负载类型相关的Xen虚拟机系统性能模型

余 勇[1] 车建华[2] 徐焕良[2] 蒋诚智[1]
(全球能源互联网研究院 南京 210003)[1] (南京农业大学信息科技学院 南京 210095)[2]

摘 要 针对Xen虚拟机系统执行网络I/O密集型负载时容易耗尽Domain0的CPU资源而过载和执行计算密集型负载时在客户域平均性能与数目之间存在线性规划的问题,提出了两个负载类型相关的性能模型。首先,通过分析Xen虚拟机系统处理网络I/O操作的CPU资源消耗规律,建立了CPU核共享和CPU核隔离两种情况下的客户域网络I/O操作请求次数计算模型;然后,通过分析多个相同客户域并行执行计算密集型负载的平均性能与一个相同客户域执行相同负载的性能表现之间的关系,建立了并行执行计算密集型负载的客户域平均性能分析模型。实验结果表明,两个性能模型能够有效地限制客户域提交的网络I/O操作请求次数以防止Xen虚拟机系统过载,并求解给定资源配置情况下执行计算密集型负载的Xen虚拟机系统客户域伸缩性数目。

关键词 Xen,虚拟化,虚拟机系统,性能模型,计算负载
中图法分类号 TP302.7 **文献标识码** A **DOI** 10.11896/j.issn.1002-137X.2016.11.041

Workload Type-dependent Xen Virtual Machine System Performance Models

YU Yong[1] CHE Jian-hua[2] XU Huan-liang[2] JIANG Cheng-zhi[1]
(Global Energy Interconnection Research Institute, Nanjing 210003, China)[1]
(College of Information Science and Technology, Nanjing Agricultural University, Nanjing 210095, China)[2]

Abstract In order to avoid the overload of Xen virtual machine system executing network I/O-intensive workload caused by the CPU resource exhaustion of Domain0 and solve the linear programming between the average perfor-mance and the number of guest domains when executing computing-intensive workload, two workload type-dependent performance models were proposed. Firstly, the network I/O request number computing models under the constraint of CPU core-sharing or CPU core-isolation were established by analyzing the CPU resource usage law of Xen virtual machine system executing network I/O-intensive workload. Secondly, the average performance analyzing models of guest domains executing computing-intensive workload were established by analyzing the relation between the average performance of multiple uniform guest domains concurrently executing computing-intensive workload and the performance of a same guest domain executing the same workload. The experimental results indicate that two performance models can effectively limit the network I/O request number of guest domains to prevent Xen virtual machine system from overloading and figure out the scalability number of guest domains in Xen virtual machine system executing computing-intensive workload with given computing resource respectively.

Keywords Xen, Virtualization, Virtual machine system, Performance model, Computing workload

1 引言

对于虚拟机系统,当其执行不同类型负载时,其性能表现各不相同。建立能够刻画虚拟机系统性能特征的分析模型,对于评价虚拟机系统执行不同类型负载时的性能表现很有帮助。以Xen虚拟机系统为例,受益于半虚拟化实现方式,其性能表现突出[1,2]。但是,Xen虚拟机系统也存在一些性能问题:

(1)由于客户域(guest domain)执行网络I/O操作时,不仅消耗自身的CPU资源,还消耗Domain0的CPU资源。当执行的网络I/O操作过于密集时,就可能会耗尽Domain0的CPU资源而造成Xen虚拟机系统过载。因此,如何限制客户域的网络I/O操作是一个非常重要的问题。

(2)由于给定资源配置后,执行计算密集型负载的Xen虚拟机系统客户域的平均性能与客户域的数目互为消长:当

到稿日期:2015-10-01 返修日期:2016-02-04 本文受国家科技支撑计划项目(2015BAK36B05),国家自然科学基金项目(71303120,61502236),江苏省农业"三新"工程项目(SXGC[2013]372,SXGC[2014]309),南京农业大学基本科研业务费专项基金项目(Y0201500222),南京农业大学青年科创基金项目(KJ2013033)资助。

余 勇(1970-),男,博士,高级工程师,主要研究方向为云计算、信息安全;车建华(1979-),男,博士,讲师,主要研究方向为云计算、大数据,E-mail:chejianhua@zju.edu.cn;徐焕良(1963-),男,博士,教授,博士生导师,主要研究方向为物联网、大数据;蒋诚智(1981-),男,博士,工程师,主要研究方向为云计算、信息安全。

客户域的数目增多时，客户域的平均性能将会下降；若要提高客户域的平均性能，则需减少客户域的数目。因此，如何寻找两者之间的最优组合是一个很有意义的问题。

为了解决上述问题，需要明确 Xen 虚拟机系统的资源消耗分布以及客户域平均性能与负载类型之间的关系，建立 Xen 虚拟机系统的网络 I/O 操作请求次数的求解模型和计算密集型负载平均性能模型。具体而言，首先针对客户域密集网络 I/O 操作容易造成 Xen 虚拟机系统过载的问题，建立限制客户域网络 I/O 操作请求次数的计算模型；然后针对给定资源配置情况下 Xen 虚拟机系统客户域平均性能与客户域数目之间的线性规划问题，建立执行计算密集型负载的客户域平均性能分析模型。

2 相关研究工作

自 1960 年分时系统出现之后，就开始了对虚拟机系统的性能评价研究。围绕虚拟机系统的性能建模，Bard 等人提出了能够分析 IBM VM/370 系统 CPU 利用率的性能模型[3]；王卅等人提出了一种基于硬件计数器的虚拟机性能互扰估算方法，从底层解决云数据中心虚拟化所带来的性能互扰问题[4]；杨雷等人构建了一个基于排队网的多层 Web 应用性能分析模型来解决虚拟机性能互扰和逻辑资源服务能力对多层 Web 应用性能的影响[5]；周东清和侣庆乾等人利用虚拟化技术提出了一个衡量不同类型物理机性能的模型和一个衡量多维资源利用率的模型，以降低异构云平台数据中心过高的能源消耗[6]；Akoush 等人分析了 Xen 虚拟机系统中影响虚拟机动态迁移时间的参数，并给出了两个性能预测模型[7]；Ye 等人提出了一种基于 Xen 的虚拟集群系统的性能分析模型[8]。

进一步，一些研究虚拟机系统中应用程序性能的模型被提出。例如，Benevenuto 等人提出了能够预测虚拟化环境中应用程序性能的模型，并分析了在 Linux 和 Xen 两种环境下该模型的有效性[9]；Meng 等人利用 See5/C5 设计了一系列依赖于应用程序类别的虚拟化环境性能预测模型[10]；Wood 等人跟踪和建模了虚拟化环境中应用程序的资源使用情况[11]。这些模型主要关注于应用程序的运行时间和资源利用率等指标。另外，诸如 Xenoprof[12]、XenMon[13] 之类的虚拟机系统性能监测工具不断出现，对虚拟机系统的性能建模有很大帮助。总体而言，上述工作为虚拟机系统的性能建模提供了很好的方法和工具，但是都没有深入研究处理网络 I/O密集型和计算密集型时 Xen 虚拟机系统的性能表现。本文在分析 Xen 虚拟机系统的网络 I/O 操作处理机制和 CPU 调度算法的基础上，设计了两个分别能够用于预测处理网络 I/O 密集型负载和计算密集型负载时 Xen 虚拟机系统性能的模型，为改进和优化 Xen 虚拟机系统的性能提供了一种有效方法。

3 网络 I/O 密集型负载性能分析模型

Xen 在早期版本中采用集成设备驱动程序的方式，使得其本身非常庞大，而且增加新设备时需要重写其驱动程序。为此，Xen 在后续版本中将所有设备的原始驱动程序存放于 Domain0 中。Domain0 负责其它客户域的创建、暂停、悬挂、唤醒、停止和销毁，以及网络 I/O 操作请求的处理，被称为驱动域(driver domain)[1]。自此，Xen 虚拟机系统的网络 I/O 操作处理流程变为：首先，客户域通过自身的网络 I/O 设备前端驱动接口发出网络 I/O 操作请求；然后，Domain0 通过网络 I/O 设备后端驱动接口接收来自客户域的网络 I/O 操作请求，并在网络 I/O 设备驱动程序列表中检索相应的驱动程序，访问硬件设备后采用 DMA 将所需的数据读入内存；最后，Domain0 通过后端驱动接口激活相应的客户域前端驱动接口以接收请求的数据，完成整个网络 I/O 设备访问过程。该处理方式的一个特点是客户域的网络 I/O 操作不仅会消耗自身的 CPU 时间片，还会消耗 Domain0 的 CPU 时间片。因此，当客户域执行的网络 I/O 操作过于密集时，Domain0 可能会由于消耗过多 CPU 时间片而造成 Xen 虚拟机系统过载。

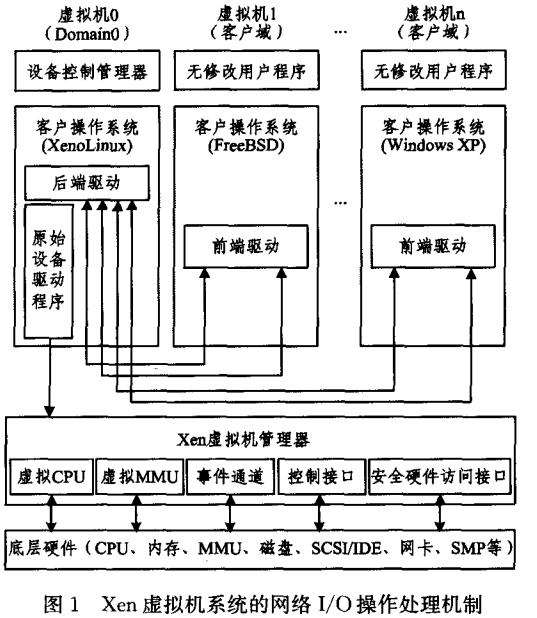

图 1 Xen 虚拟机系统的网络 I/O 操作处理机制

为了防止 Xen 虚拟机系统过载甚至崩溃，需要明确客户域执行网络 I/O 操作时 Xen 虚拟机系统的 CPU 资源消耗分布，建立限制客户域网络 I/O 操作请求次数的分析模型。

3.1 Xen 虚拟机系统的 CPU 资源消耗分布

Xen 虚拟机系统的 CPU 资源消耗分布包括两种情况：空载时的 CPU 资源消耗分布和处理网络 I/O 操作时的 CPU 资源消耗分布。

3.1.1 空载时的 CPU 资源消耗分布

当 Xen 虚拟机系统不执行任何负载时，所消耗的 CPU 资源包括 3 个部分：主机操作系统自身运行所消耗的 CPU 资源、所有域自身运行所消耗的 CPU 资源和剩余的空闲 CPU 资源。其中，所有域自身运行所消耗的 CPU 资源又包括两个部分：Domain0 自身运行所消耗的 CPU 资源和所有客户域自身运行所消耗的 CPU 资源。

假定 U_{all} 为 Xen 虚拟机系统的全部 CPU 资源，U_{host}^{self} 为主机操作系统自身运行所消耗的 CPU 资源，U_{virt}^{self} 为所有域自身运行所消耗的 CPU 资源，U_{idle} 为剩余的空闲 CPU 资源，U_0^{self} 为 Domain0 自身运行所消耗的 CPU 资源，U_g^{self} 为所有客户域自身运行所消耗的 CPU 资源，则：

$$U_{all}=U_{host}^{self}+U_{virt}^{self}+U_{idle}=U_{host}^{self}+U_0^{self}+U_g^{self}+U_{idle} \quad (1)$$

3.1.2 处理网络 I/O 操作时的 CPU 资源消耗分布

当客户域执行网络 I/O 密集型负载时，Xen 虚拟机系统将使用剩余的空闲 CPU 资源来处理网络 I/O 密集型负载。如果只考虑 Domain0 和所有客户域用来处理网络 I/O 密集型负载所消耗的 CPU 资源，而不考虑主机操作系统、Domain0 和所有客户域自身运行所消耗的 CPU 资源，则 Xen 虚拟机系统处理网络 I/O 密集型负载所消耗的 CPU 资源包括两个部分：各个客户域处理网络 I/O 密集型负载所消耗的 CPU 资源和 Domain0 处理各个客户域的网络 I/O 操作请求所消耗的 CPU 资源。

假定 $U_{I/O}$ 是 Xen 虚拟机系统处理网络 I/O 密集型负载所消耗的全部 CPU 资源，$U_0^{I/O}$ 是 Domain0 处理各个客户域的网络 I/O 操作请求所消耗的 CPU 资源之和，$U_g^{I/O}$ 是所有客户域处理网络 I/O 密集型负载所消耗的 CPU 资源之和，$U_{0i}^{I/O}$ 是 Domain0 处理第 i 个客户域的网络 I/O 操作请求所消耗的 CPU 资源，$U_{gi}^{I/O}$ 是第 i 个客户域处理网络 I/O 密集型负载所消耗的 CPU 资源，则一个运行着 n 个客户域的 Xen 虚拟机系统处理网络 I/O 密集型负载所消耗的 CPU 资源为：

$$\begin{aligned} U_{I/O} &= U_0^{I/O} + U_g^{I/O} \\ &= U_{01}^{I/O} + U_{02}^{I/O} + \cdots + U_{0n}^{I/O} + U_{g1}^{I/O} + U_{g2}^{I/O} + \cdots + U_{gn}^{I/O} \\ &= \sum_{i=1}^{n} U_{0i}^{I/O} + \sum_{i=1}^{n} U_{gi}^{I/O} \\ &= \sum_{i=1}^{n} (U_{0i}^{I/O} + U_{gi}^{I/O}) \end{aligned} \tag{2}$$

3.2 执行网络 I/O 操作的 CPU 资源消耗模式

为了帮助客户域处理网络 I/O 密集型负载，Domain0 主要负责响应客户域提交的网络 I/O 操作中断请求（包括发送数据包中断请求和接收数据包中断请求），驱动物理网络设备进行客户域网络 I/O 数据包的发送或接收，并在数据包发送或接收完成后通知相应客户域。Domain0 的整个处理过程只涉及中断请求的响应和设备驱动的激活，而不负责数据包的传输。因此，Domain0 处理网络 I/O 操作所消耗的 CPU 资源应该仅与客户域发送或接收数据包的请求次数有关，而与数据包的大小无关。

为了验证此猜测，使用网络 I/O 性能测试工具 httperf[14] 进行了两组实验。在两组实验中，通过配置 httperf 的发包率和收包率参数并使用 Xen 自带的 *Xentop* 命令来监控 Domain0 和客户域的 CPU 资源消耗情况（即 CPU 利用率）。

实验 1　固定 httperf 的发包率为 15000pkts/s，并依次从 100 字节到 1200 字节变换数据包的大小，实验结果如图 2 所示。经过多次实验发现不管发送或接收的数据包大小如何变化，Domain0 的 CPU 利用率很稳定，基本保持不变。

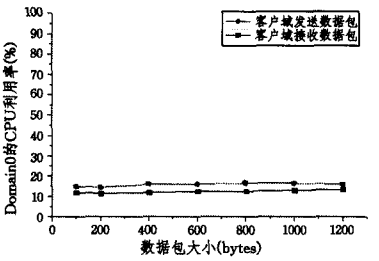

图 2　固定数据包发送或接收频率时 Domain0 的 CPU 利用率

实验 2　固定 httperf 发送或接收的数据包大小为 200bytes，并依次从 0 到 30000pkts/s 变化发包率和收包率，实验结果如图 3 所示。经过多次实验发现 Domain0 的 CPU 利用率随着发包率或收包率的递增而呈线性增长。

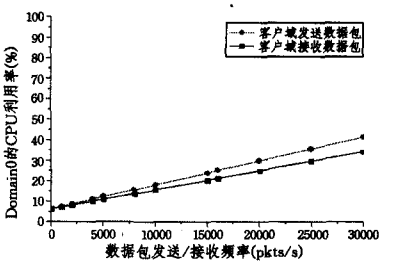

图 3　固定数据包大小时 Domain0 的 CPU 利用率

两组实验都很好地验证了对 Domain0 处理客户域网络 I/O 操作请求时 CPU 资源消耗特点的预测，即 Domain0 处理客户域的网络 I/O 操作请求所消耗的 CPU 资源只与客户域发送或接收数据包的请求次数（即客户域的发包率或收包率）有关，而与每次发送或接收数据包的大小无关。

两组实验中客户域接收数据包时 Domain0 的 CPU 利用率始终低于发送数据包时 Domain0 的 CPU 利用率，而且两者之比是一个常数，并不随发包率和收包率的变化而变化，这里将此比率称为 Domain0 处理客户域发送数据包操作时所消耗 CPU 资源相对于处理客户域接收数据包操作时所消耗 CPU 资源的权重（记为 $W_{s/r}$，反之表示为 $W_{r/s}=1/W_{s/r}$）。$W_{s/r}$ 的具体数值是可测的，但其依赖于测试平台，不同测试平台的 $W_{s/r}$ 数值将不同。根据上述实验结果，$W_{s/r}=1.2433$。

3.3 网络 I/O 操作请求次数计算模型

由于 Domain0 的空闲 CPU 资源来源有两种情况[12]：1）Domain0 与客户域共享所有的 CPU 资源，此时 Domain0 的空闲 CPU 资源即为整个 Xen 虚拟机系统的空闲 CPU 资源，而且可以被客户域占用；2）Domain0 和其它客户域分别使用不同的 CPU 或核，此时 Domain0 的空闲 CPU 资源为自己独享，不会被其它客户域占用。

3.3.1 核共享情况下的请求次数计算模型

假定 E_{0r} 为 Domain0 处理一次客户域接收数据包请求所消耗的 CPU 资源，E_{0s} 为 Domain0 处理一次客户域发送数据包请求所消耗的 CPU 资源，P_{ir} 为第 i 个客户域完成一次数据包接收操作所消耗的 CPU 资源，P_{is} 为第 i 个客户域完成一次数据包发送操作所消耗的 CPU 资源，N_{ir} 为第 i 个客户域接收数据包请求的总次数，N_{is} 为第 i 个客户域发送数据包请求的总次数，则有：

$$U_{0i}^{I/O} = N_{ir} \times E_{0r} + N_{is} \times E_{0s} \tag{3}$$

$$U_{gi}^{I/O} = N_{ir} \times P_{ir} + N_{is} \times P_{is} \tag{4}$$

其中，E_{0r}，E_{0s}，P_{ir}，P_{is}，N_{ir} 和 N_{is} 都是可测的。其中，E_{0r} 和 E_{0s} 可以使用 *Xentop* 命令获取，P_{ir} 和 P_{is} 可以使用 httperf 的 CPU 利用率参数获取，而 N_{ir} 和 N_{is} 可以通过插桩获取。由于所有客户域的发送数据包和接收数据包请求都要经过 Domain0 的后端驱动模块，因此可以在后端驱动模块中插桩代码以统计每个客户域发送数据包和接收数据包的请求次数。

根据前面的 $W_{s/r}$，可以将 Domain0 处理一次客户域发送数据包请求所消耗的 CPU 资源转化为 Domain0 处理一次客户域接收数据包请求所消耗的 CPU 资源，即 $E_{0s}=W_{s/r}\times$

E_{0r}；反之，$E_{0r}=W_{r/s}\times E_{0s}$亦成立。通过求解方程：

$$N_{ir}\times E_{0r}=N_{is}\times E_{0s} \tag{5}$$

可以得到：

$$\begin{cases}N_{ir}=N_{is}\times W_{s/r}\\N_{is}=N_{ir}\times W_{r/s}\end{cases} \tag{6}$$

当 Domain0 的空闲 CPU 资源都用来处理第 i 个客户域的网络 I/O 操作请求时，有：

$$\begin{aligned}U_{idle}&=U_{0i}^{I/O}+U_{gi}^{I/O}\\&=N_{ir}\times E_{0r}+N_{is}\times E_{0s}+N_{ir}\times P_{ir}+N_{is}\times P_{is}\\&=N_{ir}\times(E_{0r}+W_{s/r}\times E_{0s}+P_{ir}+W_{s/r}\times P_{is})\\&=N_{is}\times(W_{r/s}\times E_{0r}+E_{0s}+W_{r/s}\times P_{ir}+P_{is})\end{aligned} \tag{7}$$

由于 $W_{s/r}$ 和 $W_{r/s}$ 均为常数，而且 E_{0r}，E_{0s}，P_{ir} 和 P_{is} 都是可测的常数，因此可以计算出 Domain0 空闲 CPU 资源能够处理第 i 个客户域网络 I/O 操作请求的次数：

$$\begin{cases}N_{ir}=\dfrac{U_{idle}}{(E_{0r}+W_{s/r}\times E_{0s}+P_{ir}+W_{s/r}\times P_{is})}\\N_{is}=\dfrac{U_{idle}}{(W_{r/s}\times E_{0r}+E_{0s}+W_{r/s}\times P_{ir}+P_{is})}\end{cases} \tag{8}$$

进一步，当所有客户域的网络 I/O 操作请求次数（N_{ir} 和 N_{is}）固定时，Domain0 的空闲 CPU 资源能够支持的客户域数目（n）可以通过以下模型求解。

$$\begin{aligned}U_{idle}&=\sum_{i=1}^{n}(N_{ir}\times E_{0r}+N_{is}\times E_{0s}+N_{ir}\times P_{ir}+N_{is}\times P_{is})\\&=\sum_{i=1}^{n}N_{ir}\times(E_{0r}+W_{s/r}\times E_{0s}+P_{ir}+W_{s/r}\times P_{is})\\&=\sum_{i=1}^{n}N_{is}\times(W_{r/s}\times E_{0r}+E_{0s}+W_{r/s}\times P_{ir}+P_{is})\end{aligned} \tag{9}$$

3.3.2 核隔离情况下的请求次数计算模型

另一种建模情况是为 Domain0 和所有客户域指定不同的 CPU 核。例如，为 Domain0 指定一个或多个 CPU 核，为每个客户域指定其它单独的 CPU 核，则 Domain0 与客户域之间形成了严格的 CPU 资源隔离，不会出现客户域抢占 Domain0 的空闲 CPU 资源的情况。假定 Domain0 的空闲 CPU 资源为 U_0^{idle}，则有：

$$\begin{cases}N_{ir}=\dfrac{U_0^{idle}}{(E_{0r}+W_{s/r}\times E_{0s})}\\N_{is}=\dfrac{U_0^{idle}}{(W_{r/s}\times E_{0r}+E_{0s})}\end{cases} \tag{10}$$

同样，当所有客户域的网络 I/O 操作请求次数（N_{ir} 和 N_{is}）固定时，Domain0 的空闲 CPU 资源能够支持的客户域数目（n）可以通过如下模型求解。

$$\begin{aligned}U_0^{idle}&=\sum_{i=1}^{n}(N_{ir}\times E_{0r}+N_{is}\times E_{0s})\\&=\sum_{i=1}^{n}N_{ir}\times(E_{0r}+W_{s/r}\times E_{0s})\\&=\sum_{i=1}^{n}N_{is}\times(W_{r/s}\times E_{0r}+E_{0s})\end{aligned} \tag{11}$$

由于在上述模型中，$W_{s/r}$，$W_{r/s}$，E_{0r}，E_{0s}，P_{ir} 和 P_{is} 都是可测的常数，而 N_{ir} 和 N_{is} 均为与 i 相关的固定常数，因此上述模型都可解。

4 计算密集型负载平均性能分析模型

大量测试表明，Xen 虚拟机系统中多个相同客户域并行执行计算密集型负载的平均性能与一个相同客户域执行相同负载的性能表现存在线性关系。对此，建立了一个执行计算密集型负载的客户域平均性能分析模型。当给定 Xen 虚拟机系统运行的客户域数目时，该模型能够分析多个相同客户域执行计算密集型负载的平均性能；当给定 Xen 虚拟机系统客户域的资源配置时，该模型可以求解执行计算密集型负载的客户域伸缩性数目。该模型如式(12)所示：

$$P(m,W)=\frac{\alpha(W)}{m}\cdot P(1,W)+\beta(W) \tag{12}$$

其中，$P(m,W)$表示 m 个相同客户域并行执行负载 W 的平均性能，$P(1,W)$表示 1 个相同客户域执行负载 W 的性能，$\alpha(W)$和 $\beta(W)$均为与负载 W 的类型相关的常数因子。为了求解 $\alpha(W)$和 $\beta(W)$的值，基于主机操作系统为 Linux 2.6.18-53.1.14.el5xen 的一台 Dell™ PowerEdge™ 2900 服务器，测试了 Xen 虚拟机系统多个相同客户域并行执行计算密集型负载-Linpack 的平均性能，每个客户域分配的内存为 256MB。实验中，Xen 虚拟机系统同时运行的客户域数目从 1 逐次增加到 6，Linpack 的问题规模从 1000 逐次增加到 10000，实验结果如表 1 所列。

表 1 多个相同客户域的 Linpack 平均性能（单位：GFLOPS）

规模 \ 数量	1VM	2VMs	3VMs	4VMs	5VMs	6VMs
1000	4.4587	2.2159	1.3379	1.1267	0.8374	0.6683
2000	4.5361	2.2694	1.3752	1.1303	0.8493	0.6697
3000	4.6085	2.3109	1.3978	1.1419	0.8504	0.6701
4000	4.6807	2.2965	1.4025	1.1790	0.7757	0.6469
5000	4.7107	2.2883	1.4134	1.1749	0.7535	0.6505
6000	4.7353	2.3379	1.5032	1.1873	0.8174	0.7221
7000	4.6760	2.3563	1.4906	1.1714	0.7318	0.6898
8000	4.6694	2.3372	1.4732	1.1621	0.7299	0.6652
9000	4.6351	2.3198	1.4673	1.1365	0.7354	0.6431
10000	4.6289	2.3075	1.4927	1.1283	0.7263	0.6318

对 Linpack 的每个问题规模使用最小二乘法来回归分析对应于 $P(1,W)$的 $P(m,W)$。例如，对 Linpack 问题规模为 1000 的回归分析的结果是 $\alpha(W)=1.01375$ 和 $\beta(W)=0.071$，即：

$$P(m,L_{1000})=\frac{1.01375}{m}\cdot P(1,L_{1000})-0.071 \tag{13}$$

其中，L_{1000}表示问题规模为 1000 的 Linpack 负载。实验结果表明，$\alpha(W)$和 $\beta(W)$对于 Linpack 的所有问题规模基本保持不变，可以视为常数。如图 4 所示，$\alpha(W)$的值基本维持在 1，$\beta(W)$的值基本维持在 0。这也表明，Xen 虚拟机系统中 Linpack 的平均性能随客户域数目的增加而呈线性递减的规律，即 m 个相同客户域并行执行 Linpack 的平均性能相当于 1 个相同客户域执行 Linpack 时性能的$\frac{1}{m}$。

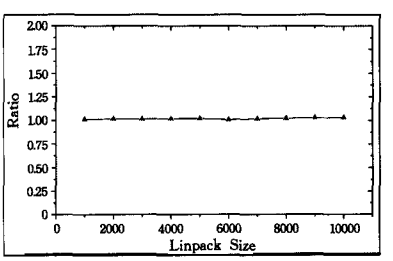

图 4 对应于 Linpack 各个问题规模的 $\alpha(W)$值

从相反角度看，使用式(12)还可以计算 Xen 虚拟机系统执行计算密集型负载的伸缩性数目。所谓伸缩性数目是指虚拟机系统可以容纳并行运行的空闲虚拟机的最大数目。但是，片面追求所能支持空闲虚拟机的最大数目而不考虑它们执行具体负载时的性能将无多大意义，因此此处分析的是执行 Linpack 时的伸缩性数目。

不失一般性，对于模型(12)，假定 $P(m,W)>0$。由于已经知道 Linpack 某个问题规模对应的 $\alpha(W)$ 和 $\beta(W)$ 的值，则根据模型(13)可以计算出对应于 Linpack 某个问题规模的伸缩性数目。例如，Linpack 问题规模为 1000 时对应的伸缩性数目 $m\leqslant 63$。其它问题规模对应的伸缩性数目(m)如图 5 所示。正如前面所预测，Xen 虚拟机系统伸缩性数目(m)的值随着问题规模的增加而下降，即 Xen 虚拟机系统伸缩性数目(m)的值随着执行计算密集型负载的客户域平均性能的增加而降低，两者之间存在反比关系。进一步，使用模型(12)可以求解运行于给定资源配置平台的 Xen 虚拟机系统执行计算密集型负载时，客户域平均性能与客户域数目的组合优化问题。

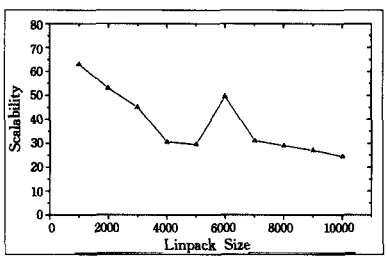

图 5　对应于 Linpack 各个问题规模的客户域数目

结束语　为了评价 Xen 虚拟机系统执行不同类型负载时的性能表现，在深入分析其网络 I/O 操作处理机制和 CPU 调度算法的基础上，提出了两个负载类型相关的性能模型：网络 I/O 操作请求次数计算模型和计算密集型负载平均性能分析模型。网络 I/O 操作请求次数计算模型主要用于计算 Domain0 的空闲 CPU 资源能够支持客户域网络 I/O 操作请求的次数，以帮助限制各个客户域能够提交的网络 I/O 操作请求次数和解决客户域密集网络 I/O 操作容易造成 Xen 虚拟机系统过载的问题；计算密集型负载平均性能分析模型主要用于分析 Xen 虚拟机系统多个相同客户域并行执行计算密集型负载的平均性能，能够解决 Xen 虚拟机系统执行计算密集型负载的客户域平均性能与客户域数目的最优组合问题。在后续工作中，将把限制客户域网络 I/O 操作请求次数的计算模型在 Xen 虚拟机系统的调度算法中加以实现。

参 考 文 献

[1] Barham P, Dragovic B, Fraser K, et al. Xen and the art of virtualization[C]// Proceedings of the 19th ACM Symposium on Operating Systems Principles (SOSP 2003). Bolton Landing, NY, USA, 2003: 164-177

[2] Shi L, Zou D Q, Jin H. Xen Virtualization Technology [M]. Wuhan: Huazhong University of Science and Technology Press. 2009: 83-89(in Chinese)
石磊，邹德清，金海. Xen 虚拟化技术[M]. 武汉：华中科技大学出版社，2009：83-89

[3] Bard Y. An analytic model of the VM/370 system[J]. Proc. of IBM Journal of Research and Development, 1978, 22(5): 498-508

[4] Wang S, Zhang W B, Wu H, et al. Approach of quantifying virtual machine Performance Interference Based on Hardware Performance Counter[J]. Journal of Software, 2015, 26(8): 2074-2090(in Chinese)
王卅，张文博，吴恒，等. 一种基于硬件计数器的虚拟机性能干扰估算方法[J]. 软件学报，2015，26(8)：2074-2090

[5] Yang L, Dai Y, Zhang B, et al. Queueing network based performance model for multi-tiered Web application with consideration of performance interference among virtual machine [J]. Computer Science, 2015, 42(1): 47-49(in Chinese)
杨雷，代钰，张斌，等. 考虑虚拟机间性能互扰基于排队网的多层 Web 应用性能分析模型[J]. 计算机科学，2015，42(1)：47-49

[6] Zhou D Q, Si Q Q. Energy-efficient virtual machine placement for heterogeneous cloud platform[J]. Computer Science, 2015, 42(3): 81-84(in Chinese)
周东清，侣庆乾. 异构云平台中能源有效的虚拟机部署研究[J]. 计算机科学，2015，42(3)：81-84

[7] Akoush S, Sohan R, Rice A, et al. Predicting the performance of virtual machine migration[C]// 2010 IEEE International Symposium on Modeling, Analysis & Simulation of Computer and Telecommunication Systems(MASCOTS). IEEE, 2010: 37-46

[8] Ye K, Jiang X, Chen S, et al. Analyzing and modeling the performance in xen-based virtual cluster environment[C]// 2010 12th IEEE International Conference on High Performance Computing and Communications(HPCC). IEEE, 2010: 273-280

[9] Benevenuto F, Fernandes C, Santos M, et al. Performance models for virtualized applications[M]// Frontiers of High Performance Computing and Networking-ISPA 2006 Workshops. Springer Berlin Heidelberg, 2006: 427-439

[10] Meng F, Du G, He H, et al. Performance Modeling on the Basis of Application Type in Virtualized Environments[J]. Journal of Software, 2013, 8(11): 2847-2854

[11] Wood T, Cherkasova L, Ozonat K, et al. Profiling and modeling resource usage of virtualized applications[C]// Proceedings of the 9th ACM/IFIP/USENIX International Conference on Middleware. Springer-Verlag New York, Inc., 2008: 366-387

[12] Menon A, Santos J R, Turner Y, et al. Diagnosing Performance Overheads in the Xen Virtual Machine Environment[C]// Proc. of First ACM/USENIX Conference on Virtual Execution Environments(VEE05). Chicago, IL, 2005: 13-23

[13] Gupta D, Gardner R, Cherkasova L. XenMon: QoS Monitoring and Performance Profiling Tool[R]. Technical Report HPL-2005-187, HP Labs, 2005

[14] HP Labs. Welcome to the httperf homepage [EB/OL]. http://www.hpl.hp.com/research/linux/httperf