基于 Eucalyptus 网站云的资源调度①

刘子锋, 范冰冰, 郑伟平

(华南师范大学 计算机学院, 广州 510631)

摘 要: 网站云是一种多租户的云部署架构. 研究了基于 Eucalyptus 实现的网站云资源综合调度策略, 提出了以 HTTP 请求为细粒度任务的贪婪调度算法, 并引入了带权机器学习算法模型, 使系统能主动感知请求高峰并进行负载均衡. 实验证明达到了较理想效果.

关键词: 网站云; Eucalyptus; 调度策略; 机器学习

Research of Resources Scheduling of Eucalyptus-based Website Cloud

LIU Zi-Feng, FAN Bing-Bing, ZHENG Wei-Ping

(School of Computer, South China Normal University, Guangzhou 510631, China)

Abstract: Website cloud is a multi-tenant cloud deployment architecture. This thesis studies the resources scheduling policy of the Eucalyptus-based website cloud, and the HTTP request as fine-grained task for the greedy scheduling algorithm. It also discusses the model of weighted machine learning algorithms, that the system can take the initiative to perceive the request peak and load balancing. Experiments show that the ideal.

Key words: website cloud; Eucalyptus; scheduling policy; machine learning

网站云是云计算应用中的一种, 以网站为应用目标的云计算系统. 传统的主机托管、购置服务器等的网站构建方式难以实现资源动态分配、伸缩管理以及负载均衡的要求, 因此, 结合云计算的网站云在资源调度上能够解决传统网站的上述难题. 本文的目标是使用 Eucalyptus[1,2]作为构建网站云的基础平台架构, 然而, Eucalyptus 使用传统的调度方式, 其缺乏 VM 实例调度和负载均衡在网站云的应用实现中显然存在不足. 为此, 本文针对 Eucalyptus 网站云建立了一种综合的资源调度模型, 旨在解决网站云在 Eucalyptus 实现上面临的调度问题.

目前云计算平台的资源调度, 一般由两种基本方式构成: 一是单个任务独占 VM 实例, 每个任务会在一定的时间内完成; 二是 VM 实例能够同时运行多个任务, 而每个任务不考虑时间. 针对第一种情况有两种调度策略: MWVF(Most Wanted VM First, 紧急优先策略)[3]和 DP(Demand Proportion, 按需分配策略)[3]. MWVF 会将某个拥有最多等待处理任务数的 VM 实例优先执行, 这种策略并不关注其它实例运行状况. DP 策略主要通过调度器监视所有 VM 实例运行状况, 并随需求比例进行强制变更, 此策略旨在平衡对所有类型任务的处理需求和资源供给. 基于能量消耗的调度算法[4]则主要针对第二种情况, 基于给定数量的机器资源以完成尽可能多的调度任务的思想, 其核心是应用了近似于装箱问题的贪婪算法并结合 VM 实例的在线迁移, 是贪婪算法在实际应用场景下的改进型.

我们将用户单次的 HTTP 请求视作一个运行在 VM 上的任务, 包括“请求-执行-响应”整个过程. 由于系统在处理用户请求时允许并发执行, 而用于单任务串行处理的 MWVF 和 DP 策略并不适用于网站云环境, 因此, 本文的研究重点将在第二种调度方式上.

为了确保网站云在随机访问中遭遇高峰时的稳定性, 本文率先在 Eucalyptus 网站云架构中引入了结合机器学习算法的调度策略, 能够主动、有效的应对系统遇到的访问高峰.

① 收稿时间:2012-02-22;收到修改稿时间:2012-04-04

1 Eucalyptus平台和网站云架构

Eucalyptus 是一种开源的云基础架构，通过计算集群实现弹性的、高效用的云计算平台[5,6]. 作为 IaaS 的开源实现，兼容 AWS(Amazon Web Services)[7]的 API. Eucalyptus 系统由一个云管理平台和五个高层组件构成，五个组件分别为: CLC(Cloud Controller，云控制器)、CC(Cluster Controller，集群控制器)、Walrus、SC(Storage Controller，存储控制器)、NC(Node Controller，节点控制器).

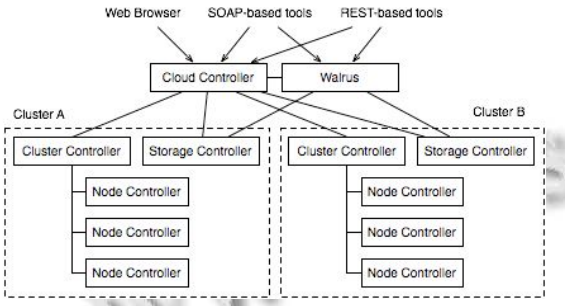

图 1 Eucalyptus 结构图

CLC: 最主要的控制器，负责管理整个 Eucalyptus 系统. CC: 管理虚拟实例网络以及连接该网络的 NC，能够访问公共或私有网络，负责收集其所在集群中 NC 的状态信息. Walrus: 一个提供存储服务管理的高层组件，管理对 Eucalyptus 内的存储服务的访问. SC: 实现块访问的网络存储，并能够与各种第三方存储系统通信. NC: 用于控制主机操作系统和相应的 hypervisor(Xen 和 KVM)，直接管理 VM 实例，包括创建、挂起、关闭及清除等[8].

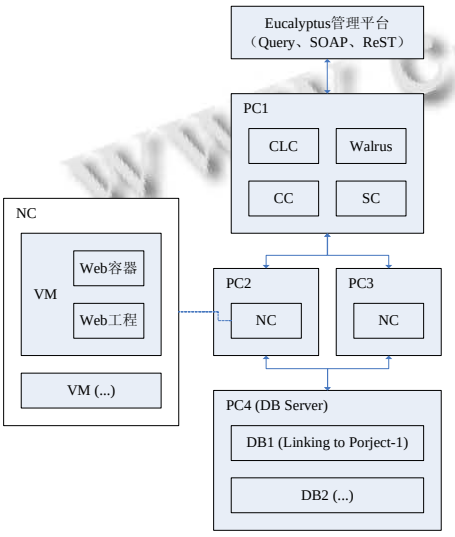

图 2 基于 Eucalyptus 的网站云基础结构

基于 Eucalyptus 的网站云基础架构[9]如图 2 所示，NC 节点上运行的 VM 实例包含 Web 容器和工程，前端节点负责系统控制和资源调度，后端节点提供 Web 服务器能力. 该平台能够通过镜像挂载的方式快速部署已有的 Web 项目，使得传统的网站向网站云系统迁移变得简单可行.

Eucalyptus 系统在资源使用上运用了最简单的轮询策略，有利于资源的快速查找和获取，但并不主动探测资源使用状况，而且没有整合 VM 实例调度策略和负载均衡，这对于网站云这样一个资源使用动态性较强的系统而言是难以接受的，因此，使用一种新的符合网站云需求的调度策略及模型变得相当重要.

2 网站云资源调度模型

本文提出的资源调度模型主要包含两部分: 1) 实现 VM 实例高利用率的贪婪算法调度模型; 2) 后台实现机器学习算法与负载均衡的调度模型. 图 3 是满足网站云需求的资源调度模型框架.

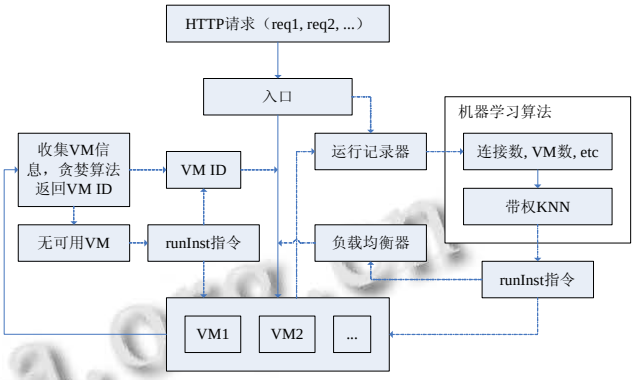

图 3 资源调度模型示意图

平台入口处拦截所有 HTTP 请求，对请求的权值进行评估，同时，VM 监视器监控所有正在运行的 VM 状态，基于请求对活动的 VM 资源消耗最多的条件，请求权值作为输入，通过贪婪算法可以直接输出目标 VM 的 ID，并告诉入口已经得到最合适处理的 VM 并转发请求至该实例.

机器学习算法不直接干预系统运行，但入口处对 HTTP 请求拦截的信息以及活动的 VM 实例运行数据都将被运行记录器记录下来，这些数据会被作为后台运行的机器学习算法的样本进行采集和计算. 如无特殊情况，算法输出的结果会持续一种稳定的状态，不会对系统作出干预；一旦预测到特殊情况如即将到来

的访问高峰, 调度系统就会发出创建 VM 的指令, 而不管当前 VM 实例池的运行状态, 同时会触发负载均衡器, 负载均衡器将会选择资源占用最少的 VM 实例作为处理请求的目标 VM 返回给入口.

2.1 基于 HTTP 的资源调度策略和算法

Eucalyptus 系统给定的 NC 节点, VM 实例可运行数是确定的, 那么, 如何提高 VM 实例运行时的利用率对 NC 至关重要[10]. 在网站云系统中, 以 HTTP 连接作为任务进行划分, 对于 VM 实例资源属于细粒度的任务, 任务粒度越细, 其利用率越高.

系统在入口处对每一个 HTTP 连接进行权值评估, 依据“请求-执行-响应”过程对资源消耗估计, 可记录连接权值, 如执行更新数据请求的权值一般要高于查询请求的, 多条记录查询请求的权值则要高于单条查询的. 处理结束的连接的真实资源消耗将被记录, 并对后续的相似连接进行权值修正, 这种循环修正的方法可不断提高连接的权值准确度. 相似连接的判定标准: 1、具有相同的域名和请求页面; 2、具有相同的参数表但不一样的参数值. 例如: http://xxx.com/queryPerson.jsp?id=1 和 http://xxx.com/queryPerson.jsp?id =2 即属相似连接. 对 HTTP 连接进行 VM 调度前, 应用贪婪算法针对连接权值将获得当前最优的 VM 实例 ID. 贪婪算法实现方式如下:

第 i 个 VM 的空闲资源数: Free(i)　i=1,2,…,n

n: 当前开启的 VM 个数

当前任务的权值(模拟所需要消耗的资源): Tsk

当前调度的 VM 的索引值:

$$\mathrm{IDX} = f(i) = \begin{cases} i & if\ \mathrm{Free}(i) \geq \mathrm{Tsk} \\ f(i+1) & if\ \mathrm{Free}(i) < \mathrm{Tsk} \end{cases}$$

假设一个 VM 实例的处理能力为 100, 那么各个请求的权值范围可设为 1-10 不等, 这样的假设符合 VM 实例能够同时处理 10-100 个 HTTP 连接的实际情况. 基于某个时间段出现的请求类型随机的情况, 单个 VM 顺序接收一段随机权值的请求, 当超越当前 VM 处理能力时, 即启动新 VM 并切换至该实例.

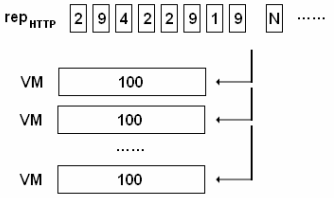

图 4　VM 顺序接收 HTTP 请求任务

根据以上规则进行模拟测试, 每次生成 100 个随机请求序列, 使用顺序方式和贪婪算法对其模拟调度, 得如下结果:

表 1　顺序方法和贪婪算法的比较

	顺序方法	贪婪算法
虚拟机 VM 数	7	6
总利用率均值	85%	99.17%
有效利用率均值	95.17%	100%

有效利用率均值是指在去除利用率一般较低的最后启动的 VM 的情况下的平均值, 更贴近普通运行的 VM 的利用率均值情况. 执行 10 次上述的测试, 对有效利用率均值进行统计, 可得下表:

表 2　进行 10 次测试的比较

次数	顺序方法	贪婪算法
1	95.17%	100%
2	96%	100%
3	94.2%	99.8%
4	95.8%	100%
5	97.2%	99.6%
6	98.4%	99.8%
7	98.4%	100%
8	96.6%	100%
9	97.2%	99.8%
10	98.2%	100%
平均值	96.72%	99.9%

实验结果表明, 对于细粒度的任务调度, VM 能够达到较高的利用率; 应用贪婪算法能够进一步提升利用率, 使其接近或达到 100%.

2.2 网站云负载预测策略和算法

在云计算平台通常调度和负载均衡策略中, 请求高峰的处理都是一种被动的方式. 为了实现智能资源调度, 我们引入了改进的机器学习算法带权 KNN (K-Nearest Neighbor, K 最近邻分类算法)[11].

KNN 的核心思想: 如果一个样本在特征空间中的 K 个最邻近的样本中的大多数属于某一类别, 则该样本也属于这个类别. 对于网站云, 某个时刻的 HTTP 连接数作为样本将被记录, 其特征空间为可能出现的连接数, 也就是离散的随机整数. 以 24 小时作为循环记录的周期, 一天中的某个被记录的时刻作为一个样本采集点, 前 6 天的同一时刻记录的数据

将被一同作为一次样本训练集. 为了能够对高峰访问到达所产生的变化较为准确地预知, 使用带权 KNN 方法对近距离的时间点样本设置高权值, 权值与样本距离成反比.

Time : 24h →									
Record :				R_{II}					Weight
Day1 :	...	1200	1230	1250	1240	1200	1160	1170 ...	w1
Day2 :	...	1320	1330	1360	1350	1360	1340	1330 ...	w2
Day3 :	...	1610	1620	1650	1630	1620	1600	1560 ...	w3
Day4 :	...	1560	1550	1580	1610	1580	1560	1540 ...	w4
Day5 :	...	1720	1720	1730	1770	1780	1750	1740 1710 ...	w5
Day6 :	...	1940	1990	2010	2020	2060	2030	2000 1960 ...	w6
Day7 :	...	2090	2110	2160	2150	2120	2110	2080 2050 ...	w7
Day8 :					v_{knn}				

图 5 实现带权 KNN 算法的模拟数据图

上图中的模拟数据展示了获取 7 天中的同一时刻的数据作为样本输入, 并对每一天设置相应的权值. 样本中 24 小时的第 N 条记录 RN 第一天的记录值为 v1, 对应的权值为 w1, 如此类推. 第 8 天的学习结果为:

$$v_{knn}=\frac{\sum_{i=1}^{7} v_i \cdot w_i}{\sum_{i=1}^{7} w_i}$$

图 5 中给出的数据, KNN 计算结果约为 1683, 使用带权 KNN 计算所得的结果为 1992. 显然, 由记录 RN 可知, 其访问量有上升趋势, 可以推断第 8 天该时刻的访问量应该是较高的, 带权 KNN 所得的结果应该更符合实际可能发生的情况. 该算法类似于期望的计算方法, 对于固定的样本数, 带权 KNN 的时间及空间复杂度只是 O(1), 计算实现简单可行.

由一个随机数模型产生一组模拟 18 天中同一时刻的连接数, 该模型产生的数据符合以下规则: 正常情况下, 一般连接数在 1000-2000 区间; 在第 10-13 天这四天会出现访问高峰, 连接数将会达到 4000-5000 之间. 由第 8 天开始, 运用带权 KNN 算法计算预测结果, 并与实际数据比较, 前 7 天是一个样本采集区间. 测试结果中预测值由第 8 天正式开始计算, 在应对第 10-13 天的高峰访问时, 带权 KNN 经过第 9、10 天的结果作为过渡期, 其预测值在第 11-13 天的时候非常接近实际值, 即预测成功. 高峰访问过后, 即第 15 天及其后恢复正常访问量, 带权 KNN 经过第 14、15 天的过渡期, 其预测值也随之下降, 并且接近实际值. 使用机器学习算法, 大概会有两天左右的延迟过渡期, 往后就能够较准确地预测高峰访问的到来.

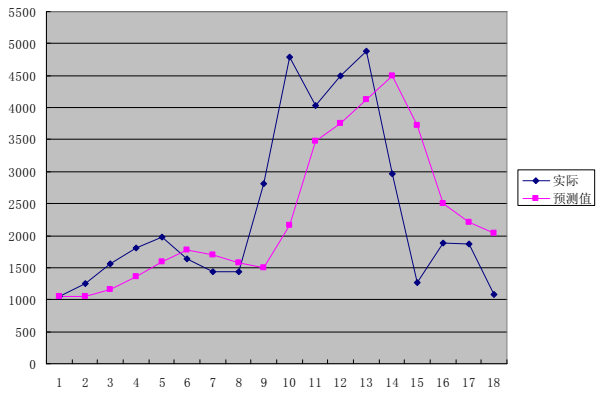

图 6 带权 KNN 算法的测试结果

2.3 高利用率与负载均衡的协同

在整个策略模型中, 贪婪算法处于常规性的运行状态, 保证 VM 的高利用率. 负载均衡器则处于等待状态, 接收来自带权 KNN 的结果指令. 负载均衡使用以下方式实现:

① 空闲资源最多的 VM 索引值: IDX = index (max(Free(1), Free(2), …, Free(n)))

② 当前接收任务调度的虚拟机: VM_{IDX}

数据采集以及带权 KNN 运算均在后台进行, 当预测访问高峰即将降临并启动 VM 后, 贪婪算法会暂停执行, 负责向入口发送目标 VM 的 ID 的工作将会由负载均衡器接管. 均衡调度将会持续一段时间 T, 该段时间过后 VM 平均利用率低于 80%便会终止负载均衡器, 使其返回等待状态, 贪婪调度重新回到常规性运行, 同时等待下一个 KNN 高峰预测结果的来临.

贪婪算法和机器学习的负载均衡协同方法的核心伪代码如下所示.

```
If high access rate Then
   suspend greedy scheduling;
   start new VM or VMs;
   query and return ID of the most free VM;
   forward request to the VM(ID);
   last for time T;
   If average usage rate bellow 80% Then
      resume greedy scheduling;
      stop VM or VMs;
   End
else
```

query and return ID of VM with greedy scheduling;

If ID is null Then

start new VM and return ID;

End

forward request to the VM(ID);

End

利用 LoadRunner 对网站云平台中的某个网站实例进行高峰访问实验，根据访问数的变化记录下 VM 数及平均利用率.

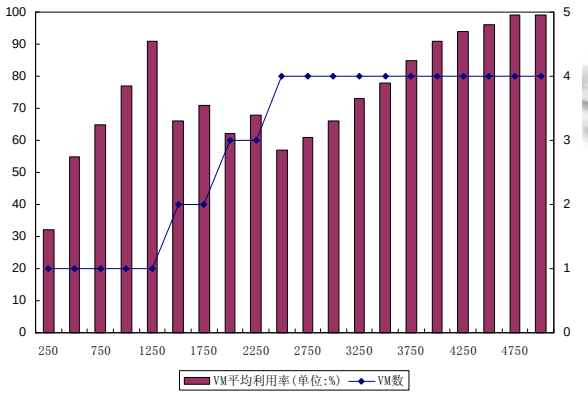

图 7 贪婪调度到机器学习负载均衡的变化实验

左侧 Y 轴为 VM 平均利用率，以柱状图表示；右侧 Y 轴为活动的 VM 数，以点线图表示; X 轴表示访问量. 由实验结果可知，访问量上升到 1500 的时候带权 KNN 根据样本数据已经预测到可能的高峰访问并新启动了一个 VM，随着访问量的不断增加并达到 2500 时，系统启动到 4 个 VM 进行处理，而尽管当时 VM 平均利用率并不高. 实验结果表明，机器学习算法能够在正式访问高峰到达之前主动启动 VM 实例等待处理，并接替贪婪调度，达到预期目标. 整个资源调度模型和策略是可行、有效的.

3 结语

本文对网站云的资源综合调度策略模型研究，将 HTTP 请求作为任务进行细粒度划分，并通过贪婪算法实现了 VM 高利用率的调度策略，基于机器学习算法实现对访问高峰的预测和负载均衡调度的协同实现. 在搭建基于 Eucalyptus 平台的网站云实验中，通过实验室环境的模拟测试，取得了较好的实际效果. 文章中基于机器学习算法对访问高峰的预测和负载均衡调度实现具有创新性，也符合一般云计算多租户 SaaS 平台的智能负载处理需求.

参考文献

1 Eucalyptus Cloud. http://open.eucalyptus.com/.

2 Nurmi D, Wolski R, Grzegorczyk C, et al. Eucalyptus: A Technical Report on an Elastic Utility Computing Architecture Linking Your Programs to Useful Systems. UCSB Computer Science Technical Report Number 2008-10, 2008.

3 Kijsipongse E, Vannarat S. Autonomic Resource Provisioning in Rocks Clusters using Eucalyptus Cloud Computing. ACM MEDES'10 Oct 26–29,2010:61–66.

4 赵玉艳,赵生慧.Eucalyptus 中基于能量消耗的调度算法研究.滁州学院学报,2011,13(2):18–20.

5 开源中国社区.http://www.oschina.net/p/eucalyptus.

6 高宏卿,张弛,何婧.基于 Eucalyptus 的教育知识服务体系模型研究.现代教育技术,2010,20(12):121–124.

7 Amazon Web Services. http://aws.amazon.com/.

8 陈仲.基于 EUCALYPTUS 的虚拟云改进研究.现代情报,2011,31(5):152–157.

9 Zhou JT, Zheng S, Jing DL, et al. An Approach of Creative Application Evolution on Cloud Computing Platform. ACM SAC'11 March 21–25,2011:54–58.

10 沈腾飞.Eucalyptus 虚拟机管理系统的研究与实现[硕士学位论文].北京:北京邮电大学,2011.

11 张顺,张化祥.用于多标记学习的 K 近邻改进算法.计算机应用研究,2011,28(12):4445,4446,4450.