基于改进K-means聚类的物流配送区域划分方法研究

谷 炜,张 群,胡 睿

(北京科技大学 经济管理学院,北京 100083)

[摘 要] 在求解大规模的车辆路径问题时,首先需要将大规模复杂的配送网络根据一定的约束条件并利用相应的方法划分为若干个小规模的配送区域,而不同的配送区域的划分方法对最后优化效果影响很大,本文从解决实际问题入手,首先明确了使用聚类算法进行配送区域的划分可以使得到的区域比较紧密且更符合实际需求,然后分析了现有基于K-means聚类算法的优劣性,在此基础上设计了一种新的配送区域均衡的划分方法——改进的两阶段K-means聚类算法,并经过仿真实验验证了方法的实用性和有效性。

[关键词] 物流配送;区域划分;K-means聚类

doi:10.3969/j.issn.1673-0194.2010.24.028

[中图分类号] F274;F224.0 [文献标识码] A [文章编号] 1673-0194(2010)24-0060-04

1 引 言

物流配送区域的划分是一个多约束、多目标决策的组合优化问题,属于NP难问题[1],现行一般求解VRP问题的算法时间复杂度大都为$O(n^2)$,若按此计算规模为10 000个左右节点的车辆路径问题预计至少需要数百小时,因此在求解大规模车辆路径问题上需要寻找合适的解法。目前的解决思想是化整为零,各个击破。即首先将大规模复杂的配送网络根据一定的约束条件并利用相应的方法划分为若干个小规模的配送区域,然后逐个求解小规模车辆路径问题即在各子区域内设计最优配送路线,以此寻找到整个问题的近似较优解。

而在选择划分方法时,不同的配送区域的划分方法对最后优化效果影响很大,所以确定一种比较合理的划分算法对整个系统是非常关键的。目前配送区域划分的主要方法有扫描法和聚类算法。扫描法的基本思路是采用逐次逼近的方法求解物流配送车辆调度问题,一般适用于客户数目不太大且配送区域不多时;而当客户网点数目较大且配送区域较多时,则采取聚类算法,把大量的d维数据对象(n个)聚集成k个聚类($k<n$),使同一聚类内的对象的相似性尽可能最大,而不同聚类内的对象的相似性尽量达到最小。也就是说形成聚类之后,同一个聚类内的对象具有很高的相似性,而且与不属于该聚类的对象有迥然的差异(即不相似)。由于聚类的特殊性,在物流系统中使用聚类算法进行配送区域的划分可以使得到的区域比较紧密且更符合实际需求。

在进行聚类时,主要有以下几方面需要考虑:

第一,聚类的数量。这是一个需要先行输入的数值,需要考虑到实际情况,根据每天总的送货量和单车容量进行计算得出。因为客户的需求每一个周期是不同的,所以每个配送周期开始时都需要重新计算该值。

第二,每个聚类内点应该相对集中,这样车辆就不会花费大量时间在行驶过程中。这也是聚类的基本原理之一。

第三,每个聚类的工作量应大体相当,这样可以避免某配送车辆送货格外少或者格外多的情况出现。

第四,每两个聚类不能有重合,否则会发生重复送货的情况;同理每一个点都必须包含在某一个类中。

第五,在实际应用中只考虑位于城区的客户分布,对于位于郊区的客户,可以利用城区的配送原理进行计算。

2 基于GIS的两点间最短距离的确定

GIS环境下针对配送道路路线进行规划,将着重分析城市的道路网络图层,道路中两个节点间的最短距离的求解就是基于这个图层。在城市道路网图层中进行最短路径分析时,必须首先将现实中的城市道路网络实体抽象化为图论中的网络图的形式,然后通过网络分析来实现道路网络的最短路径分析。在实际应用中,城市道路网的表现形式一般为数字化的矢量地图,其网络空间特征中的交叉路口坐标和道路位置坐标是在地图上借助图形来识别和解释的,而为了能够高效率地进行最短路径分析,必须首先将其按节点和边的关系抽象为图的结构。这就需要对原始道路图进行预处理,建立其相应的网络拓扑关系。经过预处理后,当对城市道路网进行最短路径分析操作时,系统就可以直接从拓扑信息表中提取道路网的网络拓扑结构。将配送仓库和每个客户点之间的最短距离,以及每个客户之间的两两最短距离都保存到数据库中,为求解路径规划问题提供数据。

研究模型与GIS结合时最大的问题在于:不再只是计算两两配送点之间的直线距离,而是利用地理信息系统计算任意两配送点间的实际路线距离以及这条最短路径所经过的道路节点。

用$G=[C',E,V,W]$表示网络图[2],其中:$C'=\{0,1,2,\cdots,n\}$为点集,0表示配送中心,其他点为客户;V为路径交点集合;$E=\{(i,j)\mid i,j=0,1,2,\cdots,n,且\ i\neq j\}$为弧集,表示车辆可能走过的路线段集合;$C=\{c_{ij}\mid(i,j)\in A\}$为距离矩阵,$c_{ij}$表示经过对应弧段$(i,j)$的长度;$W$为正的实数集($W:E\rightarrow IR^{+}$),表示对应边的权值。图$G$的一个边序列$\{e_i,e_{i+1},\cdots,e_j\}$或者说一个点序列$\{c'_i,c'_{i+1},\cdots,c'_j\}$称为图从点$c'_i$到点$c'_j$的一条路径。在点$c'_i$到点$c'_j$所有路径中,权$W(c'_i,c'_j)$最小的路径称为最短路径。

无论是距离最短、时间最快还是费用最低,它们的核心算法都是最短路径算法。在这里假设道路的通行质量是相同的,而且不随时间变化。那么,行车时间和距离之间就存在着简单的正比关系。要计算两点之间的行车时间,只要知道两点之间的距离就可以了。这样问题就集中在求两点之间的距离。两点之间的距离

[收稿日期] 2010-10-08

[作者简介] 谷炜(1982-),男,辽宁锦西人,北京科技大学经济管理学院博士研究生,主要研究方向:运营管理、企业信息化管理;张群(1950-),男,湖北黄冈人,北京科技大学经济管理学院院长,教授、博士生导师,主要研究方向:生产与运作管理,技术经济评价。

是指两点之间最短路径的长度，两点之间的路径可以有很多条，在安排行车路线的时候，必须保证车辆走最短的路程，这是总成本最低的保证，同时也是总路径最短的保证。那么求解任意两点之间的最短路径就成了解决物流路径规划问题的基础。

3 目前主要的 K-means 聚类划分方法分析

3.1 K-means 算法简介

K-means 算法是聚类思想中最早出现的算法之一，思想很成熟。它接受输入量 k，然后将 n 个数据对象划分为 k 个聚类以便使得所获得的聚类满足：同一聚类中的对象相似度较高；而不同聚类中的对象相似度较小。聚类相似度是利用各聚类中对象的均值所获得一个“中心对象”来进行计算的。K-means 算法由于技术成熟，简单易懂，在数据挖掘中被广泛应用，在实际应用中涉及聚类的也大多选用该算法。

3.2 Weighted-K-means 算法

文献[3]提出了 Weighted-K-means 算法这一改进方法。该方法根据物流配送中的实际情况，给每一点都赋以各自的权重。在聚类的时候，将每一类的权重作为一个约束条件，从而使得每一个类的权重大体相等。该算法的主要思想是：假设在做距离比较时不再采取 1:1 的比率，而是采用和权重有关的度量方式，假定点 d_i 其到聚类 C_k 中心 c_k 的划分距离为：

$$d_i = \frac{\sqrt{(x_k - x_i)^2 + (y_k - y_i)^2}}{\text{weight}(C_k)^{\gamma}}$$

上式中，x_k 和 y_k 是聚类中心的坐标，$\text{weight}(C_k)$ 表示对点集合 C_k 求其总权重。γ 称之为权重衰减系数。可见到，当 γ 为 0 的时候，上式和普通聚类的划分距离公式其实等价。可以将不考虑权重的 K-means 聚类认为是权重衰退系数为零的 Weighted-K-means 聚类。

Weighted-K-means 方法的优点十分明显：首先，权重作为外部约束条件进行约束，聚类划分的次数会大大减少。而且这样进行的聚类得出的结果更好。其次，在实际问题中，权重的引入限制了每个聚类的送货量，可以平衡各个聚类之间的工作，提高配送效率。

该方法也有一些不足：第一，虽然时间复杂度和原始的 K-means 算法一样，都是 $O(nkt)$ 这个数量级（k 为聚类个数，n 为参与的点数，t 为迭代次数），但是 Weighted-K-means 的距离公式较为复杂，如果和适当的初始点选择算法配合的话，每次运算的时间也会比较长。第二，算法的鲁棒性有待研究。该算法由于没有在实际中运用过，现实中的问题对它会产生怎样的影响还不确定，而各种可能出现的问题会对它的聚类结果产生不良影响。

3.3 基于均衡化函数的 K-means 算法

文献[4]中提出了一种基于 K-means 算法的改进算法——基于均衡化函数的方法。在这种算法中，初始中心点是根据密度定义的。基于密度的初始化中心点算法的基本思想是从样本数据库中搜索出高密度集点 H_d，然后在 H_d 中利用贪心算法搜索散布较大的中心点集 M。定义如下：

已知样本事务数据 $D = \{x_1, x_2, .., x_n\}$，其中，对象 X_i 的密度记作 $\text{density}(X_i)$，即 $\text{density}(x_i) = |\{x \in D | \text{dist}(x_i, x) \leqslant R\}|$

其中，$\text{density}(x_i, x)$ 表示对象 x_i 和 x 的距离；R 代表半径，且半径的确定问题较为困难，过大或过小均不能获得有效的密度估计，本文采用如下邻域半径公式：

$$R = \frac{\text{mean}(D)}{n^{\text{coef}R}}$$

其中，$\text{mean}(D)$ 表示所有对象间距离的平均值；n 是对象数目；$\text{coef}R$ 是邻域半径调节系数，$0 < \text{coef}R < 1$，经验表明，$\text{coef}R = 0.13$ 时，聚类效果较好。

该算法的主要优点如下：初始点根据密度进行选择，可以消除传统算法中随机性对聚类结果的影响。另外，均衡化函数能很好地反映出类内差异和类间差异的关系，使得聚类的质量有很大提高。但它也存在一些不足，例如初始点的选择利用密度函数计算十分复杂，需要遍历每一个点，每收集一个初始点就要对高密度点集 H_d 进行一次遍历，这会消耗大量的时间。另外该方法也没有考虑到外部约束的问题，使各配送区域间的工作量无法均衡，使聚类结果更加符合实际配送工作要求。

3.4 有弹出点机制的 K-means 算法

有弹出点机制的 K-means 改进算法的基本思想是在进行一次传统的 K-means 之后，计算每个聚类的送货量是否超过规定的载货量，如果超出则弹出其中一个点，这个点通常选择距离聚类中心最远的点。然后再次进行聚类和衡量，不断进行这个过程直到没有点被弹出为止。为了防止一个点总是被弹出，该方法对被弹出一次的点加以标记，下次再遇到则不处理该点。

该算法的优点在于考虑了权重因素，并且对聚类内点的调整给出了一种方案，在实际应用中可以根据实际情况进行调整。该方法的缺点在于在防止节点多次被弹出的时候，使用了一个记录表来记录一个节点是否已经被弹出，而被弹出的节点可能是次远的节点。这样就可能导致聚类交叉的现象，导致实际中会使得车辆走冤枉路，两辆或者多辆车子会走同一条路。为了防止聚类抖动，强制制定了被弹出节点不能再次被弹出的规则。虽然可以通过这条规则防止聚类抖动，但是在最坏情况下，所有点都被弹出后，则聚类恢复为普通 K-means 聚类，并不会带来任何改进。

4 两阶段的 K-means 算法

综合上述 3 种改进算法，本文提出了一种新的改进算法。

图 1 为传统 K-means 算法的详细流程，由聚类过程可以看出，传统 K-means 存在一些不足：

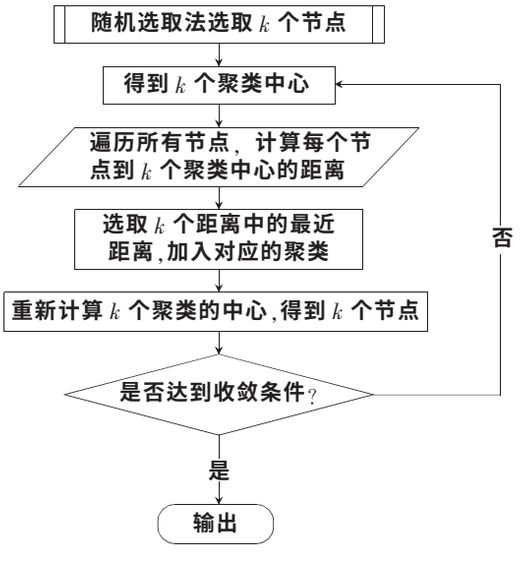

图 1 传统 K-means 算法流程

首先，它的初始点是随机选择的，这就有可能导致收敛速度过慢；其次，它的聚类完全是空间距离上的考虑，没有任何实际因素的影响；第三，原始的 K-means 算法没有任何约束条件，不

能灵活地依据各种限制进行调整。

对应到实际问题中，这几点不足体现为：第一，初始点如果随机选择，有可能集中在某一区域内，要将所有点都加入这些聚类并达到收敛条件很难；第二，这里关注的只是每一个点的经纬度，其他指标都被忽略了，而在实际问题中，载货量、时间等因素都是需要考虑的；第三，这一算法并不能满足每一个划分的工作量均衡的要求。

针对这几点不足，对 K-means 算法进行了如下改进。

4.1 两阶段的 K-means 算法思想

从所配送客户的大致分布和实际路况来看，客户分布有明显的集群特点，而且在配送过程中，对每个区域的工作量有一定的限制，例如运输量、总时间等。为了满足在实际配送过程中需要满足的约束条件，本文对 K-means 算法进行了扩展和改进。

改进后的算法中，整个聚类过程被分为两部分：第一部分就是传统的 K-means 聚类过程，经过数次迭代，将对象点划分为 k 个类。每个类内部的点之间相似性较高，而类与类之间的点相似性尽可能低；第二部分是调整过程，利用给定的工作量衡量指标，对第一部分完成后形成的 k 个聚类进行调整，最终目的是使各个类的工作量大致相同。关于不合格点的弹出，利用标识机制对其进行标识，使得每一个点只有一次被弹出的可能，从而防止了无限次循环迭代，最终导致算法不收敛的情况发生。映射到实际工作中，第二部分的目的就是每个配送区域的工作时间大致相同，工作量基本持平。

改进的 K-means 算法流程见图 2，可以看出到第 4 步的时候，传统意义上的聚类迭代过程已经结束。接下来的步骤是用来根据对工作量指定的衡量指标进行调整，以达到各个类之间工作量相对均衡的状态。

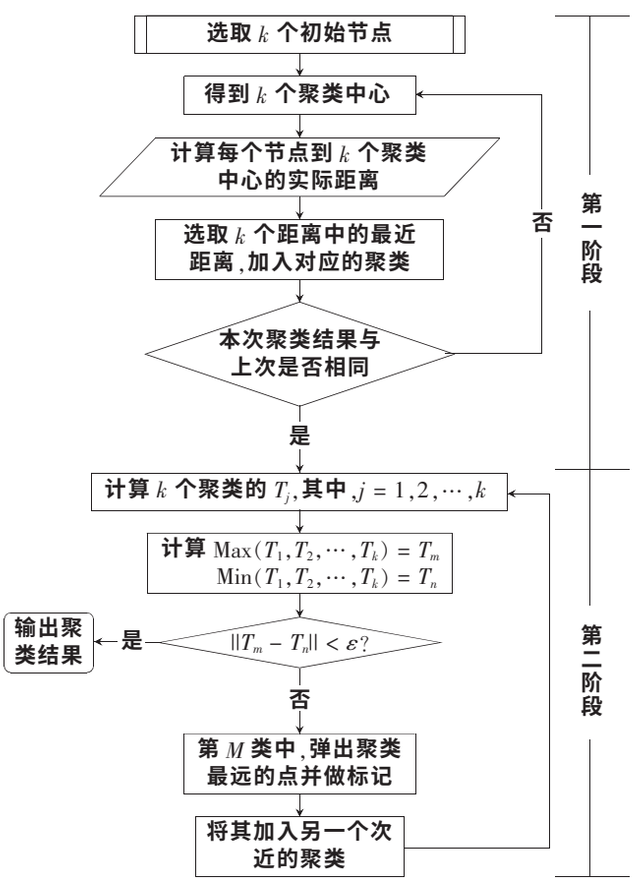

图 2 改进的 K-means 算法流程

4.2 改进的两阶段 K-means 算法

两阶段的 K-means 算法的具体实现步骤如下：

4.2.1 初始 k 值确定

对于初始 k 值，按照如下公式确定：

$k = Q / q$，其中 Q 表示这一配送周期中，需要配送的总量。q 表示每辆车的最大载货量。

4.2.2 初始聚类中心的确定

选择合适的初始点可以使算法快速收敛，而不恰当的选取会大大延长运行时间。初始点的选择常用方法有两类：随机选择法和最大距离选择法。

随机选择法是根据最终聚类中心的个数 k 来随机选择 k 个节点作为初始的聚类中心。传统的 K-means 算法就是从 n 个数据对象中任意选择 k 个对象作为初始聚类中心的。这种方法简单容易实现。但是具有以下缺点：

(1) 节点的选择完全随机，碰到初始节点过于集中的可能性很大。这样距离收敛的速度过慢，甚至可能不收敛。

(2) 因为节点的选择随机，所以节点稀疏的地方被选出初始节点的概率太小，可能使节点分布不均匀，也不利于聚类收敛。

而最大距离选择法基于这样一种假设：聚类初始点之间距离越远，聚类收敛越快。因此在初始点的选择上，就要尽量拉开彼此的距离。它主要包括边缘选择法和均分选择法两类。

这两种方法的收敛效果很接近，由于实际情况中的点分布聚集性较为明显，且大部分都在区域内部，因此采用均分选择法。

均分选择法的思想是：假如需要初始点的个数 $k = pxq$，则将整个地图划分为 p 行 q 列，初始节点选在每一个区域的正中心附近的节点。这样每个节点之间的距离接近于均匀，有利于聚类收敛，而且不会形成很明显的分布不均匀的起始状态。初始 k 值和聚类中心的确定流程如图 3 所示。

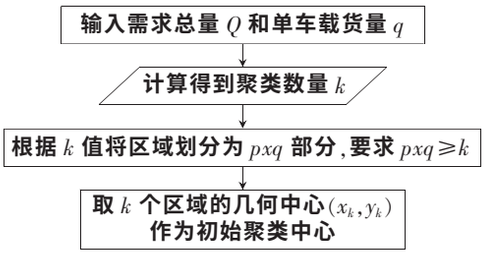

图 3 初始 k 值和聚类中心的确定

4.2.3 计算距离

K-means 要求第二步对于除了聚类中心以外的其他对象，根据它们与这些聚类中心的相似度(距离)，分别将它们分配给与其最相似的(聚类中心所代表的)聚类。

在计算距离的时候需要结合实际情况来考虑。在实际运输过程中，并不是单纯利用欧氏距离公式来进行计算的，因为有些点虽然几何距离很近，但之间没有路可达，这样即使被分到一个聚类中也不能因此提高配送效率。所以在实际操作时，采用数据点映射的方式把所有的数据点都标在实际地图上，利用地图上的城市公路信息计算两点间的距离。主要思想是在 GIS 中将点映射到实际地图上之后，标注上一步确定好的聚类中心，在聚类中心邻域内选择距离最近的公路上的一点，遍历所有点并对每一个非中心点计算它到 k 个聚类中心的实际道路距离；在计算

得到的 k 个距离中，选择最小的一个 d_i，加入该对应聚类，直到所有点都划分进某一个类。

4.2.4 第一阶段收敛性检验

K-means 算法要求在第三步计算每个所获新聚类的聚类中心（该聚类中所有对象的均值）；不断重复这一过程直到达到收敛条件。对于上述步骤运算得到的 k 个聚类进行检验。如果已经满足收敛条件，则继续第二阶段的调整；如果不满足，则根据重心法重新计算聚类中心，再次聚类，不断迭代，直到算法收敛为止。

第一阶段的收敛准则是：第 m 次聚类的结果和第 $(m-1)$ 次结果相比，没有任何变化。包括聚类中心稳定，构成各个聚类的点集不再变化。

4.2.5 均衡指标计算

首先，需要计算均衡工作量指标 T_i，这一指标的现实意义是帮助各个聚类之间实现工作量的均衡。T_j 代表第 j 个聚类的 T_{area} 值。从计算出的 k 个 T 中，筛选出最大值 T_m 和最小值 T_n，由于划分的目的是每个配送区域工作量均衡，也就表示要将最大值 T_m 和最小值 T_n 的差控制在合理范围之内，ε 表示可接受的残差，该数值是在运算前人工输入的且取值大小取决于可接受的工作量差异。

因此，第二步的收敛条件为 $\|T_m - T_n\| \leqslant \varepsilon$。当这个条件满足时，就表示现在的区域划分能够满足工作量基本均衡的条件，而且第一步聚类也满足了能使客户安排相对集中，送货效率较高的要求。如果不满足上述收敛条件，也就是有的区域工作量过大，就需要进行如下的调整。

4.2.6 点集的调整

如果前一步的检验没有通过，那么就需要对现在的聚类结果进行一些调整，$\| T_m - T_n \| \leqslant \varepsilon$ 不成立意味着在此时的聚类中，工作量极差较大。也就是说，有些配送区域的送货时间过长，而有些则只要很短的时间就可以完成任务，而希望的结果是各个配送区域需要的配送时间基本相等。因此采取如下调整方法：在有最大 T 值 T_m 的点集中选取与聚类中心距离最远的那个数据点，并将其弹出该聚类，这样该聚类的 T 值就会下降。同时将弹出的点加入到另一个聚类中，此时从该点的 k 个 d_j 中选择次小的，找到该距离对应的聚类中心，将这个点加入到该聚类中心所在的数据类中。

为了防止某一个数据点被反复弹出，因此对被弹出的点加以标记，当下次扫描又需要该点弹出时识别到特殊的标记，则对该点不予处理，转而弹出聚类中次远的点。这种机制可以防止某一点无法加入任何一个聚类，也就是说不会出现为某一家客户单独送货的情况。

4.2.7 重新迭代

在对点集进行调整之后，有两个类的 T 值发生了改变，对这两个聚类重新计算 T 值再次检验；然后不断迭代，直到所有配送区域间的工作量指标都基本均衡，极差也处在可接受的范围之内；这时停止迭代，将聚类结果以点集形式输出。

两阶段的 K-means 算法的具体步骤总结如下：

Step 1：选取 k 个初始节点作为聚类中心。

Step 2：在 GIS 中计算每一个点与 k 个聚类中心的距离。(x_k, y_k) 用于标记某个中心点坐标，(x_i, y_i) 为第 i 个点的坐标。在这里坐标值是用点的经纬度表示的，并存储在数据库表中。

Step 3：对于每一个非中心点，在计算出的 k 个距离中选择最小的一个，并将该点加入对应类。

Step 4：通过 K-means 算法进行聚类，得到 k 个聚类。计算每一个聚类 T 值，T_j 为配送区域内总工作时间，j 取值为 $1,2,\cdots,k$。N_1 为配送区域内电子结算客户数量，N_2 为配送区域内非电子结算客户数量。D_1 为配送车辆由配送中心到第一个客户的距离加最后一个客户到配送中心的距离，D_2 为配送车辆配送过程中经过的各配送点的距离和。

Step 5：对 T_j 进行排序，得到 $\max T_j = T_m$，$\min T_j = T_n$。其中 $m, n \in (1,2,\cdots,k)$。

Step 6：输入残差 ε，设定收敛条件为 $\| T_m - T_n \| \leqslant \varepsilon$，若符合则结束；否则下转第 7 步。

Step 7：将第 m 类中距中心最远的点弹出并加以标记，在今后的循环迭代中遇到此标记则不予处理，而转向处理次远的点。将该点加入到除 m 外距它最近的类中，重新计算 T_k。转向 Step 5 继续检查，直至收敛。

5 结论及展望

两阶段的 K-means 改进算法是对传统 K-means 以及一些改进算法的综合与改进，它克服了传统 K-means 不够灵活和无法添加外部约束的缺点，结合了集中改进算法中关于权重、均衡、弹出点机制、添加外部约束条件等思想，同时根据实际情况给出了初始聚类个数 k 的确定方式以及聚类中心的确定。具体的主要优点如下：

第一，在初始点的选择上摒弃了传统的随机选择方式，而采用网格式选择。虽然这种方法比随机选择要复杂，但是它能够使初始点分散，使算法更容易收敛，同时防止出现初始点过于密集的情况。

第二，算法更容易实现。在第一阶段迭代过程中，采用的是经典的 K-means 算法。像 K-means 这样比较成熟的算法，已有固定的实现模式，因此实现起来比较方便。

第三，在第二阶段引入了外部调整量 T 来衡量各个聚类的工作量，以保证每个配送区域所需要做的工作量大致相同。衡量指标的各个系数则是通过数据统计分析总结而来，该指标源自实际数据，在应用中更加可靠。

同样，该算法也存在几处需要继续研究和改进的方面：

第一，停机准则中残差 ε 的确定需要继续研究，太严则可能算法无法收敛，太宽松则聚类效果有可能不够好。

第二，这种方法对点分布较为平均的情况更加实用，如果点的分布具有较强的集中性特征，则效果可能不如利用密度聚类。

第三，该算法并未对道路约束条件进行考虑，需要进一步实验和改进。

主要参考文献

[1] K M Van Hee. Information System Engineering: A Formal Approach [M]. NewYork, NY: Cambridge University Press, 1994.

[2] 史亚蓉，万迪昉，等. 基于 GIS 的物流配送路线规划研究[J].系统工程理论与实践，2009，29(10)：76-84.

[3] 谢可.物流配送系统中聚类算法的研究与应用[D]. 杭州：浙江大学，2006.

[4] 钱雪忠，施培蓓，张明阳，汪中.基于均衡化函数的 k 均值优化算法[J].计算机工程，2008，34(14)：60-62.