计算机系统应用 ISSN 1003-3254, CODEN CSAOBN
Computer Systems & Applications,2017,26(8):141−146 [doi: 10.15888/j.cnki.csa.005928]

E-mail: csa@iscas.ac.cn
http://www.c-s-a.org.cn
Tel: +86-10-62661041

Hadoop 平台分布式 SVM 算法分类研究①

满蔚仕, 吉元元

(西安理工大学 自动化与信息工程学院, 西安 710048)

摘　要: 随着大数据的发展, 分布式支持向量机(SVM)成为该领域研究热点. 传统层级分布式 SVM 算法(Cascade SVM), 在 Hadoop 平台下寻找全局最优支持向量的过程十分缓慢. 本文提出了一种改进方法, 先将传统的网格法与粒子群(PSO)算法结合, 改进了单机 PSO 算法, 再将单机 PSO 算法与 Hadoop 平台结合实现了一种新型卫星并行 PSO 算法(NPP-PSO). 实验结果表明, 相比于单机 SVM 算法, 本文的分布式 SVM 算法, 在保证了准确率的前提下大幅提高了计算速度; 而使用 NPP-PSO 参数寻优后的分布式 SVM, 分类准确率相比于分布式 SVM 算法又有了明显提高.

关键词: 机器学习; Hadoop; MapReduce; 分布式支持向量机; 分布式粒子群算法

引用格式: 满蔚仕,吉元元.Hadoop 平台分布式 SVM 算法分类研究.计算机系统应用,2017,26(8):141–146. http://www.c-s-a.org.cn/1003-3254/5928.html

Research on Distributed SVM Classification Based on Hadoop Platform

MAN Wei-Shi, JI Yuan-Yuan

(Xi'an University of Technology, Xi'an 710048, China)

Abstract: With the development of big data, distributed support vector machine (SVM) has become a hot research topic in this field. The process of finding the global optimal support vector in the Hadoop platform is long under the traditional hierarchical Cascade SVM algorithm. This paper presents an improved method by firstly combining the traditional grid method and the particle swarm optimization(PSO) algorithm to improve the PSO algorithm. And a new satellite parallel PSO algorithm is realized by combining the single machine PSO algorithm and the Hadoop platform (NPP-PSO). The experimental results show that compared with the single SVM algorithm, the distributed SVM algorithm cannot only ensure the accuracy but can also greatly boost the computation speed. With the wide use of NPP-PSO distributed SVM, the classification accuracy has improved significantly.

Key words: machine learning; Hadoop; MapReduce; distributed suppot vector machine; distributed particle swarm optimization

引言

一直以来, 计算机和互联网在全球范围内保持着高速发展, 未来也将会有更多人接入到可无缝浏览并与移动设备绑定的互联网世界[1]. 互联网早已深入到人们生活的各个角落, 随之而来的是数据增长方式和途径越来越多. 一些大型网站, 比如国外的 Facebook, 国内的新浪微博、淘宝网等, 每天都有大批用户参与其中并因此而产生大量数据. 面对如此巨大的数据, 一台计算机已经无法存储, 也无法在可承受的时间内处理这些数据. 大数据的时代已经到来[2], 面对大量的、复杂的、具有不同来源的数据[3], 人们对其存储和管理又有了新的要求, 其中分布式平台是一种很好解决方案.

① 收稿时间: 2016-11-21; 采用时间: 2016-12-19

基于谷歌公司发表的三篇重要论文[4-6], 随之而生了许多分布式技术和产品. Hadoop 是由 Apache 基金会所开发的开源的分布式系统, 2005 年作为 Lucene 的子项目 Nutch 的一部分被正式引入. Hadoop 平台主要由分布式文件系统(HDFS)和 MapReduce 框架[7,8]组成, 还包含有若干其他子项目, 如 HBase, Hive 等. Hadoop 作为第一个流行的大数据平台被全世界各大公司广泛应用, 如雅虎、百度都有自己的集群.

现代科学技术和商业中的数据量正在以一个巨大的速度增长, 使现有技术很难处理如此大的数据量[9]. 随着数据量的增大, 许多传统的机器学习算法都面临了严峻的挑战[10], 它们可能需要根据需求而被改写或者重新设计, 支持向量机(Support Vector Machine, SVM)就是其中之一. 因为 SVM 算法本身不需要繁多的参数设置, 且在二分类问题上表现出了卓越性能因而被广泛应用. SVM 算法是一种有监督的机器学习方法, 通常用来进行模式识别、回归预测等. 然而, SVM 算法的计算复杂度主要与支持向量的数量有关, 利用支持向量机对大数据集进行分类或回归计算时, 寻找全局最优解的过程将耗费大量时间, 并且需要占用大量的计算机资源. 分布式计算是近年提出来的一种新的计算方式, 它将大规模的计算任务分配给多个计算机进行并行处理, 并将处理结果进行合并. 分布式计算实现了大规模任务的并行化处理, 大大提高了算法的工作效率. 因此在大数据集情况下对 SVM 算法的主要研究目标是如何分布式化算法结构, 并且在保证准确率的前提下提高训练速度, 使 SVM 算法可以成为在大数据集情况下更可行更有效的选择.

1 Hadoop 平台下分布式 SVM

1.1 训练样本与数据预处理

本文使用的实验数据集为 UCI 标准数据集 covtype.data. 该数据集属于 2 分类文本数据, 记录了位于美国罗斯福国家森林公园的森林覆盖种类, 总共包含样本数 581012 个, 使用了 54 维的属性数据. 除零值属性外, 非零属性一般为 12 个左右, 因此该数据集属于稀疏矩阵. 图 1 展示了 UCI 网站提供的数据说明, 包括了: 数据任务类型、总样本数、总属性数、数据是否丢失等相关信息.

本文单机 SVM 算法使用著名开源算法包 LIBSVM, 它是由台湾大学林智仁教授领导开发的一个实用的 SVM 工具包[11]. 在进行单机 SVM 训练之前需要注意的是, 训练集和测试集的数据划分要尽可能保持数据分布的一致性, 避免因数据划分而引入额外的偏差, 对最终的结果产生影响. 在分类任务中要保证类别比例相似. 统计本测试数据集的类别 1 与类别 2 的比值, 分别约为 1:1. 因此按照 1:1 的比例再次处理样本数据, 使数据均匀排列, 结果如图 2 所示.

Data Set Characteristics:	Multivariate	Number of Instances:	581012	Area:	Life
Attribute Characteristics:	Categorical, Integer	Number of Attributes:	54	Date Donated	1998-08-01
Associated Tasks:	Classification	Missing Values?	No	Number of Web Hits:	106688

图 1 covtype.data 数据集说明

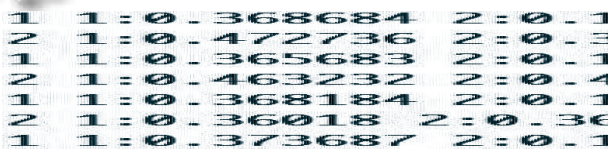

图 2 样本数据按比例排列

1.2 分布式 SVM 实现

单机 SVM 训练时间会随着训练数据量的增大呈指数级增长, 将大数据集分块并行计算可以有效提高计算效率. 本文在 Hadoop 平台下使用 MapReduce 框架实现了两层迭代的分布式 SVM 算法, 如图 3 所示.

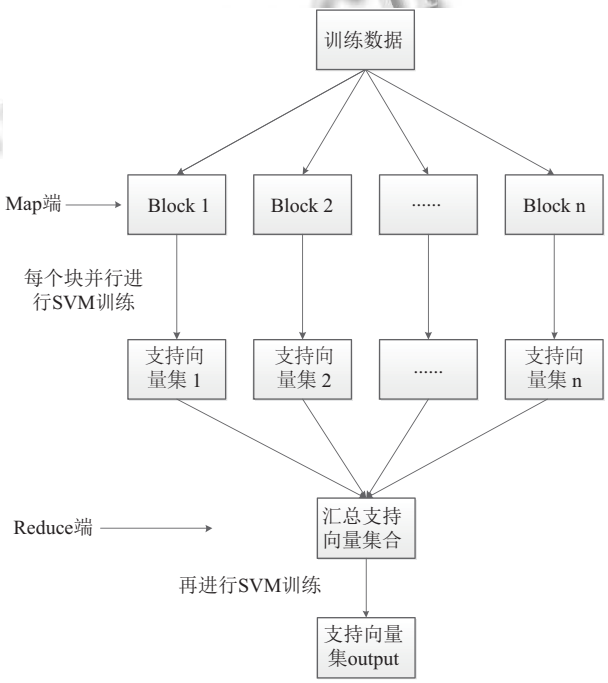

图 3 分布式 SVM 算法结构图

该分布式结构图综合考虑了训练时间, covtype.data 数据集的 SV 筛选率, 分类 model 准确率三个因素. 去掉了反馈机制, 进行 Map 端和 Reduce 端两层的 SVM 训练. 结构中 Reduce 端只使用了一个 Reduce 任务, 考虑到实验数据中最大用到了 40 万行 47.3 MB, 结合数据集的 SV 筛选率确定一个 Reduce 任务足够胜任.

由于多数机器学习算法都有训练参数选择问题, 参数不同, 训练模型的性能也会有很大差异. SVM 算法分类模型的性能与(c, g)参数的选择有很大关系. 关于 SVM 参数寻优方法众多, 但没有公认统一最好的方法. 本文针对大数据集采用分布式粒子群算法对分布式 SVM 分类模型进行参数寻优.

2 SVM 参数寻优

2.1 分布式粒子群(PSO)算法

粒子群算法(Particle Swarm Optimization, PSO)是由 James Kennedy[12]等人在 1995 年提出的. 它也是从随机解出发, 通过迭代寻找最优解, 并通过适应度来评价解的品质.

SVM 参数寻优在本质上就是选用许多组(c, g)参数大量计算对应的 SVM 分类模型准确率. 而在大数据集下, 一次单机 SVM 计算就要花费不少时间. 如进行单机 PSO 参数寻优方法, 初始化 10 个粒子, 最大迭代 30 次, 需要 5 折交叉验证计算 300 次 SVM, 这时的计算时间就已经久到无法承受了. 因此往往在面对大数据集训练时 SVM 不涉及专门的参数寻.

文献[13]实现了一种并行化 PSO 算法: 卫星并行粒子群算法(PP-PSO). PP-PSO 算法将大数据集分为若干个子集, 在每个子集上运行 PSO 算法. 子集每运行一次 PSO 算法便将所有集合的寻优结果汇总到一起进行比较, 选出其中最佳适应度粒子, 将该粒子作为中心粒子. 中心粒子以一定权重再反馈到子集中, 牵引着各个子集做下一次的 PSO 寻优计算. 这样重复迭代计算若干次就可以找到全局最优解.

图 4 为 PP-PSO 示意图, 图中外部实线圆形代表四个子集, 内部中心虚线圆形代表中心粒子. 中心粒子牵引着周围子集, 很像卫星围绕地球运动, 因此称为卫星并行粒子群算法.

基于基本 PSO 算法的 PP-PSO 算法的速度更新公式如下:

$$\begin{cases} v_{id}(t+1)=\tau_{nt}(G_1+G_2+G_3)+\tau_{pf}G_4 \\ G_4=c_3r_3(t)(P_u-x_i(t)) \end{cases} \tag{1}$$

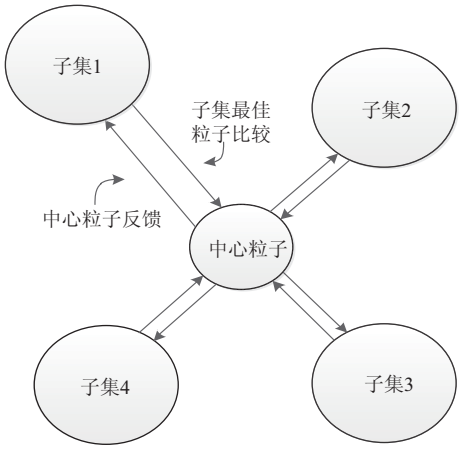

图 4 卫星并行粒子群算法

P_u 为各子群比较后的最佳粒子, 即中心粒子; c_3, r 与 G_2, G_3 前的系数没有差别. 公式中 G_4 的引入体现出了中心粒子对子群的牵引作用. τ_{nt}, τ_{pf} 为反馈系数:

$$\begin{cases} \tau_{nt}=\dfrac{2\text{atan}\left(\frac{\lambda_n}{t}\right)}{\pi} \\ \tau_{pf}=\dfrac{2\text{atan}\left(\frac{t}{\lambda_p}\right)}{\pi} \end{cases} \tag{2}$$

λ_n, λ_p 为反馈因子, 根据最大迭代次数设定. τ_{nt} 随着迭代次数 t 的增大而减小, τ_{pf} 随着迭代次数 t 的增大而增大. 这意味着在迭代初期, 各个子群的速度更新占更大权重, 中心粒子束缚性低, 有利于子群的自由搜索. 随着迭代次数的增加, 中心粒子的作用越来越大, 这有利于最后适应度的全局收敛.

2.2 Hadoop 平台分布式 PSO

在 Hadoop 平台上实现 PP-PSO 算法, 需要将算法进行 MapReduce 过程, 即在 map 端实现子群的 PSO 算法, 在 reduce 端实现中心粒子的提取. 这种方式会使得 Hadoop 集群不断提交 Job 任务, 会消耗很多时间在系统调度上. 因此本文改写 PP-PSO 算法迭代结构, 使之适合于 Hadoop 平台, 称其为 NPP-PSO(New PP-PSO)算法.

1) PP-PSO 算法在子集上进行 1 次 PSO 寻优后就进行中心粒子的提取, 而本文的 NPP-PSO 算法是在每个子集上进行若干次的迭代寻优后, 再汇聚在一起进行中心粒子提取. 增加了子集迭代寻优次数, 减小中心

粒子的提取次数, 从而将 Hadoop 集群提交 Job 的次数大幅减少.

2) 将(2)式调整为:

$$\begin{cases} \tau_{nt} = \dfrac{2\text{atan}\left(\dfrac{\lambda_n}{nt_i+t_g}\right)}{\pi} \\ \tau_{pf} = \dfrac{2\text{atan}\left(\dfrac{nt_i+t_g}{\lambda_p}\right)}{\pi} \end{cases} \tag{3}$$

n 为中心粒子提取次数, 初始值为 0; t_i 为每个子集设定迭代最大次数, t_g 为当前子集的迭代次数. NPP-PSO 算法结构图如图 5 所示.

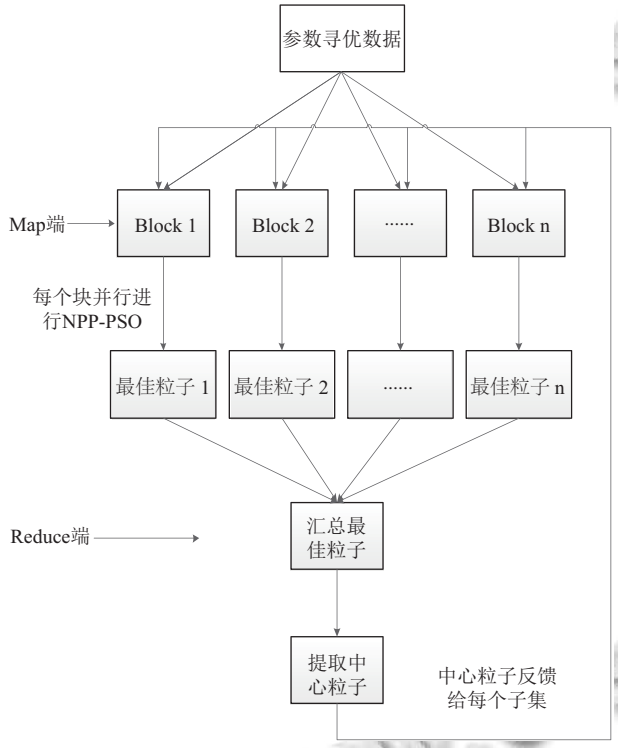

图 5 Hadoop 平台 NPP-PSO 算法

在具体 SVM 参数寻优中, 鸟群相当于初始化的粒子, 每个初始化粒子(c, g), 即 SVM 算法中需要优化的参数, 对应一个目标函数值也叫适应度, 在 SVM 分类算法中就是分类准确率. 初始的 n 个粒子在一定范围空间内进行搜索, 粒子每次移动一定距离称为速度. n 个粒子通过有限次重复迭代计算, 适应度可以在某次达到最优解, 此时迭代收敛.

数据上传至 Hadoop 集群后, 按块分成若干个子集. 在 Map 端, 第一个 map 函数首先初始化 n 个粒子, 计算它们的适应度. 因为此时的中心粒子并未初始化, 第一次的 map 函数相当于进行改进后的单机版 PSO 算法. map 函数执行完后将所有子集的最佳粒子传递给 Reduce 端, 第一次 reduce 函数对所有子集的最佳粒子进行对比, 提取出适应度最好的粒子作为中心粒子. 执行完第一次任务后, 接下来的 map 函数不用再初始化粒子, 进行 PSO 计算. Reduce 函数依旧执行中心粒子提取工作. 当达到最大迭代次数时循环结束, 整个 NPP-PSO 训练完成, 即得到 SVM 算法参数寻优的最佳参数.

3 实验

3.1 实验一

默认参数下对训练时间的比较. 首先选用不同大小的数据集: 1 万行, 10 万行, 20 万行, 30 万行, 40 万行五种数据集. 分别对单机 SVM 算法, 分布式 SVM 算法进行实验. 集群 Block 大小设定为 6 MB, 对应数据集的 Block 块数为别为: 1 块, 2 块, 4 块, 6 块, 8 块. 集群可并行运行的最大 Map 任务为 4 块, 这意味着, 30 万行和 40 万行数据会有若干 Map 任务需要排队等待. 对比效果由图 6 所示.

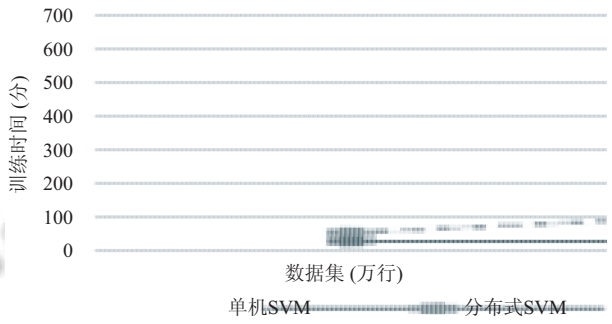

图 6 单机 SVM 与分布式 SVM 训练时间

实验结论: 随着训练样本逐渐增大, 单机 SVM 训练时间呈现指数级增长, 而分布式 SVM 训练时间增幅不大, 大大减小了训练时间. 随着训练集规模的增大, 分布式 SVM 训练的时间优势越来越明显.

3.2 实验二

在默认参数下对分类模型准确率的比较. 在实验一的基础上使用其训练出的分类模型, 同样选用不同大小的数据集(1 万行, 10 万行, 20 万行, 30 万行, 40 万行)作为预测数据集, 分别对单机与分布式 SVM 算法进行实验. 图 7 为单机 SVM 与分布式 SVM 分类模型准确率对比图.

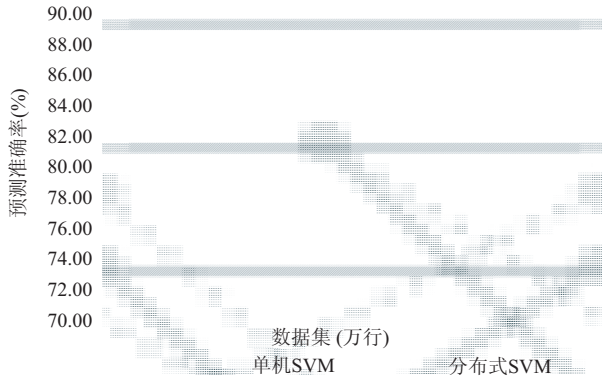

图 7 单机与分布式 SVM 分类模型准确率对比

实验结论: 随着训练样本规模的增大, 在默认参数的情况下分布式 SVM 的分类模型准确率相对于单机 SVM 没有受到明显的影响. 除第一组小数据集情况外, 其余数据集单机 SVM 分类模型准确率只略好于本文设计的分布式 SVM 结构, 平均准确率相差 1.9%.

3.3 实验三

因为训练参数的选择会影响训练速度, 因此本实验依然在默认参数下进行. 为测试分布式 SVM 算法的性能与集群中 Map 任务的数量的关系, 本实验采用了加速比来衡量该并行算法在训练时间上的提升速率. 加速比的计算方法如下所示:

$$P = T_1/T_2 \tag{4}$$

其中 P 表示加速比, T_1 表示单机 SVM 算法的训练运行时间, T_2 表示 Hadoop 平台分布式 SVM 算法的训练运行时间. 本实验在数据集 covtype.data 上按照 Block 自定义的 6 MB 大小分别抽取了四组不同大小(6 MB, 12 MB, 18 MB, 24 MB)的训练数据集. 四组数据集分别占 1 至 4 个 Block, 也分别对应 1 至 4 个 Map 任务. 实验结果加速比如图 8 所示.

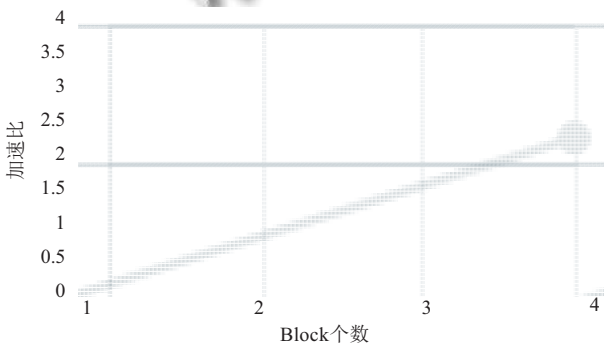

图 8 不同 Block 的加速比

实验结论: 当 Hadoop 集群中的节点为 1 个 Map 任务时, 单机 SVM 算法所用时间短. 随着集群中并行 Map 任务数的增加, 分布式 SVM 算法的加速比逐渐提高. 即训练时间提升的幅度越来越明显.

3.4 实验四

为了验证分布式 PSO 算法的参数寻优效果. 本实验使用实验一的四组数据集进行参数寻优. 1 万行数据不涉及分布式 PSO 算法, 本实验不再使用. 因为在大数据集情况下使用 K 折交叉验证的单机 PSO 寻优时间实在太长(计算时间以月为单位), 因此本实验不在训练时间上做对比.

实验选定子集粒子数为 5 个, 子集最大迭代次数 5 次, 中心粒子提取次数 30 次, 因实验数据较大, 选用 3 折交叉验证即可得到相对稳定的准确率. 对比效果由图 9 所示. 实验结论: 面对四组大数据集训练样本, 经过分布式 PSO 参数寻优后的分类模型准确率明显好于默认参数下的结果, 四组数据平均准确率提高了约 7.6%.

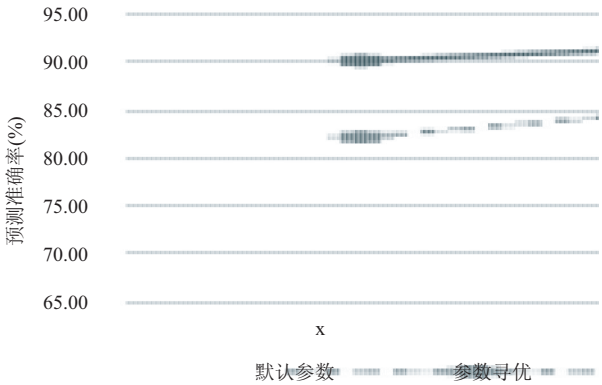

图 9 分布式 SVM 与寻优后 PSO 模型准确率对比

4 结论

随着大数据的发展, 将传统单机的机器学习方法与大数据平台有机的结合在一起成为近年来的研究热点. 本文将 Hadoop 平台与分布式 SVM 算法结合; 针对 SVM 参数寻优方法, 本文将单机 PSO 算法进行改进后, 并以其为基础, 将 PP-PSO 分析结果与 Hadoop 平台特点相结合, 实现了 NPP-PSO 算法. 本文设计的四组实验可验证 Hadoop 平台分布式算法性能. 结果表明了本文实现的分布式 SVM 结构可以在保证准确率的情况下大幅度提高训练速度; 分布式 PSO 算法在参数寻优上效果显著, 相比于默认参数情况, 大幅提升了分布式 SVM 分类模型准确率.

参考文献

1 Bernstein D. The emerging Hadoop, analytics, stream stack for big data. IEEE Cloud Computing, 2014, 1(4): 84–86. [doi: 10.1109/MCC.2014.90]

2 维克托•迈尔•舍恩伯格. 大数据时代: 生活、工作与思维的大变革. 周涛译. 杭州: 浙江人民出版社, 2012.

3 Jain A, Bhatnagar V. Crime data analysis using pig with Hadoop. Procedia Computer Science, 2016, 78: 571–578. [doi: 10.1016/j.procs.2016.02.104]

4 Ghemawat S, Gobioff H, Leung ST. The Google file system. ACM SIGOPS Operating Systems Review, 2003, 37(5): 29–43. [doi: 10.1145/1165389]

5 Dean J, Ghemawat S. MapReduce: Simplified data processing on large clusters. OSDI, 2004, 51(1): 107–113.

6 Chang F, Dean J, Ghemawat S, *et al*. Bigtable: A distributed storage system for structured data. ACM Trans. Computer Systems, 2008, 26(2): 4.

7 ISSA JA. Performance evaluation and estimation model using regression method for Hadoop WordCount. IEEE Access, 2015, 3: 2784–2794. [doi: 10.1109/ACCESS.2015.2509598]

8 Nghiem PP, Figueira SM. Towards efficient resource provisioning in MapReduce. Journal of Parallel and Distributed Computing, 2016, 95: 29–41. [doi: 10.1016/j.jpdc.2016.04.001]

9 Hbibi L, Barka H. Big data: Framework and issues. Proc. of 2016 International Conference on Electrical and Information Technologies (ICEIT). Tangiers, Morocco. 2016. 485–490.

10 Bello-Orgaz G, Jung JJ, Camacho D. Social big data: Recent achievements and new challenges. Information Fusion, 2016, 28: 45–59. [doi: 10.1016/j.inffus.2015.08.005]

11 上海交通大学模式分析与机器智能实验室. LibSVM-2.6 程序代码注释. http://www.doc88.com/p-6159926915557.html. [2016-03-25].

12 Kennedy J, Eberhart R. Particle swarm optimization. Proc. of IEEE International Conference on Neural Networks, 1995. Perth, WA, Australia. 1995, 4. 1942–1948.

13 程兴国. 仿生算法的动态反馈机制及其并行化实现方法研究[博士学位论文]. 广州: 华南理工大学, 2013.