一种基于网络感知的虚拟机再调度算法

罗刚毅 钱柱中 陆桑璐

(南京大学计算机软件新技术国家重点实验室 南京 210093)

(南京大学计算机科学与技术系 南京 210093)

摘 要 有效的虚拟机调度策略能够提高数据中心的资源利用率,降低运行时能耗.现有调度算法综合考虑了虚拟机在 CPU、内存和网络方面的需求,通过合理部署虚拟机,以期最小化计算、存储与网络的代价.然而,在线的虚拟机部署策略较少考虑由于虚拟机退出所造成的资源利用率下降与网络延迟上升的问题.为此,文中深入研究面向网络感知的周期性资源重配置问题,提出了面向网络感知的虚拟机再调度算法,通过适当的虚拟机迁移,提高部署在虚拟机上任务的性能以及数据中心整体的网络通信效率.算法通过尽可能低代价的虚拟机迁移来提高虚拟机之间的网络通信能力,以提升虚拟机组的整体运行效率,并保持物理机占用但不显著提高.作者通过两个测试平台在真实环境中验证了算法的有效性;通过真实的数据集和模拟实验,在多种虚拟机部署算法下,对比了应用虚拟机再调度算法前后虚拟机的部署效果,验证了该算法能够以较小的代价使得高网络通信代价的任务数明显减少,虚拟机组的网络通信能力显著提高.

关键词 云计算;数据中心;网络感知;虚拟机再调度

中图法分类号 TP311 **DOI 号** 10.3724/SP.J.1016.2015.00932

A Network-Aware VM Re-Scheduling Algorithm

LUO Gang-Yi QIAN Zhu-Zhong LU Sang-Lu

(*State Key Laboratory for Novel Software Technology, Nanjing University, Nanjing* 210093)

(*Department of Computer Science and Technology, Nanjing University, Nanjing* 210093)

Abstract An effective and efficient VM scheduling algorithm can improve utilization rate of physical servers and lower energy cost. Current VM allocation algorithms focus on the requirement of CPU, Memory and network bandwidth which trying to allocate VM into physical servers in a low cost way. However, when the jobs are finished, related VMs quit from the system, which leads to the decline of resource utilization and the increasing of transmission delay. This paper studied the network-aware resource reconfiguration problem and proposed a network-aware VM re-scheduling algorithm based on VM live migration. This algorithm focuses on improving communication ability among VMs to promote the overall performance in a way of low migration cost and little extra physical used. Two test beds are deployed in a real environment to examine the effectiveness of the VM reconfiguration algorithm. We compared the result of before and after using our algorithm with different consolidation algorithms based on the real workload data and simulation. The results show that our algorithm considerably reducing the amount of high-delay jobs and improving the communication ability among VMs only with very small cost.

Keywords cloud computing; datacenter; network-aware; re-scheduling

---

收稿日期:2013-05-16;最终修改稿收到日期:2014-10-31.本课题得到国家自然科学基金(61073028,61202113,61021062)、江苏省自然科学基金项目(BK2011510)、江苏省科技支撑项目(BE2013116)资助.**罗刚毅**,男,1989 年生,硕士研究生,主要研究方向为虚拟化技术.E-mail:luogangyi@dislab.nju.edu.cn.**钱柱中**,男,1980 年生,博士,副教授,中国计算机学会(CCF)会员,主要研究方向为分布式系统、虚拟化技术.**陆桑璐**,女,1970 年生,博士,教授,博士生导师,主要研究领域为分布式计算和计算机网络.

1 引　言

随着虚拟化技术的发展以及云计算概念的普及，越来越多的用户选择通过向数据中心租用计算资源的方式来完成其工作任务而非自己购买物理设备.为提高资源利用率，如何高效地将用户请求的资源合理部署到各个物理服务器上是数据中心的管理者需要解决的重要问题之一.

在云计算模式下，用户向数据中心申请一组虚拟机，并指明各个虚拟机对 CPU、内存与网络通信的需求.数据中心运用虚拟机调度算法，为这组虚拟机选择合适的物理机运行.

基于 CPU、内存的调度算法一方面关注尽可能地减少物理服务器上的 CPU、内存等碎片，以容纳更多虚拟机，提高数据中心中服务器的整体利用率[1-2]；另一方面则关注资源的预留，以应对虚拟机负载的变化[3-4].这样资源配置问题通常可转化为多维背包问题，并通过启发式的算法求解.

保证虚拟机的运行性能也是虚拟机调度算法需要关注的问题之一.除了 CPU、内存等资源之外，网络通信是影响虚拟机性能的重要因素.当前的数据中心由千台以上的物理服务器构成，服务器之间通常通过树形的拓扑结构连接，因此其端到端的网络通信能力随着物理机分布的不同有着很大的差异，也就造成了部署在物理机上的虚拟机之间的网络通信能力的差异.基于网络感知的虚拟机部署，即部署虚拟机时，应考虑其相互之间的网络通信代价，试图将用户申请的一组虚拟机部署到同一机架的物理机上以减少其通信延迟以及整个数据中心的通信流量.

上述研究工作都关注于用户请求到达时，如何合理、高效地部署虚拟机.然而，系统运行过程中，如果不执行周期性的再调度，则会因为任务结束部分虚拟机的退出，导致资源碎片和网络通信的代价升高，算法的性能就将随着时间的推移出现下降.尤其是大部分的虚拟机部署算法并没有考虑到虚拟机之间的网络通信代价，这些算法在长时间运行后，会将任务所需的虚拟机部署到由于先前任务完成导致的虚拟机退出时留下的槽位上，而这些位置所在的物理机并不一定属于同一机架，从而就导致了这部分虚拟机之间的通信代价大大提高，不仅影响了在虚拟机上执行的任务的性能，也影响了整个数据中心的网络通信能力.为此，本文深入研究了面向网络感知的周期性资源重配置问题，通过实验测试了网络通信以及动态迁移对部署在虚拟机上的任务的影响，由此提出了一种有效的、低代价的基于网络感知的虚拟机再调度算法.在尽可能不影响虚拟机性能以及不造成数据中心网络拥塞的条件下，通过适当的虚拟机迁移来提高虚拟机之间的网络通信能力，提升虚拟机组的整体运行效率.

本文第 2 节介绍虚拟机部署问题的相关工作；第 3 节介绍数据中心网络拓扑，分析网络对虚拟机组性能影响；第 4 节测试动态迁移在不同条件下对不同类型的任务可能造成的影响；第 5 节给出再调度问题的形式化定义；第 6 节提出基于贪心策略的再调度算法；第 7 节通过实验对比验证算法的性能；第 8 节总结全文的工作.

2 相关工作

虚拟机部署与整合问题的研究可分为很多类，每个大类所关注的重点各不相同.例如一些研究关注如何最大化资源利用率，如何节能；而另一些研究则关注如何提高虚拟机之间使用资源的公平性，提高虚拟机性能等方面.

以最大化资源利用率为目标的虚拟机部署算法，在初次为虚拟机分配资源时，根据其对资源的需求特点和系统所剩资源的分布，找出一种最为合适的虚拟机映射方式，以提高物理计算资源的利用率，这类问题通常被形式化为多维的装箱问题，并通过启发式算法求解[5].而虚拟机的动态迁移技术使得可以在不影响虚拟机运行的情况下，从一个物理节点移动到另一个物理节点，这就使得我们能够将一些利用率较低的节点上的虚拟机迁移出去，然后将这些物理服务器关闭或使其待机，提高资源的利用率.Xu 等人[6]提出了一种两层控制系统，第 1 层将工作负载映射到虚拟机，第 2 层再将虚拟机映射到物理机.Xu 等人同时将最小化资源浪费率、最小化能耗、最小化散热成本作为目标，将问题形式化为多目标优化，并提出了一种改进的遗传算法求解该问题.Jayasinghe 等人[7]研究了虚拟机部署中的 3 种限制：(1) CPU、内存需求限制；(2) 虚拟机可用性限制；(3) 虚拟机通信能力限制.然后提出了一种限制感知(constraint-aware)的虚拟机部署问题，并提出了一种近似度为 4 的近似算法求解该问题.Chen 等人[8]提出了一个以提高云计算提供商收益为目标的虚拟机部署和重配置的框架.Marzolla 等人[9]提出

了一种完全去中心化的虚拟机整合算法 V-MAN，V-MAN 通过一个简单的对话协议在服务器之间交换信息，根据获取到的信息进行以最大化资源利用率为目标的虚拟机整合.

面向 SLA 的虚拟机调度策略根据用户的 QoS 要求(响应时间、单位时间完成的计算量等)与虚拟机的计算能力(包括 CPU、内存、I/O 速度等)进行映射，在满足需求的条件下，进行最小的资源分配，再将分配到的资源与物理服务器进行映射. Bobroff 等人[10]提出了一种动态的虚拟机整合技术，首先对工作负载的类型进行分类，对最受益于动态迁移的负载类型进行标记；然后提出了一种 MFR 算法，每隔一段时间迭代进行. 算法分为三步：首先分析历史的虚拟机负载数据，然后对未来的负载作出预测，最后对虚拟机与物理机进行重新映射. Nguyen Van 等人[1]提出了一种 SLA 感知的虚拟资源管理策略，该策略综合了高层 SLA 的需求与资源部署的代价，由管理人员设定两部分的权重参数，算法支持异构的应用和负载类型. Breitgand 等人[11]则将 SLA 满足程度看成参数，将云计算提供商的收益作为目标，把问题形式化为组合最优化问题.

网络感知的虚拟机调度，即将虚拟机之间的网络通信能力作为调度的参考因素，以期提高虚拟机之间的网络通信带宽，降低通信时延. Alicherry 等人[12]提出了基于网络感知的虚拟机部署问题，即部署虚拟机时，应考虑其相互之间的网络通信代价，试图将用户申请的一组虚拟机部署到同一机架的物理机上以减少其通信延迟以及整个数据中心的通信流量. Meng 等人[13]研究了部署在数据中心中任务的通信流量分布，进一步提出了基于通信流量的虚拟机部署算法，算法分为两步，首先对虚拟机按流量进行聚类，同时对物理机进行聚类，然后将聚类后的虚拟机组映射到聚类后的物理机组中. Jiang 等人[14]则综合考虑了虚拟机之间的通信模式和路由策略. 由于在 BCube[15]等新型网络结构中，管理员可以手动调整路由，因此 Jiang 等人在路由策略可自定义的条件下，以最小化通信代价为目标，进行虚拟机部署. Biran 等人[16]提出了最小割感知的虚拟机调度，目标是最小化最大的割流量. 并给出了两个启发式算法求解该问题：两阶段的递归分割算法(2PCCRS)和贪心算法. Steiner 等人[17]则研究了在分布式数据中心中，网络感知的虚拟机部署问题.

3 网络对虚拟机组性能的影响

无论是部署 Web 服务、执行 Map-Reduce 等分布式计算任务或是执行高性能计算任务，都需要数据中心提供多台虚拟机协同为用户提供服务. 在这种场景中，用户的任务就会涉及到虚拟机之间大量的网络通信. 因此，虚拟机之间的网络通信能力成为了影响虚拟机群组性能的重要因素之一.

3.1 数据中心的网络拓扑

数据中心通常由上千至上万台服务器组成，这些服务器首先连接到机架交换机，机架交换机连接到聚集层的群组交换机，聚集层的交换机再与数据中心核心交换机相连[18]. 这类网络拓扑结构可抽象成图 1 所示.

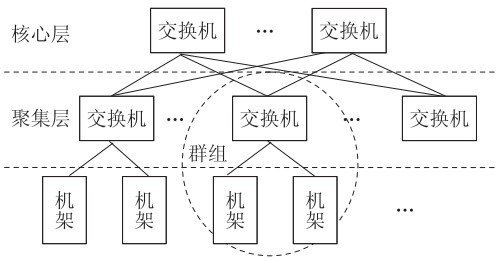

图 1 数据中心的网络拓扑

在图 1 的这种树形结构中，连接在同一机架交换机下的服务器拥有最高网络通信带宽；当服务器之间的网络通信需要经过聚集层交换机时，带宽一般会下降至机架交换机的 1/4 至 1/8[18]；当服务器之间的通信需要经过核心层的交换机时，通信带宽将进一步下降，通信的时延会增加. 因此，在这种网络拓扑的环境下，将涉及大量网络通信的虚拟机部署在通信带宽较大、通信时延较小的物理机上，就显得十分必要.

3.2 网络通信对 Web 服务性能的影响

典型的 Web 服务多采用 3 层架构，为了保证服务的性能，Web 服务的运营者通常会选择将这 3 层分别部署在若干个计算机节点上，而在云计算场景中，这些服务就会被部署在若干台虚拟机中. 这些虚拟机之间会涉及大量的数据通信，而虚拟机内部的通信的延迟则会最终反映到用户访问 Web 服务的延迟上.

为了研究虚拟机内部的网络通信对最终用户访问网站效果的影响，我们选用了 RUBiS① 作为测试

① RUBiS：Rice University Bidding System；http://rubis.ow2.org/

平台. RUBiS 是一个开源的竞价拍卖网站项目，该项目包含了一个小型拍卖网站系统以及一组标准的测试用例生成工具和性能分析工具. 我们在实验室的基础设施云平台（平台基于 Xen 虚拟机技术，由 5 个计算节点和 1 个存储节点构成）上部署了该测试平台，选用的虚拟机均为 2 GB 内存，2 个虚拟 CPU. 我们选择将虚拟机部署在多种不同网络条件下进行性能测试（由于数据中心的网络拓扑多为树形结构，因此将数据通信经过的子树层数作为分类），测试的指标为用户访问的平均延迟. 测试结果如表 1 所示.

表 1　虚拟机通信对 Web 服务性能的影响

路由层数	平均延迟/ms
0 层	35
1 层	35
2 层	48
3 层	72

实验结果表明虚拟机之间通信能力将对部署在其上的 Web 服务产生直接影响. 当虚拟机之间的通信需要经过较多的路由层数时，虚拟机群组的运行效率将下降，最终将导致用户访问部署在虚拟机上的 Web 服务的时延增加. 因此，对于此类需要多台虚拟机协同工作的 Web 服务，保证其使用的虚拟机之间的通信能力就显得十分必要.

3.3　网络通信对分布式计算任务性能的影响

短时间租用数据中心的计算资源去完成一个分布式计算任务也是云计算中主要的应用场景之一. 在这种场景中，任务被分割为多个子任务由多台虚拟机进行协同计算，计算过程中会产生大量的数据通信. 因此，虚拟机数据通信的能力成为影响任务完成时间的重要因素之一.

我们选用 PUMA① 作为标准测试工具，部署在 4 台虚拟机上，每台虚拟机均为 2GB 内存，2 个虚拟 CPU. 其中一个为 Master 节点，另外 3 个为 Slaver 节点. 我们将虚拟机部署在多种网络条件下进行测试，测试结果如表 2.

表 2　虚拟机通信对分布式计算任务的性能影响

路由层数	完成时间/s
0 层	157
1 层	154
2 层	168
3 层	175

实验结果表明虚拟机之间的网络通信能力会对涉及到大量数据交换的分布式任务产生明显的影响. 当虚拟机之间通信需要经过的路由层数越小时，其可用带宽就越大，通信时延就越小，任务执行的效率就越高（表 2 中，出现路由层数为 0 时任务完成时间比层数为 1 时任务完成时间长是由于将全部虚拟机部署在了同一物理机上，造成磁盘 I/O 竞争，使得任务完成时间延长）.

4　动态迁移对服务性能的影响

当一个用户向数据中心申请一组虚拟机用于部署其需要的应用时，很可能出现由于当时硬件条件的限制，这组虚拟机被分配到不同机架甚至不同群组之中的物理服务器上. 而如果这个用户所部署的应用需要虚拟机之间进行大量数据通信，这种跨机架、跨群组之间的部署方式不仅会造成用户所部署的应用的性能下降，一旦发生网络拥塞，还将影响整个数据中心的网络通信. 因此，在这种情况下就很有必要在不影响虚拟机正常使用的同时将其迁移到合适的位置.

动态迁移技术能够在不影响虚拟机正常使用的前提下，将其移动到目标位置. 然而，在物理服务器 CPU 处于高负载状态，动态迁移会占用待迁移虚拟机本身的 CPU 资源，该虚拟机上运行服务的性能就会受到影响. 此外，整个迁移过程中，虚拟机会有短暂的停机（通常在几百毫秒至一千毫秒）[11,19-20]，这个短暂停机时间可能会造成分布式的任务处理失败，或 Web 服务中数据包的丢失. 为此，我们进行了详细的测试.

4.1　物理机 CPU 使用率对动态迁移的影响

我们在 Web 服务和分布式计算这两种场景中，选用和第 3 节中相同的测试平台 RUBiS 和 PUMA，在不同的物理机 CPU 使用率时进行动态迁移测试，测试用的物理机有 24 个核心，每个核心均为 2.4GHz，虚拟机为 2GB 内存，2 个虚拟 CPU. 测试结果如图 2 所示.

从图 2 中可以看出，在源物理机 CPU 利用率小于 96%时，迁移发生时对部署在虚拟机上的应用影响很小，几乎可以忽略不计，只有当源物理机的 CPU 利用率超过 99%时，才会对部署在虚拟机上的应用产生明显影响. 因此，为了不影响虚拟机的性能，我们只选择在物理机 CPU 利用率不大于 95%（剩余物理核心数大于 1 个）时进行动态迁移.

① PUMA. http://web. ics. purdue. edu/fahmad/benchmarks. htm

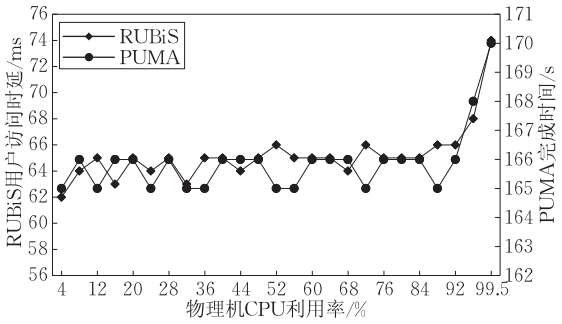

图 2 物理机 CPU 利用率对迁移时服务性能的影响

4.2 网络环境对动态迁移的影响

虚拟机的动态迁移需要将虚拟机的内存从源宿主机复制到目的宿主机，在这个过程中会涉及到大量的数据传输. 因此，源宿主机和目的宿主机之间的网络通信带宽直接影响了动态迁移的完成时间. 表 3 显示了在不同带宽下，迁移完成的时间和对部署在虚拟机上的任务的影响.

表 3 网络带宽对迁移时服务性能的影响

带宽/(Mb/s)	RUBiS/ms	PUMA/s	迁移耗时/s
未迁移	61	164	—
100	63	165	121
1000	63	165	11

从表 3 的结果中可以看出，尽管在动态迁移进行的过程中，虚拟机上所运行的服务性能有略微下降，但网络带宽只影响了迁移完成所需要的时间，带宽的差异并没有造成部署在虚拟机上的任务在迁移时性能的差异.

5 基于网络感知的虚拟机再调度

在第 4 节中，我们详细讨论了各种因素对动态迁移过程中部署在虚拟机上的服务性能的影响. 第 3 节的结果表明物理机 CPU 的利用率是影响动态迁移过程中部署在虚拟机上的服务性能的主要因素，只要选择在合适的时机，例如当服务器至少剩余 1 个空闲物理 CPU 核心时，进行动态迁移，就可以保证虚拟机运行的服务不受明显影响.

此外，虚拟机动态迁移过程中的对交换机带宽的消耗是一个重要的迁移代价. 为了防止动态迁移过程中大量的数据通信造成网络拥塞，影响正常的虚拟机之间的网络通信，因此，调度算法应当保证在不超过交换机最大交换容量的条件下进行迁移.

5.1 虚拟机调度

假定数据中心共有 M 台物理机和 K 个用户. 我们用 p_i 表示第 i 台物理机，每台物理机的物理资源用向量 $\boldsymbol{H}_i$ 表示. 如果物理机 p_i 正在运行，则记 $S(i)=1$，否则 $S(i)=0$. $\boldsymbol{L}=(0,1,1,\cdots,0)$ 是数据中心中所有路由器或交换机按某一序列组成的一维向量，$\boldsymbol{L}_p=1$ 表示数据通信会经过交换机 p，$\boldsymbol{L}_p=0$ 表示数据通信不会经过交换机 p. $\boldsymbol{L}(i,j)$ 是物理机 p_i 和 p_j 之间的路由向量，显然，$\boldsymbol{L}(i,i)=(0,0,\cdots,0)$. l_{ij} 是物理机 p_i 和 p_j 之间的数据通信经过的路由层数(在树形网络拓扑中)，同时，我们定义 $l_{ii}=0$. $\boldsymbol{Q}$ 是 $\boldsymbol{L}$ 对应的交换机的最大交换容量向量.

对于用户 u_k，他可以向数据中心申请一组虚拟机 G_k，w_k 是这个虚拟机组中虚拟机的数量. G_k 中的每台虚拟机 v_i^k 各自会对 CPU 和内存有明确的需求，这里用向量 $\boldsymbol{R}_i^k$ 表示；向量 $\boldsymbol{R}_i^k$ 可以根据实际需求被扩展为包括了 I/O 操作、网络带宽等其他资源的扩展向量. $\boldsymbol{T}^k$ 是 G_k 中虚拟机之间通信量的矩阵，其大小为 $w_k\times w_k$，矩阵中 $\boldsymbol{T}^k$ 的每一项 t_{ij}^k 是虚拟机 v_i^k 和虚拟机 v_j^k 在 Δt 时间内通信总量. 通信矩阵 $\boldsymbol{T}^k$ 可以由用户给出，或者由数据中心管理者通过监控虚拟机通信来实际测量得到. 我们用 $Z_i^k(m)=1$ 表示虚拟机 v_i^k 被放置在了物理服务器 p_m 上，否则，$Z_i^k(m)=0$. 因此，一个可行的虚拟机部署方式可被表示为

$$\mathcal{Z}=\left\{Z_i^k(m)\mid Z_i^k(m)\in\{0,1\};\sum_{m=1}^{M}Z_i^k(m)=1,\forall(i,k)\right. \tag{1}$$

$$\left.\sum_{k=1}^{K}\sum_{i=1}^{K_w}Z_i^k(m)\cdot\boldsymbol{R}_i^k<\boldsymbol{H}_m\cdot S(m),\forall m\right\} \tag{2}$$

其中，等式(1)保证了每个虚拟机 v_i^k 只被部署在一台物理机上，不等式(2)保证了一台物理机上的所有虚拟机需要的物理资源总和不超过该物理机拥有的资源总量. $G_{ij}^k(Z)$ 是在虚拟机部署策略 Z 下，虚拟机 v_i^k 和虚拟机 v_j^k 的路由函数，因此，$G_{ij}^k(Z)$ 可表示为

$$G_{ij}^k(Z)=\sum_{m=1}^{M}\sum_{n=1}^{M}[\boldsymbol{L}(m,n)\cdot Z_i^k(m)\cdot Z_j^k(n)] \tag{3}$$

而虚拟机 v_i^k 和虚拟机 v_j^k 通信所经过的路由层数可表示为

$$J_{ij}^k(Z)=\sum_{m=1}^{M}\sum_{n=1}^{M}[l_{ij}\cdot Z_i^k(m)\cdot Z_j^k(n)] \tag{4}$$

5.2 虚拟机调度的代价与收益

设在时间 t_0，虚拟机的部署方式表示为 $\{X_i^k(m)\}$，且 $\{X_i^k(m)\}\in\mathcal{Z}$；在完成一次虚拟机的再调度后，新的虚拟机部署方式记为 $\{Y_i^k(m)\}$，且 $\{Y_i^k(m)\}\in\mathcal{Z}$. 如果虚拟机 v_i^k 从物理机 p_m 被迁移到了物理机 p_n 上，则有 $X_i^k(m)\cdot Y_i^k(n)=1$，$m\neq n$. 由于动态迁移

的主要代价取决于虚拟机活动内存的大小，因此我们定义迁移的代价函数为

$$Cost(v_i^k)=\sum_{m=1}^{M}\sum_{n=1}^{M}[X_i^k(m)\cdot Y_i^k(m)\cdot(m-n)\cdot \boldsymbol{L}(m,n)]\cdot M_i^k \quad (5)$$

其中 M_i^k 是虚拟机 v_i^k 活动内存的大小. 因此，对于一个用户 k，它拥有的所有虚拟机的迁移代价和为

$$Cost(G_k)=\sum_{i=1}^{w_k}Cost(v_i^k) \quad (6)$$

对于整个数据中心而言，所有用户的虚拟机迁移造成的总的通信代价为

$$Cost(DC)=\sum_{i=1}^{K}Cost(G_k) \quad (7)$$

显然，$Cost(DC)$ 是一个一维的向量，向量中的每一列代表了动态迁移对某个路由器或交换机造成的通信总量. 而在 Δt 时间内用户 u_k 所有虚拟机的经过各个交换机的通信总量 $\boldsymbol{D}$ 可表示为

$$\boldsymbol{D}(u_k)=\sum_{i=1}^{w_k}\sum_{j=1}^{w_k}t_{ij}^k\cdot G_{ij}^k(X) \quad (8)$$

因此，整个数据中心在 Δt 内，虚拟机部署方式为 $\{X_i^k(m)\}$ 的条件下，所有虚拟机的经过各个交换机或路由器通信总量可表示为

$$\boldsymbol{D}=\sum_{k=1}^{K}\boldsymbol{D}(u_k) \quad (9)$$

为了保证动态迁移的过程中，产生的数据通信不会造成网络拥塞，定义如下限制条件：

$$[Cost(DC)+\alpha\cdot\boldsymbol{D}]/\Delta t<\boldsymbol{Q} \quad (10)$$

其中系数 α 是自定义的调节因子，由于 $\boldsymbol{D}$ 是根据历史数据得出或直接由用户给出的，并不能完全刻画当前的真实情况，因此为了提高容错性，选取 $\alpha\in[1.5,2]$，调度者也可根据实际的预测模型以及数据的波动程度调节 α.

第 3 节的实验结果表明一个任务所需要的虚拟机所处的物理机的位置，会影响这些虚拟机之间网络通信的代价，从而最终影响部署在虚拟机上的任务的执行效率. 因此，为了刻画迁移为用户带来的收益，我们定义了一个由虚拟机之间网络通信代价决定的收益函数，其定义为

$$Benefit(G_k)=\sum_{i=1}^{w_k}\sum_{j=1}^{w_k}\frac{t_{ij}^k}{B(J_{ij}^k(Y))} \quad (11)$$

其中，Y 为虚拟机再调度后的虚拟机部署方式，$\{Y_i^k(m)\}\in\mathcal{Z}$，$B(J_{ij}^k(Y))$ 是数据通信所经过的路由层数的函数. 根据思科的数据中心网络设计指导，一个典型的数据中心的树形网络拓扑中①，端到端的通信带宽每经过一层就下降至原来的 1/4 至 1/8. 因此，为了简化计算，我们定义 $B(x)=5^x$.

5.3　网络感知的虚拟机再调度问题

由 5.1 节及 5.2 节中的定义可知，$\{X_i^k(m)\}$ 为某一时刻虚拟机在数据中心中的部署方式，而 $\{Y_i^k(m)\}$ 为我们进行再调度后的虚拟机部署方式. 在再调度的过程中，必然涉及到虚拟机的迁移，也就会带来数据通信代价，我们将这个代价用 $Cost(DC)$ 来表示，而 $Benefit(G_k)$ 则是再调度后的虚拟机之间的通信收益. 在定义了收益和代价之后，我们将问题定义为在满足迁移代价不超过数据中心交换机容量的限制下，寻找一个可行的虚拟机部署策略，使得收益值最大. 因此，问题可形式化为

$$\begin{aligned}&\text{Max}\ \sum_{i=1}^{K}Benefit(G_k)\\&\text{s.t.}\ \ S(m)\in\{0,1\},\ \forall m\le M\\&\qquad\{Y_i^k(m)\}\in\mathcal{Z},\ \forall i\le w_k,\ k\le K,\ m\le M\\&\qquad[Cost(DC)+\alpha\cdot\boldsymbol{D}]/\Delta t<\boldsymbol{Q}\end{aligned}.$$

上述优化问题的目标即尽可能地在不影响部署在虚拟机的服务的性能以及不造成数据中心网络拥塞的条件下，通过动态迁移来最大化虚拟机之间的网络通信收益. 考虑到全局优化有可能导致局部出现恶化，即可能出现牺牲部分用户的利益来达到全局最优. 因此，在真实的数据中心应用场景中，必须保障我们的算法不会牺牲个别用户的利益，因此，我们对原优化加入了额外的限制条件：

$$\sum_{i=1}^{w_k}\sum_{j=1}^{w_k}\frac{t_{ij}^k}{B(J_{ij}^k(Y))}\ge\sum_{i=1}^{w_k}\sum_{j=1}^{w_k}\frac{t_{ij}^k}{B(J_{ij}^k(X))},\quad\forall i\le w_k,\ k\le K \quad (12)$$

上式的限制保证了经过再调度之后，每个用户都会获得收益.

6　网络感知的虚拟机再调度算法

由于我们定义的收益函数是关于部署策略 Y 的非线性函数，因此整个优化问题是非线性的整数规划，而 Murty 等人[21]证明了非线性规划问题是 NP 难问题，故作为子问题的非线性整数规划问题也是 NP 难问题. 为了在可接受的时间范围内，得到较优的可行解，本文基于贪心策略设计了一个启发式算法以及一个基于多次迭代的随机算法与贪心算

① Cisco Data Center Infrastructure 2.5 Design Guide. http://www.cisco.com/univercd/cc/td/doc/solution/dcidg21.pdf

法进行对比.基于贪心策略的再调度算法优先调度收益值提升空间最大的任务,效果较好,效率也较高,而基于多次迭代的随机再调度算法则随机选定任务的调度顺序,然后多次迭代选出收益值最大的一次作为最终调度,算法执行效率随着迭代次数增加而降低.算法中会使用到的符号如表 4 所示.

表 4 符号表

符号	含义
M	物理机总数
K	用户/任务总数
v_i^k	第 k 个用户的第 i 台虚拟机
$\{\boldsymbol{H}_i\}$	物理机资源的集合
$\boldsymbol{L}$	所有路由器组成的向量
$\boldsymbol{Q}$	$\boldsymbol{L}$ 对应的交换机的最大交换容量
J_{ij}^k	v_i^k 和 v_j^k 通信所经过的路由层数
G_{ij}^k	v_i^k 和 v_j^k 通信所经过的路由函数
$\{\boldsymbol{R}_i^k\}$	虚拟机需要的资源向量集合
$\{\boldsymbol{T}^k\}$	第 k 个用户/任务的通信矩阵
$\{X_i^k(m)\}$	再调度前的虚拟机部署方式
$\{Y_i^k(m)\}$	再调度后的虚拟机部署方式
α	虚拟机数据通信量的预测系数
β	启用新物理机的门限系数

6.1 基于贪心策略的再调度算法

贪心再调度算法首先计算每个用户所需要的任务 *Original* 值、*Best* 值以及 *Priority* 值,它们的定义如下:

$$Original(G_k)=\sum_{i=1}^{w_k}\sum_{j=1}^{w_k}\frac{t_{ij}^k}{B(J_{ij}^k(X))} \tag{13}$$

即任务在当前部署方式下的 *Benefit* 值;

$$Best(G_k)=\sum_{i=1}^{w_k}\sum_{j=1}^{w_k}t_{ij}^k \tag{14}$$

即该任务能达到的最大 *Benefit* 值;

$$Priority(G_k)=Best(G_k)/Original(G_k) \tag{15}$$

Priority 的值反映了任务当前的部署状态与最优状态之间的距离,*Priority* 的值越高说明了任务越应该优先得到再调度.

在计算出 *Priority* 值后,将所有用户或任务按 *Priority* 值降序排序,然后依次取出队首元素,尝试对该组虚拟机进行再调度.算法 1 首先调用过程 1,尝试通过尽可能少的移动将该组虚拟机移动到同一机架的物理机上.如果过程 1 尝试失败,则调用过程 2,过程 2 优先移动通信量大的且不在一个机架上的虚拟机.过程 2 结束后,检查该任务的虚拟机是否都处于同一群组中,如果不是的话,尝试移动虚拟机至同一群组.

算法 1. 基于贪心策略的再调度算法.

输入:$M,K,\{\boldsymbol{H}_i\},\boldsymbol{L},\boldsymbol{Q},\{G_k\},\{\boldsymbol{R}_i^k\},\{\boldsymbol{T}^k\},\{X_i^k(m)\}$

输出:新的虚拟机部署方式$\{Y_i^k(m)\}$

1. 计算每个用户/任务的优先值(*Priority*),并按降序排序,存入队列 *P_Queue*
2. WHILE(*P_Queue* 非空)
3. 取出 *P_Queue* 的队首,存入 G_k
4. 尝试,调用过程 1,将 G_k 移动到同一机架,若成功
5. 则进入下一次循环
6. 否则,调用过程 2,尝试贪心移动 G_k
7. 检查是否有跨群组的虚拟机,如果有
8. 则调用过程 3,将 G_k 移动到同一群组下
9. END WHILE
10. 输出新的虚拟机部署方式$\{Y_i^k(m)\}$

过程 1 是面向机架的虚拟机整体聚集,该过程尝试将 G_k 移动到同一机架.首先对该组虚拟机按其所在机架进行分组,统计每个机架上该组虚拟机的个数,按机架上该组虚拟机的个数进行升序排序.然后尝试在不违反限制条件的前提下,将除最后一个机架外的其他机架的虚拟机都移动到最后一个机架上,若尝试成功,则返回成功标志,如果尝试失败,则撤销此前操作,返回失败标志.过程 1 的描述如下.

过程 1. 面向机架的虚拟机整体聚集.

1. 对该组虚拟机按其所在机架进行分类,统计每个机架上该组虚拟机的个数,虚拟机个数最多的机架记为 R.
2. 将除 R 以外的其他机架上的虚拟机存入队列 Q
3. 初始化操作栈 *OP_Stack*
4. BOOL *Success*=TRUE
5. WHILE(Q 非空)
6. 取出 Q 的队首元素,存入 v
7. 如果R有空间容纳 v 且迁移 v 到 R 不违反限制条件
8. 则将 v 迁移至 R
9. *Success*=*Success*&&TRUE
10. 将进行的迁移操作压入栈 *OP_Stack*
11. 否则 *Success*=*Success*&&FALSE
12. END WHILE
13. 如果 *Success* 不等于 TRUE
14. 则遍历栈 *OP_Stack*,将进行的操作撤销
15. 返回尝试失败标志
16. 否则返回尝试成功标志

过程 2 为面向机架的虚拟机贪心聚集,尝试贪心移动 G_k.首先对虚拟机按所在的机架分组,然后计算每个虚拟机分别与其他机架上虚拟机通信的总和与本机架上其他虚拟机通信总和的差 *delta*,对 *delta* 按降序排序;取出队首元素,如果值小于 0,则结束循环,否则尝试迁移队首的虚拟机,如果失败,则尝试迁移下一个,若全部尝试失败,或出现值 0 的情况则停止循环;若尝试成功,则重复过程 2,直至

出现遍历完整个队列而没有可迁移的虚拟机. 过程 2 的具体描述如下.

过程 2.　面向机架的虚拟机贪心聚集.

1. 结束标记 *flag* 初始化为 FALSE
2. WHILE(*flag* 等于 FALSE)
3. 对该组虚拟机按其所在机架进行分组至 G_k^0, G_k^1,…,G_k^n
4. 计算 G_k 中所有虚拟机的 *delta* 值,并按降序排列,将 *delta* 值大于 0 的存入队列 Q
5. WHILE(Q 非空)
6. 取出 Q 的队首元素存入 *vm*
7. 求出和 *vm* 通信总量最大的机架 R
8. 如果移动 *vm* 至 R 不违反限制条件
9. 则移动 *vm* 至 R
10. 提前结束循环
11. END WHILE
12. 如果 Q 为空,则置 *flag* 等于 TRUE
13. END WHILE

过程 3 是将 G_k 移动到同一群组下,该过程最后再对该用户/任务的虚拟机进行一次检查,如果出现虚拟机位于不同群组的情况,则尝试在不违反限制条件的情况下进行移动,使其尽可能的被部署到同一群组或尽可能少的群组中. 该过程与过程 2 类似,只是将过程 2 中的同一机架的条件扩大到同一群组,故在此不再累述.

算法 1 只对任务遍历两遍,第 1 遍计算 *priority* 值,第 2 遍进行调度. 而计算 *priority* 值和 *benefit* 值的时候,需要遍历每个任务的通信量矩阵;在计算 *cost* 值的时候,需要遍历数据中心的所有交换机. 设任务总数为 k,每个任务所需的虚拟机数量上限为 t,数据中心的交换机总数为 l,则遍历任务的时间复杂度为 $O(k)$,计算 *priority*、*benefit* 值的复杂度为 $O(t^2)$,计算 *cost* 的时间复杂度为 $O(l)$,因此贪心再调度算法总的时间复杂度为 $O(k \cdot t^2 \cdot l)$.

贪心再调度算法中,首先通过 *Priority* 的值确定了任务调度的顺序,*Priority* 的值能较好地反映任务需要被再调度迫切程度,但并不能保证当前调度是最优的. 因此我们设计了一个基于多次迭代的随机再调度算法,将贪心再调度算法与其比较,对比其性能.

6.2　基于多次迭代的随机再调度算法

基于多次迭代的随机再调度算法首先随机产生一个调度序列,然后使用算法 1 中的几个过程进行再调度. 在每次执行再调度的过程中,都将操作过程记录在栈中,调度完后,计算 *Benefit* 值,然后从栈中依次取出操作进行恢复. 重复上述过程 n 次,选出 *Benefit* 值最大的一次调度序列作为实际调度. 当迭代的次数足够多时,算法取得的解覆盖最优解的几率就越大,离最优解的距离也就越近. 算法 2 的描述如下.

算法 2.　基于多次迭代的随机再调度算法.

输入:M,K,$\{\boldsymbol{H}_i\}$,$\boldsymbol{L}$,$\boldsymbol{Q}$,$\{G_k\}$,$\{\boldsymbol{R}_i^k\}$,$\{\boldsymbol{T}^k\}$,$\{X_i^k(m)\}$,n

输出:新的虚拟机部署方式$\{Y_i^k(m)\}$

1. 初始化操作栈 *OP_Stack*, 初始化 *Best_Benefit* 值为 0,初始化 *Best_Queue* 为空
2. WHILE(i++<n)
3. 清空 *OP_Stack*
4. 对所有任务进行随机排序,存入 *P_Queue*
5. WHILE(*P_Queue* 非空)
6. 取出 *P_Queue* 的队首,存入 G_k
7. 尝试将 G_k 移动到同一机架,若成功
8. 则进入下一次循环
9. 否则,尝试贪心移动 G_k
10. 检查是否有跨群组的虚拟机,如果有
11. 则尝试将 G_k 移动到同一群组下
12. END WHILE
13. 计算这种策略下的收益值(*Benefit*)
14. 如果 *Benefit* 大于 *Best_Benefit*
15. 则值 *Best_Benefit* 为当前的 *Benefit* 值
16. 将当前 *P_Queue* 赋值给 *Best_Queue*
17. 依次取出 *OP_Stack* 中的操作进行复原
18. END WHILE
19. 对 *Best_Queue* 重复一遍过程 1、过程 2 的调度算法
20. 输出新的虚拟机部署方式$\{Y_i^k(m)\}$

算法 2 中的面向机架的虚拟机整体聚集、面向机架的虚拟机贪心聚集和算法 1 中对应的过程类似,只是需要将进行的迁移操作全部压入栈 *OP_Stack* 中,以便在完成一次随机调度后,恢复到原始状态.

算法 2 中每次迭代只对任务遍历一遍,而计算 *priority* 值和 *benefit* 值的时候,需要遍历每个任务的通信量矩阵;在计算 *cost* 值的时候,需要遍历数据中心中的所有交换机;在迭代一次结束后,需要恢复到原始状态. 设迭代次数为 n,任务总数为 k,每个任务所需的虚拟机上限为 t,数据中心的交换机总数为 l,则遍历任务的时间复杂度为 $O(k)$,计算 *priority*、*benefit* 值的复杂度为 $O(t^2)$,计算 *cost* 的时间复杂度为 $O(l)$,每次迭代结束恢复状态的时间复杂度为 $O(t)$,因此随机再调度算法总的时间复杂度为 $O(n \cdot k \cdot t^3 \cdot l)$.

7 性能测试

7.1 实验准备

参照网址①中的 Thunder Linux Cluster，我们构建了一个由 21 个群组，269 个机架，3224 台物理机构成的虚拟数据中心环境. 数据集选择②中提供了真实的分布式任务日志，日志中包含了一个月内的 12 万个任务的起始时间、完成时间以及请求的资源总数. 由于日志中并没有提供节点之间的通信矩阵，为了不失一般性，我们随机为每个任务产生了一组通信矩阵(在实际应用中，该通信矩阵可由用户近似给出，或者由数据中心的管理者监控网络流量得到).

为了衡量算法的效果，我们首先定义“分散的任务”，即若某个任务有两台虚拟机被分配到不同机架的物理机上，则把该任务称为分散的. 第 3 节中实验结果表明了一个任务的完成时间或者服务性能会受任务分散程度的影响. 同一个任务，如果在被部署时不是分散的，则能够完成的更快或者拥有更高的性能. 因此，我们在模拟试验中，首先对比执行再调度算法前后分散任务的数量.

此外，一个任务的 *benefit* 值是关于其虚拟机之间的通信量与网络通信能力的函数，*benefit* 值越大，表明该任务拥有越好的网络通信能力. 因此，*benefit* 值是衡量算法性能的一个直接指标.

7.2 基于贪心策略的再调度算法性能

7.2.1 配合 First-Fit 算法

我们首先将贪心再调度算法配合 First-Fit 算法使用，比较了使用再调度算法前后以及以不同频率执行再调度算法时，分散的任务的数量. 实验结果如图 3 所示.

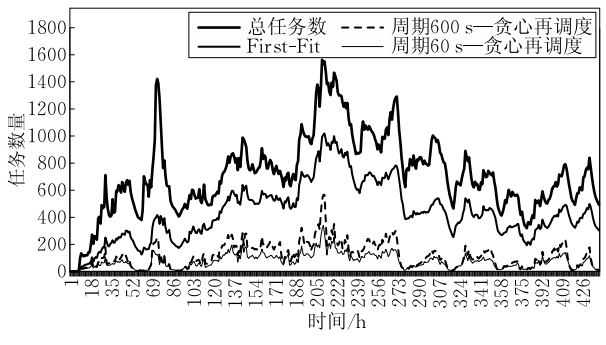

图 3 贪心再调度算法配合 First-Fit 性能

如图 3 所示，最上面的曲线是正在运行中的任务总数；第 2 条曲线是用 First-Fit 算法进行调度时，分散的任务的总数；第 3 条曲线是 First-Fit 算法配合 600 s 一次的贪心再调度算法时，分散的任务的总数；最下方的曲线是 First-Fit 算法配合 60 s 一次的贪心再调度算法时，分散的任务的总数. 从实验结果中可以明显看出，在没有执行再调度算法之前，分散的任务数随着时间的推移，逐渐增高；而执行再调度算法后，分散的任务数明显降低. 并且，执行再调度算法的周期越短，效果就越好，但是由于虚拟机的迁移平均耗时在 60 s，再调度周期过短会造成虚拟机状态的不稳定，并且当迁移集中在某一时间段时，会造成数据通信量激增，因此，在实际中，再调度的周期至少选择在 60 s 以上.

如图 4 所示，最上面的曲线是所有正在执行的任务的理论最大 *benefit* 值的和；最下面的曲线是只用 First-Fit 算法进行调度时候，所有任务的 *benefit* 值总和；中间的曲线是 First-Fit 算法配合 60 s 一次的贪心再调度算法时，任务的 *benefit* 值总和. 从实验结果中可以明显看出，在没有执行再调度算法之前，任务的网络通信能力离理论最大值有较大差距，而执行再调度算法后，任务的网络通信能力几乎接近理论最大值. 因此，从实验结果可以看出，再调度算法能够大大提高任务的网络通信能力.

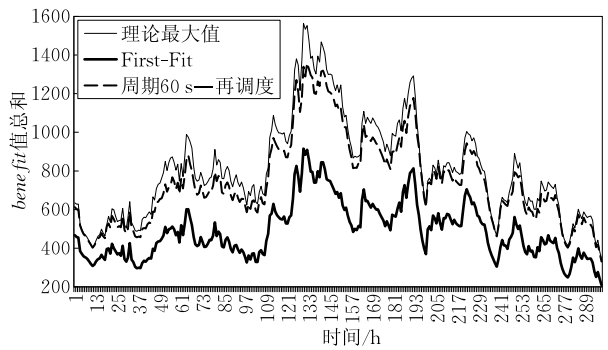

图 4 贪心再调度算法配合 First-Fit 的通信收益比较

7.2.2 配合网络感知的虚拟机部署算法

Approx 是文献[7]中提出的基于网络感知的虚拟机调度算法，该算法在初次处理用户请求时，就考虑到了网络因素，因此算法在初期表现的很好. 但是随着不断有任务完成后退出，物理机上的碎片逐渐增多，原算法的性能就出现了下降. 在实验中，我们对比了 Approx 算法在配合使用再调度算法前后，出现的分散的任务的数量的变化以及归一化后的 *benefit* 值的对比，实验结果如图 5、图 6 所示.

① https://computing.llnl.gov/?set=resources&page=index
② http://www.cs.huji.ac.il/labs/parallel/workload/

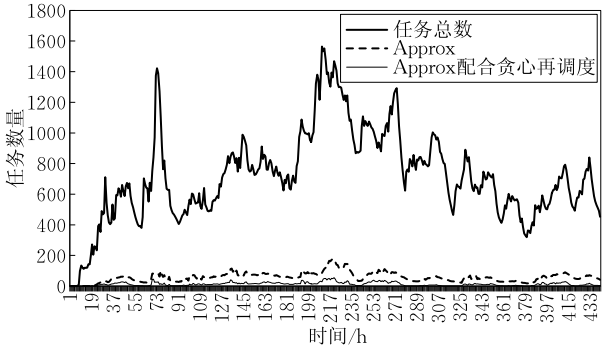

图 5　Approx 配合贪心再调度算法性能

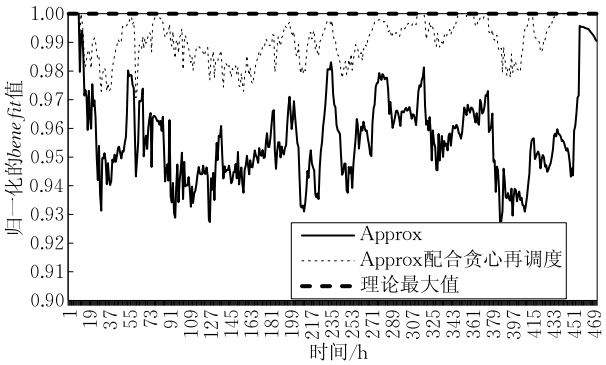

图 6　Approx 配合贪心再调度算法 *benefit* 值比较

从图 5 的实验结果中以及对比图 2 可以看出，尽管 Approx 算法相比 First-fit 算法在虚拟机的网络通信能力方面已经能取得较好的效果，但算法的性能依然会随着时间推移而下降. 而将 Approx 算法配合上我们的贪心再调度算法后，原算法的效果有进一步的提升. 从图 6 可以看出，Approx 算法配合上我们的贪心再调度算法后 *benefit* 值接近理论最大值，且相对于单独使用的 Approx 算法网络通信能力有进一步提升.

7.2.3　对比基于多次迭代的随机再算法

在这组实验中，我们在 First-Fit 算法的基础上，分别配合基于贪心策略的再调度算法以及 100-随机再调度（即迭代 100 次）和 400-随机再调度（即迭代 400 次）两个迭代次数不同的随机再调度算法对比其性能. 选取分散的任务数量作为评价指标，实验结果如图 7 所示.

从图 7 中可以看出，随机迭代的次数越多，算法的效果就越好，且基于贪心策略的再调度算法在性能上接近多次迭代的随机再调度算法. 但是基于多次迭代的随机再调度算法将消耗大量的时间用于运算，虽然对比贪心再调度算法的性能有部分提升，但提升效果并不显著. 因此我们在实际应用中，使用基于贪心策略的再调度算法更为合理、有效.

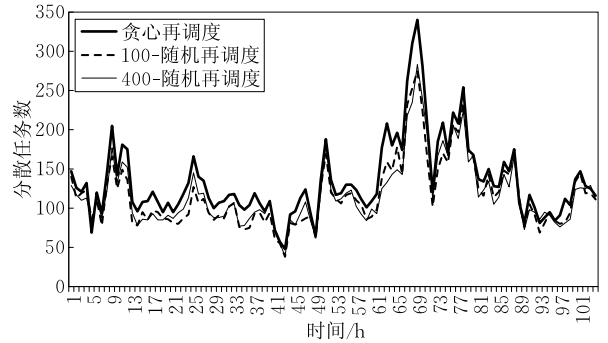

图 7　多次迭代的随机再调度算法与贪心再调度算法比较

7.3　再调度算法对数据中心利用率的影响

再调度的过程有可能导致新的物理机的启用，β 参数控制了启用新物理机的条件，当前我们设定 β 值为 0.95，即当数据中心的整理利用率达到 95% 以上时，才可以启用新的物理机. 由于我们的算法旨在将位于不同机架上且通信量较大的虚拟机迁移至同一机架，因此迁移过程不会造成机架的开关. 表 5 显示了执行贪心再调度算法前后额外使用物理机的情况.

表 5　再调度对数据中心利用率的影响

物理机额外使用比例/%	时间段/%
(0,0.5]	10
(0.5,1]	1
(1,5)	1

从表 5 中可以看出，总共只有 12% 的时间中，执行再调度算法后会比之前使用更多的物理机. 而这 12% 的时间中的大部分时间，额外使用的物理机数量都控制在 0.5% 之内. 因此可以认为再调度算法不会对数据中心整体的利用率产生较大影响.

8　总　结

虚拟机的部署与调度是数据中心管理中的核心问题. 本文深入研究了面向网络感知的周期性资源重配置问题，仔细分析并测试了网络通信能力以及各种因素对动态迁移过程中部署在虚拟机上的服务性能的影响. 在这基础上，提出了面向网络感知的虚拟机再调度算法，通过适当的虚拟机迁移，提高部署在虚拟机上任务的性能以及数据中心整体的运行效率. 我们通过 RUBiS 和 PUMA 两个测试平台验证了网络通信对部署在虚拟机上服务性能的影响；通过真实的数据集和模拟实验，分别在 First-Fit 以及 Approx 这两种虚拟机部署算法下，对比了应用虚拟机再调度算法前后虚拟机的部署效果，验证了我们

算法能够以较小的代价使得高网络通信代价的任务数明显减少,虚拟机组的网络通信能力显著提高.

参 考 文 献

[1] Nguyen Van H, Tran F D, Menaud J-M. Autonomic virtual resource management for service hosting platforms//Proceedings of the ICSE Workshop on Software Engineering Challenges of Cloud Computing. Vancouver, Canada, 2009: 1-8

[2] Dutta S, Verma A. Service deactivation aware placement and defragmentation in enterprise clouds//Proceedings of the 7th International Conference on Network and Services Management. International Federation for Information Processing. Laxenburg, Austria, 2011: 180-188

[3] Padala P, Shin K G, Zhu X, et al. Adaptive control of virtualized resources in utility computing environments. ACM SIGOPS Operating Systems Review, 2007, 41(3): 289-302

[4] Jung Gueyoung, et al. A cost-sensitive adaptation engine for server consolidation of multitier applications//Proceedings of the Middleware 2009. Urbana Champaign, USA, 2009: 163-183

[5] Caprara A, Toth P. Lower bounds and algorithms for the 2-dimensional vector packing problem. Discrete Applied Mathematics, 2001, 111(3): 231-262

[6] Xu Jing, Fortes J A B. Multi-objective virtual machine placement in virtualized data center environments//Proceedings of the 2010 IEEE/ACM International Conference on Green Computing and Communications(GreenCom) & International Conference on Cyber, Physical and Social Computing (CPSCom). Hangzhou, China, 2010: 179-188

[7] Jayasinghe D, et al. Improving performance and availability of services hosted on iaas clouds with structural constraint-aware virtual machine placement//Proceedings of the 2011 IEEE International Conference on Services Computing (SCC). Washington, USA, 2011: 72-79

[8] Chen Wei, et al. A profit-aware virtual machine deployment optimization framework for cloud platform providers//Proceedings of the 2012 IEEE 5th International Conference on Cloud Computing (CLOUD). Hawaii, USA, 2012: 17-24

[9] Marzolla M, Babaoglu O, Panzieri F. Server consolidation in clouds through gossiping//Proceedings of the 2011 IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM). Lucca, Italy, 2011: 1-6

[10] Bobroff N, Kochut A, Beaty K. Dynamic placement of virtual machines for managing sla violations//Proceedings of the 10th IFIP/IEEE International Symposium on Integrated Network Management. Munich, Germany, 2007: 119-128

[11] Breitgand D, Kutiel G, Raz D. Cost-aware live migration of services in the cloud//Proceedings of the 3rd Annual Haifa Experimental Systems Conference. Haifa, Israel, 2010: 1-6

[12] Alicherry M, Lakshman T V. Network aware resource allocation in distributed clouds//Proceedings of the 31st Annual IEEE International Conference on Computer Communications. Orlando, USA, 2012: 963-971

[13] Meng Xiaoqiao, Pappas V, Zhang Li. Improving the scalability of data center networks with traffic-aware virtual machine placement//Proceedings of the 29th Annual IEEE International Conference on Computer Communications. San Diego, USA, 2010: 1-9

[14] Jiang J W, et al. Joint VM placement and routing for data center traffic engineering//Proceedings of the 31st Annual IEEE International Conference on Computer Communications. Orlando, USA, 2012: 2876-2880

[15] Guo Chuanxiong, et al. BCube: A high performance, server-centric network architecture for modular data centers. ACM SIGCOMM Computer Communication Review, 2009, 39(4): 63-74

[16] Biran O, et al. A stable network-aware vm placement for cloud systems//Proceedings of the 2012 12th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing. Ottawa, Canada, 2012: 498-506

[17] Steiner M, et al. Network-aware service placement in a distributed cloud environment. ACM SIGCOMM Computer Communication Review, 2012, 42(4): 73-74

[18] Al-Fares M, Loukissas A, Vahdat A. A scalable, commodity data center network architecture. ACM SIGCOMM Computer Communication Review, 2008, 38(4): 63-74

[19] Verma A, Kumar G, Koller R. The cost of reconfiguration in a cloud//Proceedings of the 11th International Middleware Conference Industrial Track. Bangalore, India, 2010: 11-16

[20] Voorsluys W, et al. Cost of virtual machine live migration in clouds: A performance evaluation//Jaatun M G, Zhao Gansen, Rong Chunming eds. Cloud Computing. Berlin Heidelberg: Springer, 2009: 254-265

[21] Murty K G, Kabadi S N. Some NP-complete problems in quadratic and nonlinear programming. Mathematical Programming, 1987, 39(2): 117-129

LUO Gang-Yi, born in 1989, M. S. candidate. His current research interests include cloud computing and virtualization.

QIAN Zhu-Zhong, born in 1980, Ph. D., associate professor. His current research interests include distributed system and virtualization.

LU Sang-Lu, born in 1970, Ph. D., professor, Ph. D. supervisor. Her current research interests include distributed system and computer network.

Background

VM Consolidation is a hot topic in Cloud Computing. Since the concept of lease computing and storage resource from datacenters instead of purchasing physical equipment is widely accepted by public, how to effectively and efficiently manage datacenter's physical resource has become a major problem and attracted significant attention in recent years. An effective and efficient VM consolidation algorithm can help improve utilization rate of physical servers, lower energy cost and improve the overall performance.

Current VM consolidation researches focus on the requirement of CPU, Memory and network bandwidth which trying to allocate VM into physical servers in a low cost way. Sourav Dutta et al. proposed a service deactivation aware placement methodology which places virtual machines in a way that minimizes the cost of defragmentation in the cloud. Jiang Xu et al. proposed a two-level control system to manage the mappings of workloads to VMs and VMs to physical resources which target is simultaneously minimizing total resource wastage and power consumption.

Communication ability is an important factor that affects the performance of VM group which draw significant attention in current research. Mansoor Alicherlry et al considered resource allocation for distributed cloud systems and proposed and efficient algorithms for resource allocation which aims to minimize communication costs and latency. Meng Xiaoqiao et al focused on using traffic-aware VM placement to improve the network scalability and design a two-tier approximate algorithm to solve the problem. Jiang Wenjie et al. researched on joint VM placement and routing problem which aims to minimize traffic costs and proposed an online algorithm in a dynamic environment under changing traffic loads. Wang Meng et al. focused on modeling network bandwidth demands of VMs and formulate the VM consolidation into a Stochastic Bin Packing problem and propose an online packing algorithm.

However, these online consolidation algorithms rarely consider the problem of resource utilization decline and network delay increasing which is caused by VM quit when job is finished. Therefore, we thorough studied the network-aware resource reconfiguration problem and proposed a network-aware VM re-scheduling algorithm which aims to improve the VM performance and datacenter's efficiency through appropriate VM live migration. The algorithm focuses on improving communication ability among VMs through periodical VM re-scheduling to promote the overall performance.

This work is partially supported by the National Natural Science Foundation of China under Grant Nos. 61073028, 61021062, 61202113, Jiangsu Science and Technology Supporting Project under grant No. BE2013116, Jiangsu Natural Science Foundation under grant No. BK2011510.