面向云环境的集群资源模糊聚类划分算法的优化

董世龙 陈宁江 谭 瑛 何子龙 朱莉蓉
(广西大学计算机与电子信息学院 南宁 530004)

摘 要 传统的串行模糊聚类分析算法在应对高维矩阵运算时存在运算量大、运算效率低等问题,难以满足云环境中集群资源调度的时效性要求。为此,在基于等价关系的模糊聚类算法基础上对传递闭包法进行优化,提出一种基于多线程的云资源模糊聚类划分并发算法,并将其应用于Hadoop调度器的策略改进。仿真实验结果表明,优化策略有助于减少平方法求解模糊等价矩阵的计算量,所设计的并发算法能够有效解决中小规模云集群资源聚类的运算瓶颈问题,且具有较好的加速比。为了解决现有Hadoop调度器存在的异构性问题,对该优化并发算法进行了理论分析,结果表明它有助于解决异构性带来的调度难题。

关键词 模糊聚类,云计算,资源聚类,模糊等价矩阵,Hadoop
中图法分类号 TP311 **文献标识码** A **DOI** 10.11896/j.issn.1002-137X.2014.09.020

Optimization of Cluster Resource Fuzzy Clustering Partition Algorithm for Cloud Computing

DONG Shi-long CHEN Ning-jiang TAN Ying HE Zi-long ZHU Li-rong
(School of Computer and Electron Information, Guangxi University, Nanning 530004, China)

Abstract The classic fuzzy clustering serial algorithm has the problems of heavy computation and low efficiency in dealing with high dimensional matrix operations, so it can't be applied effectively into the fuzzy clustering partition model in cloud computing environment, and it's hard to meet the time efficiency requirement of resource scheduling. Therefore, a transitive closure method was optimized based on the equivalence relation-based fuzzy clustering algorithm. What's more, a fuzzy clustering concurrent algorithm based on multi-threading for cloud resources was applied to the improvement strategies for Hadoop scheduler. The experimental results indicate that the optimization strategy can reduce the computation for solving square-based fuzzy equivalent matrix problem. Moreover, the concurrent algorithm can effectively solve the computation bottleneck of resource clustering on small and medium-sized clusters, and it has a better speed-up ratio. To solve the problem of heterogeneity which exists in the existing Hadoop schedulers, theoretical analyses of the concurrent optimization algorithm show that it can help to solve scheduling problems caused by heterogeneity.

Keywords Fuzzy clustering, Cloud computing, Resources clustering, Fuzzy equivalence matrix, Hadoop

1 引言

随着云计算环境中集群规模不断壮大,如何有效合理地分配资源、调度任务,已成为影响云计算应用效率的重点与难点,直接影响着云平台的整体性能和用户的使用满意度[1,2]。而Hadoop[3]作为MapReduce[4]计算模型的一个开源实现,已成为一个可靠性好、扩展性强的分布式计算和存储平台,被越来越多研究机构和公司用于数据挖掘、日志分析、广告计算和科学实验等。

Hadoop作业任务调度算法的好坏直接影响MapReduce分布式计算框架的性能优劣,而其自带的3种作业调度算法(FIFO调度算法FIFO Scheduler、计算能力调度算法Capacity Scheduler和公平份额调度算法Fair Scheduler)都是建立在同构集群的前提下,即将集群中各节点的性能看成是一样的[5]。随着集群规模的增大,新老硬件的交替、集群各节点的异构性将会大大增加,各节点CPU、内存、网络、硬盘等资源情况差异增大;相同配置的节点上运行云服务的负载不同也会导致集群实际状态的异构性。在使用Hadoop对作业进行处理时,也需考虑用户对不同资源的需求偏好,像"文本搜索"这类作业,主要受集群网络带宽和各个节点硬盘I/O速度的限制,而像"日志分析"这类作业,主要依靠CPU的处理能力。

云计算环境下集群节点存在着大量的异构性、不确定性和模糊性,很难准确地描述集群中的节点资源与作业任务,这使得分配节点资源去完成不同类型作业任务存在复杂性。需要考虑如何有效地发现和区分这些不同主机节点间的相似性,对其进行聚类或者分组并按性能高低进行组间、组内排

到稿日期:2013-11-24 返修日期:2014-04-21 本文受国家自然科学基金(61063012,61363003),广西自然科学基金项目(2012GXNSFAA053222),广西高校优秀人才资助计划([2011]40),广西科学研究与技术开发计划项目(桂科攻1348020-7,桂科软13180015)资助。
董世龙(1988—),男,硕士,主要研究方向为网络分布式计算、软件工程,E-mail:wodedsl@163.com;陈宁江(1975—),男,博士,教授,主要研究方向为网络分布式计算、软件工程。

序，使同一类中的节点具有较高的相似度或相关度，而不同类中的节点之间差别较大，从而满足不同作业任务的资源需求。

聚类分析方法被大量应用于决策支持、数据挖掘、模式识别、机器学习等领域[6-8]。目前，将基于等价关系的模糊聚类分析算法[9]应用于云计算环境下的云资源分配、云任务调度等问题，借助模糊理论来构建云计算的资源划分模型、任务调度模型是业界研究的热点[10,11]。对云集群节点使用模糊聚类划分算法得到的逻辑子群，具有群内性能相近、群间性能差异较大的性质，可将每个子群看成是独立的“同构”小集群，将整个集群看成是由许多个“同构”小集群组成的“异构”大集群。因此，本文将模糊聚类划分算法引入到 Hadoop 调度器中，将聚类分类结果作为调度的决策依据，期望能够解决异构环境下 Hadoop 调度效率不高等问题。

将第三方算法引入到云资源、云任务调度模型，首要关注的问题则是获得调度决策依据的过程是否高效，换句话说，得到决策的耗时是否足够短，如果用时过长，再去调度任务执行毫无意义。云场景下的集群节点规模在不断增加，节点数量已经成千上万，而传统串行的模糊聚类分析算法在应对高维矩阵运算时存在运算量大、运算效率低等问题，无法有效应用到云计算下大规模集群资源模糊聚类划分、任务调度模型中，聚类耗时难以满足云资源、云任务调度的时效性要求。因此，有必要探讨面向云环境的集群资源模糊聚类划分算法的优化策略。

2 相关工作

在 Hadoop 调度算法的改进方面，文献[12]提出基于优先权的自适应调度算法，根据当前系统各节点的负载水平，通过动态调整作业执行队列长度，利用优先级为不同的作业分配或多或少的系统资源和执行时间来实现不同类型作业的差别服务。文献[13]提出了一种基于朴素贝叶斯分类的作业调度算法，用朴素贝叶斯分类器将作业分为好作业（不会使执行节点过载的作业）和坏作业（导致执行节点过载的作业），优先调度资源去执行好作业，通过不断学习每次任务分配决策对资源的反馈结果来改善作业分类。

在模糊理论应用于资源划分与任务调度方面，文献[10]对资源进行性能模糊聚类，根据任务参数计算资源偏好，使不同偏好任务在不同聚类中进行选择，缩小了选择范围，更好地反映了任务需求。文献[11]着眼于大规模、分布式、自治、异构、动态的网格计算环境，提出了一种能够描述用户应用程序喜好差异的启发式网格模糊聚类资源分配算法，以实现合理的预分类资源，根据用户喜好进行选择，最大化用户的目标效用，同时避免不同任务分配集中在少数资源，使网格环境的负载平衡得以改善。文献[14]针对 DAG 图提出了基于模糊聚类的网格任务调度算法。文献[15]为了提高复杂软硬件系统的资源利用率和快速响应实时任务，提出了一种基于模糊等价矩阵的资源聚类调度算法 FRCA（Fuzzy Resource Clustering Algorithm），其将资源划分为 3 种任务类型：读/写存储器任务、CPU 计算任务和 I/O 任务。文献[16]提出了基于信任与 QoS 需求聚类分析的可定制云工作流调度算法，它在单一工作流调度过程中，使用模糊聚类方法根据不同工作流的 QoS 需求参数对工作流程进行分类（时间敏感、成本敏感和平衡 3 种类型），能够很好地分析云计算客户的服务偏好，从而为不同类型的服务目标定制不同的服务策略。文献[17]提出一种新型的基于资源和任务混合聚类的网格资源分配算法，它巧妙地分配合适的资源以完全满足任务之需，并且有效预见当前和未来任务的资源需求，避免资源的滥用或分配不合理。文中对传递闭包法求模糊等价矩阵的效率低、时间复杂度过高的问题，采用了编网法直接对模糊相似矩阵进行逻辑子群划分，以提高运算效率。文献[18]试图从集群聚类的角度解决大规模集群资源的分配与调度问题，把集群中性能各异、资源状况配置不同的计算主机，按能力聚类成组，形成若干个性能均衡的逻辑子群，群内的计算资源配置相近，可支持某些有特殊资源需求的计算任务。文献[19]提出一种集群资源模糊聚类划分模型，对集群计算节点的 CPU、内存、网络、I/O 和网卡资源参数进行量化和规范化，运用模糊聚类技术实现计算节点的聚类划分。

上述工作在使用模糊聚类分析算法过程中，在求解模糊等价矩阵时，使用较多的方法是传递闭包法，时间复杂度为 $O(n^3)$，除此之外，聚类划分方法还有直接聚类法（最大树法、编网法）、图论法等。然而在主机数量较多的云集群下，要对成千上万的节点资源数据进行聚类，就会出现高维矩阵运算，传统串行的处理方式将会遇到运算瓶颈。基于 MPI、GPU 技术的聚类算法并行化[20]需要控制整个集群间的集群通信，实现方式过于复杂，无法将开发者从并行实现的细节中解放出来。而基于 Hadoop 平台的其他聚类算法如 K-均值[21]、K-近邻[22]均已在 MapReduce 上进行了实现，使得开发人员不需过多了解并行化的具体通信过程，就可实现聚类算法的高效并行化，能够具备应对海量数据快速挖掘的能力。但 MapReduce 在处理小规模数据时，在迭代过程中的等待耗时、作业交互耗时会导致运算时间与效率略低于传统的串行算法[23]。而数千数万节点的资源数据文件还只是 kB 或 Mb 级别，使用规模较小的“同构”Hadoop 集群进行分布式模糊聚类运算并不适合。

因此，本文将主要针对中小规模云集群的资源数据聚类划分，通过对传递闭包法求解模糊等价矩阵过程进行分析，发现存在能够减少运算量的方法，同时由于模糊相似矩阵、模糊等价矩阵单个元素的计算是独立的，可将这些元素的计算改成并发的方式进行运算。在基于等价关系的模糊聚类算法基础上对平方法进行优化，提出一种基于多线程的云资源模糊聚类划分并发算法，以降低为中小规模云集群提供调度依据所花费的聚类耗时开销，并将其应用于 Hadoop 调度器的改进策略研究中，以验证该优化并发算法的可行性。

3 资源聚类划分模型及其优化策略

3.1 划分模型

根据模糊理论及文献[10,15,18]对资源划分模型的定义，本文将云资源聚类划分模型总结为如下 7 个步骤。

(1)定义资源特征

设集群节点集合 $N=\{N_1,N_2,\cdots,N_n\}$，集群节点数量 $|N|=n$。使用 CPU、内存、网络带宽、I/O 和网卡这 5 个资源特征来刻画集群中的节点，资源特征集合定义为 $R=\{r_0,r_1,r_2,r_3,r_4\}$，资源特征数 $m=5$。资源种类、资源维度可根据不同场景进行调整，可对资源权重、资源需求意向进行规约。

对于集群 N 中的每个节点 N_k 都有一个资源特征矢量 $R(N_k)=(r_{k0},r_{k1},r_{k2},r_{k3},r_{k4})$，$r_{kj}$ 是第 k 个节点的第 j 维特征指标，则集群的 $n\times m$ 原始数据矩阵为 $(r_{ij})_{n\times m}$。

(2)数据标准化

采用平移标准差变换法对资源特征进行标准化：

$r_{kj}'=(r_{kj}-\bar{r}_j)/S_j, k=1,2,\cdots,n;\ j=1,2,\cdots,m$

其中，$\bar{r}_j=\frac{1}{n}\sum_{k=1}^{n}r_{kj}$ 为第 j 维特征资源的均值，$S_j=\sqrt{\frac{1}{n}\sum_{k=1}^{n}(r_{kj}-\bar{r}_j)^2}$ 为第 j 维特征资源的标准差。

(3)数据归一化

采用平移极差变换法将 r'_{kj} 归一化到[0,1]区间，得

$r''_{kj}=(r'_{kj}-r'_{j\min})/(r'_{j\max}-r'_{j\min})$

其中，$r'_{j\min}=\min(r'_{1j},r'_{2j},\cdots,r'_{hj})$，$r'_{j\max}=\max(r'_{1j},r'_{2j},\cdots,r'_{nj})$，$(k=1,2,\cdots,n;\ j=1,2,\cdots,m)$。

(4)建立模糊相似矩阵

由于原始数据矩阵中不同的列来自不同的主机节点，故可采用指数相似系数法来计算节点 N_i 和 N_j 之间的相似程度 $p_{ij}=P(N_i,N_j)\in[0,1]$，其中 $p_{ij}=\frac{1}{m}\sum_{k=1}^{m}e^{-\frac{3}{4}*\frac{(r''_{ik}-r''_{jk})^2}{S''^2_k}}$，$p_{ii}=1, 0\leqslant p_{ij}=p_{ji}\leqslant 1$，$S''^2_k=\frac{1}{n}\sum_{k=1}^{n}(r_{kj}''-\overline{r''_j})^2$，$\overline{r''_j}=\frac{1}{n}\sum_{k=1}^{n}r''_{kj}$，$1\leqslant i,j\leqslant n$，得到集群 N 的模糊相似矩阵 $P=\begin{bmatrix}p_{11}&p_{12}&\cdots&p_{1n}\\p_{21}&p_{22}&\cdots&p_{2n}\\\cdots&\cdots&\cdots&\cdots\\p_{n1}&p_{n2}&\cdots&p_{nn}\end{bmatrix}$，$P$ 具有对称性和自反性。

(5)建立模糊等价矩阵

传递闭包法也称平方法，通过求传递闭包，可将模糊相似矩阵转变为具有传递性的模糊等价矩阵。

从模糊相似矩阵 P 出发，使用平方法进行合成运算 $p_{ij}^*=\vee\{p_{ik}\wedge p_{jk},1\leqslant k\leqslant n\}=\mathrm{Max}\{\mathrm{Min}(p_{ik},p_{jk}),1\leqslant k\leqslant n\}$，$P^2\rightarrow P^4\rightarrow\cdots\rightarrow P^{2c}$，当第一次出现 $P^*\times P^*=P^*$ 时(表明 P^* 具有传递性)，得 P 的传递闭包 $P^*=\begin{bmatrix}p_{11}^*&p_{12}^*&\cdots&p_{1n}^*\\p_{21}^*&p_{22}^*&\cdots&p_{2n}^*\\\cdots&\cdots&\cdots&\cdots\\p_{n1}^*&p_{n2}^*&\cdots&p_{nn}^*\end{bmatrix}$，$p_{ii}^*=1, 0\leqslant p_{ij}^*=p_{ji}^*\leqslant 1, 1\leqslant i,j\leqslant n$，迭代次数 $c\leqslant[\log_2 n]+1$，P^* 具有对称性、自反性和传递性。

(6)聚类划分

设定不同的划分阈值 $\alpha\in[0,1]$，可得到不同的分类，从而形成一种动态聚类图。α 值越接近 1，表明聚类之间的相似程度越高；α 值越接近 0，则表明聚类之间的相似程度越低。

采用 F 统计量法[9]确定 α 的最佳值，得到聚类矩阵 $T=P^*-M_a$，其中 M_a 为全 α 的 $n\times n$ 矩阵，则矩阵 T 中大于 0 的元素下标所对应的节点即为同一个逻辑子群。

(7)聚类子群的性能建模

①全局性能排序

在文献[14]中给出的聚类逻辑子群全局性能排序公式的基础上，加入资源需求权重 $W=\{w_0,w_1,w_2,w_3,w_4\}$，以满足对各个资源特征的不同偏爱程度，则第 i 个分类 $Cluster_i$(简记为 CL_i)的综合性能为

$$PERF(CL_i)=\frac{1}{n_i}\sum_{N_k\in CL_i}\sum_{j=1}^{m}w_j\cdot r''_{kj}$$

w_j 为对第 j 维特征资源的需求权重，则可将经过聚类划分得到的子群类目，根据用户对资源需求度的不同，按其综合性能高低进行排序，得到新的划分分类。

②局部性能排序

在文献[14]给出的全局性能排序公式的基础上，本文给出局部性能排序公式，第 i 个分类 CL_i 中节点 N_k 的个体性能为

$$PERF(N_k)=\sum_{j=1}^{m}w_j\cdot r''_{kj}$$

按全局综合性能和局部个体性能排序后，又可得到新的划分结果。

3.2 Hadoop 调度器改进策略研究

“异构”大集群经过模糊聚类运算划分成多个“同构”小集群，这是模糊聚类算法应用于 Hadoop 调度器改进策略的理论依据。改进策略的原则是在原有 3 大调度器基础上，在获取集群节点待分配列表时，从原来只有一个待分配“异构”群体变成经过模糊聚类划分并按照性能高低排序得到的多个待分配“同构”子群，然后根据 3 种调度算法自带的选举执行节点规则在这些子群里进行选择，余下的调度步骤维持不变。

对 FIFO 调度器而言，改进的思路是对全局性能排序和局部性能排序后得到的集群节点序列，优先将性能优异的节点依次分配给待执行作业和任务，避免 FIFO 调度算法选择节点过程出现的节点随机性问题，最大限度地利用集群资源，以提升用户满意度。

计算能力调度算法采用多个队列，每个队列分配一定的系统容量，改进的思路是将聚类划分后的每个子群作为一个队列，通过全局性能排序，形成适合不同需求的各类型队列，如 cpu-intensive、io-intensive 等类型。用这些“同构”小集群去完成各种不同需求的作业任务，群内主机节点能力均衡则有利于负载均衡，将大大缓解各节点完成任务时间不一致现象，避免出现先天性的“拖后腿”节点，从而提高“同构”小集群处理作业的吞吐量，合理充分利用集群资源。而这些“同构”小集群的整体性能档次高低有别，则可解决作业优先级、用户特权等问题。

公平份额调度算法在异构环境下把每个节点均视为相同的个体，导致即使作业获得的执行节点数量与其他作业相同，也会出现由于实际个体性能各异导致整体性能并不相同，无法解决所谓“公平”份额的问题。改进的思路是按照聚类划分后每个子群数量的百分比形式进行组合，将部分高性能节点和部分低性能节点进行搭配，发挥局部性能优势，解决具有公平性要求的调度问题，确保每个用户或每个作业得到的资源性能总和是“近似”相等的，但这时则无法保证各节点完成任务时间的一致性。

文献[13]是从用户作业角度出发，根据作业的过载情况使用朴素贝叶斯分类器将作业区分为好节点和坏节点，而判断的标准则是根据用户的具体需求来制定，挑选出能够满足作业要求的执行节点进行分配。而被归为坏作业的用户使用满意度将会受到影响，本文则从提供服务的集群节点内聚性进行模糊划分，最大限度地去满足用户的需求。

3.3 聚类划分算法的优化

3.3.1 优化策略

在 3.1 小节步骤(5)中，P 和 P^* 都是对角阵，且主对角线上的元素全为 1，而计算 p_{ij}^* 需先对 p_{ik}，p_{jk} $(1\leqslant k\leqslant n)$ 这两行对应位置的元素两两取小，得到一个长度为 n 的行矩阵，p_{ij}^*

则等于行矩阵中元素的最大值，先取小后取大求解单个 p_{ij}^* 所需的运算量为 $G=n+n-1=2n-1$。

在矩阵 P 中存在两个特殊的元素 $p_{ik}(k=j)$ 即 p_{ij} 及 $(p_{jk}, k=i)$ 即 p_{ji}，当 $k=j$ 时，$\text{Min}(p_{ik}, p_{jk})=\text{Min}(p_{ij}, p_{jj})=p_{ij}$；当 $k=i$ 时，$\text{Min}(p_{ik}, p_{jk})=\text{Min}(p_{ii}, p_{ji})=p_{ji}=p_{ij}$。

采取的优化策略描述如下：

(1)令 $\max=p_{ij}$，将 max 与 $p_{jk}(1\leqslant k\leqslant n$ 且 $k\neq i,j)$ 逐一比较，每次比较之后产生新一轮的 max，其中可能需要再次与 p_{jk} 对应位置的 p_{ik} 进行比较，这将会出现以下两种情形。

情形 1　若 $\max\geqslant p_{jk}$，存在以下两种情况：

①当 $p_{ik}\geqslant p_{jk}$ 时，则 $\text{Min}(p_{ik}, p_{jk})=p_{jk}$，$\text{Max}(\max, p_{jk})=\max$；

②当 $p_{ik}<p_{jk}$ 时，则 $\text{Min}(p_{ik}, p_{jk})=p_{ik}$，$\text{Max}(\max, p_{ik})=\max$。

情形 2　若 $\max<p_{jk}$，存在以下两种情况：

①当 $p_{ik}\geqslant p_{jk}$ 时，则 $\text{Min}(p_{ik}, p_{jk})=p_{jk}$，$\text{Max}(\max, p_{jk})=p_{jk}$；

②当 $p_{ik}<p_{jk}$ 时，则 $\text{Min}(p_{ik}, p_{jk})=p_{ik}$，$\text{Max}(\max, p_{jk})=\text{Max}(\max, p_{ik})$。

(2)经过以上推导，得到新一轮的 max，表示为：

计算结果①：若 $\max\geqslant p_{jk}$，$\max=\max$；

计算结果②：若 $\max<p_{jk}\leqslant p_{ik}$，$\max=p_{jk}$；

计算结果③：若 $\max<p_{jk}$ 且 $p_{ik}<p_{jk}$，$\max=\text{Max}(\max, p_{ik})$。

(3)优化后求解单个 p_{ij}^* 的伪代码如下：

```
compute(p*ij):
BEGIN
1. max←pij
2. for k=1 to n(k≠i,j)
3.   if max<pjk
4.     if pjk≤pik
5.     max←pjk;
6.     endif
7.     else
8.       max←(max>pik? max:pik);
9.     endif
10.  endif
11. endfor
12. p*ij =max
END
```

3.3.2　运算量分析

当 $p_{jk}(1\leqslant k\leqslant n$ 且 $k\neq i,j)$ 全部出现计算结果①时，步骤(5)(即模糊等价矩阵计算)求解单个 p_{ij}^* 所需的运算量为 $G'=n-2=\frac{1}{2}(G-3)$，此时为最好情况，运算量减少了一半；

当 $p_{jk}(1\leqslant k\leqslant n$ 且 $k\neq i,j)$ 全部出现计算结果②或者计算结果③时，步骤(5)求解单个 p_{ij}^* 所需的运算量为 $G'=n-2+n-2=G-3$，此时为最坏情况。

p_{jk} 全部元素同时出现计算结果①、计算结果②或者计算结果③的概率非常小，因此计算单个 p_{ij}^* 所需运算量 G' 介于 $n-2$ 和 $2(n-2)$ 之间，即 $G'\in[\frac{1}{2}(G-3), G-3]$，则可推断用步骤(5)求解模糊等价矩阵的总运算量将得到降低，整体聚类运算耗时也会随之减少。

3.4　划分算法的并发优化

在模糊聚类运算中，最耗时的两个步骤是步骤(4)和(5)，需生成 $n\times n$ 规模的新矩阵，由于矩阵满足对称性且主对角线都为 1，只需计算矩阵上三角区域内的元素，需要计算的元素总量为 $f(n)=n(n-1)/2$，当集群规模较大时，串行计算 $f(n)$ 将会遇到瓶颈，并且步骤(5)需进行 c 次迭代运算，将无法满足中小规模云集群资源聚类的时效要求。在步骤(4)和(5)中，对每个 $n\times n$ 新矩阵各元素的计算是独立的，因此可将这些元素的计算改成多线程并发的方式进行执行，通过设置线程锁来确保每次并发操作后能再次形成完整的 $n\times n$ 新矩阵。相关说明如下：

①*initThreadNum* 为并发线程的数量，*finishedThreadNum* 为已完成线程数，*taskNum* 为需要计算的元素总个数，*lenPerThreadNum* 为单个线程需要计算的元素个数；则 $taskNum=n(n-1)/2$，$lenPerThreadNum=taskNum/initThreadNum$，$finishedThreadNum=0$。

为了充分利用 CPU 资源进行并发运算，设置并发线程数 *initThreadNum* 大于 CPU 核数，并通过在各核间平均分配计算任务量来达到负载均衡。

②*initThreadNum* 个线程并发做单个 p_{ij} 或 p_{ij}^* 运算，对 *finishedThreadNum* 设置同步锁，以确保每个线程完成运算后 *finishedThreadNum* 加 1；

③直到 *finishedThreadNum* 与 *initThreadNum* 相等，此时才完成一次 $n\times n$ 新矩阵的运算。

并发求解单个 p_{ij} 或 p_{ij}^* 的伪代码如下：

```
par_compute(pij or p*ij):
BEGIN
1. for ii=0 to initThreadNum par-do
2.   for index=0 to lenPerThreadNum
3.     compute(pij or p*ij);
4.     synchronized(finishedThreadNum){
5.       finishedThreadNum++;
6.     }
7.   endfor
8. endfor
9. while(finishedThreadNum! =initThreadNum){
10.   sleep();
11. }
END
```

4　仿真实验及分析

本文采用仿真实验的方法，参照小规模集群资源数据的特征，模拟生成了不同规模集群的资源数据集，节点个数为 500～4000，实验环境包括具有不同 CPU 核数(双核、四核、八核、十六核)的服务器主机，操作系统为 Cent-OS6.4，安装的 JDK 版本为 1.6.0-21，JVM 参数设置为-Xms256M -Xmx1024M -XX:PermSize=128M -XX:MaxPermSize=256M。

为了验证本文资源聚类划分优化算法的有效性，将文献[15,18]中的传统资源模糊聚类划分算法 FRCA 作为比较对象，将本文的优化策略应用于 FRCA(以下简称该情形为“优化 FRCA”)，并给出优化后的 FRCA 算法的并发实现(以下简称“并发 FRCA”)，对这 3 种实现进行对比实验。将不同规模集群的资源数据在不同 CPU 核数的服务器主机上，分别执行

传统 FRCA、优化 FRCA 及并发 FRCA 10 次，则平均执行时间即为最终的聚类运算耗时。

传统 FRCA、优化 FRCA 及并发 FRCA 的运算时长如图 1 所示，并发 FRCA 的聚类运算是在四核服务器主机上进行的。从图 1 中可以看出，随着集群规模的不断增大，传统的集群资源聚类耗时将会越来越大，在集群节点个数为 2000 时，聚类耗时已经超过 5 分钟，难以满足云资源调度的时效性要求。

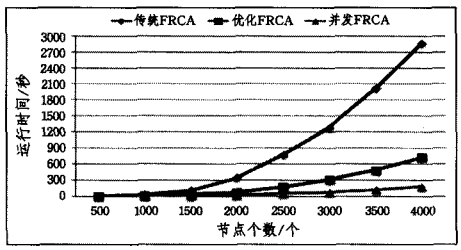

图 1　传统 FRCA、优化 FRCA、并发 FRCA 的聚类耗时

图 1 结果表明，优化后的 FRCA 聚类运算耗时得到了大幅降低，降幅达到 50%以上，原因在于对传递闭包法进行优化后，求解模糊等价矩阵的总运算量得到了减少，在迭代运算多次后，求解模糊等价矩阵整体耗时的减少将会影响整个聚类过程。如图 2 所示，随着集群规模的不断增大，传统 FRCA 聚类运算的模糊等价处理耗时几乎占据了整体聚类总耗时的 90%以上，直接决定着聚类总耗时的长短，因此在对传递闭包法求解模糊等价矩阵过程中，本文提出的优化策略是可行有效的。

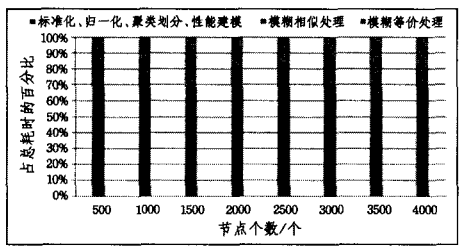

图 2　传统 FRCA 中不同步骤占总耗时的百分比

图 3 为优化后的 FRCA 在不同 CPU 核数的服务器主机上并发聚类的耗时。由于在传统 FRCA 聚类运算中，模糊相似处理和模糊等价处理占据着整体聚类总耗时的绝大部分，因此将这两个步骤求解单个矩阵元素进行线程并发是有必要的。随着处理器核数的不断增加，并发聚类耗时也相应减少，可满足不同规模集群资源聚类的时效要求，在集群节点个数为 4000 时，使用十六核服务器主机进行线程并发运算，聚类耗时接近 1 分钟。

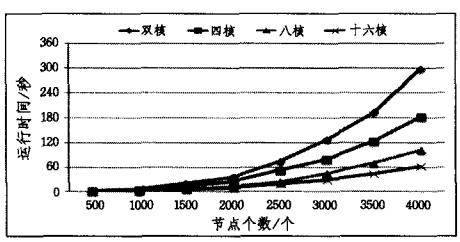

图 3　不同核数的并发聚类耗时

在优化后的 FRCA 基础上对其进行线程并发，加速比 $S=T_{串行}/T_{并行}$，$T_{串行}$ 是在单处理器下的串行执行时间，$T_{并行}$ 是在有 p 个处理器下的并发执行时间，当 $S=p$ 时，此加速比称为线性加速比。不同规模集群资源数据进行并发聚类的加速比如图 4 所示，随着处理器核数的不断增加，并发聚类算法的加速比也越来越大，其增幅也越来越大。随着集群规模的不断增大，加速比慢慢趋近于线性加速比，这表明线程并发后的优化 FRCA 具有较好的加速比。

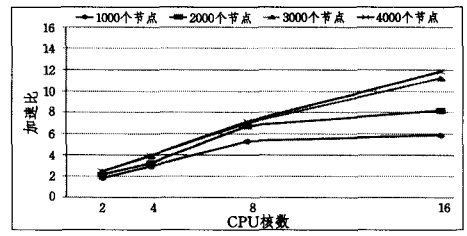

图 4　不同规模集群资源数据聚类的加速比

结束语　本文在传统模糊聚类划分算法的基础上，对传递闭包法求解模糊等价矩阵过程进行优化，能使整个模糊等价处理过程的总运算量得到减少，较之传统聚类算法，运算耗时减少超过一半，进行线程并发后的模糊聚类算法具有较好的加速比，能有效解决中小规模矩阵聚类运算存在的运算量偏大、运算效率偏低的问题，在中小规模集群环境下优化并发后的模糊聚类划分算法能够满足对云资源模糊聚类划分的时效要求。将优化并发后的模糊聚类划分算法应用到 Hadoop 自带的 3 大调度算法并进行了改进策略分析，这有助于解决调度器面临的异构性问题，下一步工作将在 Hadoop 源代码基础上进行改进策略的实现和实验。

随着集群规模的不断增大，主机节点数量扩大到数万，无论是传统的 FRCA 还是优化并发后的 FRCA，都无法应对超高维矩阵带来的运算量超大、运算空间严重不足等问题。因此，在今后的工作中需要在优化后的 FRCA 基础上，使用分布式计算框架 Hadoop 对大规模集群资源数据进行聚类运算，设想通过构造一个小规模的同构小型云来对监控到的异构大型云的资源数据进行并行模糊聚类运算，用同构小型云去调度异构大型云，以应对超高维矩阵聚类运算在时间复杂性和空间复杂性上遇到的瓶颈问题。

参 考 文 献

[1] Ji Chang-qing, Li Yu, Qiu Wen-ming, et al. Big data processing in cloud computing environments [C]// Proceedings of the 12th International Symposium on Pervasive Systems. Algorithms and Networks, 2012: 17-23

[2] Gao Zhong-wen, Zhang Kai. The research on cloud computing resource scheduling method based on time-cost-trust model [C]// Proceedings of the 2nd International Conference on Computer Science and Network Technology. 2012: 939-942

[3] Hadoop. Open-source software for reliable, scalable, di- stributed computing[EB/OL]. http://hadoop.apache.org, 2011

[4] Dean J, Ghemawat G. MapReduce: simplified data processing on large clusters[J]. Communications of the ACM 50th anniversary issue, 2008, 51(1): 107-113

[5] Zahafia M, Konwinski A, Joseph A, et al. Improving MapReduce Performance in Heterogeneous Environ- ments [C]// Proceedings of the 8th Usenix Symp on Operating Systems Design and Implementation. 2008: 29-42

[6] Raju G, Thomas B, Tobgay S, et al. Fuzzy clustering methods in data mining: A comparative case analysis [C]// Proceedings of

the 2008 International Conference on Advanced Computer Theory and Engineering. 2008:489-493

[7] Wang Zhongyuan, Qi Qing-wen, Xu Li. Cluster analysis based on spatial feature selecting in spatial data mining [C]// Proceedings of the 2008 International Conference on Computer Science and Software Engineering. 2008:386-389

[8] Yang Jiann-min, Wu Wen-chin, Liao Wei-cheng, et al. Trend analysis of machine learning—A text mining and document clustering methodology [C]// Proceedings of the 2009 International Conference on New Trends in Information and Service Science. 2009:481-486

[9] 梁保松,曹殿立. 模糊数学及其应用[M]. 北京:科学出版社,2007

[10] 李文娟,张启飞,平玲娣,等. 基于模糊聚类的云任务调度算法[J]. 通信学报,2012,33(3):146-154

[11] Sun Da-wei, Chang Gui-ran, Jin Li-zhong, et al. Optimizing grid resource allocation by combining fuzzy clustering with application preference [C]// Proceedings of the 2nd IEEE International Conference on Advanced Computer Control. 2010:22-27

[12] 陈艳金. MapReduce 模型在 Hadoop 平台下实现作业调度算法的研究和改进[D]. 广州:华南理工大学,2011

[13] 夏祎. Hadoop 平台下的作业调度算法研究与改进[D]. 广州:华南理工大学,2010

[14] 杜晓丽,蒋昌俊,徐国荣. 一种基于模糊聚类的网格 DAG 任务图调度算法[J]. 软件学报,2006,17(11):2277-2288

[15] Luo Mei, Zhang Kai-long, Yao Long-hui, et al. Research on resources scheduling technology based on fuzzy clustering analysis [C]// Proceedings of the 9th International Conference on Fuzzy Systems and Knowledge Discovery. 2012:152-155

[16] Li Wen-juan, Zhang Qi-fei, Wu Ji-yi, et al. Trust-based and QoS demand clustering analysis customizable cloud workflow scheduling strategies [C]// Proceedings of the 2012 IEEE International Conference on Cluster Computing Workshops. 2012:111-119

[17] Li Fu-fang, Qi De-yu. Research on grid resource alloca- tion algorithm based on fuzzy clustering [C]// Proceedings of the 2nd International Conference on Future Generation Communication and Networking. 2008:162-166

[18] 刘伯成,陈庆奎. 云计算中的集群资源模糊聚类划分模型[J]. 计算机科学,2011,38(10A):157-160

[19] 那丽春. 集群资源模糊聚类划分模型[J]. 计算机工程,2012,38(6):34-36

[20] Sejun K, Donald C. A GPU based parallel hierarchical fuzzy ART clustering [C]// Proceedings of the 2011 International Joint Conference on Neural Networks. 2011:2778-2782

[21] Zhao Wei-zhong, Ma Hui-fang, He Qing. Parallel kmeans clustering based on mapreduce [C]// Proceedings of the 1st International CloudCom Conference. 2009:674-679

[22] Aleksandar S, Sebastian M, Ralf S. RankReduce-proce- ssing k-nearest neighbor queries on top of mapreduce [C]// Proceedings of the 8th Workshop on Large-Scale Distributed Systems for Information Retrieval. 2010:13-18

[23] 江小平,李成华,向文,等. 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,2011,39:120-124

(上接第 103 页)

根据 3.3 节的仿真结果,由 Spartan-3AN 系列芯片实现并行 32 位 CRC 计算器,生成多项式 IEEE 802.3 CRC-32,一般需要 32 个寄存器和 238 个四输入的查找表实现,而并行 64 位的 CRC 计算器则需要 32 个寄存器和 378 个四输入的查找表实现。考虑到查找表的利用率,并行 64 位的 CRC 计算器所需要的资源约为 32 位计算器的 2 倍,同样,随着通道数量的增加,所需的硬件资源成倍增加。这种情况下,若直接采用单通道传统的计算方法,为了满足 10Gbps 的速度,32 位的并行计算器的工作频率为 312.5M,64 位的计算器的工作频率为 156.25M。由表 1 可以看出,Q 值取最佳值,当通道数量较小时,随着通道数 M 的成倍增加,计算频率成倍减小,当通道数较多时,计算器工作频率的减小趋势减弱,因此,在实际使用过程中,当硬件资源有限时,可以采用较少的通道数量获得较小的计算频率来满足较大的计算带宽,若硬件资源足够,也可以设置足够的通道数量以及单通道并行计算的位数来实现所需的工作频率。因此嵌套 CRC 码可以很好地用于 10Gbps 通信网络,通过取合适的 M、Q、W 的值,同样可以将嵌套 CRC 码应用于 40Gbps 甚至 100Gbps 的通信网络。

结束语 本文提出了一种嵌套 CRC 码来实现数据的差错控制,通过 Spartan-3AN FPGA 芯片实现了嵌套 CRC 码计算器。通过对嵌套 CRC 码的计算性能和差错控制能力的分析,讨论了嵌套 CRC 码中通道数量、嵌套次数以及通道并行处理位数对计算性能的影响,并给出了取值的依据。该方法有望为日益发展的聚合网络和数据中心提供高速数据差错控制途径。

参 考 文 献

[1] 路渭华. 下一代以太网发展趋势[J]. 光通信技术,2007,12:7-9

[2] Gai S, DeSanti C. 思科数据中心 I/O 整合[M]. 陈柳,译. 北京:邮电出版社,2013:2-12

[3] 彭建辉. 10G 以太网接口并行 CRC 校验的一种简化算法[J]. 微计算机信息,2006,20:213-215

[4] Renuka H K, Jayashree C N. Design and Computation of Cyclic Redundancy Code for Ethernet Application: an Implementation Using FPGA [J]. World Journal of Science and Technology, 2011,1(8):68-73

[5] 杨梅娟,尹德春. CRC 算法的研究[J]. 计算机与数字工程,2005,34:30-32

[6] Kounavis M E, Berry F L. Novel Table Lookup-Based Algorithms for High-Performance CRC Generation[J]. IEEE Trans. on Computer Society, 2008,57(11):1550-1560

[7] 梁海华,盘丽娜,赵秀兰,等. CRC 查询表及其并行矩阵生成方法[J]. 计算机科学,2012,39(B06):154-158

[8] 岳天天. 一种并行 CRC 校验算法的 IP 设计与实现[J]. 广东通信技术,2013(3):78-79

[9] Sprachmann M. Automatic Generation of Parallel CRC Circuits [J]. IEEE Design & Test of Computers, 2001,18(3)

[10] 俞迅. 32 位 CRC 校验码的并行算法及硬件实现[J]. 信息技术,2007(4):71-74

[11] 廖海红. 通信系统中的 CRC 算法的研究和工程实现[D]. 北京:北京邮电大学,2006:60-65

[12] 徐展琦,裴昌幸,董淮南. 一种通用多通道并行 CRC 计算及其实现[J]. 南京邮电大学学报:自然科学版,2008(2):53-57