Deploying Database Appliances in the Cloud

Ashraf Aboulnaga, Kenneth Salem, Ahmed A. Soror, Umar Farooq Minhas, Peter Kokosielis, Sunil Kamath

本文翻译：厦门大学计算机系 林子雨（http://www.cs.xmu.edu.cn/linziyu）

翻译时间：2010 年 7 月

本文英文论文引用方式：

[AboulnagaSSMKK09]Ashraf Aboulnaga, Kenneth Salem, Ahmed A. Soror, Umar Farooq Minhas, Peter Kokosielis, Sunil Kamath: Deploying Database Appliances in the Cloud. IEEE Data Eng. Bull. (DEBU) 32(1):13-20 (2009).

本文原始英文目录：

3 Tools and Techniques

Performance Models

Optimization and Control Algorithms

Tools for System Administrators

Co-tuning and Hint Passing

4 Virtual Machine Configuration

5 Future Directions

Reference

[本文翻译的原始出处：厦门大学计算机系数据库实验室网站林子雨老师的云数据库技术资料专区 http://dblab.xmu.edu.cn/cloud_database_view]

林子雨翻译内容如下：

Abstract

云计算是一种访问计算资源的、日益受欢迎的计算模型。一类比较受欢迎的计算云就是IaaS——基础设施即服务，比如 Amazon EC2。在这些云中，用户可以访问虚拟机器，并在虚拟机器上面安装和运行任意的软件，包括数据库系统。用户也可以在云环境中部署数据库应用工具(database appliance)，它们实际上就是预装了数据库系统并经过相应配置的虚拟机。在 IaaS 云中部署数据库应用工具，并执行调优和优化工作，带来了一些有趣的研究挑战。本文中，我们阐述了一些挑战，并且给出了一些解决挑战的工具和技术。我们为云环境中的调优问题提出了一个端到端的解决方案，也就是，把一个物理机器的 CPU 处理能力，在多个数据库应用工具之间进行分配。同时，我们也给出了未来的研究方向。

1 Introduction

云计算已经成为一个强大的高效的模型，它可以为用户提供计算能力。在云计算模型中，用户使用 Internet 或者 Intranet，来访问一个共享的计算云，这个云包含了许多互相连接的机器，这些机器组成一个或多个簇。这就为提供云服务的供应商和用户都带来了收益。对于计算能力的供应商而言，发展云计算的动力在于经济规模的扩展。通过在经过特殊设计和仔细

规划的数据中心里运营大量的服务器簇，供应商可以降低管理和运营成本，比如电力和冷却费用[15,16]。此外，硬件、软件和网络的单元费用，在大规模的情况下变得很低廉[4]。对于用户而言，云计算提供了简单而灵活的资源供应，不需要高端的设备和配置开销，也不需要管理和维护成本。用户可以在云中运行软件，并且，可以根据负载的变化，增加或减少计算资源。

云计算有许多种选择，主要取决于用户需要对运行在云环境中的软件拥有多少灵活性。本文中，我们关注的云计算环境是，用户看到一个主干服务器，只有一个操作系统，并且在安装和配置软件时可以获得很大的灵活性。这些云就被称为 IaaS 云。这种类型的云的一个很好的例子就是 Amazon EC2[2]，它可以让用户从 Amazon 租用计算资源来运行自己的软件。这种类型的云计算的其他类型的供应商是 GoGrid[13]和 AppNexus[3]。此外，许多组织都自己构建 IaaS 云作为内部使用[6,22]。

在 IaaS 云当中，用户通常被授予访问虚拟机[5,23]的权限，用户可以在这些虚拟机上安装和运行软件。这些虚拟机是通过一个 VMM（虚拟机监视器）来创建和管理的，VMM 是一个位于物理机器和操作系统之间的软件层。VMM 控制物理机器的资源，并且可以创建多个 VM，这些 VM 共享这个物理机器的资源。这些 VM 有相关的操作系统运行相应的应用，并且彼此隔离。VMM 控制如何把物理机器的资源分配给不同的 VMM。VMM 同时也提供了一些其他的功能，比如，对一个运行中的 VM 镜像进行存储和恢复，或者在物理机器之间迁移 VM。

在虚拟机器环境中部署软件的一个公共模型是，虚拟器件（virtual appliance）模型。一个虚拟器件是一个 VM 镜像，具有预装和配置的应用。配置一个应用，只需要把这个 VM 镜像拷贝到一个物理机器上面，启动这个 VM，然后执行任何配置任务即可。在 VM 上安装和配置应用的代价，是一次性的，这个代价发生在当这个器件(appliance)被创建的时候，此后，这个器件的用户就不需要再发生这个代价。一个数据库器件是一个虚拟器件，即预装的应用是一个数据库系统。随着虚拟化和云计算的日益普及，我们期望可以采用一种更加普遍的方式来提供数据库服务，也就是通过部署在 IaaS 云当中的数据库器件来提供数据库服务。作为一个部署这种应用器件的例子，Amazon 在 EC2 平台上提供了 MySQL\ORACLE 和 Microsoft SQL Server 应用器件服务。

现在需要提出的一个重要的问题就是，如果在这些环境下获得最好的数据库性能。云供应商对两个方面的性能目标感兴趣：最大化云资源的使用和以最小的资源满足用户的需求。用户的兴趣主要在于最小化应用响应时间，或者最大化应用吞吐量。在云中部署数据库器件和调整数据库和虚拟化参数来优化性能，带来了一些有趣的研究挑战。本文中，我们列出了一些挑战（第 2 部分），我们提出了不同的工具和技术来解决这些问题（第 3 部分）。我们给出了在不同的数据库器件之间分配 CPU 处理能力的例子，来阐述虚拟环境调优方案（第 4 部分）。最后在第 5 部分我们做总结，并展望未来研究方向。

2 Deployment and Tuning Challenges

我们的焦点是，对运行数据库系统（即数据库器件）的虚拟机器进行部署和调优，这些虚拟机器位于大型机器簇中。这里就提出了部署和计算的挑战，下面我们会详细阐述。

2.1 Deployment Challenges

创建一个很容易在云中部署的数据库器件，并从这个器件当中获得一个可访问、可使用的数据库实例，需要解决许多与部署相关的问题。这些问题不是我们研究工作的重点，但是，这里我们仍然要提出来，因为，这些看似简单而平凡的工作，实际上可能非常复杂和耗时。问题包括如下几个方面：

Localization

当我们从一个数据库器件的拷贝当中启动一个虚拟机的时候，我们需要给这个新的虚拟机以及运行在它上面的数据库系统一个唯一的标识。我们把这个过程称为“定位”。例如，我们需要为 VM 分配一个 MAC 地址、IP 地址和主机名称。我们也需要把运行在这个 VM 伤的数据库实例调整到这个 VM 的新名称。例如，一些数据库系统要求每个数据库实例都有一个唯一的名称，通常这个名称是基于 IP 地址或者主机名称的。VMM 和下面的操作系统以及网络设施，可以帮助解决这类问题，比如分配 IP 地址，但是，很少支持对数据库实例进行定位。这个特殊的定位操作与数据库系统有关，不同的数据库系统的定位是不一样的，这就增加了创建数据库器件所需要做的工作。

Routing

除了给每个 VM 和数据库实例分配一个唯一的标识以外，我们必须把应用需求引导到相应的 VM 和数据库实例。这包括 IP 层的路由，把 IP 数据包引导到某个 VM，同时也包括，必须把数据库请求正确引导到相应的端口，而不会被防火墙阻塞；I/O 请求必须被路由到正确的虚拟存储设备。

Authentication

虚拟机必须了解所有需要连接到自己的用户的身份标识，不管用户在云的哪个位置登陆。

2.2 Tuning Challenges

现在我们把焦点转移到另一个挑战，即对虚拟环境和数据库器件的参数进行调优从而获得最优的性能。这是本文的主要研究内容，主要包括以下几个方面：

Placement

虚拟化允许云提供者在任何可用的机器上运行用户的 VM。虚拟机器到物理机器的映射对性能会有显著的影响。一个简单的问题是，决定需要在一个物理机器上运行多少个虚拟机器。云供应商可能希望最小化物理机器的数量，但是，在一个物理机器上运行过多的虚拟机，会恶化这些 VM 的性能。因此，平衡这两个矛盾的目标是非常重要的：最小化物理机器的数量，同时，为用户维持可接受的性能。

虚拟机器到物理机器的更加复杂的映射，可能不仅仅需要考虑一个物理机器需要包括的虚拟机器的数量，也需要考虑这些 VM 的资源需求。例如，配置算法可以避免把多个 I/O 密集的 VM 映射到同一个物理机器上，从而最小化这些 VM 之间的互相干扰。这种类型的映射需要理解运行在 VM 上的应用的资源使用特性。这个任务对于数据库系统而言相对比较简单，因为数据库系统具有高度类型化的、可预测的资源使用模式。

Resource Partition

另一个调优的挑战就是，如何把一个物理机器的资源，分配给运行在它上面的多个 VM。许多 VMM 都提供了工具或者 API 来决定如何为 VM 分配资源。例如，VMM 调度参数，可以用来为所有的 VM 分配 CPU 处理能力，或者控制如何在物理 CPU 和虚拟 CPU 之间进行映射。其他 VMM 参数可以用来控制每个 VM 可用的物理内存的数量。为了获得最好的性

能，需要考虑运行在 VM 上的应用的特性，从而可以使得我们在分配资源时，可以实现收益最大化。

Service Level Objectives:

为了最优化云环境中的数据库器件的性能，可以表达不同级别的服务水平是很有用的。高级别的调优目标是，最小化所需的云资源，同时为数据库器件维护足够的性能。要表达这种“足够的性能”并不是一件简单的事情。数据库系统通常是用来满足服务应用需求的多层软件栈中的一个部分。服务水平协议通常是根据端到端应用性能来表达的，而不会显示这些性能预算有多少对数据库系统是可用的，又有多少是提供给其他层的软件（比如应用服务器和 WEB 服务器）使用的。为一个特定的应用计算出可以提供给数据库系统使用的资源，不是一件容易的工作，因为，一个应用需求可以导致不同类型的数据库请求，这些数据库请求在复杂程度方面各不相同，主要取决于所采用的 SQL 表达式。在云环境中进行调优，需要开发一个实际可行、直观的方式来表达服务水平目标。不同的负载可能具有不同的服务水平目标，调优算法需要考虑这些不同的服务水平目标。

Dynamically Varying Workloads

调整一个数据库器件的性能，需要器件负载的知识。这个负载可以是 SQL 语句的全集。但是，这里有个有趣的问题，那就是，是否存在一种更加简洁而有用的负载表达形式。另外一个有趣的问题就是，一些调优决定是否可以不需要 SQL 语句的知识。同时很重要的是，需要探测到负载发生了变化，可以通过对负载进行分类[11]，然后，判断负载类型是否发生了变化。这种探测算法需要能够处理动态变化的负载，这些动态负载具有不同的服务水平目标。

3 Tools and Techniques

接下来，我们把焦点转移到一些工具和技术，它们可以用来解决上面提出的调优挑战。具体包括以下几个方面：

Performance Models

预测不同调优动作对数据库器件的性能产生的影响，是任何调优解决方案的重要组成部分。这就需要为虚拟环境当中的数据库系统开发准确的、高效的性能模型。有两大类的模型：

白盒子模型和黑盒子模型。白盒子模型是基于数据库系统的内部知识；黑盒子模型，通常是基于对数据库系统性能的外部的、经验的观察而得到的统计模型。

白盒子模型对于数据库系统是非常吸引人的，主要有两个原因。第一，数据库系统为用户请求设计了类型化的和受限制的接口，数据库系统接受并执行 SQL 语句。这就简化了性能模型的输入。第二，更加重要的是，数据库系统已经具有高度精炼的性能优化模型。构筑白盒子模型的一种方式是，把这些数据库系统的内部优化模型暴露给调优算法，并且根据调优算法的调整而变化。例如，查询优化器代价模型，它已经在自动化物理数据库设计[7]当中被广泛使用，我们可以使用它来对“把不同数量的资源分配给数据库器件的效果”进行量化评估。自我管理的数据库系统有其他内部模型可以暴露给调优算法，包括被自调内存管理器使用的内存消耗模型[8,21]，以及自动诊断或性能问题诊断[10]所使用的模型。

白盒子模型的缺点是，所需要的性能模型并不总是可以在数据库系统中得到，从头开始白盒子模型既困难又耗时。即使当内部的模型确实存在于数据库系统当中，这些模型有时候也不能恰好提供所需要的性能指标，它们有时候需要假设忽略掉问题的重要方面。例如，查询优化器代价模型，最初是被设计成对不同的查询执行计划进行代价评估，而不是准确地评估资源消耗。这种代价模型在同一个时间只关注一个查询，而忽略掉与它同时运行的其他多个查询[1]。因为白盒子模型的这些缺点，有时候就需要构造黑盒子模型，即利用观察到的性能测试结果来设计统计模型[1]。在构造这些模型的时候，需要仔细决定开展哪些性能试验，因为，这些试验可能代价很高，它们对模型的准确性有很大的影响[18]。但是，IaaS 云所能够提供的无限资源的幻觉，实际上可以简化数据库系统的黑盒子试验模型，因为，我们现在可以很容易地提供尽可能多的机器来执行性能试验，从而构造一个精确的模型。

一个有趣的研究问题是：把黑盒子模型和白盒子模型的各自优点，进行最好的结合，即通过把数据库系统的内部模型作为模型设计起点，然后，根据实验结果对模型进行不断的改进[12]。

Optimization and Control Algorithms

解决云计算环境当中的性能调优问题，需要开发组合优化或者自动控制算法，它们可以使用上面描述的性能模型，来决定最好的调优动作。这些算法可以是静态算法，假设负载是

不发生变化的，也可以是动态算法，假设负载是动态变化的。算法可以简单地把最大努力性能优化[20]作为目标，或者，它们也可以把满足不同负载的服务水平作为目标[17]。

Tools for System Administrators

除了上述描述的模型和算法，系统管理员需要部署和调整数据库器件的工具。这些工具应该把 VM 的性能特点展现出来，而且还应该把运行在该 VM 上的数据库系统的性能展现出来。例如，把数据库系统的 what-if 性能模型展现给系统管理员，从而帮助他们做出调优决定，诊断性能问题，或者改进自动调优算法。

Co-tuning and Hint Passing

前面讨论的焦点在于，调整虚拟机器的参数。对运行在 VM 上的数据库系统进行调优也是非常重要的。例如，如果我们决定减少运行数据库系统的某个 VM 的可用的内存的数量，我们就需要减少这个数据库系统的内存池的大小。VM 和数据库系统参数的伴随调优（co-tuning）是非常重要的，它可以保证，在某个层上的调优动作，可以和其他层上的调优动作保持协调。对 VM 调优和数据库系统调优进行协调的另一种方式就是，把数据库系统调优所用的参数作为 hint 传递给虚拟化层。例如，这些 hint 可以保证，存储数据库对象的、需要同时被访问的 VM 磁盘，不会被映射到同一个物理磁盘。关于哪些对象需要一起访问的信息，在数据库系统中是很容易获得的，并且这些信息对于虚拟化层是非常重要的。

4 Virtual Machine Configuration

本节中，我们将考虑如下的调优问题：给定共享同一个物理机器的 N 个虚拟机器，每个 VM 都运行一个独立的数据库系统实例。我们如何能够把物理机器可用的 CPU 处理能力，在 N 个虚拟机器之间进行最优分配？回忆一下，VMM 提供了一种机制，可以决定如何把 CPU 资源分配给不同的 VM。我们在下面概述了针对这个资源分配问题的一个解决方案，有关的详细细节可以参考文献[19,20]。

我们把“在 N 个虚拟机器之间分配物理机器的 CPU 处理能力”和“把 N 个工作负载的汇总吞吐量最大化”一起来考虑。这是“尽最大努力性能目标”，它不需要考虑不同负载的显示的服务水平目标。

每个数据库系统可以从增加的 CPU 处理能力中获得的收益，取决于数据库工作负载。我们假定有一个 SQL 语句集合，它构成了 N 个数据库系统的工作负载。这些工作负载代表了在相同的时间间隔内，不同的数据库系统所执行的 SQL 语句。我们假定负载是固定的，我们不需要处理动态变化的负载。

为了决定最好的 CPU 分配方法，我们需要一个数据库负载的性能模型，作为把 CPU 分配给运行这些负载的不同 VM 的函数。在我们的解决方案当中，我们使用了数据库系统的查询最优的代价模型，作为一个 What-if 模型，来预测在不同的 CPU 处理能力下的性能。这就需要查询优化器代价模型，能够了解改变 CPU 处理能力分配会带来什么影响。代价模型依赖一个或多个模型参数，来描述 CPU 处理能力与预测一个查询的 CPU 代价。我们对不同的 CPU 分配方案使用不同的 CPU 模型参数，从而更清楚地认识 CPU 分配对查询优化代价模型的影响。我们把这样一个代价模型称为“virtualization aware”。对于每个 CPU 分配方案都需要不同的模型参数，决定这些模型参数所需的测量过程，对于每个数据库和物理配置而言，只执行一次，并且可以用于运行在该数据库系统之上任何工作负载。

我们在一个贪婪算法当中，使用关于这 N 个 VM 上的数据库系统的虚拟化感知代价模型，来决定最优的 CPU 分配方案。基于比较预测性能和实际性能的对比，我们也提供了启发算法来改进代价模型。我们周期性地提供改进的启发算法，这样，每次对代价模型进行改进以后，我们就可以获得一个新的 CPU 分配方案。

为了解释我们方法的有效性，考虑下面的例子（如图 1 所示）。使用 Xen VMM[5]，我们创建了两个虚拟机器，每个虚拟机都运行一个 PostgreSQL 实例。我们在同一个物理机器上运行这两个 VM，这个物理机器的配置是：Sun 服务器，2.2GHz 双核 AMD Opteron Model 275*64 处理器和 8GB 内存，运行 SUSE Linux 10.1。对于这个例子而言，我们使用了一个 TPC-H 数据库，scale factor=1。在一个 PostgreSQL 实例上，我们运行一个工作负载，它包含了 TPC-H 查询 Q4 的三个实例。在另一个 PostgreSQL 实例上，我们运行另一个工作负载，它包含了 TPC-H 查询 Q13 的九个实例。最开始的时候，我们为每个 VM 分别分配可用的 CPU 资源的 50%，然后，测试两个负载的执行时间。其次，我们重复试验，但是，这次的 CPU 分配是由我们 CPU 分区算法来决定的。

5 Future Directions

前面的章节阐述了云计算环境当中的一个简单的性能调优问题和它的解决方案。对前面章节列出研究进行扩展，为未来工作开启了许多可能性，这些也正是我们以后要从事的研究。不是把单个物理机器的资源在多个 VM 之间进行分配，相反，我们可以考虑多个物理机器，把多个物理机器的资源在多个 VM 之间进行分配，也就是说，要决定每个 VM 使用哪台物理机器，每个物理机器的多少资源会被分配给这个 VM。我们也将把我们的工作扩展到动态变化的负载的情形，可能具有不同的外在服务水平目标。另一个有趣的研究方向就是根据观察实验结果，对基于查询优化器的代价模型的升级方法进行改进。

另一个有趣的研究方向是，对不同 VM 之间的 I/O 资源分配进行优化。一些 VMM，比如 VMWare ESX 服务器[23]，提供了一种机制，可以控制把一个物理机器的多少 I/O 带宽分配给运行在该机器上面的不同 VM。另一种控制 I/O 带宽到 VM 的分配的机制是，控制 VM 磁盘到物理磁盘的映射。使用这两种机制来优化数据库器件的性能，是一个有趣的研究方向，尤其是许多数据库负载都是 I/O 限制的。

还有一个有趣的问题就是，是否能够把内部的数据库系统模型暴露出来，而不是把查询优化器代价模型暴露出来，从而使用这些模型来进行 VM 调优，或者 VM 和数据库系统的伴随调优。例如，内存管理器性能模型，可以用来控制内存分配。

除了调整数据库器件的挑战，云环境同时提供了新的机遇。例如，由于我们可以随需提供 VM，那么，对一个数据库系统进行扩展，在新增加的 VM 上放置数据库系统的副本，来处理工作负载当中突发的高峰，就是可能的。这就需要在复制过程当中保证数据一致性，把访问请求引导到老的或者新的 VM 上，并且开发一些策略，来决定什么时候增加副本，什么时候减少副本。

最后，虚拟化环境当中的“application-informed”的调优的概念，不只局限于数据库系统。这种概念可以用于其它类型的运行在云环境中的应用，比如运行在 Map-Reduce 类型平台[9,14]上的大型数据分析程序。（厦门大学计算机系 林子雨 翻译）

References

[1] Mumtaz Ahmad, Ashraf Aboulnaga, Shivnath Babu, and Kamesh Munagala. Modeling and exploiting query interactions in database systems. In Proc. ACM Int. Conf. on Information and Knowledge Management (CIKM), 2008.

[2] Amazon EC2. http://aws.amazon.com/ec2/.

[3] AppNexus. http://www.appnexus.com/.

[4] Michael Armbrust, Armando Fox, Rean Griffith, Anthony D. Joseph, Randy H. Katz, Andrew Konwinski, Gunho Lee, David A. Patterson, Ariel Rabkin, Ion Stoica, and Matei Zaharia. Above the clouds: A Berkeley view of cloud computing. Technical report, EECS Department, University of California, Berkeley, Feb 2009.

[5] Paul T. Barham, Boris Dragovic, Keir Fraser, Steven Hand, Timothy L. Harris, Alex Ho, Rolf Neugebauer, Ian Pratt, and Andrew Warfield. Xen and the art of virtualization. In Proc. ACM Symp. on Operating Systems Principles (SOSP), 2003.

[6] Luiz Andr´e Barroso, Jeffrey Dean, and Urs H¨olzle. Web search for a planet: The Google cluster architecture. IEEE Micro, Jan/Feb 2003.

[7] Surajit Chaudhuri and Vivek R. Narasayya. An efficient cost-driven index selection tool for Microsoft SQL Server. In Proc. Int. Conf. on Very Large Data Bases (VLDB), 1997.

[8] Beno ̂ıt Dageville and Mohamed Za ̈ıt. SQL memory management in Oracle9i. In Proc. Int. Conf. on Very Large Data Bases (VLDB), 2002.

[9] Jeffrey Dean and Sanjay Ghemawat. MapReduce: Simplified data processing on large clusters. In Proc. Symp. on Operating System Design and Implementation (OSDI), 2004.

[10] Karl Dias, Mark Ramacher, Uri Shaft, Venkateshwaran Venkataramani, and Graham Wood. Automatic performance diagnosis and tuning in Oracle. In Proc. Conf. on Innovative Data Systems Research (CIDR), 2005.

[11] Said Elnaffar, Patrick Martin, and Randy Horman. Automatically classifying database workloads. In Proc. ACM Int. Conf. on Information and Knowledge Management (CIKM), 2002.

[12] Archana Ganapathi, Harumi Kuno, Umeshwar Dayal, Janet Wiener, Armando Fox, Michael Jordan, and David Patterson. Predicting multiple metrics for queries: Better decisions enabled by machine learning. In Proc. IEEE Int. Conf. on Data Engineering (ICDE), 2009.

[13] GoGrid. http://www.gogrid.com/.

[14] Hadoop. http://hadoop.apache.org/.

[15] James R. Hamilton. Cost of power in large-scale data centers, Nov 2008.

http://perspectives.mvdirona.com/2008/11/28/CostOfPowerInLargeScaleDataCenters.aspx.

[16] Randy H. Katz. Tech titans building boom. IEEE Spectrum, Feb 2009.

[17] Pradeep Padala, Kang G. Shin, Xiaoyun Zhu, Mustafa Uysal, Zhikui Wang, Sharad Singhal, Arif Merchant, and Kenneth Salem. Adaptive control of virtualized resources in utility computing environments. In Proc. European Conf. on Computer Systems (EuroSys), 2007.

[18] Piyush Shivam, Varun Marupadi, Jeffrey S. Chase, Thileepan Subramaniam, and Shivnath Babu. Cutting corners: Workbench automation for server benchmarking. In Proc. USENIX Annual Technical Conference, 2008.

[19] Ahmed A. Soror, Ashraf Aboulnaga, and Kenneth Salem. Database virtualization: A new frontier for database tuning and physical design. In Proc. Workshop on Self-Managing Database Systems (SMDB), 2007.

[20] Ahmed A. Soror, Umar Farooq Minhas, Ashraf Aboulnaga, Kenneth Salem, Peter Kokosielis, and Sunil Kamath.Automatic virtual machine configuration for database workloads. In Proc. ACM SIGMOD Int. Conf. on Management of Data, 2008.

[21] Adam J. Storm, Christian Garcia-Arellano, Sam Lightstone, Yixin Diao, and Maheswaran Surendra. Adaptive selftuning
memory in DB2. In Proc. Int. Conf. on Very Large Data Bases (VLDB), 2006.

[22] Virtual Computing Lab. http://vcl.ncsu.edu/.

[23] VMware. http://www.vmware.com/.

[本文翻译的原始出处：厦门大学计算机系数据库实验室网站林子雨老师的云数据库技术资料专区 http://dblab.xmu.edu.cn/cloud_database_view]