·体系结构与软件技术· 文章编号:1000-3428(2016)05-0054-06 文献标识码:A 中图分类号:TP18

基于 Hadoop 的微阵列数据两阶段并行 K 近邻基因提取

齐向明[1],郑 帅[1],魏 萍[2]

(1. 辽宁工程技术大学软件学院,辽宁 葫芦岛 125105;
2. 中国石油大学地球物理与信息工程学院,北京 102249)

摘 要: 基因信息选取工作中由于数据量庞大,传统单线程运行的分类查询方法无法满足实时性与提取精度要求。为此,利用 Hadoop 框架设计两阶段并行计算模型。其中第 1 阶段用于候选基因子集并行选取,第 2 阶段用于并行 K 近邻基因信息选取,从而实现并行计算的全过程覆盖。为降低算法的计算复杂度,针对基因信息微阵列数据,定义数据筛选指标对其进行采样,在降低数据处理量的同时消除数据冗余。实验结果表明,该算法具有较高的运行效率,并且继承了 Hadoop 编程模型的可扩展特性,可移植性较强。

关键词: Hadoop 框架;并行计算;微阵列采样;大数据;K 近邻;基因信息

中文引用格式: 齐向明,郑 帅,魏 萍. 基于 Hadoop 的微阵列数据两阶段并行 K 近邻基因提取[J]. 计算机工程,2016,42(5):54-59.

英文引用格式: Qi Xiangming, Zheng Shuai, Wei Ping. Micro-array Data Two-stage Parallel K Nearest Neighbor Gene Extraction Based on Hadoop[J]. Computer Engineering, 2016, 42(5):54-59.

Micro-array Data Two-stage Parallel K Nearest Neighbor Gene Extraction Based on Hadoop

QI Xiangming[1], ZHENG Shuai[1], WEI Ping[2]

(1. College of Software, Liaoning Technical University, Huludao, Liaoning 125105, China;
2. College of Geophysics and Information Engineering, China University of Petroleum, Beijing 102249, China)

【Abstract】 Because of huge amount of data in gene information extraction, whose real-time requirements can not be met by traditional methods with single threaded operation, the Hadoop framework is used to design the two-stage parallel computing model. The first stage is used to extract candidate gene subset, and the second stage is used to extract parallel K nearest neighbor genetic information, and it implements whole process cover of parallel computing. At the same time, in order to further reduce the computational complexity of the algorithm, the microarray data sampling method is used to reduce the amount of data processing and eliminate data redundancy. Experimental results show that the proposed algorithm has better running efficiency, inherits the extensible features of Hadoop programming model, and has strong portability.

【Key words】 Hadoop framework; parallel computing; micro-array sampling; big data; K nearest neighbor; gene information

DOI: 10.3969/j.issn.1000-3428.2016.05.010

1 概述

随着基因微阵列矩阵(又称基因芯片)技术的发展,在微型芯片上可实现对众多基因的信息表达,这项技术为计算机处理基因信息提供了帮助[1-2]。

微阵列矩阵技术逐渐被应用到疾病分类和基因调控领域,在生物信息学及临床诊断中发挥着重要作用。在国内文献中,如文献[3]指出微阵列基因表达技术已广泛应用于各种癌症的诊断研究中;文献[4]提出一种时序微阵列同异步的基因调控技术。在国外研究中,文献[5]提出基于微阵列技术和进化计算的基因调控聚类方法;文献[6]提出基于交叉验证估计的癌症微阵列数据诊断等。此类文献较多,不再一一赘述。

基金项目: 辽宁省教育厅基金资助项目(L2012113)。

作者简介: 齐向明(1966-),男,副教授、硕士,主研方向为数据挖掘、大数据技术、图形图像处理;郑 帅,硕士;魏 萍,讲师、博士。

收稿日期: 2015-04-10 **修回日期:** 2015-05-19 **E-mail:** lkdqxming@163.com

在微阵列矩阵处理过程中,存在的最大问题是处理数据量过于庞大,这些数据通常含有成千上万的基因(有时会超过 10 000 个基因,甚至更多),每个基因又含有大量 DNA 数据,尽管这些基因都可在基因芯片中进行表示,但是由于噪声的存在,相互之间并无可借鉴性。因此,如何更为有效地对基因数据进行处理,是该领域研究的难点[7]。

目前,国内在该领域的研究,多集中在单线程聚类或者排序查询方法,如文献[8]微阵列数据挖掘算法中,对于频繁闭合项集需事先给定最小支持度的问题,利用 top-k 算法进行改进来替换最小支持度;文献[9]基于图论理论,设计有向无权图 DNA 微阵列数据分析结构,并获得最小割和第二小割权值。此外,还有基于采样技术[10]、机器学习,遗传算法[11]、支持向量机(SVM)[12]等方法。

上述算法从不同角度对基因选取进行研究,但存在一个共同点,单线程计算方法,计算效率不高。针对该问题,本文设计多线程并行微阵列矩阵处理算法,从并行计算框架上选择目前大数据计算常用到的 Hadoop 框架下 MapReduce 并行计算模型。定义筛选指标 $BWG(g_i)$ 进行微阵列矩阵数据采样,进一步降低数据处理量。在分类算法选取方面,文献[13]指出 top-k 算法比 SVM、神经网络、粗糙集等分类算法的计算复杂度更低,因此,本文选取 top-k 算法作为并行分类算法。

2　并行基因信息提取步骤

2.1　Hadoop 框架

首先对 Hadoop 框架并行基因选择算法模型进行描述,该框架核心部分为 MapReduce 模型,图 1 给出标准 Hadoop 框架下 MapReduce 并行计算的编程模型。

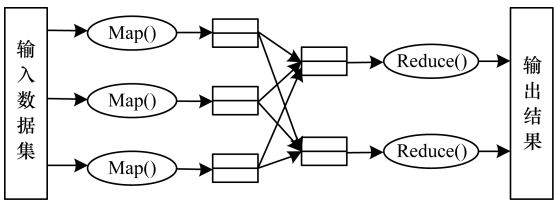

图 1　Hadoop 编程模型

在标准 Hadoop 模型中的 Map 过程和 Reduce 过程,是根据定义的〈key,$value$〉键值对进行操作的,上述 2 个主要操作过程具体定义如下[14]:

Map 过程:

$$(k_1,v_1)\rightarrow[(k_2,v_2)] \tag{1}$$

Reduce 过程:

$$(k_2,[v_2])\rightarrow[(k_3,v_3)] \tag{2}$$

在通常情况下,微阵列数据可表示为 $N\times M$ 的数据矩阵,其中,N 为基因数量;M 为参与实验的样本数量。由于在数量上 $N\gg M$,通常对微阵列数据矩阵采取转置的方式进行处理。图 2 给出 HMP-KNN 并行化计算过程。

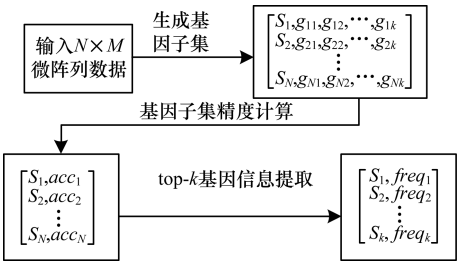

图 2　HMP-KNN 基因提取流程

从图 2 可看出,HMP-KNN 基因提取流程主要包含以下 3 个过程:潜在基因子集的生成,基因子集精度计算,top-k 基因信息提取。在通常情况下,微阵列数据中含有许多独立的基因信息,这些信息不影响分析结果,且与其他基因没有相关性,因此,在基因分析中,这部分信息可不予考虑。

2.2　微阵列数据 Hadoop 并行化

首先定义数据筛选指标 BWG,则潜在基因子集产生过程如下:

步骤 1(潜在基因)　令 $G=\{g_1,g_2,\cdots,g_N\}$ 为基因数据集,$S=\{s_1,s_2,\cdots,s_M\}$ 为基因数据微阵列采样数据集。若满足 $g_i\in G$ 且 $BWG(g_i)\geqslant BWG_{th}$,则称 g_i 为潜在基因,其中,BWG_{th} 为阈值。筛选指标 $BWG(g_i)$ 定义如下:

$$\begin{aligned}BWG(g_i)&=\frac{SS_{B_j}}{SS_{W_j}}\\&=\frac{\sum_{i=1}^{m}\sum_{c=1}^{k}I(y_i=c)(\bar{x}_{c,j}-\bar{x}_j)^2}{\sum_{i=1}^{m}\sum_{c=1}^{k}I(y_i=c)(\bar{x}_{i,j}-\bar{x}_{c,j})^2}\end{aligned} \tag{3}$$

其中,m 为训练样本数;k 为分类数量;c 为对应的类别标签;$\bar{x}_j$ 为训练样本基因 g_i 的整体均值;$\bar{x}_{c,j}$ 为训练样本基因中隶属于类别 c 的基因整体均值。

潜在基因判别实例如下,假定 g_i 实际取值为:

$$g_i=(g_{i,1}=2,g_{i,2}=2,g_{i,3}=6,g_{i,4}=42) \tag{4}$$

则式(3)中有关参量计算过程如下:

$$\begin{cases}\bar{x}_i=\sum_{j=1}^{4}g_{i,j}/4=13\\\bar{x}_{1,i}=(2+2)/2=2\\\bar{x}_{2,i}=(6+42)/2=24\\BWG(g_i)=0.385\end{cases} \tag{5}$$

如果 $0.385\geqslant BWG_{th}$,则称 g_i 为潜在基因。

同时,对于 $BWg_2=2.0$,$BWg_4=1.0$,$BWg_6=0.8$,$BWg_7=1.5$,设置阈值 $BW_{th}=0.5$,则 BWg_2

=2.0预示着 g_2 为最重要的基因,其出现频率最高,g_7 重要性其次。

步骤 2(微阵列数据) 令候选基因为 g_c,$g_c=\{g_c\in G_p\cap g_c\in \exists S_k\}$,$S_k$ 显示精度比用户给定精度阈值更高。例如,$Acc(S_k)\geqslant Acc_{th}$,则称 S_k 组建的数据集为候选基因集 G_c。

在确定所有潜在基因后,通过步骤 2 生成一个与之同等规模的预定义微阵列采样数据集 S_k。然后,利用 K 近邻算法计算每个子集的分类精度。在 K 近邻分类算法中,需测量训练样本和测试样本间距,并据此选取前 k 个基因组成最终微阵列采样数据。

步骤 3(Hadoop 并行化) 如前所述,本文创新点在于实现基因信息提取的并行化,根据 2.1 节 Hadoop 并行计算框架,令 Map 过程的输入为上述步骤确定的微阵列采样数据集$\langle g_i,(g_{i,1},g_{i,2},\cdots,g_{i,M})\rangle$,输出数据为$\langle S_1,g_i\rangle,\langle S_2,g_i\rangle,\cdots,\langle S_k,g_i\rangle$,构成键值对。然后,在 Reduce 过程中,利用键值对及子集标识符 id,输出$\langle S_k,(g_{i_1},g_{i_2},\cdots,g_{i_k})\rangle$,实现基因数据的归约。Hadoop 并行化过程见图 1,算法伪代码具体过程如算法 1 所示。

算法 1 潜在基因数据集选取

输入 微阵列数据集$[N\times M]$

输出 潜在基因数据集$\langle S_k,(g_{i_1},g_{i_2},\cdots,g_{i_k})\rangle$

```
Begin Map operation
    read(g_i);//获取微阵列数据集中基因 g_i 表达式
    //根据式(3)对 g_i 进行评价
    BWG(g_i) = SS_Bj / SS_wj = Σ_{i=1}^{m} Σ_{c=1}^{k} I(y_i = c)(x̄_{c,j} − x̄_j)^2 / Σ_{i=1}^{m} Σ_{c=1}^{k} I(y_i = c)(x̄_{i,j} − x̄_{c,j})^2
    if BWG(g_i) ≥ BWG_th then
      generate(⟨S_k,g_i⟩);
      //根据 BWG(g_i)生成⟨S_k,g_i⟩
      输出⟨S_k,g_i⟩;
    end
end
Begin Reduce operation
    ⟨g_i,(g_{i,1},g_{i,2},⋯,g_{i,M})⟩→⟨S_1,g_i⟩,⟨S_2,g_i⟩,⋯,⟨S_k,g_i⟩;//在⟨S_k,g_i⟩提取同类 g_i
    输出⟨S_k,(g_{i1},g_{i2},⋯,g_{ik})⟩;
end
```

步骤 4(基因子集分类精度) 在分类方面,使用基于欧氏距离函数的 K 近邻排序方法。首先,每个 Map 过程的输入为$\langle S_j,(g_{i_1},g_{i_2},\cdots,g_{i_k})\rangle$,并计算每个训练样本和测试样本在类别 S_j 下的欧式距离。欧氏距离可定义如下:

$$dis_j=\sqrt{(g_{i_1j}-t_{s_li_1})^2+(g_{i_2j}-t_{s_li_2})^2+\cdots+(g_{i_kj}-t_{s_li_k})^2} \tag{6}$$

其中,dis_j 为训练样本 j 和测试样本 l 的距离。在执行完 Map 后,Reduce 输入为$\langle S_i,(dis_1,dis_2,\cdots,dis_j)\rangle$,并确定$\langle S_i,(ts_1,T/F)\rangle$,$\langle S_i,(ts_2,T/F)\rangle,\cdots,\langle S_i,(ts_k,T/F)\rangle$,其中,$T/F$ 表示测试样本是否被正确分类,并且输出 $Acc(S_i)$。

在并行化过程中,将每个子集同时分配给多个 Map 和 Reduce 执行过程。在该过程中,输入文件包含$\langle S_j,(g_{i_1},g_{i_2},\cdots,g_{i_k})\rangle$,假定微阵列矩阵可被每个 Map 和 reduce 过程进行访问。首先,每个 Map 过程计算训练样本的距离值,并且输出$\langle S_j,dis_{j,s}\rangle$键值对。

在执行完所有 Map 过程后,Reduce 过程根据 $dis_{j,s}$对键值对$\langle S_j,dis_{j,s}\rangle$进行排序,并获得 top-$k$ 最小键值对$\langle S_j,dis_{j,s}\rangle$。然后,在接下来的 Map 过程中,对 top-$k$ 样本标签进行收集,并且根据 K 近邻训练样本对测试样本类别标签进行预测。Reduce 过程对正确的类标签进行检测,并据此生成每个子集的预测精度值。如果 $Acc(S_j)\geqslant Acc_{th}$,其中,$S_j$ 为候选集,Reduce 过程输出$\langle S_j,Acc(S_j)\rangle$。

HMP-KNN 算法伪代码具体过程如算法 2 所示。

算法 2 HMP-KNN 并行算法

算法 2.1 基因分类精度计算

输入 潜在基因数据集 $read(\langle S_k,(g_{i_1},g_{i_2},\cdots,g_{i_k})\rangle)$

输出 top-k 排序基因

```
Begin HMP 过程 1
  Map:
  read(⟨S_k,(g_i1,g_i2,⋯,g_ik)⟩);//读取潜在基因子集
  for 测试样本 t_sk do
    for 训练样本 tr_sj do
      //计算每个训练样本和测试样本的欧式距离
      dis_j = √((g_i1j − t_sli1)^2 + (g_i2j − t_sli2)^2 + ,⋯,(g_ikj − t_slik)^2)
      累积每个潜在基因子集的欧式距离;
      g_i←dis_i;S_i←(dis_1,dis_2,⋯,dis_j);
    end
      emit(⟨S_i,(dis_1,dis_2,⋯,dis_j)⟩);
    end
    Reduce:
    for 潜在基因子集 Sido
      根据⟨S_i,(ts_k,T/F)⟩获得所有 t_sk 的 K 近邻样本;
      根据⟨S_i,(ts_k,T/F)⟩计算 Acc(S_i);
      ifAcc(S_j)≥Acc_th thenemit(⟨S_i,Acc(S_i)⟩);end
  end
end
```

算法 2.2 top-k 排序提取

```
Begin HMP 过程 2
  Map:
  rand(⟨G_c,(g_id1,g_id2,⋯,g_idk)⟩);//获取⟨G_c,(g_id1,
//g_id2,⋯,g_idk)⟩
```

```
for 候选基因子集 GC 个体 g_i do
    emit(〈g_i,1〉);//输出〈g_i,1〉
end
Reduce:
for i = 1:k do
    if g_i = g_id_i then freq_i = freq_i + 1;end
    emit(〈g_i,freq_i〉) //计算 g_i 出现频率,并输出〈g_i,
//freq_i〉
end
sort(〈g_i,freq_i〉)←freq_i;//按照 freq_i 对〈g_i,freq_i〉进行
//排序
extract(〈g_i1,freq_i1〉,〈g_i2,freq_i2〉,…,〈g_ik,freq_ik〉);//提
//取 top-k 基因个体
```

步骤 5(top-k 排序提取)　在执行完算法伪代码 2.1:基因分类精度计算后,可获得候选基因子集 G_C,具有较高预测精度。在该步骤中,根据候选子集 G_C 中个体 g_i 的出现频率 $freq_i$,作为评价个体 g_i 重要性的指标,构建〈g_i,$freq_i$〉键值对,并且根据 $freq_i$ 值进行排序,以此获取 top-k 个基因个体,作为最终预测数据集进行输出。具体过程参见算法伪代码 2.2:top-k 排序提取过程。步骤 4 和步骤 5 算法框架如图 3 所示。

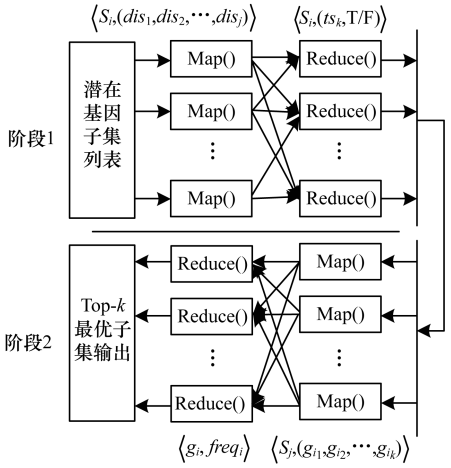

图 3　两阶段并行基因信息提取

本节并行基因信息提取算法共有 2 个步骤,均涉及到基因的 Map-Reduce 过程,其中,潜在基因数据集选取过程计算复杂度为 $O(M \cdot N)$,M 和 N 为基因微阵列数据规模;HMP-KNN 并行算法过程计算复杂度为 $O(k\lg(k))$,k 为基因查询提取数量。

3　实验结果与分析

3.1　测试数据集

测试数据集采用文献[15]用到的 3 种基因测试数据集,具体情况如下:

测试数据集 1(Leukemia 数据集)　2 种类型的急性白血病基因数据:髓系急性白血病(AML)和淋巴细胞急性白血病(ALL)。采用 Affymetrix 寡核苷酸检测到 6 817 条基因,并表达为微阵列形式,在该阵列数据中,含有 80 例急性淋巴细胞白血病数据(63 例B 细胞 ALL,17 例 T 细胞 ALL)和 50 例髓系急性白血病数据。

测试数据集 2(SRBCT 数据集)　小圆蓝细胞瘤样本数据库,该样本库中有 162 例样本,共有 2 308 条基因数据。有 4 种类型:Burkitt 淋巴瘤(BL),Ewing 肉瘤(EWS),横纹肌肉瘤(RMS)和神经母细胞瘤(NB)。

测试数据集 3(Colon 数据集)　使用 Affymetrix 寡核苷酸采集的 47 例正常组织和 93 例结肠癌的共 6 500 条基因样本。

3.2　精度提取结果与分析

实验采用 8 台主机构建并行集群,处理器为 i5-2.5 GHz,内存大小为 6 GB,存储容量为 500 GB,操作系统为 Ubuntu 11.10。其中,一台主机作为主节点;其余主机作为工作节点。每个节点执行两阶段的 Map 和 Reduce 操作,因此,该系统共需执行 16 次Map 和 Reduce 操作,HDFS 块大小为 32 MB,每个 HDFS 块都有 3 个镜像。对比算法选取文献[15]实验数据及单线程 top-k 算法实验结果。图 4给出top-k和并行 top-k 排序中 k = 10 ~ 100 不同取值的基因提取正确率。

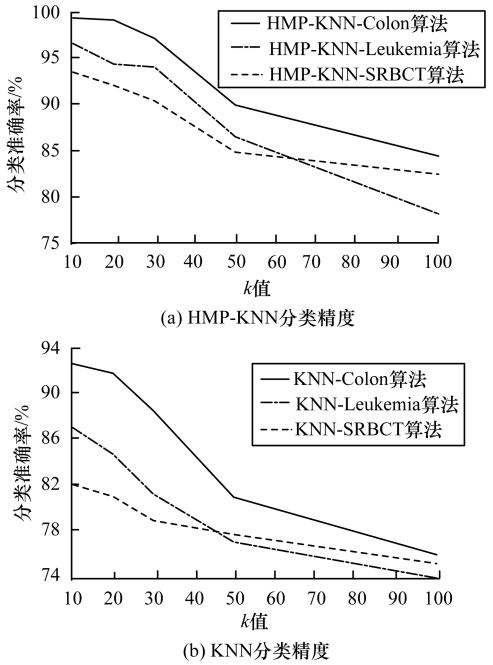

图 4　不同 top-k 数量下的分类精度

图 4 分别给出并行 HMP-KNN 和单线程 K 近邻算法在不同 top-k 数值选取情况下的分类精度,测试数据集选取 3.1 节 3 种数据集。从曲线变化趋势看,当 top-k 数值选取过大时,2 种算法的分类精度均

出现下降。候选基因越多,产生的噪声越多,影响预测准确率,特别是超过临界状态时。但并非该数值越小越好,需根据用户提取基因数量而定,在规定的提取数值下,选择下限提取数量。

选取文献[15]中算法实验数据、单线程 top-k 算法及本文 HMP-KNN 算法的预测精度进行对比,同时为考虑分类全面性,top-k 数量选取 $k=30$,如表 1 所示。

表 1 提取精度对比 %

算法	Leukemia	SRBCT	Colon
top-k 算法	83.2	80.8	89.3
文献[15]算法	90.1	86.7	94.2
HMP-KNN 算法	94.6	90.3	97.3

从表 1 数据看,本文 HMP-KNN 算法在 3 种测试数据库上的提取精度均超过 90%,要优于另外 2 种对比算法。这也体现了并行计算在精度选取上的优势。

3.3 运行时间结果与分析

对应于精度指标,算法的另一指标是运行时间,该指标影响算法的实用性。在算法计算时间指标上,3 种算法的仿真结果如表 2 所示。HMP-KNN 算法运行时间随节点数变化情况如图 5 所示。分类精度随节点数变化情况如图 6 所示,选取参数 $k=50$。

表 2 运行时间对比 s

算法	Leukemia	SRBCT	Colon
top-k 算法	213.4	187.5	116.3
文献[15]算法	256.4	216.7	168.4
HMP-KNN 算法	129.3	126.4	53.7

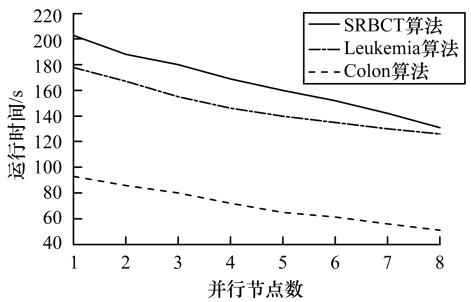

图 5 节点数变化对运行时间的影响

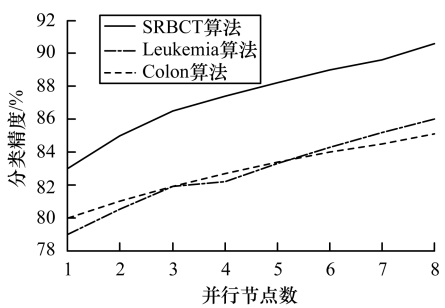

图 6 节点数变化对分类精度的影响

图 5 给出 HMP-KNN 算法在不同计算节点情况下,在 3 种测试数据库上的运行时间变化曲线。从图中可以看出,在上述测试对象中,运行时间都随着并行节点数的增加,呈现逐渐下降的趋势,体现了节点数增加对计算效率的提升。表 2 给出 3 种对比算法在 3 种测试数据库上的对比结果,从表 2 中可看出,HMP-KNN 算法的运行时间最短(并行节点数选取 8),top-k 算法计算速度其次,文献[15]算法计算速度较慢。

图 6 给出算法在 3 种测试数据库上分类精度指标随并行节点数增加的变化情况,可以看出,随着并行节点数增加,算法的分类精度均在提高。在 Leukemia 数据库上的精度提高值为 7.6%,在 SRBCT 数据库上的精度提高值为 7.1%,在 Colon 数据库上的精度提高值为 5.2%。当节点数为 1 时,等同于 3.2 节单线程 K 近邻算法。

综合前述基因提取精度计算指标,可知 HMP-KNN 算法不仅在计算精度上要优于文献[15]算法及单线程 top-k 算法,并且在计算效率上要快于这 2 种对比算法。从计算精度指标看,单线程 top-k 算法虽然低于文献[15]算法,但在运行效率上,单线程 top-k 算法要快于文献[15]算法,2 种对比算法各有特点。

3.4 大数据测试集构造

由于前述数据集规模相对于 MapReduce 并行计算模型仍不是很大,但为进一步验证算法在大数据集上的性能的可扩展性,从前列腺癌数据集中产生 3 个合成数据集,数据集信息如表 3 所示。

表 3 测试数据集信息

数据集	采样点	基因数	类别	数据集大小/MB
PX_1	220	15 600	2	26
PX_2	310	15 600	2	45
PX_3	530	15 600	2	79

选取评价指标与增加并行节点数,对于分类精度及算法运行时间影响的仿真结果如图 7 和表 4 所示。

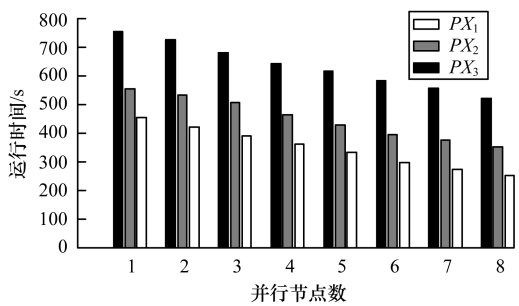

图 7 算法运行时间

表 4　算法分类精度　%

节点数	PX_1	PX_2	PX_3
1	80.1	75.5	72.3
2	82.3	76.9	73.8
3	83.5	78.3	75.1
4	84.7	79.2	76.9
5	85.4	80.4	77.5
6	86.2	81.6	79.3
7	86.6	82.5	80.2
8	87.6	83.6	80.7

从图 7 算法运行时间看,随着节点数量的增大,算法的运行时间大幅降低,体现了并行计算在大数据上的可行性及优势,并且随着计算节点的继续增加,算法运行效率会有进一步提升。在分类精度上,随着节点数的增加,分类精度亦在提高,多节点并行执行有助于分类精度的提高。

4　结束语

本文利用 Hadoop 框架 MapReduce 并行计算模型,实现两阶段并行 top-k 基因信息选取,提高了算法精度和运算效率。仿真结果显示,所提算法优于文献[15]算法及单线程 top-k 算法,并证明了其有效性。在算法扩展性方面,通过前列腺癌数据集构造大数据集,验证了算法在大数据集上的可扩展性。算法能够继承 Hadoop 编程模型特点,实现更大型数据库的简单化构建。由于受实验条件所限,对于并行节点的优化选取,还需要进一步研究。

参考文献

[1] Katsigiannis K,Zacharia E,Maroulis D. Grow-cut Based Automatic cDNA Microarray Image Segmentation[J]. IEEE Transactions on NanoBioscience, 2015, 14(1): 138-144.

[2] Sakashita H, Akamine S, Ishida T. Erratum to: Identification of the NEDD4L Gene as a Prognostic Marker by Integrated Microarray Analysis of Copy Number and Gene Expression Profiling in Non-small Cell Lung Cancer[J]. Annals of Surgical Oncology, 2014,21(4):783-792.

[3] 于化龙,顾国昌,赵　靖. 基于 DNA 微阵列数据的癌症分类问题研究进展[J]. 计算机科学,2010,37(10):16-21.

[4] 印　莹,赵宇海,张　斌. 时序微阵列数据中的同步和异步共调控基因聚类[J]. 计算机学报,2007,30(8):1302-1309.

[5] Patrick C H, Keith C C, Yao Xin. An Evolutionary Clustering Algorithm for Gene Expression Microarray Data Analysis[J]. IEEE Transactions on Evolutionary Computation,2006,10(3):296-314.

[6] Chan S C,Wu Haichang,Tsui K M. A New Method for Preliminary Identification of Gene Regulatory Networks from Gene Microarray Cancer Data Using Ridge Partial Least Squares with Recursive Feature Elimination and Novel Brier and Occurrence Probability Measures[J]. IEEE Transactions on Systems, Man and Cybernetics, Part A:Systems and Humans,2012,42(6):1514-1528.

[7] 张　靖,胡学钢,李培培. 基于迭代 Lasso 的肿瘤分类信息基因选择方法研究[J]. 模式识别与人工智能,2014,27(1):49-59.

[8] 史建军,缪裕青. 微阵列数据中 Top-k 频繁闭合项集挖掘[J]. 计算机工程,2011,37(2):60-62.

[9] 宋　佳,许　力,孙　洪. 基于图论的 DNA 微阵列数据聚类算法[J]. 计算机工程,2015,40(5):36-39.

[10] Lee C P, Leu Y. A Novel Hybrid Feature Selection Method for Microarray Data Analysis[J]. Application Software Computing,2011,11(1):208-213.

[11] Dolled-Filhart M,Ryden L,Cregger M. Classification of Breast Cancer Using Genetic Algorithms and Tissue Microarrays[J]. Clinical Cancer Research, 2006, 12(21):6459-6468.

[12] 罗美淑,刘世勇,石　磊. 一种基于微阵列数据的集成分类方法[J]. 计算机应用研究,2010,27(1):104-106.

[13] 马友忠,慈　祥,孟小峰. 海量高维向量的并行 top-k 连接查询[J]. 计算机学报,2015,38(1):86-97.

[14] 彭敦陆,王　丽,霍　欢. MapReduce 环境下支持精确查询的嵌套式数据索引技术[J]. 小型微型计算机系统,2015,36(2):220-226.

[15] 杨广源,付旭平,黄　燕. 一种基于非线性降维和 Procrustes 分析的基因选取方法[J]. 复旦大学学报:自然科学版,2009,48(3):338-347.

编辑　索书志