第 39 卷 第 7 期 西南师范大学学报(自然科学版) 2014 年 7 月
Vol. 39 No. 7 Journal of Southwest China Normal University (Natural Science Edition) Jul. 2014

DOI:10.13718/j.cnki.xsxb.2014.07.023

基于带宽约束的云计算负载调度算法①

郑 卉[1], 郭 平[2], 李 琪[3], 陈海珠[1]

1. 重庆电子工程职业学院 软件学院, 重庆 401331; 2. 重庆大学 计算机学院, 重庆 400044;
3. 中国电信股份有限公司重庆分公司, 重庆 401122

摘要: 负载调度是云计算得以大规模应用及提高服务性能的关键技术, 对提高云供应商服务质量、用户满意度以及数据中心集群资源利用率等有极其重要的意义. 云计算环境中, 由于用户任务类型的不同, 对带宽的需求也不尽相同, 若不区分不同任务对不同带宽的要求, 可能会造成资源的浪费, 增长用户等待时间. 本文对经典 Min-Min 算法进行改进, 提出了 BCLL-Min-Min 算法, 该算法满足带宽需求约束, 并且实现相对负载均衡调度. 仿真实验表明, BCLL-Min-Min 算法能够适应云计算环境下任务多样性和不确定性的情况, 使用该调度算法可以提高集群的吞吐率、较大改善数据中心的负载均衡性.

关 键 词: 云计算; 负载均衡; BCLL-Min-Min 算法

中图分类号: TP302 **文献标志码**: A **文章编号**: 1000-5471(2014)7-0121-08

云计算通过 IaaS, PaaS 和 SaaS 来表示各种计算应用, 伸缩计算方法、云存储和负载均衡是云计算的主要研究内容[1]. 云计算环境下, 云数据中心是容纳计算设备、存储设备的集中之地, 它使用多节点服务器组成的集群对外提供服务[2]. 集群是解决服务器超载问题、提高资源利用率、减少用户等待时间的最好解决方案[3-4]. 优化集群中各个节点的负载调度, 对提高云数据中心的服务质量、集群吞吐率以及节约运营成本具有极其重要的意义[5].

现有的负载调度算法主要分为静态调度和动态调度两类[6]. 静态调度算法主要有轮转法(RR)[7]、加权轮转法(WRR)[8]等, 它们使用少量静态特征信息来描述可用计算资源的情况, 当任务到达时, 依照资源情况的固有顺序依次分配任务, 此类算法适用于一些小规模、单一配置的静态服务系统, 当任务出现多样性和不可确定性时, 负载均衡的效果不佳. 最小链接数优先法(LC)、加权最小链接数法(WLC)、最小负载优先法属于动态调度算法[8], 在任务到达时, 此类算法根据不同的考虑因素, 先计算一段时间内各个服务器的负载情况, 再将任务分配到不同的服务器上. 因此, 使用动态调度算法进行调度会有额外的计算开销, 但随时可以把握节点当前的负载状态, 故其性能比使用静态均衡算法的要好[9-10].

Min-Min 算法[11]也是一种典型的动态调度算法, 该算法通过计算两次最小值来完成资源的选择, 将大量的任务调度到能够最快执行对应任务的虚拟机上, 使得全部任务的总完成时间最小. 但是, 当任务大小差距较大、资源异构性大时, Min-Min 算法性能降低, 在云计算环境中表现不佳. LL-Min-Min[12]算法基于相对负载均衡的思想, 从完成时间最小的映射中选择相对负载差异最大的任务进行分配, 从而避免由于差异造成任务运行时间增长、虚拟机负载增加的情况. 当用户在使用云服务时, 带宽需求是较为常见的服务

① 收稿日期: 2014-01-12
基金项目: 国家自然科学基金(61201347); 重庆市自然科学基金项目(cstc2012jjA40022); 重庆市教委科学技术研究项目(KJ120634).
作者简介: 郑 卉(1982-), 女, 四川攀枝花人, 硕士, 讲师, 主要从事云计算, 生物计算方面的研究.

要求. 云数据中心作为服务提供方为了提高用户的满意度、满足用户对带宽的需求，必须优先将符合用户需要的虚拟机资源分给用户，并缩短任务完成时间. 由此，我们基于带宽约束条件，对LL-Min-Min算法进行改进，提出满足带宽约束的负载均衡调度算法BCLL-Min-Min.

1 云环境下的调度模型

云计算的体系结构主要包括三层：应用层、平台层和基础设施层[13]. 而云计算特殊的系统结构，决定了云计算的负载调度是一个二级的调度模式[14].

其中，第一级调度是用户任务－虚拟机的调度，第二级调度是虚拟机－物理主机资源的调度. 通过两级调度，不同任务的性能和其他因素需求能够得到较好的满足，不会出现分配给任务的资源少于需求，令执行时间增加，或是分配给任务的资源大于需求，造成资源浪费的情况. 由于虚拟化技术的存在，云计算任务的需求调度可以转化为任务－虚拟机资源的映射，即第一级调度模式.

1.1 云计算任务参数模型

设$C=\{C_1, C_2, \cdots, C_n\}$为云计算任务集合. 对于每一项任务$C_i$，可以用一个集合来表示，集合里的元素为该任务的参数，则

$$C_i=\{N_{\mathrm{ID}}, L_{\mathrm{length}}, W_{\mathrm{ExpB}}, S_{\mathrm{Bwsat}}\} \tag{1}$$

集合中N_{ID}表示任务的编号；L_{length}表示任务的预计执行长度；W_{ExpB}表示用户的期望带宽；S_{Bwsat}是用户满意度量标准，计算公式如下

$$S_{\mathrm{Bwsat}}=\begin{cases}1 & \text{if} \quad W_{\mathrm{ExpB}}<=w_{\mathrm{bw}}\\ 0 & \text{otherwise}\end{cases} \tag{2}$$

即如果分配给该任务的虚拟机资源满足了任务的需求，S_{Bwsat}值为1，否则值为0. 衡量算法性能时，需要统计所有任务完成时的总满意度.

1.2 云计算虚拟机资源模型

设$V=\{V_1, V_2, V_3, \cdots, V_m\}$为云计算数据中心的虚拟机集合，每个虚拟机用一个包含6个参数的集合表示，则

$$V_i=\{N'_{\mathrm{ID}}, n_{\mathrm{PE}}, V_{\mathrm{mips}}, m_{\mathrm{ram}}, w_{\mathrm{bw}}, f_{\mathrm{vl}}\} \tag{3}$$

其中，N'_{ID}表示虚拟机的编号；n_{PE}表示虚拟机中执行单元个数；V_{mips}表示虚拟机的执行速度；m_{ram}表示内存大小；w_{bw}表示虚拟机带宽大小，该参数与用户满意度量标准S_{Bwsat}相对应；f_{vl}表示虚拟机的负载. 另外虚拟机还有外部存储大小等因素，但是其他因素一般不在负载调度的考虑范围之类，所以没有列入该集合之中.

本文算法研究的问题是，结合任务模型的参数以及虚拟机参数模型，采用适当的负载调度策略，为公式(1)中的任务选择适当的公式(3)中的虚拟机资源.

1.3 云计算负载模型

我们将每个任务在每一个虚拟机上预计执行时间所构成的矩阵记为ETC(Expected time to compute)矩阵[12]，用矩阵$\boldsymbol{E}$表示. 本文算法的核心在于建立BCETC矩阵，用矩阵$\boldsymbol{B}$表示，在任务C_i和虚拟机V_j上的预计执行时间记为$\boldsymbol{B}[C_i, V_j]$，当某虚拟机不能满足用户需求时，则将矩阵中对应位置的值置为N. 影响$\boldsymbol{B}[C_i, V_j]$的因素很多，在本文中我们约定任务C_i在虚拟机V_j上的执行时间只跟任务长度和虚拟机执行速度有关. 在这个前提下，以下规则成立：

1) 如果一个虚拟机上同时有很多任务在执行，这些任务的总完成时间是一个固定值；

2) 如果一个任务相对于其他任务，在某个虚拟机上的执行时间最短，那么在其他虚拟机上该任务相对于其他任务，所需的执行时间也是最短的；

3) 如果一个虚拟机的执行速度最快，那么它执行任何一个任务都会比其他虚拟机执行该任务快.

根据以上规则，可以计算任务 C_i 在虚拟机 V_j 上的执行时间

$$T_{ij} = L_{\text{length}}/V_{\text{mips}} \tag{4}$$

$$\boldsymbol{B}[C_i, V_j] = T_{ij} \tag{5}$$

由公式(4)，(5)可以求出 n 个任务在 m 个虚拟机的上的执行时间，当某虚拟机不能满足用户的带宽需求时，则将矩阵中对应位置的值置为 N 来建立 $\boldsymbol{B}$ 矩阵.

$$\boldsymbol{B} = \begin{bmatrix} T_{11} & \cdots & T_{1m} \\ T_{21} & \cdots & T_{2m} \\ \cdots & \cdots & \cdots \\ T_{n1} & \cdots & T_{mm} \end{bmatrix} \text{（虚拟机 } V_j \text{ 不能满足用户需求任务 } C_i \text{ 时，} T_{ij} \text{ 的值为 } N\text{）} \tag{6}$$

在实际情况下，用户任务的大小不一，云数据中心虚拟机资源的处理能力不同，加大了负载的定义难度. 在本算法中，为了简化计算模型，我们可以做出如下两个假设：

假设 1：任务的负载只跟任务的大小(L_{length}) 有关，即任务越大，该任务的负载越大.

假设 2：虚拟机的处理能力只跟虚拟机的执行速度(V_{mips}) 有关，不考虑内存等其他因素. 可以根据虚拟机的历史表现，为虚拟机的处理能力定义一个权重值，用以表示虚拟机的执行速度.

根据以上的两个假设，定义任务在虚拟机上产生的负载为 f_{vl}，那么任务 C_i 分配到虚拟机 V_j 上产生的负载 f_{vl}，也就是任务 C_i 的长度除以虚拟机 V_j 的执行速度. 即

$$f_{\text{vl}} = L_{\text{length}}/V_{\text{mips}} \tag{7}$$

而从公式(4) 中可知，$L_{\text{length}}/V_{\text{mips}}$ 等于任务 C_i 在虚拟机 V_j 上的执行时间 T_{ij}，所以虚拟机 V_j 的总负载便为其上所有任务的执行时间总和，即

$$f'_{vl} = \sum T_{ij} \tag{8}$$

由公式(8) 可以得出，所谓负载均衡调度，就是使得所有虚拟机上的任务执行时间大致一致. 由此，我们可以定义判定负载均衡程度的指标

$$\rho = T_{\min}/T_{\max} \tag{9}$$

其中，$T_{\min}$ 表示所有虚拟机中任务总完成时间的最小值，$T_{\max}$ 表示所有虚拟机中任务总完成时间的最大值，负载均衡指标即虚拟机中任务总完成时间的最短时间和最长时间的比值.

由公式(9)，可以得到以下结论

1) $T_{\max} = 0$，说明此时任务未开始调度.

2) $\rho = 0$ 且 $T_{\max} \neq 0$，说明虚拟机空闲.

3) $\rho = 1$ 且 $T_{\min} = T_{\max}$，说明虚拟机执行任务总时间的最短时间和最长时间相等，负载均衡度最好.

4) 为了实现负载均衡，ρ 越接近 1 越好.

2　BCLL-Min-Min 算法

当用户使用云服务申请虚拟机资源时，经常会提出对带宽的要求. 作为服务提供方为了提高用户的满意度，满足用户带宽需求，必须优先将符合用户需要的虚拟机资源分给用户. 同时，任务的完成时间也应当尽量的小. BCLL-Min-Min 算法的提出正是为了满足这些目标.

2.1　算法的思想

BCLL-Min-Min 算法是在满足用户带宽需求情况下，保证一定的负载均衡性. 其调度优化目标主要有：尽量实现负载均衡；满足不同任务的带宽需求；保证使用较短时间完成任务.

我们约定，BCLL-Min-Min 算法中任务的调度满足以下两个条件：

1) 任务与任务之间不存在关联性；

2)将某任务分配给某虚拟机后,在该任务执行完毕之前,其他分配到该机器的任务只能等待.

基于以上条件,本文提出以下调度原则:

1)若有满足任务带宽需求的虚拟机,则将任务分配给虚拟机.具体任务的带宽需求可通过查询系统历史信息,或是在用户申请资源时提出.

2)若多个虚拟机资源都满足任务带宽需求,则将任务分配给执行时间最小的虚拟机.

基于$\boldsymbol{B}$矩阵的调度思想是:如果虚拟机V_j上的资源满足任务C_i对带宽的需求,则,执行C_i,$\boldsymbol{B}[C_i, V_j]$的值为任务C_i在虚拟机V_j上的预计执行时间;如果虚拟机V_j的资源不足以满足任务C_i对带宽的需求,则不能执行任务C_i,$\boldsymbol{B}[C_i, V_j] = N$($N$表示不能建立任务$C_i$到虚拟机$V_j$的映射).

以表1的$\boldsymbol{B}$矩阵为例,虚拟机V_1,V_2,V_3,V_4的可以提供的带宽依次增大.任务C_2的带宽需求较低,4个虚拟机都能满足,因此第二行不变;任务C_1的带宽只有V_1不能满足,令$\boldsymbol{B}[C_1, V_1] = N$,即不能映射,同理可得出其他任务和虚拟机之间的映射关系.

表1 $\boldsymbol{B}$矩阵示例(单位:ms)

	V_1	V_2	V_3	V_4
C_1	N	22	36	38
C_2	15	19	33	30
C_3	N	N	23	27
C_4	N	N	20	22
C_5	N	N	N	13

在以上调度原则的基础上,为了达到一定的负载均衡,满足带宽需求的虚拟机个数越少的任务越要优先调度,即必须满足的调度顺序为:

1)只有一台虚拟机满足其要求的任务;

2)至少两台虚拟机满足其要求的任务;

3)所有虚拟机都能满足其要求的任务;

4)所有虚拟机都不能满足其要求的任务.

若不采用这样的调度顺序,选择范围大的任务可能会先分配到带宽小的虚拟机上,而选择范围小的任务极可能迟迟不能执行,这样会造成严重的负载不均.同时为了保证$\boldsymbol{B}$矩阵里的N所对应的任务(虚拟机映射对中的任务)不能分配给虚拟机,N的取值十分重要.为了实现这两个目标,同时考虑到优先选择平均值和最小值差距最大的任务.那么,显然可令N为一较大的值(令N值大于所有任务的完成时间总和),这样在任务分配进行平均值计算时,满足任务要求的虚拟机越少,该值便会越大,与任务最小完成时间的差值也就越大,算法会优先选择该任务.

对于表1,在同时考虑负载均衡的情况下,当前的映射方案为$[C_5, V_4]$,$[C_3, V_3]$,$[C_4, V_3]$,$[C_1, V_2]$,$[C_2, V_1]$.任务C_5最先映射的原因是只有V_4虚拟机满足C_5的要求,任务C_4,任务C_3都有两台虚拟机满足要求,在这种情况下,需要计算任务的最小完成时间和除去最小完成时间的平均时间的差值,差值大的任务先进行映射,任务C_3的最小完成时间为23,除去最小完成时间的平均完成时间是27,二者差值为4,而任务C_4的最小完成时间是20,除去最小完成时间的平均完成时间是22,二者差值为2,则差值大的任务C_3先进行映射.所有虚拟机都能满足任务C_5的要求,此时将任务C_5最后映射,并分配给执行时间最小的虚拟机.

2.2 算法描述

BCLL-Min-Min算法描述如下:

输入:虚拟机资源参数、任务参数

输出:映射方案

步骤:

初始化：令 C 为所有任务集合；V 为所有虚拟机集合；集合 $CTOV$ 记录当前的映射方案；数组 $VMLoad$ 记录当前所有虚拟机的负载；
先根据 n 个任务在 m 个虚拟机的上的执行时间得到 $\boldsymbol{E}$ 矩阵；
for C 中的每一个任务 C_i；
　令 $\boldsymbol{E}$ 矩阵中 T_{ij} 加上 $VMLoad[j]$；
　if E 矩阵中虚拟机 V_j 不能满足任务 C_i 的带宽需求
　　$\boldsymbol{B}$ 矩阵中 $T_{ij}=N$；
　end if
　for　C 中的每一个任务 C_i；
　　取满足带宽需求的虚拟机个数最少的任务；
　　if 满足带宽需求的虚拟机个数为 1
　　　建立任务－资源映射对保存进 $CTOV$；
　　　从任务集合 C 中删除已分配的任务，更新 $VMLoad$ 数组；
　　else
　　　for　V 中的每一个虚拟机 V_j；
　　　　找到每一个任务的最小完成时间；
　　　　计算每一个任务的除去最小完成时间的平均时间；
　　　　计算平均时间和任务最小完成时间的差值得到 LL 值；
　　　end for
　　　选择差值最大的一组任务 -资源映射对保存进 $CTOV$；
　　　从任务集合 C 中删除已分配的任务，更新 $VMLoad$ 数组；
　　end if
　end for
end for

BCLL-Min-Min 算法既对相对负载值最大的任务优先进行分配，又加入了用户满意度的限制，很好的克服了 Min-min 算法在优化目标上的单一性问题以及负载不均的问题.

3　仿真实验结果与分析

为验证 BCLL-Min-Min 算法的性能，我们采用由澳大利亚墨尔本大学开发的云计算仿真工具 CloudSim[7] 对 BCLL-Min-Min 算法、轮转法(CloudSim 平台自带的轮转调度算法，简称 RR 算法)、Min-Min 调度算法(简称 MM 算法)在任务的完成时间、虚拟机的负载、不同数量带宽需求类任务的完成情况三个方面进行对比实验.

文献[12]已经针对异构任务、异构环境下任务长短数量参差不齐的情况，对 LL-Min-Min 算法进行了实验和验证. 本文主要针对 Min-Min 算法的强项，即在任务大小差距不大的情况下进行对比验证，观察带宽约束下的任务执行情况.

以任务的总执行时间和虚拟机的负载均衡度为标准，选择以下两组实验数据：

第一组：分别有 200，400，600 个任务和 15 个虚拟机. 主要考查的指标为任务的时间跨度、虚拟机的负载情况以及有带宽需求任务的完成情况. 所有参数均随机产生，任务和虚拟机性能满足异构但差距不大的条件，有带宽需求的任务比例为 50%.

第二组：在第一组的基础上，设计了 400 个异构的任务且长短任务数量差距不大，15 个异构虚拟机资源，调整有带宽需求任务的比例，分别为 40%，60%和 80%，观察不同比例下的有带宽需求的任务在三种

算法下的具体表现.

由图1可以看出,任务总完成时间,即执行跨度最长的是RR算法,其次是BCLL-Min-Min算法,Min-min算法的时间跨度与BCLL-Min-Min算法的时间跨度十分接近.这是由于RR算法不考虑虚拟机和任务的特性,只是按序将任务分配给虚拟机,在这种情况下,RR算法的性能最差.同时从图1中也能看到,BCLL-Min-Min算法在满足用户带宽需求的基础之上,完成时间也有所保障,与MM算法相差并不大.

虚拟机的负载均衡度对比如图2所示.从图中可以看出,BCLL-Min-Min算法在50%带宽需求下,虚拟机负载均衡度最高,负载最为均衡.由于算法特性,并且有带宽需求的任务只占50%,其余任务一定程度上弥补了有带宽需求任务所造成的负载不均现象,很好的实现了基于负载均衡的优化目标.Min-Min算法在任务数量较多的情况下表现也不错,但仍然差于BCLL-Min-Min算法,RR算法性能最差.

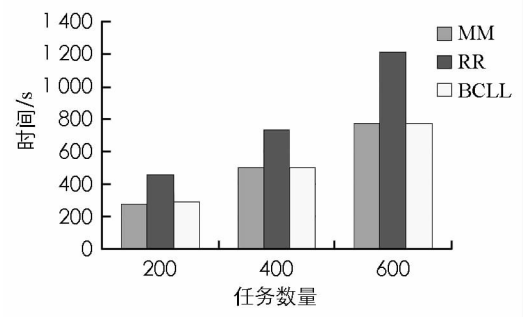

图1 完成时间对比

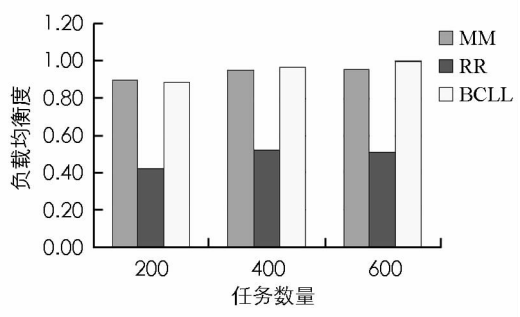

图2 虚拟机负载均衡度ρ对比

如果虚拟机满足了有带宽需求任务的需求,由公式(2)可知S_{Bwsat}值为1;200个任务中,有带宽需求的任务的比例为50%时,有带宽需求的任务数最大值为100;同理,400个任务中有带宽需求的任务数是200,600个任务有带宽需求的任务数是300.此时,有带宽任务的完成情况对比如图3所示,从图3中可以看出BCLL-Min-Min算法每次都能很好地满足用户任务的带宽需求.Min-Min算法由于没有带宽需求的优化目标,能够满足一部分任务,而RR算法采用固定的轮转分配,必定会满足一些有带宽需求的任务,其用户满意度和Min-Min算法的表现不相上下.BCLL-Min-Min算法对比Min-Min算法,通过产生微小的时间差距换来了较大幅度的用户任务的满意度增长.

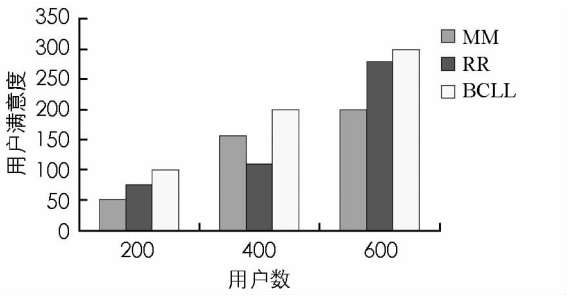

图3 带宽需求类任务的完成满意度对比

为了考察不同比例下有带宽需求的任务对负载均衡度的影响,在第二组实验中,我们调整有带宽需求任务的比例.为了体现BCLL-Min-Min算法与Min-Min算法的比较,并且针对Min-Min算法的强项,设计了400个异构的任务(长短任务数量差距不大),15个异构虚拟机资源,所有参数随机产生.三种比例下的用户满意度实验结果比较如图4所示.

从图4可以清晰地看出,针对不同比例下的有带宽需求的任务,BCLL-Min-Min算法基本能够满足用户的带宽需求,而且随着比例的增长增幅也比较明显.但Min-Min算法和RR算法随着带宽比例的增加增长的幅度并不大,并且在满足带宽需求的表现上也不如BCLL-Min-Min算法.

图5所示是有带宽需求的任务占总任务三种不同比例下,不同算法的时间跨度对比.实验表明,随着有带宽需求任务的数量的增加,BCLL-Min-Min算法的时间跨度越来越高.其原因是增加了带宽限制后,可供分配的虚拟机数量计算量增加,满足了带宽需求必然后损失一定的时间跨度.但在实际情况中,一般不会有这么多有带宽需求的任务.

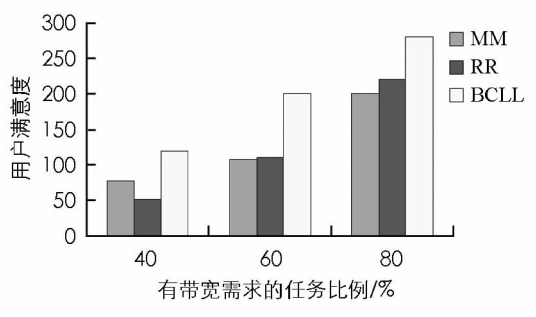

图 4　不同比例有带宽需求的任务满意度对比

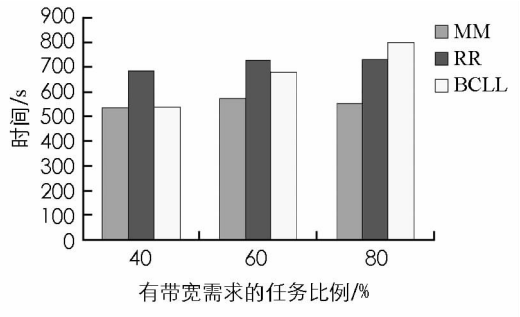

图 5　时间跨度对比

总体而言，BCLL-Min-Min 算法在异构任务、异构环境下的表现优于另外两个算法，且能最大限度的满足用户的带宽需求. 在各种情况下，负载均衡度、任务完成满意度上都是最优的，只是当有带宽需求任务的比例增加时，任务的完成时间会略微降低.

4　结束语

云计算的计算环境复杂，而且执行的计算任务也是多种多样的. 本文针对云计算环境的特性，综合考虑了虚拟机异构性和任务长短不一的情况，对传统的 Min-Min 算法以及相对负载均衡的 LL-Min-Min 算法进行改进，提出了 BCLL-Min-Min 算法，该算法可以同时满足带宽需求约束，实现了相对负载均衡调度. 通过云计算仿真工具 CloudSim 上的仿真实验表明，BCLL-Min-Min 算法在完成任务耗时长和耗时短任务数量差距不大的情况下，负载均衡性最优，由于加入了带宽限制，在优化目标上较其余二者最优，并且保证了负载均衡能满足复杂云计算环境下的任务调度要求. 随着带宽需求任务数的增加，会对任务的执行时间产生一定的影响，但任务的带宽需求在任何比例下都能够得到满足，从而提高了集群的吞吐率，较大改善数据中心的负载均衡性.

参考文献：

[1] 周相兵，马洪江，苗　放. 云计算环境下的一种基于 Hbase 的 ORM 设计实现 [J]. 西南师范大学学报：自然科学版，2013，38(8)：130－135.

[2] 曾龙海，张博锋，张丽华，等. 基于云计算平台的虚拟集群构建技术研究 [J]. 微电子学与计算机，2010，27(8)：31－35.

[3] 杨　越，闫连山，张志勇，等. 面向集群服务器大规模并发的改进负载均衡调度系统 [J]. 微电子学与计算机，2013，30(12)：54－56.

[4] 赵　钢. 基于分布式多引擎架构的网格工作流管理系统 [J]. 西南大学学报：自然科学版，2012，34(11)：110－107.

[5] BUYYA R. High Performance Cluster Computing：Systems and Architectures [M]. Michigan：Prentice Hall PTR，1999.

[6] DESAI T，PRAJAPATI J. A Survey Of Various Load Balancing Techniques And Challenges In Cloud Computing [J]. International Journal of Scientific & Technology Research，2013，11(2)：158－161.

[7] The clouds lab. CloudSim：A Framework for Modeling and Simulation of Cloud Computing Infrastructures and Services [EB/OL]. [2011－09－02]. http：// www. gridbus. org/ cloudsim/.

[8] CHOI E. Performance Test and Analysis for an Adaptive Aoad Balancing Mechanism on Distributed Server Cluster Systems [J]. Future Generation Computer Systems，2004，20(2)：237－247.

[9] PING G，NING L J，SU L P，et al. A New Strategy of Resource Management for Cloud Computing [J]. Information Technology Journal，2013，12(17)：3964－3969.

[10] KATYAL M，MISHRA A. A Comparative Study of Load Balancing Algorithms in Cloud Computing Environment [J]. International Journal of Distributed and Cloud Computing，2013，12(1)：5－14.

[11] BRAUN T D，SIEGEL H J，BECK N，et al. A Comparison of Eleven Static Heuristics for Mapping a Class of Independ-

ent Tasks onto Heterogeneous Distributed Computing Systems [J]. Journal of Parallel and Distributed computing, 2001, 61(6): 810－837.

[12] 李 琪. 云环境下负载均衡策略研究 [D]. 重庆: 重庆大学, 2013.

[13] SADHASIVAM S, NAGAVENI N, JAYARANI R, et al. Design and Implementation of an Efficient Two-Level Scheduler for Cloud Computing Environment [C]//Advances in Recent Technologies in Communication and Computing, 2009. ARTCom '09. International Conference on, 2009: 884－886.

[14] LIU G, LI J, XU J C. An Improved Min-Min Algorithm in Cloud Computing [C]// Proceedings of the 2012 International Conference of Modern Computer Science and Applications. Berlin: Springer, 2013: 47－52.

On a Load Balancing Algorithm Based on Bandwidth Constraintsin Cloud Computing

ZHENG Hui[1], GUO Ping[2], LI Qi[3], CHEN Hai-zhu[1]

1. *Software School, Chongqing College of Electronic Engineering, Chongqing 400044, China;*

2. *College of Computer Science, Chongqing University, Chongqing 400044, China;*

3. *China Telecom Corporation Limited Chongqing Branch, Chongqing401122, China*

Abstract: The scheduling strategy on load balancing, which is one of key techniques in cloud computing, plays an important role inimproving high service performance, customer satisfaction and utilization of clusterresource in data center. In cloud computing, allocating the same bandwidth to different tasks indiscriminately may cause the computing resources to be wasted and users' waiting time to be lengthened since different tasks require different bandwidth. In this paper, we have improvedMin-Min algorithm which is one of classical load balancing algorithms and present a new improved algorithm named BCLL-Min-Min. It can satisfy the bandwidth constraint and implement the relative load balancing scheduling. The simulated experiments show that our proposed algorithm is more available for the diverse and uncertain tasks in cloud computing. It improves the load balance in data center and enhances the throughput in the cluster.

Key words: cloud computing; load balance; BCLL-Min-Min algorithm

责任编辑 崔玉洁