# Routing methods for warehouses with multiple cross aisles

Kees Jan Roodbergen and René de Koster

Erasmus University Rotterdam
Rotterdam School of Management / Faculteit Bedrijfskunde
P.O. box 1738, 3000 DR Rotterdam, The Netherlands
Phone: +31-10-4088723, Fax: +31-10-4089014

**Please refer to this article as:**
Roodbergen, K.J. and De Koster, R. (2001), Routing methods for warehouses with multiple cross aisles. *International Journal of Production Research* 39(9), 1865-1883.

**Abstract**

This paper considers routing and layout issues for parallel aisle warehouses. In such warehouses order pickers walk or drive along the aisles to pick products from storage. They can change aisles at a number of cross aisles. These cross aisles are usually located at the front and back of the warehouse, but there can also be one or more cross aisles at positions in between.

We describe a number of heuristics to determine order picking routes in a warehouse with two or more cross aisles. To analyse the performance of the heuristics, a branch-and-bound algorithm is used that generates shortest order picking routes. Performance comparisons between heuristics and the branch-and-bound algorithm are given for various warehouse layouts and order sizes. For the majority of the instances with more than two cross aisles, a newly developed heuristic appears to perform better than the existing heuristics.

Furthermore, some consequences for layout are discussed. From the results it appears that the addition of cross aisles to the warehouse layout can decrease handling time of the orders by lowering average travel times. However, adding a large number of cross aisles may increase average travel times because the space occupied by the cross aisles has to be traversed as well.

## 1 Introduction

Warehouses form an important link in the supply chain. Products can be stored temporarily in warehouses and customer orders can be filled by retrieving products from storage. However, warehousing generally requires a considerable amount of product handling, which is time consuming. One way to decrease handling time is an entirely new design of the warehouse. But often it is also possible to decrease handling time by less radical methods such as changing the operational procedures.

The order picking process is the process of retrieving products from specified storage locations on the basis of customer orders. The order picking process is in general one of the most time consuming processes in warehouses and contributes for a large extent to warehousing costs (see e.g. Tompkins *et al.* 1996). The productivity of the order picking process depends on factors such as the storage systems (racks), the layout and the control mechanisms. Order picking productivity can be improved by reducing handling time, i.e. reducing the time needed for picking an order. This total picking time can be roughly divided in time for driving or walking to locations (travel time), time for picking the products and time for remaining activities (such as obtaining a picklist and an empty pick carrier). In warehouses with manual picking operations, travel time often forms the largest component of total picking time (Tompkins *et al.* 1996).

Several methods can be used to reduce travel times by means of more efficient control mechanisms. One approach is to determine good order picking routes. The problem of determining order picking routes consists of finding a sequence in which products have to be retrieved from storage such that travel distances are as short as possible. For a warehouse with two cross aisles, one at the front and one at the back, an efficient algorithm to determine shortest order picking routes has been developed by Ratliff and Rosenthal (1983). Heuristics for warehouses with two cross aisles can be found in Hall (1993). Performance comparisons between optimal routing and heuristics for this type of warehouses are given in Petersen (1997) and De Koster and Van der Poort (1998). Another method to reduce travel times is zoning, i.e. an order picker picks only that part of an order that is in his or her assigned zone. Also storage assignment rules can reduce travel times by assigning products to the right storage locations. For example, frequently demanded products can be located where they are easily accessible. Important in this respect is the interaction between the routing method and the storage assignment rule, see e.g. Petersen and Schmenner (1999). Finally, we can think of batching as a means of reducing travel times. Batching is concerned with combining several (partial) orders in a single order picking route. For batching strategies see e.g. Gibson and Sharp (1992), De Koster *et al.* (1999) or Ruben and Jacobs (1999).

This paper will focus on routing methods for warehouses with more than two cross aisles. A preliminary study on heuristic routing in this type of warehouses was done by Roodbergen and De Koster (1998). They compare three heuristics for a number of situations, including a narrow-aisle high-bay warehouse where order picking trucks are used. A routing heuristic, using dynamic programming, for warehouses with more than two cross aisles is presented by Vaughan and Petersen (1999).

In this paper we will extend heuristics, that exist for warehouses with two cross aisles, so they can be used in warehouses with more than two cross aisles. Furthermore, we will present a new routing heuristic, called the *combined heuristic*, and give some improvements for it, which will be tested under the name *combined*$^+$. Like the heuristic of Vaughan and Petersen (1999), the combined heuristic uses dynamic programming to determine order picking routes. The main difference lies in the restrictions on the sequencing of the picks. The performance of all heuristics is compared for the situation of a shelf area where order pickers walk through the warehouse to pick items, using a small pick cart. A branch-and-bound procedure that generates shortest order picking routes is used as a benchmark. It is shown that in most cases, the combined$^+$ heuristic has a better performance than the other methods.

Section 2 describes warehouse layout and routing issues and extensions of some existing routing methods. In section 3 the combined routing heuristic is described to determine routes in a warehouse with two or more cross aisles. Section 4 compares the performance of all routing methods. Section 5 contains conclusions.

## 2 Warehouse layout and routing

A graphical sketch of the warehouse layout considered in this paper is given in figure 1. The warehouse is rectangular with no unused space and consists of a number of parallel *pick aisles*. The warehouse is divided into a number of *blocks,* each of which contains a number of subaisles. A *subaisle* is that part of a pick aisle that is within one block. The term *aisle* is used when a statement holds for both pick aisles and subaisles. At the front and back of the warehouse and between each pair of blocks, there is a *cross aisle*. Cross aisles do not contain storage locations, but can be used to change aisles. Every block has a *front cross aisle* and a *back cross aisle*; the front cross aisle of one block is the back cross aisle of another block, except for the first block.

The number of cross aisles equals the number of blocks plus one. This holds because there is one cross aisle in the front, one in the back and one between each two adjacent blocks.

[Insert figure 1 about here]

Order pickers are assumed to be able to traverse the aisles in both directions and to be able to change direction within the aisles. The aisles are narrow enough to allow picking from both sides of the aisle without changing position. See Goetschalckx and Ratliff (1988) for issues concerning aisle width. Each order consists of a number of items that are usually spread out over a number of subaisles. We assume that the items of an order can and will be picked in a single route. Aisle changes are possible in any of the cross aisles. Picked orders have to be deposited at the depot, where the picker also receives the instructions for the next route. The depot is located at the head of the first pick aisle in the front cross aisle. Note that the location of the depot can potentially influence the average travel time. Petersen (1997) evaluated the effect of depot location for five pick list sizes and four warehouse layouts. On average the difference in route length between a depot located in the corner and a depot in the middle was less than 1%.

In the remainder of this section we describe four different types of routing. Two heuristics are based on well known heuristics for a layout with two cross aisles: S-shape and largest gap. Furthermore, the routing strategy of Vaughan and Petersen (1999) is described briefly. The fourth routing method, consists of finding a shortest route.

## 2.1 S-shape

Basically, any subaisle containing at least one pick location is traversed through the entire length. Subaisles where nothing has to be picked are skipped. In the following more elaborate description of the heuristic, letters between brackets correspond to the letters in the example route depicted in figure 2a.

1. Determine the left most pick aisle that contains at least one pick location (called *left pick aisle*) and determine the block farthest from the depot that contains at least one pick location (called *farthest block*).

2. The route starts by going from the depot to the front of the left pick aisle **(a)**.

3. Traverse the left pick aisle up to the front cross aisle of the farthest block **(b)**.

4. Go to the right through the front cross aisle of the farthest block until a subaisle with a pick is reached **(c)**. If this is the only subaisle in this block with pick locations then pick all items and return to the front cross aisle of this block. If there are two or more subaisles with picks in this block, then entirely traverse the subaisle **(d)**.

5. At this point, the order picker is in the back cross aisle of a block, call this block the *current block*. There are two possibilities.
   (1) There are picks remaining in the current block (not picked in any previous step). Determine the distance from the current position to the left most subaisle and the right most subaisle of this block with picks. Go to the closer of these two **(e)**. Entirely traverse this subaisle **(f)** and continue with step 6.
   (2) There are no items left in the current block that have to be picked. In this case, continue in the same pick aisle (i.e. the last pick aisle that was visited in either step 7 or in this step) to get to the next cross aisle and continue with step 8.

6. If there are items left in the current block that have to be picked, then traverse the cross aisle towards the next subaisle with a pick location **(g)** and entirely traverse that subaisle **(h)**. Repeat this step until there is exactly one subaisle left with pick locations in the current block.

7. Go to the last subaisle with pick locations of the current block **(i)**. Retrieve the items from the last subaisle and go to the front cross aisle of the current block **(j)**. This step can actually result in two different ways of traveling through the subaisle (1) entirely traversing the subaisle or (2) enter and leave the subaisle from the same side.

8. If the block closest to the depot has not yet been examined, then return to step 5.

9. Finally, return to the depot **(k)**.

### 2.2 Largest gap heuristic

The largest gap heuristic basically follows the perimeter of each block entering subaisles when needed. The heuristic first goes to the farthest block and then proceeds block by block to the front of the warehouse. A route resulting from this heuristic is depicted in figure 2b. Letters between brackets correspond to the letters in figure 2b. In this description we say that each subaisle is entered as far as the 'largest gap' Here we mean with a *gap* the distance between any two adjacent pick locations within a subaisle, or between a cross aisle and the nearest pick location. The *largest gap* is the largest of all gaps in a subaisle. The largest gap divides the pick locations in a subaisle into two sets. One set of pick locations is accessed from the back cross aisle; the other set from the front cross aisle. Note that one or both of the sets may be empty, making it unnecessary to enter the subaisle from that side.

1. Determine the left most pick aisle that contains at least one pick location (called *left pick aisle*) and determine the block farthest from the depot that contains at least one pick location (called *farthest block*).

2. The route starts by going from the depot to the front of the left pick aisle **(a)**.

3. Traverse the left pick aisle up to the front cross aisle of the farthest block **(b)**.

4. Go to the right through the front cross aisle of the farthest block until a subaisle with a pick is reached **(c)**. If this is the only subaisle in this block with pick locations then pick all items and return to the front cross aisle of this block. If there are two or more subaisles with picks in this block, then entirely traverse the subaisle **(d)**.

5. At this point, the order picker is in the back cross aisle of a block, call this block the *current block*. There are two possibilities.
   (1) There are picks remaining in the current block (not picked in any previous step). Determine the subaisle of the current block with pick locations that is farthest from the current position. Call this subaisle the *last subaisle of the current block*. Continue with step 6.
   (2) (2) There are no items left in the current block that have to be picked. Continue in the same pick aisle (i.e. the last pick aisle that was visited in either step 7, step 8 or in this step) to get to the next cross aisle and continue with step 9.

6. Follow the shortest path through the back cross aisle starting at the current position, visiting all subaisles that have to be entered from the back **(e)** and ending at the last

subaisle of the current block **(f)**. Each subaisle that is passed has to be entered up to the largest gap. Note that this step may require the order picker to walk part of the cross aisle both from left to right and from right to left (see the example of figure 2b)

7. Entirely traverse the last subaisle of the current block to get to the front cross aisle **(g)**.
8. Start at the last subaisle of the current block and move past all subaisles of the current block that have picks left. Enter these subaisles up to the largest gap to pick the items **(h)**.
9. If the block closest to the depot has not yet been examined, then return to step 5.
10. Finally, return to the depot **(k)**.

### 2.3 Aisle-by-aisle

This heuristic for warehouses with multiple cross aisles is presented in Vaughan and Petersen (1999). Order picking routes resulting from this heuristic visit every pick aisle exactly once. That is, first all items in pick aisle 1 are picked, then all items in pick aisle 2, and so on. Dynamic programming is used to determine the best cross aisles to go from pick aisle to pick aisle.

The order picking route starts at the depot. For every cross aisle $i$ the distance is calculated, that is needed to start at the depot, pick all items in pick aisle 1 and exit the pick aisle via cross aisle $i$. If there are $m$ cross aisles, then this results in $m$ distances each with a corresponding partial order picking route. Now for each cross aisle $j$, we determine cross aisle $i$ such that the distance to start at the depot, pick all items in pick aisle 1, pick all items in pick aisle 2 and exit pick aisle 2 at cross aisle $j$, is shortest if we go from pick aisle 1 to pick aisle 2 via cross aisle $i$. This gives us again $m$ distances and partial order picking routes. Continuing in a similar fashion, we determine for each cross aisle $j$ exiting pick aisle 3 the best cross aisle to go from pick aisle 2 to pick aisle 3. This process is repeated until all pick aisles have been considered. Then the order picker returns to the depot. An example route is given in figure 2c.

Note that the algorithm originally assumed that the order picker starts at the head of the left most pick aisle of the warehouse and ends at the right most pick aisle of the warehouse. For reasons of compatibility with the other routing methods, a minor change in the heuristic was made such that routes start and end at the depot.

### 2.4 Optimal algorithm

The routing of order pickers in a warehouse is a special case of the Travelling Salesman Problem. A number of locations have to be visited with the objective to travel as little as possible. For the Traveling Salesman Problem there is no polynomial-time algorithm known that can find shortest routes. For warehouses with two cross aisles however, an efficient routing algorithm was developed by Ratliff and Rosenthal (1983). Their method uses dynamic programming to solve the problem. Extensions of the algorithm to more cross aisles are non-trivial and the number of equivalence classes and possible transitions increase rapidly. Therefore, we determine shortest order picking routes in this paper with a branch-and-bound procedure for the Travelling Salesman Problem (see e.g. Little *et al.* 1963). In figure 2d an example route is depicted. The results from the branch-and-bound procedure will be used as a benchmark in the performance analysis of the heuristics described in this paper.

[Insert figure 2 about here]

# 3 Combined heuristic

For practice, it is generally preferred to have a routing method that generates routes that have a clear and easy to understand structure. Routes having a clear pattern reduce the time spent by order pickers on searching for locations and reduces the risk of pick errors. The combined routing method generates such routes. Every subaisle, that contains items, is visited exactly once. The route starts and ends at the depot. The order picker goes through the left most pick aisle that contains items towards the block farthest from the depot that contains items. The subaisles of the farthest block are visited sequentially from left to right. Then the order picker goes to the next block (one block closer to the depot). The items in this block are picked. This process is repeated until all blocks with items have been visited. See figure 6a for an example route. The subaisles are either entirely traversed or the order picker enters and leaves the subaisle from the same side. These choices are made with a dynamic programming method which will be explained in section 3.2. The construction of a complete route is discussed in section 3.3.

## 3.1 Definitions

We define the following variables:
$k$ the number of blocks
$n$ the number of pick aisles
We denote some physical locations in the warehouse as follows:
$a_{ij}$ the back end of subaisle $j$ in block $i$, for each block $i, i = 1, ..., k$ and for each subaisle $j, j = 1, ..., n$
$b_{ij}$ the front end of subaisle $j$ in block $i$, for each block $i, i = 1, ..., k$ and for each subaisle $j, j = 1, ..., n$
$d$ the depot

Note that for $i = 1, 2, ..., k-1$ it holds that $b_{ij} = a_{i+1,j}$. This holds because we assume that the order pickers walk through the middle of the cross aisles. The distance from the end of a subaisle to the centre of a cross aisle is administrated as if it belongs to the subaisle.

For each block we will use a dynamic programming method. We will first describe this dynamic programming method for a single block in section 3.2. In section 3.3 we describe how the heuristic creates routes, using the dynamic programming method for each single block.

## 3.2 Dynamic programming method

This section describes a dynamic programming method to route an order picker through a single block $i$ ($i = 1, ..., k$). The route starts at the left most subaisle that contains items ($\ell$) and ends at the right most subaisle that contains items ($r$). We define $L_j$ to be a partial route visiting all pick locations in subaisles $\ell$ through $j$ and we distinguish two classes of partial routes:

$L_j^a$ which is a partial route that ends at the back of subaisle $j$
$L_j^b$ which is a partial route that ends at the front of subaisle $j$

We distinguish two ways to go from subaisle $j-1$ to subaisle $j$ (see figure 3):

$t_a$ which goes along the back of the block
$t_b$ which goes along the front of the block

Furthermore, we distinguish four ways to pick all items in subaisle $j$. These four transitions are (see figure 3):

$t_1$ entirely traverse the subaisle
$t_2$ do not enter this subaisle at all
$t_3$ enter and leave the subaisle from the front of the block
$t_4$ enter and leave the subaisle from the back of the block

Clearly, transition $t_2$ is only allowed if the subaisle does not contain any items.

With $L_j+t_w$ we denote that partial route $L_j$ is extended with transition $t_w$ ($w = 1, 2, 3, 4, a, b$). The function $c(.)$ gives the travel time associated with its argument, e.g. $c(L_j^b + t_1)$ gives the time needed to walk the partial route $L_j^b$ plus the time needed to walk transition $t_1$. Note that the transitions only contain information on how to enter and leave the subaisles. The exact path within the subaisle – and therefore the travel time associated with a transition – is dependent on the item locations within the subaisle under consideration.

[Insert figure 3 about here]

Using the potential states, the possible transitions between the states and the costs (travel time) involved in such transition, we now give the dynamic programming method. This method will determine a partial route, going through one block. The construction of the full order picking path, connecting the partial routes for the individual blocks, will be described in section 3.3.

**Step 1**

The block under consideration is block $i$.
If block $i$ is the block farthest from the depot, that contains items, then start with the two partial routes:

$L_\ell^a$ which starts at node $b_{i\ell}$, ends at node $a_{i\ell}$ and consists of transition $t_1$
$L_\ell^b$ which starts and ends at node $b_{i\ell}$ and consists of transition $t_3$

Otherwise, start with two partial routes:

$L_\ell^a$ which starts and ends at node $a_{i\ell}$ and consists of transition $t_4$
$L_\ell^b$ which starts at node $a_{i\ell}$, ends at node $b_{i\ell}$ and consists of transition $t_1$

**Step 2**

For each consecutive subaisle $j$ $(\ell < j < r)$ we determine $L_j^a$ and $L_j^b$ as follows.
If subaisle $j$ contains items then:

$$L_j^a = \begin{cases} L_{j-1}^a + t_a + t_4 & \text{if } c(L_{j-1}^a + t_a + t_4) < c(L_{j-1}^b + t_b + t_1) \\ L_{j-1}^b + t_b + t_1 & \text{otherwise} \end{cases}$$

$$L_j^b = \begin{cases} L_{j-1}^b + t_b + t_3 & \text{if } c(L_{j-1}^b + t_b + t_3) < c(L_{j-1}^a + t_a + t_1) \\ L_{j-1}^a + t_a + t_1 & \text{otherwise} \end{cases}$$

If subaisle $j$ does not contain items then:

$$\begin{aligned} L_j^a &= L_{j-1}^a + t_a \\ L_j^b &= L_{j-1}^b + t_b \end{aligned}$$

**Step 3**

For the last subaisle of the block (subaisle $r$), we determine

$$L_r^b = \begin{cases} L_{r-1}^b + t_b + t_3 & \text{if } c(L_{r-1}^b + t_b + t_3) < c(L_{r-1}^a + t_a + t_1) \\ L_{r-1}^a + t_a + t_1 & \text{otherwise} \end{cases}$$

The resulting partial route $L_r^b$ will be used to form the complete order picking route. In this step we do not need $L_r^a$ anymore. This is because once all items have been picked in a block, then we need to go to the front of the block to be able to continue to the next block (see section 3.3).

### 3.2.1 Example

Consider one block with 3 subaisles for which we will apply the dynamic programming algorithm. Figure 4a gives the block with pick locations for this example. This block is assumed to be the block farthest from the depot that contains items. In total 7 items have to be picked from this block. The length of a subaisle is 8 metres (1 metre for each section plus 0.5 metre on both sides to go to the centre of the cross aisle). The distance between two neighbouring subaisles is 4 metres. Travel speed is 1 metre per second. Figure 4b depicts the situation of figure 4a with nodes for the pick locations and the heads of the subaisles and with edges for the possible travel paths. Figure 5 visualises the steps of the dynamic programming algorithm. Note that in this example $\ell = 1$ and $r = 3$. All travel times in this example are expressed in seconds.

Step 1 Since the block under consideration (block $i$) is assumed to be the block farthest from the depot, we start with two partial routes $L_1^a$ and $L_1^b$. $L_1^a$ starts at node $b_{i1}$, ends at node $a_{i1}$ and consists of transition $t_1$, with associated travel time $c(t_1) = 8$. $L_1^b$ starts and ends at node $b_{i1}$ and consists of transition $t_3$, with associated travel time $c(t_3) = 14$.

Step 2 We have two possibilities for creating $L_2^a$, namely as $L_1^a + t_a + t_4$ (travel time $8+4+4 = 16$) or as $L_1^b + t_b + t_1$ (travel time $14 + 4 + 8 = 26$). We choose the shortest of the two. Thus $L_2^a = L_1^a + t_a + t_4$.

Similarly, we have two possibilities for creating $L_2^b$, namely as $L_1^b + t_b + t_3$ (travel time $14 + 4 + 12 = 30$) or as $L_1^a + t_a + t_1$ (travel time $8 + 4 + 8 = 20$). We choose the shortest of the two. Thus $L_2^b = L_1^a + t_a + t_1$.

Step 3 We have two possibilities to create $L_3^b$. Clearly, $L_2^a + t_a + t_1$ (travel time $16 + 4 + 8 = 28$) is faster than $L_2^b + t_b + t_3$ (travel time $20 + 4 + 10 = 34$). Therefore, $L_3^b = L_2^a + t_a + t_1$.

This completes the partial route created by the dynamic programming algorithm. The creation of a full order picking route, going through multiple blocks, will be discussed in the next section.

[Insert figure 4 about here]

[Insert figure 5 about here]

### 3.3 Route construction

So far we have described a dynamic programming method for routing in a single block. In this section we give a description of the combined routing heuristic, which uses this dynamic programming method for the individual blocks. The route starts and ends at the depot. First, the order picker goes to the block farthest from the depot that contains items via the left most pick aisle that contains items. The items in the left most pick aisle are picked and next the items in the farthest block are picked. Then the order picker moves one block towards the depot and picks all items from that block. This process is repeated until all blocks with items have been visited.

An example route is depicted in figure 6a. Each number in this figure corresponds to one of the numbers of the steps in the description below. Note that step 6 requires applying the dynamic programming algorithm from section 3.2. It can easily be seen that we can use the results from the example in section 3.2.1 for the farthest block in the example of figure 6a. The stepwise procedure for the combined heuristic is given below.

1. Determine the left most pick aisle that contains at least one pick location (called *left pick aisle*) and determine the block farthest from the depot that contains at least one pick location (called *farthest block*).

2. The route starts by going from the depot to the front of the left pick aisle.

3. Traverse the left pick aisle up to the front cross aisle of the farthest block (block $i_{\min}$).

4. Set $i = i_{\min}$.

5. Determine whether or not block $i$ contains items that have not been picked in step 3.

   If no items have to be picked in block $i$: traverse the nearest subaisle of block $i$ to reach the next block. Continue with step 7.

   If items have to be picked in block $i$: determine the left most subaisle and the right most subaisle that contains items (respectively subaisle $\ell$ and subaisle $r$), excluding any subaisle that was already visited in step 3. Go from the current position to the nearest of these two ($j_{\min}$).

6. Apply the dynamic programming method of section 3.2 to block $i$. If in step 5 $j_{\min} = \ell$, then add the partial route resulting from the dynamic programming algorithm to the order picking path. If $j_{\min} = r$, then reverse the partial route resulting from the dynamic programming algorithm (the route then starts in subaisle $r$ and ends in subaisle $\ell$). Add this reversed partial route to the order picking path. Reversing the path means that the order picker will visit the subaisles from right to left. The calculations were performed from left to right.

7. When block $k$ (the block closest to the depot) has been evaluated, the order picker returns to the depot. Otherwise increase $i$ by 1 and return to step 5.

[Insert figure 6 about here]

### 3.4 Improvements

First of all, consider the routing in the block closest to the depot. The starting point for routing in this block is determined by the position where the order picker ends his route through the previous block. This could lead to a route in which aisles are visited from the left to the right. This implies that the order picker ends his route somewhere at the right of the front cross aisle. After this, a considerable part of the front cross aisle has to be traversed before reaching the depot. This can be prevented by forcing the route to visit aisles from the right to the left in the block closest to the depot. It can easily be seen that this change in the heuristic will either decrease travel time or leave the travel time unaltered.

Secondly, consider the path of the order picker from the depot to the farthest block. This path goes through the left most pick aisle with pick locations. However one can think of situations where it can be advantageous to deviate from this path. In the example of figure 6a, travel time can be decreased by going through the second pick aisle towards the back of the warehouse instead of through the first pick aisle. Stated more general, we could create routes such that the order picker picks items from the left $x$ pick aisles on his way to the farthest block and picks items from the right $n - x$ pick aisles when returning to the front. The dynamic programming method is applied to the left $x$ subaisles of each block on the way to the back and to the right $n-x$ subaisles of each block on the way to the front. By optimising over $x$ we obtain a route that is guaranteed to be shorter or at most as long as the route generated by the original combined heuristic.

We adapt the combined heuristic to incorporate both improvements suggested in this section and call the result the *combined*$^+$ heuristic. An example route is given in figure 6b. Note that the two improvements could also be added to the largest gap and S-shape heuristic. However, the main advantage of these two heuristics is meant to be their ease of use, which would diminish by such substantial alterations. Therefore, we do not alter the other heuristics.

## 4 A comparison of heuristics

This section compares the optimal and heuristic solutions in a practical order picking system, namely a shelf area. We consider a shelf area where order pickers walk through the warehouse to pick small items, using a small pick cart. The following assumptions are made. The average walking speed in both cross aisles and pick aisles is 0.6 metres per second. The centre-to-centre distance between two neighbouring pick aisles is 2.5 metres and no additional time is needed for aisle changing. Cross aisle width is 2.5 metres. Picking of items can be performed simultaneously from both sides of a pick aisle since the aisles are fairly narrow. Order pickers are assumed to walk through the middle of the pick aisles and cross aisles.

For this type of warehouse we assume the following measures to be representative. Pick aisle length varies between 10 and 30 metres. Each order picker works in a zone consisting of 7 to 15 pick aisles. Each picking route has to visit between 10 and 30 locations. These values are based on observations of numerous actual manual shelf warehouse operations by the authors. We use the extremes of these values for our simulation experiments, which gives eight different configurations. For each configuration, we generate a number of random orders. The locations of the items in an order are uniformly and independently distributed over the order picking area. That is we assume that products are stored randomly in the storage area. No positioning according to demand frequencies is used. See for example Caron *et al.* (1998) for issues involving non-random storage.

For each random order, the route length in a warehouse with two cross aisles is calculated for

the S-shape, largest gap, aisle-by-aisle, combined and combined$^+$ heuristics and for the optimal algorithm. Then, an additional cross aisle is added and route length for each of the routing methods is calculated. Another cross aisle is added, route length is calculated, and so on. Averages are taken over the instances with the same number of cross aisles.

We can distinguish between two ways to increase the number of cross aisles. Firstly, we can fix the number of storage locations. In this approach the length of the warehouse will increase if cross aisles are added. Secondly, we could fix the size of the warehouse and add cross aisles by losing storage locations. Since usually in design the amount of storage space is decided in advance, we choose the first approach. This is consistent with Vaughan and Petersen (1999). The minimum number of cross aisles is two. Such a warehouse with two cross aisles has one cross aisle in the front and one cross aisle in the back. This restriction is needed since all routing methods assume that the order picker is able to enter and leave the subaisles both from the front and from the back. Additional cross aisles are inserted such that the centre distance between any two adjacent cross aisles is equal. Other ways of cross aisle distribution may be interesting especially in situations where a storage method other than random storage is used. The maximum number of cross aisles is 11 for the experiments. Since the shortest pick aisles in the experiments are 10 metres, having 11 cross aisles means that the shortest subaisle encountered in the experiments is 1 metre. One metre can be considered to be the minimum rack length that is practically feasible in a shelf warehouse.

For each simulation experiment, the necessary number of replications needs to be determined such that the estimate for the mean travel time has a relative error smaller than some $\gamma$, for $0 < \gamma < 1$. An approximation for the necessary number of replications, such that the relative error is smaller than $\gamma$ with a probability of $1 - \alpha$, is given in Law and Kelton (1991). For all situations considered in this paper, a replication size of 2000 orders has appeared to be sufficient to guarantee a relative error of at most 1% with a probability of 95%.

Table 1 gives the average travel time for each combination of the 8 instances, 10 cross aisle configurations and 6 routing methods. For each configuration, the best heuristic is indicated by making the corresponding cell of the table grey. From the table we can see that the S-shape heuristic never had the best performance of the five heuristics. Largest gap had the best performance in 5 situations each of which has a layout with two cross aisles. Aisle-by-aisle had the best performance in 4 situations, of which 3 equal the travel time of the combined and combined$^+$ heuristics. The combined$^+$ heuristic gave the best results in 74 of the 80 instances, of which 3 equal the travel time of the aisle-by-aisle and combined heuristics. For each of the heuristics the average calculation time for a single route was less than 0.1 seconds on a 350Mhz computer.

[Insert table 1 about here]

If we compare the combined and the S-shape heuristic on theoretical grounds, we can state that for each individual order combined will give a route that is equal to or shorter than the S-shape route. This is because S-shape entirely traverses every subaisle containing items, whereas the combined heuristic chooses between entirely traversing the subaisle or returning to the same side of the subaisle, depending on which gives the shortest travel time. Thus, the combined heuristic is capable of generating routes that are exactly the same as those of S-shape, however – if possible – it will give shorter order picking routes by returning within subaisles. The difference in performance between S-shape and combined depends on the situation under consideration. For the situations analysed in this paper it holds that average travel time for S-shape is at least 7% higher than for combined in warehouses with three cross aisles. The difference between

S-shape and combined tends to zero when increasing the number of cross aisles. The percentage difference between combined and S-shape is smaller if the pick density is high (30 items instead of 10 items). This is due to the fact that optimal routes generally tend to entirely traverse more aisles if pick density is higher. Therefore S-shape routes are closer to optimal and the room for improvement is smaller for the combined heuristic.

A second interesting property of the heuristics is the fact that routes of the aisle-by-aisle, combined and combined$^+$ heuristics are identical for warehouses with two cross aisles. This is due to the fact that the heuristics use the same system of dynamic programming. Aisle-by-aisle creates routes through all blocks and uses one equivalence class for each cross aisle. Combined and combined$^+$ apply dynamic programming to each block individually and use two equivalence classes, for each cross aisle one. If the warehouse has two cross aisles (i.e. consists of one block) then both heuristics use the same two equivalence classes and consequently give the same routes.

Now let us look at the effect of the improvements for the combined heuristic suggested in section 3.4. Combined and combined$^+$ are the same for warehouses with two cross aisles, because the changes only apply to blocks between the depot and the farthest block. For situations with three cross aisles, the difference between combined and combined$^+$ is minor. The improvements are however substantial for situations with more than three cross aisles. Average travel time of combined was in these cases up to 24.6% over that of combined$^+$.

In the situations with three or more cross aisles, combined$^+$ had the best performance of the heuristics for all situations except one. For picklists of 10 items and with three or more cross aisles, the difference between combined$^+$ and optimal is less than 7.5%. For the situations considered in this paper, the size of the gap between combined$^+$ and the optimal algorithm varies between 1% and 25% (see table 2). The largest differences occurred for picklists of 30 items. The situations with 7 pick aisles gave a smaller difference between optimal and combined$^+$ than the situations with 15 pick aisles. Generally we can say that the gap between optimal and combined$^+$ tends to be larger if the situation is more complex, i.e. more aisles and/or more items.

[Insert table 2 about here]

The observed gap between the best heuristic and the optimal algorithm gives rise to two different approaches for further research: (1) develop better heuristics or (2) use an optimal algorithm for routing order pickers. Both approaches have their advantages and disadvantages. The heuristics are fairly simple in structure and therefore easy to implement in a practical situation. For situations with a large number of cross aisles, it may be worthwhile to analyse the performance of standard routing heuristics for the traveling salesman problem. On the other hand optimal routing gives significantly shorter routes. However, the logic may be unclear to the order picker which may cause him to accidentally walk the wrong way or to override the system and walk the way he thinks is best. Furthermore, in this paper we used a branch-and-bound method to calculate the shortest routes. Such a method has unpredictable computation times, which is an undesirable property for practical implementations.

From table 1, we can also see the effect of adding cross aisles to the layout. In general we can say that travel time decreases if we change the layout from two to three cross aisles. Two exceptions occur: (1) with largest gap for warehouses with short pick aisles and (2) for a small warehouse with many picks. Both exceptions seem intuitively clear. For the first exception consider that if a warehouse has short pick aisles then the distance travelled in the cross aisles accounts for a relatively large amount of the travel time. This is even more the case when using largest gap in warehouses with three or more cross aisles, since routes resulting from this

heuristic often traverse cross aisles (except those at the back and front) twice. Consequently, a large increase in travel time occurs if a third cross aisle is added to the layout. For the second exception consider a warehouse with a few short pick aisles and many pick locations. The main advantage of adding cross aisles is that more possibilities arise to route the order picker. However, if the order picker has to visit many locations, then entirely traversing pick aisles is close to optimal. Any extra cross aisle only increases the warehouse size and therefore travel times.

More cross aisles may or may not improve the average time needed to pick an order. For most situations it holds that adding cross aisles decreases average travel time up to a certain point after which average travel time starts increasing again. The optimal number of cross aisles with respect to travel time seems to depend on the number of pick aisles, the aisle length and the number of items. Especially aisle length seems to be important in the sense that longer aisles most often require more cross aisles. Important for warehouse design is that more cross aisles implies higher space requirements. Therefore, the cost reductions from adding cross aisles have to be weighed against increased costs for the building.

# 5 Concluding remarks

Performances of heuristics in warehouses with two cross aisles have been studied extensively. In this paper, we have introduced several methods for routing order pickers in a warehouse with multiple cross aisles. Two methods, the S-shape and largest gap heuristics, are straight forward extensions of existing methods for warehouses with two cross aisles. The aisle-by-aisle heuristic was introduced by Vaughan and Petersen (1999). The combined and combined$^+$ heuristics are introduced in this paper.

For the majority of the situations (74 of 80) evaluated in this paper, the combined$^+$ heuristic had the best performance of the heuristics. Largest gap was found to be useful in situations with two cross aisles and low pick densities, which is consistent with Hall (1993).

The performance of the heuristics was also compared to the results of a branch-and-bound procedure that generates shortest order picking routes. It has to be noted that the gap between this optimal routing method and the best heuristic varies substantially. Implementation of the optimal procedure in practical situations may however give rise to problems such as unpredictable computation times. It could therefore be desirable to improve heuristic performance or find more efficient methods to calculate shortest routes.

In this paper we considered only situations where products are stored randomly. Other storage assignment rules may cause a different ranking among the heuristics. Furthermore, the positioning of the cross aisles may be an interesting aspect to consider when using non-random storage, since there will be much activity in the area with the frequently requested products.

Generally, average travel times decreases when changing the layout from two to three cross aisles. Two exceptions are the situation of a small warehouse with many picks for all routing methods and the situations with short pick aisles for largest gap. More cross aisles may or may not decrease travel times, depending on the routing method and the situation under consideration.

# 6 References

Caron, F., Marchet, G., and Perego, A., 1998, Routing policies and COI-based storage policies in picker-to-part systems. *International Journal of Production Research,* **36(3)**, 713-732.

De Koster, R., and Van der Poort, E., 1998, Routing orderpickers in a warehouse: a comparison between optimal and heuristic solutions. *IIE Transactions,* **30**, 469-480.
De Koster, R., Van der Poort, E., and Wolters, M., 1999, Efficient orderbatching methods in warehouses. *International Journal of Production Research,* **37(7)**, 1479-1504.
Gibson, D. R., and Sharp, G. P., 1992, Order batching procedures. *European Journal of Operations Research,* **58(1)**, 57-67.
Hall, R. W. H., 1993, Distance approximations for routing manual pickers in a warehouse. *IIE Transactions,* **25(4)**, 76-87.
Goetschalckx, M., and Ratliff, H. D., 1988, Order picking in an aisle. *IIE Transactions,* **20(1)**, 53-62.
Law, A. M., and Kelton, W. D., 1991, *Simulation Modeling & Analysis*, 2nd ed. (New York: McGraw-Hill, Inc.), chapter 9, pp. 522-581.
Little, J. D. C., Murty, K. G., Sweeney, D. W., and Karel, C., 1963, An algorithm for the traveling salesman problem. *Operations Research,* **11**, 972-989.
Petersen, C. G., 1997, An evaluation of order picking routeing policies. *International Journal of Operations & Production Management,* **17(11)**, 1098-1111.
Petersen, C.G., and Schmenner, R.W., 1999, An evaluation of routing and volume-based storage policies in an order picking operation. *Decision Sciences,* **30(2)**, 481-501.
Ratliff, H. D., and Rosenthal, A. S., 1983, Orderpicking in a rectangular warehouse: A solvable case of the traveling salesman problem. *Operations Research,* **31**, 507-521.
Roodbergen, K. J., and De Koster, R., 1998, Routing orderpickers in a warehouse with multiple cross aisles. In R. J. Graves, L. F. McGinnis, D. J. Medeiros, R. E. Ward and M. R. Wilhelm (eds.), *Progress in Material Handling Research: 1998* (Charlotte, NC: Material Handling Institute), pp. 451-467.
Ruben, R. A., and Jacobs, F. R., 1999, Batch construction heuristics and storage assignment strategies for walk/ride and pick systems. *Management Science,* **45(4)**, 575-596.
Tompkins, J. A., White, J. A., Bozer, Y. A., Frazelle, E. H., Tanchoco, J. M. A., and Trevino, J., 1996, *Facilities Planning*, 2nd ed. (New York: John Wiley & Sons, Inc.).
Vaughan, T. S., and Petersen, C. G., 1999, The effect of warehouse cross aisles on order picking efficiency. *International Journal of Production Research,* **37(4),** 881-897.

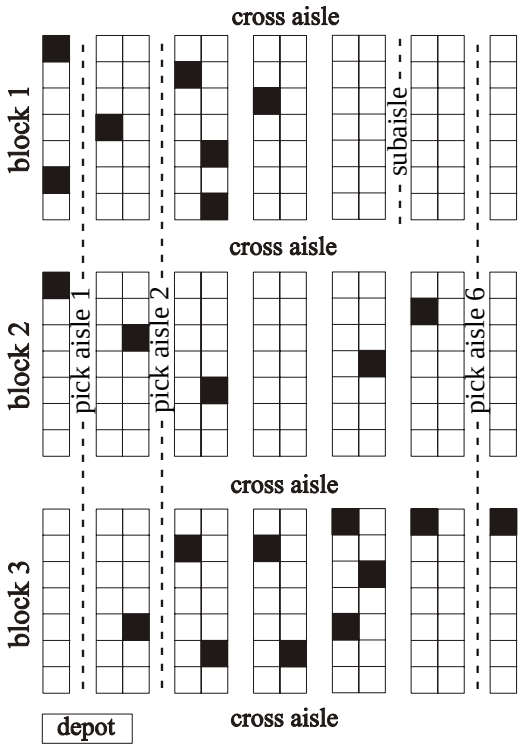



Figure 1 Example of a warehouse layout with multiple cross aisles.

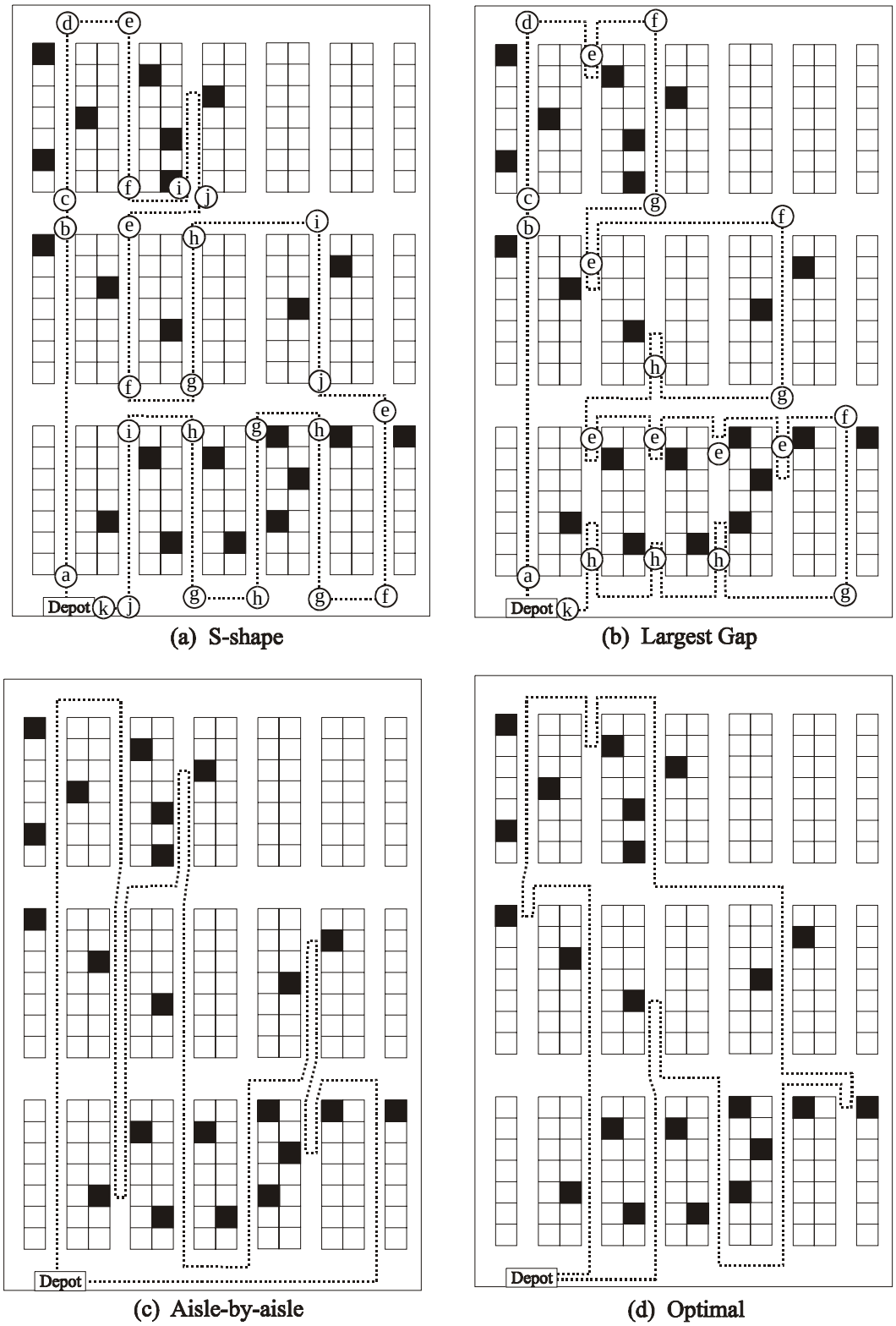



Figure 2 Example routes for four routing methods.

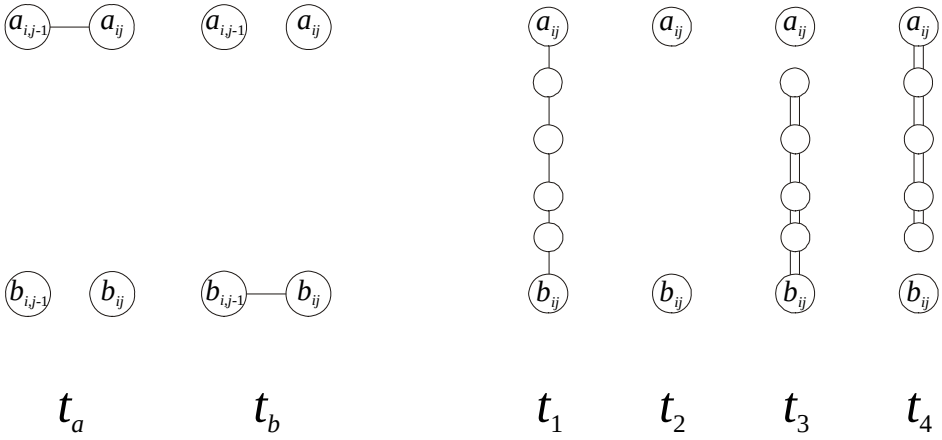



Figure 3 Transitions used by the combined routing heuristic.

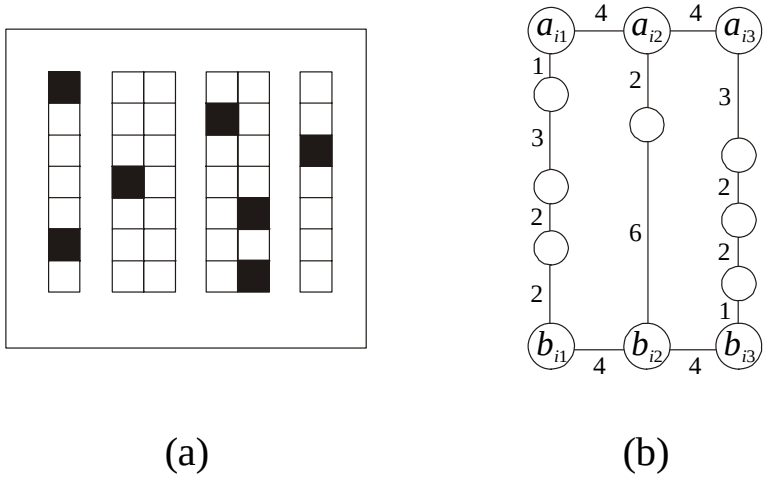



Figure 4 Example situation for the dynamic programming method.

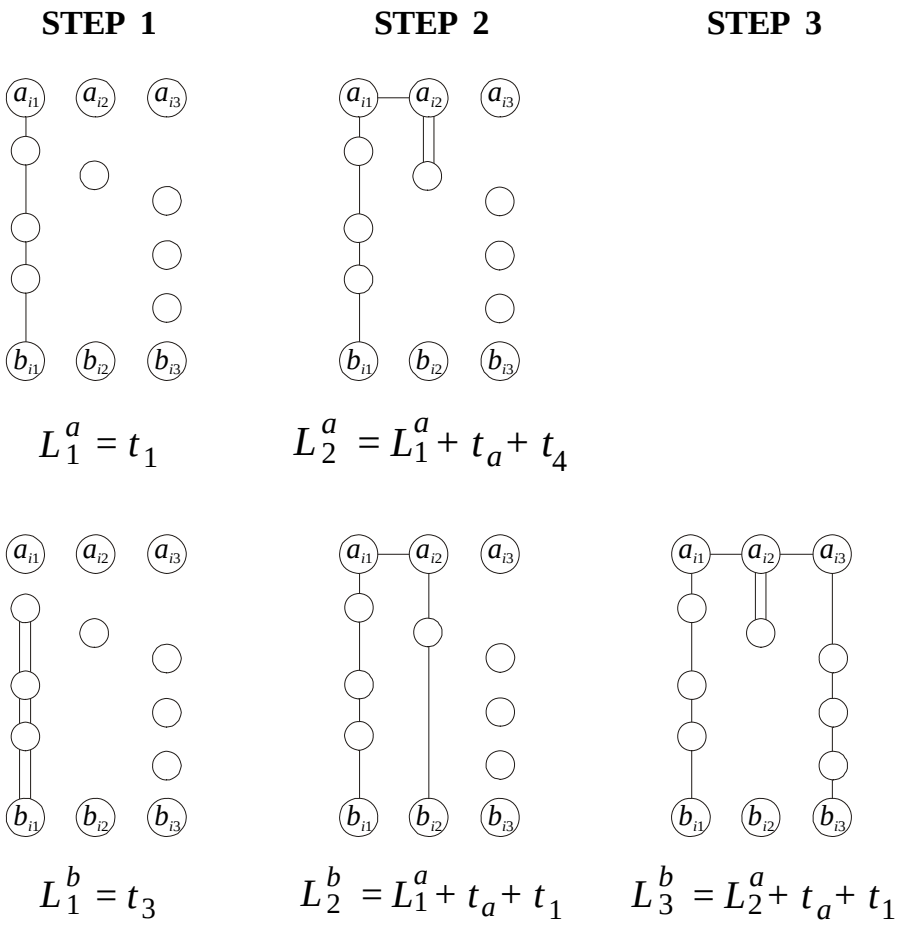
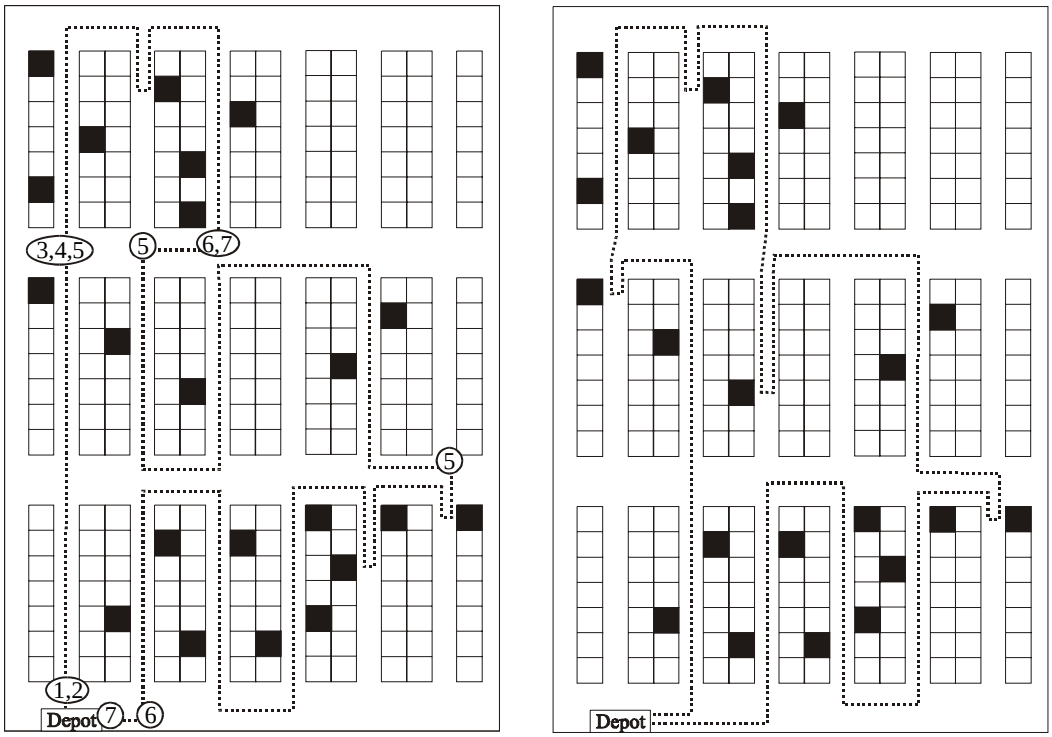



Figure 5 Visualisation of the steps taken by the dynamic programming method.




(a) Combined (b) Combined+

Figure 6 Route resulting from applying the combined routing method.

| optimal | | | number of cross aisles | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| # aisles | length | # items | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 |
| 7 | 10 | 10 | 138.7 | 129.7 | 131.5 | 135.7 | 141.7 | 148.0 | 155.5 | 162.0 | 169.6 | 177.4 |
| 7 | 10 | 30 | 186.6 | 191.4 | 198.6 | 207.1 | 216.3 | 224.9 | 235.0 | 243.2 | 252.8 | 261.6 |
| 15 | 10 | 10 | 219.6 | 202.0 | 201.4 | 205.2 | 211.4 | 218.2 | 226.7 | 233.8 | 242.2 | 251.0 |
| 15 | 10 | 30 | 337.5 | 314.3 | 311.6 | 315.7 | 324.5 | 333.4 | 346.1 | 355.6 | 368.7 | 381.4 |
| 7 | 30 | 10 | 269.6 | 222.9 | 211.1 | 209.0 | 211.4 | 215.8 | 221.3 | 227.4 | 233.9 | 240.2 |
| 7 | 30 | 30 | 398.3 | 361.1 | 342.9 | 336.5 | 333.8 | 334.2 | 337.0 | 340.8 | 345.0 | 349.7 |
| 15 | 30 | 10 | 377.3 | 308.0 | 290.9 | 287.7 | 289.3 | 293.5 | 299.2 | 305.4 | 312.0 | 318.5 |
| 15 | 30 | 30 | 665.5 | 540.6 | 495.9 | 479.8 | 473.3 | 472.8 | 475.9 | 480.7 | 486.0 | 491.7 |

| Largest gap | | | number of cross aisles | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| # aisles | length | # items | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 |
| 7 | 10 | 10 | 146.6 | 156.9 | 164.6 | 169.3 | 176.3 | 181.9 | 187.4 | 194.7 | 201.9 | 209.3 |
| 7 | 10 | 30 | 208.6 | 240.9 | 273.9 | 303.5 | 330.3 | 348.6 | 363.4 | 379.6 | 394.4 | 408.4 |
| 15 | 10 | 10 | 227.3 | 265.2 | 287.2 | 296.2 | 305.7 | 312.2 | 317.1 | 324.6 | 331.4 | 338.4 |
| 15 | 10 | 30 | 357.5 | 413.5 | 484.5 | 552.1 | 614.6 | 660.1 | 692.3 | 724.7 | 750.1 | 776.3 |
| 7 | 30 | 10 | 295.1 | 259.9 | 250.7 | 246.3 | 247.4 | 250.3 | 254.5 | 259.7 | 264.9 | 270.6 |
| 7 | 30 | 30 | 451.7 | 424.7 | 425.7 | 435.9 | 446.9 | 456.1 | 464.1 | 472.2 | 478.4 | 488.7 |
| 15 | 30 | 10 | 401.0 | 377.6 | 379.1 | 377.2 | 379.5 | 382.4 | 386.6 | 390.6 | 395.2 | 400.6 |
| 15 | 30 | 30 | 715.6 | 646.0 | 665.4 | 705.2 | 746.3 | 779.9 | 805.0 | 826.5 | 842.8 | 863.7 |

| S-shape | | | number of cross aisles | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| # aisles | length | # items | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 |
| 7 | 10 | 10 | 165.1 | 145.7 | 152.6 | 155.7 | 161.4 | 167.7 | 174.6 | 181.8 | 188.6 | 196.4 |
| 7 | 10 | 30 | 203.5 | 210.3 | 250.1 | 253.4 | 278.3 | 287.8 | 301.0 | 311.9 | 322.3 | 332.4 |
| 15 | 10 | 10 | 266.2 | 224.6 | 245.0 | 252.6 | 261.2 | 270.2 | 278.9 | 288.3 | 295.5 | 304.6 |
| 15 | 10 | 30 | 391.3 | 359.5 | 431.1 | 422.8 | 478.3 | 491.1 | 518.4 | 539.0 | 557.8 | 576.0 |
| 7 | 30 | 10 | 353.1 | 276.2 | 256.5 | 245.0 | 242.5 | 243.4 | 247.1 | 251.7 | 257.1 | 262.5 |
| 7 | 30 | 30 | 452.0 | 426.8 | 438.2 | 420.1 | 427.7 | 423.4 | 427.0 | 429.7 | 432.9 | 437.1 |
| 15 | 30 | 10 | 517.6 | 376.9 | 361.0 | 349.4 | 347.6 | 350.1 | 354.7 | 360.4 | 366.4 | 372.7 |
| 15 | 30 | 30 | 833.3 | 686.0 | 688.2 | 636.4 | 663.4 | 653.6 | 666.5 | 675.2 | 684.4 | 695.1 |

| Aisle-by-aisle | | | number of cross aisles | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| # aisles | length | # items | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 |
| 7 | 10 | 10 | 148.5 | 144.3 | 153.7 | 164.6 | 177.5 | 189.9 | 205.4 | 216.4 | 230.3 | 245.6 |
| 7 | 10 | 30 | 192.1 | 207.2 | 227.0 | 246.7 | 268.5 | 289.5 | 315.7 | 333.6 | 357.6 | 383.8 |
| 15 | 10 | 10 | 235.2 | 220.7 | 229.4 | 241.0 | 255.1 | 268.9 | 286.3 | 298.4 | 313.6 | 330.9 |
| 15 | 10 | 30 | 356.7 | 349.3 | 369.1 | 392.5 | 421.3 | 449.8 | 486.3 | 509.8 | 542.1 | 578.6 |
| 7 | 30 | 10 | 304.7 | 268.0 | 268.4 | 276.5 | 287.5 | 299.6 | 313.0 | 325.7 | 339.0 | 351.8 |
| 7 | 30 | 30 | 418.8 | 413.9 | 422.8 | 438.5 | 457.2 | 477.7 | 500.4 | 522.5 | 544.1 | 565.8 |
| 15 | 30 | 10 | 427.2 | 362.0 | 358.6 | 366.8 | 378.3 | 391.7 | 406.6 | 420.6 | 435.3 | 449.7 |
| 15 | 30 | 30 | 732.7 | 648.8 | 642.8 | 658.6 | 682.0 | 709.1 | 739.7 | 769.4 | 798.9 | 828.9 |

| Combined | | | number of cross aisles | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| # aisles | length | # items | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 |
| 7 | 10 | 10 | 148.5 | 134.6 | 145.4 | 151.2 | 158.2 | 165.5 | 172.9 | 180.4 | 187.5 | 195.6 |
| 7 | 10 | 30 | 192.1 | 196.5 | 236.1 | 240.4 | 267.0 | 277.7 | 292.2 | 304.2 | 315.4 | 326.3 |
| 15 | 10 | 10 | 235.2 | 208.6 | 235.3 | 246.7 | 257.1 | 267.2 | 276.6 | 286.6 | 294.1 | 303.6 |
| 15 | 10 | 30 | 356.7 | 324.6 | 402.5 | 399.5 | 459.0 | 474.8 | 504.2 | 526.8 | 547.1 | 566.7 |
| 7 | 30 | 10 | 304.7 | 243.4 | 235.1 | 231.2 | 232.5 | 236.1 | 241.5 | 247.4 | 253.5 | 259.7 |
| 7 | 30 | 30 | 418.8 | 386.2 | 397.4 | 382.2 | 394.9 | 394.7 | 401.7 | 407.5 | 413.8 | 420.7 |
| 15 | 30 | 10 | 427.2 | 330.1 | 332.5 | 331.8 | 334.8 | 340.8 | 347.6 | 355.0 | 362.0 | 369.2 |
| 15 | 30 | 30 | 732.7 | 584.6 | 605.6 | 569.7 | 609.2 | 608.6 | 628.2 | 642.5 | 656.5 | 671.3 |

| Combined+ | | | number of cross aisles | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| # aisles | length | # items | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 |
| 7 | 10 | 10 | 148.5 | 133.2 | 136.2 | 140.2 | 146.0 | 152.1 | 159.4 | 165.9 | 173.3 | 180.8 |
| 7 | 10 | 30 | 192.1 | 196.0 | 224.7 | 232.0 | 244.2 | 253.2 | 260.7 | 268.9 | 276.6 | 283.9 |
| 15 | 10 | 10 | 235.2 | 207.0 | 214.5 | 220.8 | 227.6 | 234.3 | 242.0 | 249.3 | 257.0 | 264.7 |
| 15 | 10 | 30 | 356.7 | 323.6 | 373.3 | 379.6 | 401.4 | 415.3 | 425.5 | 437.2 | 447.6 | 457.2 |
| 7 | 30 | 10 | 304.7 | 235.4 | 219.3 | 214.9 | 216.2 | 219.8 | 225.2 | 230.8 | 237.2 | 243.4 |
| 7 | 30 | 30 | 418.8 | 381.7 | 379.3 | 365.8 | 362.7 | 360.6 | 361.6 | 363.6 | 366.6 | 370.6 |
| 15 | 30 | 10 | 427.2 | 323.1 | 305.3 | 300.5 | 301.0 | 304.3 | 310.1 | 315.7 | 322.0 | 328.5 |
| 15 | 30 | 30 | 732.7 | 577.6 | 567.6 | 539.8 | 538.5 | 536.7 | 537.8 | 541.6 | 545.2 | 550.7 |

Table 1 Average travel time for six routing methods.

| Percentage difference | | | number of cross aisles | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| # aisles | length | # items | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 |
| 7 | 10 | 10 | 7.1 | 2.7 | 3.6 | 3.3 | 3.0 | 2.8 | 2.5 | 2.4 | 2.2 | 2.0 |
| 7 | 10 | 30 | 2.9 | 2.4 | 13.2 | 12.0 | 12.9 | 12.6 | 10.9 | 10.6 | 9.4 | 8.5 |
| 15 | 10 | 10 | 7.1 | 2.5 | 6.5 | 7.6 | 7.7 | 7.4 | 6.8 | 6.6 | 6.1 | 5.5 |
| 15 | 10 | 30 | 5.7 | 2.9 | 19.8 | 20.2 | 23.7 | 24.6 | 22.9 | 22.9 | 21.4 | 19.9 |
| 7 | 30 | 10 | 13.0 | 5.6 | 3.9 | 2.8 | 2.2 | 1.9 | 1.8 | 1.5 | 1.4 | 1.4 |
| 7 | 30 | 30 | 5.2 | 5.7 | 10.6 | 8.7 | 8.6 | 7.9 | 7.3 | 6.7 | 6.3 | 6.0 |
| 15 | 30 | 10 | 13.2 | 4.9 | 4.9 | 4.5 | 4.1 | 3.7 | 3.6 | 3.4 | 3.2 | 3.2 |
| 15 | 30 | 30 | 10.1 | 6.8 | 14.5 | 12.5 | 13.8 | 13.5 | 13.0 | 12.7 | 12.2 | 12.0 |

Table 2 Percentage difference in average travel time between the combined+ and the optimal routing method.