分布式最小生成树聚类的设计与实现①

金 欣，王 晶，沈奇威

(北京邮电大学 网络与交换技术国家重点实验室，北京 100876)
(东信北邮信息技术有限公司，北京 100191)

摘 要：聚类是数据挖掘的主要问题之一，聚类算法能够在没有任何数据先验知识的情况下对数据进行分群，从而找到数据中的有价值的信息。近年来数据挖掘在电信领域的应用越来越广泛，但是由于数据量、数据类型、计算复杂度等原因，聚类算法应用的却不多。提出一种新的适合于分布式计算的最小生成树算法，结合适合的相似度度量，设计了一种用于解决海量数据分析的分布式聚类算法，并给出了基于 mapreduce 编程模型的分布式实现。

关键词：聚类；分布式；hadoop；mapreduce；数据挖掘；最小生成树

Design and Implementation of Distributed MST Clustering

JIN Xin, WANG Jing, SHEN Qi-Wei

(State Key Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications, Beijing 100876, China)
(EB Information Technology Co. Ltd., Beijing 100083, China)

Abstract: Clustering is one of the most important problems in data mining. Clustering algorithm can classify data without any knowledge about it, and find out the information that valuable. Recently, data mining is more and more widely used in the telecommunication area, but because of some problems, such as the size of the data, the type of the data and the complication of the computation, clustering is not used widely. This article gives a MST algorithm that suit for distribute computing. Combining with the method to represent the similarity that suitable for this algorithm, it designs a new clustering algorithm to solve the problem of sea size data analysis. Then, it shows how the algorithm is realized based on the distribute computing model called mapreduce.

Key words: clustering; distribute; hadoop; mapreduce; data mining; MST

1 概述

聚类问题是数据挖掘的主要问题之一。简单来说，数据挖掘就是在庞大的数据中寻找出有价值的隐藏信息，为企业的问题建立不同的模型，为企业决策提供依据。聚类算法能够在没有任何数据先验知识的情况下对数据进行分群，将更加相似的数据聚集在一起，通过对聚类结果的分析，可以找到数据中隐含的模式并将有用的信息提取出来，作为今后决策的依据。

随着电信运营商之间的竞争愈发激烈，正确的商业策略成为提高竞争力的关键环节。电信运营商拥有海量数据信息，利用数据挖掘技术，可在海量用户数据中发现商业知识，为市场的精准营销打下基础。现在的数据挖掘在电信领域的应用主要集中在客户流失分析、业务推荐等方面，主要使用如分类、协同过滤、关联分析等算法，而聚类算法应用的并不多，原因主要有如下几个：

1. 聚类算法是一种无监督的分群算法，不像分类算法可以使用训练集，因此面对海量的用户数据，计算量是难以承受的；

2. 移动用户属性主要包括用户属性、业务属性等

① 基金项目:国家杰出青年科学基金(60525110);国家 973 计划(2007CB307100,2007CB307103);国家自然科学基金(60902051);中央高校基本科研业务费专项资金(BUPT2009RC0505);电子信息产业发展基金

收稿时间:2010-11-03;收到修改稿时间:2010-12-15

等，这些属性主要是以离散属性为主，采用传统的如 k-means 等基于距离的聚类算法聚类效果不佳，而适合离散属性的基于图形聚类算法计算复杂度普遍很高，进一步增加了计算难度；

3. 随着业务的增加和复杂化，移动用户的属性数量越来越多，使得用户聚类问题更加困难。

本文将给出一种基于 hadoop 分布式计算平台的可扩展聚类算法实现，该算法首先实现了基于 map-reduce 计算模型的分布式最小生成树算法，并基于这个最小生成树算法，设计并实现了分布式最小生成树的聚类算法，为电信领域的海量用户分群问题提出了新的解决方案。

下面的论文主要内容包括：第二章描述聚类研究现状以及存在的问题；第三章给出算法的基本思想和步骤；第四章简单介绍基于 mapreduce 的分布式计算模型，然后给出基于该模型的分布式聚类算法实现；第五章给出试验结果并对后续研究进行展望。

2 聚类研究现状

聚类问题一直是数据挖掘研究的一个重要课题，根据算法实现的策略，聚类算法可以分为基于划分的方法（k-means[1]），基于层次的方法（BIRCH[2]、CURE[3]、ROCK[4]、CHAMELEON[5]），基于密度的方法（DBSCAN[6]、OPTICS[7]）等等。这些方法各有利弊，共同的缺点是计算复杂度相对较高，当数据量急剧增大的时候性能下降很快。而基于网格的方法（STING[8]、WaveCluster[9]、CLIQUE[10]）通过将数据点分布在划分为一个个网格的空间中，通过用对网格的聚类代替点聚类的方法解决了数据量增大的问题，但是基于网格的聚类算法缺点也很明显。首先它丢失了很多数据信息，使得聚类结果失真问题比较严重，其次由于随着数据空间维度的扩大，数据在空间中的密度会越来越小，因此基于网格的聚类算法不适合高维数据聚类。

对于有着大量离散属性的数据聚类是聚类研究的另一个重点。因为传统的基于距离的聚类方法在处理离散数据上效果都不令人满意，无论是欧式距离、马氏距离，还是用于计算点簇之间距离的各种方式，都会导致结果的严重偏差。通常离散数据聚类采用关联性等度量标准进行聚类，而基于图形的聚类是对离散数据进行聚类的重要手段。基于图形的聚类算法有着计算复杂度高，可扩展性差的缺点。基于图形的聚类方法主要属于分层聚类，分为合并和分裂两种方式。其中合并的方式初始的时候将每个点看作一个集群，每次迭代将距离最近的点合并为一个集群；分裂方式将所有点看作一个集群，然后不断的将其划分为多个集群，划分的标准是使得集群内的边尽量密集，而集群间的边尽量稀疏。这两种方式的计算复杂度通常要达到 $o(n^3)$ 以上。

METIS[11]是一个用于进行图形划分的包，是现在最流行的也是速度较快的图形划分工具。文献[6]中对该包进行了详细的性能和结果分析。该算法采取了合并、划分再展开的方式来解决大数据量的图形划分问题。合并时采取一定的策略将多个点合并成为一个点从而减小图的规模；划分是指在合并后的点上将整个图划分成为多个子图；展开是指将合并后的图还原成为原图并对划分结果进行一定的调整。经过对算法的分析和测试发现[12]，合并对于正确划分整个图的影响随着合并的次数急剧恶化，也就是说，当数据量很大的时候，会导致多次合并，而多次合并会严重影响图形划分结果。因此该算法只适合较大数据量集的图形划分，而不适合海量数据的划分。

MST（最小生成树）是经典的基于图形的聚类算法之一，他利用最小生成树极大的简化了基于图形聚类算法的计算复杂度。在生成了一个图的最小生成树后，通过去掉最小生成树中距离最大的边，可以将整个数据集划分成为多个点集，从而达到聚类的效果。MST 算法本身可以作为聚类算法来使用，也可以作为其他算法（如 CHAMLENE 算法）的预处理算法，先将整个集群划分成为多个小集群，再用其他基于图形的算法进行处理。

MST 算法的缺点有两个，一是 MST 算法由于通过去掉最小生成树的边来划分集群，所以可能会导致两个应该被分离的集群由于两个点的紧密相连而被合并在一起；二是 MST 算法本身又引入了一个计算复杂度较高的算法，就是最小生成树的生成。如何高效的生成图的最小生成树是又一个需要解决的问题。

近年来分布式技术的发展为海量数据的聚类分析提供了新的思路，通过将海量的数据分布到不同的节点上进行并行处理，可以大幅的提高运算效率。这种计算方式解决了许多领域的海量数据处理问题[13]，利用这种方式来解决数据挖掘问题，可以使得一些复杂

度较高的算法能够经过改造用于解决大数据量的聚类问题。但是并不是所有的算法都可以进行分布式计算，只有保证数据在计算过程中互不影响，也就是说只有在数据是可切割的情况下才可以进行分布式计算。如何将复杂的聚类算法分解、改造成可以分布计算的模式是在利用分布式技术解决海量数据聚类问题过程中必须要考虑的问题。

3 算法思想和基本步骤

本文给出的算法主要是为了解决移动用户分群问题而提出的，下面将结合实际问题给出算法的基本思想和步骤。

3.1 相似度度量选择

对于一个聚类算法而言，相似度度量的选择尤为重要。所谓相似度度量，就是描述两个用户的相似程度的标准，结合我们的问题，就是说两个移动用户的相似程度描述。移动用户的属性特点有三个：离散、多维、稀疏。离散是指属性大多是离散属性，例如是否订购某业务，是否经常漫游等等；多维是指属性的数量很多，而稀疏是指用户具有的属性（例如订购某业务）和总属性的数量相比相对较少。这种数据类似购物篮分析的数据类型，这种数据类型如果采用常规的相似度度量会产生较大的偏差。

参考文献[4]中描述了这种偏差的存在和原因，并给出了一种适合这种数据分析的相似度度量标准，该度量结合 Jaccard 系数给出了邻居的概念，并用共有邻居数来描述两个点的相似程度。假设数据 A 具有的属性集合为 T1,数据 B 具有的属性集合为 T2，Jaccard 系数的描述为 $\frac{T1 \cap T2}{T1 \cup T2}$，我们用 J(θ)表示。J(θ)大于某个阈值的两个点被认为是邻居，而文献[4]中用两个点公共邻居的个数作为描述两个点相似程度的度量。该度量方式的优点是不仅描述了数据之间的相似性，还考虑了数据所处的环境。但是这种方式有一个缺陷，那就是存在两个数据没有一个相同的属性，但是他们却有可能相连，甚至有可能相连的距离很近。

本文对该度量方式进一步改进，提出了一种更加适合聚类分析的相似度度量方式，即两点之间相连的权重用公共邻居个数乘以 J(θ)来描述，这种方式一方面避免了完全没有相同属性的点相连，另一方面是的连接紧密的点更加紧凑，连接较远的点更加稀疏，有利于后续聚类计算的准确性提高。

但是是否采用直接相乘的方式呢？在我们给出的相似性计算中，有两种因素会影响最终结果，一是就 Jaccard 系数，二是共同的邻居个数，前者会较偏重于属性少的数据，而后者会偏重于属性多的对象。Jaccard 系数的取值范围是 0 到 1，而公共邻居数通常是比较大的，因此直接相乘会导致属性多的点被优先合并，Jaccard 系数对结果的影响非常小。

因此我们应当对公共邻居数进行归一化处理。在归一化过程中,我们首先计算出具有最多公共邻居的两个点,设公共邻居为 n,然后对每两个点之间的公共邻居数取对数再除以 $\log(n)$，从而达到归一化的效果。选择这样的策略是为了防止因为有个别公共邻居数很多的点对使得所有的描述公共邻居的值变得很小。

综上所述，设两个点的公共邻居数为 neighbourCount，所有点对之间的最大公共邻居数为 maxNeighbourCount，则生成图中边的权重为 $J(\theta)*\frac{\log(neighbourcount)}{\log(\max neighbourcount)}$。

3.2 最小生成树算法

关于最小生成树有如下定义：

a) 一个连通且无回路的无向图称为树。

b) 若图 G 的生成子图是一棵树,则该树称为 G 的生成树。

c) 在图 G 的所有生成树中,树权最小的那棵生成树,称作最小生成树。

Kruskal 算法和 Prim 算法都是经典的最小生成树算法，这两个算法都是采用了贪婪算法的思想，从一个点出发不断找到最合适的点归并到生成树当中，最终生成完整的最小生成树。显然这两种经典算法都不能并行处理，不适合解决大数据量的最小生成树问题。

文献[14]给出了一种基于消息传递的最小生成树方法，这种方法将每一个点作为一个结点自主的维护自身状态，然后通过消息与邻近结点进行通信，通过自身状态和消息交互最终找到每个点上存在于最小生成树中的边。这种方式需要将每一个点作为一个进程或者线程操作，当数据量急剧增长的时候，消息交互和同步等操作会出现很多问题，因此不适合大数据量的处理。

在最小生成树理论中，有一个非常重要的定理，

就是一个点的最小边一定是在最小生成树中的，此定理在文献[14]中有详细证明。因此本文提出一种可并行的最小生成树生成方式，下面描述了这种并行处理方式的过程：

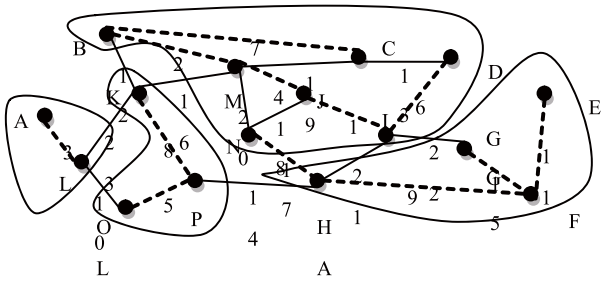

图 1　最小生成树算法实例图

图 1 描述了一个简单的加权无向图。为了找出图 1 中的最小生成树，我们首先标注每个点的最小边，图 1 中虚线表示了这些被标注的边，根据定理，这些边一定是在最小生成树当中的。这些被标注的边将整个图划分成了多个子图，在图 1 中已经将每个子图圈在了一起。将每个子图作为一个点，点的标识选择子图中度数最大的点。这个样子我们可以得到一个新的图。因为两个子图之间可能会有多条边，所以我们从这多条边中选取最短的一条作为新图中的边，于是我们可以得到一个新的图，如图 2 所示：

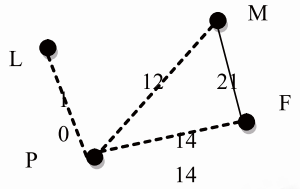

图 2　最小生成树算法一次迭代后的结果

对该图再进行一次上述相同操作就可以得到所有最小生成树中的边。如果仍然会将图划分为多个子图，那么就继续进行迭代。

这种生成最小生成树的方式的优点在于每个点单独进行处理，这样子数据就是可分割的，从而可以进行分布式计算。由于每次迭代都会减少至少一半的点，所以 n 个点的最小生成树最多进行 $\log_2^n$ 次迭代，但是大多数情况下迭代次数会远远少于这个数，在一次实验中 13 万个点共迭代 5 次找到所有最小生成树中的边。

根据第 3.2 节描述，MST 算法还有一个缺点，就是会由于特殊点导致不正确的聚类划分。但是由于我们的相似性度量不仅考虑了两个点之间的相似性，还考虑了周围环境的相似性，所以很自然的解决了这个问题。

3.3 基于最小生成树的聚类划分

在生成最小生成树后，我们要做的是去掉最小生成树中距离最远的边。首先我们需要将所有最小生成树中的边进行排序，然后按照某中策略去掉一些边，策略的选择可以是去掉距离大于某阈值的边，去掉一定数量的边，或者是要求每个聚类的大小小于一定值。

当去掉一条边的时候对整个数据的划分是我们要做的另外一项工作。如何确定一个数据是在哪个集群里呢？在去掉了最小生成树之外的边后，整个图实际上已经划分成为多个互不连通的子图，只要再做一次广度优先并用统一的值标识在同一个子图中的点，就可以将整个数据集划分成为多个聚类。

3.4 算法步骤和瓶颈

算法过程总共分为三个大的步骤：

1. 计算用户相似度并以每个用户作为一个点构建一个图；

2. 计算生成图的最小生成树；

3. 根据最小生成树划分集群。

算法瓶颈主要在两个方面，一是图的生成过程，这个过程需要所有的点对进行两次比较，复杂度均为且直观上不可分布式计算；二是最小生成树的生成过程。

4　基于mapreduce的算法实现

mapreduce 是一种计算模型，每个 mapreduce 分为一个 map 过程和一个 reduce 过程，map 和 reduce 的输入输出数据都是采用键值对的方式进行描述，map 会对每一个键值对进行处理，处理结果用键值对的形式进行输出，输出的结果相同 key 值的会汇总到一起由 reduce 进行处理。详细的 mapreduce 编程模型介绍可以参照文献[15]。

	a	b	c	d
A	1	1	0	1
B	1	1	0	0
C	0	1	1	1
D	0	0	1	1

图 3　示例数据

下面将通过一个简单的例子来描述计算过程。假设有如图 3 四条数据，每条数据有四个属性。对这四条数据的聚类过程将经过三个步骤，每个步骤由多个 mapreduce 过程组成。假设该过程中最多由四个 map 和四个 reduce 组成。

步骤 1：生成图

MapReduce1.1：如图 4 所示，程序输入是用户数据，在 map 中会以用户具有的每个属性为键输出用户信息，用户信息中包含用户的 id 和属性数。reduce 中将具有相同属性的用户两两输出，键值有字母排序靠前的数据 id 表示。运行结果是输出所有具有相同属性的用户对，如果两个用户有三个属性相同，那么他们就有三个点对。设结果为 result1.1

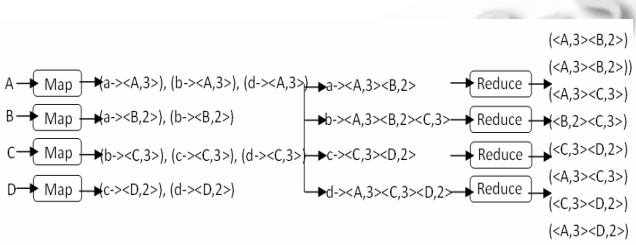

图 4 MapReduce1.1 过程描述图

MapReduce1.2：如图 5 所示以 mapreduce1.1 的结果为输入，map 中不做任何处理将输入数据输出到 reduce 中，reduce 中接收到的键是点的 id，值是所有与该点有相同属性的点的 id 和该点具有的属性数量。有几个相同属性，就有几个该数据的记录。在接收到数据后，会计算这些点之间的 jaccard 系数并去掉 jaccard 系数少于某阈值的关联。假设阈值为 0.4，那么例子中 A 和 D 的关联将被去掉。

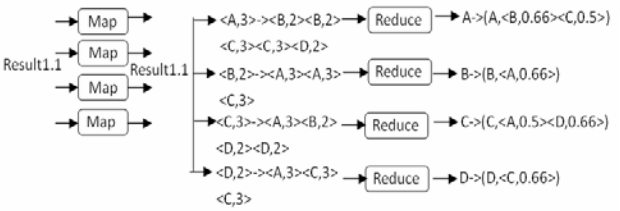

图 5 MapReduce1.2 过程描述图

经过如上两步 mapreduce 过程，我们相当于生成了一个图 6 中的 G1，jaccard 系数比较大的点对有边相连。再用同样的方式经过如上两步操作，不同的是将属性换成这个图中的邻居，这样我们就可以生成一个图 6 中的 G2，这个图描述的是所有有公共邻居关系构建的图。因为实例比较简单，所以公共邻居数对权重没有产生影响。将这两个图作为输入，根据之前提到的计算方式进行一次计算就可以得到用于聚类的结果图，结果图和 G1 一样。

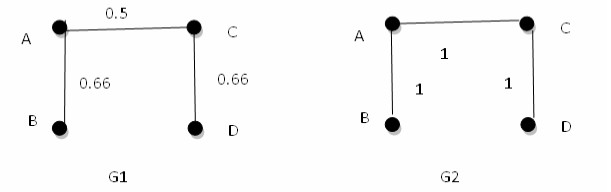

图 6 生成图过程的中间结果描述图

步骤 2：生成最小生成树：

MapReduce2.1：如图 7 所示，选取每个点的最小权重边，并去掉所有的其他边。由于 map 到 reduce 过程中的排序不会影响结果，所以程序描述图中没有给出排序过程。这个过程还会将所有选出来的最小权重边。

图 7 MapReduce2.1 过程描述图

很明显这个时候剩下的边将整个图分成了两个子图。通过广度优先遍历，可以用不同的标识区分不同的子图。限于篇幅，广度优先遍历过程就不再详细描述了。

MapReduce2.2:如图 8 所示，假设 A、B 两个点所在的子图标识为 E，C、D 两点所处子图标识为 F，将打好标识后的结果和原图同时作为输入,map 过程不做任何工作直接输出，reduce 过程中删除掉所有已经在最小生成树中的边，然后对剩下的边向每个邻居发送自己的状态，然后输出点本身。假设结果为 result2.2。

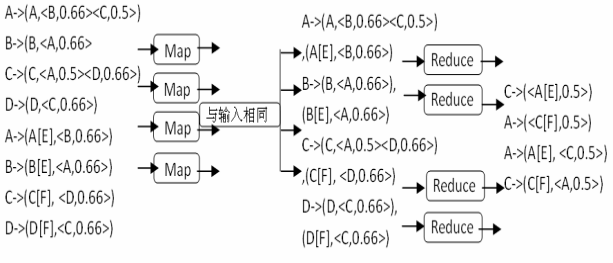

图 8 MapReduce2.2 过程描述图

MapReduce2.3:如图 9 所示，将 result2.2 作为输入，map 直接将输入输出，reduce 中以每个点的所在子图 id 作为键值，并且将自己的所有边用该边对面点所在的子图替代。如果有多个点在相同的子图中，则保留其中权重最大的一个。这个例子比较简单，所以这一步就得到了这次迭代的结果，实际上每一个子图可能会存在多条记录，还需要一次 mapreduce 过程将其合并成为一条记录。限于篇幅这里就不再详细描述了。

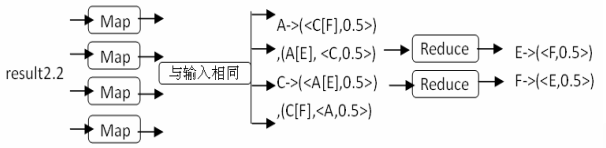

图 9　MapReduce2.3 过程描述图

经过上面一系列的运算得到的结果图如图 10，这里的 E 和 F 代表的是两个子图。这样子对新的图再进行一次步骤二操作，直到只剩一个点，或者是只剩多个独立的点。由于每个过程中我们都将最小生成树中的边输出，所以在这个步骤的结尾，我们将所有最小生成树的边收集起来就得到了整个图的最小生成树。

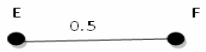

图 10　最小生成树过程一次迭代的结果图

步骤三：聚类划分

有了上面的最小生成树，聚类过程就简单的多了。首先统计最小生成树中的所有边并进行排序，根据排序我们可以选择最小的 n 条边或者选择一个阈值来进行聚类划分，从而确定了哪些最小生成树中的边式要被去掉的。

决定了需要去掉的边，我们通过一次 mapreduce 程序将这些边从去掉，去掉这些边自然就将整个最小生成树划分成为几个分离的子图，这样通过一次简单的广度优先过程，就可以将整个数据集划分成为多个集群了。

5　试验结果和后续工作展望

5.1 实验参数

该算法涉及到了两个实验参数：jaccard 系数的选择和最终集群划分参数的选择。这两个参数都和数据集的本身特征紧密结合，不同的数据集应当采用不同的参数。结合计算过程中的中间结果，我们可以比较简单的找到这两个参数参考。本算法中为了保证图的完整性，用所有点中能够保证图联通的最小值作为 jaccard 系数的阈值。而后一个参数，可以根据不同的聚类需求选择不同的参数。这也体现出了该算法的灵活性。

5.2 实验数据

实验数据是模拟真实用户数据的测试数聚集。数据集包含 100 条数据，每条数据有 100 个属性。为了使得该数据能够被划分成为 10 个不同的集群，在数据生成过程中，每 10 条数据会有 10 个属性生成概率比较大，其他的属性生成概率比较小。比如 1 到 10 条数据第 1-10 个属性存在概率为 90%，其他属性存在概率 10%，而 11-20 条数据第 11 到第 20 条属性存在概率为 90%，其他属性存在概率为 10%，依次类推。

这个数据集比较简单，但是有两个优点：1、从概率上保证了所有点被划分为 10 个集群，便于聚类结果正确性的验证；2、符合用户数据离散、多维、稀疏的特点。

5.3 实验环境

因为本次试验主要是为了比较试验结果，所以只采用了一台普通的 PC 机来运行 hadoop 计算平台。

5.4 试验结果描述

本次试验主要比较了该算法和 k-means 的聚类结果。图 11 描述了生成的最小生成树中边的权值分布。为了便于查看，图中横坐标是边的权重做倒数处理后的值，纵坐标是权重在这个范围内的边的个数。

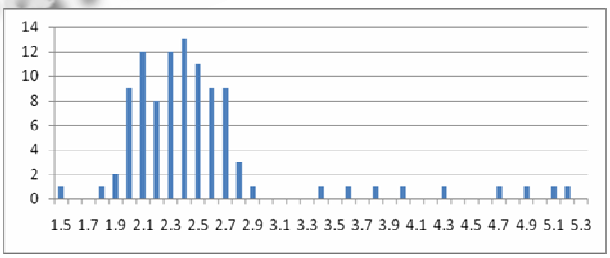

图 11　最小生成树边权重分布图

在上述分布图中，越往右说明两个点之间的距离越大。可以明显的看出，在右边有 9 条边的权重明显小于其他边，这九条边就是我们要去掉的边。从这个过程也可以看出，本算法解决了 k-means 和许多聚类算法一个很大的问题，就是聚类个数的确定问题。

通过聚类结果来看，本算法成功的将 10 个聚类分

开，没有错分一条数据。K-means 算法错分了大约 20%的数据。

为了更好的查看聚类结果，我们采用了另外一种度量来评价聚类结果。定义一个点和同一个聚类中其他所有点的平均 jaccard 系数为该点的聚类效用系数。一个点的聚类效用系数描述了该点与同一个聚类中的其他的点相似程度。所有点的平均聚类相似系数一定程度上反映了聚类的效果。

本文算法结果的平均聚类效用系数为 0.34657，而 k-means 聚类结果的平均聚类效用系数为 0.26854，可以看出本文算法的聚类效果明显好于 k-means。

本文算法运算时间为 20 分钟左右，自然远远长于单机上运行 k-means 的时间。但是该算法是为了解决海量数据的聚类问题，且充分考虑了数据可划分问题，适合于分布式处理，所以随着点的数量增加，算法在运行时间上会逐渐体现出优势。

5.5 后续工作展望

本文主要描述了算法过程和算法结果的比较。后续主要还会从三个方面展开工作。

1. 随着数据量增加和集群结点增加，算法的可扩展性分析；

2. 在生成图的过程中不同参数对聚类结果的影响，以及对参数设置方面的改进；

针对真实海量数据的测试和聚类结果测试。

参考文献

1 Han JW, Kamber M.数据挖掘概念与技术.北京:机械工业出版社,2001.

2 Zhang T, Ramakrishnan R, Livny M. BIRCH: An efficient data clustering method for very large databases. Jagadish HV, Mumick IS, eds. Proc. of the 1996 ACM SIGMOD Int'l Conf. on Management of Data. Montreal: ACM Press, 1996. 103−114.

3 Guha S, Rastogi R, Shim K. CURE: An efficient clustering algorithm for large databases. In: Haas LM, Tiwary A, eds. Proc. of the ACM SIGMOD Int'l Conf. on Management of Data. Seattle: ACM Press, 1998.73−84.

4 Guha S, Rastogi R, Shim K. ROCK: a robust clustering algorithm for categorical attributes. Proc. of the 15th Int'l Conf. on Data Eng., 1999.

5 Karypis G, Han EH, Kumar V. CHAMELEON: A hierar chical clustering algorithm using dynamic modeling. Technical Report, #99-007, Department of Computer Science and Engineering, University of Minnesota, 1999.

6 Ester M, Kriegel H, Sander J, Xu XW. A density-based algorithm for discovering clusters in large spatial databases with noise. Simoudis E, Han JW, Fayyad UM, eds. Proc. of the 2nd Int'l Conf. on Knowledge Discovery and Data Mining (KDD'96). Portland: AAAI Press, 1996. 226–231.

7 Ankerst M, Breunig MM, Kriegel HP, Sander J. OPTICS: Ordering points to identify the clustering structure. Delis A, Faloutsos C, Ghandeharizadeh S, eds. Proc. ACM SIGMOD Int'l Conf. on Management of Data. Philadelphia: ACM Press, 1999. 49–60.

8 Wang W, Yang J, Muntz RR. STING: A statistical information grid approach to spatial data mining. In: Jarke M, Carey MJ, Dittrich KR, Lochovsky FH, Loucopoulos P, Jeusfeld MA, eds. Proc. of the 23rd Int'l Conf. on Very Large Data Bases. Athens: Morgan Kaufmann, 1997. 186–195.

9 Sheikholeslami G, Chatterjee S, Zhang AD. WaveCluster: A multi-resolution clustering approach for very large spatial databases. In: Gupta A, Shmueli O, Widom J, eds. Proc. of the 24th Int'l Conf. on Very Large Data Bases. New York: Morgan Kaufmann, 1998. 428–439.

10 Rakesh A, Johanners G, Dimitrios G, Prabhakar R. Automatic subspace clustering of high dimensional data for data mining applications. In: Snodgrass RT, Winslett M, eds. Proc. of the 1994 ACM SIGMOD Int'l Conf. on Management of Data. Minneapolis: ACM Press, 1994. 94–105.

11 Karypis G. Kumar V. hMETIS 1.5: A hypergraph partitioning package. Technical report, Department of Computer Science, University of Minnesota, 1998.

12 Karypis G, Kumar V. METIS 4.0: Unstructured graph partitioning and sparse matrix ordering system. Technical report, Department of Computer Science, University of Minnesota, 1998.

13 杨戈,廖建新,朱晓民,樊秀梅.流媒体分发系统关键技术综述.电子学报,2009,(1):137−141.

14 Gallager RG, Humblet PA. Spira PM. A Distributed Algorithm for Minimum Weight Spanning Trees. ACM Trans. on Program. Lang. & Systems, 1983, 5: 66－77.

15 Dean J, Ghemawat S. MapReduce: Simplied Data Processing on Large Clusters. Proceedings of the 6th Symp. Operating System Design and Implementation (OSDI04). UsenixAssoc, 2004. 137–150.

16 Karypis G, Kumar V. Analysis of multilevel graph partitioning. Technical Report TR 95-037, Department of Computer Science, University of Minnesota, 1995.

17 Jain AK. Dubes RC. Algorithms for Clustering Data. Prentice Hall, 1988.