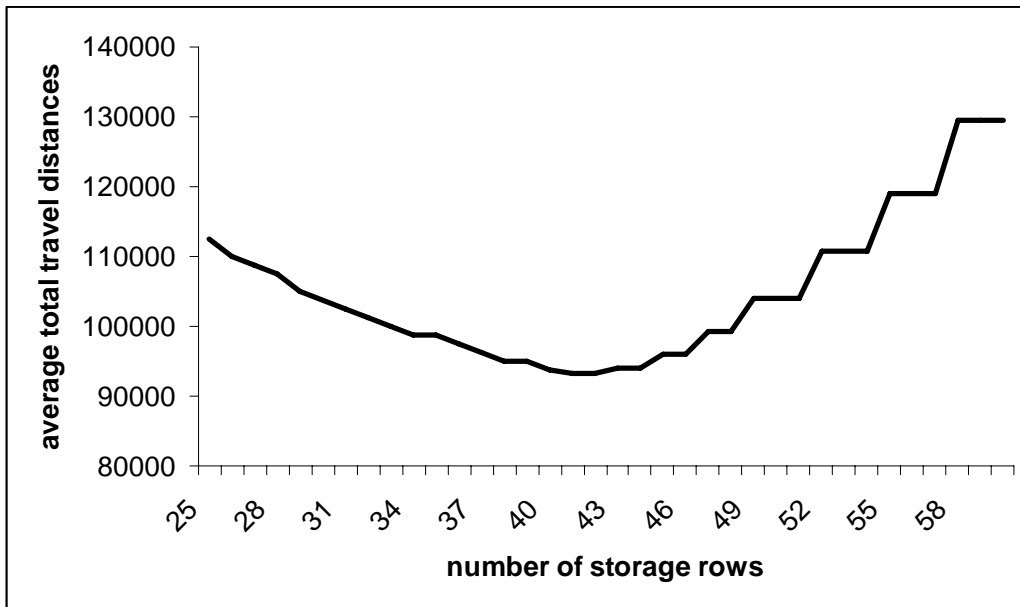


Table 1:

L	M	R	total arriving loads	optimal	heuristic	% difference
25	25	25	1000	67704	92709	26.97
25	50	50	1000	122756	133960	8.36
50	50	50	2000	222092	319671	30.52
50	75	75	2000	316645	376060	15.80
75	75	75	3000	463160	679427	31.83
75	100	100	3000	593426	742993	20.13
100	100	100	4000	789403	1180757	33.14

CAPTIONS FIGURES

Figure 1: Simplified layout of a cross-docking centre

Figure 2: Layout of the cross-docking centre used in the example

Figure 3: directed network G with (lower bound, upper bound) and costs

Figure 4: Average total travel distances in metres for both the optimal and heuristic method as a function of the distribution of arriving loads over the middle one third of the dock doors.

Figure 5: Percentage difference between the optimal and heuristic method as a function of f (the available storage capacity divided by the number of arriving loads).

Figure 6: Average total travel distances in metres for the optimal method as a function of the number of rows. Storage capacity is constant and equivalent to a total row length of 1000 metres.

CAPTIONS TABLES

Table 1: Average total travel distances in metres for both the optimal and heuristic method and the differences between both for different types of layouts and a varying number of arriving loads.