Far East Journal of Mathematical Sciences (FJMS)

http://www.pphmj.com
http://dx.doi.org/10.17654/MS101050931
Volume 101, Number 5, 2017, Pages 931-971 **ISSN: 0972-0871**

AN EFFICIENT ALGORITHM FOR PROBLEM PRODUCT FAMILY ALLOCATION IN WAREHOUSES

Yung Chin Shih

Department of Mechanical Engineering
School of Engineering of São Carlos
University of São Paulo
São Carlos, Av. Trabalhador São Carlense, 400
CEP 13566-590, Brazil

Abstract

This investigates the problem of assigning pieces in shelves aiming at to minimize the routing distances. The idea of proposed global index is composed by other specific indices to identify which unallocated product is preferred to be allocated in corresponding section of a shelf. It is considered to be the most common allocation practice of storage item assignment and for the improvement of algorithms for comparison. The computational results show that it is rarely desired to minimize the total traveling distance since it implies in an increment of the total picking time due to delays in congestions. In addition to this, surprisingly, when the performance parameter is the total order consolidation time, it is preferable to work under large size and large variety of products circumstances.

Received: September 7, 2016; Accepted: November 20, 2016
2010 Mathematics Subject Classification: 90-XX.
Keywords and phrases: order picking, storage item assignment, global index.

List of Parameters

ABC	Classical allocation of products based on the degree of importance
B	Number of blocks
$B'_{si,\,j}$	One of B that pursues the section $S_{i,\,j}$
ch	Number of horizontal corridors, that is, cross aisles plus side aisles
C_k	Number of components classified as belonging to the level k
CRA	Components randomly allocated
cv	Number of vertical corridors, that is, subaisles plus side aisles
d_{cart}	Size of the picking cart, assuming square format
$D_{c_{k_p}}$	The demand of the piece located in level k of product p
$Dr_{Si,\,j}$	Number of drawers in the section $i,\ j$
f	A family of products
F	Total number of families, that is, all families f
F'	All unallocated pieces of different families
f'	The selected f of F'
h	Number of sections horizontally positioned per shelf
$i,\ j$	Alternatively x and y, but are used to distinguish x and y of hamming distance calculation
idx	Number of used indexes
K_f	Number of levels of family f, including the level corresponding to the module

$K_{\max}$	Highest value of K_f
$K_{\min}$	Lowest value of K_f
L	Capacity of the picking cart
LBS	Local Beam Search
$L_{subaisle}$	Length of subaisle
MRA	Modules randomly allocated
n	Number of picking cart used in picking process
P	All demanded products
p	Demanded product
P^*	New P after batching quantities of similar products of the demand
Sa	Number of sections per subaisle
Sc	Number of sections per shelf
$S_{i,j}$	Section of the warehouse, located in $x = i$ and $y = j$
$St_{i,j,t,c_{k_f}}$	The level of stock of piece c_{k_f} located in position i, j in tth drawer
Sw	Number of shelves in the warehouse
TOCT	Total order consolidation time
TPT	Total picking time per each picker
u.d.	Unit of distance
v	Number of sections vertically positioned per shelf
V_f	Variety or number of products of the family f
v_{speed}	Speed of picking cart
x	Abscissa coordinate of any allocated position of piece in the warehouse

X	Number of rows of warehouse
x^*	Abscissa coordinate of the centroid of warehouse
y	Ordinate coordinate of any allocated position of piece in the warehouse
Y	Number of columns of the warehouse
y^*	Ordinate coordinate of the centroid of warehouse
$y_{ch'}$	Ordinate coordinate of one of the selected *ch*
$y_{ch''}$	Ordinate coordinate of other selected *ch*
W	Width of aisles (cross aisles or subaisles)
$w_{c_{k_p}}$	Weight of a certain piece c_{k_p} (one of C_{k_p})

1. Introduction

One of the most challenging problems faced by the companies is the ability for the quick tooling setups to make possible the production of a large variety of products in short period of time. However, in many cases, a constant tool setup may not be possible because of machine features, of tooling setups costs, of technical issues, of products shapes and so forth. Therefore, developing products is an important opportunity discovered by companies in which similar components (called *modules*) are of the common use among of them, yielding the desired variety.

The rapid response to the variety of products demanded by customers induces the necessity of overall constant operations improvement, which are regarding to the storage and order picking activities of components in warehouses specially. Extra routes and time might be observed once these operations are conducted inefficiently. Thus, the direct consequence is incrementing the cost of final product and likewise the long wait of customers to receive the requested product.

Traditionally, different pieces to be collected are dependent on each other. That is, the allocation and the order picking are conducted based on the customer level of demand information among other criterions. However, a lack of investigation is observed in the literature concerning the relation of multiple pieces (in some regions of Brazil is named module and components of the same product). This paper aims to investigate the relation among the multiple pieces of the same product and how it affects the cited activities.

The method consists in an index based on dynamic weights and it is proposed which basically consists in defining the order priority among unallocated products (module and its respective components) for allocation to result in the lowest total traveling distance during the order picking.

Eight methods are used for comparison where each method has three layout sizes as small, medium and large resulting three indicators. The proposed method is implemented in an available software where it is also modeled the most common allocation practices of storage item assignment (classical ABC, MRA and CRA) and allocation improved algorithms (local beam search, GA, 2-opt and 3-opt) totalizing 8 comparisons of performances for three sizes warehouse (large, medium and small). For each warehouses size, there are three performance indicators in 8 items storage assignment such as total traveling distance, total order picking time and CPU time.

This paper consists of five sections. The second section presents a brief definition of terms which are used in order picking activities and also a literature review. The proposed mathematical model and its respective constraints such as the description for the implementation are presented in the third section. Likewise, the description of other methods is used in comparison. Next, the definition of input data, simulation results, discussion and comparison of methods are presented in the fourth section. Finally, in the last section, the conclusions and suggestions for future works.

2. Literature Review

The operation improvement in a warehouse generally involves the decision related to the following classical issues: zoning, layout, routing strategy, orders batching and storage item assignment. This section aims to present the review of some articles regarding these activities, showing techniques, discussing results and limitations.

According to De Koster et al. [4], this picker-to-item system (while the picker moves to section) and low level-picking systems (there are no difficulties during the collecting process due to the altitude of stored item) are the most common practices in Western Europe and the reason those picking systems are considered in this present research. Azadnia et al. [1] claimed in most practical situations, customers actually define a due date, when the warehouse operator must satisfy the customers request until this moment. So, the authors proposed a model in order to minimize the tardiness of the order (difference between the time completion of order and the established due date).

There are already many researches with emphasis on heuristics for the batching orders, storage item assignment, routing problems, etc. But in practice, there are congestions of pickers in aisles generally omitted in researches, see Kłodawski and Żak [6]. To solve concomitantly the problem of storage assignment, and batching and picking tour, Ene and Öztürk [9] proposed an integer programming based on mathematical model. Henn et al. [5] focused on the development of metaheuristics (iterated local search and ant colony optimization) to determine how orders should have been batched to result in minimization of total length of all tours. An important analysis was observed in Mowrey and Parikh [7], where the influence of width of aisles on the performance during the order picking process was evaluated.

Based on the reviewed articles, there is a clear evidence that the structure of the product during the piece storage assignment has been omitted. We may raise some issues about it: should the allocation begin from the modules or from the components? What is the best position for each module? If the remained drawers of best position are not capable in allocating a determined module, which section with empty drawers should be chosen and which module now is preferable for the allocation? Are all of them equally important?

The contribution of this research is to propose an index method based on dynamic weights by converting the raised issues into indexes varying from 0 to 1 to define order of unallocated components for the allocation in the shelf considering three performance comparisons: the total traveling distance, total order consolidation time and CPU time.

3. The Method

3.1. Formulation of the mathematical model

This subsection presents the adopted mathematical model (z) for the waving picking problem, when all pickers start the picking at the same time in a narrow aisle warehouse. In this type of setting, we assume manual picker may reach both shelves of the same subaisle, omitting any zigzag travelings.

We may formulate the following mathematical model z, see expression (1), which aims to calculate the total traveling distance of collected pieces (module and components) of each family, repeating it for all families. The z_1 expression aims to maximize the distance of modules of the same family f, finding the ideal position (x_{v_1}, y_{v_1}) to the module, and the z_2 expression aims to minimize the distance of components to the ideal position, that is, allocating components of the product closer to the ideal position. Subjects to:

$$z = \underbrace{\left[\sum_{f=1}^{F} \max\left(\sum_{v_1=2}^{V_f} \sum_{v_2=1}^{V_f - 1} | x_{v_1} - x_{v_2} | + | y_{v_1} - y_{v_2} | \right) \right.}_{z_1}$$

$$\left. + \underbrace{\min\left(\sum_{v_1=1}^{V_f} \sum_{k=1}^{K_f - 1} | x_k - x_{v_1} | + | y_k - y_{v_1} | \right)}_{z_2} \right], \tag{1}$$

$$\forall x,\ y \in \{1 \ldots X, 1 \ldots Y\}, \quad X > 0, \quad Y > 0, \tag{2}$$

$$F > 0, \tag{3}$$

$$X = cv \cdot h + (cv - h); \quad h > 0; \quad cv \geq 2, \tag{4}$$

$$Y = ch \cdot v + (ch - v); \quad v > 0; \quad ch \geq 2, \tag{5}$$

$$1 \leq K_f < K_{\max}; \tag{6}$$

$$V_f = \prod_{k_f=1}^{K_f-1} C_{k_f}, \tag{7}$$

$$1 \leq C_{k_f} \leq C_{\max}; \quad C_{\max} < K_{\max}, \tag{8}$$

$$\sum_i \sum_j Dr_{Si,\,j} = \sum_{f=1}^{F} K_f . V_f, \tag{9}$$

$$Dr_{Si,\,j} > 0, \tag{10}$$

$$i,\ j \in \{1 \ldots X, 1 \ldots Y\}; \quad i,\ j > 0. \tag{11}$$

The constraint (2) refers to the horizontal *x* and to the vertical *y* coordinates of the section where family, product or component is allocated. The number of families must be higher than 0 (see expression (3)). Next, both constraints (4) and (5) aim to deduce the number of rows and columns for the warehouse. The incognita *h* is established to be higher than 0 because the shelf must pursue at least one section horizontally positioned. Also, $cv \geq 2$ assures that two side vertical aisles will exist. For *v*, it must be higher than 0 because there must exist at least one section vertically positioned in the shelf. As the same manner, $ch \geq 2$ assures there are at least two side horizontal aisles. The definition of number of levels for each *f* is defined in constraint (6). The inequality is because $K_{\max}$ must not be equal to 1, because there are at least one level for module and one level for component. Next constraint defines the total number of products per family *f*; see expression (7). The constraint (8) assures that there is at least one component in each level, and also, there is at least one level for module. Although K_f and C_k are defined, both must be equal to the number of drawers in the sections *i* and *j*, $Dr_{Si,\,j}$, exactly what the constraint (9) does. So, based on these last two constraints, we may affirm that each drawer will store exactly one type of piece. In each section $S_{i,j}$, there must be at least one drawer, $Dr_{Si,j} > 0$, in constraint (10). And finally, in (11), i and j represent the position of section $S_{i,j}$ in the warehouse, in Cartesian coordinates.

3.2. Computational implementation

This subsection describes steps for the computational implementation of the proposed mathematical model for the total traveling distance calculation, which consists in:

Step 0. Define the number of families F, the number of levels K_f and the number of components belonging to each level. Due to the dependency between $K_{\max}$ and $C_{\max}$, only one of them needs to be defined, see Appendix 3. In this research, we defined a value for $C_{\max}$. Moreover, it defines the weight of each component w_{fk}, and attributes to $it \leftarrow 0$ and $D^* \leftarrow \{\text{large enough}\}$.

Step 1. It defines the stochastic demand for f. We assume η [0.1 to 1] (that is, if the demand for a certain f is deterministic, then $\eta = 1$).

Step 2. It generates the layout of the warehouse. The incognitos X and Y are obtained by using the next expressions (12) and (13) instead of previous (4) and (5), see Appendices 2, 3 and 1:

$$X = 2 \cdot \left(\frac{Sw}{B}\right) + cv^* + \left(\frac{Sw}{B}\right) - 1, \text{ if } Sw \bmod B \text{ is equal to zero,} \tag{12}$$

$$Y = \left(\frac{Sc}{2}\right) \cdot B + ch^* + B - 1, \text{ if } Sc \bmod 2 \text{ is equal to zero.} \tag{13}$$

Step 3. It calculates the ideal position (section) for modules of each product, by (14):

$$Cycle_{f=1\ldots F} = \frac{Sw \cdot Sc}{V_f}. \tag{14}$$

Step 4. Allocate pieces of the product of f. Repeat it for all other products of family f in other positions defined in Step 3. Repeat it up to F. All f allocated in the section are denoted by F^*. A set of F^* is $F^{*'}$. Note that many different products from different families may be allocated in the same $S_{i,j}$. Due to over-allocation, we need to define the degree of the importance of f families allocated.

Step 5. The definition of degree of the importance is based on an objective function, see (15). The pieces (module and components of one product) of f family, which are not preferred in comparison to other f for allocation the referred section (ideal position) are temporally removed. Repeat it for all sections that are over-allocated:

$$\min\left(Dr_{Si,j} - \max\left\{ \sum_{f^{*'}}^{F^{*'}} \sum_{1}^{K_{f^{*'}}} 1 \right\} \right). \tag{15}$$

Step 6. Due to the impossibility of F' being allocated in the ideal position, the idea is to find sections closest to the ideal position that pieces should be allocated. To do that, calculate $x^* = |1 + X|/2$ and $y^* = |1 + Y|/2$. Find the closest $x^{*'}$ and $y^{*'}$ to x^* and y^*. Update $x^* \leftarrow x^{*'}$ and $y^* \leftarrow y^{*'}$.

Step 7. Define a new value for disturbance factor for each used index; Attribute $it \leftarrow it + 1$.

Step 8. From x^* and y^*, find the closest new candidate to be $S_{i,j}$, see (16) and (17):

- If $S_{i,j} = S_{x^*, y^*}$, then $d' = \min\{| x^* - x_{Si,j} | + | y^* - y_{Si,j} |\} = 0$. (16)

- If $S_{i,j} \neq S_{x^*, y^*}$, then:

- If $(B'_{Si,j} \neq B'_{Sx^*, y^*})$ or $(B'_{Si,j} = B'_{Sx^*, y^*}$ and $y_{Si,j} = y_{Sx^*, y^*})$, then

$$d' = \min\{| x_{Si,j} - x_{Sx^*, y^*} | + | y_{Si,j} - y_{Sx^*, y^*} |\}. \tag{17}$$

- If $(B'_{Si,\,j} = B'_{Sx^*,\,y^*}$ and $y_{Si,\,j} \neq y_{Sx^*,\,y^*})$, then

$$d' = \min[|\, x_{Si,\,j} - x_{Sx^*,\,y^*} \,| + |\, y_{Si,\,j} - y_{Sx^*,\,y^*} \,|$$

$$+ 2^* \min\{|\, y_{Si,\,j} - y_{ch'} \,|; |\, y_{Sx^*,\,y^*} - y_{ch''} \,|\}].$$

Step 9. Calculate the global index (Appendix 1). So, the f' of all F', which pursues highest value of global index, is then firstly selected to be allocated in the empty section (18):

$$\text{Max}\{Global\_Index_{Si,\,j,\,f'=1}, Global\_Index_{Si,\,j,\,f'=2}, \ldots,$$

$$Global\_Index_{Si,\,j,\,f'=F'}\}. \tag{18}$$

During the allocation process if the number of pieces of family f' is higher than the number of remained drawers, then go to Step 10, otherwise go to Step 11.

Step 10. The position of the last section is now x^* and y^* (19):

$$x^* \leftarrow x_{Si,\,j},$$

$$y^* \leftarrow y_{Si,\,j}. \tag{19}$$

Repeat this step if necessary up to allocate all remained pieces of f'.

Step 11. Update F'. Repeat Steps 8-10 for all F'.

Step 12. Define the level of stock. For a given piece c_{k_p}, the demand may be represented as $D_{c_{k_p}}$ (20):

$$\sum_{p=1}^{P} D_{c_{k_p}} = D'_{c_{k_p}}$$

$$= \sum_{i=1}^{X} \sum_{j=1}^{J} \sum_{t=1}^{Dr_{Si,\,j}} St_{i,\,j,\,t,\,c_{k_f}} \quad \text{if piece of } c_{k_p} = \text{piece of } c_{k_f}. \tag{20}$$

The calculation is for all different c_{k_p}, resulting in different $D'_{c_{k_p}}$. We select the highest obtained value to be the stock for all other pieces (21), through:

$$St_{i,\,j,\,t} = \max\{D'_{c_{k_p}}\} \text{ for all different pieces } c_{k_p};\ \forall i,\ j,\ t. \tag{21}$$

Step 13. Calculate the traveling distance. For all p, the total number of used picking carts n can be represented as (22) and (23), where $L > 0$ and $w_{c_{k_p}} < L$:

$$\sum_{p=1}^{P} \sum_{k_p=1}^{K_p} \sum_{c_{k_p}=1}^{1} w_{c_{k_p}} \cdot D_{c_{k_p}} \le \sum_{cart=1}^{n} 1 \cdot L_{cart}. \tag{22}$$

If $L_1 = L_2 = \cdots = L_n = L,$ then we may write the expression (22) as:

$$n \ge \frac{\sum_{p=1}^{P} \sum_{k_p=1}^{K_p} \sum_{c_{k_p}=1}^{1} w_{c_{k_p}} .D_{c_{k_p}}}{L}. \tag{23}$$

Step 13.1. Attribute to the incognitos D_{total}, $W_{totalweight}$, k_p, n', which represent the values of the current total traveling distance, total current load of the picking cart, the piece of level k_p and the number of used picking carts, values equal to zero, zero, zero and one, respectively; attribute also $k_p \leftarrow k_p + 1$.

Step 13.2. Check up the weight of the total current load of the picking cart:

- **Step 13.2.1.** For piece $k_p = 1$. If $(W_{totalweight} + w_{c_{k_p}} \le L)$, then the total current weight is updated to $W_{totalweight} \leftarrow W_{totalweight} + w_{c_{k_p}}$. Go to 13.3.
- **Step 13.2.2.** For $k_p > 1$. If $(W_{totalweight} + w_{c_{k_p}} \le L)$, then the total

current weight is updated to $W_{totalweight} \leftarrow W_{totalweight} + w_{c_{k_p}}$. Go to Step 13.4.

- **Step 13.2.3.** For $k_p = 1$ or $k_p > 1$. If $(W_{totalweight} + w_{c_{k_p}} > L)$, then $D_{total} \leftarrow D_{total} + | x_{out} - x_{Si,\,j} | + | y_{out} - y_{si,\,j} |$; $W_{totalweight} \leftarrow 0$; go to Step 13.2.

Step 13.3. Case $k_p = 1$, identify a random $S_{i,\,j}$ where the piece k_p is stored. If $St_{i,\,j,\,t} \neq 0$, then $D_{total} \leftarrow D_{total} + | x_{in} - x_{Si,\,j} | + | y_{in} - y_{Si,\,j} |$. If $St_{i,\,j,\,t} = 0$, then choose randomly the other $S_{i',\,j'}$ up to $St_{i',\,j',\,t} \neq 0$. Make $S_{i,\,j} \leftarrow S_{i',\,j'}$. Go to Step 13.5.

Step 13.4. Case $k_p > 1$, identify the closest $S_{i',\,j'}$ (non empty stock) in relation to $S_{i,\,j}$. The incognito $S_{i,\,j}$ represents the current section, and the incognito $S_{i',\,j'}$ represents the next section.

- **Step 13.4.1.**

  If $S_{i,\,j} = S_{i',\,j'}$, then $D_{total} \leftarrow D_{total}$.

- **Step 13.4.2.**

  If $S_{i,\,j} \neq S_{i',\,j'}$, then

  If $(B'_{Si,\,j} \neq B'_{Si',\,j'})$ or $(B'_{Si,\,j} = B'_{Si',\,j'}$ and $y_{Si,\,j} = y_{Si',\,j'})$, then:

  $$D_{total} = | x_{Si,\,j} - x_{Si',\,j'} | + | y_{Si,\,j} - y_{Si',\,j'} | + D_{total}.$$

  If $(B'_{Si,\,j} = B'_{Si',\,j'}$ and $y_{Si,\,j} \neq y_{Si',\,j'})$, then

  $$D_{total} \leftarrow | x_{Si,\,j} - x_{Si',\,j'} | + | y_{Si,\,j} - y_{Si',\,j'} | + 2^* \min\{| y_{Si,\,j} - y_{ch'} |; | y_{Si',\,j'} - y_{ch''} |\} + D_{total}.$$

Step 13.5. Update the level of the stock in the section.

$St_{i,\,j,\,t} \leftarrow St_{i,\,j,\,t} - D_{k_p}.$

If $k_p > 1$, then

$$S_{i,\,j} \leftarrow S_{i',\,j'} \text{ and } St_{i,\,j,\,t} \leftarrow St_{i,\,j,\,t} - D_{k_p}.$$

Step 13.6. If $k_p < K_p$, then make $k_p \leftarrow k_p + 1$. Else, for the next p, attribute $k_p \leftarrow 1$. For both cases, repeat Steps 13.2, 13.3, 13.4, 13.5 and 13.6 up to $n' = n$.

Step 14. If $D_{total} > D^*$, then

$D^* \leftarrow D_{total}$ and keep the best used disturbance factors by calling, for instance, α^*, β^*, γ^*, etc.

Step 15. Reset the warehouse. Repeat Step 6 up to Step 15 by varying values for disturbance factors up to *it* = *iterations*.

3.3. An analytical model for the congestion problem in subaisle

To evaluate the wasted time in congestion, let us construct the idea of the suggested analytical model. If all cart pickers (or cart picking) are in the same section, such as shown in Figure 1, then (24):

$$\sum_{cart=1}^{n} d_{cart} \leq W, \tag{24}$$

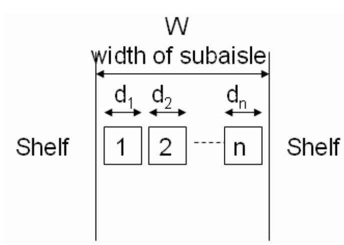

Figure 1. Position of cart pickers in the same section of subaisle.

The probability to have n vehicles in the section may be expressed as:

$\sum_{cart=1}^{n} \psi_{cart}.d_{cart} \leq W.$ Based on the reviewed articles, we did not find the exact proportion of the width W and length of subaisle $L_{subaisle}$, but we found typical narrow aisle warehouse, wherein $L_{subaisle}$ must be longer enough in relation to the other. In other words, $\frac{W}{L_{subaisle}} \to 0.$ By assuming this constraint, the zigzag traveling time in subaisle may be omitted.

The sensation of congestion is noticed when the obtained number of carts in the section is higher than the maximum acceptable number of carts, n^* (25):

$$\sum_{cart=1}^{n} \psi_{cart}.d_{cart} \leq \sum_{cart=1}^{n^*} \psi_{cart}.d_{cart}. \tag{25}$$

It is important to note that the congestion of one section will block the subaisle. Assuming any cart as same size, $d_1 = d_2 = \cdots = d_n = d \neq 0,$ the blocking time in congestion is (26) and (27):

$$\sum_{cart=1}^{n} \psi_{cart} \leq n^* \to t_{blocking\_in\_congestion} = 0. \tag{26}$$

This means no congestion, and consequently no blocking time in congestion.

$$\sum_{cart=1}^{n} \psi_{cart} > n^* \to t_{blocking\_in\_congestion}$$
$$= \left\{ \left[trunc\left(\frac{\sum_{cart=1}^{n} \psi_{cart}}{n^*} \right) + 1 \right] . \frac{L_{section\, S_{i,j}}}{v_{speed}} \right\} \tag{27}$$

which means that the last picker of $\sum_{cart=1}^{n} \psi_{cart}$ will wait

$\left[trunc\left(\frac{\sum_{cart=1}^{n} \Psi_{cart}}{n^*} \right) + 1 \right]$ pickers leaving the section $S_{i,\,j}$, where $L_{section\,Si,\,j}$ is the length of each section, given in unit of distance.

Nevertheless, the probability of cart to be in a specific section depends on the demand (28):

$$\Psi_{cart} = \sum_{cart=1}^{n} \sum_{p'_{cart}=1}^{P'_{cart}} \sum_{k_{p'cart}=1}^{K_{p'cart}} \Theta_{cart,\, p'_{cart},\, k_{p'cart}} \cdot \frac{1}{V_{cart,\, p'_{cart},\, k_{p'cart}}}, \quad (28)$$

where $\Theta_{cart,\, p'_{cart},\, k_{p'cart}}$ is 0 if the requested piece of $k_{p'cart}$ of *cart*th is not located in drawers of the section $S_{i,\,j}$, and 1 else. P' refers to the number of products which belongs to cart, $K_{p'cart}$ refers to the number of levels of product p' of cited cart, $V_{cart,\, p'_{cart},\, k_{p'cart}}$ refers to the total possible products of the family (to which p' belonged to) and *n* refers to the number of used picking carts. The expression (28) is for only one section $S_{i,\,j}$ of a subaisle. Since the number of picking carts in the subaisle depends on all sections S_a of the same subaisle (29),

$$\Psi_{cart} = \sum_{s_a=1}^{S_a} \sum_{cart=1}^{n} \sum_{p'_{cart}=1}^{P'_{cart}} \sum_{k_{p'cart}=1}^{K_{p'cart}} \Theta_{cart,\, p'_{cart},\, k_{p'cart}} \cdot \frac{1}{V_{cart,\, p'_{cart},\, k_{p'cart}}}. \quad (29)$$

Assuming the worst case, that is, the pickers are running in the opposite flow and the pickers n^* are located in the opposite side of the subaisle, the blocking time in the subaisle is (30):

$$\sum_{cart=1}^{n} \Psi_{cart} > n^* \rightarrow t_{blocking\_in\_congestion}$$

$$= \left\{ \left[trunc\left(\frac{\sum_{cart=1}^{n} \Psi_{cart}}{n^*} \right) + 1 \right] \cdot \frac{L_{subaisle}}{v_{speed}} \right\}. \tag{30}$$

It is worth noting that when the adopted value of n^* is higher, the time in congestion tends to be lower, or vice versa.

The blocking time (30) is the time that nth picker wastes in the subaisle up to his turn to collect only one piece. Due to each picker is designed to collect several pieces of p' up to reach the picking cart capacity, the estimative of total picking time per each picker (including the traveling time, time in congestion and picking time) is (31):

$$TPT = \frac{D_{total}}{n \cdot v_{speed}} + \sum_{p'_{cart'}=1}^{P'_{cart'}} \sum_{k_{p'cart'}=1}^{K_{p'cart'}} \left\{ \left[trunc\left(\frac{\sum_{cart=1}^{n} \Psi_{cart}}{n^*} \right) + 1 \right] \cdot \frac{L_{subaisle}}{v_{speed}} \right\}$$

$$+ t_{picking} \cdot \sum_{p'_{cart}}^{P'_{cart}} \sum_{k_{p'cart}}^{K_{p'cart}} 1. \tag{31}$$

Since the order picking is considered consolidated if all pickers finish the picking process, we concerned about the one who demanded highest time (32):

$$TOCT = \max\{TPT_{cart=1};\ TPT_{cart=2};\ \ldots;\ TPT_{cart=n}\}. \tag{32}$$

4. Computational Experiments

4.1. Definition of input parameters

This subsection describes the input parameters used to simulate the constructed models. First, the simulation is conducted to the most common practices (classical ABC, MRA and CRA). Then local search algorithms are executed LBS, GA, 2-opt and 3-opt, and finally, the proposed method. In order to achieve this, some input parameter must be firstly defined, and others then derived from it:

$F = 300$; $B = 2$; $Sc = 20$; $In = 1$; $Out = 5$; $L = 100\text{kg}$; $P = 90$; $K = [1 \ldots K_{\max}]$; $C = [1 \ldots 5]$; $\varepsilon = 1$; $Dr_{Si,\,j} = [1 \ldots 5]$; $w_{flq} = [100\text{g} \ldots 1\text{kg}]$; $L_{section\, S_{i,\,j}} = W = 1u.d$; $L_{subaisle} = 20u.d$; $\Phi_1 = \Phi_{III} = 60\%$; $\Phi_2 = \Phi_{II} = 30\%$ and $\Phi_3 = \Phi_I = 10\%$; $p_{crossover} = 0.8$; $p_{mutation} = 0.02$; *iterations* = 4200; *pop* = 14; *generations* = 299 (that is, 299 times of multiple of 14, plus 14 of *ger* = 0, totalizing 4200 iterations) (each individual represents one iteration).

Note there are still unknown incognitos, *X*, *Y*, $St_{i,\,j,\,h}$ and V_f. Having *X* and *Y*, the size of the warehouse is determined. We intend to conduct experiment for three sizes of warehouse. By replacing the input value of *Sc* and *B* to the expression (A2.2), *Y* is obtained. Now, *Sw* is necessary to deduce *X*, which according to the expression (A3.4) depends on Q'. Therefore, to simulate three sizes of warehouse, three values of Q' must be defined. The suggested height of shelf for manual picking is when the dimension is lower than 1.8m. Considering $Q' = 1$, $Q' = 2$ and $Q' = 3$ equal to 7, 13 and 19, respectively (which means the dimension of each drawer varies from 9cm up to 25cm), then from (A3.3), (A3.4) and replacing $R = 7$:

$$Sw_1 20(4) \approx 300(3.5)\left(\frac{3.5}{2.5}\right)^{2.5} \approx 2436 \rightarrow Sw_1 \approx 30.$$

Using the same calculation procedure, we obtain also $Sw_2 \approx 20$ and $Sw_3 \approx 10$.

From (21), the level of stock may be estimated from the demand. Statistically, according to the percentage of products (see Appendix 4), it is also possible to estimate the level of stock. To make this becomes possible, it is necessary V_f (33):

$$V_f = \prod_{k_f=1}^{K_f-1} C_{k_f}. \tag{33}$$

As $p_1 = 60\%$ and $p_I = 10\%$, then from (A4.4):

$$St_{i,j,h} = \max\left\{\frac{V_f \cdot p_1 \cdot P}{p_I \cdot F}\right\}.$$

For $V_f = 1$, $St_{i,j,h}^1 = 1.8$.

For $V_f = 5$, $St_{i,j,h}^2 = 9$.

Therefore, $St_{i,j,h} = \max\{1.8; 9\} = 9$ units.

The models are developed in platform Pascal, and simulated in a microcomputer with the following settings: 2-GHz Pentium(R) Dual-Core CPU, 4GB DDR2 RAM.

4.2. Results of total traveling distance

The average of the results (in units of distance) and the respective standard deviation of a total of 30 samples are shown in Table 1. The consumed CPU time for 30 samples is, in average, 12 minutes.

Comparing results presented in Table 1, for small size warehouse, MRA and CRA present quite similar results, but both with better performance in comparison to the allocation based on ABC (reduction in 18.62% of the total traveling distance). For medium size, the reduction is about 16.27% and finally, for large size, the reduction is about 20.54%. It is observed the proportionality of traveling distance in relation to the size of the warehouse, that is, as the size of the analyzed warehouse increases, the traveling distance also increases.

Table 1. Performances of three layout size

Allocation (30 samples)	10 shelves	20 shelves	30 shelves
MRA	Average=2114.63 S.D.=95.26	Average=3017.60 S.D.=126.15	Average=4027.03 S.D.=171.60
CRA	Average=2147.13 S.D.=99.75	Average=3101.50 S.D.=141.47	Average=4083.77 S.D.=147.71
ABC	Average=2618.37 S.D.=117.12	Average=3654.03 S.D.=155.36	Average=5103.70 S.D.=236.95

Figures 2-4 exhibit results for the improved algorithms and the proposed method. Based on the preliminary results, in general, increasing the warehouse size, the performance of GA for allocation of pieces increases. The lowest total traveling distance is obtained in about 35~45 generations (that is, around 504~644 iterations). No improvement evidence is observed from it up to the last adopted generation 299. The consumed CPU time is about 36h.

On the other hand, the use of LBS seems to be the most appropriate for small warehouse size. Reduced total traveling distance may be obtained by incrementing the number of iterations. The consumed CPU time up to 4200 iterations (or 299 generations) is 33h50min.

In comparison to others methods (2-opt and 3-opt), the performance is between LBS and GA, with CPU time between 27h30min and 31h24min.

About the proposed method, comparing results of the proposed method we note, in general, reduced traveling distance (which means best allocations of pieces to drawers) may be obtained in a short period of CPU time (in average, about 1h10min is necessary to reach 100 iterations).

For some more details, we present Table 2, showing the evolution of total traveling distance, in units of distance, as the number of iterations increases.

For 10 shelves, the 1718u.d. of traveling distance is reached in about 5 generations (or 5 iterations of multiple of 14 plus 14), which is equivalent to about 80th iteration (with $\alpha^* = 0.8$; $\beta^* = 0.7$; $\gamma^* = 0.1$; $\delta^* = 0.4$). The consumed CPU time is about 50min. While in LBS, this value is reached in about 160 generations (or 160 × 14 iterations plus 14) with 18.27h of CPU time.

For 20 shelves, the best traveling distance is 2359u.d., reached in 95th iteration, weighing of $\alpha^* = 1.0$; $\beta^* = 0.2$; $\gamma^* = 0.1$; $\delta^* = 0.3$ (CPU ≈ 66.5min). The best result may be reached only in about $(270 \times 14 + 14)$ iterations with a CPU time of 29h.

Lastly, for 30 shelves, the best value of traveling distance is 2860u.d. obtained in 72th iteration weighing $\alpha^* = 0.1$; $\beta^* = 0.2$; $\gamma^* = 0.9$; $\delta^* = 0.4$ (CPU ≈ 50.4min), while in GA, the only best total traveling distance is 3057u.d. (reached in about 35 generations with CPU ≈ 4.21h). After this, no improve evidence is observed.

Based on the analysis, while the warehouse size increases, the performance of the proposed method also increases. We notice also larger warehouses tend to retain also long trips.

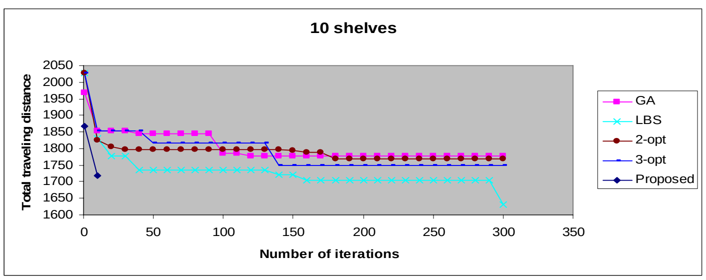

Figure 2. Total traveling distance, in units of distance.

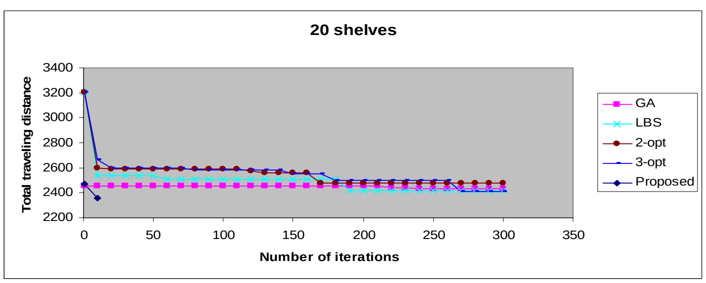

Figure 3. Total traveling distance, in units of distance.

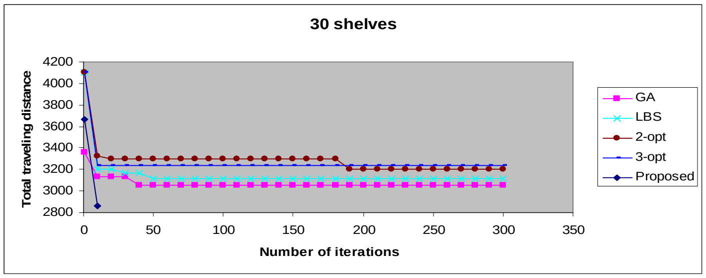

Figure 4. Total traveling distance, in units of distance.

There is a notable evidence of the proposed method outperforms in comparison to other methods. Observe although the total traveling distance increases for larger warehouse size, the performance of the proposed method is higher since lowest result is obtained in few iterations.

It is interesting to note that the CPU time is not affected by the warehouse size.

Table 2. Evolution of the total traveling distance

Number of iterations	**Number of shelves**		
	10	**20**	**30**
1	1866	2470	3665
10	1810	2470	3194
20	1742	2391	3182
30	1742	2391	3182
40	1730	2391	3046
50	1730	2391	3032
60	1730	2371	3032
70	1730	2371	3032
80	1718	2371	2860
90	1718	2371	2860
100	1718	2359	2860
Best weights	$\alpha^* = 0.80$; $\beta^* = 0.70$; $\gamma^* = 0.10$; $\delta^* = 0.40$	$\alpha^* = 1.00$; $\beta^* = 0.20$; $\gamma^* = 0.10$; $\delta^* = 0.30$	$\alpha^* = 0.10$; $\beta^* = 0.20$; $\gamma^* = 0.90$; $\delta^* = 0.40$

4.3. Total time for order picking

This subsection consists in evaluating the relation between the perception of congestion n^* and total order picking for each storage item assignment.

The simulation of the order picking time requires: picking time = 0.1min or 6s, see Lin and Lu [3]; $v_{speed} = 0.6\,\text{m/s}$ (Roodbergen and De Koster [8]), omitting the influence of weight on the velocity and the congestion in cross aisles; distance between two neighboring aisles = 2.5m (Roodbergen and De Koster [8]), that is, each unit of distance adopted in this research will be 0.8m. Other incognitos are obtained as simulation progresses.

Figures 5 to 10 show that it is highly suggested to work with MRA and CRA in cases of large picklist because, in general, the total traveling distance and the picking time are lower than ABC. Actually, it was expected (and now confirmed) higher picking time in strategy ABC of allocation, since higher concentration of pickers in some areas will also promote more waiting during the pickings. On the other side, the performances of ABC, MRA and CRA are quite similar for a small picklist.

Comparing iterative improvement methods and the proposed method, there is a clear evidence that increasing the warehouse size, there is no effect of congestion (this result actually was expected) explaining the reason that picking time is quite similar among themselves, especially for small picking list. It is because the probability of pickers in the same subaisle is lower. On the other hand, smaller warehouse provides lower total traveling distance, but increment in the picking time due to congestion.

It is noticed a trade-off behavior. That is, rarely is suggested to reach lowest total traveling distance because it promotes higher total picking time. However, in general, when the number of picklists is kept constant ($P = 90$), the lowest total traveling distance also will result in the lowest total picking time.

In general, keeping the size of warehouse, if the variety of demanded products increases, then the picking time increases too. However, when the warehouse size is increased, the performance in picking time improves. It is preferable to work with larger warehouses and larger variety of demanded products than smaller warehouses and lower variety of demanded parts.

Especially, Figures 8-10 demonstrate that if the goal is to minimize the total traveling distance, then it is suggested to employ only one cross aisle (that is, only two blocks). It is useful to explain why Roodbergen and De Koster [8] considered only one cross aisle in the analysis, that under circumstances, it was simply defined.

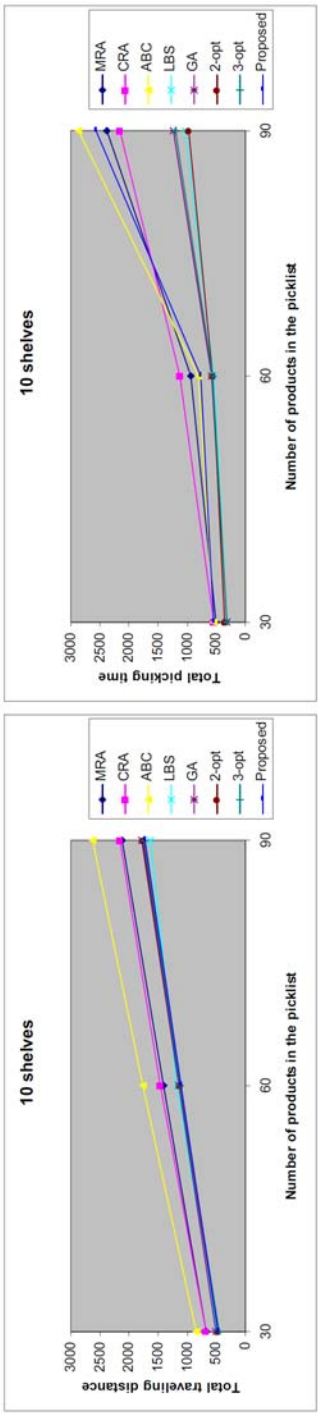

Figure 5. Performances with one cross aisle.

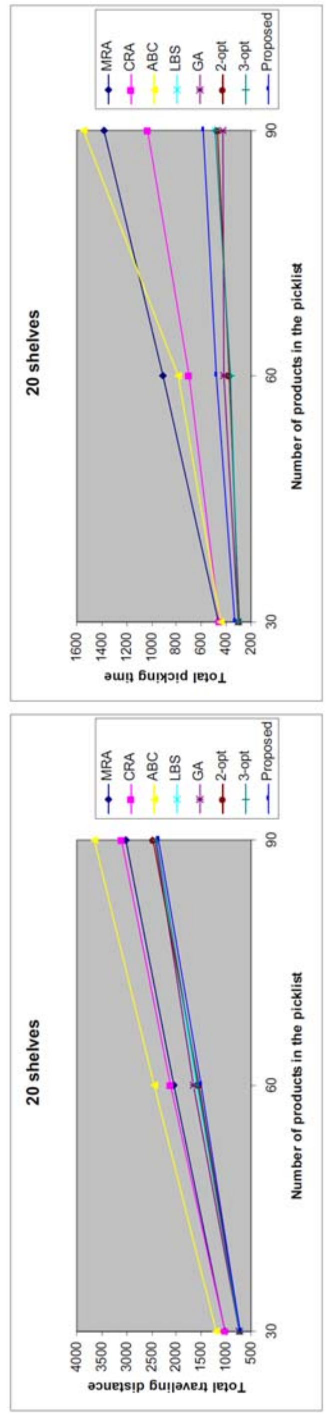

Figure 6. Performances with one cross aisle.

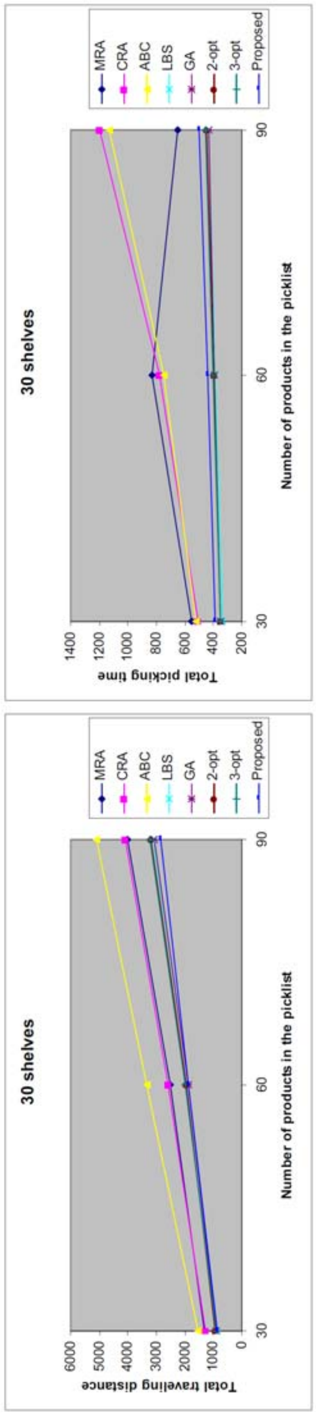

Figure 7. Performances with one cross aisle.

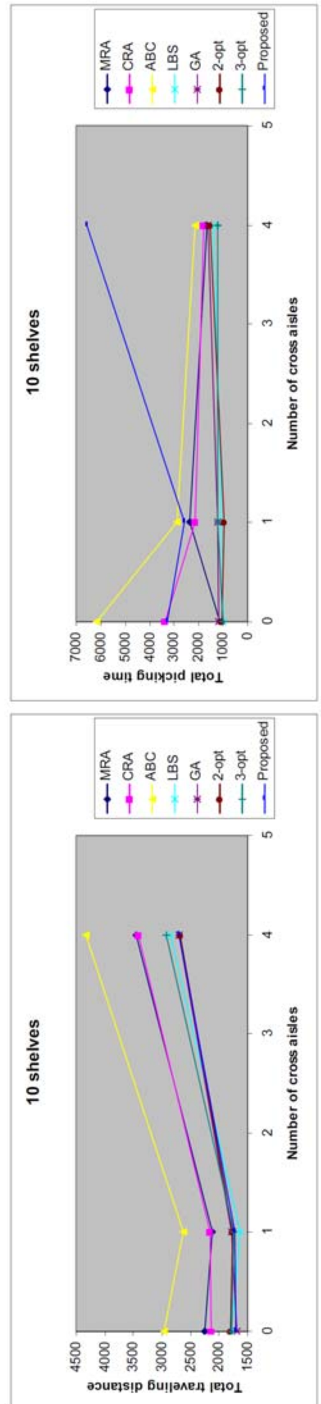

Figure 8. Performances by varying the number of cross aisles (with $P = 90$).

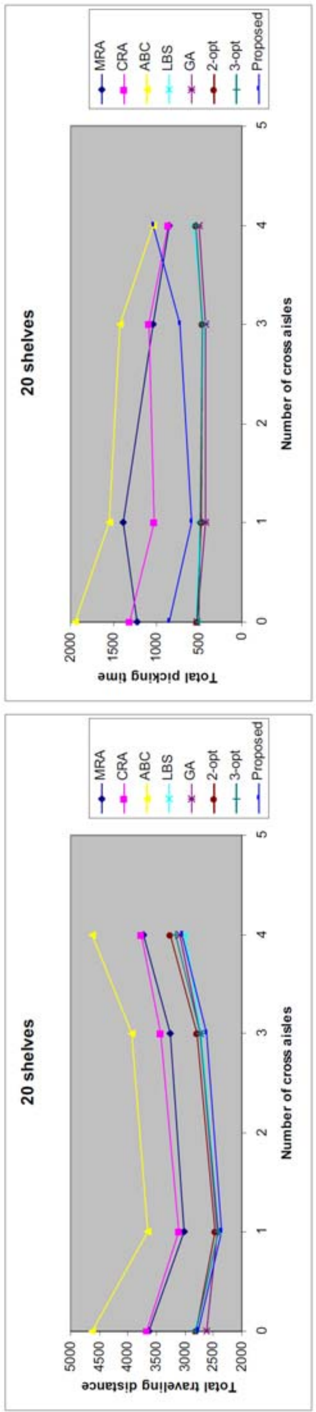

Figure 9. Performances by varying the number of cross aisles (with $P = 90$).

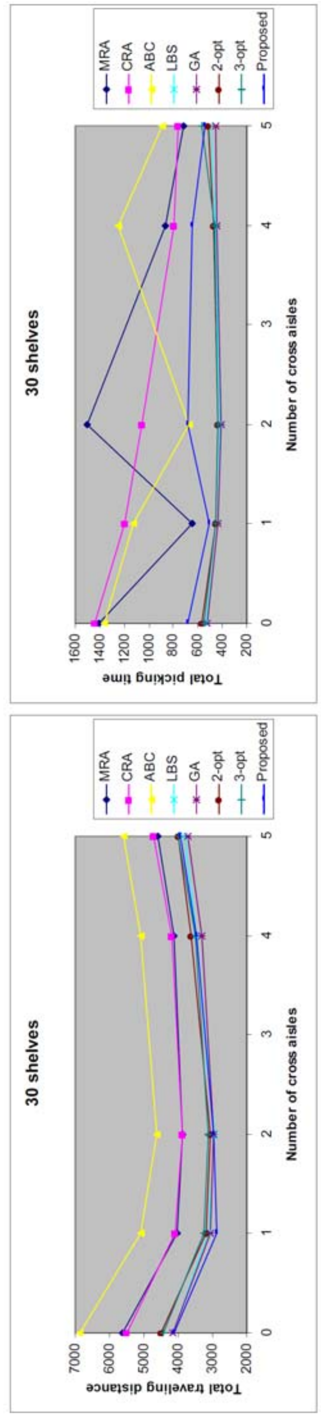

Figure 10. Performances by varying the number of cross aisles (with $P = 90$).

For smaller warehouses, the analyzed heuristics (LBS, GA, 2-opt and 3-opt) demonstrate to be efficient (short CPU time, reduced total traveling distance and reduced total picking time).

On the other hand, the proposed procedure permits to obtain the total traveling distance reduced for large warehouses, and especially, in short CPU time. The total picking time is lower in a smaller warehouse. Rarely, the total traveling distance is desired to be lowest because we notice the efficiency in total picking time worse in comparison with other iterative improvement methods. Due to this situation, we need to evaluate in which n^* (perception of congestion) permits to reduce the total picking time through minimizing the congestion, as shown in Figures 11 and 12 with 1 cross aisle and $P = 90$.

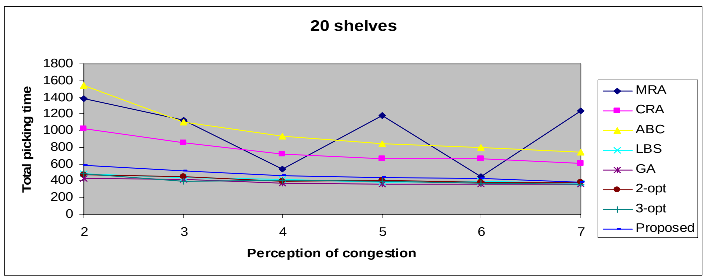

Figure 11. Influence of congestion on the picking time.

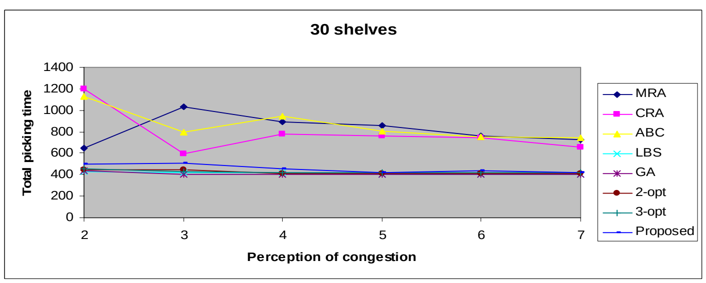

Figure 12. Influence of congestion on the picking time.

Observing Figures 11 and 12, those results were expected, since higher n^* (which means higher number of pickers is permitted to pass through the subaisle at the same time) will yield lower total picking time because of short waiting.

The reason we simulated up to 7, the perception of congestion was the simplicity of simulation. However, it is necessary to remind effectively that increasing the perception of congestion n^* will also demand larger subaisle, which will increment the zigzag travelings (as reported in Caron et al. [2]). Based on the results for the medium and large warehouses, the adopted value for the perception of congestion larger than 3 will already provide, relatively speaking, benefits in terms of reduced picking time.

Therefore, by using the proposed method for larger warehouse size, it is possible to reduce the total picking time, the total traveling distance and CPU time by adopting higher n^*.

5. Conclusions and Future Works

Although the use of iterative improvement methods (LBS, GA, 2-opt and 3-opt) may provide reduced total traveling distance as simulation advances, the results are worse in comparison to the proposed method, except for smaller warehouse. The proposed method shows to be efficient in allocating pieces in drawers in short period of time (1h10min) in comparison to others, which demanded from around 27h30min up to 36h. It is important to note that the consumed CPU time is not affected by the warehouse size.

On the other hand, for large size warehouse, the proposed method permitted to reach lower total traveling distance in shorter CPU time. Due to trade-off behavior, rarely it is desired to minimize the total traveling distance since it implies in increasing of total picking time. Fortunately, the effects of the congestion may be minimized by selecting larger value for the perception of the congestion n^*.

Surprisingly, if the performance parameter is the total picking time, then it is preferable to work with larger warehouse and variety of products than smaller warehouse and variety of products.

The best result in total traveling distance for any size of the warehouse is reached by employing one cross aisle. Although the results indicated great results, it is still not conclusive because other simulation circumstances need to be evaluated. For instance, test other number of sections per shelf, instead of 20; test other picking cart capacity and so forth.

For future works, consider other layouts shape of shelves. For instance, analyze shelves horizontally positioned, instead of vertically positioned as adopted here. Extend the number of products in the picklist and also, evaluate the congestion of picking carts in the cross aisles.

References

[1] A. H. Azadnia, S. Taheri, P. Ghadimi, M. Z. Mat Saman and Kuan Yew Wong, Order batching in warehouses by minimizing total tardiness: a hybrid approach of weighted association rule mining and genetic algorithm, The Scientific World Journal 2013 (2013), 1-13.

[2] F. Caron, G. Marchet and A. Perego, Layout design in manual picking systems: a simulation approach, Integrated Manufacturing Systems 11(2) (2000), 94-104.

[3] Che-Hung Lin and Iuan-Yuan Lu, The procedure of determining the order picking strategies in distribution center, International Journal of Production Economics 60-61(1) (1999), 301-307.

[4] R. De Koster, T. L. Duc and K. J. Roodbergen, Design and control of warehouse order picking: a literature review, European J. Oper. Res. 182(2) (2007), 481-501.

[5] S. Henn, S. Koch, K. F. Doerner, C. Strauss and G. Wascher, Metaheuristics for the order batching problem in manual order picking systems, Business Research 3(1) (2010), 82-105.

[6] M. Kłodawski and J. Żak, Order picking area layout and its impact on the efficiency of order picking process, Journal of Traffic and Logistics Engineering 1(1) (2013), 41-45.

[7] C. H. Mowrey and P. J. Parikh, Mixed-width aisle configurations for order picking in distribution centers, European J. Oper. Res. 232 (2014), 87-97.

[8] K. J. Roodbergen and R. De Koster, Routing order pickers in a warehouse with a middle aisle, European J. Oper. Res. 133(1) (2001), 32-43.

[9] S. Ene and N. Öztürk, Storage location assignment and order picking optimization in the automotive industry, International Journal of Advanced Manufacturing Technology 60 (2012), 787-797.

Appendix 1

For each drawer of $S_{i,\, j}$, we propose the global index (A1.1):

$$\left.\begin{array}{l} Global\_Index_{Si,\, j,\, f'} = \displaystyle\sum_{idx=1}^{IDX=\infty} Index_{idx} \\ Global\_Index_{Si,\, j,\, f'} = 0 \end{array}\right\}, \qquad \text{(A1.1)}$$

where $Index_{idx}$ corresponds to the standardization value.

The number of indexes depends on how variables that possibly affect the total traveling distance interact to each other. Once the variables are analyzed, it is converted to an index. By using (A1.1), we will notice that the process of allocation of pieces occurs, the value of each used index will change, which means the impact on the total traveling distance may or not be significant. Although still unknown since it depends on the status of allocation, each index will have different degree of importance φ. Therefore we adapted it to (A1.2). Note the calculation occurs F' times, since the expression is calculated one time per f':

$$Global\_Index_{Si,\, j,\, f'} = \sum_{idx=1}^{IDX=\infty} \varphi_{idx} \cdot \eta_{idx} \cdot Index_{idx}, \qquad \text{(A1.2)}$$

where

φ_{idx} corresponds to the disturbance factor (weight) for each index, varying from 0 up to 1; per iteration, we assume φ_{idx} equal for all F', independent of the status of allocation, but may be different among "*idx*";

η_{idx} corresponds to the probability of occurrence of demand for product *p*. Note it is only for index 2, because it is the only one which handle with the demand. To simulate it, we consider a probability η%.

$p \in f'$ {particularly important for index 2}.

For illustrating, we raise some variables and how the interactions are converted to indexes.

A1.1. Index 1 - Relation of products of same f'

This first index refers to the relation of proportionality between products (allocated and unallocated) of the same f'.

Assume two families, f_1' and f_2'. The first is composed of $V_{f'1} = 5$, that is, 5 products, 2 of them are already allocated in their best position. And the second is composed of $V_{f'2} = 7$ products, 3 of them are already allocated. Choosing an empty drawer of $S_{i,j}$ closest to the x^* and y^*, a simple calculation permits to decide which one to select. For f_1', if the total number of products is 5, and 3 of them are unallocated (so unallocated away of best position), we have $3/5 = 0.6$. And for the second, $4/7 = 0.57$. We conclude one of 3 remained unallocated products of a total of 5 of f_1' is preferable to be allocated first in comparison to f_2'.

The general expression for the conversion to an index is (A1.3):

$$Index\_1_{Si,j} = \frac{V_{f'} - V_{f'}^*}{V_{f'}} \text{ varying from 0 to 1, where } V_{f'}^* \le V_{f'}, \quad \text{(A1.3)}$$

where $V_{f'}$ represents the total number of products f' and $V_{f'}^*$ set of already allocated products which belongs to f'.

A1.2. Index 2 - Relation between the demand and the number of levels

The second index is about the structure of each product that belongs to the same family. Since each piece type is allocated in different drawer, increasing the number of levels of each family will also require higher number of drawers. So, if there are insufficient drawers in the section, then extra traveling distance will be necessary. However, the traveling distance is also affected by the demand. The conclusion is a product with several levels and high demand which tends to require long trips. We may construct an index (A1.4), in which in one side, a family with reduced number of levels and low demand tends to obtain a value closer to 1, and else 0:

$$Index\_2_{Si,\,j} = \frac{D'_p.(K_p - 1)}{[\max\{D_1, D_2, ..., D_{p^*}\}] \cdot [\max\{K_1, K_2, ..., K_{p^*}\} - 1]}. \quad (A1.4)$$

A1.3. Index 3 - Relation between the chosen section and the ideal position

This index aims to evaluate the distance between the selected section $S_{i,\,j}$, where a piece of f' will be supposed allocated, in relation to the closest ideal position established in Step 3. If the position of section is the closest position, then the rank tends to be closer to 1, and 0 else. Note that highest distance is just the distance between two subsequent modules of the same family. We propose (A1.5):

$$Index\_3_{Si,\,j} = 1 - \frac{|\, x_{Si,\,j} - x^*_{f'} \,| + |\, y_{Si,\,j} - y^*_{f'} \,|}{|\, x_v - x_{v-1} \,| + |\, y_v - y_{v-1} \,|}, \qquad (A1.5)$$

where

$x^*_{f'}$ and $y^*_{f'}$ represent the closest ideal position of a product belonging to f';

$x_{Si,\,j}$ and $y_{Si,\,j}$ represent the Cartesian coordinates of the chosen $S_{i,\,j}$.

The denominator measures the distance between two subsequent modules of f'.

A1.4. Index 4 - Relation between the number of pieces and the number of drawers

This index compares the difference between the number of pieces of a product belonging to f' and the number of remained drawers in the chosen section. When the number of pieces is higher than the number of drawers, it means that extra traveling will be required, so the correspondent family f' will receive a lower score, and vice versa:

If $Dr_{Si,\,j} - \max\left\{\sum_{f^{*'}}^{F^{*'}} \sum_{1}^{K_{f^{*'}}} 1\right\} \geq K_{f'}$, then $Index\_4_{Si,\,j} = 1$,

else, see (A1.6):

$$Index\_4_{Si,\,j} = 1 + \left(\frac{\left(Dr_{Si,\,j} - \max\left\{\sum_{f^{*'}}^{F^{*'}} \sum_{1}^{K_{f^{*'}}} 1\right\} - K_{f'} \right)}{\frac{\left(\sum_{s=1}^{Sc\,.\,Sw} Dr_{Si,\,j,\,s} \right) / (Sc.Sw)}{(X-1)+(Y-1)}} \right). \tag{A1.6}$$

The numerator represents the remained drawers in $S_{i,\,j}$ after the supposed allocation of a product of f'; the upper denominator means that the average number of drawers in each section and the lower denominator represents the maximum trip in horizontal and vertical directions.

Finally, the global index for each f' becomes (A1.7):

$$Global\_Index_{Si,\,j} = \alpha^{*} Index\_1_{Si,\,j} + \beta^{*}\eta^{*} Index\_2_{Si,\,j} + \gamma^{*} Index\_3_{Si,\,j}$$

$$+ \delta^{*} Index\_4_{Si,\,j}. \tag{A1.7}$$

Appendix 2

Proof 1. The constraint (4) also may be written when it is informed the number of blocks *B*, number sections *Sc*, number of shelves *Sw* as input:

We know $cv = \frac{Sw}{B} + 1$, then from the expression (4),

$$X = \left(\frac{Sw}{B} + 1\right) \cdot h + \frac{Sw}{B} + 1 - h.$$

As $h = 2$, which means both sides of shelf, then

$$X = 2\frac{Sw}{B} + \frac{Sw}{B} + 1.$$

Since there are two side vertical aisles, $cv^{*} = 2,$ expression becomes (A2.1):

$$X = 2\frac{Sw}{B} + cv^{*} + \frac{Sw}{B} - 1. \qquad \text{(A2.1)}$$

Proof 2. From the expression (5), v may be expressed in terms of Sc since we know that $v = \frac{Sc}{2}$. So, rewriting (5), Y becomes:

$$Y = (B+1)\cdot\frac{Sc}{2} + B + 1 - \frac{Sc}{2},$$

$$Y = \left(\frac{Sc}{2}\right)\cdot B + B + 1 = \left(\frac{Sc}{2}\right)\cdot B + ch^{*} + B + 1$$

or (A2.2):

$$Y = \left(\frac{Sc}{2}\right)\cdot B + ch^{*} + B + 1, \quad ch^{*} = 2 \qquad \text{(A2.2)}$$

because there are two side horizontal aisles.

Appendix 3

In the constraints, we have already shown expressions (4) and (5) to define the size of the warehouse. However, note that the definition is extremely subjective. We need to use consistent information to develop it. From (9), that is, each drawer must pursue exactly one type of piece:

$$\sum_{i}\sum_{j} Dr_{Si,\,j} = \sum_{f=1}^{F} K_f . V_f .$$

Once all f, that is, F, are determined, the right side of the sentence will be defined. From (33), the right side becomes (A3.1):

$$\sum_{f=1}^{F} K_f \cdot V_f \approx \sum_{f=1}^{F} K_f \cdot \left(\prod_{k_f=1}^{K_f-1} C_{k_f} \right) \approx F \cdot \overline{K} \cdot \overline{V}$$

$$\approx F . \left(\frac{K_{\min} + K_{\max}}{2} \right) . \left(\frac{C_{\min} + C_{\max}}{2} \right)^{\left(\frac{K_{\min}+K_{\max}}{2} \right) - 1} . \quad \text{(A3.1)}$$

Note that in general, $K_{\max}$ and $K_{\min}$ may vary independently in relation to $C_{\max}$ and $C_{\min}$.

When the number of components per product of each *f* is limited in C_f, where $C_f > 0$ and $C_f \leq C_{\max}$, the independency now is undone. We may write:

$$\left(\frac{C_{\min} + C_{\max}}{2} \right)^{\left(\frac{K_{\min}+K_{\max}}{2} \right) - 1} \rightarrow \left(\frac{C_f}{\frac{K_{\min} + K_{\max}}{2} - 1} \right)^{\left(\frac{K_{\min}+K_{\max}}{2} \right) - 1} .$$

Note that $K_{\min}$ and $K_{\max}$ are now varying according to C_f. For illustrating, it is possible to assume the value of $K_{\min}$ as constant. In this case, only $K_{\max}$ will depend on C_f or vice versa. Note also that, if C_f receives a higher value, then $K_{\max}$ will receive a lower or vice versa to keep the above expression as constant. Because of several *f*, we have (A3.2):

$$\left(\frac{\overline{C}}{\frac{K_{\min} + K_{\max}}{2} - 1} \right)^{\left(\frac{K_{\min}+K_{\max}}{2} \right) - 1} . \quad \text{(A3.2)}$$

Due to the paradoxical issue, the intervals for these two averages $\overline{C}$ and $\overline{K}$ may be defined as opposite interval $\{C_{\min} \ldots \overline{C} \ldots C_{\max}\}$ and $\{K_{\max} \ldots \overline{K} \ldots K_{\min}\}$, respectively. Note that $\overline{C}$ is between $C_{\max}$ and $C_{\min}$. It is because if $C_{\min}$ and $C_{\max}$ are used to estimate $\overline{C}$, when

$C_{\max} \geq C_{\min}$, then the lowest value that $\overline{C}$ will assume will be $C_{\min}$ and the highest, $C_{\max}$. The same idea may be applied to $\overline{K}$.

The relation R between these two intervals may be expressed as:

$$R = C_{\min} + K_{\max} \quad \text{or} \quad R = K_{\min} + C_{\max}.$$

Merging both expressions,

$$2R = C_{\max} + C_{\min} + K_{\max} + K_{\min}.$$

Therefore, in general, $\overline{K} + \overline{C} = R.$

However, from (6) and (8), where the values of $K_{\min}$ and $C_{\min}$ are equal to one, we conclude $C_{\max} = K_{\max}$, which fails to the constraint (8). We need to assure $K_{\max} > C_{\max}$ by adding ε (where $\varepsilon > 0$) to make the inequality a true sentence.

Rewriting:

$$R = C_{\min} + K_{\max} \quad \text{or} \quad R = K_{\min} + C_{\max} + \varepsilon.$$

Note only $C_{\max}$ needs to be added. We then have:

$$2R = C_{\max} + C_{\min} + K_{\max} + K_{\min} + \varepsilon,$$

$$\overline{K} + \overline{C^*} = R.$$

The expression (A3.2) becomes:

$$\left(\frac{\overline{C^*}}{\frac{K_{\min} + K_{\max}}{2} - 1}\right)^{\left(\frac{K_{\min} + K_{\max}}{2}\right) - 1}.$$

However, $K_{\max}$ is unknown being dependent on $C_{\max}$. Therefore, we need to write above sentence in function of R:

$$\left(\frac{\overline{C^*}}{R - \overline{C^*} - 1}\right)^{R - \overline{C^*} - 1}.$$

Finally, from the initial expression (A3.1), we obtain (A3.3):

$$\sum_{f=1}^{F} K_f \cdot V_f \approx F \cdot \overline{K} \cdot \overline{V} \approx F.(R - \overline{C^*}) \cdot \left(\frac{\overline{C^*}}{R - \overline{C^*} - 1} \right)^{(R - \overline{C^*} - 1)} . \quad \text{(A3.3)}$$

From the left side of the expression (9):

$$\sum_i \sum_j Dr_{Si,\,j} = \sum_{sw=1}^{Sw} \sum_{sc=1}^{Sc} Dr_{sc,\,sw}.$$

In average, $Dr_{sc,\,sw}$ = constant = $\overline{q}$, $\forall sc$ and $\forall sw$. Therefore,

$$\sum_i \sum_j Dr_{Si,\,j} \approx Sw.Sc.\overline{q}.$$

This expression pursues 3 degrees of freedom, since by keeping these 3 incognitos, the remaining one will be automatically determined. So, keeping following incognito as constant (F, the average number of drawers per section $(\overline{q})$ and the size of the shelf (Sc) as input information), Sw is automatically determined or vice versa.

Although, usually used as input, the question is which Sw we need to keep constant. The answer depends on $\overline{q}$, if we intend to keep other incognitos as constants F and Sc. It is important to note that Sw and $\overline{q}$ are inversely proportional. Assuming a priori pickers may reach a limited number of drawers, we say up to Q', we may write $Dr_{sc,\,sw}$, $\forall sc$ and $\forall sw$ as:

$$Dr_{sc,\,sw} = [Q_o \ldots Q'].$$

The mean drawers for the warehouse may be obtained by:

$$Dr_{sc,\,sw} = \overline{q} = \left(\frac{Q_o + Q'}{2} \right).$$

Since $Dr_{Si,\, j} > 0$, see constraint (10), and assuming that $S_{i,\, j}$ must pursue at least one drawer, we have $Q_o = 1$. Therefore, the previous equation becomes (A3.4):

$$\sum_i \sum_j Dr_{Si,\, j} \approx \frac{Sw \cdot Sc \cdot (1 + Q')}{2}. \tag{A3.4}$$

Appendix 4

Proposition. *If the level of stock satisfies the deterministic demand, then it satisfies also the stochastic demand.*

To prove the cited proposition, let us start the premise that if the demand is known (that is, deterministic), we know also the percentage. We denote Φ_1, Φ_2 and Φ_3 as percentages of demanded *f* of batched orders, where Φ_1 is a highly demanded set of *f* and in the opposite side set, Φ_3; Φ_I, Φ_{II} and Φ_{III} represent the percentages of *f* in stock which belong, respectively, to Φ_1, Φ_2 and Φ_3, where the value of Φ is limited in $0 \le \forall\Phi \le 1$. Since there are V_f products in each *f*, we may represent the required level of stock for each component of V_f as (A4.1):

$$\max\left\{ \cdots \frac{\dfrac{1}{\Phi_I \cdot F} + \cdots + \dfrac{1}{\Phi_I \cdot F}}{\dfrac{1}{V_f}} \cdots ; \cdots \frac{\dfrac{1}{\Phi_{II} \cdot F} + \cdots + \dfrac{1}{\Phi_{II} \cdot F}}{\dfrac{1}{V_f}} \cdots ; \cdots \right.$$

$$\left. \frac{\dfrac{1}{\Phi_{III} \cdot F} + \cdots + \dfrac{1}{\Phi_{III} \cdot F}}{\dfrac{1}{V_f}} \cdots \right\}$$

$$\cdot \max\left\{ \sum^{\infty} \frac{V_f}{\Phi_I . F} ; \sum^{\infty} \frac{V_f}{\Phi_{II} . F} ; \sum^{\infty} \frac{V_f}{\Phi_{III} . F} \right\}. \tag{A4.1}$$

Considering limited batched orders, we have

$$\max\left\{\sum^{\Lambda_1}\frac{V_f}{\Phi_I.F};\sum^{\Lambda_2}\frac{V_f}{\Phi_{II}.F};\sum^{\Lambda_3}\frac{V_f}{\Phi_{III}.F}\right\}$$

$$\cdot\max\left\{\frac{V_f\cdot\Lambda_1}{\Phi_I\cdot F};\frac{V_f\cdot\Lambda_2}{\Phi_{II}\cdot F};\frac{V_f\cdot\Lambda_3}{\Phi_{III}\cdot F}\right\}.$$

But $\Lambda = F\cdot P$:

$$\max\left\{\frac{V_f\cdot\Phi_1\cdot P}{\Phi_I\cdot F};\frac{V_f\cdot\Phi_2\cdot P}{\Phi_{II}\cdot F};\frac{V_f\cdot\Phi_3\cdot P}{\Phi_{III}\cdot F}\right\}.$$

Since $\sum\Phi = 1$, $\Phi_1+\Phi_2+\Phi_3 = 1$ and $\Phi_I+\Phi_{II}+\Phi_{III} = 1$.

Assuming

$$\Phi_1>\Phi_2>\Phi_3,\quad \Phi_1=\Phi_{III},\quad \Phi_2=\Phi_{II},\quad \Phi_3=\Phi_I,$$

we may conclude

$$\frac{V_f\cdot\Phi_1\cdot P}{\Phi_I\cdot F}>\frac{V_f\cdot\Phi_2\cdot P}{\Phi_{II}\cdot F}>\frac{V_f\cdot\Phi_3\cdot P}{\Phi_{III}\cdot F}\ \text{or simply}\ \frac{\Phi_1}{\Phi_I}>\frac{\Phi_2}{\Phi_{II}}>\frac{\Phi_3}{\Phi_{III}}.$$

Finally,

$$\max\left\{\frac{V_f\cdot\Phi_1\cdot P}{\Phi_I\cdot F};\frac{V_f\cdot\Phi_2\cdot P}{\Phi_{II}\cdot F};\frac{V_f\cdot\Phi_3\cdot P}{\Phi_{III}\cdot F}\right\}=\frac{V_f\cdot\Phi_1\cdot P}{\Phi_I\cdot F}. \quad \text{(A4.2)}$$

In stochastic demand, from the expression (A4.1), we obtain:

$$\max\left\{\cdots\frac{V_f}{F}+\cdots+\frac{V_f}{F}\cdots\right\},\ \text{which is simplified to}\ \sum^{\infty}\frac{V_f}{F}.$$

Limiting $\Lambda = 1\cdot P = P$, so:

$$\sum^{\Lambda}\frac{V_f}{F}=\sum^{P}\frac{V_f}{F}=\frac{V_f\cdot P}{F}. \quad \text{(A4.3)}$$

Comparing (A4.2) and (A4.3), when $\Phi_1 > \Phi_I$, we conclude:

$$\frac{V_f \cdot \Phi_1 \cdot P}{\Phi_I \cdot F} > \frac{V_f \cdot P}{F}.$$

That is, if the level of stock for highly demanded component is satisfied, then the stochastic demanded component will be satisfied too.

So the level of stock for each component, $St_{i,j,h}$, is defined as:

$$\frac{V_f \cdot \Phi_1 \cdot P}{\Phi_I \cdot F} = St_{i,j,h}.$$

Note that reducing *P*, the required stock is lower, since both are directly proportional. It means that if the level of stock is satisfied for *P*, this same level will satisfy also a reduced value of *P*. This is particularly important for other simulation conditions when different values of *P* (30 and 60, both lower than 90) were tested. Actually, we adopt (A4.4):

$$\max\left\{\frac{V_f \cdot \Phi_1 \cdot P}{\Phi_I \cdot F}\right\} = St_{i,j,h}. \tag{A4.4}$$