

## 基于 MapReduce 的 K-Medoids 并行算法

张雪萍, 龚康莉\*, 赵广才

(河南工业大学 信息科学与工程学院, 郑州 450001)

(\* 通信作者电子邮箱 likangong2006@126.com)

**摘要:**为了解决传统 K-Medoids 聚类算法在处理海量数据信息时所面临的内存容量和 CPU 处理速度的瓶颈问题,在深入研究 K-Medoids 算法的基础之上,提出了基于 MapReduce 编程模型的 K-Medoids 并行化算法思想。Map 函数部分的主要任务是计算每个数据对象到簇类中心点的距离并(重新)分配其所属的聚类簇;Reduce 函数部分的主要任务是根据 Map 部分得到的中间结果,计算出新簇类的中心点,然后作为中心点集给下一次 MapReduce 过程使用。实验结果表明:运行在 Hadoop 集群上的基于 MapReduce 的 K-Medoids 并行化算法具有较好的聚类结果和可扩展性,对于较大的数据集,该算法得到的加速比更接近于线性。

**关键词:** K-Medoids; 云计算; MapReduce; 并行计算; Hadoop

**中图分类号:** TP18 **文献标志码:** A

### Parallel K-Medoids algorithm based on MapReduce

ZHANG Xueping, GONG Kangli\*, ZHAO Guangcai

(College of Information Science and Engineering, Henan University of Technology, Zhengzhou Henan 450001, China)

**Abstract:** In order to solve the bottleneck problems of memory capacity and CPU processing speed when the traditional K-Medoids clustering algorithm is used to deal with massive data, based on the in-depth study of K-Medoids algorithm, a parallel K-Medoids algorithm based on the MapReduce programming model was proposed. The part of Map function is to calculate the distance of each data object to the center point of the cluster and (re) allocation of their respective clusters, and the part of Reduce function is to calculate the new center point of each cluster according to the intermediate results of the Map section. The experimental results show that the parallel K-Medoids algorithm in the Hadoop cluster based on the MapReduce running has good clustering results and scalability, and for large data sets, the algorithm may get close to linear speedup.

**Key words:** K-Medoids; cloud computing; MapReduce; parallel computing; Hadoop

## 0 引言

基于划分的聚类算法<sup>[1]</sup>以其简单、准确的特点,被广泛应用于科学研究和生产实践当中。其中, K-Means 和 K-Medoids 算法作为经典基于划分的聚类算法,无疑成为人们研究的起点和热点。针对 K-Means 算法思想的特点,当前已有人提出了基于 MapReduce 的 K-Means 并行算法<sup>[2]</sup>,并对其进行了优化。但 K-Means 算法本身的特点决定了它不适合很多生产实践当中的聚类分析,比如地理信息系统中带障碍约束的空间聚类分析<sup>[3]</sup>。带障碍约束的聚类分析在实际中运用越来越广泛,也是项目组前期研究的重点,而基于 K-

Medoids 的优化算法可以很好地解决障碍聚类的问题,因此基于 MapReduce 的 K-Medoids 并行算法研究非常必要。

## 1 MapReduce 编程模型

MapReduce 是一种简化的分布式编程模型和高效的任务调度模型<sup>[4]</sup>,用于大规模数据集的并行运算。MapReduce 主要通过 Map(映射)和 Reduce(化简)这两个步骤来并行处理大规模数据,它先把分割开来的相互独立的数据块文件通过 Map 过程进行高度的并行处理和计算,再通过 Reduce 过程将结果进行汇集整理并返回输出。MapReduce 的模型如图 1<sup>[5]</sup>所示。

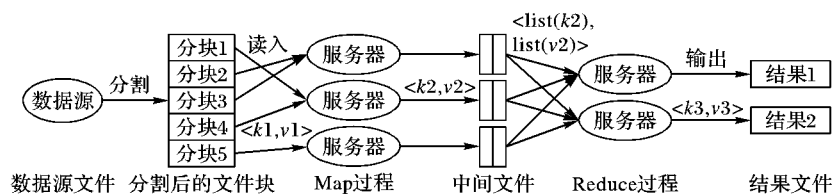


图1 MapReduce 模型

MapReduce 过程首先把源文件数据根据它所提供的库分成  $M$  份数据块文件;然后 Master 服务器分配不同的数据块文件给不同的 Worker 服务器进行 Map 过程,不同的 Map 之间

是相互独立、高度并行的,它以  $\langle \text{Key}, \text{Value} \rangle$  的形式读入数据,把处理后的中间结果也以  $\langle \text{Key}, \text{Value} \rangle$  的形式写入本地硬盘;然后 Master 服务器分配不同的 Worker 服务器进行 Reduce 过

收稿日期:2012-10-12;修回日期:2012-12-18。

基金项目:教育部新世纪优秀人才支持计划项目(NCET-08-0660);河南省高校科技创新人才支持计划项目(2008HASTYT012);海南省自然科学基金资助项目(610221);河南工业大学研究生创新计划基金资助项目(11YJCX69)。

作者简介:张雪萍(1968-),女,河南郑州人,教授,主要研究方向:智能信息处理、空间数据挖掘;龚康莉(1987-),女,江苏泰州人,硕士研究生,主要研究方向:智能信息处理;赵广才(1983-),男,河南漯河人,硕士研究生,主要研究方向:智能信息处理。

程,把〈Key, Value〉形式的中间结果进行合并输出。

MapReduce 的优点主要有两个方面<sup>[6]</sup>:其一,通过 MapReduce 这个分布式处理框架,不仅能用于处理大规模数据,而且能将很多繁琐的细节隐藏起来,比如,自动并行化、负载均衡和灾备管理等,这样将极大地简化程序员的开发工作;其二,MapReduce 的伸缩性非常好,也就是说,每增加一台服务器,其就能将差不多的计算能力接入到集群中,而过去的大多数分布式处理框架,在伸缩性方面都与 MapReduce 相差甚远。

## 2 K-Medoids 算法

针对 K-Means 算法在产生簇类的大小相差不会很大和它对“噪声”与孤立点数据较为敏感等缺点<sup>[7]</sup>,人们提出了 K-Medoids 算法,它选取一个叫作中心点的对象来替代 K-Means 算法中的重心点,从而很好地解决了 K-Means 算法的一些缺点。

### 2.1 K-Medoids 算法思想

基于划分的 K-Medoids 算法的思想是<sup>[8]</sup>:对于有  $n$  个数据对象组成的数据集,用户需要指定要划分成聚类的数目  $k$  值,首先任意选择  $k$  个不同的数据对象作为初始簇中心  $O_i$ ,然后根据其他每个对象到簇中心的距离,将它们(重新)分配给距离最近的簇;然后在每个簇内部顺序选择一个非簇中心点的数据对象  $O_{tmp}$ ,计算用  $O_{tmp}$  来代替  $O_i$  的消耗代价  $E_{tmp}$ ,比较  $E_{tmp}$  与簇中心为  $O_i$  的消耗代价  $E_i$  大小,如果  $E_{tmp}$  小于  $E_i$ ,则表明聚类在收敛过程中,把  $O_{tmp}$  赋给  $O_i$ ,更新当前的簇中心,继续迭代聚类,直到  $E_{tmp}$  大于等于  $E_i$ ,各簇中心不再发生变化,表明生成的簇结构不再调整,此时聚类准则函数收敛,算法结束。

判断聚类准则函数是否收敛,其目的就是为了使簇内数据对象之间的相似性尽可能大,而簇间数据对象的相似性尽可能小。可以在每次迭代过程中验证聚类的准则函数是否收敛来确定聚类是否应该结束,如果不收敛,就继续对数据对象进行聚类,否则,聚类完成,算法结束。

K-Medoids 算法用来衡量类内相似性的标准通常是使用欧几里得距离,其定义如下:

$$d_{(x,y)} = \sqrt{|x_1 - y_1|^2 + |x_2 - y_2|^2 + \dots + |x_p - y_p|^2} \quad (1)$$

其中,  $x = (x_1, x_2, \dots, x_p)$  和  $y = (y_1, y_2, \dots, y_p)$  是数据集中两个  $p$  维的数据对象。

评价 K-Medoids 等划分聚类效果的目标函数  $E$  可定义为:

$$E = \sum_{i=1}^k \sum_{x_j \in C_i} |x_j - o_i|^2 \quad (2)$$

其中:数据对象的总数为  $n$ ,划分成的聚类数目为  $k$ ,  $|x_j - o_i|^2$  为欧几里得距离,它表示某个簇类  $C_i$  中的一个数据对象  $x_j$  到该簇类中心  $o_i$  之间的距离,  $o_i$  表示簇类  $C_i$  的中心点。由于该公式中的欧氏距离指的是数据对象与它所在簇类的中心点之间的误差平方和,因此该目标函数又称为误差平方和准则函数。

### 2.2 K-Medoids 算法模型

K-Medoids 算法的数学描述<sup>[9]</sup>如下:假设由  $n$  个数据对象组成的数据集  $X = (x_1, x_2, \dots, x_n)$  是待聚类数据集,其中  $x_i = (x_{i1}, x_{i2}, \dots, x_{ip})$  表示  $p$  维的数据对象,算法是要找到  $X$  的一个划分  $P_k = \{C_1, C_2, \dots, C_k\}$ ,使其目标函数  $E$  的值最小,即式(2)取得最小值。

K-Medoids 算法模型<sup>[10]</sup>描述如下:

Algorithm: K-Medoids.

Input: 包含  $n$  个数据对象的数据集  $S$ ,要划分的聚类数目  $k$ 。

Output:  $k$  个聚类簇  $C_i$ ,满足聚类准则函数  $E$  最小。

Method:

1) 从数据集的  $n$  个数据对象中任意选择  $k$  个不同的对象作为初始簇中心  $O_i$ ;

2) Repeat;

3) 根据式(1)计算其他非簇中心对象到各个簇中心的距离,将剩余的对象分配到离它最近的簇类中;

4) 根据式(2)计算当前消耗代价  $E_i$ ;

5) 在每个簇内部顺序地选择一个非中心点  $O_{tmp}$ , 计算其消耗代价  $E_{tmp}$ ;

6) if  $E_{tmp} < E_i$ , then 把  $E_{tmp}$  赋值给  $E_i$ ,用  $O_{tmp}$  替代  $O_i$ ,来更新簇中心,形成  $k$  个新簇;

7) Until 值  $E$  不再发生变化。

K-Medoids 算法是聚类分析中的一个经典算法,从其算法模型中,可以看到该算法的核心部分是数据对象到簇中心的距离和簇中心的消耗代价的计算,而这部分又是最耗时的,当簇类结构一直在变的时候,就要一直迭代计算,直到簇类结构不再发生变化为止,其算法的时间复杂度为  $O(tk(n-k)^2)$ ,其中  $t$  是迭代次数,  $k$  是簇类的数目,  $n$  是数据集中数据对象的数目,通常情况下  $n$  值都远远大于  $t$  值和  $k$  值。当数据量  $n$  的规模达到海量级别时,算法耗时快速增长,算法本身的运行效率大幅下降,其运行时间难以满足实际需求的需求,因此需要思考如何提高算法运行速度。海量规模的数据在传统的串行运算的情况下,对计算机本身的计算处理速度以及内存存储状况是一种很大的考验,因此并行化计算和存储便自然地进入了人们的研究视线,把 K-Medoids 算法并行化,可以很好地解决其执行效率的问题,充分利用分布式资源,提高算法的运行速度。

## 3 基于 MapReduce 的 K-Medoids 并行算法

### 3.1 基于 MapReduce 的 K-Medoids 算法思想

K-Medoids 算法的并行化思想:对算法的每次迭代启动一次 MapReduce 计算过程,即在每次迭代内部实现并行计算,其中 Map 函数部分的主要任务是计算每个数据对象到簇类中心点的距离并(重新)分配其所属的聚类簇,这部分输入为待聚类的分块数据对象集和上一次迭代(或初始聚类)得到的簇类中心点,其输入为〈对象号,数据对象信息〉形式的〈key, value〉键值对,并且每个 Map 函数都读取簇类中心点集信息,然后对所输入的每个数据对象,计算其与各个中心点的距离,并把它分配给距离最近的簇类,然后输出形式为〈聚类簇 ID,数据对象信息〉的〈key, value〉键值对中间结果,并把中间结果交给相应的 Reduce 过程;Reduce 函数部分的主要任务是根据 Map 部分得到的中间结果,计算出新簇类的中心点,然后作为中心点集给下一次 MapReduce 过程使用,它的输入为〈聚类簇 ID, {数据对象集}〉的〈key, value〉键值对,Map 部分把所有聚类簇 ID 相同的数据对象传给同一个 Reduce 任务,同时 Reduce 任务累加 key 相同的点的个数,并不断增加数据对象的和,然后求出每个聚类簇中所有数据对象的平均值,然后找到距离均值最近的数据对象作为其新的簇类中心点,其输出形式为〈聚类簇 ID, 中心点〉的〈key, value〉键值对结果,并判断该聚类是否收敛,若收敛,则算法

结束;否则,新启动一次 MapReduce 过程,继续聚类。

### 3.2 基于 MapReduce 的 K-Medoids 算法流程

基于 MapReduce 的 K-Medoids 并行算法的处理过程主要分为两个部分,第一部分是初始化簇类中心点信息文件,并把数据集分成一定大小的  $m$  块,供并行处理;第二部分就是启动 Map 和 Reduce 任务进行算法的并行化计算,产生聚类结果。其算法流程如图 2<sup>[11]</sup> 所示。其中,每次迭代都要启动一次 MapReduce 计算过程,每次计算过程都有多个 Map 和

Reduce 任务,而且每个 Map 任务除了要读取一个数据块之外,还都要读取当前的聚类中心点信息文件。Map 任务负责各数据对象到簇类中心点之间距离的计算和分配数据对象到距离最近的簇类中;Reduce 任务负责把各个簇类中的对象汇集到一块,并计算寻找新的簇类中心点和判断聚类过程是否应该结束;这里增加了多个 Combine 任务,它的作用主要是分块计算各簇类的平均值,得到局部结果,再传送给 Reduce 任务,来减轻节点间的通信负担。

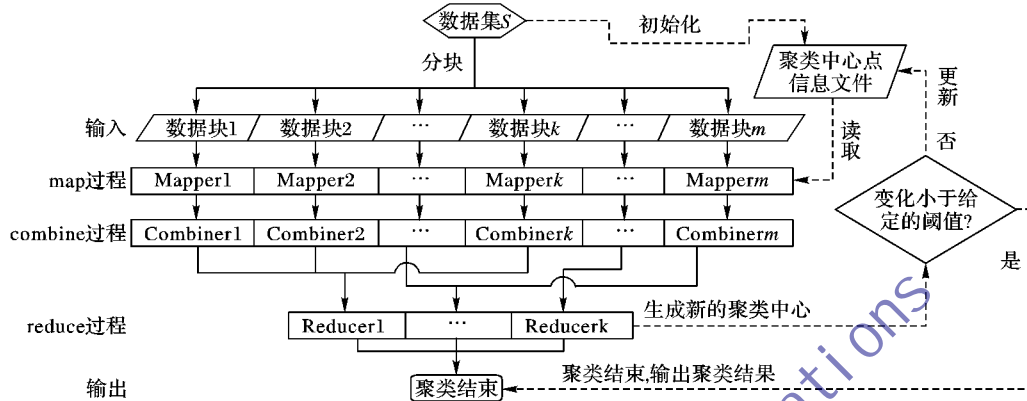


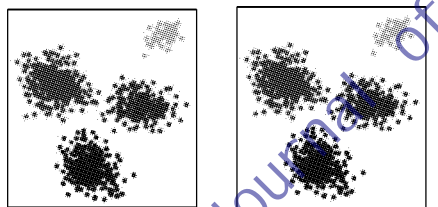
图2 基于 MapReduce 的 K-Medoids 算法并行流程

## 4 实验结果与性能分析

本实验采用多组测试数据在 Hadoop 分布式集群平台上进行仿真测试<sup>[12-13]</sup>,分别从算法的有效性、加速比和可扩展性对实验结果进行验证分析。

### 4.1 并行聚类算法的有效性分析

实验共有 12 000 个数据点,采用 K-Medoids 串行算法和并行算法进行实验,设置生成聚类 4 个,其实验的聚类结果如图 3 所示。



(a) K-Medoids串行算法 (b) K-Medoids并行算法  
图3 K-Medoids 串行和并行算法聚类结果

从图 3 可以看出 K-Medoids 并行算法具有和其串行算法较高一致性的聚类结果。计算结果表明其并行算法较串行算法的误差率都不超过 1.0%,它们都达到了较好的聚类效果和有效性。

### 4.2 并行聚类算法的加速比分析

并行计算的性能可以通过其加速比和可扩展性来衡量<sup>[14]</sup>。加速比是指通过并行计算使运行时间减少所获得的性能提升,它是验证并行计算性能的一个重要指标,其计算公式为  $S_p = T_s/T_p$ ,其中  $T_s$  表示串行算法(即在单个节点上)计算所消耗的时间, $T_p$  表示并行算法(即在  $p$  个相同节点上)计算所消耗的时间。加速比越大,表明并行计算消耗的相对时间越少,并行效率和性能提升越高。实验分别用 10 000 个数据点的数据集 DS1、50 000 个数据点的数据集 DS2 和 100 000 个数据点的数据集 DS3,经过多次运行测试,得到 K-Medoids 并行算法的平均加速比曲线如图 4 所示。从图 4 可以看出,随着集群中计算机节点的增加,其加速比的增加逐渐变慢,这

主要是因为节点的增加导致节点之间的通信量及其时间开销也逐渐变大;随着数据集中数据量的增大,其加速比的值也变大,而且曲线变缓、速度变慢,这表明对于较大的数据集,并行算法具有更高的效率,其加速比更接近于线性增大。由数据分析可知,基于 MapReduce 的 K-Medoids 并行算法具有较好的加速比。

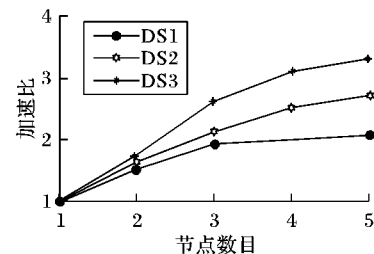


图4 基于 MapReduce 的 K-Medoids 并行算法的加速比曲线

### 4.3 并行聚类算法的可扩展性分析

当集群中的计算机节点的数目不断增加时,并行算法的加速比并不能无限地增大,它并不能反映集群的利用率情况,因此本文引入了并行算法效率的概念<sup>[15]</sup>。并行算法的效率表示并行算法执行过程中集群的利用率情况,其公式为  $E = S_p/P$ ,其中  $S_p$  表示算法的加速比, $P$  表示集群中的节点数。K-Medoids 并行算法的效率曲线如图 5 所示。

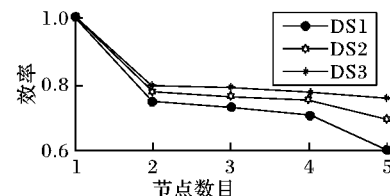


图5 基于 MapReduce 的 K-Medoids 并行算法的效率曲线

从图 5 中可以看到,它们的效率曲线呈整体下降趋势,这主要是由并行算法的通信调度等其他开销造成的。随着数据集中数据规模的增大,K-Medoids 并行算法的效率值越大,而且效率曲线越平稳。这两种并行算法在大数据集上具有较好的可扩展性。

(下转第 1035 页)



测试的实验结果表明,多量子态编码、单个体种群和基于平均收敛概率的收敛标准提高了算法的求解速度,引入定位概率函数的定位规则能获得更优的布局来提高算法的求解精度,故该算法在求解带平衡约束的圆形布局问题上是相当高效的。另外,本文所提出的 MQSQIEA 不仅可以在布局问题上用于优化定位序启发式策略的布局顺序,还可以用于解决有关优化顺序的其他问题(如 TSP)。

#### 参考文献:

- [1] GAREY M R, JOHNSON D S. Computers and intractability: a guide to the theory of NP-completeness[M]. San Francisco: W. H. Freeman and Company, 1979.
- [2] 李宁, 刘飞, 孙德宝. 基于带变异算子粒子群优化算法的约束布局优化研究[J]. 计算机学报, 2004, 27(7): 897-903.
- [3] 周驰, 高亮, 高海兵. 基于粒子群优化算法的约束布局优化[J]. 控制与决策, 2005, 20(1): 36-40.
- [4] HUANG W Q, CHEN M. Note on: An improved algorithm for the packing of unequal circles within a larger containing circle[J]. Computers & Industrial Engineering, 2006, 50(3): 338-344.
- [5] 王奕首, 史彦军, 滕弘飞. 用改进的散射搜索法求解带平衡约束的圆形 Packing 问题[J]. 计算机学报, 2009, 32(6): 1214-1221.
- [6] 刘景发, 李刚. 求解带平衡性能约束的圆形装填问题的吸引盘填充算法[J]. 中国科学: 信息科学, 2010, 40(3): 423-432.
- [7] 李刚, 刘景发. 基于禁忌搜索的启发式算法求解带平衡约束的圆形装填问题[J]. 中国科学: 信息科学, 2011, 41(9): 1076-1088.
- [8] HUANG W Q, LI Y, LI C M, et al. New heuristics for packing unequal circles into a circular container[J]. Computers and Operations Research, 2006, 33(8): 2125-2142.
- [9] XU Y C, XIAO R B, AMOS M. A novel genetic algorithm for the layout optimization problem[C]// CEC 2007: Proceedings of IEEE Congress on Evolutionary Computation. Singapore: IEEE, 2007:

3938-3943.

- [10] 徐义春, 肖人彬. 用蚁群算法求解带平衡约束的圆形布局问题[J]. 控制与决策, 2008, 23(1): 25-29.
- [11] 季美, 肖人彬. 基于蚁群算法的带平衡约束矩形布局问题的启发式求解[J]. 计算机应用, 2010, 30(11): 2898-2901.
- [12] 黄建江, 须文波, 孙俊, 等. 量子行为粒子群优化算法的布局问题研究[J]. 计算机应用, 2006, 26(12): 3015-3018.
- [13] HAN K H, KIM J H. Quantum-inspired evolutionary algorithm for a class of combinatorial optimization [J]. IEEE Transactions on Evolutionary Computation, 2002, 6(6): 580-593.
- [14] 申抒含, 金炜东. 多进制概率角复合位编码量子进化算法[J]. 模式识别与人工智能, 2005, 18(6): 657-663.
- [15] 申抒含, 金炜东, 陈维荣. 一种基于量子进化算法的概率进化算法[J]. 计算机工程与应用, 2005, 41(33): 64-67.
- [16] HAN K H, KIM J H. Quantum-inspired evolutionary algorithms with a new termination criterion,  $H_g$  gate, and two-phase scheme [J]. IEEE Transactions on Evolutionary Computation, 2004, 8(2): 156-169.
- [17] 杨丽, 李平, 秦亚玲. 改进的量子进化算法及其在 TSP 问题中的应用[J]. 信息与电子工程, 2006, 4(6): 412-416.
- [18] LAU T W, CHUNG C Y, WONG K P G, et al. Quantum-inspired evolutionary algorithm approach for unit commitment [J]. IEEE Transactions on Power Systems, 2009, 24(3): 1053-1062.
- [19] XING H, LIU X, JIN X, et al. A multi-granularity evolution based quantum genetic algorithm for QoS multicast routing problem in WDM networks[J]. Computer Communications, 2009, 32(2): 386-393.
- [20] HAN K H, KIM J H. On the analysis of the quantum-inspired evolutionary algorithm with a single individual[C]// CEC 2006: Proceedings of IEEE Congress on Evolutionary Computation. Piscataway, NJ: IEEE Press, 2006: 2622-2629.

(上接第 1025 页)

## 5 结语

本文对传统的  $K$ -Medoids 算法进行并行化研究,对提出的并行化算法进行多次的测试分析。测试结果表明该并行算法的误差率较小,具有很好的有效性,尤其随着数据集中数据规模的增大,其效率更高。本文的研究结果还可进一步在空间聚类分析以及在线实时聚类分析中进行扩展研究和实际应用。

#### 参考文献:

- [1] HAN J, KAMBER M. 数据挖掘: 概念与技术[M]. 范明, 孟小峰, 译. 2 版. 北京: 机械工业出版社, 2007.
- [2] 江小平, 李成华, 向文.  $k$ -Means 聚类算法的 MapReduce 并行化实现[J]. 华中科技大学学报: 自然科学版, 2011, 39(1): 120-124.
- [3] ZHANG X, DENG G, LIU Y. Spatial obstructed distance based on the combination of ant colony optimization and particle swarm optimization[C]// Proceedings of 2009 4th IEEE Conference on Industrial Electronics and Applications. Piscataway, NJ: IEEE Press, 2009: 106-111.
- [4] 刘鹏. 云计算[M]. 2 版. 北京: 电子工业出版社, 2011.
- [5] 成全, 邓倩妮. 云计算及其关键技术[J]. 计算机应用, 2009, 29(9): 2562-2567.
- [6] 吴朱华. 云计算背后的秘密——MapReduce[EB/OL]. [2012-10-10]. <http://cloud.51cto.com/art/201011/235046.htm>.

- [7] NG R T, HAN J. Efficient and effective clustering method for spatial datamining[C]// Proceedings of 1994 International Conference on Very Large Data Bases. Washington, DC: IEEE Computer Society, 1994: 144-155.
- [8] SHENG W G, LIU X H. A genetic  $k$ -medoids clustering algorithm [J]. Journal of Heuristics, 2006, 12(6): 447-466.
- [9] 张雪萍, 王家耀. 粒子群  $K$ -Medoids 带障碍约束空间聚类分析研究[J]. 小型微型计算机系统, 2009, 30(10): 2025-2029.
- [10] YANG T F, ZHANG X P. Spatial clustering algorithm with obstacles constraints by quantum particle swarm optimization and  $k$ -medoids[C]// Proceedings of 2010 International Conference on Computational Intelligence and Natural Computing. Washington, DC: IEEE Computer Society, 2010: 105-108.
- [11] 李应安. 基于 MapReduce 聚类算法的并行化研究[D]. 广州: 中山大学, 2010.
- [12] 王宏宇. Hadoop 平台在云计算中的应用[J]. 软件, 2011, 32(4): 36-38.
- [13] Apache. Hadoop[EB/OL]. [2012-10-10]. <http://hadoop.apache.org>.
- [14] WILKINSON B, ALLEN M. 并行程序设计[M]. 陆鑫达, 译. 北京: 机械工业出版社, 2002.
- [15] XU X W, JAGER J, KRIEGER H-P. A fast parallel clustering algorithm for large spatial databases[J]. Data Mining and Knowledge Discovery, 1999, 3(3): 263-290.