International Journal of Physics and Mathematical Sciences ISSN: 2277-2111 (Online)
An Open Access, Online International Journal Available at http://www.cibtech.org/jpms.htm
2016 Vol. 6 (2) April-June, pp. 11-17/Shi
Research Article

APPLICATION OF IMPROVED K-MEANS ALGORITHM IN THE DELIVERY LOCATION

***Ming Shi**
Department of Mathematics and Information Science, Hebei University, Baoding 071002, China
**Author for Correspondence*

ABSTRACT
This paper had proposed an improved k-means algorithm and applied it to a living example of delivery location. At first, established an undirected weighted graph based on the actual path; Then, applied Floyd algorithm to calculate a nearest path; Finally, utilized a improved k-means algorithm which the Euclid distance was substituted by the nearest path to cluster and acquired a rational outcome. Therefore, it verified this algorithm could be employed in practical situation.

Keywords: *Undirected Weighted Graph, the Shortest Path Matrix, Floyd Path, K-Means Algorithm*

INTRODUCTION
Quantity of goods delivery is directly related to the express company's earnings. Therefore, whether the substation and their delivery location is reasonable becomes important. These sites which the path between of them is shortest are the most important criteria to select these points.

At present, there are a number of researchers had carried out research and exploration in this area. According to the superiority-inferiority of the existing k-means, (Gu *et al.*, 2010) designed a new equilibrium distribution region division method, and verified the practicability and validity of this method; (Yan *et al.*, 2000) obtained a shortest path algorithm of city road network, and verified its practicality and reliability through the GIS; (Bao, 2001) adjusted apical order of the existing method, and optimized the Dijkstra algorithm, significantly improved the computing speed and algorithmic efficiency.

Combined with actual situation, the article puts forward an improved k-means algorithm, using the Floyd path instead of the commonly Euclidean distance, to determine the number k and the clustering center grounded on the actual circumstance. It can avoid the result is blindness or irrationality, and the result conforms to the actual need, therefore can be applied into the actuality.

Floyd Shortest Path Algorithm
Almost all of the classic clustering algorithms are based on the Euclidean distance. The Euclidean distance evaluates the similarity of data is obtained from the linear properties between two points in space. In actually, the real distance between two point restrained by the actual circumstance is possible far greater than the Euclidean distance. It only obtains a result distorted the fact by utilizing the Euclidean distance under the circumstance. Therefore, it is the crux whether the adopted distance is reasonable to realize the algorithm.

Calculation of distance algorithm is named the shortest path algorithm, including Dijkstra algorithm, A* algorithm, Johnson algorithm, Floyd algorithm (Zhang and Wu, 2009) and so on. These algorithms had been systematic classification and comparison by (Lu, 2001).

Floyd algorithm can be exploited to find the shortest path between every pair of vertices in a weight graph with positive or negative edge weights but no aggregative cycles. It is endowed with concision and elegance, and pretty easy to understand. In addition, it is simple and extra effective, especially on dense graph.

Related Concepts

Some of the definitions used in this paper are derived from (Bondy and Murty, 2008).

Definition 1 A undirected weighted graph *G*, showed in Figure 1, is an ordered pair $(e(G), v(G), \Psi_G)$ consisting of a set *v(G)* of vertices and a set *e(G)*, disjoint from *v(G)*, of edges, together with an incidence function Ψ_G that associates with each edge of *G* an unordered pair of (not necessarily distinct) vertices of *G*. If *e* is an edge, *u* and *v* are vertices, such that $\Psi_G(e) = \{u, v\}$, then *e* is said to join *u* and *v*, and the vertices *u* and *v* are called the ends of *e*. We denote the numbers of vertices and edges in *G* by *v(G)* and *e(G)*; these two basic parameters are called the order and size of *G*, respectively.

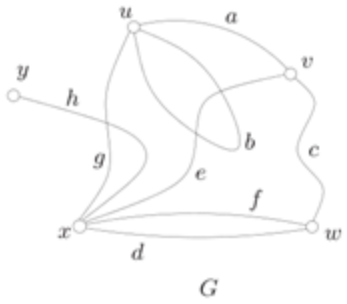

Figure 1: Undirected Graph

Here, *u*, *v*, *w*, *x* and *y* represents vertices, and *a*, *b*, *c*, *d*, *e*, *f*, *g*, *h* represents the edge.

Definition 2 The incidence matrix *M* of *G*, is showed at the left of the table 1, is the $n \times m$ matrix $M_G := (m_{ve})$, where m_{ve} is the number of times (0, 1, or 2), and vertex *v* and edge *e* are incident. Clearly, the incidence matrix is just another way of specifying a graph.

Definition 3 The adjacency matrix *A* of *G*, is showed at the right of the table 1, is the $n \times n$ matrix $A_G := (a_{uv})$, where a_{uv} is the number of edges joining vertices *u* and *v*, each loop counting as two edges.

Table 1: Incidence Matrices M and Adjacency Matrices A of a Graph G

M	A	b	c	d	e	f	g
u	1	2	0	0	0	0	1
v	1	0	1	0	1	0	0
w	0	0	1	1	0	1	0
x	0	0	0	1	1	1	1
y	0	0	0	0	0	0	0

A	u	v	w	x	y
u	2	1	0	1	0
v	1	0	1	1	0
w	0	1	0	2	0
x	1	1	2	0	1
y	0	0	0	1	0

Definition 4 The shortest path matrix A, can be seen from the Table 2, is defined as a $n \times n$ matrix. If

vertices $v_i(1 \le i \le n)$, $v_j(1 \le j \le n)$, there is at least one path. A_{ij} is the value of the element and d_{ij} is the Floyd path of the vertices v_i to the vertices v_j.

If vertices v_i and vertices v_j does not exist a path between, than the value of element A_{ij} is denoted ∞. That is:

$$A_{ij} = \begin{cases} d_{ij} = \min\{W(P) | \text{Path from } v_i \text{ to } v_j \text{ for } P \text{ in } G\} \\ \infty \quad \text{When the vertices } v_i \text{ to } v_j \text{ is not reachable} \end{cases}$$

Table 2: The Shortest Path Matrix A

$$A = \begin{pmatrix} 0 & 8 & 3 & 7 & 9 & 12 & 9 \\ 8 & 0 & 5 & 9 & 7 & 4 & 8 \\ 3 & 5 & 0 & 10 & 11 & 9 & 6 \\ 7 & 9 & 10 & 0 & 2 & 5 & 7 \\ 9 & 7 & 11 & 2 & 0 & 3 & 5 \\ 12 & 4 & 9 & 5 & 3 & 0 & 8 \\ 9 & 8 & 6 & 7 & 5 & 8 & 0 \end{pmatrix}$$

Algorithm Principle

Floyd algorithm compares all possible paths through the graph G with vertices V between each pair of vertices.

The shortest path d_{ij} that returns the shortest possible path from i to j using vertices only from the set $\{1, 2, \ldots, k\}$ as intermediate points along the path. For any vertices i and any vertices j, the true shortest path could be either: (1) a path that goes directly from i to j, or (2) a path that goes from through a number of others vertices to j.

Assuming d_{ij} is the shortest path to the vertices i to vertices j, for each vertices k, examines the formula $d_{ik} + d_{kj} < d_{ij}$ to speculate whether it still holds. When it is, explains the path from vertices i to vertices k and then to vertices j is shorter than from vertices i directly to vertices j, then sets $d_{ij} = d_{ik} + d_{kj}$. When traverses all vertices k, d_{ij} is the shortest path that we wants to know. Its time complexity is $O(n^3)$, and the space complexity is $O(n^3)$.

Process

Some symbols given by (Wang *et al.*, 2010) used in the paper are given as follows:

w represents undirected graph weighted matrix;

v_0 and v_t represents an arbitrary source vertex and target vertex respectively;

$d(v_0, v_t)$ is the shortest path to the source vertex v_i to the target vertex v_j $(1 \le i \le n,\ 1 \le j \le n)$;

A is the weight matrix of the shortest path;

Input: The weight matrix w of the undirected graph; the source vertex v_0; the target vertex v_t.

Output: The source vertex v_0 to the target vertex v_t, the shortest path $d(v_0, v_t)$.

Step 1, initialize the weight matrix $A^{(0)} = \left(d_{ij}^{(0)}\right)_{n \times n}$ $(i, j = 1,2,\ldots n;\ k = 1)$ (Mao and Shi, 2016).

Among, $d_{ij}^{(n)} = \begin{cases} w(i,j) & i \neq j \\ 0 & i = j \\ \infty & i, j \text{ is not adjacent or have no way to go} \end{cases}$

Step 2, when $k \neq i$, j (k from 1 to n), for all the i and j, checks whether the $d_{ij} > d_{ik}$ or $d_{ij} > d_{kj}$, if meet, then sets $k=k+1$, and continues step2; otherwise, step 3;

Step 3, compare the size of d_{ij} and $d_{ik} + d_{kj}$, replace d_{ij} with the smaller, that is $d_{ij} = \min\left(d_{ij},\ d_{ik} + d_{kj}\right)$, turn step 4;

Step 4, to determine whether the k is less than n, if established, return to Step 2; otherwise, step 5;

Step 5, output source point v_0 to the target point v_t the shortest path $d(v_0, v_t)$.

K-Means Clustering Algorithm

K-means algorithm (Liu, 2011) is popular for cluster analysis in data mining. It divides n observations into k clusters in which each observation belongs to the cluster with the nearest mean, serving as a prototype of the cluster.

Principle

Given a set of observations $X = \left\{x_m \middle| m = 1,2,\text{K } total\right\}$, the sample of X is denoted by d description attribute $A_1, A_2, A_3, \ldots A_d$ and these attributes belong to continuous attribute. Data sample is $x_i = (x_{i1},\ x_{i2},\ .....x_{id})$, $x_j = (x_{j1},\ x_{j2},\ .....x_{jd})$. Among them, $x_{i1},\ x_{i2},\ .....x_{id}$ and $x_{j1},\ x_{j2},\ .....x_{jd}$ denote the specific value of d description attributes $A_1, A_2, A_3, \ldots A_d$ respectively corresponded sample x_i and sample x_j. The similarity between sample x_i and sample x_j is usually denoted by their Euclidean distance $d(x_i, x_j)$. The definition is as follow:

$$d(x_i, x_j) = \sqrt{\sum_{k=1}^{d} (x_{ik} - x_{jk})^2} \quad \ldots\ldots (1)$$

The similarity is calculated by the average of objects in one cluster:

$$\overline{x_i} = \frac{1}{C_i} \sum_{x \in C_i} x \quad \ldots\ldots (2)$$

Where C_i is the number of data point in cluster i.

Clustering performance is assessed by the criterion function E:

$$E = \sum_{i=1}^{k} \frac{1}{C_i} \sum_{x \in C_i} \left| x - \overline{x_i} \right|^2 \ldots\ldots (3)$$

For a data set of size n, the specified number of clusters is k, D is the dimension of the data object, the

total time complexity of the algorithm is $O(ndk)$.

Process

Input: the number k of clusters D and a data set containing n object.

Output: a set of k clusters that the criterion function is the least.

Step 1. Assigns arbitrary k objects from data D as the initial cluster centers;

Step 2. Calculates the similarity between the remaining objects to the k cluster centers, and divides these objects into a cluster to that the similarity is least;

Step 3. Recalculates k cluster center based on the criterion function of every cluster;

Step 4. Continue to cluster based on k new cluster center;

Step 5. Repeat;

Step 6. Until criterion function is no more obvious changes.

Improved K-Means Algorithm

K-means is a classic algorithm, it has the advantages of simple principle, and it is easy to implement, especially for large data sets, high efficiency and so on (Kanungo and Mount, 2002). It has been widely used.

But the algorithm also has many problems due to limited: (1) It need a given class number in advance; (2) The initial cluster centers influence the effect and quality of clustering directly; (3) to deal with categorical data; (4) It is sensitive to outliers and can only be found in the spherical classes; (5) sometimes fall into local optimal solution, and can't get the global optimal solution. In order to improve these shortcomings, an improved k-means algorithm is proposed that the Euclidean distance is replaced by the Floyd path.

Example

In order to verify the feasibility and effectiveness of the improved k-means clustering algorithm, the example are given to verify the effectiveness of the algorithm.

Figure 2 displays undirected weighted graph of a cargo transport line. In this figure, seven circles of v1、v2、v3、v4、v5、v6、v7 respectively represents the customer's location, the number represents actual required time between the two vertices, the unit is hour (h), therefore, the figure can be viewed as a time path diagram.

In order that the courier can complete the delivery task in the shortest time, there are two vertices in these locations that should be chosen as a distribution site. To this end, Floyd algorithm is applied to calculate the shortest path matrix A, as shown in Table 3.

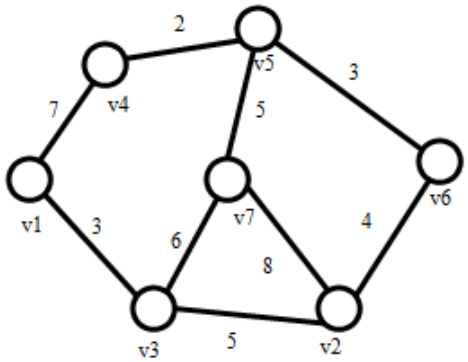

Figure 2: Undirected Weighted Graph

Table 3: Shortest Path A

$$A=\begin{pmatrix} 0 & 8 & 3 & 7 & 9 & 12 & 9 \\ 8 & 0 & 5 & 9 & 7 & 4 & 8 \\ 3 & 5 & 0 & 10 & 11 & 9 & 6 \\ 7 & 9 & 10 & 0 & 2 & 5 & 7 \\ 9 & 7 & 11 & 2 & 0 & 3 & 5 \\ 12 & 4 & 9 & 5 & 3 & 0 & 8 \\ 9 & 8 & 6 & 7 & 5 & 8 & 0 \end{pmatrix}$$

Then we use the improved k-means clustering analysis algorithm to find the two distribution sites:

In the above matrix A, $d(v_i, v_j)$ denotes the Floyd's distance between the vertices v_i and the vertices v_j. Set the clustering category k equals 2, and the clustering algorithm process is as follows:

Step 1: respectively calculates the distance between the vertices v3, v4, v5, v6, v7 and centroid of the v1, v2.

As to centroid of v1, there is $d(v1, v2)=8, d(v1, v3)=3$, $d(v1, v4)=7$, $d(v1, v5)=9$, $d(v1, v6)=12$, $d(v1, v7)=9$.

As to centroid of v2, there is $d(v2, v3)=5$, $d(v2, v4)=9$, $d(v2, v5)=7$, $d(v2, v6)=4$, $d(v2, v7)=8$

Index of similarity is served as the Floyd path, it is easy to know from the shortest path matrix, $d(v1, v2)=d(v2, v1)$, $d(v1, v3)<d(v2, v3)$, $d(v1, v4)<d(v2, v4)$, $d(v1, v5)>d(v2, v5)$, $d(v1, v6)>d(v2, v6)$, $d(v1, v7)>d(v2, v7)$. Therefore, the result of the clustering is C1={v1, v3, v4} as a cluster, and C2={v2,v5, v6, v7} as another cluster.

Step 2: updates the centroid of the cluster C1 and the cluster C2.

According to the formula $\bar{x}_i=\frac{\sum_{x_j\in C_i} d(x_i, x_j)}{|C_i|}$, obtain $Average(v1, x)=\frac{3+7}{3-1}=5$, .where $d(v1, v3)=3$ and $d(v1, v4)=7$, the vertices closest approach to 5 is respectively the vertices of v3 and v4, chooses the vertices v4 at random to substitute the quondam centroid v1 here. In the same way, chooses the vertices v7 to substitute the quondam centroid v2.

Step 3: repeat Step 2, until the v6 as the centroid of the cluster $\{v6, v4, v5, v7\}$ and v2 as the center of the cluster $\{v2, v1, v3\}$, the mean found that the two clusters of the center of mass will not change, so the end of the cluster, cluster and cluster is the final result. The clustering results are showed in Figure 3.

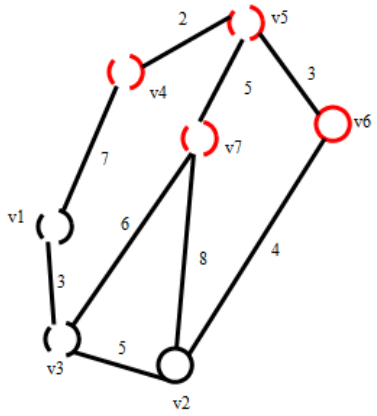

Figure 3: Clustering Results

The redness and the blackness represent there are two clusters, v4, v5, v6 and v7 is one cluster, and v1, v2, v3 is another cluster. The solid rim v6 and v2 are reprehensively the centroid of the two clusters.

Conclusion

In this paper, based on the distribution location center of express company, started at the influencing factors that courier required time is the shortest in the logistics distribution, we apply Floyd algorithm to calculate the shortest time path of courier required, then adopts the improved k-means clustering algorithm to achieve this goal. The customer location acts as the object clustered of the k-means clustering algorithm here, and the *k* input value is in according to the actual situation choice rather than the experience, to avoid the blindness of the optional *k* value.

ACKNOWLEDGMENTS

This paper is granted by NSF of China (61572011) and NSF of Hebei province (A2013201119).

REFERENCES

Bao PM (2001). A Optmization Algorithm based on dijkstra's algorithm in search of shortcut. *Journal of Computer Research & Development* **38**(3) 307.

Bondy JA and Murty USR (2008). *Graph Theory,* (San Francisco: Springer Press, California) 12-30.

Gu W, Zhang Q and Hu R (2010). Research of logistics distribution region partition method based on improved k-means clustering. *University of Science and Technology Beijing* **13**(24).

Kanungo T and Mount D (2002). An efficient k-means clustering algorithm: Analysis and implementation. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **24**(7) 881-892.

Liu GH (2011). *A Dijkstra Distance-based Clustering Algorithm and Application in Logistics*, (China, Lanzhou: Lanzhou University) (in Chinese).

Lu F (2001). Shortest Path Algorithms: Taxonnomy and Advance in Research. *Acta Geodaetical Et Cartographic Sinical* **30**(3) 270-274.

Mao H and Shi M (2016). The application of article *K-th* shortest time path algorithm. *International Journal of Physics and Mathematical Sciences* **6**(1) 24-36.

Wang HY, Huang Q, Li CT and Chu BZ (2010). *Graph Theory Algorithm and its MATLAB Implementation,* (Beijing: Beijing Beihang University Press, China) 20-48.

Yan HB and Liu YC (2000). A new algorithm for finding short cut in a city's road net based on GIS technology Chinese. *Journal of Computers (in Chinese)* **23**(2) 210-215.

Zhang DQ and Wu GL (2009). Optimized Floyd Algorithm for Shortest Paths Problem. *Journal of Xuchang University* **28**(2) 10-13.