Multi-objective Path Planning Based on K-D Fusion Algorithm

Supeng Kong¹, Surong Chen¹*, Zhaoxi Zhong², Bin Xu¹, Tianlei Shi³, Xiaohui Zhu³, Yong Yue³ and Wei Wang⁴

¹School of Information Science and Technology, Nantong University, Nantong, China
²School of Computer Science and Technology, Suzhou University, Suzhou, China
³Department of Computing, Xi’an Jiaotong-Liverpool University, Suzhou, China
⁴College of Computer and Cyber Security, Hebei Normal University, Shijiazhuang, China

*Email: firstemail@ntu.edu.cn

Abstract. How to arrange sales staff to visit offline stores reasonably is a critical task in the Fast Moving Consumer Goods (FMCG) industry. Based on the K-means and Dijkstra algorithms (K-D fusion algorithm), this paper proposed an algorithm to automatically allocate offline stores for sales staff and optimize the visiting path, thereby improving management efficiency. A new initial cluster center selection approach was proposed for the K-means algorithm to select its initial clustering center with the consideration of outlier points. The sales staff’s visiting path to the store was planned by the Dijkstra algorithm. The performance of our K-D fusion algorithm was evaluated in terms of grouping rationality and path planning optimization. Experimental results show that our algorithm can comprehensively consider several factors such as the number of stores, store types, and geographical locations, and distribute the workload more evenly to all the sales staff. In addition, it can also optimize the visiting path for sales staff, which can effectively improve the efficiency of sales staff to visit offline stores.

1. Introduction

With the rapid development of the Internet, production promotion activities in the industry of Fast Moving Consumer Goods (FMCG) have also expanded to online. For example, sellers can communicate with customers anytime and anywhere through WeChat applets. Many companies are trying “one-thing, one QR code” sales strategy to transfer offline users to WeChat public account to strengthen the communication between companies and users, and developing online malls to directly sale fast-consuming products to users. However, the effectiveness of these online sales is based on the stickiness of offline customers. So such online sales cannot stabilize their customers. Thus, sales staff need to maintain customer relationship through offline marketing such as telephone marketing, distribution of flyers and unfamiliar visits (visits to strangers for the first time without an appointment are a common way for business personnel to find customers) to complete the corresponding sales tasks. Therefore, how to reasonably arrange sales staff to visit different customer stores and arrange the visiting path to improve sales staff’s work efficiency and obtain first-hand sales information from customer stores has become a critical issue that the FMCG industry needs to solve. At present, the common approach is to manually set the Key Performance Indicator(KPI) of each sales staff for the overall sales and products’
exhibition area, and follow up each month. Nonetheless, the manual approach has the disadvantages of high cost and low planning accuracy.

This paper mainly aims at the improvement of the K-means algorithm to the problem of uneven processing between outlier customer stores and central customer stores and proposes an improved K-means algorithm to significantly alleviate the negative impact of the outlier points to the cluster of customer stores. In addition, the Dijkstra algorithm is used to optimize the visiting path for each sales staff to improve work efficiency. The second section of the paper introduces K-means and Dijkstra algorithms. Section 3 describes a real data set of customer stores used in this paper and application requirements and constraints. In the fourth section, an improved k-means algorithm is proposed and optimized to cluster the customer stores. Experimental results show that the algorithm has good stability and can be used to cluster different stores and plan reasonable visiting paths for sales staffs [1,2]. Section 5 demonstrated a cluster of customer stores based on Baidu map and actual store data. We analysed the experimental data in detail in Section 6. Finally, we summarized the paper and discussed future work.

2. Algorithm Introduction

2.1. K-means Algorithm

The K-means algorithm is classified as a basic division approach for clustering and is widely used in text search, pattern recognition, artificial intelligence, image analysis and other fields[3]. The sum of squared errors is often used as a clustering criterion function. The specific steps are as follows: (1) randomly select K objects as the initial clustering centers; (2) calculate the distance between each object and each seed clustering center, and (3) assign each object to the nearest cluster. The cluster centers and the objects assigned to them represent a cluster. Each time a sample is assigned, the cluster center of the cluster will be recalculated based on the existing objects in the cluster. This process will repeat until a termination condition is satisfied. The termination condition may be that no (or minimum number) objects are reassigned to different clusters, or no (or minimum number) cluster centers change again, or the sum of squared error is a local minimum [4].

The specific process is as follows [5]:
Input: The number of clusters K and a data set containing N objects.
Output: K clusters with minimal square errors.
Method:
(1) K objects are randomly selected from all the data samples as the initial clustering centers;
(2) According to the principle of being closest to the center, the distance from other data objects to the center of each cluster is calculated, and the data objects will be assigned to each corresponding cluster;
(3) For each cluster, the average of all its objects is calculated as the new cluster center;
(4) According to the principle of being closest to the center, reassign data objects;
(5) Return to step (3) for loop execution. When the objective function no longer changes, the algorithm ends.

As a classic clustering algorithm, the K-means algorithm has some advantages as well as disadvantages. For example, the initial cluster number K must be given in advance, so that the initial cluster center selection has randomness. However, in practice, K is difficult to be accurately determined in advance, which makes the algorithm ineffective for some practical problems. In addition, the algorithm is easy to fall into a local optimal solution and can be significantly affected by outlier points [6,7]. Due to the influence of outlier stores and special clustering criteria for particular stores, there is a high probability that some visiting paths are too long while others are too short, which is not reasonable in practice. Therefore, the original K-mean algorithm cannot satisfy the practical scenario to plan an optimized visiting path for sales staff.

2.2. Dijkstra Algorithm

Dijkstra algorithm is a typical shortest path selection algorithm, which is widely used in path planning scenarios. The algorithm is actually to ensure the shortest local distance to obtain the shortest overall
distance. It is an algorithm based on a greedy strategy. For graph G = (V, E), Dijkstra algorithm needs to maintain two sets of nodes [8]:

S: The set of vertices contained in the shortest path from the source point s to the destination point d;

V - S: The set of vertices except the vertices contained in the shortest path. Reserve 2 arrays for each vertex: (1) Estimated value for the shortest path from the source point to each vertex, and (2) The parent vertex of each vertex in the shortest path tree. The Dijkstra algorithm adds the sub-vertices of G with the vertices in the S set as the parent points, and the vertices in the V-S set as the sub-points to the shortest side into S until all the vertices are added or when all the points in the V-S point set are infinite. If the path from the source point to a point does not exist, it can be assumed that the shortest path at that point is a virtual path with an infinite length [9].

3. Experimental Data and Clustering Constraints

Application scenario: The sales staff needs to start from a fixed start point every day and visit each store on the planned path without returning to the start point. Some customer store data are shown in Table 1.

| Store number | Store type | Sales channels | Monthly visit frequency | Longitude     | Latitude     |
|--------------|------------|----------------|-------------------------|---------------|--------------|
| 1            | α          | B              | 6                       | 112.568882    | 37.888571    |
| 2            | α          | B              | 5                       | 112.547825    | 37.888285    |
| 3            | β          | D              | 12                      | 112.596169    | 37.899583    |
| 4            | β          | D              | 6                       | 112.546842    | 37.807010    |
| 5            | α          | E              | 7                       | 112.560050    | 37.825230    |
| ...          | ...        | ...            | ...                     | ...           | ...          |

There are two types of customer stores: α and β. α type stores are ordinary stores, while β type stores are extremely remote from the city center. Stores are also divided into seven sales channels: A, B, C, D, E, F, and G. The G channel is a special sales channel, which needs to be clustered separately. The monthly visit frequency is the number of times the sales staff needs to visit the corresponding store per month. Longitude and latitude are the geographic location coordinates of the store based on the Baidu map.

There are also some store clustering restrictions as follows:

(1) Workload saturation for each staff should be equal to or greater than 90% (standard saturation).

\[
\text{Workload saturation} = \frac{A \text{ frequency}}{12} + \frac{B \text{ frequency}}{14} + \frac{C \text{ frequency}}{14} + \frac{D \text{ frequency}}{4} + \frac{E \text{ frequency}}{4} + \frac{F \text{ frequency}}{11} + \frac{G \text{ frequency}}{15} \right) / 22
\]

(2) The travel time between the two customer stores should be within one hour.

(3) Particular constrained customer stores should be clustered independently.

Particular constraint customer stores are divided into β type stores (remote area stores) and G channel stores. The distance between the stores is relatively large, and they need to be clustered separately by a particular sales staff in charge. When the workload saturation of a cluster is less than the standard saturation, we can add other types of customer stores to the cluster to increase the workload.

(4) Stores with different sales channels have different stay time shown in Table 2.
Table 2. Stay time for stores with different channels.

| Sales channels | Stay time for each store (maximum) |
|----------------|-----------------------------------|
| A              | 25 minutes                        |
| B              | 20 minutes                        |
| C              | 20 minutes                        |
| D              | 120 minutes                       |
| E              | 120 minutes                       |
| F              | 30 minutes                        |
| G              | 15 minutes                        |

4. Algorithm Implementation and Optimization

4.1. Clustering G Channel Stores with Particular Constraints
There is a constraint for G channel stores that all G channel stores should be allocated to one sales staff. If the workload saturation of the G channel store cluster is less than the standard workload saturation, then other types of stores can be added into the G channel store cluster.

Input: Latitude and longitude coordinates for all G channel customer stores.
Output: store clusters and the workload saturation.

Method:
1. Filter out all G channel stores and calculate the total workload saturation;
2. Compare the total workload saturation with the standard workload saturation;
3. If the total workload saturation $\leq$ standard workload saturation:
   Find the center of all the G channel stores, and add the closest store to the cluster of G channel stores one by one according to the distance between the store and the cluster center of all G channel stores until the workload saturation of the cluster reaches the standard workload saturation;
4. If the workload saturation $> \text{standard workload saturation}$:
   Calculate the distance from the center of the cluster to the remaining stores; find a store A that is farthest from the cluster center. Then take store A as a new center, and cluster stores that are nearest to the store A one by one into a new cluster until the workload saturation of the new cluster reaches the standard workload saturation;
5. Cluster the remaining unassigned stores by steps (2), (3), (4) and (5).

4.2. Clustering $\beta$ Type of Stores with Particular Constraints
Input: Latitude and longitude coordinates for all $\beta$ type stores
Output: Store clusters and corresponding workload saturation

There is also a constraint for $\beta$ type of stores that all $\beta$ type stores should be allocated to one sales staff. So the clustering procedure is similar to G channel stores.

By the approach proposed above, we combine remote stores with center stores. After getting a new cluster, the algorithm continues to search for the remotest store, which is farthest from the center store, and then let the remotest store be the center of a new cluster. The approach not only ensures that the workload saturation of each cluster reaches the standard workload saturation but also prevents remote stores in different regions from being assigned to the same cluster, which significantly reduces the travel time within stores in a cluster.

4.3. Clustering Common Stores
Input: Latitude and longitude coordinates for all the remaining stores except G channel and $\beta$ type stores
Output: Store clusters and corresponding workload saturation
Method:
Search the farthest store A by calculating the Euclidean distance between all stores and the start location.

Let store A be the center of a cluster, add the stores which are nearest to store A into a cluster, respectively, until the total workload of the group reaches the standard workload.

Cluster the remaining stores by the steps (1) and (2) until all stores are clustered.

The pseudocode of the algorithm is shown in Algorithm 1. where the distance \( \text{dis}(x, y) \) represents the distance between store x and store y. Formula (1) shows the distance calculation.

\[
\text{Distance} = 2 \times \text{asinh} \left( \frac{\text{dis}_x}{2} \right) = \text{asinh} \left( \frac{\text{dis}_y}{2} \right) = \frac{\text{EARTH RADIUS} \times 1000}{2} \times \sin \left( \frac{\text{dis}_x}{2} \right) \times \cos \left( \frac{\text{dis}_y}{2} \right) \times \cos \left( \frac{\text{dis}_x}{2} \right) \times \sin \left( \frac{\text{dis}_y}{2} \right)
\]

where \( \text{dis}_x \) and \( \text{dis}_y \) are the difference in radians between stores x and y. EARTH_RADIUS is 6378.137, a constant radius of the earth. \( A_x, A_y, B_x, B_y \) are radians converted from the angle according to the latitude and longitude.

The main procedure is shown in Algorithm 1. The Delete () function deletes the stores that have been clustered and update the data set to prepare for the next clustering. Sort (s, k) function sorts all stores in set s according to the distance between each store in s and the store k and then clusters them according to the standard workload saturation. The Group() method clusters the sorted stores by workload saturation and returns the cluster after clustering.

**Algorithm 1: Clustering**

1: Initialize: store, workload_saturation_standard, start_store
2: group ← 0
3: While store
4:    group ← group + 1
5:    Distance(store, start_store)
6:    farthest_store ← Sort(store, start_store)
7:    Distance(farthest_store, store)
8: Group()
9: Delete()
10: if last_store_workload_saturation << workload_saturation_standard then
11:    center_point ← last_store_cluster
12:    remaining_store ← Distance(remaining_store, center_point)
13: Sort(remaining_store, center_point)
14: add to the group
15: End

The clustering algorithm synthetically considers the selection of the initial cluster center and the outlier stores. During each clustering procedure, the algorithm selects the most outlier store instead of the cluster center to start clustering, which can significantly reduce the negative impact of particular outlier stores[10].

5. **Algorithm Implementation**

5.1. **Getting the Latitude and Longitude according to the Store Address**

We use "geocoding" service on the Baidu map developer platform to convert the structured store address into BD-09 coordinates (latitude and longitude). The service has a specific GET request format as follows:

http://api.map.baidu.com/geocoding/v3/?address=DetailedAddress&output=json&ak=Applied Ak&callback=showLocation. We submit the address to the request, and it returns the coordinates to us.
5.2. Get travel Time for Different Ways of Transportation

To obtain accurate travel time using different transportation tools, we use the "batch calculation" service on the Baidu map developer platform. The route distance and travel time can be calculated based on the coordinates of two stores. Based on this approach, the travel time for bus, driving, and walking between two stores in each cluster can be obtained for path planning, respectively.

5.3. Planning a Visiting Path
(1) Get all the stores in a cluster.
(2) Let a store which is closest to the starting location be a visiting point and push into the end of the path.
(3) Use the Dijkstra algorithm to find a store which is closest to the visiting point and included in the path.
(4) Let the store be the visiting point and add it to the end of the path.
(5) Repeat steps (3) and (4) until all the stores are included in the path.

6. Experimental Results and Analysis

6.1. Actual Data and Experiment Results

The experimental data set has 460 customer stores in a city with properties of address, store type, channel type and visiting frequency. We obtain the longitude and latitude for each store based on Baidu map developer platform mentioned above. The horizontal coordinate range of all the stores can be set between 36.0E ∼ 40.0E and the vertical coordinate range between 110.0N ∼ 114.0N.

To determine the number of clusters, we calculate the total workload for all the 460 stores. Compared to standard workload saturation, we find that we at least divide all the store into seven clusters. However, the workload saturation of seven clusters is 103.3%, 99.3%, 99.1%, 99.6%, 99.6%, 100.4%, 98.3%, respectively, which is higher than the standard workload saturation. Thus we set the number of clusters to 8. The workload saturation of eight clusters is shown in Figure 1.

![Figure 1. Workload saturation and distribution of 8 clusters.](image)

We find that the last cluster has a much lower workload saturation than other clusters. So we slightly decrease the standard workload saturation to 85%.

We let K be 8, and the standard workload saturation be 85%. We both use the original and optimized K-means algorithms to cluster the 460 stores. The workload saturation and cluster distribution are shown in Figure 2.
The clustering result based on the original K-means algorithm is shown in Figure 2(a). Different clusters are distinguished with different colors. We find that the cluster in the center of the figure (red color) has more stores than other clusters, while outlier clusters such as cluster 3, 4 and 5 only have a small number of stores, which does not satisfy the requirement of averaging workload among sales staffs. When using the improved algorithm, we obtain more even clustering results in Figure 2(b). Figure 3 shows the difference of cluster centers based on the original K-means and the improved K-means algorithms.
Figure 3(a) shows that the eight cluster centers generated by the original K-means algorithm are all in the middle of the center. Nearly 80% of stores are clustered into one cluster, and the remaining 20% are distributed within the other seven clusters. The disadvantage of the original K-means algorithm is that some isolated stores may be independently clustered, which violates the principle of even workload distribution. From Figure 3(b) we find that 4 cluster centers are in the city center and 3 cluster centers are located at outlier stores, which is more reasonable than the distribution of cluster centers in Figure 3(a). In addition, the workload saturation in Figure 3(b) is 85.3%, 89.4%, 89.9%, 91.0%, 85.4%, 83.3%, 86.5% and 83.6%, respectively, which is more even than the workload distribution in Figure 3(a). Figure 4 compares the workload saturation based on the three methods of original K-means store clustering, improved K-means store clustering (seven groups), and improved K-means store clustering (eight groups).
Based on the clustering results above, we use the Dijkstra algorithm to plan an optimized visiting path for a sales staff by driving. Figure 5 shows the optimized visiting path for a sales staff.

In Figure 5, the employee needs to visit 13 α type stores in one day. Among them, there are 10 G channel stores, and 1 C, F, and B channel stores, respectively. The total stay time in the store is 3.83 hours. It takes 2.27 hours to visit by driving. The total working time of the day was 6.1 hours, and the work saturation reached 90%.
7. Conclusion
In this paper, we developed an A-D fusion algorithm to allocate customer stores into different clusters and evenly distribute workload to different sales staff. Experimental results show that our algorithm can allocate the workload evenly to different sales staff and optimize the visiting path as well as satisfying clustering constraints. In the future, we will cooperate with external companies and apply this multi-objective optimization algorithm to a real application with more customer stores.

Acknowledgements
This work was partly supported by the AI University Research Centre (AI-URC) through XJTLU Key Programme Special Fund (KSF-P-02 and KSF-A-19) and Research Development Fund of XJTLU (RDF-19-02-23).

References
[1] Bozyigit A & Alankuş G & Nasiboglu E 2017 Public transport route planning: Modified dijkstra's algorithm. pp 502-505. doi: 10.1109/UBMK.2017.8093444.
[2] Biswas S, Anavatti S G and Garratt M A 2019 Multiobjective mission route planning problem: a neural network-based forecasting model for mission planning. IEEE Transactions on Intelligent Transportation Systems 99 pp 1-13
[3] Jain A K 2010 Data clustering: 50 years beyond k-means. Pattern Recognition Letters 31(8) 651-666
[4] Yusoff I and Mat I, Nor A & Hasikin K 2013 Automated two-dimensional K-means clustering algorithm for unsupervised image segmentation. Computers & Electrical Engineering. 39 907–917
[5] Sun B & Wu D 2012 Self-Organizing-Queue Based Clustering. IEEE Signal Processing Letters. 19 902-905
[6] Xiuman D and Guoxia S and Tao Y 2011 Moving target detection based on Genetic K-means Algorithm. doi: 10.1109/ICCT.2011.6157992.
[7] Hartigan J A and Wong M A 1979 A k-means clustering algorithm. Applied Statistics 28(1) 100-108
[8] Wu T-F, Tsai P-S, Hu N-T, Chen J-Y 2017 Combining turning point detection and Dijkstra’s algorithm to search the shortest path. Advances in Mechanical Engineering. doi:10.1177/1687814016683353
[9] Yujin and Xiaoxue G 2017 Optimal Route Planning of Parking Lot Based on Dijkstra Algorithm 2017 International Conference on Robots & Intelligent System (ICRIS), (Huai'an, China) pp. 221-224
[10] Si-Si W, Jing-Lei Z and Jian-Xing W 2018 Outlier detection of reckless driving behaviors base on improved swarm intelligent K-means algorithm Science Technology and Engineering 18(34) 208-214