Proceedings of the 2020 Winter Simulation Conference
K.-H. Bae, B. Feng, S. Kim, S. Lazarova-Molnar, Z. Zheng, T. Roeder, and R. Thiesing, eds.

GRAVITY CLUSTERING: A CORRELATED STORAGE LOCATION ASSIGNMENT PROBLEM APPROACH

Mohammadnaser Ansari
Jeffrey S. Smith

Industrial & Systems Engineering Department
Auburn University
Auburn, AL 36849, USA

ABSTRACT

Warehouses and warehouse-related operations have long been a field of interest for researchers. One of the areas that researchers focus on is the Storage Location Assigning Problem (SLAP or Slotting). The goal in this field is to find the best location in a warehouse to store the products. With the current COVID-19 pandemic, there is a shopping paradigm shift towards e-commerce, which even after the pandemic will not return to the old state. This paradigm shift raises the need for better performing multi-pick warehouses. In this paper, we propose a clustering method based on the gravity model. We show that for warehouses in which there is more than one pick per trip, our proposed method improves the performance.

1 INTRODUCTION

Order picking is one of the most labor-intensive and expensive processes in any warehouse. De Koster et al. (2007) estimated the order-picking cost as high as 55 percent of the total costs of warehouse operations. It is natural to try to reduce the cost associated with order-picking processes. Van Gils et al. (2018) developed a full factorial ANOVA model and proved that the processes of storage, batching, zoning, and routing as well as their two-way and three-way interactions have significant effects on average travel distance. Considering that storage is a significant factor in determining the total travel distance and, hence, the total cost associated with the order-picking process, there have been numerous research efforts on this subject. Heskett (1963) introduced Cube-per-Order Index (COI), and Kallina and Lynn (1976) proved its optimality in distribution warehouses under the assumption of forward-reserve warehouse structure and single item per trip order-picking policy.

Later, Frazele and Sharp (1989) developed the concept of the correlated assignment in warehouses in which an order-picker is assigned to pick multiple items in one trip. The idea is simple; stock-keeping units (SKUs) that are ordered together should be stored together as well. This concept resulted in various methods and research to extract the correlated items and store them together to optimize the order-picking storage for a multi-item per trip order-picking policy.

Bindi et al. (2007) and Bindi et al. (2009) introduced a new similarity index to create clusters of correlated SKUs based on it. Xiao and Zheng (2010) and Xiao and Zheng (2012) modeled a production warehouse in which the BOM is known beforehand. Then, the authors proposed a heuristic model and genetic algorithm, respectively, to solve the correlated SLAP in the mentioned warehouse. Kim and Smith (2012) developed a mixed-integer programming model in order to solve the correlated SLAP in a zone-based warehouse in which each aisle has its dedicated picker. Chuang et al. (2012) define a mathematical model to solve the correlated SLAP in a single-aisle warehouse. It first clusters the SKUs into K different clusters. K can be any number between one and the total number of SKUs. Then, it finds the K that results in the lowest travel distance to pick all items. Zhang (2016) proposed heuristic methods to solve the correlated SLAP. The proposed methods for solving the correlated storage assignment strategy (CSAS) are named

Sum-Seed and Static-Seed. In both methods, using the similarity matrix, the author starts from a highly correlated SKU, X, and creates a cluster from the SKUs correlated to it.

All of the previously mentioned models are limited in one or more of these aspects:

- The methods were tested only in single-block warehouse
- Significant assumptions were made regarding the warehouse operations (e.g., a single-aisle warehouse)
- The methods were not tested in conjunction with realistic routing methods such as S-Shaped.

We propose a new clustering approach based on the gravity model in Section 2. We later test the method with different warehouse designs (single- and multi-block warehouses) and different routing methods, (S-Shaped, largest-gap, random).

2 GRAVITY CLUSTERING METHOD FOR SOLVING SLAP

Sir Isaac Newton formulated the gravitational force between two objects as:

$$F = G\frac{m_1 m_2}{r^2}$$

where F is the force, G is the gravitational constant, m represents the mass of objects, and r represents the distance between the two objects.

This formula later inspired researchers in fields other than physics. For example, in international economics, the bilateral trade flow between two countries is calculated with the same formula in which m represents the economic dimension of each country and r is the distance between their economic capitals (Bergstrand 1985). Figure 1 is an example that illustrates the gravity model in international trades of UK-EU and UK-India. As another example, in human geography, the amount of human interaction between two cities is calculated based on the gravity model, in which the mass is the population of each city (Wikibooks. 2019).

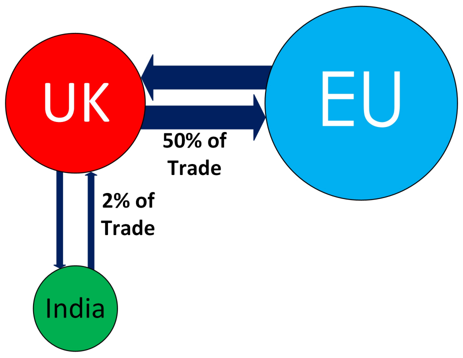


Figure 1: Implication of gravity model in economics.

As mentioned in Section 1, one method to solve the correlated SLAP is to cluster the SKUs that are ordered together, and assign them next to each other. We propose a clustering method for SKUs based on the gravity formula. For a known set of order data, the order frequency of each SKU will represent its mass, and one over relative support of their rule will represent their distance.

$$F_{ij} = \frac{Support_{\{i\}} Support_{\{j\}}}{\left(\frac{N}{Support_{\{i,j\}}}\right)^2}$$

where

- F_{ij} represents the attraction between SKU i and j
- $Support_{\{i\}}$ is the number of orders containing SKU i
- $Support_{\{j\}}$ is the number of orders containing SKU j
- $Support_{\{i,j\}}$ is the number of orders containing both SKUs i and j
- and N is the total number of orders

Calculating the $Support_{\{i\}}$ and $Support_{\{j\}}$ in the aforementioned formula is not complicated. However, calculating $Support_{\{i,j\}}$ where $i,j \in K$ will need a triangular $K \times K$ matrix. For large K values, calculating this matrix requires significant computational time and power, if it is even possible. To overcome this, we utilized *Market Basket Analysis* concepts and tools (Agrawal et al. 1993). The first step in analyzing the market basket data is to mine the *Association Rules*. Most aspects of *Association Rule Mining* are outside the scope of this research, but the part we are interested in is calculating the support values for itemsets $\{i,j\}$.

$$Support_{\{i,j\}} = Support(X+Y)\text{; times that SKUs } X \text{ and } Y \text{ appeared in the same order where } X \text{ and } Y \text{ were two distinct SKUs.}$$

By using the available algorithms for Association Rule Mining such as *Apriori* and its faster successor *FP-Growth Apriori*, and limiting the itemset size to two, we will have the set of $Support_{\{i,j\}}$ values where $i,j \in K$. However, as Ansari et al. (2018) described, the overall process of Association Rule Mining will result in a loss of data. This data loss means that the extracted rule set will not incorporate all $i,j \in K$. However, the data loss is a result of ignoring non-significant rules (a.k.a small support values) and will not have any effect on *Gravity Clustering* calculations.

Table 1 provides the notation used in this paper:

Table 1: Table of notations.

$S_{\{i\}}$	Support value of i, where $i \in K$
$S_{\{ij\}}$	Support value of itemset $\{i,j\}$, where $i,j \in K$
t	Minimum cluster threshold
K	SKU Matrix ($K = 1,...,k$)
S	$k \times 2$ Support Matrix
O	Order Matrix
M	$m \times 2$ Relative Support Matrix where m is the total number of mined rules

Then, we take the following steps to cluster the SKUs:

1. Calculate the *support* value for every SKU, S
2. Store the mined association rules in matrix M
3. Sort S in descending order
4. Select the first row of S, SKU i
5. Set SKU i as the core SKU of the new cluster
6. Loop through every SKU, K
7. For every SKU j, calculate F_{ij}, the attraction between SKU i, and SKU j, based on S_i, S_j, and $M_{\{ij\}}$
8. For all SKUs K, if F_{ij} is larger than the predefined threshold t, add them to the current cluster
9. Remove all current cluster SKUs from the support matrix S
10. Return to Step 3 until there is no SKU left

Algorithm 1 illustrates the programming approach in developing clusters based on the Gravity Clustering method.

Algorithm 1: Gravity Clustering pseudo code.

```
Count the frequency of appearance by each SKU $\in K$ and add it to $S$;
Mine the rules with itemset size of 2 from $O$;
Write the results in $M$;
Sort($S$) in decreasing order based on support value;
while $S.Size() > 0$ do
    Choose the first SKU in $S$, $i$;
    Create a new cluster with $i$ as the core SKU;
    Remove $i$ from $S$;
    for every $j \in K$ do
        Calculate $F_{i,j}$ based on $M_{\{i,j\}}$, $S_{\{i\}}$, and $S_{\{j\}}$;
        if $F_{i,j} > t$ then
            Add SKU $j$ to current cluster;
            Remove $j$ from $S$;
        end
    end
    Sort current cluster based on SKU frequency;
end
```

After creating clusters, they are sorted based on their Cube-per-Order Index (COI) in descending order. Also, in each cluster, SKUs are sorted based on the frequency ($S_{\{i\}}$) value. Clusters and SKUs in each cluster are assigned to the sorted list of the best locations in that order.

It is worth mentioning that the threshold value acts in the same way as a galaxy's centrifugal force in astrophysics. As the centrifugal force prevents a galaxy from collapsing into a single object, the threshold prevents forming a single cluster of all SKUs in our Gravity Clustering method.

Our proposed method incorporates both the frequency of SKUs as well as their joint occurrences in creating the clusters. This means that based on the gravity model, two SKUs with low support values need to be extremely close to each other to be grouped in the same cluster. We expect our model to be an improvement over the current correlated SLAP methods.

3 SIMULATION MODEL

To test the proposed method, we developed a simulation model using the AnyLogic® software (Grigoryev 2012). Although using programming languages such as Python™ in modeling warehouse operations (as done by Ansari and Smith 2017) might result in a faster modeling and replication process compared to multi-purpose simulation software such as AnyLogic®, considering the following benefits of using a simulation software, we decided to use AnyLogic® in this research.

- Easier validation and verification process: The graphic representation of steps makes it easier to validate various methods coded in the simulation.
- GUI representation of simulation parameters: The developed model is highly flexible, which means that the user inputs many design parameters.
- Predefined experiments such as *Parameter Variation*: AnyLogic® is equipped with various predefined experiments such as parameter variation and optimization, which are helpful in testing effects of using multiple designs, SLAP, batching, and routing methods.

In this section, the input data used in our simulation model are explained in Section 3.1. Then, we discuss the configuration of simulation in Section 3.2. In Section 3.3, we explain the verification process in more detail, and in the last section we describe the simulation experiment.

3.1 Input Data

The data we used for our experiment consist of a set of open-source retail data provided by the Instacart retail company (Instacart 2017). Table 2 represents a preliminary analysis of data.

Table 2: Instacart retail data.

SKUs	49,688 SKUs
Orders	3,214,490 orders
Line items	32,434,490 items ordered
Extracted rules	56,810 pairs of itemsets with support value bigger than 320 (equivalent of 0.01 % of the total number of orders)

As can be seen in Table 2, the total number of pairs with a support value of 0.0001 shows the importance of using Association Rule Mining instead of a $K \times K$ similarity matrix. The resulting similarity matrix would have been a triangular matrix with a whopping 1 *billion* values, which is computationally time-consuming (if not impossible) by an above-average workstation. Table 3 gives a small sample of the 57,000 extracted rules.

Table 3: Sample of extracted rules.

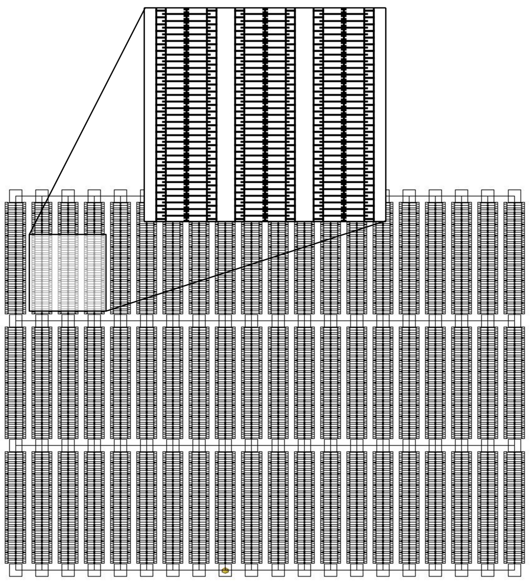
Itemset	Relative support value
[142,630]	0.00424
[84,177]	0.00412
[125,6202]	0.00016
[31,52]	0.01939
[386,35423]	0.00388
...	...

3.2 Simulation Parameters

The simulation model is highly flexible and user-friendly. The simulation is capable of modeling a warehouse, configurable in width, length, number of aisles, cross-aisles, and levels. Figure 2a illustrates a warehouse with a width of 800, length of 600, 4 cross-aisles (including the first and last cross-aisles), 20 aisles, 50 locations per each side of an aisle, and nine levels for each location. The same set of warehouse parameter values is used in this paper to analyze the effect of the Gravity Clustering method (Section 4.1) and the gravity threshold value (Section 4.2) in model performance. Also, other SLAP methods, including *Random* and *COI*, were coded in the simulation model for performance comparison. The methods can simply be chosen at the beginning of the simulation. Figure 2b is a screenshot of the parameter configuration tab in our simulation model.

3.3 Model Verification

The developed simulation model consists of various SLAP and routing methods. We started with verifying the most straightforward configuration of the model, and step by step validated each implemented method.



(a)

Number of Aisles	20
Number of Cross-Aisles	4
Warehouse Length	600
Warehouse Width	800
Locations in Aisle	50
Levels in Each Location	9
Number of SKUs	49,688
Number of Orders	10,000
SLAP Method	Random Freq COI Gravity Clustering
Gravity Threshold	100
Sampling Orders	True
Large Data Size	3,200,000
Random Generator	Fixed Seed Random Seed

(b)

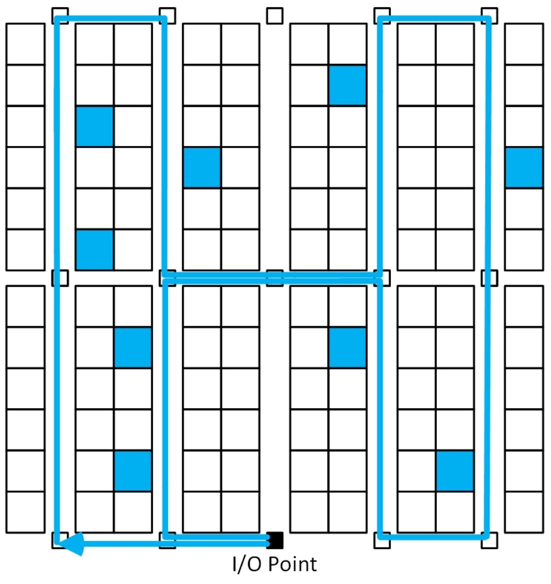
Figure 2: Warehouse simulation parameters and designs: (a) sample warehouse design, (b) set of Parameters from simulation configuration tab.

The initial model consisted of a single block warehouse with just two cross-aisles (no middle cross-aisle) and five aisles. With each added process, we started with visually verifying it. As an example, for *S-Shaped Routing*, we visually checked the path that the worker takes to pick all SKUs of an order from the warehouse. We compared the resulting path from the simulation to what we expect for an S-Shaped routing in single-block and multi-block warehouses (Figure 3a). Also, to verify the COI SLAP method, we developed a heat-map of the SKU frequencies after their location assignment (Figure 3b). As expected, the resulting graphic matched our expectations.

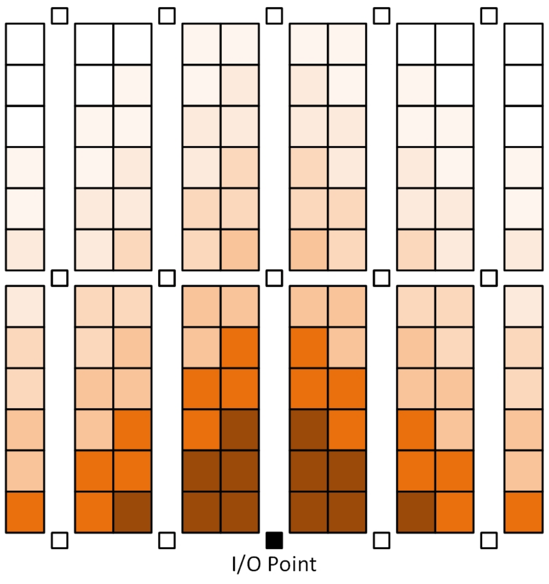
3.4 Simulation Experiment

As we mentioned in Section 3.1, the input dataset consists of more than 3.2M orders. We used the bootstrapping method to randomly sample $10,000$ orders from the original dataset for each run of the simulation. In addition, considering the simulation complexity, we used the *Common Random Numbers* (CRN) concept to test various configurations of our simulation model with the same sequence of random numbers (Yang and Nelson 1991). By using a fixed seed in generating the random numbers, *CRN* reduces randomness for pairwise comparisons (Nelson and Matejcik 1995). Also, considering that there is no queue or buffer in the system, and all aspects of the model are considered to be at their steady state, we eliminated the warm-up period for our experiments.

Our experiment response, as well as our main criteria for comparison, is the simulation termination time. The model terminates when the last order is picked and delivered to the *I/O point* and, hence, represents the pick wave makespan. These simulation time stamps are later analyzed by statistical methods to compare the effects of various configurations. In our experiment, the number of replications are chosen based on experiment design and simulation complexity. For comparing the performance of Gravity Clustering vs. COI (Section 4.1), since the experiment is only one factor with two levels, we replicated the model 50 times. However, although our sensitivity analysis to find the optimal threshold value (Section 4.2) is also a single factor experiment, we decided to run only 10 replications considering the large number of levels. All other simulation parameters are as discussed in Section 3.2.



(a)



(b)

Figure 3: Illustration of sample routing and the SLAP method in a multi-block warehouse; (a) S-shaped routing, (b) COI SLAP.

4 RESULTS

In this section, we compare various configurations for our simulation model. As mentioned in Section 3.4, the criterion for comparison is the overall time that it takes to pick all SKUs from the warehouse. We use *Common Random Numbers* to reduce the randomness caused by the random selection of orders. We perform a t-test for all statistical analyses. In Section 4.1, we compare the results of our suggested Gravity Clustering SLAP method to the widely used and researched COI. In Section 4.2, we perform a sensitivity analysis to find the value of gravity threshold that results in the lowest picking time in the Gravity Clustering method.

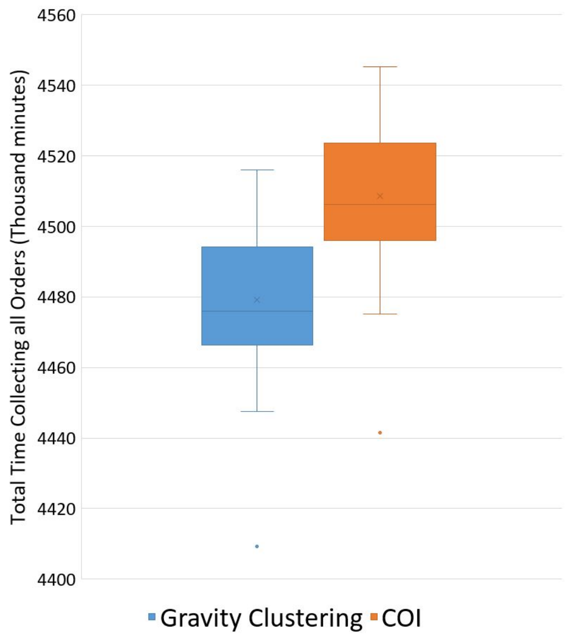
4.1 Performance Comparison of Gravity Clustering and COI

Since the goal in this section is to compare the two SLAP methods, the only difference between the two models is the storage location assigning method. We used the configuration discussed in Section 3.2 and a default value of 100 for the gravity threshold (discussed later in Section 4.2). Figure 4a represents the box plot comparing the simulation results with 50 replications. In Table 4, a paired t-test has been performed to statistically validate our hypothesis that Gravity Clustering performance is superior to that of COI.

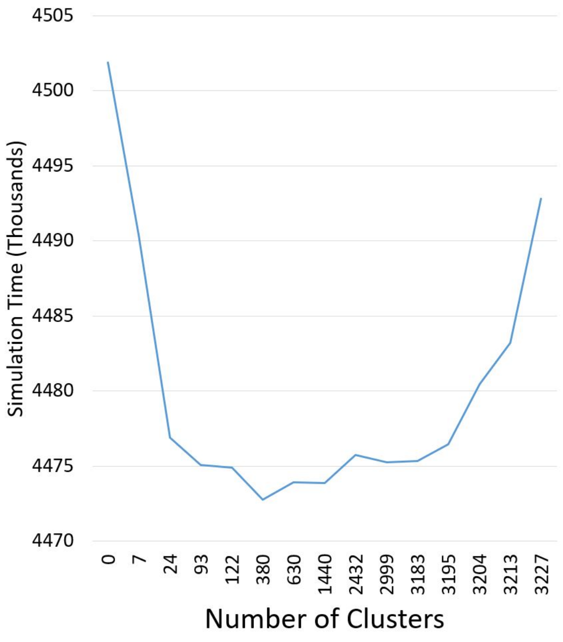
4.2 Optimal Gravity Threshold Value

In this section, we evaluate the effect that the number of created clusters has on the performance of the Gravity Clustering method. Figure 4b is illustrating the effect of cluster count on the total simulation time. It is obvious that as the gravity threshold decreases, more inter-SKU gravitational forces pass the predefined limit and lead to forming clusters. This change in created clusters also means that increasing the gravity threshold high enough will lead to not forming any clusters, which is practically the COI method.

As can be seen in Figure 4b, a Gravity Clustering with no clusters is the same as the COI method. With increasing number of clusters (by decreasing the gravity threshold) the performance of the order picking process improves. The slope of this improvement is at highest for the first clusters and decreases as the



(a)



(b)

Figure 4: Performance analysis of the Gravity Clustering method: (a) box plot comparison of the Gravity Clustering vs. the COI, (b) effect of cluster count of gravity clustering performance.

number of clusters increases. In our case, The improvement stops at around 100 clusters. The performance is almost constant till 3,000 clusters, where not only the results of the simulation do no longer improve, but we can see a pattern of simulation results getting worse. We expected such behavior in Gravity Clustering.

- The first wave of clusters is the most important. This wave consists of the clusters with highly correlated and frequently ordered members. Forming those clusters improves the performance of the order picking process in the simulation model (clusters 0 to 100).
- The less-correlated or less-frequent SKUs form the second wave of clusters. If SKUs are less correlated, although some members of such clusters might be frequently ordered, since the orders that contain both items are not that common, the change in their location will not have any effect on order picking performance. The same happens for the SKUs that are correlated but less frequently

Table 4: Paired t-Test: Gravity Clustering vs. COI with $\alpha = 0.05$

	Gravity	COI
Mean	4,479,116	4,508,524
Variance	401,551,189	401,926,675
Observations	50	50
Hypothesized Mean Difference	0	
df	49	
t Stat	-7.3359	
p value one-tail	3.216E-11	
p value two-tail	6.433E-11	

ordered. Creating a cluster of these SKUs will result in the improvement of that cluster, but since they are rarely ordered, they do not affect model performance (clusters 100 to 3,000).

- The third wave of clusters contains those that are better kept separate. Decreasing the gravity threshold will cause some SKUs that are not ordered that much (e.g., SKU *x*), to be grouped in the same cluster with an SKU that is more frequently ordered (e.g., SKU *y*). Doing this causes the not-so-frequent SKU *x* to move forward and occupy a slot close to the frequently picked one (*y*). This relocation means that all SKUs with order frequency between SKU *x* and *y* will be pushed one slot back. Although this cluster will perform better, all other SKUs will perform worse which causes a decrease in model performance.

We also developed a chart showing the change in cluster count with the increase of the gravity threshold (Figure 5). Since the gravity threshold ranges from 0.0001 to 20M, the horizontal axis of the chart is on a logarithmic scale. Although we expected that as we decrease the gravity threshold, the number of clusters increases exponentially, this did not happen. The reason is the data loss discussed in Section 3.1. The Association Rule Mining Algorithm did not extract many rules that could have contributed to creating clusters. However, as we mentioned, these lost clusters will not have improved the performance of the model.

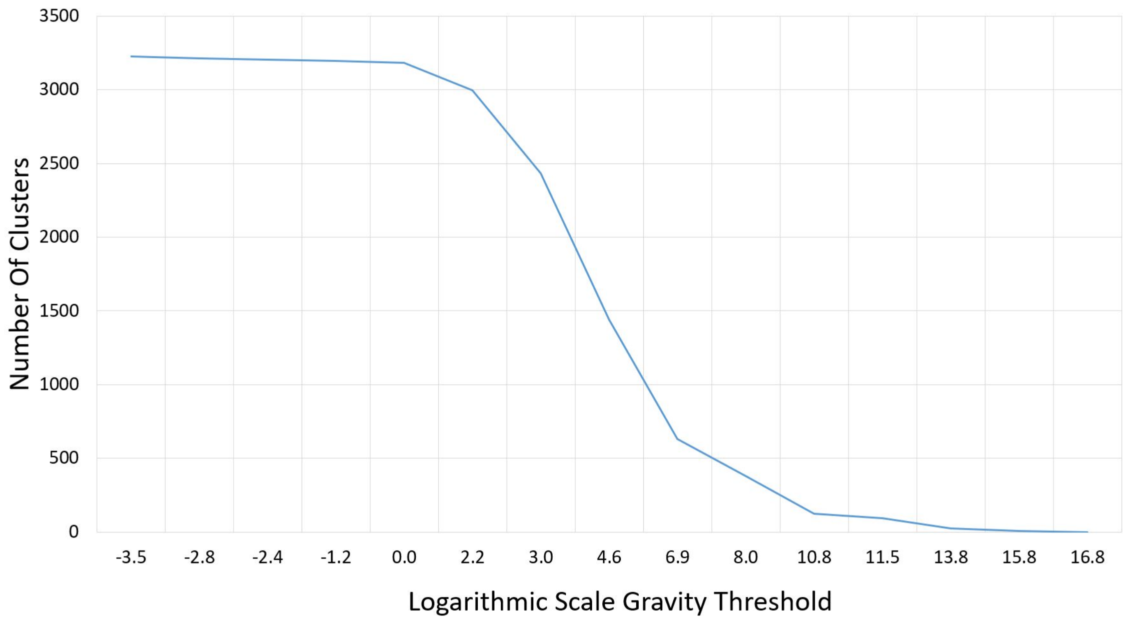


Figure 5: Logarithmic scale representing the effect of the gravity threshold on the cluster count.

We developed a pairwise t-test table to analyze not only the gravity clustering performance, but also the effect of the gravity threshold on the overall simulation time. Table 5 represents a triangular matrix of t-test for various cluster numbers as well as COI (cluster count of 0). The test was performed on results of ten replications for each cluster count.

# of Clusters	3227	3204	3195	3183	2999	2432	1440	630	380	122	93	24	7	0
3227	1													
3204	0.1224	1												
3195	0.0455	0.6045	1											
3183	0.0339	0.5080	0.8836	1										
2999	0.0325	0.4998	0.8751	0.9919	1									
2432	0.0369	0.5406	0.9264	0.9562	0.9479	1								
1440	0.0227	0.3967	0.7376	0.8503	0.8577	0.8067	1							
630	0.0235	0.4027	0.7445	0.8571	0.8645	0.8136	0.9936	1						
380	0.0170	0.3256	0.6339	0.7406	0.7472	0.6984	0.8858	0.8798	1					
122	0.0301	0.4722	0.8378	0.9533	0.9612	0.9095	0.8966	0.9033	0.7851	1				
93	0.0317	0.4865	0.8559	0.9717	0.9797	0.9279	0.8785	0.8852	0.7677	0.9817	1			
24	0.0515	0.6463	0.9530	0.8376	0.8290	0.8796	0.6942	0.7011	0.5935	0.7925	0.8103	1		
7	0.7577	0.2079	0.0833	0.0631	0.0608	0.0686	0.0431	0.0444	0.0327	0.0564	0.0591	0.0935	1	
0 (COI)	0.2427	0.0106	0.0033	0.0024	0.0022	0.0026	0.0015	0.0016	0.0011	0.0021	0.0022	0.0038	0.1448	1

Table 5: t-test matrix calculating statistical differences among various cluster sizes with $\alpha = 0.05$.

5 CONCLUSION AND NEXT STEPS

In this paper, we introduced a new clustering method for the *Storage Location Assigning Problem* (SLAP) based on the famous gravity model and formula. We provided a statistical comparison that proves our method to outperform the widely used and accepted COI under S-Shaped routing method and pre-defined order data structure assumptions. We also investigated various configurations for our method and analyzed the optimal value for the clustering threshold that returns the best results.

Although the introduced method appears to be promising, it is far from a comprehensive approach in solving the storage location assignment problem. More testing and comparisons are needed, such as a side-by-side comparison of our method with other clustering methods introduced by researchers. This comparison can be extended by incorporating different routing methods such as *Mid-Point*, *Largest Gap*, and an optimal routing method provided by *Gurobi*® or other optimization software. Also, the performance of the Gravity Clustering method should be tested with various order data structures. By analyzing the performance with different order data, we can also investigate the behavior of the optimal gravity threshold value in more depth.

REFERENCES

Agrawal, R., T. Imieliński, and A. Swami. 1993. "Mining Association Rules Between Sets of Items in Large Databases". In *Proceedings of the 1993 ACM SIGMOD International Conference on Management of Data*, 207–216. May 26th–28th, Washington, DC.

Ansari, M., B. Rasoolian, and J. S. Smith. 2018. "Synthetic Order Data Generator for Picking Data". In *Proceedings of 15th IMHRC*. July 23rd–26th, Savannah, Georgia. https://digitalcommons.georgiasouthern.edu/pmhr_2018/15.

Ansari, M., and J. S. Smith. 2017. "Warehouse Operations Data Structure (WODS): A Data Structure Developed for Warehouse Operations Modeling". *Computers & Industrial Engineering* 112:11–19.

Bergstrand, J. H. 1985. "The Gravity Equation in International Trade: Some Microeconomic Foundations and Empirical Evidence". *The Review of Economics and Statistics* 67(3):474–481.

Bindi, F., R. Manzini, A. Pareschi, and A. Regattieri. 2007. "Similarity Coefficients and Clustering Techniques for the Correlated Assignment Problem in Warehousing Systems". In *Proceedings of 19th International Conference on Production Research*. July 29th–August 2nd, Valparaiso, Chile.

Bindi, F., R. Manzini, A. Pareschi, and A. Regattieri. 2009. "Similarity-Based Storage Allocation Rules in an Order Picking System: An Application to the Food Service Industry". *International Journal of Logistics: Research and Applications* 12(4):233–247.

Chuang, Y.-F., H.-T. Lee, and Y.-C. Lai. 2012. "Item-Associated Cluster Assignment Model on Storage Allocation Problems". *Computers & Industrial Engineering* 63(4):1171–1177.

De Koster, R., T. Le-Duc, and K. J. Roodbergen. 2007. "Design and Control of Warehouse Order Picking: A Literature Review". *European Journal of Operational Research* 182(2):481–501.

Frazele, E., and G. P. Sharp. 1989. "Correlated Assignment Strategy Can Improve Any Order-Picking Operation". *Industrial Engineering* 21(4):33–37.

Grigoryev, I. 2012. *AnyLogic 6 in Three Days: A Quick Course in Simulation Modeling*. AnyLogic North America.

Heskett, J. L. 1963. "Cube-per-Order Index: A Key to Warehouse Stock Location". *Transportation and Distribution Management* 3(1):27–31.

Instacart 2017. "The Instacart Online Grocery Shopping Dataset". https://www.instacart.com/datasets/grocery-shopping-2017, accessed 2nd March 2020.

Kallina, C., and J. Lynn. 1976. "Application of the Cube-per-Order Index Rule for Stock Location in a Distribution Warehouse". *Interfaces* 7(1):37–46.

Kim, B. S., and J. S. Smith. 2012. "Slotting Methodology Using Correlated Improvement for a Zone-Based Carton Picking Distribution System". *Computers & Industrial Engineering* 62(1):286–295.

Nelson, B. L., and F. J. Matejcik. 1995. "Using Common Random Numbers for Indifference-Zone Selection and Multiple Comparisons in Simulation". *Management Science* 41(12):1935–1945.

Van Gils, T., K. Ramaekers, K. Braekers, B. Depaire, and A. Caris. 2018. "Increasing Order Picking Efficiency by Integrating Storage, Batching, Zone Picking, and Routing Policy Decisions". *International Journal of Production Economics* 197:243–261.

Wikibooks. 2019. "Human Geography AP/Gravity Model; Wikibooks, The Free Textbook Project". https://en.wikibooks.org/w/index.php?title=Human_Geography_AP/Gravity_model&oldid=3599717, accessed 21st February 2020.

Xiao, J., and L. Zheng. 2010. "A Correlated Storage Location Assignment Problem in a Single-Block, Multi-Aisles Warehouse Considering BOM Information". *International Journal of Production Research* 48(5):1321–1338.

Xiao, J., and L. Zheng. 2012. "Correlated Storage Assignment to Minimize Zone Visits for BOM Picking". *The International Journal of Advanced Manufacturing Technology* 61(5-8):797–807.

Yang, W.-N., and B. L. Nelson. 1991. "Using Common Random Numbers and Control Variates in Multiple-Comparison Procedures". *Operations Research* 39(4):583–591.

Zhang, Y. 2016. "Correlated Storage Assignment Strategy to Reduce Travel Distance in Order Picking". *IFAC-PapersOnLine* 49(2):30–35.

AUTHOR BIOGRAPHIES

MOHAMMADNASER ANSARI is a teaching assistant and PhD candiate in the Department of Industrial and Systems Engineering at Auburn University. His research interests are in simulation, visualization analytics, supply chain management, and distribution center design and operations. His email address is: ansarim@auburn.edu.

JEFFREY S. SMITH is the Joe W. Forehand Professor of Industrial and Systems Engineering at Auburn University. He has served as the WSC Business Chair (2010) and General Chair (2004) and is currently on the WSC Board of Directors. He has a BIE from Auburn University and a MS and PhD (both in Industrial Engineering) from Penn State University. His email and web addresses are: jsmith@auburn.edu and http://jsmith.co.