虚拟环境下虚拟机应用性能建模①

黎丰泽[1,2], 杨 达[1], 周 鹏[1], 武延军[1]

[1](中国科学院软件研究所 基础软件国家工程研究中心, 北京 100190)
[2](中国科学院大学, 北京 100190)

摘 要: 得益于虚拟化技术的成熟发展, 当下私有云和公有云数据中心已经越来越多的出现在企业、学校和研究机构当中. 相对于物理机, 虚拟机拥有更好的迁移性、可扩展性和相对低廉的购入与维护成本, 所以越来越多的中小创业者倾向于购买虚拟机部署服务. 对于云服务提供者来说, 如何在满足 SLA 情况下对云环境下或者运行于同一物理资源池上的虚拟机合理分配资源从而实现硬件资源池最大化利用变得越来越重要. 本文分析了影响虚拟环境下虚拟机应用性能的关键参数, 并证明了虚拟机应用性能与硬件资源之间存在着复杂的非线性关系, 通过一种 SVD 特征拓展+非线性模型的方法对运行于同一物理资源池上的虚拟机应用性能进行建模研究, 实验表明该模型有较好的效果, 并且平均预测误差可以达到 12%左右.

关键词: 虚拟化; 性能建模; 硬件资源分配; SVD

Modeling Application Performance in a Virtualized Environment

LI Feng-Ze[1,2], YANG Da[1], ZHOU Peng[1] , Wu Yan-Jun[1]

[1](Institute of Software, Chinese Academy of Sciences, Beijing 100190, China)
[2](University of Chinese Academy of Science, Beijing 100190, China)

Abstract: Thanks to the development of virtualization technology, the private and public cloud data center has more and more appeared in enterprises, schools and research institutions. Relative to the physical machine, the virtual machine has better mobility,scalability,and relatively cheap to buy and maintenance, so more and more small and medium-sized entrepreneurs tend to buy virtual machine deployment services. So for cloud service providers, how to allocate physical resources to the VMs in one cloud environment and maximize the utilization of hardware resources pool meeting SLA is becoming more and more important. In this paper, we analyzed the key parameters that affect the VMs' application performance, we used a method of SVD + non-linear model in the application performance modeling and a good results were obtained in the experiment that makes the average prediction error can reach about 12%.

Key words: virtualization; performance modeling; hardware resource allocation; SVD

1 引言

随着虚拟化技术的普及, 物理机的硬件资源得到了更有效的利用. 得益于虚拟化技术的成熟发展, 当下私有云和公有云数据中心已经越来越多的出现在企业、学校和研究机构当中. 相对于物理机而言, 虚拟机拥有更好的迁移性、可扩展性和相对低廉的购入与维护成本, 所以人们也越来越倾向于购买虚拟机. 因此对于云服务提供者和用户来说, 如何对云环境下或者基于同一物理资源池上的虚拟机合理的分配和最大化的利用硬件资源变得越来越重要.

云服务提供者(如亚马逊的 EC2[1])当前提供的虚拟机硬件资源配置策略是一种粗粒度的配置策略. 为了简单, 其向用户提供固定硬件资源配置和计算能力的计算单元, 如提供微型、小型、大型与极大型的硬

① 基金项目:国家自然科学基金(91318301,61432001);国家高技术研究发展计划(863)(2012AA011206);中国科学院知识创新工程重要方向项目(KGCX2-YW-174);中国科学院战略性科技先导专项课题(XDA06010600)

收稿时间:2014-12-30;收到修改稿时间:2015-03-12

件资源配置实例, 每一种类型的实例映射为不同大小的 CPU、内存和其他资源, 并根据用户所选择的类型进行收费. 这种硬件分配和收费策略存在明显的缺点: 对用户来说, 为了避免虚拟机使用过程中的性能问题, 其可能会购买超额的配置, 不仅使资源未能充分利用, 同时增加了预算的成本; 再者, 对于这种成比例的硬件资源配置策略, 用户可能得到一个使得性能次优的实例. 比如对于一个运行着内存密集型应用的实例, 其 CPU 的使用量上升并不一定是其 CPU 分配不充分, 有可能是因为其内存分配不充分而导致 CPU 频繁的进行垃圾回收; 又比如一个运行着 I/O 密集型应用的实例, 其 I/O 带宽的使用量上升并不一定是 I/O 带宽分配不充分, 也有可能是内存分配不充分导致数据在内存与硬盘间频繁的写入写出. 而对于目的在于收益最大化的云服务提供商来说, 其粗粒度的硬件资源分配策略不仅不能提供给用户最优的性能体验, 而且超额配置的资源没有得到有效的利用, 那些资源本可以提供给共享同一物理资源池上的其他虚拟机, 因此也没能达到最大的盈利.

在这种情况下, 对云环境下或共享同一物理资源池上的虚拟机应用性能进行建模就变得有意义了. 对虚拟机上应用的性能进行精确的建模能够使用户合理的分配和购买自己虚拟机的硬件资源, 避免资源的浪费和不必要的花销, 同时也能够使虚拟机运行在一个性能相对较优的资源配置中. 对于云服务提供者来说, 虚拟机应用性能建模不仅能够使其给用户提供合理的资源分配建议, 最大化利用物理硬件资源池, 使用户达到更好的用户体验, 而且还提供了一种新的细粒度的收费模型标准.

本文将通过对虚拟环境下小型虚拟机集群应用性能进行建模, 探索虚拟架构下影响虚拟机应用性能的关键参数, 同时通过比较传统典型建模方法在虚拟机应用性能建模问题上的优劣, 从而提出一种适用于该问题的建模方案.

2 相关工作

当前对虚拟环境下虚拟机应用性能建模的研究工作主要分为两类: 第一类是基于排队和控制理论的建模方法; 第二类是基于机器学习的建模方法.

基于排队和控制理论的建模方法: Doyle 等人通过基于分析模型的排队理论在不同负载和资源分配下预测网络服务的响应时间[2]; Bennani 等人提出了多类排队网络来预测在线虚拟工作负载的响应时间和吞吐量[3]. 还有人应用控制理论来调整虚拟机的资源分配来达到想要的应用性能, 比如有人应用一阶自回归的模型来管理网络服务器的 CPU 分配[4,5]; 有人应用多输入多输出(MIMO)模型来分配 CPU 资源, 同时修正基于同一物理资源上虚拟机间的性能干扰[6]. 但是排序和控制理论建立的线性模型不能准确的捕获虚拟环境下虚拟机应用性能的非线性行为. 而本文将致力于证明虚拟机应用性能和硬件资源存在复杂的非线性关系, 同时将提出建模解决方案.

基于机器学习的建模方法: Bodik 等人将贝叶斯分类应用于数据中心的自动性能危机诊断及修复, 其研究专注于识别和预测是否特定的资源使用将会导致违背 SLA 协定, 但是并没有提出怎样的资源分配能够避免今后违背 SLA 协定[7]; Xu 等人提出的将混淆逻辑的建模方法应用到虚拟环境下网络应用的资源需求预测上[8]; Rao 等人提出的 VCONF 项目通过强化学习结合 ANN 的方法对系统范围内的 VM 性能进行建模, 从而通过模型自动的调整 CPU 和内存资源来使系统范围内的 VM 性能达到最优[9]. 以上几篇文章的解决方案主要针对于 CPU, 本课题将探索证明更多的硬件资源关键参数(比如内存, I/O 竞争等)在虚拟机应用性能建模上的有效性.

3 参数选择

选择合适的参数来调节虚拟机的硬件资源至关重要. 在参数的选择上, 我们认为应该注意以下几点: ①所选参数必须能够直接或间接的映射或反映已知资源的使用情况, 同时参数必须是容易观察和控制的. ②可以通过所选参数调节虚拟机应用性能. ③基于同一物理资源的虚拟环境下, 不同虚拟机中的应用性能可能受到资源竞争的影响, 这点应该被考虑和刻画. ④硬件资源之间的相互影响的因素应该被考虑. ⑤应该需要确立能够捕捉应用性能的最小参数集合.

我们选择 Xenserver6.2[10]作为实验的虚拟化环境, 并用 Filebench[11]中五种典型的工作负载(webserver, fileserver, varmail,OLTP,webproxy)来分别模拟五种典型的企业级应用程序, 性能参数通过操作/秒(operations/sec)来衡量. 基于上述选择参数的考虑, 我们分别探索了 cpu 时间周期权重(cpu_time)、虚拟 cpu

的数量(vcpu)、内存(memory)以及 I/O 竞争(I/O contention)这四种硬件资源单独对虚拟机应用性能的影响. 实验中应用控制变量法来观察五种应用模拟程序在不同硬件资源变动下的性能情况, 同时在对某种参数进行实验的时候, 会将其他参数值设置为较大值或者零, 以防止其他参数成为性能影响的瓶颈.

cpu 时间周期权重: Xenserver 通过 VCPU-params 参数中 weight 参数来控制 cpu 时间周期权重, 它表示的是一个虚拟机统计上可以占有处理器时钟周期的部分. 一个 cpu 时间周期权重为 512 的虚拟机所占有的处理器时钟周期是一个 cpu 时间周期权重为 256 的虚拟机的两倍. 权重范围在 1-65535 之间浮动, Xenserver 对于每个虚拟机的默认值为 256. cpu 时间周期权重对虚拟机应用性能的影响见图 1.

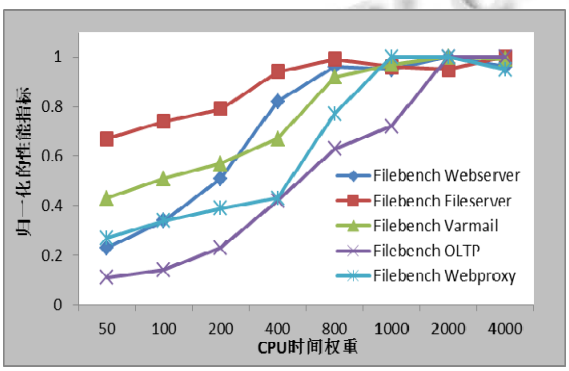

图 1 cpu 时间周期权重对虚拟机应用性能的影响

虚拟 cpu 数量: 该参数决定了一个虚拟机可以使用的物理 cpu 的最大数量, 它和 cpu 时间周期权重共同决定了一个虚拟机可以使用的 cpu 资源. Xenserver 通过 VCPU-params 中的 cap 参数来调控它, 100 表示一个物理 cpu, 50 表示半个物理 cpu. 虚拟 cpu 数量对虚拟机应用性能的影响见图 2.

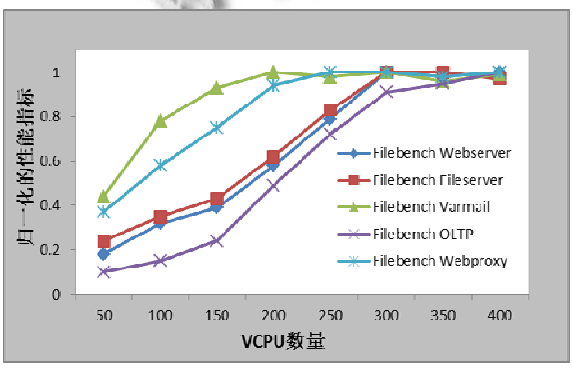

图 2 虚拟 cpu 的数量对虚拟机应用性能的影响

内存: 内存决定了一个虚拟机可以使用的内存数. Xenserver 中可以设置动态内存最大、最小值, 当 Xenserver 中的内存不够用时, 其会动态调整虚拟机的内存来使整个虚拟环境的虚拟机正常运行. 实验过程中可以将每台虚拟机的动态内存最大、最小值设为同一数值以保证虚拟机间内存的物理隔离. 内存对虚拟机应用性能的影响见图 3.

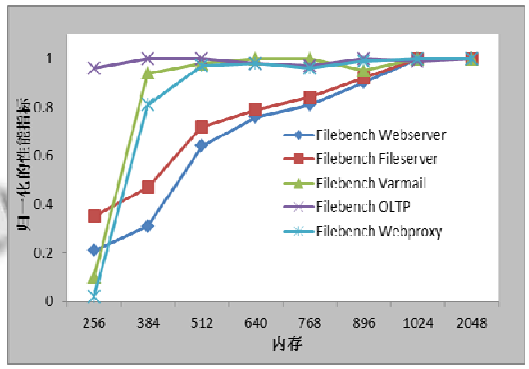

图 3 内存对虚拟机应用性能的影响

I/O 竞争: 将 fio[12]程序运行在单独的一台虚拟机上可以用来模拟 I/O 竞争, 并用模拟应用程序采集到的 I/O 延时数据来作为该参数的具体数值. 可以通过 fio 程序模拟不同层级的 I/O 竞争, 并且将其映射成不同的 I/O 延时. I/O 延时对虚拟机应用性能的影响见图 4.

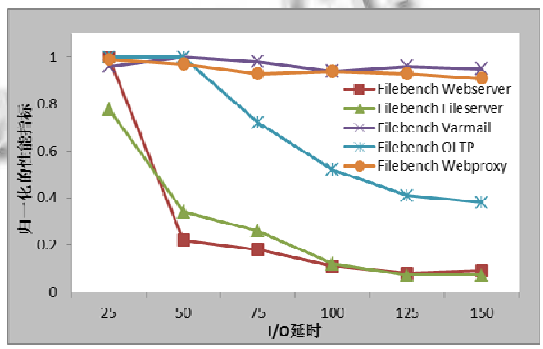

图 4 I/O 延时对虚拟机应用性能的影响

通过实验我们发现, 不同类型的应用程序在不同的参数的影响下表现异同. 以内存为例, 一些应用程序(如 varmail、webproxy)对内存非常敏感, 一些则表现出和内存的一个正相关的关系(如 webserver, fileserver), 一些应用程序如 OLTP 则对内存没有体现出太大的依赖性. 通过以上的实验表明: ① CPU 时间权重(cpu_time)、虚拟 CPU 的数量(vcpu)、内存(memory)

以及 I/O 竞争(I/O contention)这四种硬件资源容易观测且易于控制, 并且可以通过调节它们影响虚拟机应用性能, 可以作为实验建模的特征参数. ②不同类型的应用程序在不同参数下的表现各不相同, 单一的线性模型不能完全捕捉它们之间的关系. ③各参数与虚拟机应用性能间存在着复杂的非线性关系, 通过这些参数对性能进行建模存在一定的难度, 尤其再考虑上虚拟机硬件资源之间相互作用的因素.

4　方法介绍

本节将详细介绍我们提出的基于 SVD 的特征拓展方法和虚拟机应用性能建模方法.

4.1 基于 SVD 的特征拓展

SVD算法是线性代数中一种重要的矩阵分解算法, 在信号处理、统计学等领域都有着重要的应用. 近些年来矩阵分解算法被广泛应用到推荐领域, 这一类型的算法通常会有很高的预测精度, 也活跃于各大推荐系统竞赛上面. 其基本的思想是将原始矩阵分解成两个矩阵乘积的逼近, 具体算法如下: 假设R是一个$m*n$的矩阵, 其分解得到的矩阵U是一个$m*m$阶的方阵, S是一个半正定的$m*n$阶对角矩阵, V^T(V的转置)是一个$n*n$阶的方阵, 可以用公式表示为:

$$A_{m*n} = U_{m*m} * S_{m*n} * (V_{n*n})^T \quad (1)$$

上述分解会构建一个对角矩阵S, S的对角元素是从大到小排列的, 这些对角元素称为奇异值, 它们对应了原始数据集矩阵A中的奇异值. 奇异值反映了数据集中的重要特征, 并且奇异值和特征值是有关系的, 这里的奇异值就是矩阵$A*A^T$特征值的平方根. 上面提到过, 矩阵S只有从大到小排列的对角元素, 这就意味着数据集中仅有r个重要特征, 在很多情况下, 前10%甚至1%的奇异值的和就占了全部奇异值之和的99%以上上. 也就是说, 也可以用前r大的奇异值来近似描述矩阵, 所以(1)式可以近似表述为:

$$A_{m*n} = U_{m*d} * S_{d*d} * \left(V_{d*n}\right)^T \quad (2)$$

其中d是一个远小于m、n的数, 右边三个矩阵相乘的结果将会是一个接近于A的矩阵, d越接近于n, 则相乘的结果越接近于A.

在前面的工作中已经提到, 虚拟机中的每个硬件资源与虚拟机应用性能之间存在着复杂的非线性关系, 尤其当它们共同作用于应用性能时, 情况将变的更加复杂. 因此本文希望通过SVD拓展特征的方法将各参数之间相互作用的情况考虑进去, 并将生成的特征应用到模型建立中去. 在实验中, 我们将每个硬件资源在可调的范围内等分成k个层级, 比如可以将虚拟cpu的数量从(0,400]按每50一个层级等分成8个层级, 并且我们假设相似层级的硬件资源对于虚拟机应用性能的影响行为是相似的. 对于上节确定的4个硬件资源参数, 两两考虑它们相互作用的信息, 通过SVD的方法来拓展每种资源每个层级的特征, 特征拓展后每种资源每个层级都将得到一个d维的隐性特征, 见图5.

$$\begin{bmatrix} P11 & P12 & \cdots & P1n \\ P21 & P22 & \cdots & P2n \\ \vdots & \vdots & \ddots & \vdots \\ Pm1 & Pm2 & \cdots & Pmn \end{bmatrix} = \begin{bmatrix} f11 & f12 & \cdots & f1d \\ f21 & f22 & \cdots & f2d \\ \vdots & \vdots & \ddots & \vdots \\ fm1 & fm2 & \cdots & fmd \end{bmatrix} * \left(\begin{bmatrix} f11' & f12' & \cdots & f1d' \\ f21' & f22' & \cdots & f2d' \\ \vdots & \vdots & \ddots & \vdots \\ fn1' & fn2' & \cdots & fnd' \end{bmatrix} \right)^T$$

图 5　基于 SVD 的特征拓展

这个图是公式(2)的展开, 其中 A 矩阵中的每一元素表示的是当某两个层级的参数共同作用时虚拟机应用性能的指标(此时将另外两个参数层级设置为最大或零, 以防止其它参数成为性能的瓶颈), U 矩阵和 V 矩阵分别是分解出的两个参数每个层级的 d 维隐性特征. 以虚拟 CPU 的数量和内存这两个参数为例(此时将CPU时间权重和I/O竞争分别设置为65536和0, 以防止其它参数成为性能的瓶颈), A 矩阵中的每一行表示 CPU 数量分别为 50、100、150 … 400 时, 在内存为 256、512、768 … 2048 时的性能指标, 通过 SVD 算法将原始 A 矩阵分解成 U 矩阵和 V 矩阵, U 矩阵中的每一行表示 CPU 数量分别为 50、100、150 … 400 时其对应的隐性特征, 同理, V 矩阵中的每一行表示内存分别为 256、512、768 … 2048 时其对应的隐性特征. 所以如果问题定义了 k 个原始特征, 根据原始特征可以进行C_k^2次 SVD 分解, 设每次分解每个特征得到 d_i 个二维作用的隐性特征, 这样对于每个特征最终将得到$\sum_{i=1}^{k} d_i$个拓展特征.

4.2 模型训练

在模型训练中, 我们主要考虑三大类的模型训练方法: ①线性模型+线性特征(即原始特征, 后面为简单起见, 都将此类特征称为线性特征)的训练方法; ②线性模型+非线性特征(即包含基于SVD特征拓展的特征, 后面为简单起见, 都将此类特征称为非线性特征)的训

练方法; ③非线性模型+非线性特征的训练方法. 最终将观察三种训练方法的实验结果以评估线性模型、线性特征、非线性模型以及非线性特征在虚拟机应用性能建模中的有效性及优劣.

在实验中线性模型选择了最基本的线性回归模型, 其可以用公式表示如下:

$$y=\sum_{i=1}^{k}\omega_i\alpha_i+b+regularization \tag{3}$$

其中 $\omega=(\omega^{(1)},\omega^{(2)},\ldots,\omega^{(n)})$为参数向量, α 为输入向量即特征数据, 关于特征数据我们会在下一节有更加详细的介绍, $b\in R$ 为偏置信息, *regularization* 为正则化项. 非线性模型来自于一种叫做 SVDFeature[13]的算法改进, 同样, 该模型可以用公式表示如下:

$$b=\sum_{i=1}^{a_0}b_i \tag{4}$$

$$r=\sum_{k=1}^{a_0}\sum_{i=a_{k-1}+1}^{a_k}\alpha_i p_{i-a_{k-1}-1}^{(k)}\left(q_{i-a_{k-1}-1}^{(k)}\right)^T \tag{5}$$

$$y=\sum_{i=1}^{a_0}\omega_i\alpha_i+b+r+regularization \tag{6}$$

其中 b 为偏置信息, (4)式表示的是不同参数偏置的线性和, 比如在我们的实验中, 选择了 cpu 时间权重、虚拟 cpu 的数量、内存和 I/O 竞争这四种硬件资源作为建模的原始参数, 则在(4)式中, a_0就等于 4, 表示的是不同参数对性能影响的偏置信息. r 为各参数各层级硬件资源对性能的影响因素, 在我们的实验中, 由于各层级的参数特征由基于SVD的方法拓展而来, 所以(5)式实际上相当于包含了参数之间两两交互影响的信息. 在(5)式中, a_0的意义与(4)式相同, 且$a_0\in a$, $a=(a_0,a_1,a_2,\ldots,a_n)$,除$a_0$外, a_i代表每个参数的硬件层级数, 比如将内存从 256M-2048M, 每隔 128M 分成一个层级, 则内存的层级包括(256, 384,512, …, 2048)共 15 个层级, 则内存对应的a_i则等于 15, 引入向量 a 的目的是为了使处理数据更好的匹配模型公式, 同时细心的人还会发现其实a_n中的 n 即等于a_0,下节详细分析数据特征时这里就更加容易理解了. $p=(p^{(1)},p^{(2)},\cdots,p^{(k)})$,$p^{(i)}$为每个参数的拓展特征矩阵; 同样 $q=(q^{(1)},q^{(2)},\cdots,q^{(k)})$,$q^{(i)}$为每个参数拓展特征的参数矩阵. (6)式为最终的模型公式, 其实它本质上就是由三部分信息的线性和组成: ①原始特征信息, 即线性特征信息, 这部分主要反映每个参数单独对虚拟机应用性能的影响. ②偏置信息, 这部分主要反映每个参数的偏置信息对虚拟机应用性能的影响. ③交互特征信息, 即非线性特征信息, 这部分主要反映参数之间的交互作用对虚拟机应用性能的影响.

5 实验及分析

5.1 实验数据

我们利用脚本通过 Dom0 动态的调节各虚拟机资源来采集数据集. 在这个过程中, 对于 cpu_time, virtual_cpu, memory 以及 I/O 竞争级别这四种参数进行等分的调节, 共采集到五种 benchmark 下不同组合的应用性能数据 62000 多条, 作为后续实验的源数据进行处理.

对于采集到的数据, 将进行三种处理: ①将源数据处理成线性特征, 其中只包含了原始特征的信息. ②将源数据处理成包含了基于 SVD 拓展特征的非线性特征. ③将源数据处理成包含了参数层级信息的非线性特征. 图 6 展示了三种被处理好的特征形式.

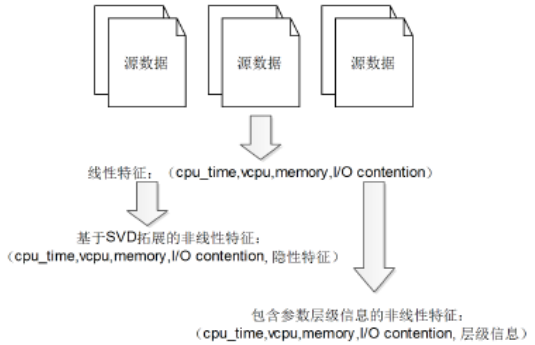

图 6 三种数据特征处理流程

5.2 实验描述

在此次实验中, 我们利用一台 4 核 8 线程, 拥有 16G 内存的 DELL Optiplex 9010 台式机作为实验的服务器, 并在其上运行 XenServer6.2 作为虚拟化层来搭建小型虚拟机集群. 虚拟集群包括 7 台虚拟机, 除 Dom0 和 I/O 竞争模拟机外, DomU 都运行 Ubuntu Server12.04,并分别在其上运行 Webserver、Fileserver、Varmail、OLTP、Webproxy 五种有代表性的标准检验程序. DomU 分配最多可使用 2G 内存、4 个虚拟 CPU; I/O 竞争模拟机通过运行 fio 软件包来模拟不同程度的 I/O 竞争, 并且分配 1G 内存, 可使用 4 个虚拟 CPU; 对于 Dom0, 分配足够的 CPU 和内存以防止 Dom0 的资源不足成为影响客户虚拟机应用性能的瓶颈. 在 Xenserver6.2 中, 可以通过类似于调节 DomU 硬件资源的方式调节 Dom0 的硬件资源, 因此可以保证在整个实验过程中 Dom0 至少可以使用 25%以上的 CPU 分配以及 2G 的内存, 并且 Dom0 可以通过 SSH 进行对

各客户虚拟机的控制以及资源分配. 对于采集到的数据通过 NFS 传输到另一台独立的 HP Elite7100 计算机上进行建模分析, 整个实验的架构流程图见图 7.

5.3 实验结果分析

利用 5.1 所处理的实验数据, 根据上节提到的三种建模方法分别对五种不同标准检验程序下采集到的数据建立模型. 对于建立好的模型, 用平均预测误差作为评价指标, 并采用 10 折交叉验证, 最终得到的实验结果见表 1, 图 8 展示了不同建模方法的结果比较.

表 1　三种建模方法实验结果

标准检验程序	建模方法	%每折的平均预测误差										%平均误差
		1	2	3	4	5	6	7	8	9	10	
Webserver	线性方法+线性特征	75.06	74.68	80.91	88.24	71.78	69.36	74.61	77.21	72.3	73.58	75.77
	线性方法+非线性特征	31.28	29.34	41.01	27.84	33.24	29.67	28.45	31.01	35.67	37.91	32.54
	非线性方法+非线性特征	15.31	14.72	13.6	13.88	17.94	15.76	16.42	24.71	20.01	19.22	17.16
Fileserver	线性方法+线性特征	50.21	59.33	51.29	49.01	56.98	48.04	51.09	49	53.51	46.23	51.47
	线性方法+非线性特征	24.59	22.09	28.51	19.08	24.55	27.81	20.54	23.71	39.93	25.78	25.66
	非线性方法+非线性特征	10.65	9.01	17	13.51	7.05	11.19	10.21	8.21	7.01	16.81	11.07
Varnmail	线性方法+线性特征	277.56	256.91	278.12	249.21	301.91	223.01	242.9	189.81	223.69	249.37	246.25
	线性方法+非线性特征	29.77	39.01	35.71	28.78	33.49	32.7	27.51	42.61	28.13	38.52	33.62
	非线性方法+非线性特征	6.97	6.92	5.68	7.27	8.21	6.82	7.08	7.37	5.83	7.61	6.98
OLTP	线性方法+线性特征	24.64	22.13	31.01	25.81	22.51	27.86	21.6	18.72	29.01	24.69	24.80
	线性方法+非线性特征	8.71	12.69	10.71	7.39	8.51	9.46	7.71	8.62	11.28	8.66	9.37
	非线性方法+非线性特征	6.21	8.59	7.46	6.14	7.08	8.3	6.85	7.23	8.1	6.15	7.21
Webproxy	线性方法+线性特征	384.09	351.79	371.97	295.48	362.64	336.69	371.71	324.44	367.91	311.67	347.84
	线性方法+非线性特征	100.46	114.7	105.83	103.47	92.94	106.82	117.08	115.35	107.83	110.1	107.46
	非线性方法+非线性特征	17.31	16.12	14.69	15.29	27.97	25.36	19.77	17.72	14.57	16.76	18.56

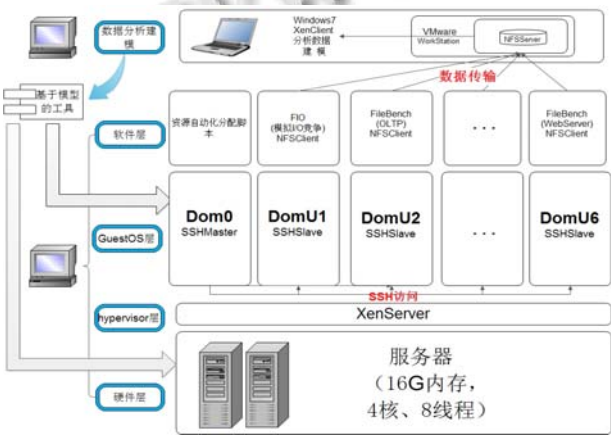

图 7　实验架构流程图

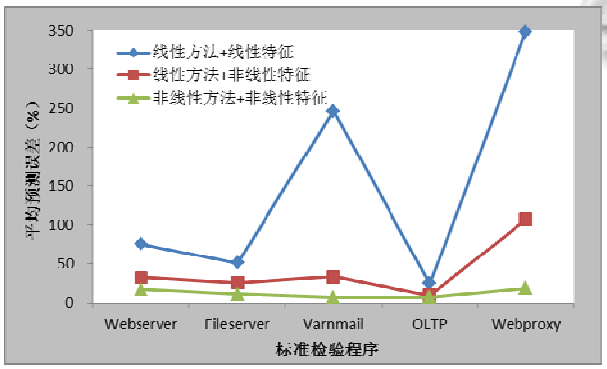

图 8　三种建模方法平均预测误差对比

根据表-1 和图-8 可以看到线性模型+线性特征的建模方法在不同类型的应用程序上表现迥异, 最好的情况在OLTP下的平均预测误差达到了24.8%, 而最差情况在 Webproxy 下的平均预测误差则达到了 300%以上. 这说明用线性模型+线性特征的方法建立的模型比较单一, 预测的随机性很大, 很难精确的捕捉到虚拟环境下虚拟机应用性能的复杂行为. 就像在第三节参数选择中所提到了, 虚拟环境下虚拟机的应用性能与多种硬件资源存在的异常复杂的关系, 不同类型的应用程序的性能瓶颈不同, 对于相同的硬件资源, 不同类型的应用程序表现不一致, 因此很难用单一的线性方法对其建模.

线性模型+非线性特征的建模方法从结果上看相对于线性模型+线性特征的建模方法有了较大的提升, 这可能有两点原因: ①基于 SVD 的特征拓展使得建模的特征增加, 线性建模的信息量增加, 模型中增加了特征间相互作用的信息. ②基于 SVD 的特征拓展相当于对非线性的特征进行了线性的转换, 这类似于 SVM 中的核技巧, 使得原始的非线性问题转换成了线性问题进行建模. 这种方法在一定程度上能够捕捉到虚拟机应用性能的复杂行为, 但效果还是不够好, 对于 5 种不同类型的应用程序, 其模型的平均预测误差为 41%左右.

最后非线性模型+非线性特征的建模方法在五种不同的应用程序中的平均预测误差达到了 12%, 相对于前两种建模方法在结果上有了较大的提高, 且对于不同的应用程序其建立的模型平均预测误差相对比较稳定, 都浮动在 12%左右, 这说明非线性模型+非线性

特征的方法能够较好的捕捉虚拟环境下虚拟机应用性能的复杂行为. 因为这种建模方法不仅包含了每种硬件资源单独作用于虚拟机应用性能的信息, 同时还包含了不同硬件资源相互作用对于虚拟机应用性能的影响信息, 以及每种资源的偏置信息; 同时, 仔细分析(4)、(5)、(6)式会发现, 我们使用的非线性模型相当于对每个参数每个层级的各种组合情况都进行了细粒度的建模, 使得该模型能够处理虚拟环境下虚拟机应用性能在各种硬件资源配置下的性能预测.

6 讨论

虽然我们提出的方法从结果上看取得了比较好的效果, 但对于方法中的一些假设和前提还是要在这里阐述一下.

首先, 基于SVD的特征拓展方法是一种纯数学的方法, 其物理的可解释性不是很强. 在实验中, 通过SVD的方法将每个参数每个层级两两相互作用的数据分解为细粒度的层级参数隐含特征, 但这些特征具体代表什么涵义, 对于不同的研究人员, 出发的角度不同, 可能理解的层次也不一样.

其次, 实验中我们将 cpu 时间周期权重分为了 30 个层级, 虚拟 cpu 数 8 个层级, 内存 15 个层级, I/O 竞争 8 个层级, 因此与其相对应的 U、V、p、q 矩阵维度也不是很大, 模型训练时间开销不是很大. 但随着数据中心增大, 硬件资源分配粒度更细、更多的时候, 基于 SVD 的特征拓展方法及模型训练的计算复杂度可能增长, 对于实时的虚拟机硬件资源调控应用, 在矩阵的计算上可能需要考虑更加优化的算法(如随机梯度下降)或者更加优秀的分布式计算框架(如 spark).

7 结语

精准的对虚拟环境下虚拟机应用性能进行建模对于用户和云服务提供商来说都是非常有用的. 然而, 硬件资源对于虚拟机应用性能非线性的影响、硬件资源之间的相互影响以及资源竞争都使得对于虚拟环境下的虚拟机应用性能变得异常复杂. 在我们的工作中, 确定了影响虚拟机应用性能的四种关键硬件资源作为建模参数, 同时提出了一种基于 SVD 特征拓展+非线性模型的建模方法, 使得对于各种不同的应用程序其平均预测误差能够达到 12%左右. 希望我们工作能对云服务提供商以及用户在合理的出售和配置虚拟机硬件资源方面做出贡献.

参考文献

1 Amazon elastic compute cloud (amazon EC2). http:// aws.amazon.com/ec2/.

2 Doyle RP, Chase JS, Asad OM, Jin W, Vahdat A. Model-based resource provisioning in a web service utility. In USENIX Symposium on Internet Technologies and Systems, 2003.

3 Bennani MN, Menasce DA. Resource allocation for autonomic data centers using analytic performance models. ICAC. IEEE Computer Society. 2005. 229–240.

4 Liu X, Zhu X, Singhal S, Arlitt MF. Adaptive entitlement control of resource containers on shared servers. IM, IEEE. 2005. 163–176.

5 McCullough JC, Agarwal Y, Chandrashekar J, Kuppuswamy S, Snoeren AC, Gupta RK. Evaluating the effectiveness of model-based power characterization. Proc. of USENIX Annual Technical Conference. 2011.

6 Padala P, Hou KY, Zhu X, Uysal M, Wang Z, Singhal S, Merchant A, Shin KG. Automated control of multiple virtualized resources. Proc. of the 4th ACM European Conference on Computer Systems. EuroSys. 2009. 13–16.

7 Bodik P, Goldszmidt M, Fox A, Woodard DB, Andersen H. Fingerprinting the datacenter: Automated classification of performance crises. EuroSys '10 Proc. of the 5th European conference on Computer Systems. 2010. 111–124.

8 Xu J, Zhao M, Fortes JAB, Carpenter R, Yousif MS. Autonomic resource management in virtualized data centers using fuzzy logic-based approaches. Cluster Computing, 2008, 11(3): 213–227.

9 Rao J, Bu X, Xu CZ, Wang LY, Yin GG. VCONF: a re-inforcement learning approach to virtual machines auto-configuration. ICAC, ACM, 2009. 137–146

10 Xenserver: an open source virtualization platform for managing cloud, server and desktop virtual infrastructures. https://wiki.xenserver.org/index.php?title=Main_Page.

11 Filebench: a framework for simulating applications on file.systems.http://www.solarisinternals.com/wiki/index.php/FileBench.

12 fio: Flexible I/O tester. http://freshmeat.net/projects/fio/.

13 Feature-based_Matrix_Factorization. http://www. recsyswiki.com/wiki/Feature-based_matrix_factorization.