资料说明文档

文献 01 Bubble Up：通过合理的协同定位提高现代仓库级计算机的利用率

背景

随着世界上大部分计算继续进入云计算，在现代数据中心中，为确保延迟敏感任务（如网络搜索）的性能隔离而过度规划计算资源是导致机器利用率低的主要原因。由于无法准确预测多核系统上共享资源的争用导致的性能下降，导致了简单地禁止高优先级、延迟敏感的任务与其他任务共处一地的高压方法。执行这种精确的预测一直是一个具有挑战性且尚未解决的问题。

仓库规模计算机（WSC）容纳大型 Web 应用程序和云服务，这些数据中心的建设和运营成本从数千万美元到数亿美元不等。随着越来越多的计算转移到云中，尽可能高效地利用 WSC 中的资源变得极其重要。然而，现代 WSC 中计算资源的利用率仍然很低，通常不超过 20%。

数据中心中的每台计算机都包含多个内核，通常每个插槽 4 到 8 个核心，每台计算机 2 到 4 个插槽。但是，鉴于单台计算机上并行性的巨大潜力，内核之间共享了许多资源。这种共享可能会导致内核之间的性能干扰，对面向用户和延迟敏感的应用程序线程的服务质量 （QoS） 产生负面且不可预测的影响。为避免潜在的干扰，不允许对延迟敏感的应用程序使用共置，从而使核心处于空闲状态，并导致过度预配，从而对整个数据中心的利用率产生负面影响。

任务

这种过度预配通常是不必要的，因为共置可能会也可能不会导致严重的性能干扰。**这项工作的目标是能够精确预测由于内存子系统中的共享资源争用而导致的性能下降。精确预测是在位于同一位置时提供预期性能损失量的预测。使用此信息，可以允许不违反应用程序 QoS 阈值的共置，从而提高数据中心的利用率。**

方法

在本文中，我们介绍了 Bubble Up，这是一种表征方法，**能够准确预测内存子系统中共享资源争用导致的性能下降。通过使用气泡向生产数据中心处理器上的内存子系统施加可调的“压力"，我们的方法可以预测同位置应用程序之间的性能干扰，准确率在实际性能下降的 1% 至 2% 以内。**使用这种方法在谷歌的生产数据中心与现实世界中的大型应用程序中实现“合理"的协同定位，我们可以将 500 机器集群的利用率提高 50% 至 90%，同时保证对延迟敏感的应用程序的高质量服务。

Bubble Up 的关键见解是，预测协同运行应用程序的性能干扰可以分解为 1）测量应用程序生成的内存子系统上的压力，以及 2）测量不同级别的压力对应用程序的影响。基本假设是压力和灵敏度都可以使用通用的压力度量进行量化。具有这样的度量降低了同位置分析的复杂性。与分析和表征每一个可能的成对共定位的暴力方法相反，Bubble Up 只需要表征每个应用一次，就可以产生精确的成对干扰预测（例如 O（N））。

冒泡是一个分两步的表征过程。首先，针对膨胀的气泡对每个应用程序进行测试，以产生灵敏度曲线。气泡是一种为内存子系统精心设计的压力测试，它为整个系统施加的压力提供了一个“刻度盘"

具体贡献如下

我们介绍了 Bubble Up 的设计，这是一种通用的表征方法，能够精确预测任意应用程序在同一位置时所遭受的性能退化。

我们介绍了 17 个谷歌生产工作负载，并描述了它们在生产服务器上共存时对性能干扰的倾向。

除了在 SmashBench 套件中展示我们的 Bubble Up 方法对有争议内核的预测准确性外，我们还评估了在生产数据中心环境中应用 Bubble Up 方法来引导谷歌应用程序成对共处时的预测准确性和利用率的提高。

使用 Bubble Up，我们能够准确预测由于谷歌应用程序的任意位置导致的性能下降，误差最多为 2.2%，通常小于 1%。为了评估使用 Bubble Up 来引导生产工作负载的 QoS 强制共存，我们在一个 500 台机器的集群中进行了一项研究，并能够将集群中的机器利用率提高 50%-90%，这取决于延迟敏感应用程序的允许 QoS 阈值。

结论

在这项工作中，我们提出了一种新的方法，用于精确预测由共享资源争用导致的性能下降。这种机制在新兴的仓库规模计算领域尤为重要。通过将应用程序对内存子系统有争议的压力的敏感性及其在子系统中产生的压力的敏感性解耦，我们能够预测成对共址，平均预测误差仅为 1%。在我们的实验群集设置中使用 Bubble- Up，我们能够将数据中心的利用率提高 50% 到 90%，同时实施一系列 QoS 策略。

文献 02 DeepDive：透明地识别和管理虚拟环境中的性能干扰

背景

许多企业和个人已经将工作负载转移到基础设施即服务（IaaS）提供商，如亚马逊和Rackspace。云计算扩展的一个关键因素是虚拟化技术。IaaS 提供商使用虚拟化来:
（1）将每个客户的应用程序打包到一个或多个虚拟机（VM）中
（2）隔离行为不端的应用程序
（3）通过在多个 VM 中多路复用其物理机（PM）来降低运营成本
（4）简化 VM 在 PM 之间的放置和迁移。

尽管虚拟化有很多好处，包括它能够在 CPU 和内存空间分配方面很好地分割 PM，但在这些环境中，性能隔离还远远不够完美。具体来说，提供商面临的一个挑战是识别（和管理）位于每个 PM 的虚拟机之间的性能干扰。例如，两个虚拟机在一起运行时可能会在共享硬件缓存中颠簸，但当每个虚拟机都单独运行时，它们很好地适应了共享硬件缓存。另一个例子是，两个虚拟机在隔离运行时各自具有顺序磁盘 I/O，在一起运行时可能会在共享磁盘上产生随机访问模式。更糟糕的是，技术趋势指向拥有数百甚至数千个核心的多核 PM。在这些 PM 上，遇到干扰的机会将增加。

干扰会严重削弱客户对云提供可预测性能能力的信任。因此，干扰可能会成为吸引对性能敏感的客户的绊脚石。有效处理干扰具有挑战性，原因有很多。首先，IaaS 提供商忽视了客户的应用程序和工作负载，并且无法轻易确定干扰正在发生。

此外，IaaS 提供商不能依赖应用程序来报告其性能水平（从而知道何时发生干扰），因为这可能会让应用程序开发人员负担过重，他们也不可信。这一挑战与不透明的方法背道而驰。其次，干扰本质上很复杂，可能是由于任何服务器组件（例如，共享硬件缓存、内存、I/O）造成的。

只有当共存的虚拟机同时竞争硬件资源时，干扰才会显现出来。预测性能下降的现有方法不适用，因为它们要求提供商在部署前长时间访问位于同一位置的虚拟机。干扰检测必须是一种更快的在线活动。最后，大型公共提供商每天部署的大量新虚拟机可能会导致可扩展性问题。

任务

我们描述了 DeepDive 的设计和实现，该系统用于**透明地识别和管理基础设施即服务云环境中位于同一物理机器上的虚拟机（VM）之间的性能干扰。**

方法

DeepDive 成功地解决了几个重要挑战，包括缺乏来自应用程序的性能信息，以及详细干扰分析的巨大开销。我们首先**展示了使用易于获取的低级别指标来清楚地辨别干扰何时发生以及是什么资源造成的是可能的。接下来，使用现实的工作负载，我们展示了 DeepDive 可以快速了解共存虚拟机之间的干扰。最后，我们展示了 DeepDive 在检测到干扰时有效处理干扰的能力，通过使用低开销的方法来识别减轻干扰的 VM 位置。**

鉴于这些挑战，提出了 DeepDive，这是一个透明高效地识别和管理 IaaS 提供商干扰的系统。

贡献

1. 一种用于透明地获得关于干扰的基本事实的方法，包括应用行为的黑匣子检测以及仅使用低级别度量来精确定位干扰的罪魁祸首资源的能力。
2. 一种警告系统，通过了解正常的、非干扰的行为来减少详细干扰分析的开销。
3. 一种利用全局信息来提高可扩展性的技术，该技术使用在其他 PM 上运行相同工作负载的 VM 的行为。
4. 一种透明且廉价地迁移罪魁祸首虚拟机的机制，通过使用简单的合成基准来模拟虚拟机的低级别行为及其在实际迁移之前对其他虚拟机的影响。
5. 使用实际工作负载的结果显示：
   (a) DeepDive 以高精度透明地推断性能损失（平均误差小于 5%），并精确定位罪魁祸首资源；
   (b) 它是高度准确的（没有假阴性）并且具有低开销（很少的仿形机）；
   (c) 它做出快速（不到一分钟）和准确的 VM 放置决策。

据我们所知，DeepDive 是第一个透明高效地处理对任何主要服务器资源（包括 I/O）的干扰的端到端系统。它的部署将有两个主要好处。首先，它将使云提供商能够使用更少的资源来实现其服务级别目标，这将提高用户满意度并降低能源成本。其次，更智能的虚拟机布局将使云客户能够从提供商那里购买更少的资源。

文献 03 PACMan：性能感知虚拟机整合

背景

许多数据中心的平均服务器利用率很低，估计在 5%-15% 之间。这是浪费的，因为空闲服务器通常消耗其峰值功率的 50% 以上，这意味着低利用率的服务器消耗的能量明显高于高利用率的较少服务器。此外，低利用率意味着要使用更多的服务器，从而导致资本浪费。防止这种浪费的一个解决方案是在更少的服务器上整合应用程序。

整合不可避免地会引入资源争用，从而导致性能下降。为了减少这种争用，数据中心对资源进行虚拟化，并将它们拆分到整合在共享硬件上的应用程序中。但是，虚拟化并不能防止所有形式的争用，因此并不能完全消除性能下降。特别是，共享缓存和内存带宽中的争用会显著降低性能，这是针对各种工作负载测量的。执行时间增加了几十个百分点。

为了减少退化，先前的工作已经测量了可能的 VM 组合的退化，然后将导致最小退化的 VM 共同定位。但这种方法不尊重目标性能约束。性能对于互联网服务来说往往是至关重要的。对亚马逊、微软和谷歌服务的测量显示，延迟增加几分之一秒可能导致高达 1% 至 20% 的收入损失。下意识的反应是放弃从整合中节省的全部或部分。例如，在谷歌数据中心，整合的工作负载仅使用 50% 的处理器核心。每隔一个处理器核心都不使用，只是为了确保性能不会降低。

任务

我们希望保留整合虚拟机的性能，但不要在这样做时浪费过多的资源。挑战在于
（1）确定每个虚拟机在与不同的待整合虚拟机集放置时会降低多少
（2）确定可以在服务器上放置哪些虚拟机和多少虚拟机，以保持所需的性能。

方法

识别合适的 VM 的问题被证明是 NP 完全的，我们设计了一种计算高效的算法，我们证明该算法的性能接近理论最优。因此，我们的方法中未使用的多余资源明显低于目前的做法。

具体而言，我们做出了以下

贡献

首先，我们提出了一个具有性能意识的整合管理器 PACMan，它可以最大限度地降低资源成本，如能耗或使用的服务器数量。PACMan 整合虚拟机，使性能下降保持在指定范围内。由于这个问题是 NP 完全的，PACMan 使用了一种近似但计算高效的算法，我们证明该算法在对数上接近最优。

其次，当面向客户的应用程序优先考虑性能时，批处理过程（如 Map Reduce）可能会优先考虑资源效率。对于这种情况，PACMan 提供了一种 "Eco" 模式，可以填充所有服务器核心，并最大限度地减少最坏情况下的降级。我们特别考虑最坏的情况，因为在 Map Reduce 中，在所有地图任务完成之前，Reduce 无法启动，因此，只有最坏的地图任务的降级才重要。我们表明，很难为这种情况设计可证明的近似最优方法，并提出合适的启发式方法。

最后，我们使用 SPEC CPU 2006 应用程序上测量的降解情况来评估 PACMan。为了在保持性能的同时最大限度地减少浪费的资源，PACMan 在最佳方案的 10% 以内运行，与不考虑干扰的合并方案相比，节省了 30% 以上的能源，并且与当前实践相比，将总运营成本提高了 22%。对于 Eco 模式，与传统方法相比，PACMan 的降解率降低了 52%。

文献 04 Bubble-flux：精确的在线 QoS 管理可提高仓库规模计算机的利用率

这篇文章探讨了提供更高机房服务器利用率的同时,也保证延迟敏感应用的服务质量(QoS)的问题。

作者提出了 Bubble-Flux 机制来解决这个问题。Bubble-Flux 包含两个主要组件:动态 Bubble 和在线 Flux 引擎。

动态 Bubble 可以实时地在服务器上进行轻量级的应用特性检测,生成延迟敏感应用当前的服务质量敏感度曲线。这使得调度器可以利用该曲线更准确地预测并发运行作业对服务质量的影响,从而选择更多"安全"的并发作业来提高服务器利用率。

在线 Flux 引擎使用了"淡入/淡出"机制来动态地监测和控制批处理作业的执行,以保证延迟敏感应用的服务质量。一旦监测到服务质量下降,它会通过加快淡出批处理作业来防止服务质量违规。Flux 引擎也能逐步淡入之前未见过的新作业,保证服务质量的同时也增加利用率。

两者结合可以实时获取延迟敏感应用当前的服务质量敏感度,基于此更好地选择并发作业,使用 Flux 引擎动态调整作业执行来适应变化,从而实现机房服务器利用率的提升同时也保证服务质量。作者通过实验表明,Bubble-Flux 机制在保证服务质量的同时,服务器利用率的提升能达到先前静态方法的 2.2 倍。

文献 05 Stay-Away：保护敏感应用程序免受性能干扰

这篇文章介绍了一个名为 Stay-Away 的机制,用于在与批处理应用共存环境下保护敏感应用的性能不受影响。

主要内容包括:

1. 虚拟机资源共享虽然可以提高利用率,但也会造成不同虚拟机之间的性能互相影响,这对 QoS 敏感的应用来说是一个难题。
2. Stay-Away 采用一个三步机制来解决这个问题:定期采样各虚拟机的资源指标,将其映射到二维状态空间,并预测执行是否会接近违反 QoS 的状态,必要时通过限制批处理应用来避免这种情况。

3. 它使用 MDS 技术将高维指标映射到二维空间,保留指标之间的相对距离关系,并标记出过去观测到的违反 QoS 状态。
4. 它预测未来状态的可能移动轨迹,如果预测将靠近违反状态则限制批处理应用,同时增加探索空间以逐步学习新的状态。
5. 它通过监控敏感应用在限制批处理应用后状态的变化来判断应该恢复批处理应用的时机。
6. 实验结果表明,Stay-Away 可以很好地保证敏感应用的 QoS,同时通过有效调度提高资源利用率。

总的来说,它提供了一种通用和自适应的机制来在共存环境下保护敏感应用不受性能影响,而无需过于保守地隔离它们。

文献 06 应用程序减速模型：量化和控制共享缓存和主内存的应用程序间干扰的影响

这篇文章提出了应用程序延迟模型(ASM),这是一个在线模型,可以准确估算应用程序由于共享缓存容量和主存储带宽干扰而产生的延迟。

ASM 的主要思想是:应用程序的性能大致成比例于它访问共享缓存的速率。即缓存访问率可以作为应用程序性能的一个代理指标。因此,ASM 将应用程序延迟定义为缓存访问率当应用独立运行/缓存访问率当应用共享运行的比率。

ASM 的一个关键挑战是如何准确估算应用程序独立运行时的缓存访问率(CARalone)。为此,ASM 采取了以下两步:

1. 定期将主存储带宽完全给予每一个应用程序,以此来最大程度降低主存储带宽干扰对 CARalone 的影响。
2. 使用辅助标签存储跟踪如果应用独立运行时缓存的状态,来识别哪些失效请求本应该命中缓存,并计算这些额外失效请求带来的延迟开销,从而量化共享缓存容量干扰对 CARalone 的影响。

通过这两步,ASM 能够较为准确地估算每个应用的 CARalone,进而估算应用的延迟。

与以往工作不同,ASM 采用聚合请求行为来量化干扰,而不是试图估算每个请求的延迟,这可以避免由于内存子系统并行带来的不确定性。实验结果显示,ASM 的延迟估算误差明显小于以往最佳方法。

最后,文章提出利用 ASM 估算的延迟来改进公平性的一些机制,如延迟感知缓存分区和存储带宽分配,这些机制的评估结果显示与其他方法相比,公平性和性能都有明显提升。

总之,这篇文章提出了一个在线和准确的 ASM 模型来估算应用程序由于共享硬件资源带来的延迟,并展示了如何利用这个模型来改进系统的公平性和其他属性。

文献 07 C2QoS: CPU-Cycle based Network QoS Strategy in vSwitch of Public Cloud

主要内容是关于在公共云环境中的虚拟交换机（vSwitch）中基于 CPU 周期的网络服务质量（QoS）策略。

在公共云中，vSwitch 是一个关键组件，用于管理虚拟机之间的网络流量。网络 QoS 是确保在共享网络资源下提供稳定和一致性网络性能的重要因素之一。C2QoS 提出了一种新的 QoS 策略，以提高公共云中 vSwitch 的性能，并更好地适应不同工作负载的需求。

C2QoS 的核心思想是将网络 QoS 与底层宿主机的 CPU 周期相关联。它通过评估每个虚拟机的 CPU 使用情况和网络流量的特征，动态地调整每个虚拟机分配的 CPU 周期资源。这种关联机制可以更好地适应不同虚拟机之间的资源需求差异，提高整个网络的性能和响应能力。

此外，C2QoS 还提供了一种基于 CPU 周期的调度算法，以减少网络流量的拥塞并改善网络资源的利用率。通过根据虚拟机的 CPU 周期资源和网络流量特征优化虚拟机之间的调度，C2QoS 可以提供更好的网络服务质量，同时降低网络延迟和拥塞情况。

综上所述，C2QoS 旨在通过基于 CPU 周期的网络 QoS 策略，在公共云的 vSwitch 中提供更优秀的网络性能和服务质量，以满足不同虚拟机工作负载的需求。

文献 08 DRL-Scheduling: An Intelligent QoS-Aware Job Scheduling Framework for Applications in Clouds

主要内容是关于云环境中应用程序的智能 QoS 感知作业调度框架。

云计算环境中，作业调度是一个关键任务，用于合理分配和管理资源，以确保应用程序的高性能和服务质量。DRL-Scheduling 提出了一种基于深度强化学习（DRL）的智能作业调度框架，旨在自动优化作业的调度决策，以满足应用程序的 QoS 要求。

DRL-Scheduling 框架的核心思想是利用深度神经网络和强化学习技术来模拟作业调度决策过程，并根据当前环境和作业的特征动态调整调度策略。通过引入 QoS 感知机制，框架能够考虑作业的各种性能指标，如延迟、响应时间等，并据此进行智能调度决策。

此外，DRL-Scheduling 还考虑了云环境的动态性和复杂性。它能够实时监控云环境中的资源利用情况和作业状况，并根据实时数据进行智能调度决策的优化。框架还具备自适应性，能够根据不同的应用程序和 QoS 要求进行灵活调整。

总之，DRL-Scheduling 旨在通过基于深度强化学习的智能作业调度框架，在云环境中提供智能、QoS 感知的作业调度策略，以优化应用程序的性能和服务质量。

文献 09 FECBench: A Holistic Interference-aware Approach for Application Performance Modeling

FECBench 是一种综合干扰感知的方法，用于应用程序性能建模。在计算环境中，多个应用程序同时共享计算资源，这可能导致性能干扰和资源竞争，从而影响应用程序的性能。为了更好地理解和预测应用程序在干扰环境下的性能，FECBench 提供了一种全面的解决方案。

FECBench 的核心思想是考虑多个因素并综合考虑干扰感知。它不仅考虑了计算资源的分配情况，还将网络带宽、存储延迟等因素纳入考虑。这种综合考虑的方法可以更准确地建模应用程序的性能，因为性能不仅仅取决于计算资源，还受到其他因素的影响。

在干扰感知方面，FECBench 特别关注多租户云环境中的应用干扰问题。它引入了干扰感知机制，以更准确地模拟干扰对应用程序性能的影响。通过对不同应用程序和资源之间的干扰进行建模，FECBench 可以更好地预测应用程序在干扰环境下的性能表现。

具体而言，FECBench 使用实验和测量数据来建立性能模型，并使用机器学习技术进行预测和优化。通过对大量数据进行训练和分析，它能够捕捉不同因素对应用程序性能的影响，并使用建立的模型为应用程序提供准确的性能预测。

使用 FECBench 进行应用程序性能建模有几个优势。首先，它的综合性考虑了多个因素，提供了更全面和准确的性能模型。其次，干扰感知机制使其能够更好地模拟实际环境中的性能干扰，增强了性能预测的准确性。此外，FECBench 的机器学习技术还能够根据新的数据进行实时优化，提供更准确的建议和决策。

总之，FECBench 是一种综合干扰感知的方法，用于应用程序性能建模。通过综合考虑多个因素，并引入干扰感知机制和机器学习技术，FECBench 可以为应用程序提供准确的性能预测，并帮助优化应用程序在干扰环境下的性能表现。

文献 10 Holistic VM Placement for Distributed Parallel Applications in Heterogeneous Clusters

主要内容是关于异构集群中分布式并行应用的综合虚拟机（VM）部署策略。

在分布式并行应用中，任务的并行执行需要多个虚拟机来协同工作。在异构集群中，不同节点的计算能力和资源特性存在差异，因此，如何将虚拟机合理地部署在集群中，以获得良好的性能和资源利用率就成为一个关键问题。"Holistic VM Placement" 提出了一种综合性的虚拟机部署策略，以优化分布式并行应用的性能。

该方法综合考虑了任务之间的通信开销、节点之间的网络带宽、计算资源和存储容量等因素。通过分析应用程序的通信模式和需求，以及节点的性能特征，该方法可以确定最佳的虚拟机部署方案。它考虑了不仅仅是单个任务的性能优化，还注重整体系统的性能和资源利用率的综合优化。

这种综合虚拟机部署策略的目标是最小化并行应用的总体执行时间，并提高集群资源的利用效率。通过优化虚拟机的部署位置和调度策略，该方法可以减少节点之间的通信延迟和拥塞情况，改善任务之间的协同性能和整体应用的响应能力。

此外，"Holistic VM Placement" 还考虑了集群中节点的负载均衡问题。它会根据节点的负载情况，动态地调整虚拟机的部署和迁移，以确保节点负载的均衡和更好的资源利用率。

综上所述，"Holistic VM Placement for Distributed Parallel Applications in Heterogeneous Clusters" 提供了一种综合性的虚拟机部署策略，以优化并行应用在异构集群中的性能和资源利用率。通过综合考虑通信开销、网络带宽、计算资源和存储容量等因素，并实现节点负载均衡，该方法可以提供更好的应用性能和资源管理。

文献 11 OMBM: Optimized Memory Bandwidth Management for Ensuring QoS and High Server Utilization

主要介绍了一种针对优质服务（QoS）和高服务器利用率的优化内存带宽管理方法。

在当今的数据中心环境中，服务器通常需要同时处理多个任务，并共享有限的内存带宽资源。这可能导致不同任务之间的内存带宽竞争，从而影响应用程序的性能和用户体验。为了解决这个问题，OMBM 提出了一种优化的内存带宽管理方法，以确保 QoS 并提高服务器的利用率。

OMBM 的核心思想是根据每个任务的重要性和优先级，动态分配和管理内存带宽资源。该方法综合考虑了任务的 QoS 要求、服务器的内存带宽容量以及当前任务的重要性。通过智能地调整内存带宽的分配，OMBM 可以在满足任务的 QoS 需求的同时，尽可能地提高服务器的利用率，确保系统的整体性能。

OMBM 具有自适应性和可扩展性。它能够实时监测任务的 QoS 状态和内存带宽使用情况，并根据实时数据进行动态调整。此外，OMBM 采用了一系列优化策略，如任务迁移、带宽限制和队列管理等，以优化内存带宽的使用效率和任务调度的决策过程。

通过使用 OMBM 进行内存带宽管理，可以获得多个方面的好处。首先，它可以确保关键任务和服务的 QoS 要求得到满足，提高用户体验和应用程序性能。其次，优化的内存带宽管理可以提高服务器的利用率，减少资源浪费，增加数据中心的效率和经济性。

总之，"OMBM: Optimized Memory Bandwidth Management for Ensuring QoS and High Server Utilization" 提供了一种针对 QoS 和高服务器利用率的优化内存带宽管理方法。通过动态分配和管理内存带宽资源，该方法可以在满足任务的 QoS 需求的同时，最大程度地提高服务器的利用率和系统性能。

文献 12 OMBM-ML: 保证服务质量和提高服务器利用率的有效内存带宽管理

是一篇研究论文，提出了一种在服务器系统中管理内存带宽以提高服务质量(QoS)和提高服务器利用率的新方法。

本文首先强调了对基于云的服务的不断增长的需求以及对服务器系统中有效资源利用的需求。它强调内存带宽是服务器应用程序的关键资源，并提出了一种基于机器学习(ML)的方法，称为OMBM-ML。

OMBM-ML 利用 ML 技术来预测服务器系统上运行的不同应用程序所需的内存带宽。通过准确预测内存带宽，OMBM-ML 可以动态分配内存资源并对其进行优先级排序，以确保关键应用程序的QoS，并最大限度地提高服务器的总体利用率。

然后，本文深入研究了 OMBM-ML 的技术细节。它引入了一个三步流程: 分析、培训和分配。在分析期间，系统收集有关应用程序内存访问模式和性能指标的数据。然后使用这些数据在下一步中训练 ML 模型。机器学习模型根据每个应用程序的特定特征和工作负载学习预测所需的内存带宽。最后，在分配步骤中，OMBM-ML 根据预测的需求动态地将内存资源分配给不同的应用程序。

实验结果证明了 OMBM-ML 的有效性。这些实验是在一个真实的云环境中使用一组不同的应用程序进行的。结果表明，与传统的内存分配方法相比，OMBM-ML 显著提高了服务器利用率并保证了 QoS。

总之，"OMBM-ML: 确保 QoS 和提高服务器利用率的高效内存带宽管理" 介绍了一种新的基于 ml 的方法来管理服务器系统中的内存带宽。它强调了内存带宽对服务器应用程序的重要性，并演示了 OMBM-ML 如何动态分配内存资源以增强 QoS 和提高服务器利用率。

文献 13 Providing High and Controllable Performance in Multicore Systems Through Shared Resource Management

该论文探讨了在多核系统中实现高性能和可控性的挑战，并介绍了一种名为 "共享资源管理" 的方法来解决这些挑战。

首先，论文指出了多核系统中出现的资源共享问题。在多核系统中，多个核心共享集中的资源，例如缓存、内存带宽和其他高速缓存。当多个线程同时访问这些共享资源时，可能会引发争用和冲突，从而导致性能下降和不可预测的行为。

接下来，论文介绍了共享资源管理的基本原则和方法。该方法主要包括两个方面：资源隔离和资源调度。资源隔离的目标是将多个线程或任务分配到不同的资源池中，以避免资源争用。资源调度的目标是有效地调度线程或任务的执行，以最大程度地提高系统的整体性能。

然后，论文详细介绍了一些具体的共享资源管理技术。其中包括缓存隔离、并发控制、调度器增强和任务划分等。这些技术旨在通过减少资源冲突和优化资源的使用来提高系统的性能和可控性。

最后，论文通过一系列的实验证明了共享资源管理在多核系统中的有效性和实用性。实验结果表明，通过合理地管理共享资源，可以提高系统的吞吐量、响应时间和可预测性。

总的来说，《Providing High and Controllable Performance in Multicore Systems Through Shared Resource Management》这篇论文介绍了一种在多核系统中实现高性能和可控性的方法。通过资源隔离和资源调度等技术，可以解决资源共享引发的争用和冲突问题，并提高系统的整体性能和可预测性

文献 14 QUALITY-OF-SERVICE-AWARE SCHEDULING IN HETEROGENEOUS DATACENTERS WITH PARAGON

是一篇关于在异构数据中心中使用 Paragon 进行服务质量感知调度的

该论文介绍了在异构数据中心中使用 Paragon 进行服务质量感知调度的方法。异构数据中心是指由不同类型的计算资源（例如 CPU、GPU、FPGA 等）组成的数据中心，具有不同的性能和处理能力。

首先，论文介绍了异构数据中心中面临的挑战。由于不同类型的计算资源具有不同的性能特征和处理能力，传统的调度算法往往无法满足不同应用的需求，导致资源利用率低和服务质量下降。

接下来，论文介绍了 Paragon 调度器的设计原理和关键特性。Paragon 是一种服务质量感知的调度器，它能够根据不同应用的需求和优先级来动态地分配资源和调整任务的执行顺序。它通过多维度的资源评估和动态调整策略，实现了更好的性能和服务质量。

然后，论文详细介绍了 Paragon 调度器的实现细节和算法。其中包括任务优先级的设置、资源评估的方法、任务调度的策略等。这些技术和算法的目标是在异构数据中心中实现更好的服务质量和资源利用率。

最后，论文通过实验评估了 Paragon 调度器在异构数据中心中的性能和效果。实验结果表明，Paragon 能够有效提高服务质量和资源利用率，提供更好的用户体验和系统性能。

总的来说，《QUALITY-OF-SERVICE-AWARE SCHEDULING IN HETEROGENEOUS DATACENTERS WITH PARAGON》这篇论文介绍了一种在异构数据中心中实现服务质量感知调度的方法。通过动态分配资源和调整任务执行顺序，Paragon 调度器能够提供更好的服务质量和资源利用率。

文献 15 基于贝叶斯网的虚拟机服务质量预测

该论文提出了一种基于贝叶斯网的方法来预测虚拟机的服务质量。在云计算环境中，虚拟机的性能和服务质量对于用户应用的响应速度和可靠性至关重要。因此，准确地预测虚拟机的服务质量是一个重要的研究方向。

首先，论文介绍了贝叶斯网的基本概念和原理。贝叶斯网是一种概率图模型，它能够表示和推理变量之间的概率依赖关系。在虚拟机服务质量预测中，贝叶斯网可以用于建模虚拟机性能参数、环境变量和应用特征等之间的关系。

接下来，论文详细介绍了基于贝叶斯网的虚拟机服务质量预测方法。该方法主要包括数据收集、贝叶斯网络模型构建和预测三个步骤。首先，通过收集虚拟机性能参数、环境变量和应用特征等数据。然后，使用这些数据来构建贝叶斯网络模型，通过学习变量之间的概率依赖关系。最后，利用构建好的贝叶斯网络模型进行虚拟机服务质量的预测。

然后，论文介绍了实验设计和评估方法。通过使用真实的虚拟机服务质量数据集，评估了基于贝叶斯网的预测方法的准确性和可靠性。实验结果表明，该方法能够准确地预测虚拟机的服务质量，为用户提供可靠的性能估计。

最后，论文讨论了未来的研究方向和应用场景。虚拟机服务质量预测在云计算和大规模分布式系统中具有广阔的应用前景，可以用于资源调度、负载均衡、故障诊断等方面。

总的来说，《基于贝叶斯网的虚拟机服务质量预测》这篇论文介绍了一种利用贝叶斯网来预测虚拟机服务质量的方法。通过构建贝叶斯网络模型，该方法能够准确地预测虚拟机的性能和服务质量，为云计算环境中的应用提供可靠的性能估计。

文献 16 云数据中心下基于服务组件级控制的资源利用率优化研究

是一篇关于在云数据中心中通过服务组件级控制来优化资源利用率的研究论文。

该论文主要研究了如何通过对服务组件级别进行控制，来优化云数据中心的资源利用率。云数据中心是一个集中管理大量计算、网络和存储资源的环境，为用户提供可扩展的服务和应用。

论文首先介绍了云数据中心资源利用率优化的重要性和现有挑战。资源利用率的提高可以降低数据中心的能耗和成本，并提高服务的可用性和性能。然而，传统的资源管理方法通常只关注整体资源利用率，缺乏对服务组件级别的精细控制，造成了资源浪费和性能瓶颈的问题。

接着，论文提出了基于服务组件级控制的资源利用率优化方法。该方法通过将服务拆分为多个组件，每个组件具有不同的资源需求和优先级。然后，根据实时的负载情况和性能要求，动态地分配和调度资源给不同的服务组件，以实现资源的最优利用和性能优化。

论文进一步详细介绍了基于服务组件级控制的关键技术和算法。其中包括组件拆分和资源需求建模、资源调度策略和优化算法等。这些技术和算法旨在实现资源的细粒度控制和自适应调整，以满足不同服务组件的资源需求，并优化整体资源利用率和性能。

最后，论文通过仿真实验和性能评估来验证该方法的有效性和可行性。实验结果表明，基于服务组件级控制的资源利用率优化方法能够在满足服务质量要求的前提下，降低资源浪费，提高云数据中心的资源利用率和性能。

总的来说，《云数据中心下基于服务组件级控制的资源利用率优化研究》这篇论文研究了通过服务组件级别控制来优化云数据中心的资源利用率。通过细粒度地管理和调度服务组件的资源需求，该方法能够降低资源浪费，提高资源利用率，并优化云数据中心的性能和可用性。

文献 17 云数据中心虚拟机性能干扰预测与部署研究

是一篇关于在云数据中心中预测和部署虚拟机性能干扰的研究论文。

该论文的主要研究目标是预测和部署虚拟机性能干扰，以提高云数据中心中虚拟机的性能和可靠性。在云计算环境中，虚拟机是用户应用的基本运行单元，虚拟机之间的性能干扰可能会导致服务质量下降和性能波动。

论文首先介绍了云数据中心中虚拟机性能干扰的原因和现有的挑战。虚拟机同时运行在共享的物理资源上，如 CPU、内存和网络带宽等，当虚拟机之间的资源需求和利用存在差异时，会引发性能干扰。传统的资源调度方法往往无法准确预测和解决虚拟机性能干扰问题。

接着，论文介绍了虚拟机性能干扰预测与部署的研究方法。通过分析虚拟机的资源需求、应用特征和调度策略等因素，建立预测模型来预测虚拟机性能干扰的可能性和影响程度。然后，根据预测结果采取相应的部署策略，如选择合适的主机、分配适当的资源等，以减轻虚拟机性能干扰。

论文进一步详细介绍了虚拟机性能干扰预测与部署的关键技术和算法。其中包括虚拟机性能特征提取、性能干扰预测模型构建、部署策略评估和优化等方面。这些技术和算法旨在提高性能干扰预测的准确性和可靠性，并优化虚拟机的部署方案。

最后，论文通过实验证明了虚拟机性能干扰预测与部署研究的有效性和实用性。通过使用真实的云数据中心环境和虚拟机工作负载数据，评估了预测模型和部署策略的性能。实验结果表明，该研究能够有效减轻虚拟机性能干扰，提高虚拟机的可用性和性能。

总的来说，《云数据中心虚拟机性能干扰预测与部署研究》这篇论文研究了在云数据中心中预测和部署虚拟机性能干扰的方法。通过建立预测模型和采取相应的部署策略，该研究能够减轻性能干扰，提高虚拟机的性能和可靠性。