

基于云计算的负载均衡方法研究

白延敏, 吕树红

(正德职业技术学院 电子与信息技术系, 江苏 南京 211106)

摘 要: 针对虚拟机层和物理机层之间存在的链式负载关系, 建立了双层控制的负载均衡框架, 对负载度和相对负载均衡度进行建模, 引入了综合权值和迁移概率的计算, 提出了任务调度算法和动态迁移算法. 实验结果表明该方法在实现了物理机和虚拟机层的负载均衡的同时, 提高了资源利用率.

关键词: 负载均衡; 云计算; 综合权值; 迁移概率

中图分类号: TP393.1

文献标识码: A

文章编号: 1000-7180(2013)12-0077-04

Research on Load Balancing Method Based on Cloud Computing

BAI Yan-min, LV Shu-hong

(Department Electronics and Information Technology, Zhengde Polytechnic College, Nanjing 211106, China)

Abstract: Aiming at the chain relationship between the load of the virtual machine and the physical machine, the paper builds a load balancing framework based on the double-layer controlling, and the model of load degree and relatively load balancing degree. The paper also introduces the computing method of integrated weight and migration probability. With this, the task scheduling and live migration algorithm effect on the load balancing. The result shows that this method implements the load balancing while improving the resource utilization.

Key words: load balancing; cloud computing; integrated weight; migration probability

1 引言

云计算是分布式计算、并行计算和网格计算等技术的综合发展^[1]. 典型的云架构一般分为三个层次, 基础设施(Infrastructure)层, 平台(Platform)层和软件(Application)层. 在大规模的资源集群环境中, 负载的不均衡将会导致部分节点资源利用率过低, 从而导致资源浪费; 同时部分节点资源利用率过高, 影响到上层服务的性能. 因此, 负载均衡成为云计算中的关键技术.

虚拟化是云计算的重要支撑技术. 通过虚拟化, 实现了物理资源的抽象和整合, 从而在云架构的基础设施层, 形成了虚拟机(Virtual Machine)、物理机(Physical Machine)的两层结构, 两层之间存在着多对一的映射. 虚拟机和物理机作为两层资源提供的主体, 统称为节点. 其区别在于二者负载不同. 因而, 云环境中, 负载均衡具有两种不同的意义. 第一级体

现在执行任务的虚拟机层, 第二级体现在为虚拟机提供资源的物理机层, 两级之间形成链式负载关系^[2].

当前的研究多集中于单层的负载均衡. 文献[3-4]的负载均衡方法, 主要针对的是物理机层, 通过虚拟机动态迁移的方式, 提高物理机资源的利用率, 实现虚拟机对物理机层的重新映射. 文献[5-6]则对于虚拟机集群, 使用传统的服务器负载均衡方法, 通过任务的调度均衡分配到各个虚拟机中. 本文针对两层的链式负载关系, 提出了一种自适应的负载均衡方法, 给出了该方法的实现模型, 并提出了基于该方法的负载均衡算法, 最后得到了实验结果.

2 负载均衡模型

2.1 负载均衡框架

基于云计算的负载均衡采用动态反馈的方式, 通过实时采集节点负载数据, 考虑节点的负载情况,

通过重新分配负载,达到负载均衡的目的.其框架设计如图 1 所示.

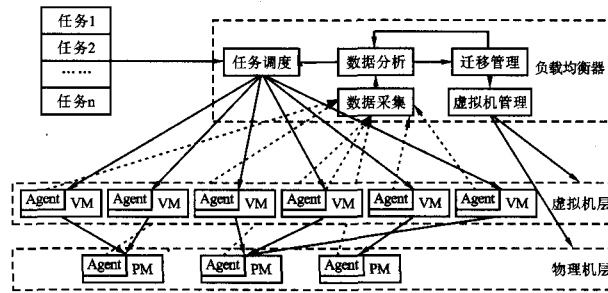


图 1 负载均衡框架

负载均衡框架主要由物理机层、虚拟机层和负载均衡器组成.物理机层由物理机(PM)集群组成.物理机的资源,通过虚拟化技术进行抽象整合,并分配给创建的虚拟机,从而在其上形成虚拟机层.虚拟机层由虚拟机(VM)集群组成.虚拟机内部部署有应用程序,负责执行任务.物理机和虚拟机内部均驻守有 Agent. Agent 定时采集宿主的负载数据,定期向负载均衡器发送.

负载均衡器主要有五个模块组成.

(1)数据收集模块:负责接收来自 Agent 的负载数据,并进行修正,提交给数据分析模块.

(2)数据分析模块:根据负载数据,通过模型计算出整个集群的负载度和各个节点的综合权值,维护节点的综合权值表.数据分析模块将虚拟机的综合权值表提供给任务调度模块,将物理机的综合权值表和整个集群的负载度提供给迁移管理模块.

(3)任务调度模块:根据虚拟机的综合权值表和任务调度算法,将任务调度至虚拟机中执行.

(4)迁移管理模块:考虑到虚拟机迁移本身将消耗额外的资源.而频繁的迁移会增加物理机过多的负载.因此采取先预测,再迁移的策略,根据物理机的综合权值表和动态迁移算法,作出迁移、关闭虚拟机的指令发送给虚拟机管理模块,将更新虚拟机综合权值表的决策反馈给数据分析模块.

(5)虚拟机管理模块:负责管理虚拟机和物理机的生命周期.接收来自迁移管理的指令,监控执行虚拟机的迁移工作.

2.2 负载的度量和相对负载均衡度

虚拟机的集合为 VM,物理机的集合为 PM.物理机和虚拟机统称为节点.虚拟机 j 位于物理机 i 上,那么即存在从虚拟机到物理机的映射 m_k .从虚拟机到物理机的映射集合为 M .

$$m_k: j \rightarrow i \quad i \in \text{PM}, j \in \text{VM}$$

$$M = \{m_1, m_2, \dots, m_n\} \quad (1)$$

定义 1 节点资源:使用资源指标,用以量化节点可以支配的资源.节点 i 的资源用集合 R_i 表示.

节点的资源指标如表 1 所示.

表 1 资源指标

资源类型	资源说明	度量单位	资源表示
CPU	CPU 处理能力	MIPS	Rcpu
内存	内存容量	MB	Rmem
硬盘	硬盘容量	GB	Rdisk
网络	网络带宽	MB/s	Rnet

$$R_i = \{Rcpu_i, Rmem_i, Rdisk_i, Rnet_i\} \quad (2)$$

定义 2 节点负载数据:节点在各个负载指标上的消耗情况.节点 i 的负载数据用集合 D_i^t 表示.

本文使用的负载指标有 CPU 使用率 CPU_i^t ,内存使用率 MEM_i^t ,硬盘使用率 $DISK_i^t$,网络使用率 NET_i^t .

$$D_i^t = \{CPU_i^t, MEM_i^t, DISK_i^t, NET_i^t\} \quad (3)$$

定义 3 节点负载度:用以量化描述节点在某一时刻的负载状态.节点 i 在时刻 t 的负载度用 L_i^t 表示.

引入一组可以动态调整的权值 W_i 表示各个负载指标的重要性.

$$L_i^t = W_1 \times CPU_i^t + W_2 \times MEM_i^t + W_3 \times DISK_i^t + W_4 \times NET_i^t$$

$$\sum_{i=1}^4 W_i = 1 \quad (4)$$

为描述所有节点组成的集群目前的负载状态,引入平均负载度 $L_{average}$.

$$L_{average} = \frac{1}{n} \sum_{i=1}^n L_i^t \quad (5)$$

定义 4 安静状态:节点 i 在时刻 t 没有负载.那么安静状态下, $L_i^t = 0$.考虑负载指标中, CPU 和网络是与时间相关的资源.对于负载的度

量,需要对于目前等待计算的任务量和等待发送的数据量进行预测,从而估计执行完成现有任务所需要的时间.因此使用CPU和网络指标作为评估负载的主要依据.同时,内存和硬盘具有空间相关的特性.节点在接受任务分配时,不能接受需求超过其剩余内存或硬盘的任务.因此,使用内存和硬盘指标作为任务分配的约束条件.

定义5 相对负载均衡度:用以评估节点*i*在整个集群中的相对负载程度.节点*i*在时刻*t*的相对负载均衡度为 P_i^t .

$$P_i^t = \frac{L_i^t}{\frac{1}{n} \sum_{i=1}^n L_i^t} \quad (6)$$

$P_i^t < 1$ 说明节点*i*负载在集群中负载相对较轻,而 $P_i^t > 1$ 说明相对负载较重.设 P_i^t 在区间 $[0, P_{\max}]$ 之间.如果 $P_i^t > P_{\max}$,则设置 $P_i^t = P_{\max}$.

2.3 综合权值和迁移概率

考虑到虚拟机与物理机的链式负载关系,在物理机负载过重时,则应对于其上的虚拟机分配较少任务,以减轻虚拟机对于物理机资源的消耗,从而降低其物理机的负载;反之,在物理机负载较轻时,则可以迁移虚拟机至该物理机中,或调度任务至物理机内的各虚拟机中,增加其负载,以提升资源利用率.

根据虚拟机与物理机的链式负载关系,引入综合权值,为任务调度和虚拟机迁移提供参数依据.在设计综合权值时,为物理机和虚拟机分别设计综合权值模型.当物理机综合权值为零时,物理机内部的各个虚拟机综合权值将自动更新为零,从而等待关闭或迁移,以完成物理机的关闭.零权值的设计使得物理机逐渐趋于安静状态,这对于物理机的维护和缩减云资源、提高资源利用率均有重要意义.

定义6 物理机综合权值:物理机*i*在时刻*t*综合权值为 C_i .

考虑相对负载均衡度越高,则其在集群中负载较重,因而其可用的资源越少,因而其综合权值越低.相对负载均衡度与物理机综合权值成反比例关系. C_i 计算公式为

$$C_i = \frac{1}{P_i^t} \quad i \in \text{PM} \quad (7)$$

定义7 虚拟机综合权值:虚拟机*j*在时刻*t*综合权值为 V_j^t .

$$V_j^t = \frac{1}{P_j^t} \times C_i \quad j \in \text{VM}, i \in \text{PM}, j \rightarrow i \in \text{M} \quad (8)$$

在式(8)中,虚拟机*j*属于物理机*i*. W_i^t 是物理

机*i*的综合权值.

定义8 迁移概率:用以提供虚拟机迁移的参数依据.虚拟机*j*从物理机*i*迁移到物理机*l*的概率为 U_j^t .

目前虚拟机*j*的资源为 R_j ,物理机*l*的资源为 R_l ,负载为 D_l^t .假设虚拟机*j*迁移到物理机*l*中,那么虚拟机*j*作为物理机的负载将被转移至物理机*l*中.那么物理机*l*的负载将变为 D_l^{t+1} .对于 D_l^{t+1} 中各类型资源进行分别计算,计算通式为

$$D_l^{t+1} = \frac{R_l \times D_l^t + R_j}{R_l} \quad (9)$$

根据通式,可以分别计算出各类型资源的负载数据 CPU_l^{t+1} , MEM_l^{t+1} , DISK_l^{t+1} , 以及 NET_l^{t+1} .根据式(4),则可以计算此时物理机*l*的负载度为 L_l^{t+1} .考虑接受虚拟机*j*的物理机*l*,在整个集群中应相对负载较低,且在接受了虚拟机*j*后,负载相对变化较小. U_j^t 的计算公式为

$$U_j^t = C_l \times \left(1 - \frac{L_l^{t+1} - L_l^t}{L_{\text{average}} - L_l^t}\right) \quad (10)$$

3 负载均衡算法

3.1 任务调度算法

任务调度算法其基本思想是首先对虚拟机集群运行中的每台虚拟机的负载度进行计算,并进而计算出平均负载度和每台虚拟机的相对负载均衡度及综合权值.在任务到达时,将任务调度至综合权值最大的虚拟机中.假设 K 为任务集合.

Step1:判断 K 是否为空,为空则转向 Step7;

Step2:任选 $k \in K$;

Step3:获取虚拟机负载数据,根据式(4),对所有的 $j \in \text{VM}$,求 L_j^t ;

Step4:根据式(8),对每个 $j \in \text{VM}$,求 V_j^t ;

Step5:根据 V_j^t 进行排序筛选,选择虚拟机 v ,使得 $V_v^t = \max\{V_j^t \mid j \in \text{VM}\}$;

Step6:将任务 k 分配给虚拟机 v ;

Step7:返回 Step1;

3.2 动态迁移算法

动态迁移算法其基本思想是首先对物理机集群中的每台运行中的物理机的负载度进行计算,并进而计算出平均负载度和每台物理机的相对负载均衡度及综合权值.依据平均负载度,估计整个集群的负载情况.整个集群负载低于阈值 L_{low} 时,说明整个集群资源利用率偏低,因此综合权值最高的物理机综合权值重置为零,执行关闭操作,从而降低能耗,提

高资源利用率;而在整个集群负载高于阈值 L_{high} 时,说明整个集群整体负载过重,资源不足,需要增加物理机,并在其上开辟虚拟机,从而增加整个集群的能力。

在对综合权值最高的物理机关闭前,对其中的虚拟机有迁移或者关闭两种基本策略.在虚拟机进行迁移时,采用先预测的方式,首先计算迁移到其他物理机的迁移概率.当迁移概率 U_j^i 高于阈值 $U_{migration}$ 时,实行虚拟机 j 向物理机 l 的迁移.如果没有符合条件的物理机,则将该虚拟机的综合权值更新为零.那么根据任务调度算法,该虚拟机将不再能接受任务.在该虚拟机执行完现有任务后,对该虚拟机进行关闭.在该物理机上所有虚拟机完成迁移或关闭后,关闭该物理机.具体算法如下:

Step1:获取物理机负载数据,根据式(5),计算物理机的平均负载度 $L_{average}$;

Step2:判断 ($L_{average} < L_{low}$), 如果否则转向 Step14;

Step3:根据式(7),对每个 $i \in PM$,求 C_i ;

Step4:根据 C_i 进行排序筛选,选择物理机 g ,使得 $C_g = \max\{C_i | i \in PM\}$;

Step5:更新 $C_i = 0$;

Step6:选取虚拟机 $j, j \rightarrow g \in M$,转向 Step7,如果不存在虚拟机 j ,则转向 Step13;

Step7:根据式(9)、(10),对于所有的物理机 $l \in PM$,计算 U_j^l ;

Step8:根据 U_j^l 进行排序筛选,选择物理机 h ,使得 $U_j^h = \max\{U_j^l | l \in PM\}$;

Step9:判断 ($U_j^i > U_{migration}$), 如果否则转向 Step11;

Step10:检查物理机 g 是否有足够资源提供给虚拟机 j ,如果有,则将虚拟机 j 从物理机 g 迁移至 h ,为 M 删除 $j \rightarrow g$,增加 $j \rightarrow h$,转向 Step6;如果没有则转向 Step11;

Step11:更新 $V_j^i = 0$;

Step12:在等待虚拟机 j ,执行完现有任务后,关闭虚拟机 j ,为 M 删除 $j \rightarrow g$,转向 Step6;

Step13:关闭物理机 g ,转向 Step16;

Step14:增加新的物理机 e ,加入物理机集群;

Step15:在物理机 e 上开辟新的虚拟机 b ,加入虚拟机集群,为 M 增加 $b \rightarrow e$;

Step16:转向 Step1.

4 实验结果分析

为评估算法的有效性,模拟了真实的云环境.在

本仿真实验中,实现了本文的基于云计算的负载均衡算法(Cloud Load Balancing, CLB).对比算法在物理机机层采用单限定值的虚拟机动态迁移的调度算法(Simple Threshold, ST),在虚拟机的任务分配时采用轮询调度(Round Robin, RR)算法.本文方法和对比算法主要通过以下三个指标进行评估:

(1)物理机集群和虚拟机集群的负载均衡度.全局的负载均衡度用公式(11)进行计算.

$$B = \sqrt{\frac{1}{n} \times \sum_{i=1}^n (L_i^t - L_{average})^2} \quad (11)$$

(2)任务平均响应时间,指从任务进入云计算系统到任务完成时的时间.

(3)资源利用率,使用平均负载 $L_{average}$ 进行计算.

仿真实验配置了 100 个相同的物理机,部署了 300 个性能相同的虚拟机.任务模拟 web 服务器请求,使用 98 年世界杯网站的日志访问数据^[7] 作为仿真数据.设定负载度模型中权值 $W = \{0.4, 0.2, 0.1, 0.3\}$.设定动态迁移算法的阈值 $L_{low} = 0.4$, $L_{high} = 0.9$, $U_{migration} = 0.8$.由仿真结果可以看出,随着时间的迁移,在任务量快速增加的 0~12pm 时间段里,本算法将物理机和虚拟机的负载均衡度均保持在较低的范围,而对于任务也均能及时响应.而在任务量减少的 4pm~12am 时间段里,本方法通过迁移或者关闭虚拟机,较之对比算法,仍然保持了较高的资源利用率.同时在物理机集群和虚拟机集群中仍能保持负载平衡,且能够及时响应任务.

从实验可以看出本方法在物理机层和虚拟机层,较之对比算法均保持了较好的负载均衡.同时本方法在任务响应时间上优于对比算法,有效提高了资源利用率.

5 结束语

本文针对虚拟机层和物理机层之间的链式负载关系,提出了双层控制的负载均衡框架.实验结果表明,本方法在实现了物理机和虚拟机层的负载均衡的同时,提高了资源利用率.

参考文献:

- [1] Foster I, Zhao Y, Raicu I, et al. Cloud computing and grid computing 360-degree compared: grid computing environments workshop [C]// GCE 2008. Austin, TX, United states, 2008.

(下转第 84 页)

$P_{Num_RT2}, \dots, P_{Num_RTi} \} \setminus \{P_{Num_RT1}, P_{Num_RT2}, \dots, P_{Num_RTj} \}$
进行均值计算,如式(9)、(10)所示:

$$\bar{P}_{RTi} = \frac{\sum_{i=1}^I P_{Num_RTi}}{I} \quad (9)$$

$$\bar{P}_{nRTj} = \frac{\sum_{j=1}^J P_{Num_nRTj}}{J} \quad (10)$$

式中, I, J 表示两个数列的项数。

结合历史预测值的均值以及前一次预测值对当前预测值进行优化,如式(11)、(12)所示,

$$^{\circ}P_{Num_RTi} = (1 - \beta)\bar{P}_{RTi} + \beta P_{Num_RT(i-1)} \quad (11)$$

$$^{\circ}P_{Num_nRTj} = (1 - \beta)\bar{P}_{nRTj} + \beta P_{Num_nRT(j-1)} \quad (12)$$

式中, β 表示控制参数,当预测值与均值相差较大时, β 取值区间为 $(0, 0.5)$, 反之取值区间设为 $(0.5, 1)$ 。通常 β 取经验值 0.3。

根据优化后的预测值,确定调度窗口 δ_n 内预留时隙的调整参数,如式(13)所示:

$$\partial = \frac{{}^{\circ}P_{Num_RTi}}{{}^{\circ}P_{Num_nRTj}} \quad (13)$$

最后根据调整参数通过式(14)确定当前调度窗口 δ_n 内预留时隙的大小:

$$Pt = \left\lceil \frac{\partial}{\partial + 1} \times 256 \right\rceil \quad (14)$$

通过历史预测值的均值和前一次的预测结果对当前预测值进行优化,使得当前预测值可以更好的描述队列中两种业务的到达情况,由此动态的为每个 δ_n 设置合适的预留时隙。

4 结束语

本文提出了一种基于区分服务的时隙分配机制(DSSA)。仿真结果表明 DSSA 可以有效的提高队

列的服务强度,降低业务的排队时间。

参考文献:

- [1] Akyildiz I F, Wang X, Wang W. Wireless mesh networks: A survey [J]. Computer Networks, 2005 (47):445-487.
- [2] Claudio Cicconetti, I F. Akyildiz and Luciano Lenzini, a bandwidth allocation algorithm for service differentiation in IEEE802.16 MESH networks[J]. IEEE/ACM Trans on Networking, 2009, 17(3):884-897.
- [3] 刘仙灵, 张勇, 戴超, 等. 基于预测和优先级的时隙调度策略[J]. 吉林大学学报: 信息科学版, 2009, 27(1): 12-16.
- [4] Jianhua He, Kun Yang, Ken Guild. Application of IEEE802.16 mesh networks as the backhaul of multi-hop cellular networks[J]. IEEE Communications Magazine, 2007, 45(9):82-91.
- [5] Mukul R, Singh P. An adaptive bandwidth request mechanism for qos enhancement in wimax real time communication[C]// Wireless and Optical Communication Networks, [s. l.]: IEEE, 2006:5-5.
- [6] 刘占军, 胡蓉, 余文魁, 等. IEEE802.16 Mesh 协同分布式模式下的延迟性能分析[J]. 重庆邮电大学学报: 自然科学版, 2010, 22(2):165-169.
- [7] 朱旭东, 常会友, 衣杨, 等. 基于调度窗口的弱硬实时系统改进 DBP 调度算法研究[J]. 安徽大学学报: 自然科学版, 2010, 34(4):70-75.

作者简介:

俞 浚 男, (1989—), 硕士研究生. 研究方向无线网状网 QoS 保障技术.

(上接第 80 页)

- [2] 刘之家. 一种基于云计算的负载均衡技术的研究[J]. 广西师范学院学报: 自然科学版, 2011(2):93-96.
- [3] 方义秋, 唐道红, 葛君伟. 云环境下基于虚拟机动态迁移的调度策略研究[J]. 微电子学与计算机, 2012, 29(4):45-48.
- [4] Jinhua H, Jianhua G, Guofei S, et al. A scheduling strategy on load balancing of virtual machine resources in cloud computing environment: parallel architectures, algorithms and programming[C]// 2010 Third International Symposium on. Dalian, China, IEEE Computer Society, 2010.
- [5] Zhang Bo, Gao Ji, Ai Jieqing. Cloud loading balance algorithm[C]// proceedings of the 2nd International Conference on Information Science and Engineering. Hangzhou, China, 2010.
- [6] 张鹏伟, 陈景霞, 贾小云. SOA 中一种带有预测功能的自适应负载均衡算法[J]. 微电子学与计算机, 2011, 28(11):174-177.
- [7] Martin Arlitt. A collection of 1.35 billion Web requests recorded at servers for the 1998 FIFA World Cup [EB/OL]. [2012-12-05]. <http://www.ircache.net/>

作者简介:

白延敏 女, (1982—), 硕士研究生, 讲师. 研究方向为嵌入式系统应用、云计算.

吕树红 女, (1980—), 硕士研究生, 讲师. 研究方向数据库技术、云计算.