一种 MapReduce 架构下基于遗传算法的 K-Medoids 聚类

赖向阳 宫秀军 韩来明

(天津大学计算机科学与技术学院 天津 300072) (天津市认知计算与应用重点实验室 天津 300072)

摘 要 由互联网时代快速发展而产生的海量数据给传统聚类方法带来了巨大挑战,如何改进聚类算法从而获取有效信息成为当前的研究热点。K-Medoids是一种常见的基于划分的聚类算法,其优点是可以有效处理孤立、噪声点,但面临着初始中心敏感、容易陷入局部最优值、处理大数据时的CPU和内存瓶颈等问题。为解决上述问题,提出了一种MapReduce架构下基于遗传算法的K-Medoids聚类。利用遗传算法的种群进化特点改进K-Medoids算法的初始中心敏感的问题,在此基础上,利用MapReduce并行遗传K-Medoids算法提高算法效率。通过带标签的数据集进行实验的结果表明,运行在Hadoop集群上的基于MapReduce和遗传算法的K-Medoids算法能有效提高聚类的质量和效率。

关键词 海量数据,K-Medoids,MapReduce,遗传算法,聚类效率

中图法分类号 TP301.6 **文献标识码** A **DOI** 10.11896/j.issn.1002-137X.2017.03.006

Genetic Algorithm Based K-Medoids Clustering within MapReduce Framework

LAI Xiang-yang GONG Xiu-jun HAN Lai-ming

(College of Computer Science and Technology, Tianjin University, Tianjin 300072, China)

(Tianjin Key Laboratory of Cognitive Computing and Application, Tianjin 300072, China)

Abstract Huge volumes of data are increasing exponentially with the rapid development of Internet, which poses significant challenges to traditional clustering technologies. Thus, improving the accuracy and computing performance of clustering has become a research hotspot. As one of the partition-based clustering algorithms, K-Medoids can effectively deal with the problems with isolate and noise points. However, it also suffers from problems such as sensitive to initial centers, easily falling into local optimum, CPU and memory bottlenecks with big data sets. We proposed a genetic algorithm based K-Medoids clustering under MapReduce framework. The algorithm solves the center sensitivity problem of the K-Medoids by using the genetic algorithm. Also, it is built on the MapReduce framework to boost the efficiency both for K-Medoids and the genetic algorithm. The experiments demonstrate that the proposed algorithm can effectively improve the quality and efficiency of clustering.

Keywords Big-data, K-medoids, MapReduce, Genetic algorithms, Clustering efficiency

1 引言

随着信息时代的快速发展,直接或间接地产生了难以估量的海量数据,传统的数据挖掘方法在处理海量数据时面临很多挑战,如何从这些数据中挖掘有效的信息成为当前的研究热点。聚类是一种有效的分析数据的方法,主要的聚类方法有基于划分的K-Means和K-Medoids算法,以及基于层次的BIRCH和CURE算法等[1]。其中基于划分的聚类算法以其简单、准确的特点,被广泛应用于科学研究和生产实践中。K-Means是一种最基本的基于划分的聚类算法,因其理论可靠、算法简单、收敛速度快、能有效处理大数据集而被广泛使用,其缺点是对初始聚类中心敏感,容易受到极值等因素的影响[2]。K-Medoids是一种改进的K-Means算法,选取的聚类中心点是实际的数据点,对噪声与孤立点数据不敏感,但同样K-Medoids算法也容易受到初始中心选取的影响,容易陷入局部最优值。将遗传算法加入到K-Medoids中可以有效避免这些问题。遗传算法作为一种高效的全局优化搜索算法,被广泛应用于各种优化问题中,该算法以适应度函数为依据,通过对个体施加遗传操作,实现个体的一代代优化并逐渐逼近最优解[3]。虽然遗传算法能有效解决初始值敏感问题,但是由于遗传算法计算需求较高并且K-Medoids算法计算复杂,执行性能低,单机上进行传统的串行聚类分析就会面临内存容量、CPU处理速度等瓶颈问题,只能处理中小数据集,对大数据有一定局限性。因此并行化基于遗传算法的K-Medoids

到稿日期:2015-12-16 返修日期:2016-04-27 本文受国家自然科学基金项目(61170177),国家重点基础研究发展计划项目(2013CB32930X)资助。

赖向阳(1990—),男,硕士生,主要研究方向为数据挖掘、机器学习,E-mail:xiangyanglai@126.com;宫秀军(1972—),男,博士,副教授,主要研究方向为贝叶斯学习理论、分布式数据挖掘、生物信息网络构建等;韩来明(1990—),男,硕士生,主要研究方向为大数据、机器学习。

算法研究非常有必要。

目前已存在的并行 K-Medoids 方案有两种：1)以多核平台为基础利用 OpenMP 实现 K-Medoids 算法的并行化，充分发挥多核平台的 CPU 性能来提高算法性能[4]；2)基于MapReduce模型的 K-Medoids 并行化算法[5-9]，通过将消耗计算的工作分解为 Map 和 Reduce 功能，利用 Hadoop 集群的计算优势弥补了单机上的内存和 CPU 瓶颈等问题。此外，也存在多种并行遗传算法的研究或思路[10-12]，其中一种是基于 GPU 的并行方案，通过将大问题分解成较小的任务，利用良好的抽象数据并行体系结构进行并行，显示出了良好的加速比[13]，另一种是基于 Hadoop MapReduce 的并行方案，利用 Hadoop 框架解决遗传算法所面临的计算问题，可以让用户将遗传算法集中到特定的要解决的问题上，显示出了很好的性能[14-16]。

本文提出了基于 MapReduce 模型和遗传算法的 K-Medoids 算法，利用遗传算法解决 K-Medoids 算法的初始值敏感以及易陷入局部最优值的问题，用 MapReduce 解决 K-Medoids 算法的计算需求较大的问题，将基于遗传算法的 K-Medoids 算法进行并行化设计不仅可以提高算法执行的时间效率，在一定程度上还可以防止早熟现象的发生，从而提高聚类的效果。

本文第 2 节主要介绍基于遗传算法的 K-Medoids 算法的思路以及方案；第 3 节展示基于 MapReduce 模型和遗传算法的 K-Medoids 算法的并行方案；第 4 节主要介绍实验结果及其分析；最后总结与展望，描述目前的工作结论以及下一步需要改进的地方。

2 基于遗传算法的 K-Medoids

遗传算法是一种启发于生物界的智能搜索算法，它不依赖于问题的本身，所需要的仅是通过事先设定的适应度函数，对算法产生的染色体进行评价，使适应度高的染色体得到更好的繁殖机会。算法的终止条件有两种：达到事先设定的最大遗传代数或者是达到最佳适应度值。将遗传算法加入到 K-Medoids 算法中，利用遗传算法的优势可以较好地克服 K-Medoids 算法的一些不足。现有已存在的基于遗传算法的 K-Medoids 算法有一种新的空间聚类方法[17]，在良好的约束条件下，可以兼顾较好的收敛速度和较强的全局搜索能力，缺点是聚类速度较慢。还有一种混合遗传算法和 K-Medoids 算法的聚类算法[18]，通过可变长度的个体编码以及启发式操作，提高了算法的有效性。为提高基于遗传算法的 K-Medoids 算法的性能，本文提出了通过特定的编码方式提高聚类的表达能力，合理地利用适应度函数提高聚类效果。

2.1 算法的主要步骤

(1)初始化运行参数(种群个数 p、聚类个数 k、最大遗传代数)。

(2)初始化种群即随机选取 p 条染色体，每条染色体表示一个聚类的结果。

(3)对每条染色体进行 K-Medoids 操作，然后评价其适应度并记录。

(4)判断是否达到结束条件(满足最大代数或问题最优解)，未满足则进行以下操作：

1)选择操作

2)交叉操作

3)变异操作

4)产生新种群

(5)转到步骤(3)。

2.2 编码方式

在大多数的遗传算法中，编码方式多使用二进制编码和浮点数编码等，但多数情况会受到上下文不敏感的问题困扰，因此，在遗传操作中要设计对应的解决对策。本文提出一种简单自然的编码方式，从{1,2,…,n}中选取一个值表示染色体，每条染色体的等位基因代表所选对象的中心，染色体的长度代表聚类中心的个数。例如，个体{1,2,5,8,10}表示第 1 个、第 2 个、第 5 个、第 8 个和第 10 个对象被选为中心，聚类个数为 5 个，个体代表一个聚类结果；该表示可以在多数传统的遗传操作中实现上下文敏感。

2.3 适应度函数

一个个体的适应度函数代表了一条染色体的适应程度，因此适应度函数的设计关系到遗传算法的优劣。由于 K-Medoids 是聚类算法，聚类结果的评判标准是“内部紧密，外部松散”，因此适应度函数设计为：

$$f=\frac{\max(c_i,c_j)}{\sum_{i=1}^{k} dist(x,c_i)} \tag{1}$$

其中，分子表示一条染色体中各聚类中心之间的最大距离，分母表示各个聚类中数据点到中心点的距离总和。这样的设计保证个体适应度越大，聚类效果越好。

2.4 遗传算子

(1)选择算子。采用轮盘赌选择方法，即各个个体被选中的概率与其适应度函数值的大小成正比。设群体大小为 N，个体 x_i 的适应度为 $f(x_i)$，则个体 x_i 的选择概率为：

$$P(x_i)=\frac{f(x_i)}{\sum_{j=1}^{N} f(x_j)} \tag{2}$$

其中，分子表示个体 x_i 的适应度，分母表示所有染色体的适应度之和。这样保证了适应度大的个体有较大的概率保留到下一代，而适应度值小的个体容易被淘汰，但这不是绝对的。

(2)交叉算子。依据交叉概率 Pc 对两个相互配对的染色体按某种方式相互交换其部分基因，从而形成两个新的个体。交叉过程中，必须保证后代至少有 2 个但是不多于 K 个，并且要避免后代产生的新个体和父代重复。

(3)变异算子。通过指定的变异概率 Pm，按某种方式进行基因变异，形成新个体。种群中的个体变异可以维护种群的多样性，也可在一定程度上防止发生种群早熟现象。

3 基于 MapReduce 和遗传算法的 K-Medoids 算法(MRGAK-Medoids)

MRGAK-Medoids 算法的基本思想是：初始化运行参数、种群，将 P 条染色体写入初始聚类中心文件 clusters，数据集

和 clusters 存放在 Hadoop 的文件系统 HDFS 上，在 Map 处理阶段读取上一轮的聚类中心文件（初始为 cluster），然后读取分配给该 Mapper 的数据点，分别计算出该数据点与每条染色体最近的聚类中心点；Combine 阶段处理每个聚类中心点的局部结果；最后 Reduce 汇总结果，计算新中心 clusters，判断是否达到条件，不满足则进行遗传算子操作，生成新种群，更新 clusters 文件，继续完成迭代。

3.1 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，用于大规模数据集的并行运算，最早由 Google 提出并应用[19]，现在为 Apache 下 Hadoop 的主要编程模型。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 所示。

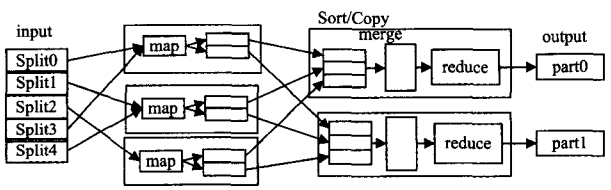

图 1　MapReduce 模型

MapReduce 编程模型的核心在于 Map 和 Reduce 的功能实现，Map 的功能是在一定的规则下将输入的〈key1，value1〉对转化为另一〈key2，value2〉对。Reducer 接收从 Mapper 传递而来的 key/value 对，根据 key 来排序、分组，最终生成〈K2，list〈V2〉〉，然后 Reducer 根据〈K2，list〈V2〉〉生成〈K3，V3〉并输出。

MapReduce 主要有两方面优点：1）MapReduce 框架不仅能有效处理海量数据，还能将许多繁琐的细节隐藏起来，从而简化程序员的开发工作；2）MapReduce 的可扩展性非常好，每增加一台计算节点，就可以将节点的计算能力接入到集群中，这一方面比过去的大多数分布式处理框架更好。

3.2 Map 阶段

Map 的主要功能是将数据点划分到每条染色体最近的 cluster 中，算法复杂度为 $O(k*n_i)$，k 是 cluster 的个数，n_i 是当前 split 中数据实例的数量。

算法 1　Map(key，value)

```
输入：包含多个染色体的聚类中心文件 clusters，包含若干数据点的数据分块
输出：〈K，V〉对，K 是 cluster，V 是 datapoint
foreach(value in dataset){
    Instance instance=ConstructInstance(value);
    minDistance=Double. MAX_VALUE;
    minIndex=-1;
    for i=0 to clusters. length do{ tempDistance=ComputeDis(instance,clusters[i]. getCenter[i]);
      if(tempDistance<minDistance){
        minDistance=tempDistance;
        minIndex=i;
      }
    }
Output〈cluster[minIndex]，instance〉;
}
```

3.3 Combine 阶段

完成 Map 阶段后，计算节点将计算 K-Medoids 算法以及遗传算法的操作，计算新中心的步骤是算法中最耗时的操作，时间复杂度约为 $O(n_i*n_i/k)$，k 是 cluster 的个数，n_i 是当前 split 中数据实例的数量。因此增加 Combine 操作计算中间结果，减轻 Reduce 端的压力。

算法 2　Combiner(key，Iterator〈Writable〉 values)

```
输入：cluster，所有此 cluster 的数据实例
输出：〈K，V〉对，K 是 cluster，V 是该 cluster 的局部结果
foreach(c in clusters){
  foreach(v in values){
      minCost=Double. MAX_VALUE;
    minIndex=-1;
    for i=0 to values. length do{
    tempCost+=ComputeDis(v,values[i]);
    }
    if(tempCost<minCost){
      minCost=tempCost;
      minIndex=i;
    }
  tmpValue=values[minIndex]+values. length;
    }
  Output〈c，tmpValue〉;
}
```

3.4 Reduce 阶段

Reduce 的任务之一是计算新中心，该算法的时间复杂度为 $O(k*s)$，其中 k 是 cluster 的个数，s 是 data 分成 split 的数量。Reduce 的第二个任务就是遗传算法的操作，该部分的时间复杂度为 $O(p*t)$，p 表示种群中染色体的个数，t 表示每条染色体遗传操作的时间。

算法 3　Reduce(key，Iterator〈Writable〉tmpValues)

```
输入：所有的 cluster，每个 cluster 对应的 tmpValue
输出：〈K，V〉对，K 是当前 key，V 是新 cluster
对种群中每个个体
foreach(c in clusters){
    Number[] ns;
    Values[] vs;
    For i=0 to tmpValues. length do{
      ss[i]=tmpValues. getValue();
      vs[i]=tmpValues. getNumber();
    }
    newCenter nc=getNewCenter(ns,vs);
    fitness=Fitness(nc);
}
bestFitness=max(fitness);//选择最大适应度值的个体
bestValue=getBestValue();
saveBestValueToHDFS();
```

```
if(满足最优解||达到最大遗传代数){
    Output(key,bestValue);
}else{
  //对每个子种群进行遗传操作
  按轮盘赌选择方法选择复制下一代个体;
  按交叉概率 Pc 进行交叉,产生新个体;
  按给定的变异概率,随机选择变异位置进行变异操作,产生新个体
  生成新种群 P'
Output(key,P');
}
```

4 实验结果与分析

本节主要介绍实验环境以及实验结果及其分析,实验采用多组测试数据在 Hadoop 分布式集群平台上进行测试,分别从算法的准确率、加速比对实验结果进行验证分析。

4.1 实验环境

实验采用 Hadoop 集群,以主从方式进行连接,其中 1 个节点作为主节点,4 个节点作为计算节点,其中主节点的 CPU 为 InterCoreQ6600,内存为 4GB;工作节点为 4 台工作站,CPU 是 IntelXeonE5-2620,有 24 个逻辑 CPU,2 个物理 CPU,每个 CPU 含 6 核,内存是 32G,连接主节点和计算节点的网卡是 1000Mbps,Hadoop 的版本为 1.2.1,java 的版本为 1.7.0_79。

实验数据采用的是多组来自 UCI 公共数据库的数据集,详细信息如表 1 所列。

表 1 测试数据集信息

数据集	实例数量	属性数量	标签数量
Iris	150	4	3
Susy	5000000	28	2
Higgs	11000000	18	2

4.2 实验

实验选用 3 组带标签的数据来验证 MRGAK-Medoids 算法的准确率和算法加速比。

4.2.1 聚类准确率

由于本文主要采用带标签的数据集,因此准确率 P 定义如下:

$$P=\frac{C}{N} \tag{3}$$

其中,C 为数据点的标签和对应的聚类中心点的标签相同的数量,N 为所有的点数量。

采用运行 10 次的平均值作为最终的实验结果。相关的参数设置:K-Medoids 算法选取的 K 的值等于当前数据集的标签个数,遗传算法的交叉概率为 0.9,变异概率为 0.05,种群大小设为 40,算法的最大迭代次数设为 100。

可以看出,K-Medoids 算法初始聚类中心的选取的敏感性较大,容易陷入局部最小值,并非每次都能得到最优解,特别是对于 Susy 和 Higgs 这种较高维度的数据集,所需平均迭代次数较高。而改进的算法对每组数据集的实验平均迭代次数较少,避免了局部最优解,聚类效果较好。K-Medoids 算法在处理小数据集时准确率较高,随着数据集增大其准确率普遍偏低,而经过改进的 K-Medoids 算法的准确率有明显提高,这主要是因为并行遗传算法能够有效防止早熟现象的发生从而收敛到较好的结果。不同数据集测试 K-Medoids 算法的结果如表 2 所列。

表 2 不同数据集测试 K-Medoids 算法的结果

	K-Medoids		MRGAK	
数据集	平均迭代次数	准确率/%	平均迭代次数	准确率/%
Iris	45	90	31	93
Susy	65	45	34	68
Higgs	89	51	28	71

4.2.2 加速比

为了验证不同数据集在不同节点上运行的时间效果,调整集群节点数目如表 3 所列。

表 3 不同数据集在不同节点数量下的运行时间/s

节点数	Iris	Susy	Higgs
1	2476	19587	42720
2	2752	12411	21361
3	2748	10309	15822
4	2251	9326	13350

加速比是指在单处理器上和分布式处理器上运行同一任务的时间比率,其是用来衡量并行系统的性能和效率。加速比定义如下:

$$S_p=\frac{T_1}{T_p} \tag{4}$$

其中,p 为 Hadoop 的计算节点个数,T_1 为使用 Hadoop 的 1 个节点的运行时间,T_p 为在 p 个节点构成的 Hadoop 集群下的运行时间。当 $S_p=p$ 时称其为线性加速比。

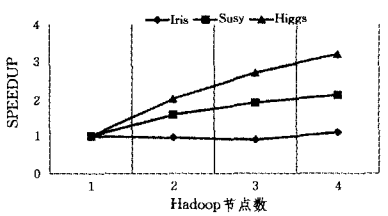

图 2 节点数和加速比的关系

通过上述结果可以看出,随着 Hadoop 节点数量的增加,MRGAK-Medoids 算法的性能逐渐增强,增长的速率受到新增加的节点之间的通讯开销影响。当数据集较小时,由于计算量较小,并行 K-Medoids 算法并不占优势,加速比容易陷入瓶颈;当数据集较大时,所需要的计算量剧增,因此单节点的 K-Medoids 算法将长期处于计算消耗中,导致效率变慢。而并行遗传 K-Medoids 算法一方面利用 MapReduce 提高计算效率,另一方面通过遗传算法多方位地搜索最优解,并且通过变异等操作可以有效避免 K-Medoids 结果陷入局部最优值,因此可以在有限的迭代中得到最接近的结果,加速比可以接近线性比。

结束语 为了提高大数据聚类的准确率和效率,本文提出了一种 MapReduce 架构下基于遗传算法的 K-Medoids 聚类算法,利用 MapReduce 框架的集群优势提高算法效率,同时通过遗传算法的进化计算避免聚类陷入局部最优解,提高

(下转第 58 页)

and efficient provable data possession[C]// 4th International Conference on Security and Privacy in Communication Networks. 2008.

[4] ERWAY C, KUPCU A, PAPAMANTHOU C, et al. Dynamic provable data possession[C]//16th ACM CCS. 2009:213-222.

[5] SHACHAM H, WATERS B. Compact Proofs of Retrievability [C]// Proceedings of the 14th International Conference on the Theory and Application of Cryptology and Information Security (ASIACRYPT'08). Melbourne, Australia, Berlin: Springer-Verlag, 90-107.

[6] TAN S, JIA Y, HAN W H. Research and Development of Provable Data Integrity in Cloud Storage [J]. Chinese Journal of Computers, 2015, 38(1): 164-177. (in Chinese)
谭霜,贾焰,韩伟红. 云存储中的数据完整性证明研究及进展[J]. 计算机学报,2015,38(1):164-177.

[7] BOWERS K D, JUELS A, OPREA A. Proofs of Retrievability: Theory and Implementation[C]// Proceeding(s) of ACM Workshop on Cloud Computing Security. Chicago, USA, 2009: 43-53.

[8] WANG C, WANG Q, REN K, et al. Privacy-preserving public auditing for data storage security in cloud computing[C]// 29th IEEE INFOCOM. 2010.

[9] CHEN L, ZHOU S, HUANG X, et al. Data dynamics for remote data possession checking in cloud storage[J]. Computers and Electrical Engineering, 2013, 39: 2413-2424.

(上接第 26 页)

聚类的准确率。实验结果表明,MRGAK-Medoids 算法在处理大数据时具有接近线性的加速比,同时聚类质量得到提升。

接下来的工作是如何改进遗传算子使得遗传操作可以快速收敛、如何降低 Reduce 的任务负载等问题。

参 考 文 献

[1] HAN J, KAMBER M. 数据挖掘:概念与技术(第 2 版)[M]. 范明,孟小峰,译. 北京:机械工业出版社,2007.

[2] LAI Y X, LIU J P, YANG G X. K-Means Clustering Analysis Based on Genetic Algorithm[J]. Computer Engineering, 2008, 34(20): 200-202. (in Chinese)
赖玉霞,刘建平,杨国兴. 基于遗传算法的 K 均值聚类分析[J]. 计算机工程,2008,34(20):200-202.

[3] TANG Z X. K-means Clustering Algorithm Based on Improved Genetic Algorithm[J]. Journal of Chengdu University (Nature Science Edition), 2011, 30(2): 162-164. (in Chinese)
唐朝霞. 一种改进的基于遗传算法的 K 均值聚类算法[J]. 成都大学学报(自然科学版),2011,30(2):162-164.

[4] LI J B, YANG L, HUA B. Research on parallel K-Medoids algorithm on multi-core platform[J]. Application Research of Computers, 2011, 28(2): 498-500. (in Chinese)
李静滨,杨柳,华蓓. 基于多核平台并行 K-Medoids 算法研究[J]. 计算机应用研究,2011,28(2):498-500.

[5] LI J B, YANG L, CHEN N J. An improved parallel K-Medoids algorithm based on MapReduce[J]. Journal of Guangxi University (Nature Science Edition), 2014(2): 341-345. (in Chinese)
李静滨,杨柳,陈宁江. 基于 MapReduce 的改进 K-Medoids 并行算法[J]. 广西大学学报(自然科学版),2014(2):341-345.

[6] ZHANG X P, GONG K L, ZHAO G C. Parallel K-Medoids algorithm based on MapReduce[J]. Journal of Computer Application, 2013, 33(4): 1023-1025. (in Chinese)
张雪萍,龚康莉,赵广才. 基于 MapReduce 的 K-Medoids 并行算法[J]. 计算机应用,2013,33(4):1023-1025.

[7] JIANG Y B, ZHANG J M. Parallel K-Medoids Clustering Algorithm Based on Hadoop[C]// IEEE Beijing Section. Proceedings of 2014 IEEE 5th International Conference on Software Engineering and Service Science. IEEE Beijing Section, 2014: 4.

[8] ZHU Y, WANG F, SHAN X, et al. K-medoids clustering based on MapReduce and optimal search of medoids[C]// 2014 9th International Conference on Computer Science & Education (ICCSE). IEEE, 2014: 573-577.

[9] SRINIVASULU D L, REDDY A V, AKULA V S G, et al. Improving the Scalability and Efficiency of K-Medoids By Map Reduce [J]. International Journal of Engineering and Applied Sciences (IJEAS), 2015(4): 88-90.

[10] ALBA E, TROYA J M. A survey of parallel distributed genetic algorithms [J]. Complexity, 1999, 4(4): 31-52.

[11] GUO T C, MU C D. The Parallel Drifts of Genetic Algorithms [J]. System Engineering-Theory & Practice, 2002, 22(2): 15-23, 41. (in Chinese)
郭彤城,慕春棣,并行遗传算法的新进展[J]. 系统工程理论与实践,2002,22(2):15-23,41.

[12] WANG X L, LI Q. Application and Research on Parallel Genetic Algorithm[J]. Microcomputer Information, 2007, 23(9): 205-206. (in Chinese)
王小良,李强. 并行遗传算法研究及其应用[J]. 微计算机信息,2007,23(9):205-206.

[13] JOHAR F M, AZMIN F A, SUAIDI M K, et al. A review of genetic algorithms and parallel genetic algorithms on Graphics Processing Unit (GPU)[C]// 2013 IEEE International Conference on Control System, Computing and Engineering (ICCSCE). IEEE, 2013: 264-269.

[14] FERRUCCI F, SALZA P, KECHADI M T, et al. A Parallel Genetic Algorithms Framework based on Hadoop MapReduce[C]// The ACM Symposium. ACM, 2015: 1664-1667.

[15] Jin C, Vecchiola C, Buyya R. Mrpga: an extension of mapreduce for parallelizing genetic algorithms[C]// IEEE Fourth International Conference on EScience, 2008 EScience'08. IEEE, 2008: 214-221.

[16] FERRUCCI F, KECHADI M, SALZA P, et al. A Framework for Genetic Algorithms Based on Hadoop[J]. arXiv preprint arXiv: 1312.0086, 2013.

[17] ZHANG X, WANG J, WU F, et al. Genetic K-Medoids Spatial Clustering with Obstacles Constraints[C]// 2006 3rd International IEEE Conference on Intelligent Systems. IEEE, 2006: 826-831.

[18] SHENG W, LIU X. A genetic k-medoids clustering algorithm [J]. Journal of Heuristics, 2006, 12(6): 447-466.

[19] DEAN J, GHEMAWAT S. MapReduce: Simplified Data Processing on Large Clusters[C]// Conference on Symposium on Operting Systems Design & Implementation. 2004: 107-113.