ISSN 1000-9825, CODEN RUXUEW
Journal of Software, Vol.21, Supplement, December 2010, pp.284−289

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel/Fax: +86-10-62562563

一种基于迭代聚类的并行应用性能分析方法*

朱 鹏+, 李 巍, 李云春

(北京航空航天大学 网络技术北京市重点实验室,北京 100191)

An Iterative Clustering Based Approach for Parallel Performance Analysis

ZHU Peng+, LI Wei, LI Yun-Chun

(Beijing Key Laboratory of Network Technology, Beihang University, Beijing 100191, China)

+ Corresponding author: cookpan001@gmail.com, http://www.buaa.edu.cn

Zhu P, Li W, Li YC. An iterative clustering based approach for parallel performance analysis. ***Journal of Software***, **2010,21(Suppl.):284−289.** http://www.jos.org.cn/1000-9825/10029.htm

Abstract: With the development of supercomputers, the CPU core numbers of which come to several hundreds of thousands, and on which the complexity of the applications run are increasing. Therefore, in order to optimize the source code of the programs, developers of parallel applications need to measure the performance of parallel applications and make a useful analysis, so that they can improve the performance of the applications. However, due to a substantial increasing of the CPU core numbers, performance measurement will produce vast amounts of performance data, and then, how to deal with massive data is a very critical problem for parallel performance analysis. A new approach, named Iterative based Clustering Approach for Parallel Performance Analysis (ICAPPA), is proposed for parallel performance analysis in this paper. In this approach, clustering method of data mining technique, which is used to processing massive data, will be carried out iteratively for the result in some conditions after previous clustering, to find out the dominating functions and processes of the parallel performance. And Bayesian Information Criteria (BIC) is applied to evaluate the result of clustering method. By using BIC score, whether iterative clustering applied to the result is reliable or not can be decided. And at the end of this paper, the validity of that approach is verified by experimental analysis.

Key words: massive data; parallel application; clustering analysis; performance measurement; performance analysis

摘 要: 随着超级计算机的发展,其使用到的核心数逐渐达到数十万,而且运行于其上的应用的复杂性也不断加大.因此,开发人员需要对并行应用的性能进行测量,并做出分析,以便对程序源码进行优化,提高程序的执行效率.但是由于核心数的大量增加,对并行程序性能进行测量将得到海量的性能数据,如何处理海量性能数据,以便分析并行程序性能成为一个难点.介绍了一种基于迭代聚类的并行应用性能分析方法,该方法使用数据挖掘的聚类算法处理处理海量性能数据,并可以根据条件迭代执行,确定影响并行程序性能的函数和进程,然后通过贝叶斯信息准则评价聚类结果,以确定迭代聚类的可靠性,最后用实验证明了方法的有效性.

关键词: 海量数据;并行应用;聚类分析;性能测量;性能分析

* Supported by the National High-Tech Research and Development Plan of China under Grant No.2007AA01A127 (国家高技术研究发展计划(863))

Received 2010-06-15; Accepted 2010-12-10

随着超级计算机的发展,其使用到的核心数达到数十万,IBM 的 RoadRunner 核心数有 122 400 个,而 Cray XT5 使用的计算核心更是达到了 224 162 个,而且运行于其上的应用的复杂性也不断加大,其复杂性和规模使程序的运行效率难以达到预期.因此,开发人员需要对并行应用进行监测,获得丰富的性能数据,并做出分析,以便对程序进行优化,提高执行效率,达到提高程序性能的目的,因此,如何处理海量性能数据成为了一个难点.

数据挖掘试图从海量数据中找出有用的信息,并可以对海量数据进行处理和分析,针对并行应用性能监测得到的海量性能数据,随着超级计算机的大规模并行程序的应用,只使用人工可视化工作来进行性能分析,变得很不现实.而借助数据挖掘算法,可以减少性能数据的规模、发现性能数据之间的关系,大大提高性能分析的效率,减少了人工分析的复杂度.聚类分析是一种探查数据结构的工具.聚类分析的核心是聚类,即将对象划分为簇,使得同一个簇的对象相似,而不同的簇的对象相异.

本文提出一种基于聚类的并行应用性能分析方法.第 1 节介绍相关研究工作.第 2 节讲述基于迭代聚类的并行应用性能分析方法,及评价方法.第 3 节为实验分析.第 4 节总结全文并提出下一步研究工作.

1 相关研究

SimPoint[1−3]是用来查找并行程序合适的模拟点,即能够表征程序性能的若干程序段,减少多次重复对程序进行模拟运行,以分析程序的性能问题.该项目将应用程序的行为表征为一个基本块向量,在基本块向量中的每一维表示某一段代码运行时间的百分比.然后将程序的执行过程沿时间轴分成若干小段,小段之间进行对比,使用 *k*-means 算法[4]进行聚类总结.SimPoint 中将程序分成若干基本块向量,每一维代表一段代码执行时间比例,在基本块向量中的每一维表示某一段代码运行时间的百分比.不过没有从进(线)程角度考虑,即没有考虑并行应用程序的各个进(线)程在执行过程中的差异性.

TAU[5,6]中提供了一个用于复杂性能分析的工具 PerfExplorer[7,8],该工具目标是适用于大规模的并行性能分析场景.PerfExplorer 使用了包括聚类等数据挖掘算法,用于减少性能数据,降低分析复杂度,进行简单分析工作.PerfExplorer 调整了 SimPoint 的方法,将簇中每个向量转换为针对每个进(线)程的执行,每个向量中的每一维数据,表示某段代码运行时间或时间比例.这种方法没有考虑各个函数总体上的性能问题,只考虑到进(线)程一级,剩余的工作由可视化来完成.

2 基于迭代聚类的性能分析方法

2.1 数据预处理

对于大规模并行应用来说,产生的性能数据是多维的,而随着数据维数的增加,数据之间的区分界限变得模糊,不容易使用聚类算法进行分析.因此就需要使用降维技术对数据进行预处理,压缩数据属性.对于数据属性较少的情况,数据预处理可以设定阈值过滤相关属性;但对于数据属性量较大的情况,可以需要使用数据降维方法减少数据属性,本文采用的降维方法是主成分分析.

主成分分析(PCA)[9]是通过对原始变量的相关矩阵或协方差矩阵内部结构的研究,将多个变量转换为少数几个综合变量即主成分,从而达到降维目的的一种线性降维方法.这些主成分能够反映原始变量的绝大部分信息,它们通常表示为原始变量的线性组合.对于本文提出的性能分析方法,可以在进行每次迭代之前,用于对输入数据进行一次数据预处理,剔除掉其中差异性较小的属性,并将结果作为聚类分析的输入.

2.2 方法描述

本文提出的性能分析方法,需要分别从函数和进程两个角度组织性能数据,分别进行聚类.

1) 为每个进(线)程建立多维向量,向量中的每一维数据表示该进程中各个函数的开销,可以确定在哪些进(线)程中的哪些函数的开销较大,可能存在性能问题,再辅之以可视化分析,可以提高性能分析的可扩展性,适合于大规模并行应用性能分析的场景.

2) 然后再为各个函数建立一个多维向量,向量中的每一维表示该函数在各个进(线)程中的时间比例,通过

对这些向量进行聚类,可以定位在程序运行过程中开销较大的若干函数.

对于进(线)程向量的聚类,首先需要把程序中需要跟踪的函数进行编号,范围为[1,n],n 为函数的数量,然后可以将参与聚类的性能数据描述为一个向量: $d_i=(x_1,x_2,...,x_n)$.

其中,d_i 表示第 i 个进程的性能数据,i 的取值范围为[1,p],p 为进程数量, $x_1,x_2,...,x_n$ 表示第 i 个进程中各个函数的时间开销.因此进行进程性能数据为 $D_p=\{d_i \mid i\in[1,p]\}$.

同理对于函数向量,性能数据可以描述为 $d'_j=(y_1,y_2,...,y_p)$.

其中 d'_j 表示第 j 个函数的性能数据,j 的取值范围为[1,n], $y_1,y_2,...,y_p$ 表示第 j 个函数在各个进程中的时间开销,函数性能数据集可以表示为 $D_n=\{d'_j \mid j\in[1,n]\}$.

由于超级计算机上的大规模应用的规模和复杂性较大,本文采用迭代多次聚类进行分析,在每次聚类中,需要将性能数据分为 2 个簇,然后再从其中选择开销较大的簇进行再次聚类,通过多次迭代定位影响性能的函数和进程.这样可以从海量数据中定位性能较差的若干进(线)程或者函数,便于通过可视化继续分析.

整个分析过程可以简单描述为:

1) 将性能数据进行格式化,生成两个数据集 D_p 和 D_n;
2) 对 D_p 进行预处理,然后执行聚类分析,分为两个簇,A 和 B;
3) 对 D_n 进行预处理,然后执行聚类分析,分为两个簇,C 和 D;
4) 如果 A 和 B 中有属性差异明显,则确定其中开销较大的簇,对其执行 2);或者执行 6);
5) 如果 C 和 D 中有属性差异明显,则确定其中开销较大的簇,对其执行 3);或者执行 6);
6) 结束分析过程.

其中,4)和 5)中所指属性差异明显,将其定义为,对于某一属性,各簇中心点开销相差达到百分之五,且簇内数据点分布不收敛,收敛的条件是各簇内 80%数据点的该属性值相互之间的差小于百分之五.

2.3 评价方法

贝叶斯信息准则[10],简称 BIC 评分,是在大样本前提下对边缘似然函数的一种近似.SimPoint 使用这种评价标准,用于在 k-means 算法中选择 k 的值,其中的 BIC 评分法对于同一组数据的不同聚类算法都给出一个分数,然后通过比较 BIC 分数的大小来帮助确定 k 值的大小.

本文提出的基于聚类分析的性能分析方法,也使用 BIC 评分法对聚类结果进行评价,每次只将性能数据分为两个簇,BIC 评分用于确定每次聚类分析是否有效,聚类结果是否可靠,以及是否可以再次执行迭代聚类.

BIC 评分的第 1 项是模型的取对数的极大似然度估计,它的度量是模型与数据的拟合程度,第 2 项是一个关于模型复杂度的罚项.目标是选择一个 k 较小的模型.

$$BIC(M_j)=\hat{l}_j(D)-\frac{p_j}{2}\cdot\log R,$$

其中 $R=|D|$, $\hat{l}_j(D)$ 为数据 D 对于模型 M_j 的极大似然估计,p_j 是模型 M_j 中参数的个数,即数据属性的个数.方差的极大似然估计和节点的概率分别为

$$\hat{\sigma}^2=\frac{1}{R-K}\sum(x_i-\mu_{(i)})^2,\ \hat{P}(x_i)=\frac{R_{(i)}}{R}\cdot\frac{1}{\sqrt{2\pi}\hat{\sigma}^M}\exp\left(-\frac{1}{2\hat{\sigma}^2}\left\|x_i-\mu_{(i)}\right\|^2\right).$$

数据集的极大似然估计为

$$l(D)=\log\prod_i P(x_i)=\sum_i\left(\log\frac{1}{\sqrt{2\pi}\sigma^M}-\frac{1}{2\sigma^2}\left\|x_i-\mu_{(i)}\right\|^2+\log\frac{R_{(i)}}{R}\right).$$

对于方法本身的时间复杂度,若采用的聚类算法的时间复杂度为 $O(n)$,则在理想情况下,将数据集分成两个簇,对其中一个簇进行迭代聚类,其时间复杂度为 $O(n)+O(n/2)$,因此,进行 k 次迭代后,其中 k 为 2 的完成平方数,因此,整个分析过程的时间复杂度可表示为 $T(n)\approx O(2n)$.

同理,对于时间复杂度为 $O(n^2)$ 的聚类算法来说,进行 k 次迭代之后,时间复杂度为 $T(n)\approx O(1.75n^2)$,由此可见,本文所提方法并没有明显增加时间复杂度.

3 实验分析

本文中的实验采用 *K*-means 算法进行聚类分析,使用开源工具 TAU 完成性能数据的收集,并由开源工具 Weka 完成聚类分析工作,对一个简单并行 MPI 应用例子使用本文提出的性能分析方法,以检验方法的有效性,需要先对并行应用的性能数据进行处理,分别生成进程性能数据和函数性能数据.以北京航空航天大学计算机学院高性能集群为运算环境,该集群使用 CentOS 5.3 操作系统,4T 存储.该并行程序包含 5 个主要的函数,运行在 29 个节点上的 200 个进程中,每个进程生成一个性能日志,日志大小共约 920M.

在整个实验分析过程中,将初始进程性能数据集标识为 D,对其进行聚类后,得到的结果,对于簇 1 将其标识为 D_1,簇 2 标识为 D_2.以此类推,聚类后的结果中,各个簇的标识为所在数据集标识再加簇编号.

完成函数性能数据的收集和处理后,对数据进行预处理,再使用 Weka 进行聚类,结果如图 1 所示.从图 1 可以看到,对函数进行聚类分析,可以将其分为两个簇,这两个簇的时间开销差异较大,可以确定左侧簇 1 的两个函数 *A* 和 *B* 所占时间开销较大,对性能的影响最大.本例中函数数量较小,如果数据较大,可以簇 1 进行迭代聚类,进一步查找开销较大的函数.图 1 中横轴表示函数编号,纵轴表示开销,单位为微秒.

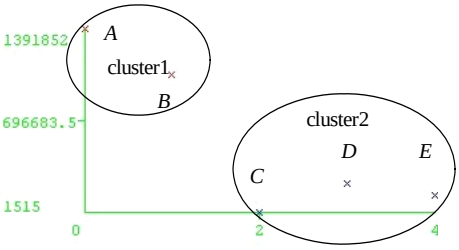

Fig.1 Clustering result for functions

图 1 函数性能数据聚类结果图

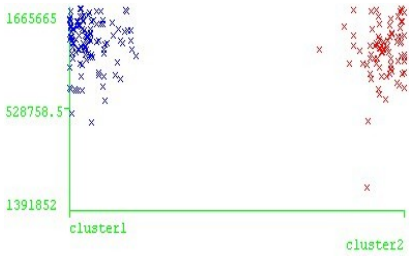

Fig.2 Overhead distribution of function *A*

图 2 各簇中函数 *A* 的开销分布

在完成对进程性能数据收集和预处理后,对其进行聚类分析,结合图 1 的分析结果,主要关注各个进程中开销较大的函数 *A* 和 *B*.进程性能数据集 *D* 中函数 *A* 的分布结果如图 2 所示,图 2 中横轴表示簇的编号,纵轴表示开销,单位为微秒.对性能数据集 *D* 聚类分析后各簇中函数 *A* 和 *B* 的开销分布如图 2 和图 3 所示.可以看出,两簇中函数 *A* 的开销差异差不大,而 D_1 中函数 *B* 的开销比 D_2 中的要大些,因此,可以继续对数据集 D_2 进行聚类.

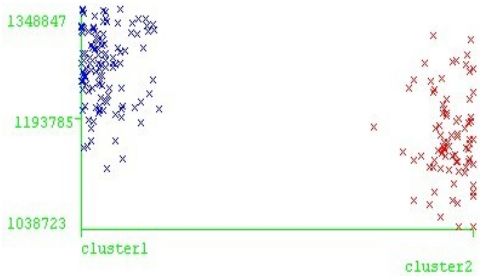

Fig.3 Overhead distribution of function *B*

图 3 各簇中函数 *B* 的开销分布

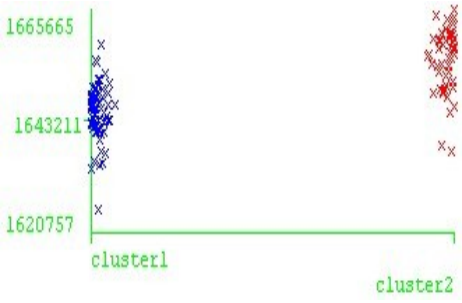

Fig.4 Overhead distribution of function *A* for D_2

图 4 对 D_2 聚类后函数 *A* 的开销分布

对数据集 D_2 进行迭代聚类,由图 4 和图 5 可以看出,在 D_2 中 D_{22} 的开销较大,而且 D_{21} 和 D_{22} 中函数 *A* 的开销差距变小,函数 *B* 的开销占据了主导,同时 D_{21} 和 D_{22} 都达到了收敛.

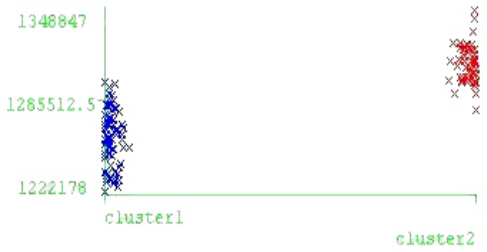

Fig.5 Overhead distribution of function *B* for D_2

图 5 对 D_2 聚类后函数 *B* 的开销分布

通过对函数性能数据的聚类,可以定位出影响程序开销的函数为 *A* 和 *B*,通过进程数据进行聚类,确定函数 *A* 在两个簇中的中心点开销差异较小,而函数 *B* 在两个簇的中心点开销差异较大,因此程序的开销差异主要是由函数 *B* 的开销所决定.迭代收敛后,可以确定对程序开销影响较大的进程.因此,可以看出函数 *A* 的效率比较合理,但开销较大,函数 *B* 需要进行优化,提高并行效率,使其任务划分更加合理,对于开销较大的进程,需根据其所在运行环境,确定问题来源.从函数性能数据和进程性能数据两个角度结合起来分析,可以比较快速的定位影响性能的因素,更加深入的区分影响性能的主要和次要因素,了解各因素影响程序性能的程度和相互关系.

4 结束语

综上所述,本文提出的一种灵活的聚类分析方法能够更好的适用于并行应用性能分析的情况,并且取得了良好的实验效果.使用迭代聚类的分析方法,可以有效的得出并行应用运行过程中时间开销大和开销较小的函数,也可以对程序运行中不同进程的各个函数的开销,结果均表现为两个簇.将两种情况结合可以缩小定位性能问题的范围,然后可以继续对开销过大的簇进行迭代聚类分析,进一步分析其开销来源.但聚类分析结果的可解释性较差,如何更好的解释通过聚类分析解释并行程序的性能问题,这是下一步的研究方向.

致谢 在此,我们向对本文的工作给予支持的北京航空航天大学网络技术北京市重点实验室的老师和同学表示感谢.

References:

[1] Sherwood T, Perelman E, Calder B. Basic block distribution analysis to find periodic behavior and simulation points in applications. In: Proc. of the Int'l Conf. on Parallel Architecture and Compilation Techniques (PACT). 2001. 3−14. [doi: 10.1109/PACT.2001.953283]

[2] Perelman E, Hamerly G, Van Biesbrouck M, *et al*. Using SimPoint for accurate and efficient simulation. ACM SIGMETRICS Performance Evaluation Review, 2003,31(1):318−319. [doi: 10.1145/781027.781076]

[3] Hamerly G, Perelman E, Lau J, *et al*. Simpoint 3.0: Faster and more flexible program phase analysis. Journal of Instruction Level Parallelism, 2005,7(4):1−28. [doi: 10.1.1.113.888]

[4] MacQueen J. Some methods for classification and analysis of multivariate observations. California, 1967. 14.

[5] Malony AD, Shende S, Morris A, *et al*. Evolution of a parallel performance system. Tools for High Performance Computing, 2008. 169-190. [doi: 10.1007/978-3-540-68564-7_11]

[6] Shende SS, Malony AD. The TAU parallel performance system. Int'l Journal of High Performance Computing Applications, 2006, 20(2):287−311. [doi: 10.1177/1094342006064482]

[7] Huck KA, Malony AD. PerfExplorer: A performance data mining framework for large-scale parallel computing, In: Proc. of the 2005 ACM/IEEE Conf. on Supercomputing (SC 2005). 2005. 41−52. [doi: 10.1109/SC.2005.55]

[8] Huck KA, Malony AD, Shende S, Morris A. Knowledge support and automation for performance analysis with PerfExplorer 2.0. Scientific Programming, 2008,16(2-3):123−134. [doi: 10.3233/SPR-2008-0254]

[9] Jolliffe IT. Principal Component Analysis. 2nd ed., New York: Springer-Verlag, 2002.

[10] Schwarz G. Estimating the dimension of a model. The Annals of Statistics, 1978,6(2):461−464.

朱鹏(1985－),男,山西霍州人,硕士生,主要研究领域为高性能计算,性能分析.

李云春(1972－),男,博士,副教授,主要研究领域为自主计算,高性能计算.

李巍(1970－),女,博士,副教授,主要研究领域为网络测量,性能评价.