Material handling improvement in warehouses by parts clustering

Mohammad Moshref-Javadi & Mark R. Lehto

To cite this article: Mohammad Moshref-Javadi & Mark R. Lehto (2016) Material handling improvement in warehouses by parts clustering, International Journal of Production Research, 54:14, 4256-4271, DOI: 10.1080/00207543.2016.1140916

To link to this article: http://dx.doi.org/10.1080/00207543.2016.1140916

Published online: 09 Feb 2016.

Submit your article to this journal

Article views: 120

View related articles

View Crossmark data
Material handling improvement in warehouses by parts clustering

Mohammad Moshref-Javadi* and Mark R. Lehto

School of Industrial Engineering, Purdue University, West Lafayette, IN, USA

(Received 28 September 2014; accepted 30 December 2015)

A major part of warehouse operations is related to the collection of parts from the warehouse which is called the Order Picking Problem. To improve order picking operations, the total travel distance and generally picking time must be reduced. In this paper, a two-level approach is proposed that determines the locations of parts in the warehouse. The first step clusters parts into part families. Four different clustering methods based on principal component analysis, singular value decomposition and Two-Step Cluster Component are applied. In the second step, four different heuristics are proposed to determine the locations of parts. In addition to the minimisation of travel distance, we also consider the minimisation of the total congestion in aisles due to multiple workers. The proposed algorithms also consider the interactions between part families to minimise intergroup movements. As a result of the implementation, we achieved more than 40% reduction in material handling compared to the current set-up of the warehouse. The applied algorithms can easily be modified to be used for warehouses with different configurations. The algorithms utilised in this case study can be helpful to researchers to become familiar with new heuristics, as well as practitioners to design improved warehouses.

Keywords: order picking problem; Singular Value Decomposition; Principal Component Analysis; Two-Step Cluster Component; warehouse design; heuristic

1. Introduction

Facility design includes organising the tangible fixed assets, such as material handling systems and machines used in operations, for example, warehousing, production or distribution, such that the utilisation of the resources is improved to achieve the defined goals (Ashayeri, Heuts, and Tammel 2005). The focus of this research is on warehouse activities which involve a variety of movements and operations within warehouses or distribution centres. These operations could encompass receiving, storage, order picking, accumulation, sorting and shipping (Van den Berg 1999). The primary resources used in warehouses are material handling tools, vehicles and workers which are responsible for picking, moving and storing parts, packages, pallets, etc.

Parts are stored in warehouses to be used in subsequent operations inside or outside the facility. Thus, two main steps related to the warehouse operations are: moving parts to the warehouse for storage (parts storing) and moving parts out of the warehouse for production or distribution (order picking). This paper addresses the latter problem; however, the results could also help improving parts storing operations. Among all warehouse activities, 65% are order picking operations (Coyle, Bardi, and Langley 1996).

Several approaches have been used to minimise the total time of collecting parts in warehouses. One approach is determining the best locations of parts in the storage area to minimise the travel distances. This problem is called the Storage Assignment Problem which has been addressed in the literature. Jewkes, Lee, and Vickson (2004) considered storage assignment problem where $nk$ types of products are located in $n$ bins and $k$ shelves. Authors assumed a linear layout and multiple pickers and the objective of the problem was to minimise the order cycle time by obtaining product locations, picker home base location and allocation of a set of products to each picker. A dynamic programming approach was proposed to deal with this problem. Hsieh and Tsai (2006) studied the effects of different factors, for example, storage assignment policy, picking route and density inside multiple cross aisles using simulation. Chan and Chan (2011) proposed class-based storage systems and studied 21 different approaches based on policies combined storage assignment and routing using simulation. Chuang, Lee, and Lai (2012) presented a two-stage approach for storage assignment problem in a single-aisle warehouse. In the first step, parts are grouped together by solving an assignment formulation with the objective of minimising dissimilarities between parts. In the second step, parts are assigned to the locations in the warehouse by solving another assignment problem. More problems associated with order picking

*Corresponding author. Emails: moshref@purdue.edu, mmoshre@gmail.com

© 2016 Informa UK Limited, trading as Taylor & Francis Group
problem are batching parts together for pickup (Ho and Tseng 2006; Hsieh and Huang 2011; Henn 2012; Henn and Wäschler 2012; Kulak, Sahin, and Taner 2012), determining the best route for pickers (Theys et al. 2010; Hsieh and Huang 2011; Kulak, Sahin, and Taner 2012; Pan, Wu, and Chang 2014), minimising blocking and congestion (Hong, Johnson, and Peters 2012; Pan, Shih, and Wu 2012; Chen et al. 2014), and warehouse layout design (Onut, Tuzkaya, and Dogac 2008; Rao and Adil 2013; Öztürkoglu, Gue, and Meller 2014).

In this study, we propose new approaches to deal with order picking and storage assignment problems. The goal is to determine the locations of each part in the warehouse to improve travel distance and consequently order picking time. Four different methods based on principal component analysis (PCA), singular value decomposition (SVD) and Two-Step Cluster Component are used to group parts into part families. To assign the part families to the warehouse, four different storage assignment algorithms are applied and compared. The algorithms also consider groups’ interactions to minimise intergroup travel distances.

The remainder of the paper is organised as follows. Section 2 presents the problem description and assumptions. Section 3 describes four clustering methods and the parts assignment algorithms are provided in Section 4. Section 5 presents the routing subproblem. A case study is explained in Section 6 and the algorithms are implemented and compared on the case. Finally, Section 7 concludes the paper.

2. Problem description

This paper considers the storage assignment problem in warehouses with a focus on improving order picking operations. The examined warehouses are linear with a cross aisle and each part has a specific bin location in the picking (storage) area. When a customer submits an order, a list of parts in the order (picklist) is assigned to a worker to be collected from the storage area. Each picklist represents a batch of parts and it is assumed that all of the parts in a picklist are collected by one worker. After collecting the parts, the worker takes the parts to the packing station (base location) for packing and finally shipment. If the rate of the order arrival is high, it is assumed that orders are kept in the system until a worker becomes available. Although we assume that human resources collect the parts, the results can also be implemented in automated warehouses with automated guided vehicles. The objective is to minimise the workers’ travel time/distance by determining the locations of parts in the warehouse. In addition, we consider congestion in the aisle to reduce potential blockings in the aisles.

To minimise workers’ travel distance and congestion, the defined problem is to determine the locations of parts in the storage area. The main idea is to classify parts which are ordered together more frequently and locate them as close as together in the storage area. As a result, workers travel less distance collecting the parts from the warehouse and average picking time reduces. The total order frequency for each part family is also considered to minimise the travel distance from/to the packing station. In addition, the interactions between part families are also taken into account to place highly related groups as close as together to minimise intergroup travel distances. The results are compared with the current set-up of the system.

3. Parts clustering

In this section, we describe four different approaches to cluster parts into families. The clustering methods are developed based on SVD, PCA and Two-Step Cluster Component.

3.1 Clustering using SVD

SVD is a factorisation of a real or complex matrix. This method has many real-world applications in classifications, signal processing and pattern recognition (Wall, Rechtsteiner, and Rocha 2003; Drineas et al. 2004). SVD decomposes a matrix \( A \) to three matrices: \( U \), \( S \) and \( V \) such that:

\[
A = U \cdot S \cdot V
\]

\( U \), \( S \) and \( V \) are \( m \times m \), \( m \times n \) and \( n \times n \) matrices where \( m \) is the number of parts and \( n \) is the number of picklists.

In order picking problems, the \( A \) matrix is a part-order matrix which is shown in Figure 1. Each X indicates that Part \( Z \) was ordered in order \( Y \).

Two different approaches are applied based on the SVD method (Douglas 2008):

1. Clustering by signs.
2. Clustering by gaps.
3.1.1 Clustering by signs

This method starts with selecting $K$ columns of matrix $U$ with the largest eigenvalues. The eigenvalues are represented by the diagonal values in matrix $S$. Since the numbers in each row of the columns can be either positive or negative, this method leads to at most $2^K$ clusters. For instance, if three columns are selected, the maximum number of clusters is 8. Thus, each part is assigned to a group according to the combinations of signs in the $K$ selected columns. Note that the number of formed families could be less than $2^K$.

3.1.2 Clustering by gap values

Figures 2 and 3 illustrate the process of clustering by the SVD-Gap method. Assuming two columns of matrix $U$, Figure 2 illustrates one step of the algorithm using the first column of $U$. This method decomposes the parts to three clusters according to the gap values. In each step of the algorithm, one column of matrix $U$ is selected. Among all columns of $U$, the columns with greater eigenvalues are selected first. Then, parts are sorted according to the values in the selected column and the gaps between the sorted values are calculated. If the gap between two values is less than a predefined gap ($\alpha$), the parts remain in the same group (Douglas 2008). Thus, in every group the gaps between each value and its adjacent values are less than $\alpha$. If termination criteria are not met, the next column of matrix $U$ is selected and groups are decomposed to smaller groups.
The pre-defined gap ($\alpha$) value is recommended by Douglas (2008) to be $\alpha \in [a, b]$ where:

$$a = \text{AVG} + 1.5 \times \text{(STD)}$$

$$b = \text{AVG} + 2.5 \times \text{(STD)}$$

AVG and STD are the average and standard deviation of the values in the selected column, respectively. We selected $\alpha = \text{Average of the column} + 1.5 \times \text{(Standard Deviation of the column)}$. The hierarchical process is shown in Figure 3. The termination criterion is the number of parts in a group ($\beta$). Therefore, those groups which still contain more than $\beta$ parts are decomposed to smaller groups using columns of matrix $U$. This process continues until all parts are clustered to groups with less than $\beta$ parts.

---

**SVD – Clustering by signs**

1: (Decompose matrix $A$ to matrices $U$, $S$, and $V$
2: Define gap value ($\alpha$)
3: Until termination criteria are met
4: Select an unused column of $U$ with the greatest eigenvalue
5: Sort the values in the selected column
6: Decompose groups to smaller groups according to gap values
7: End Until
8: Return part families

---

### 3.2 Clustering using PCA

PCA is a popular multivariate analysis tool which is used to convert a set of variables to a few linearly uncorrelated variables, called Principal Components. The first principal component accounts for the highest amount of variability in the data. The second variable accounts for the maximum amount of variability in the data after the first component, and so forth. In the following, we use terms factor, axis, group and column of $COR$ interchangeably. The steps of this method are as follows (Albadawi, Bashir, and Chen 2005):

**Step 1:** Input to this method must be in the correlation format. To obtain a parts correlation matrix, we count the number of times that each pair of parts are ordered together to generate a matrix $COR_{m \times m}$. Since the numbers of times that parts are ordered are not equal, the values in the matrix are normalised to correlation values in range $-1$ and $1$.

**Step 2:** Based on the amount of variance that is taken into account, we select the number of eigenvectors, which represent the number of uncorrelated variables. For example, according to our case study, which will be...
explained in Section 6, if 75% of the variance of the data is retained, 88 columns of COR must be selected. We will use different percentage values and compare the results. Note that the number of selected columns of COR represents the maximum number of groups that can be formed.

**Step 3:** One part may be correlated to more than one factor (axis). Thus, we need to rotate the axes such that each part is highly correlated to only one factor. After rotation, an MCOR matrix is obtained.

**Step 4:** Assign the parts to the groups. Part $i$ is assigned to group $j$, if in the $i$th row of matrix $MCOR$, $\max_{k} (MCOR(i,k)) = MCOR(i,j)$. This step is repeated until all of the parts are assigned to a group.

### 3.3 Clustering using Two-Step Cluster Component

The Two-Step Cluster Component (Mooi and Sarstedt 2011) is a clustering method that can handle both categorical and continuous variables. This method is composed of two main steps. The first step clusters the data to some sub-clusters to reduce the size of the matrix of distances between all possible pairs of cases. Every part is read to decide whether it merges with an existing cluster or forms a new cluster. The second step clusters the data to the desired number of clusters using the standard hierarchical procedure. The distances can be calculated by Euclidean or log-likelihood measures. This method is also able to determine the proper number of clusters automatically.

To start the algorithm, the method uses the columns of normalised matrix COR. Each column serves as one variable. The number of columns, similar to the PCA method, depends on the amount of retained variance. Having provided the input matrix, this method continues with steps one and two explained above. We used SPSS to implement the Two-Step Cluster Component method and the details of the method can be found in (SPSS Two-Step Cluster Component 2001).

### 4. Storage Assignment

To locate part families, four different assignment algorithms are used. The algorithms are used in two different scenarios: with consideration of groups’ interactions and without consideration of interactions. The interaction is defined as the movements between part families. The algorithms are implemented on both cases with and without parts clustering and the results are compared.
4.1 Storage assignment algorithms

The storage assignment algorithms are designed based on two steps: (1) select a part family, (2) locate the part family. The four different locating algorithms are shown in Figure 4. The small box on top of each layout represents the packing station. Each column represents a rack and every two columns between two bold lines represent two racks on the sides of an aisle. The darkness of each column represents the priority level; the darker the column, the higher the priority of the rack for assigning parts. Among all racks, darker racks are selected first unless they are full. If two columns have equal priority, the rack with less assigned parts is selected.

Storage Assignment Algorithm (Bipartite, rack_mirror, aisle_mirror)

1: Input part families
2: Input rack length and width, aisle width and cross aisle width
3: Assign priorities to racks //Depending on Bipartite, rack_mirror or aisle_mirror
4: Until all groups are assigned
5: Select the next group //Depending on Interaction or NoIntrcn
6: For all parts in the selected group
7: Select the part with the highest order frequency
8: Select all racks which are not full and have the highest priority
9: Among the selected racks, select one rack that has the least assigned parts
10: Assign the part to the rack as closer as to the packing station
11: Remove the part from the group
12: End for
13: End Until
14: Return parts locations

To begin the assignment process, a part family with the largest average order frequency is chosen. The average order frequency of a part family is defined as the sum of parts’ orders in the family divided by the number of parts in the family. NoIntrcn represents that the interactions between part families are not considered. In this case, the heuristics use the average order frequency of the group to select the next part family. On the other hand, Interaction takes into account the interactions between part families, that is, among the groups that have not been assigned yet, the group with the highest interaction with the last assigned group is selected. As we described before, the interaction is defined as the number of times that a worker goes from one part family to another for collecting parts. Group indicates that the grouping of parts is taken into account, whereas NoGrpg indicates that grouping is not considered; that is, each group has one part.

Storage Assignment Algorithm (zig-zag)

1: Input part families
2: Input rack length and width, aisle width and cross aisle width
3: Select Rack #1 as the Current Rack
4: Until all groups are assigned
5: Select the next group //Depending on Interaction or NoIntrcn
6: If Current_Rack’s space < group’s size
7: Assign as many parts as possible to the Current Rack
8: Current Rack = Rack + 1
9: Assign the rest of the parts of the group to the Current Rack
10: Else If
11: Assign all parts of the group to the current rack
12: End If
13: End Until
14: Return parts locations

The bipartite algorithm first assigns parts to one side of the storage area. Among the racks with the same priority, those racks with less assigned parts are selected. The aisle-mirror algorithm assigns the same priority to both racks in an aisle, while the rack-mirror algorithm assumes higher priority on the rack of the aisle which is closer to the packing area. In addition, the rack-mirror algorithm seeks to fill a rack completely before filling another rack. However, both aisle-mirror and rack-mirror place the parts with higher order frequency as closer as to the packing station. In bipartite, rack mirror and aisle mirror, the assignment of parts starts from the top side of the rack (column) which is closer to the
packing area. The zig-zag algorithm, however, locates parts in an S-shaped format from one of the two corners of the warehouse that are closer to the packing station. In all of the algorithms, when assigning the parts of a family to a rack, parts are first sorted according to their order frequency and those parts with higher order frequency are selected first and placed closer to the packing station.

4.2 Performance measures

4.2.1 Total travel distance

The first performance measure is the total travel distance considering the frequency of orders. This is calculated by the following formula:

\[ \sum_{(i,j)} f_{ij} \cdot c_{ij} \]  

where \( f_{ij} \) is the number of times that parts \( i \) and \( j \) are ordered together and \( c_{ij} \) is the distance between parts \( i \) and \( j \). This measure also includes travels from and to the packing station. The distance between two parts is calculated in three different cases due to the cross aisle which enables pickers to go from one aisle to another. If we assume the length of the first and second parts of the racks are \( A1 \) and \( A2 \), respectively, the width of the aisles is \( L \), and the width of the cross aisle is \( T \), three cases are as follows:

(a) If both parts are in rack section \( A1 \): In order to calculate the travel distance, we define four distance segments. If the distances between part \( i \)'s location in column \( c \) and part \( j \)'s location in column \( v \) from the beginning of a rack are \( X_{1c} \) and \( X_{2v} \), respectively, we have the following (Figure 5):

\[ E1 = X_{1c} \]
\[ E2 = X_{2v} \]
\[ E3 = A1 - X_{1c} \]
\[ E4 = A1 - X_{2v} \]

In our study, the width of the columns is half of the width of the aisle. Thus, the travelled distance is:

\[ c_{ij} = \begin{cases} 
\min\{E1 + E2, E3 + E4\} + |c - v|L, & \text{if } c \neq v \\
|E2 - E1|, & \text{if } c = v 
\end{cases} \]  

(b) If both parts are in rack section \( A2 \): In this case, it is assumed that both parts \( i \) and \( j \) are in section \( A2 \) of the layout. Therefore, we define four distances as follows (Figure 5):

\[ E1 = X_{1c} - A1 - T \]
\[ E2 = X_{2v} - A1 - T \]
\[ E3 = X_{3c} \]

Figure 5. Distance definitions and calculations in three cases (a), (b) and (c).
thus, formula (2) is used to calculate $c_{ij}$.

(c) If one part is in rack section A1 and another one is in rack section A2: it is assumed that one part is in section A1, while another one is in section A2. If we define:

$E_1 = X_{1c}$

$E_2 = X_{2c}$

thus, the distance between two parts is (Figure 5):

$$c_{ij} = |E_2 - E_1| + T + |c - v|L$$

(3)

**4.2.2 Congestion**

Due to multiple workers, if many workers travel in the same aisle, it causes congestion in the aisle and causes a delay due to blocking or picking parts from the same rack. The aim of the congestion analysis was to avoid having many pickers travelling in the same aisle. This analysis considers the total number of times that workers need to go to an aisle to pick the items from the same rack. Thus, we calculated the total times that workers visit a rack and the total standard deviation on all racks is computed; the less the standard deviation, the better the scatter of parts among the racks and consequently less congestion.

**5. Routing problem**

When a worker receives a list of parts for collecting, he needs to decide about the sequence of pickups with the goal of minimisation of the length of travel distance. This problem represents the Travelling Salesman Problem, the well-known routing problem (Applegate et al. 2007). In this problem, a salesperson travels from a depot, visits the cities exactly one time and finally returns to the depot with the objective of minimising the travel distance. We used the following formulation for each picklist to minimise the travel distance (Miller, Tucker, and Zemlin 1960).

$$\text{Min} \sum_i \sum_j c_{ij} \cdot x_{ij}$$

(4)

$$\sum_i x_{ij} = 1 \ \forall j$$

(5)

$$\sum_j x_{ij} = 1 \ \forall i$$

(6)

$$u_i + 1 \leq u_j + n(1 - x_{ij}), \ \forall i = 2, \ldots, n, \ i \neq j, \ j = 2, \ldots, n$$

(7)

$$x_{ij} \in \{0, 1\} \ \forall i, j$$

(8)

$x_{ij}$ is a binary variable equal to 1 if the worker travels from part $j$ location to part $i$ location; otherwise, 0. The objective function (4) minimises the total travel distance. Constraints (5) and (6) guarantee that each part location is visited exactly once and constraints (7) are sub-tour elimination constraints. Finally, constraints (8) specify the types and ranges of the variables.

**6. A case study**

Company X stores and sells more than 1200 types of products to customers. All of the parts, either manufactured or purchased, are stored in the storage area. The storage area has three aisles with one rack on each side – totally six racks – and a cross aisle in between. Company X needs an efficient way to minimise the travel distance/time.

The main necessary data are the list of picklists during the last periods. Each picklist represents a batch of ordered parts with their quantities. Analysing the data of the last eight months shows that 1278 different parts have been ordered and 14,818 picklists have been generated during this period. Each picklist may include several parts ranging from 1 to 25 parts with the average of 6.4 parts per batch.
6.1 Clustering by SVD

The $A$ matrix used in this method is a 1278 ($=m$) by 14,818 ($=n$) matrix and is decomposed to the following three matrices: $U_{1278 \times 1278}$, $S_{1278 \times 14818}$ and $V_{14818 \times 14818}$.

6.1.1 Clustering by signs

Different values were selected for $K$ and Table 1 represents the results. The second and third columns are the maximum possible groups ($2^K$) and the number of formed groups, respectively. The largest formed group’s size is given in the fourth column. The fifth column represents the average number of parts in the formed groups.

According to Douglas (2008), the best value for $K$ is where there is not much decrease in eigenvalues (eigenvalues are the diagonal values in matrix $S$). In our analysis, the best value for $K$ is 7 and the maximum number of clusters that can be formed is $2^7 = 128$.

Using this method, there exist some clusters which contain many parts. For example, there are some clusters which have more than 150 parts. These clusters need to be decomposed to smaller clusters by some methods, such as hierarchical clustering. Furthermore, this method is not efficient enough due to the fact that many clusters are empty while others contain many parts. Another drawback of this method is considering only the signs of the values and the large-ness of them are not considered. For example, two parts with values $+1 \times 10^{-10}$ and $-1 \times 10^{-10}$ form two groups while two parts with values 1 and 100 could fall in the same group.

6.1.2 Clustering by gaps

The SVD-Gap method uses a hierarchical process for parts clustering. To terminate the hierarchical process, the number of parts in all of the groups must be less than a certain number ($\beta$). The SVD-Gap method was applied for three different values of $\beta$. As we stated earlier, every pair of parts has a correlation value. After forming the groups, if two parts are in the same group, the correlation between these two items is satisfied. We calculate the satisfied percentage for every part and finally compute the average percentage of satisfied correlation on all parts. This measure is called the Pearson Correlation measure. The Jaccard measure is also used to compare the methods and is calculated by the following formula:

$$\text{Jaccard} = \frac{a}{b + c - a}$$  \hspace{1cm} (9)

where

$$a = \sum_{g=1}^{C-1} \sum_{g'=g+1}^{C} \sum_{i \in R_g} \sum_{j \in R_{g'}} \text{COR}_{ij}$$  \hspace{1cm} (10)

$$b = \sum_{i \in R_g} \sum_{j=1}^{m} \text{COR}_{ij}$$  \hspace{1cm} (11)

$$c = \sum_{i \in R_{g'}} \sum_{j=1}^{m} \text{COR}_{ij}$$  \hspace{1cm} (12)

In Equations (10)–(12), $C$ is the total number of clusters, $R_g$ is the set of parts in cluster $g$, $m$ is the total number of parts and $\text{COR}_{ij}$ represents the number of times that parts $i$ and $j$ have been ordered in the same picklist. Thus, $a$ represents the number of times that each pair of parts in groups $g$ and $g'$ are ordered together although they are not in the same group.

<table>
<thead>
<tr>
<th>$K$</th>
<th>Max. groups</th>
<th>Number of formed groups</th>
<th>Max. assigned parts to a group</th>
<th>Avg. number of parts per group</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>32</td>
<td>14</td>
<td>321</td>
<td>91.29</td>
</tr>
<tr>
<td>6</td>
<td>64</td>
<td>24</td>
<td>200</td>
<td>53.25</td>
</tr>
<tr>
<td>7</td>
<td>128</td>
<td>41</td>
<td>193</td>
<td>31.17</td>
</tr>
<tr>
<td>8</td>
<td>256</td>
<td>72</td>
<td>158</td>
<td>17.75</td>
</tr>
</tbody>
</table>
same group. In other words, it represents intergroup movements. Also, b and c indicate the total numbers of times that parts in clusters g and g' are ordered, respectively. Thus, Equation (9) represents the total number of times that parts in groups g and g' are ordered together divided by the number of times that all parts in these groups are ordered without intersection. Hence, smaller values of Jaccard measure are better because less intergroup movements are desired.

The results are represented in Table 2. It can be seen that as the value of $\beta$ increases, the number of formed groups decreases. Therefore, the number of parts in each family increases which leads to higher Pearson correlation value.

The SVD-gap method forms many part families, for example, when $\beta = 11$, 513 groups are formed with averagely 2.49 parts in each group. Also, the number of groups with only one part is significant. When $\beta = 11$, for instance, the minimum number of groups was $\frac{1278}{11} = 117$; however, 513 groups have been formed. This method is not efficient in breaking a group into smaller groups if the gaps between each pair of consecutive parts are almost equal.

### 6.2 Clustering using PCA

The PCA method was implemented for different percentage values of retained variance (Table 3). The Retained Variance is calculated as the sum of all considered eigenvalues of matrix $\text{COR}$ divided by the sum of all eigenvalues. The number of formed groups is equal to the number of columns of matrix $\text{COR}$. The results are given in Table 3. As this method considers more variance, more groups are formed and the average Pearson correlation reduces accordingly. The Pearson correlation value by the PCA method is higher compared to the SVD-Gap method, for example, when comparing with the SVD-Gap method with $\beta = 51$. The Jaccard value is also much better compared to the SVD-gap method with $\beta = 51$.

### 6.3 Clustering using Two-Step Cluster Component

The number of variables used for clustering depends on the amount of retained variance. Thus, different values of retained variance was used and the number of formed groups was set equal to the number of variables; therefore, the results can be compared with the PCA method. Also, we used the Two-Step method to automatically determine the number of clusters. However, it obtained a few clusters with many parts that make the comparison difficult and storage assignment stage inefficient. The results in Table 4 show that both Pearson correlation and Jaccard values are slightly worse than the PCA's.

<table>
<thead>
<tr>
<th>$\beta$ (Max. number of parts per group)</th>
<th>Number of formed groups</th>
<th>Max. assigned parts to a group</th>
<th>Avg. number of parts per group</th>
<th>Pearson correlation measure</th>
<th>Jaccard measure</th>
</tr>
</thead>
<tbody>
<tr>
<td>11</td>
<td>513</td>
<td>11</td>
<td>2.49</td>
<td>3.08</td>
<td>3.10</td>
</tr>
<tr>
<td>31</td>
<td>295</td>
<td>31</td>
<td>4.33</td>
<td>10.55</td>
<td>3.64</td>
</tr>
<tr>
<td>51</td>
<td>212</td>
<td>51</td>
<td>6.02</td>
<td>16.69</td>
<td>3.95</td>
</tr>
</tbody>
</table>

Table 2. Clustering results using the SVD-Gap method for different values of $\beta$.

<table>
<thead>
<tr>
<th>Retained variance</th>
<th>Number of formed groups</th>
<th>Max. assigned parts per group</th>
<th>Avg. number of parts per group</th>
<th>Pearson correlation measure</th>
<th>Jaccard measure</th>
</tr>
</thead>
<tbody>
<tr>
<td>70%</td>
<td>62</td>
<td>57</td>
<td>20.61</td>
<td>43.68</td>
<td>2.19</td>
</tr>
<tr>
<td>75%</td>
<td>88</td>
<td>53</td>
<td>14.52</td>
<td>37.49</td>
<td>2.13</td>
</tr>
<tr>
<td>80%</td>
<td>128</td>
<td>54</td>
<td>9.98</td>
<td>31.55</td>
<td>1.97</td>
</tr>
</tbody>
</table>

Table 3. Clustering results using the PCA method for different values of retained variance.

<table>
<thead>
<tr>
<th>Retained variance</th>
<th>Number of formed groups</th>
<th>Max. assigned parts per group</th>
<th>Avg. number of parts per group</th>
<th>Pearson correlation measure</th>
<th>Jaccard measure</th>
</tr>
</thead>
<tbody>
<tr>
<td>70%</td>
<td>62</td>
<td>73</td>
<td>20.61</td>
<td>39.43</td>
<td>2.44</td>
</tr>
<tr>
<td>75%</td>
<td>88</td>
<td>58</td>
<td>14.52</td>
<td>34.56</td>
<td>2.40</td>
</tr>
<tr>
<td>80%</td>
<td>128</td>
<td>55</td>
<td>9.98</td>
<td>29.14</td>
<td>2.16</td>
</tr>
</tbody>
</table>

Table 4. Clustering results using the Two-Step Cluster Component method for different values of retained variance.
To fairly compare the methods, the number of groups was set to the number of groups obtained by the SVD-Gap method. Table 5 compares the methods with respect to the Pearson correlation and Jaccard measures. The results show that the PCA method performs better than Two-Step and SVD-Gap methods.

Since the PCA results are robust and the method is more efficient, we use the results of the PCA method with 80% retained variance as the input to the storage assignment phase.

We also used the mathematical model in Chuang, Lee, and Lai (2012) to solve smaller size problems with 50, 100, 150 and 200 parts for 10,800 s. GAMS 24.2.2 was used on a computer with 2.7 GHz CPU and 8 GB RAM. The results are given in Table 6. Math. Model represents the solution by the mathematical model. The PCA algorithm obtains better results than the mathematical model, except the Pearson correlation value on 50-part problem (with small %gap). The results indicate that the PCA method can achieve promising results in reasonable time (less than 5 s) compared to the mathematical model in 10,800 s.

6.4 Storage assignment results

6.4.1 Comparison of the heuristics

The proposed heuristics in Section 4 were used to assign parts to the storage area. The results are given in Table 7 and Figure 6. The comparison indicates that the heuristics that consider parts grouping lead to less travel distance (Heuristics Table 5. SVD-gap, PCA, and Two-Step comparison.

<table>
<thead>
<tr>
<th>Number of groups</th>
<th>Pearson correlation measure</th>
<th>Jaccard measure</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>SVD-gap PCA Two-Step</td>
<td>SVD-gap PCA Two-Step</td>
</tr>
<tr>
<td>513</td>
<td>3.08 8.40 6.71</td>
<td>3.10 1.21 1.73</td>
</tr>
<tr>
<td>295</td>
<td>10.55 16.83 13.58</td>
<td>3.64 1.52 1.97</td>
</tr>
<tr>
<td>212</td>
<td>16.69 23.76 19.50</td>
<td>3.95 1.72 2.12</td>
</tr>
</tbody>
</table>


<table>
<thead>
<tr>
<th>Number of parts</th>
<th>Pearson correlation measure</th>
<th>Jaccard measure</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Math. Model PCA %Gap</td>
<td>Math. Model PCA %Gap</td>
</tr>
<tr>
<td>50</td>
<td>76.35 72.68 4.80</td>
<td>0.45 0.34 -23.84</td>
</tr>
<tr>
<td>100</td>
<td>53.75 54.97 -2.27</td>
<td>0.87 0.84 -2.62</td>
</tr>
<tr>
<td>150</td>
<td>14.74 51.38 -248.62</td>
<td>1.42 0.94 -33.65</td>
</tr>
<tr>
<td>200</td>
<td>12.22 50.09 -309.99</td>
<td>1.59 1.07 -32.01</td>
</tr>
</tbody>
</table>

Table 7. Comparison of storage assignment heuristics.

<table>
<thead>
<tr>
<th>Heuristic</th>
<th>Total distance (ft.)</th>
<th>Rank</th>
<th>Congestion</th>
<th>Rank</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 NoIntrcn. – Group – (bipartite)</td>
<td>12,749,475.0</td>
<td>5</td>
<td>14,832.9</td>
<td>5</td>
</tr>
<tr>
<td>2 NoIntrcn. – Group – (rack-mirror)</td>
<td>12,968,467.8</td>
<td>6</td>
<td>24,937.1</td>
<td>12</td>
</tr>
<tr>
<td>3 NoIntrcn. – Group – (aisle-mirror)</td>
<td>13,360,914.9</td>
<td>7</td>
<td>21,553.2</td>
<td>11</td>
</tr>
<tr>
<td>4 NoIntrcn. – Group – (zigzag)</td>
<td>13,822,966.3</td>
<td>11</td>
<td>25,595.5</td>
<td>13</td>
</tr>
<tr>
<td>5 Interaction – Group – (bipartite)</td>
<td>12,683,149.3</td>
<td>4</td>
<td>14,207.0</td>
<td>2</td>
</tr>
<tr>
<td>6 Interaction – Group – (rack-mirror)</td>
<td>12,196,844.4</td>
<td>2</td>
<td>16,838.1</td>
<td>8</td>
</tr>
<tr>
<td>7 Interaction – Group – (aisle-mirror)</td>
<td>13,689,863.8</td>
<td>10</td>
<td>14,552.1</td>
<td>4</td>
</tr>
<tr>
<td>8 Interaction – Group – (zigzag)</td>
<td>11,870,082.5</td>
<td>1</td>
<td>17,442.3</td>
<td>9</td>
</tr>
<tr>
<td>9 NoIntrcn. – NoGrpg – (bipartite)</td>
<td>13,594,780.1</td>
<td>9</td>
<td>15,906.7</td>
<td>6</td>
</tr>
<tr>
<td>10 NoIntrcn. – NoGrpg – (rack-mirror)</td>
<td>13,478,346.4</td>
<td>8</td>
<td>32,019.3</td>
<td>15</td>
</tr>
<tr>
<td>11 NoIntrcn. – NoGrpg – (aisle-mirror)</td>
<td>14,295,540.2</td>
<td>14</td>
<td>26,515.7</td>
<td>14</td>
</tr>
<tr>
<td>12 NoIntrcn. – NoGrpg – (zigzag)</td>
<td>13,827,146.1</td>
<td>12</td>
<td>32,251.5</td>
<td>16</td>
</tr>
<tr>
<td>13 Interaction – NoGrpg – (bipartite)</td>
<td>13,849,449.4</td>
<td>16</td>
<td>14,402.0</td>
<td>3</td>
</tr>
<tr>
<td>14 Interaction – NoGrpg – (rack-mirror)</td>
<td>15,261,620.2</td>
<td>15</td>
<td>16,224.5</td>
<td>7</td>
</tr>
<tr>
<td>15 Interaction – NoGrpg – (aisle-mirror)</td>
<td>14,168,472.5</td>
<td>13</td>
<td>14,206.9</td>
<td>1</td>
</tr>
<tr>
<td>16 Interaction – NoGrpg – (zigzag)</td>
<td>12,244,434.4</td>
<td>3</td>
<td>18,452.9</td>
<td>10</td>
</tr>
</tbody>
</table>
Figure 6. Illustration of the results by different heuristics.
#1–#8) in comparison with the heuristics without grouping of parts (Heuristics #9–#16). This fact is also confirmed in Table 8 in which bold values represent algorithms with better performance measure values. This table also shows that considering interaction between groups yields less average total distance. Among all of the parts locating algorithms, the zigzag algorithm obtains the best results.

According to Table 7, Heuristic #8 which uses grouping, interaction and zigzag algorithm yields the least total distance. The only heuristic which obtains acceptable total distance without grouping of parts is heuristic #16 which uses the zigzag method. With respect to the congestion measure, considering grouping, as well as interaction leads to less average congestion. Among the four storage assignment algorithms, the bi-partite algorithm obtains the least congestion value.

To decide which heuristic should be implemented according to the two performance measures, we use the weighted ranking of the heuristics. Weights $w_1$ and $w_2$ are assigned to the total distance and congestion measures, respectively. The results are illustrated in Figure 7. In four cases ($w_1 = 0.1$, $w_2 = 0.9$; $w_1 = 0.3$, $w_2 = 0.7$; and $w_1 = 0.5$, $w_2 = 0.5$; and $w_1 = 0.7$, $w_2 = 0.3$), heuristic 5 is selected, while in two cases ($w_1 = 0.7$, $w_2 = 0.3$ and $w_1 = 0.9$, $w_2 = 0.1$) heuristic #8 is chosen; using ($w_1 = 0.7$, $w_2 = 0.3$), both heuristics 5 and 8 are selected. It should be noted that both of heuristics #5 and #8 consider grouping and interactions of groups. The only difference is the type of algorithm for locating parts in the layout, i.e. in the first four sets of weights that heuristic #5 is selected, part locating algorithm is bi-partite, whereas zigzag algorithm is selected in the two latter weight sets.

Contrary to the approach used in Chuang, Lee, and Lai (2012), we do not fix the allocated space to each group. Therefore, all of the presented algorithms are applicable to any group sizes. Allocating a pre-determined space for groups would make inexact results since different groups have different sizes. In addition, the distance calculation in our algorithms is exact, while Chuang, Lee, and Lai (2012) calculate the distance to the mid-point of a group location. The proposed methods can be used for any group size since interaction between part families and intergroup travel distances are also taken into account. They can also be applied to warehouses with any numbers of main and cross aisles.

### Table 8. Comparison of the average results.

<table>
<thead>
<tr>
<th>Heuristic</th>
<th>Average total distance</th>
<th>Average congestion</th>
</tr>
</thead>
<tbody>
<tr>
<td>Grouping</td>
<td>12,917,720.52</td>
<td>18,744.76</td>
</tr>
<tr>
<td>No Grouping</td>
<td>14,089,979.28</td>
<td>21,247.44</td>
</tr>
<tr>
<td>No Interaction</td>
<td>13,512,204.62</td>
<td>24,201.47</td>
</tr>
<tr>
<td>Interaction</td>
<td><strong>13,495,495.18</strong></td>
<td><strong>15,790.73</strong></td>
</tr>
<tr>
<td>Bi-partite</td>
<td>13,719,224.71</td>
<td><strong>14,837.15</strong></td>
</tr>
<tr>
<td>Aisle-mirror</td>
<td>13,476,319.71</td>
<td>22,504.74</td>
</tr>
<tr>
<td>Rack-mirror</td>
<td>13,878,697.85</td>
<td>19,206.99</td>
</tr>
<tr>
<td>Zigzag</td>
<td><strong>12,941,157.35</strong></td>
<td>23,435.53</td>
</tr>
</tbody>
</table>

Figure 7. Weighted ranking of the heuristics.
6.4.2 Comparison with randomised algorithms
Randomised algorithms were used to obtain random solutions for comparison with the proposed heuristics. The randomised algorithms select the groups randomly and use the storage assignment algorithms to assign them to the storage area. The obtained results are the average of 12 randomly generated solutions. The average total travel distance of the algorithms is 20,470,680.24 and the average congestion is 12,907.2. Random generated solutions have very high transportations, while the average congestion is low. The low congestion value is due to the fact that parts are randomly distributed in all racks.

6.4.3 Comparison with the current layout
Using the data provided by the company about the current locations of parts in the warehouse, we compare the results of the heuristics with the current layout. We simulated the current system using the provided data (picklists) of the last eight months. According to the current locations of the parts in the storage area, the total travel distance is 21,451,467.7. By comparing with the best proposed heuristics, Heuristic#8 and Heuristic#5, 44.66% and 40.88% reduction in the total travel distance was achieved, respectively.

7. Conclusions
This paper proposed new approaches to deal with the order picking problem in warehouses by assigning parts to the storage area. To reduce material handling in warehouses, parts were grouped into part families and located in the storage area. Four different clustering approaches based on SVD, PCA and Two-Step Cluster Component were utilised and compared. Additionally, several different storage assignment algorithms were implemented. The algorithms also considered minimising intergroup movements. The results showed that considering grouping of parts and intergroup movements lead to less average travel distance. In addition to material handling, we also introduced a simple and effective measure to minimise congestion in the aisles. The results showed more than 40% reduction in material handling comparing to the current set-up of the system.

The proposed approaches in this paper can initiate new methods for warehouse operations improvement and some extensions to these methods are discussed here:

- One future research is to examine the effects of high turn-over rate, as well as demand trends on the results. A robust approach that leads to less reconfiguration of the layout of such facilities would be a practical extension to this paper.
- We used four different clustering algorithms in this paper. Another direction for future research is to improve the performance of some of the algorithms, such as SVD-Sign and SVD-Gap methods. More clustering algorithms, for example, Latent Class Clustering can also be applied and compared.
- One can analyse the effects of more precise congestion measures on the total collecting time of parts. Congestion could be examined for the cases when a forklift blocks an aisle or when multiple workers take some parts from the same rack.
- The results of the paper are valid for the warehouses with one shelf or multiple handy shelves. However, one can study the warehouses with multiple shelves with significant vertical distances. The storage assignment algorithms may assign those parts with less turn-over rate to the top-level shelves.
- We considered that all parts in a picklist are collected by one worker. Another approach is to define a region in the warehouse for every worker and each worker is responsible for only collecting parts from the assigned region for any picklist. The results could be compared with the results of this paper.

Acknowledgement
We would like to thank the anonymous reviewers for their valuable comments.

Disclosure statement
No potential conflict of interest was reported by the authors.
Funding
This work was supported by the Indiana Next Generation Manufacturing Competitiveness Center as a part of Technical Assistant Program.

References