一种面向云资源调度的热点迁移策略

刘文洁, 李战怀

(西北工业大学 计算机学院, 陕西 西安 710072)

摘 要:在云环境下的大数据中心,虚拟机数目和虚拟机的负载会随用户和应用的需求而经常变化,虚拟机需要进行动态资源调整,及时移除系统中的热点资源,从而达到整个系统的负载均衡。现有的方法主要采用启发式算法计算迁移调节方案,选择 CPU、内存、网络等资源的混合负载最小的机器作为迁移对象。此类方法由于没有区分资源类型,可能会导致从物理机上迁移了单一资源负载高而其他资源负载低的虚拟机。提出了一种云环境下,面向云资源调度的热点迁移方法,其特点是通过判断物理机上所承载虚拟机的 CPU 或内存各自的容量总和,来判断是否有热点发生,并根据对热点虚拟机迁移代价进行计算的结果来选择合适的目标物理机进行迁移。该方法可以区分不同资源种类的热点,能快速准确地找到热点虚拟机,并保证在迁移过程中代价最小,避免误迁移。

关 键 词:云环境;大数据;资源调度;虚拟机迁移;代价削减;多目标优化
中图分类号:TP311 **文献标志码**:A **文章编号**:1000-2758(2017)01-0138-05

云数据中心通常包含大量计算节点,当某节点由于负荷过高则会出现节点失效的情况,从而导致该物理机上的业务无法继续,造成巨大损失。这些负载过高的节点就是整个云数据中心的热点服务器。为了保证业务持续稳定地运行,需要移除该热点服务器上的某些虚拟机及其业务,降低其负载压力,并能保证被移除的业务能够在其他负载较低的服务器上继续运行,且不会形成新的热点。虚拟机热迁移是移除过程中经常使用的主要技术,但在现有的技术中,通常是基于整个集群的负载计算和资源的重新分配,其计算过程复杂,迁移代价较大,且容易造成不同资源的误迁移,在实际应用中并不理想。

本文在对云环境下虚拟机的资源特点以及现有资源调度算法进行分析研究的基础上,提出了一种基于虚拟机热点的迁移方法,其特点是将物理机上的虚拟机进行主从划分,每个物理机上的虚拟机由一个主虚拟机(master)和若干个从虚拟机(slave)组成,每个业务运行在 master 上。为了保证 master 由于负载过高突然崩溃时,业务的正常运转,每个 master 有若干个 slave 分布在其他的物理机上,并定期和 master 进行数据同步。每个物理机由节点控制器来收集每个虚拟机上的资源信息,如 CPU 和内存使用状况等。当节点控制器发现某物理机上存在热点时,则通知集群管理器。集群管理器根据各个物理机上的热点情况进行判断(是否超过资源阈值),选择合适的物理机将热点进行迁移,从而消除热点。此方法能够快速消除集群内部的热点,减少迁移代价,迅速在服务器集群内部实现负载均衡。

1 相关研究

目前,针对云数据中心的热点资源,通常采用 2 种方式进行资源迁移,一种是静态的服务器资源整合[1],另一种是系统动态的资源调整,即热迁移技术[2]。静态服务器整合的第一步是用虚拟机的历史负载和服务品质协议 SLA,确定虚拟机的资源配额;第二步用虚拟机的资源配额和物理服务器的容量来计算确定服务器整合配置,即每个虚拟机应该

收稿日期:2016-09-08
基金项目:国家“973”重点基础研究发展计划基金(2012CB316203)与国家自然科学基金(61303037)资助
作者简介:刘文洁(1976—),女,西北工业大学副教授,主要从事云计算、大数据处理及高可用性系统等研究。

在哪些物理服务器上运行。静态服务器整合的优点是实施简单、稳定可靠。但是,寻找优化的组合方案很困难。主要是因为应用的负载变化难以精确把握,虚拟机的资源配额只能是一个近似估计;而且即使确定了虚拟机的资源配额,找到最优方案也是个 NP 完全问题。

热迁移技术旨在动态调整虚拟机在物理服务器上的优化组合。由于动态迁移技术可以无缝地将虚拟机从失效的物理机转移到稳定的物理机上,快速部署新节点来代替失效的节点,实现故障隔离,因此云计算中心通常采用虚拟机动态调整的方式来调节负载。

WOOD T[3]设计了一个负载均衡调度器来处理虚拟机的热迁移问题,该调度器由集中控制器和监控器组成,监控器定期将虚拟机的 CPU、内存和网络 IO 使用统计数据发送给集中控制器,集中控制器根据各物理机和虚拟机的统计数据判断哪里发生了资源短缺,然后用启发式算法计算迁移调节方案,派发给监控器实施。该方法采用 CPU、内存和网络 3 类资源空闲率乘积的倒数 volume 来表示物理机或虚拟机的繁忙程度,为其寻找系统中最小 volume 的物理机作为迁移目标,直到所有热点被消除。该方法的优点在于将多种资源统一处理,可方便调度算法决策,但并不能表达哪个资源紧张,可能会导致从过载的物理机上迁移走一个内存大而 CPU 并不繁忙的虚拟机。

文献[4]是第一个用虚拟机迁移技术进行绿色计算的工作,但总的说来,该文献所提的算法还不能算是一个真正的调度算法,其原因是没有考虑如何提高系统性能的问题。针对不同云用户的需求以及不同的云环境,采用不同的负载均衡算法进行处理,例如静态环境下采用轮询调度算法;动态环境下可以采用最小权重连接算法(WLC)[6]、LBMM[7]算法。文章比较了不同的负载均衡算法所适用的场合,但并没有从具体虚拟机资源的角度对系统负载变化进行进一步分析,也没有对算法的性能进行评估,因此在实际应用中尚有所不足。

文献[8]提出了通过动态的优化服务器数量来支持绿色计算的方法,文章将虚拟机调度问题抽象成一个装箱问题,在真实的系统上,设计了一个资源调度算法,对虚拟机进行资源分配,该方法可以有效提高资源调度效率,具有良好的性能。但是该算法采用 VMM 作为热点资源的收集器,并没有对热点资源种类进行区分,也会对单一资源负载高而整体负载低的虚拟机进行迁移,存在一定的误迁移率。

因此,现有的资源调度算法和虚拟机热迁移技术在热点资源选择、性能指标评估等方面还有所欠缺,无法适应云计算中心的复杂性要求,必须针对目前的热点迁移技术进行改进,设计新的资源调度算法,研究新的热点迁移技术来改善迁移效率,以期提高虚拟机迁移的准确率,避免误迁移。

2　基于云资源调度的热点迁移方法

在云系统中,由于各个物理机上的所承受的业务不同,因此出现热点的可能性很大。针对热点,必须选择合适的算法进行热点移除。在热点移除过程中,需要判断资源种类,同时还要保证不产生新的热点。此外,针对云系统中用于业务恢复的 master-slave 的关系,还要保证同一个业务的 master 和 slave 不能被迁移到同一台物理机上,否则在物理机崩溃时,master 上的业务将无法继续正常运转,这是现有迁移方法中没有考虑到的问题。图 1 是一个云系统中热点迁移的示意图。

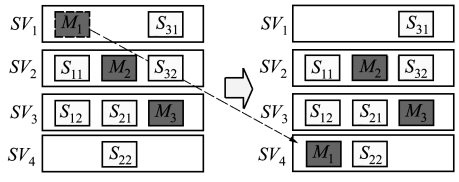


图 1　云系统中热点迁移示意图

在图 1 中,系统包含 4 个物理服务器 SV_1,$SV_2\cdots,SV_4$,$M_1,\cdots,M_3$ 是运行主要业务的 master 虚拟机,每个 master 虚拟机有 2 个备份的虚拟机,如 M_1 的备份机器是 S_{11} 和 S_{12}(slave 机器),分别运行在物理服务器 SV_2 和 SV_3 上。同样的,M_2 和 M_3 的 slave 机器也分布在和 master 机器不同的物理机上,从而保证在物理机故障发生时,不同机器上的 slave 可以快速接管 master 的业务,在不停机的情况下进行业务的无缝切换。因此,当热点出现时,如 SV_1 的 M_1 成为热点,需要将 M_1 迁移到其他物理机上,但目标物理机本身不能具有热点虚拟机,其次目标机器上不能有 M_1 的 slave 机器。

为了解决上述问题,我们针对云系统产生的热点,设计了热点移除算法,当发现某个物理机上存在

热点时(单个资源阈值越界或全部越界),节点控制器会通知集群管理器运行该算法,根据热点分布状况和迁移代价进行计算,选择合适的目标物理机进行热点迁移,如在图1中,通过计算,SV_2 和 SV_3 上存在 M_1 的备份机器,不满足迁移条件,只有 SV_4 可作为目标机器,因此可以将 M_1 迁移到 SV_4。

热点移除算法可以保证云系统业务的正常运转和系统内部的负载均衡。为了描述算法,我们对算法中用到的资源及相关概念进行形式化定义。

2.1 资源定义

定义1 (租户)对于一组指定配置的服务器,每个服务器上有若干承担业务的虚拟机,称为租户(tenants),若干个租户形成一组,分配一个管理员(master)。

如果集群中存在超过负载的服务器(例如服务器的CPU值、内存容量其中之一超过负荷,或者两者都超过负荷),那么解决问题的办法可以是从每个超过负载的服务器上将master租户迁移出去。

定义2 (热点服务器)集群系统包含 N 台物理服务器,$SV_1, SV_2, \cdots, SV_N$,每个租户 t_i^K 都有内存使用率 $t_i^K.mem$ 和CPU的使用率 $t_i^K.cpu$,每个服务器的CPU使用量和内存使用量是所有租户的CPU和内存的使用量的总和。即:$SV_i.mem = \sum_{k=1}^{K_i} t_i^K.mem$, $SV_i.cpu = \sum_{k=1}^{K_i} t_i^K.cpu$。服务器上CPU和内存的单位是不可变的,每个服务器上都有2个指标,一个内存阈值 $SV_i.memTh$ 和一个CPU阈值 $SV_i.cpuTh$,该阈值不能被超过,如果服务器上任何一个指标被超过,我们称该服务器为一个热点(Hotpot)。

由上述定义,我们可以看出每个服务器都有2个属性,内存容量 $SV_i.memCap$ 和CPU容量 $SV_i.cpuCap$,如果 $SV_i.memCap < 0$ 或 $SV_i.cpuCap < 0$,该服务器就变成了热点服务器。很明显,我们可以得到下面2个等式:

$$SV_i.memCap = SV_i.memTh - \sum_{k=1}^{k_i} t_i^k.mem, \text{并且}$$

$$SV_i.cpuCap = SV_i.cpuTh - \sum_{k=1}^{k_i} t_i^k.cpu$$

定义3 (资源属性)

1) 对于每个服务器 SV_i,具有如下属性:

(1) $SV_i.memCap$:服务器 SV_i 的内存剩余容量;

(2) $SV_i.cpuCap$:服务器 SV_i 的CPU剩余容量;

(3) $SV_i.memTh$:服务器 SV_i 的内存阈值,超过则变成热点;

(4) $SV_i.memTh$:服务器 SV_i 的CPU阈值,超过则变成热点;

(5) $SV_i.tenants$:服务器 SV_i 上租户的集合$\{t_i^1, \cdots, t_i^{ki}\}$。

2) 对于每个租户 t_i^K,都有 $t_i^K \in SV_i.tenants$,且每个租户具有如下属性:

(1) $t_i^K.tid$:租户的标识符;

(2) $t_i^K.ismaster$:租户是否为管理员的布尔值;

(3) $t_i^K.mem$:租户的内存使用率;

(4) $t_i^K.cpu$:租户的CPU使用率;

(5) $t_i^K.datasize$:租户的数据量的大小。

2.2 基本定理

为了设计下面的算法,我们先做一些假设,用 X 来表示任何一种资源(如CPU等)。对于 N 个数字 $\{x_1, x_2, \cdots, x_N\}$,设平均值 $\mu_x = \frac{1}{N}\sum_{i=1}^{N} x_i$,方差 $\sigma_x^2 = \frac{1}{N}\sum_{i=1}^{N}(xi - \mu x)^2$。我们可以得到如下的定理:

定理1 如果 $\sum_{i=1}^{N} xi$ 保持不变,当 $x_1 = x_2 = \cdots = x_N$ 时 σ_x^2 最小。

定理2 对于 $X = \{x_1, x_2, \cdots, x_N\}$,可以得到:$\sigma_x^2 = \frac{1}{N}\sum_{i=1}^{N} x_x^2 - \mu x^2$。

为了控制CPU和内存之间的平衡,引入变量 C_{mem} 和 C_{cpu},并令:

1) $C_{mem} = 1/\mu_{memCap}^2$,单个服务器上的内存校正参数;

2) $C_{cpu} = 1/\mu_{cpuCap}^2$,单个服务器上的CPU校正参数;

3) $\sigma^2 = C_{mem} \cdot \sigma_{memCap}^2 + C_{cpu} \cdot \sigma_{cpuCap}^2$,单个服务器上的CPU和内存的混合校正参数;

4) $t_{\text{cost}} = \frac{\sigma^2(t)}{t.dataSize}$,某个租户的迁移代价。

2.3 热点移除算法

算法输入:集群中的一组服务器 $SV_1, SV_2, \cdots, SV_N$,其中可能包含有热点服务器。

算法输出:迁移的策略集合 $S = \{(t, target)\}$,其中每个策略是一个二元组,包含一个要迁移的租户 t 和其预备迁移的目标服务器 $target$。

初始值:策略集合 $S = Null$;计数器 $i = 0$;

算法流程:

Step1　从服务器列表 $SV_1, SV_2, \cdots, SV_N$ 中取得第 i 个服务器,判断其内存容量 $SV_i.memCap$ 或 CPU 容量 $SV_i.cpuCap$ 是否小于 0,如果是,则该服务器为热点服务器;

Step2　从 1 到 N 开始遍历服务器,如果某 j 个服务器 $SV_j.memCap > 0$ 并且 $SV_j.cpuCap > 0$,且 $i \neq j$,则分别计算第 i 个服务器和第 j 个服务器的 CPU 容量差 $diff_{mem}$ 和内存容量差 $diff_{cpu}$,其中:

$$diff_{mem} = SV_i.memCap - SV_j.memCap,$$

$$diff_{cpu} = SV_i.cpuCap - SV_j.cpuCap$$

Step3　针对 SV_i 上的每个租户 t_k 遍历,如果 t_k 是 master 且 t_k 不属于 SV_j,且满足如下条件:

$$t_i^k.mem + SV_i.memCap > 0,$$

并且,$t_i^k.cpu + SV_i.cpuCap > 0$(迁移后热点资源负载正常)

并且,$SV_j.memCap - t_i^k.mem > 0$

并且, $SV_j.cpuCap - t_i^k.cpu > 0$(目标服务不成为新热点)

则计算包含迁移代价的 CPU 和内存的混合校准量 C:

$$c = [c_{mem} \cdot t_i^k.mem \cdot (diff_{mem} + t_i^k.mem) + c_{cpu} \cdot t_i^k.cpu \cdot (diff_{cpu} + t_i^k.cpu)] / t_i^k.dataSize$$

Step4　如果由 SV_i 上的某个租户 t_i^k 所计算出的 c 值在所有租户中最小,则该 t_i^k 是要迁移的对象,SV_j 是要迁移的目标服务器。将 t_i^k 和 SV_j 加入迁移策略集合 S,并执行下面操作:①SV_i 的租户列表中删除 t_i^k;②SV_j 的租户列表中加入 t_i^k;③SV_i 的内存和 CPU 容量减去 t_i^k 的内存和 CPU 容量;④SV_j 的内存和 CPU 容量加上 t_i^k 的内存和 *CPU* 容量;⑤ 返回 Step1。

Step5　如果 S 不为空,返回 S;否则返回失败。

3　算法分析

本文所设计的算法可以在云系统业务运转过程中根据资源负荷迅速的找到热点服务器,并计算集群内部的迁移代价,找到迁移代价最小的目标服务器,输出要迁移的热点和目标服务器,供集群管理器进行热点迁移,从而达到系统的负载均衡。为了说明算法的效果,我们通过一个例子来分析算法的执行过程。

假设云系统中运行着 5 台物理服务器 $SV_1, SV_2, \cdots, SV_5$,每台服务器上有 0~1 个 master 虚拟机(0 表示该物理机上还没有运行主业务)和 0~2 个 slave 虚拟机(0 表示该物理机上没有放置别的备份虚拟机),虚拟机分布图和 CPU 使用率(百分比)如图 2 所示:

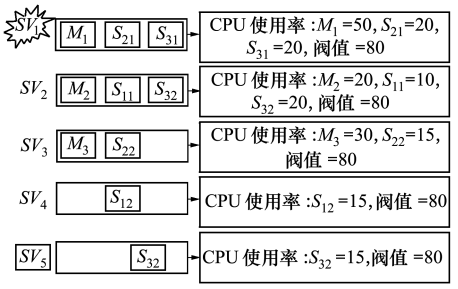


图 2　云系统中资源构成图及其 CPU 使用率

在图 2 中, M_i 表示第 i 台服务器上的 master 物理机($i = 1,2,3$),S_{ij} 表示第 i 台服务器的第 j 个备份 slave 虚拟机。根据热点移除算法,发现 SV_1 的 CPU 剩余容量 = 80 - 50 - 20 - 20 = - 10 < 0,所以 SV_1 成为热点服务器,现在需要在寻找目标服务器来将 SV_1 的 master 进行迁移。通过计算,发现 SV_4 和 SV_5 的 CPU 负荷符合要求,可以作为迁移目标。但是 SV_4 上存在 SV_1 的备份虚拟机 S_{12},不符合算法要求,因此选择 SV_5 作为目标服务器并计算迁移代价(这里符合条件的目标服务器只有 1 个,因此直接选择)。由于只有 1 个热点,所以返回迁移策略集合 S 为 $\{M_1, SV_5\}$。迁移后 SV_1 的 CPU 使用率为 20 + 20 = 40,SV_5 为 15 + 50 = 65,无新热点形成,系统负载均衡。

该算法由于考虑了主业务虚拟机和备份虚拟机不能同时放置在同台物理机的要求,可快速筛选目标服务器,减少计算过程。并根据单一资源和混合资源阈值来计算热点,因此能够区分负载资源的类型,减少误迁移。算法计算可用资源的迁移代价,因此保证了在迁移过程中的代价最小。

4　结　论

为了准确定位云环境下集群服务器中热点资源,并根据资源使用状况灵活地选择合适的服务器对热点进行迁移,本文提出了一种云系统中,面向资源调度的热点迁移策略,可以检测云环境下集群服务器中的热点,通过计算资源阈值和迁移代价来寻

找适合迁移的目标服务器,快速找到迁移目标并移除热点。算法按照同台物理机上,不能同时存在一个业务的主虚拟机和备用虚拟机的规则计算迁移目标,避免了物理机崩溃时业务的中断;且判断迁移后的资源阈值是否超标,以保证迁移后不会形成新热点。针对待选目标服务器,算法根据待迁移热点的数据量来计算源服务器到目标服务器的迁移代价,保证了迁移过程在整个集群可用服务器中的代价最小。该策略已经应用在了某国际合作项目的云计算系统中,实践表明,该迁移策略能有效移除服务器集群中的热点,可有效实现系统资源的负载均衡。

参考文献:

[1] Stillwell M, Schanzenbach D, Vivien F, et al. Resource Allocation Algorithms for Virtualized Service Hosting Platforms[J]. Journal of Parallel and Distributed Computing,2010,70(9): 962-974

[2] LD D B, Krishna P V. Honey Bee Behavior Inspired Load Balancing of Tasks in Cloud Computing Environments[J]. Applied Soft Computing, 2013, 13(5): 2292-2303

[3] Wood T,Shenoy P, Venkataramani A. Black-Box and Gray-Box Strategies for Virtual Machine Migration[C]//Proceedings of the ACM Symposium on Networked Systems Design and Implementation,2007: 229-242

[4] Nathuji R, Schwan K. VirtualPower: Coordinated Power Management in Virtualized Enterprise Systems[C]//Proceedings of 21th ACM SIGOPS Symposium on Operating Systems Principles, 2007: 265-278

[5] Mahajan K, Makroo A, Dahiya D. Round Robin with Server Affinity: a VM Load Balancing Algorithm for Cloud Based Infrastructure[J]. Journal of Information Processing Systems, 2013, 9(3): 379-394

[6] Chen H, Wang F, Helian N, et al. User-Priority Guided Min-Min Scheduling Algorithm for Load Balancing in Cloud Computing [C]//2013 National Conference on Parallel Computing Technologies, 2013: 1-8

[7] Wang S C, Yan K Q, Liao W P. Wang S S. Towards a Load Balancing in a Three Level Cloud Computing Network[C]//Proceedings of 3rd International Conference on Computer Science and Information Technology, 2001: 108-113

[8] Song W, Xiao Z, Chen Q, et al. Adaptive Resource Provisioning for the Cloud Using Online Bin Packing[J]. IEEE Trans on Computers, 2014, 63(11): 2647-2660

A Hot-Spot Migration Strategy for Resource Scheduling in Cloud Environment

Liu Wenjie, Li Zhanhuai

(School of Computer Science, Northwestern Polytechnical University, Xi'an 710072, China)

Abstract: In the big data center of cloud environment, the number and load balance of virtual machines will change frequently. The resources of virtual machines should be adjusted dynamically to remove the hot-spot therefore achieve the load balance in the system. Current techniques often use heuristic methods to compute the migration measures, which select the machine whose CPU, memory, network and other resources are combined to achieve the minimum workload. As the kind of method does not distinguish the resource type, it may cause to migrate a machine whose CPU's workload is the highest and other resources' workload is very low. This paper proposed a hot-spot migration method oriented cloud resources. The method judges a hot-spot by checking the sum of all the resources workload of a virtual machine, and selects a proper physical server to migrate by computing migration cost. This kind of method can distinguish hot-spot of different resources and find the host virtual machine quickly. It can also assure the minimum migration cost and avoid the fault migration.

Keywords: cloud environment; big data; resource scheduling; virtual machine migration; cost reduction; multi-objective optimization