文章编号:1671-4598(2014)08-2523-03 中图分类号:TP393 文献标识码:A

基于 SOM 和 PSO 的云计算异构资源聚类 MPI 并行算法

邢永峰
(南阳理工学院 软件学院,河南 南阳 473000)

摘要:针对云计算环境下大量用户请求同时到来造成的短暂峰值,使得用户的 QOS 服务需求不能得到有效保证,设计了一种基于 MPI 并行计算模型和 SOM 的异构资源聚类方法;首先,设计了一种改进的 MPI 的树状层次结构模型,然后,定义了基于 SOM 自组织映射的资源聚类算法,为了提高资源聚类精度,将 PSO 算法用于 SOM 的参数优化中,使得 SOM 在初始时刻就具有一个较好的连接结构;最后,为了充分满足用户请求的 QOS 需求,将 MPI 树状层次结构与基于 PSO 的 SOM 资源聚类算法相结合,并提出了具体的基于 MPI 的 SOM 资源聚类算法;为了验证文中方法,在 Matlab 仿真环境中进行测试,实验结果表明文中方法聚类精度为 100%,且与其它方法比较,具有较高的聚类精度和较少的执行时间,是一种云计算环境下的可行资源聚类方法。

关键词:云计算;并行算法;自组织映射;粒子群

MPI Parallel Algorithm for Heterogeneous Resource Scheduling Based on SOM and Particle Swarm Optimization

Xing Yongfeng
(School of Software Engineering, Nanyang Institute of Technology, Nanyang 47300, China)

Abstract: Aiming at the transient peak when the amounts of users appear at the same time in Cloud environment, leading to the QOS of user can not be guaranteed, a heterogeneous resource clustering method based on MPI parallel computing model and SOM is proposed. Firstly, the improved MPI tree hierarchical model is designed, then the resource clustering method based on SOM is defined, in order to improve the accuracy of resource clustering, the PSO algorithm is used to the optimization of SOM parameters and the SOM has a better structure at the initial time; finally, in order to satisfy the QOS need of users, the MPI tree hierarchical structure is applied in SOM clustering algorithm based on PSO, and the specific algorism based on MPI and SOM is proposed. For verifying the method in this paper, the experiment is simulated in the Matlab environment, the experiment shows the method in this paper has the clustering precision 100%, and compared with the other methods, it has the high cluster accuracy and less executing time, therefore, it is a feasible resource clustering method in cloud environment.

Keywords: cloud computing; parallel algorism; self organizing mapping; particle swarm

0 引言

云计算[1-3] (cloud computing) 是一种可以被称作系统平台或某类型应用程序的综合体。云计算平台的体系结构[4]主要是由 3 个部分组成,即基础设施即服务、软件即服务和应用级服务。作为底层的基础设施即服务是云计算体系结构的核心,Amazon 和 Google 都能提供基础设施即服务,当新的服务请求到达时,往往需要服务需求和资源池中已有的资源按某种算法[6]进行匹配,因此,用户服务的 QOS 往往不能得到保证。

目前已有的对资源进行聚类的主要工作有:文献 [7] 在本体的元数据模式和语义相似度的基础上,设计了一种基于语义特征树的混合聚类算法实现资源的聚类;文献 [8] 以最小总加工成本为目标建立了制造单元重构数学模型,并通过粒子群优化算法实现资源聚类;文献 [9] 提出了一种基于模糊 K 均值的资源聚合算法。

上述工作均研究资源聚类,但只有文献 [9] 是针对云计算环境,由于云计算环境下资源数量巨大,且性能各异,如果采用传统的聚类算法会使用户的需求得不到满足,因此文献 [9] 方法在云计算环境进行应用具有局限性,为此,本文提出了一种基于 MPI 模型和 SOM 网的云计算环境资源聚类方法,通过对聚类算法进行并行运行,以极大程度提高资源聚类的效率,通过实验表明了文种方法的有效性。

1 树状 MPI 并行编程模型

并行编程环境 (message passing interface, MPI) 是一个基于消息传递的并行程序函数库,MPI 系统通常是由标准库、头文件和运行、调试环境组成的,MPI 应用程序可以按照程序结构分为单程序模式、多程序模式和主从模式 (Master/Slave)。

主从模式 (Master/Slave) 作为一种广泛应用的 MPI 程序结构,其结构模型主要包含两类节点即 Master 节点和 Slave 节点,其功能主要可以描述为:

1) Master 节点负责任务的调度和数据划分,以及调度 Slave 节点进行处理任务处理和数据操作;

2) Slave 节点负责任务执行,当任务执行完毕后,将处理结果发送给 Master 节点进行汇总。

收稿日期:2013-12-06; **修回日期**:2014-03-12。

作者简介:邢永峰(1977-),男,硕士研究生,讲师,主要从事网络工程,数据库和软件工程方向的研究。

基本的主从式结构只有两层结构：第一层为 Master 节点；第二层为 Slave 节点。这不仅会因层次过少而导致 Slave 节点并行能力不高，同时当计算量过大时 Master 节点会形成瓶颈效应。

为提高基本主从结构的并行能力，文中设计了一种树状结构的 MPI 模型，将传统的两层主从结构扩展为多层的 MPI 结构，即由多个 MasterNode 层和一个 SlaveNode 层组成，如图 1 所示。

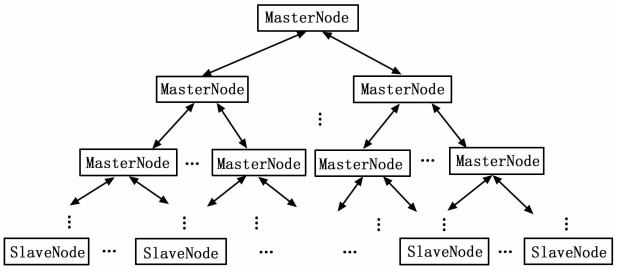

图 1　树状 MPI 并行结构模型

2　基于 PSO 的 SOM 资源聚类算法

2.1　SOM 资源聚类

自组织映射网络（self－organizing feature map，SOM）[10] 是由 Kohonen 提出的一种通过竞争、合作和权值调节来实现无监督训练的机器学习方法，一个 SOM 通常包含输入层和输出层两部分，输出层的神经元不仅与输入层神经元具有全连接的关系，同时输出层神经元也与输出层部分其它神经元有连接关系，如图 2 所示。

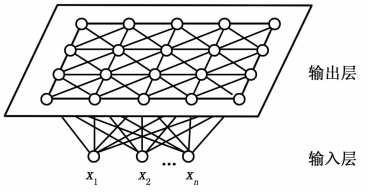

图 2　SOM 自组织映射网络

采用 SOM 对资源进行聚类，可以根据资源的各维属性如 CPU、带宽、内存等属性定义训练样本数据，根据样本数据对网络进行无监督的学习，直到所有训练样本与对应的激活神经元之间的距离最小，输入层的神经云个数即为资源属性维数，输出层神经元个数对应了资源的种类。

算法 1：SOM 资源聚类算法

初始化：输入神经元个数 n，输出神经元个数 m，当前迭代次数 t，最大迭代次数 T；

步骤 1：学习速率因子 $\psi(1)$，邻域半径 $N(1)$，以及初始权值 $W_{ij}(1)$；

步骤 2：对所有样本数据、输入层神经元与输出神经元之间连接的权值 $W_{ij}(1)$ 进行归一化，将其变换到（0，1）之间；

步骤 3：对于每个训练样本数据：计算其与各个输出层神经元之间的距离，将具有最小距离的输出层神经元作为获胜神经元，其中距离的计算定义为样本数据与对应的神经元连接权值之间的欧式距离，如下所示：

$$D_{ij}(t)=\sum_{i}^{n}(w_{ij}(t)-x_i(t))^2 \qquad j=1,2,\ldots,m \tag{1}$$

步骤 4：将具有最小 $D_{ij}(t)$ 的神经元作为获胜神经元 j：

$$j=\arg\min_{j}D_{ij}(t) \tag{2}$$

步骤 5：根据输出层获胜神经元连接的领域，对对应的神经元权值进行调整，如下所示：

$$w_{ij}(t+1)=w_{ij}(t)+\psi(t)(x_i-w_{ij}(t)) \tag{3}$$

步骤 6：对输出层获胜神经元的邻域及 SOM 的学习率进行更新：

$$\begin{cases}\psi(t)=\psi(t)(1-t/T)\\ N(t)=N(1)(1-t/T)\end{cases} \tag{4}$$

步骤 7：判断是否满足下列两个条件之一，如果满足，算法结束，输出各资源聚类：

1）学习率 $\psi(t)$ 下降为 0；

2）当前迭代次数达到最大值 T；

否则 $t=t+1$，并返回步骤 2。

2.2　基于 PSO 的资源聚类

SOM 算法相对经典的无监督学习方法 K－Means 等聚类算法，具有收敛速度快和可以采用有标签的样本数据进行更精确地学习等优点，但其收敛效果的好坏取决于 SOM 权值的初始分布是否与数据模式的解空间接近或一致，为此，提出了采用 PSO 对 SOM 网的初始权值进行优化。

PSO 算法（particle swarm optimization，PSO）源于对鸟群捕食行为的研究，采用粒子表示候选解，粒子具有两个特征：位置和速度，假设问题空间为 D 维，则粒子的位置 P 和速度 V 可以表示为：

$$\begin{cases}P=\{p_1,p_2,\ldots,p_D\}\\ V=\{v_1,v_2,\ldots,v_D\}\end{cases} \tag{5}$$

在每轮迭代中，粒子均通过两个最优值对自身的位置 $x_i(t+1)$ 和速度 $v_i(t+1)$ 进行不断地更新：

$$\begin{cases}x_i(t+1)=x_i(t)+v_i(t+1)\\ v_i(t+1)=wv_i(t)+c_1\lambda_1(pbst_i-p_i(t))+\\ c_2\lambda_2(gbst_i-p_i(t))\end{cases} \tag{6}$$

在式（6）中，$pbst_i$ 表示个体最优粒子，$gbst_i$ 表示全局最优粒子，w 被称为惯性权重，表示粒子现有速度对原有速度的依赖程度。c_1 和 c_2 为学习因子，其取值为（1，2）之间的随机数，λ_1 和 λ_2 为取值在（0，1）区间的随机数。

采用粒子群算法优化 SOM 权值的过程可以描述为：

1）粒子编码：采用粒子群算法对 SOM 的初始权值进行优化时，可以将粒子位置编码为：

$$P=\{w_{11},w_{12},\ldots,w_{1m}\ldots,w_{n1},\ldots,w_{nm}\} \tag{7}$$

其中，n 和 m 分别表示资源的属性数和资源的聚类数，分别对应了 SOM 中输入神经元和输出神经元的个数。

2）粒子位置更新：采用式（6）对粒子当前的位置 $x_i(t+1)$ 和速度 $v_i(t+1)$ 进行更新；

3）根据式（8）计算粒子适应度值，并记录全局最优粒子 $gbst_i$ 和个体最优粒子 $pbst_i$：

$$fit(p)=1/\sum_{i=1}^{n_s}D_{ij}(t) \tag{8}$$

其中，$D_{ij}(t)$ 表示样本 i 与其对应获胜神经元 j 的距离；

4）返回步骤 2）进行迭代直到全局最优粒子不再变化。

3　基于 MPI 的并行资源聚类

上述基于 PSO 和 SOM 聚类算法不是一种并行模式，不能

很好地适用于云计算环境，因此，设计了一种基于树状 MPI 结构模型的 SOM 资源聚类方法，资源聚类过程可以描述如下：

输入：MPI 结构模型中树的高度 u，MasterNode 节点的层数 v，SlaveNode 节点的个数 u，资源样本数据对象集合 $X=\{x_1,x_2,\ldots,x_n\}$，资源聚类 $S=\{S_1,S_2,\ldots,S_m\}$；

输出：将资源样本数据对象集合 $X=\{x_1,x_2,\ldots,x_n\}$ 划分到 m 个资源聚类 $S=\{S_1,S_2,\ldots S_m\}$ 中，并使得式（8）中取最大值；

步骤 1：根据 MPI 结构模型中树的高度 u，MasterNode 节点的层数 v，SlaveNode 节点的个数 u 来初始化图 1 所示的树状层次 Master/Slave 结构；

步骤 2：树状 MPI 中的根节点 MasterNode 根据 SlaveNode 节点的个数 u，将资源样本数据对象集合 $X=\{x_1,x_2,\ldots,x_n\}$ 分为 u 个集合 $X=\{X_1,X_2,\ldots,X_u\}$，并沿着树向下传递，直到到达叶子节点 SlaveNode 的父节点；

步骤 3：任意一个叶子节点 SlaveNode 节点 r 的父节点处，根据其所接收的数据分组 X_s 和 X_t，运行基于 PSO 的 SOM 权值优化算法，得到 SOM 中 nn 个权值 $w_{11},w_{12},\ldots,w_{1m}\ldots,w_{n1},\ldots,w_{nn}$ 的值，采用该值初始化 SOM，并将初始化的 SOM 传递给叶子节点 SlaveNode；

步骤 4：任意一个叶子节点 SlaveNode 节点 i，在接收了其父节点 MasterNode 的数据子集 X_s 或 X_t 后，运行算法 1，对 SOM 进行训练，得到 SOM 中 nn 个权值 $w_{11},w_{12},\ldots,w_{1m}\ldots,w_{n1},\ldots,w_{nn}$ 的最终值，并采用该 SOM 对资源数据子集 X_s 或 X_t 的聚类；

步骤 5：叶子节点 SlaveNode 节点在得到了资源数据子集 X_s 或 X_t 的聚类结果后，将资源聚类结果以及 SOM 网的参数值发送给上一层的 MasterNode 节点。

步骤 6：MasterNode 节点在收到其所属各 SlaveNode 节点或者下层的 MasterNode 节点发送给的资源聚类结果和 SOM 参数后，依次将其传递直到 MasterNode 节点；

步骤 7：顶层 MasterNode 节点在收集了 u 个资源样本数据子集 $X=\{X_1,X_2,\ldots,X_u\}$ 的聚类结果和 u 组 SOM 网的参数值后，计算任意两个数据子集 X_i 和 X_j 训练出的参数的欧式距离，选择与其它 $u-1$ 组参数距离最小的参数作为 SOM 最终的参数；

步骤 8：当新增加资源或资源发生变化后，将表征资源的样本数据向量输入 SOM，得到对应的资源聚类 $S=\{S_1,S_2,\ldots,S_m\}$。

4　仿真实验

采用 Matlab 工具进行仿真实验，对于云计算环境[11]下的任意资源，采用三元组对其进行描述：$R=\{cpu,memory,bandwidth\}$，其中 CPU 表示资源运行速度，单位为 GHz，$Memory$ 表示内存容量，单位为 G，Bandwidth 表示网络带宽，其单位为 Mbps/s，采集一组资源样本，部分样本如表 1 所示。

采用平移极差法对表 1 所示的数据进行归一化处理，然后采用文中基于 MPI 和 SOM 的并行聚类方法进行聚类，并与基于 K－Means 的聚类方法[9]进行比较，结果如图 3 和 4 所示。

表 1　部分资源样本

资源号	CPU/GHz	$Memory$/G	$Bandwidth$/(Mbps/s)	类型
1	1.2	8.1	33.1	1
2	0.8	9.5	62.8	1
3	1.8	9.4	79.5	2
4	1.6	9.7	95.3	2
5	1.7	7.5	23.6	3
6	1.5	6.6	14.3	3
7	0.6	3.8	11.1	4
8	0.2	3.5	62.4	4

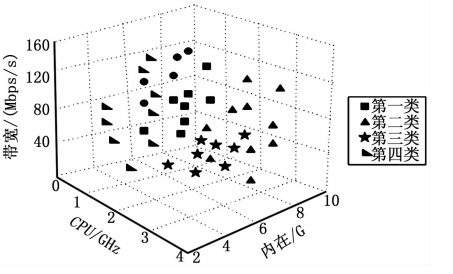

图 3　K－Means 聚类方法仿真结果

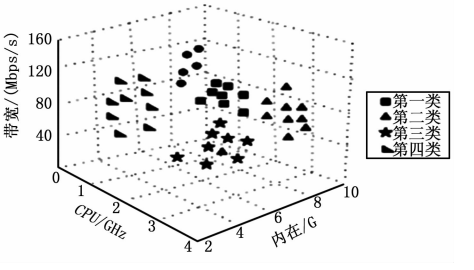

图 4　文中方法聚类仿真结果

从图 3 和图 4 中可以看出，文献［9］无法实现资源的精确聚类，存在较大误差，而文中方法较文献［9］的 K－Means 聚类方法能有效地对样本进行聚类，且具有较小的类内距离和较大的类间的距离，聚类精度为 100%，同时文中方法聚类的总仿真时间为 25 ms，而文献［9］中基于 K－Means 的聚类方法总仿真时间为 200 ms，显然，由于文中方法采用了 MPI 并行 SOM 聚类算法，使得聚类效率较其他方法高，具有很大的优越性。

5　结语

为实现云计算环境下用户任务的高效响应，设计了一种基于 MPI 的 SOM 并行聚类方法。首先，定义了基于 SOM 自组织映射的资源算法，为了进一步提高资源聚类的精确度，采用 PSO 算法对 SOM 中的各参数进行优化，通过获胜神经元来获取资源样本的聚类输出。最后，通过引入 MPI 并行树状结构模型，设计了一种基于 MPI 并行编程和 SOM 的资源聚类算法。仿真实验结果表明文中算法能较为精确地实现资源聚类，较传统方法具有分类精度高和效率高的优点，是一种适合异构云计算环境的资源聚类方法，具有很强的可行性。

（下转第 2549 页）

```
[FIFONumber]. RIR;
    RxMessage->DLC = (uint8_t)0x0F & CANx->sFIFOMail-
Box[FIFONumber]. RDTR;
    RxMessage->FMI = (uint8_t)0xFF & (CANx->sFIFOMail-
Box[FIFONumber]. RDTR >> 8);
    RxMessage->Data[0] = (uint8_t)0xFF & CANx->sFIFO-
MailBox[FIFONumber]. RDLR;
    RxMessage->Data[1] = (uint8_t)0xFF & (CANx->sFIFO-
MailBox[FIFONumber]. RDLR >> 8);
    RxMessage->Data[2] = (uint8_t)0xFF & (CANx->sFIFO-
MailBox[FIFONumber]. RDLR>>16);
    RxMessage->Data[3] = (uint8_t)0xFF & (CANx->sFIFO-
MailBox[FIFONumber]. RDLR>>24);
    RxMessage->Data[4] = (uint8_t)0xFF & CANx->sFIFO-
MailBox[FIFONumber]. RDHR;
    RxMessage->Data[5]= (uint8_t)0xFF & (CANx->sFIFO-
MailBox[FIFONumber]. RDHR >> 8);
    RxMessage->Data[6]= (uint8_t)0xFF & (CANx->sFIFO-
MailBox[FIFONumber]. RDHR>>16);
    RxMessage->Data[7]= (uint8_t)0xFF & (CANx->sFIFO-
MailBox[FIFONumber]. RDHR>>24);
    CAN_FIFORelease(CANx, FIFONumber); }
```

4　上层服务器监控软件的设计

本文中的上层服务器软件即机舱温度信号监控软件以 Windows XP 作为操作系统，在 .Net Framework 开发环境下，利用 Visual C#编程语言面向对象（OPP）的程序设计方法进行设计。机舱温度信号监控软件以面向船舶操作者为开发导向。本软件人机界面友好，用户能够从窗体界面上获得各种系统信息还可以通过软件设置修改系统的相关信息。运行界面如图 5 所示。

由图 5 可知，该软件能直接对 2 路 PT100 模拟量、4 路热电偶模拟量输入进行直观查询显示，也可以对报警值等参数进行设置。通过上述工作，船用温度数据采集单元具备了完好的实时控制和信息交互功能，符合实际应用中的要求。

5　系统调试

调试时需要采集模块与 PC 机软件信息交互，考虑到 PC 机上一般没有 CAN 总线的接口本系统采用广州致远电子有限公司生产的 USBCAN－Ⅱ模块与上位机相连，该模块集成了

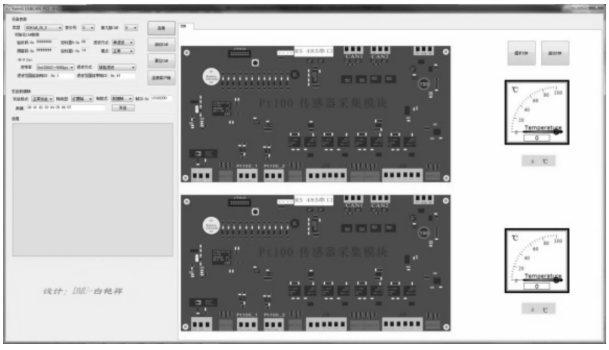

图 5　机舱温度信号监控软件运行界面图

两路 CAN－bus 接口和 USB 接口，通过 USB 电缆与 PC 机进行连接。然后打开上位机监控软件，并设置好相关参，温度信息便能能直观的反应给用户，故障或越限时能声光报警显示报警信息。系统运行情况良好，目前已成功应用于全任务轮机模拟器中。

6　总结

本文设计了一种船舶机舱温度数据采集单元，并完成了服务器端机舱温度信号监控软件的构建，该系统符合船舶机舱温度采集的要求。实现了当初的设计目标，已经多次成功应用于轮机模拟器中，下一步要完成的工作进一步提高系统的可靠性，应用于实船机舱自动化控制中。

参考文献：

[1] 王红旗，李　辉，陶　慧. 基于 CAN 现场总线的智能温度采集模块的设计与实现 [J]. 计算机测量与控制，2006，14 (1)：125－127.

[2] 邬宽明. CAN 总线原理和应用系统设计 [M]. 北京：北京航空航天大学出版社，1996.

[3] ST. STM32F107XX Datasheet [Z]. 2009.

[4] 张　强. 船用以太网 IO 单元及网络的设计与实现 [D]. 大连：大连海事大学，2012.

[5] 张雄飞，段军政，陈志坤. 热电偶冷端补偿器 LT1025 及其应用 [J]. 现代电子技术，2003，24：10－11.

[6] 张　华，赵文柱. 低功耗热电偶冷端补偿器 LT1025 原理及应用 [J]. 仪器仪表用户，2004，02：66－68.

[7] 王　帅. 基于 CAN 总线的船舶机舱智能监控系统的研究 [D]. 镇江：江苏科技大学，2011.

(上接第 2525 页)

参考文献：

[1] Alysson B, Rudiger K, Data P, et al. A look to the old－world sky：EU－funded dependability cloud computing research [J]. ACM AIGOPS Operating Systems Review, 2012, 46 (2)：43－56.

[2] 陈　康，郑纬民. 云计算：系统实例与研究现状 [J]. 软件学报，2009，20 (5)：1337－1348.

[3] Alvaro P, Condie T, Conway N, et al. Boom analytics：exploring data－centric, declarative programming for the cloud [A]. Proceedings of the 5th European Conference on Computer Systems [C]. New York, 2010：223－236.

[4] 张建华，吴　恒，张文博. 云计算核心技术研究综述 [J]. 小型微型计算机系统，2013，34 (11)：2418－2424.

[5] Lakshman A, Malik P. Cassandra：a decentralized structured storage system [J]. ACM Sigops Operating System Review, 2010, 44 (2)：35－40.

[6] 曹　洁，曾国荪，钮俊，许金超. 云环境下可用性感知的并行任务调度方法 [J]. 计算机研究与发展，2013，50 (7)：1563－1572.

[7] 熊　芳，黄宏斌，黄玉成，等. 一种基于语义相似度的信息资源语义聚类算法 [J]. 计算机工程与科学，2012，34 (11)：175－179.

[8] 王　雷，唐敦兵，许美健，等. 基于粒子群优化算法的制造单元聚类研究 [J]. 计算机集成制造系统，2009，15 (2)：328－332.

[9] 王溢琴，秦振吉. 云环境中改进模糊聚类的资源聚合 [J]. 计算机仿真，2013，30 (4)：365－368.

[10] Kohonen T. Self－Organizing Maps [M]. 2nd ed Berlin Springer－Verlag, 1997：145－152.

[11] 王永贵，韩瑞莲. 基于改进蚁群算法的云环境任务调度研究 [J]. 计算机测量与控制，2011，19 (5)：1203－1211.