

本科生毕业设计

|  |
| --- |
| 应用可感知的边缘分布式调度系统设计与实现 |

|  |  |
| --- | --- |
| 院 系 | 计算机科学与技术 |
| 专业班级 | CS1807 |
| 姓 名 | 李世铭 |
| 学 号 | U201814682 |
| 指导教师 | 刘方明 |

2022年06月10日

目 录

[1 绪 论 1](#_Toc101806840)

[1.1 课题背景 1](#_Toc101806841)

[1.2 国内外研究现状 2](#_Toc101806842)

[1.3 研究目的和主要内容 3](#_Toc101806843)

[1.4 论文结构 3](#_Toc101806844)

[1.5 课题来源 4](#_Toc101806845)

[参考文献 5](#_Toc101806846)

# 

# 绪 论

本章首先介绍了边缘计算推出背景以及当前边缘计算平台亟待解决的可以实现资源均衡的负载调度问题，然后分析了当前边缘计算平台发展现状以及在负载调度方面存在的问题，介绍了国内外在边缘计算平台负载调度领域的相关研究工作，并对本文的主要研究内容及工作意义作了具体说明。

## 课题背景

### 研究背景和趋势

边缘计算[1]是在云计算的基础上推出一种新型计算范式。边缘计算是一种分布式运算架构，将应用程序、数据资料与运算服务，由网络中心节点移往离终端用户更近的边缘节点来处理。通过将计算负载放在离终端用户更近的边缘计算节点，边缘计算可以减少网络往返时延，满足延迟敏感型应用的要求。并且由于用户数据在边缘节点完成计算，不会上传到中心云，边缘计算还具有安全隐私保护的功能。

边缘计算可以服务于多种应用程序，比如云游戏，云AR和直播等。对应用进行分析后发现，不同的应用在响应时间要求和资源占用方面（CPU，GPU，内存和带宽等）存在差异。如云游戏[2]和高清直播[3]相比，前者对计算资源的要求更高，而后者则对带宽的要求比较高。

边缘计算节点在地理位置上表现出分布广的特点，应用的响应时间与计算节点到终端用户的物理距离高度相关，因此需要对边缘计算节点进行合理建模，以便于寻找满足应用响应时间要求的计算节点。

边缘计算节点的各种资源较为有限，不合理的负载调度可能会导致资源浪费。比如，在同一个计算节点部署大量带宽主导型应用，会造成严重的带宽资源竞争，影响应用的服务效果。同时，节点大量的计算资源（如CPU和GPU）处于闲置状态，造成资源浪费。因此，负载调度器需要能够感知应用的资源占用特点，在负载调度时实现资源互补，尽可能提高各计算节点的资源利用率。

### 面临的问题和挑战

**指数级的搜索空间。**将n个应用部署到m个边缘节点，一共有种分配方式，采取枚举法求解全局最优解的效率可能会非常低下，无法满足用户的需求。因此需要采用启发式算法的思想设计负载调度算法，以在多项式时间求解全局最优解或近似最优解。

**应用的多资源需求。**现有的调度策略研究中，大多关注应用的总体资源占用情况，而忽略了应用会占用多种不同资源，而且不同类型的应用在资源占用方面也有不同的侧重。因此本研究会首先测试不同应用在不同资源类型的占用方面是否确实存在差异，然后会设计一种负载调度算法，使各节点的资源利用率尽可能达到均衡，避免某些节点的资源利用率过高导致应用执行出现问题，即“热点”现象。均衡各节点的资源利用率有两层含义，一是使节点内各资源的利用率尽可能接近一致，二是使不同节点之间的资源利用率尽可能一致。

## 国内外研究现状

### NEP客户预留策略

Xu等[1]提到了一个目前在中国大规模使用的边缘计算平台NEP。NEP采用的是一种边缘计算平台提供商（NEP）-云应用提供商（客户）-终端用户的商业模式。客户根据预期的用户需求，从NEP租用位于不同地理位置的虚拟机资源，客户拥有虚拟机的全部控制权，包括编排用户请求的工作负载。虚拟机分配策略和终端用户请求调度策略分别由边缘提供者和客户独立制定。这种分离阻碍了负载平衡。

这种以用户为中心，缺乏全局视角的资源调度，不利于节点间的负载均衡和节点内资源利用率的提升。文中给出的数据显示出NEP平台确实存在着严重的负载不均衡现象，例如，CPU使用量的第95%分位是第5%分位的8.7倍，带宽使用量的第95%分位是第5%分位的731倍。边缘计算平台往往部署着延迟敏感型的应用，过高的 CPU 使用率会导致计算任务延迟，而过高的带宽使用率可能会导致流量堵塞和长时间的网络延迟，这些因素会使延迟敏感型的应用不能满足用户的需求。

### 贪心启发策略

VideoEdge[4]是一个视频流分析规划系统。论文中提出了视频流层次结构：摄像机-边缘集群-公共云。VideoEdge中的工作负载是一系列视频查询，VideoEdge的目的是联合优化所有查询，以在可用资源范围内最大化平均查询精度。VideoEdge可以调度工作的负载位置和工作的任务量，对于同一项查询VideoEdge可以降低查询精度来减少它对资源的需求。一项查询由多种组件构成，例如对象跟踪查询由一个“解码器”组件、一个对象“检测器”组件和一个“关联器”组件组成。每个组件都有多种资源需求，相对的，每个节点也会提供多种资源，论文提出了主资源需求的概念，来反映特定资源的稀缺程度，通过将工作负载对多种资源的需求转化为对主资源的需求，可以将多维对比转化为单维对比，降低了算法的复杂度。文中提出了一种贪心启发算法，在部署新的查询工作负载时，减少集群内主资源的消耗，并尽可能提升查询的准确率。文中的调度算法将梯度定义为不同的分配策略下，查询的准确率之差与查询的主资源需求之差的比值，每次迭代每个查询都选取使梯度最大的分配作为自己新的分配方式，在剩余资源不足以支持任何一个查询采用更高的精度的设置或所有查询都已经采用了最高的查询精度的设置时，完成迭代并输出VideoEdge找到的最优分配方案。

### 启发式算法

启发式算法[8]是指在可接受的计算成本内去搜寻最好的解，但不一定能保证所得的是最优解。启发式算法可以将全局的搜索空间逐渐划分为有限的搜索空间，降低算法的复杂度。

## 研究目的和主要内容

本次毕业设计的主要研究内容是设计一个应用于边缘计算平台的负载调度器，其可以根据应用的响应时间要求和资源需求，动态调整不同节点上的工作负载，均衡各计算节点的资源利用率。

## 论文结构

本文的主要内容如下：

第一章我们首先介绍了边缘计算平台提出的背景，然后分析了边缘计算平台负载调度技术的产生及发展现状，以及当前边缘计算平台在负载调度方面面临的挑战，介绍了国内外在边缘计算平台负载调度策略和启发式算法领域的相关研究工作，并对本文的主要研究内容及工作意义作了具体说明。

## 课题来源

课题来源于国家自然科学基金中德国际合作项目，边缘计算基础理论与关键技术，No.61761136014。

# 方案论证

本章主要提出本次设计的方案以及对方案进行论证，首先简要阐述问题，然后对问题进行建模，提出系统的需求，引出本次设计的基本方案，并对方案进行可行性分析。

## 问题阐述

边缘计算节点的各种资源较为有限，不合理的负载调度可能会导致资源浪费。如果某个节点有不合理的资源占用情况，如图2‑1所示，此节点CPU资源占用率较高而内存和带宽资源仍有较多余量，但是由于CPU资源限制，此节点不能再部署更多的应用，那么这个节点就会有一部分内存和带宽资源始终处于闲置状态，在这个节点的部分应用释放CPU资源之前，处于闲置状态的资源无法得到其他应用的利用，在某种程度上降低了节点的资源利用率。而且某种资源过高的占用率可能也会造成一些不良影响，比如应用在运行过程中，资源实际的占用情况可能会发生变化，因此节点中占用率过高的资源可能会出现暂时的超载现象，可能会造成节点上的应用执行卡顿，甚至节点服务器出现宕机现象。而如果能将将资源占用率过高的节点中的某些应用迁移到其他资源占用率较低的节点，可以避免上述问题的出现。

因此，本次设计的调度器将根据应用的响应时间要求和资源需求，在满足各个应用的延迟要求的条件下，尽可能均衡各节点的资源占用率，一方面使各节点的总体资源占用率尽可能达到均衡，另一方面使节点内的各资源占用率也尽可能达到均衡。



图2‑1 不合理的资源占用情况

## 问题建模

对问题进行数学建模，首先阐述各符号定义。

应用总数用N表示，节点总数用M表示，资源类型有三种（CPU、内存和带宽）。

表示应用*i*需要资源*k*的量。

表示节点*j*的资源*k*总量。

是一个N行M列的01矩阵，表示一种分配方式，如果矩阵的第*i*行第*j*列的元素为1表示在这种分配方式下，将应用*i*部署在节点*j*上。

例如，如上面的矩阵所示，集群中有2个节点并且部署着3个应用，在矩阵表示的分配方式下，应用1被部署在节点1，而应用2、3被部署在节点2。

表示应用*i*的处理时延和传播时延之和。

表示应用*i*部署在节点*j*的传播时延。

表示应用*i*的延迟SLA（Service Level Agreement，服务级别协议，指服务提供商承诺可以提供给用户的服务质量）。

为了使目标函数能够体现各节点资源利用率的离散程度，借鉴方差公式构造目标函数：

(2‑1)

其中：

(2‑2)

因为不同的分配方式，应用数、节点数和资源类型数不会发生变化，所以去掉方差公式的分母，只保留各节点资源利用率与集群整体资源利用率均值的差值平方和。

式(2-1)所示的目标函数与集群的资源利用不均衡程度呈正相关，使目标函数取最小值的分配方式，也就是使集群的资源利用率达到最均衡状态的分配方式，即我们求解的目标。

式(2-2)给出了函数的定义，函数是一个关于、的二元函数，表示在分配方式下，集群中资源的利用率均值。

接下来给出约束函数：

(2‑3)

(2‑4)

(2‑5)

式(2-3)表示矩阵的每一行有且只有一个元素为1，即每个应用都在集群中部署，且每个应用只能部署在一个节点上。

式(2-4)表示每个节点部署的应用对资源的需求量之和不能超过该节点可以提供的资源总量。

式(2-5)表示每个部署的应用的处理时延、传输时延和传播时延之和不能超过该应用的时延SLA。

## 系统需求分析

本研究将设计一个应用可感知的边缘负载调度系统，系统需要模拟一个边缘节点集群，集群中每个节点都有自己的坐标与可以提供的资源信息，使用者可以使用模板创建边缘节点集群，也可以手动增删节点，并对各节点进行配置。系统需要模拟一系列在集群中部署的应用，每个应用都有自己的资源占用、延迟要求和用户位置信息。系统模拟集群持续运行，随时会产生若干应用的申请，同时也有若干已经部署在集群中的应用停止运行，释放占用的资源。系统中的核心组件，负载调度器会随着应用的产生与销毁，重新规划集群中的工作负载，使集群中各节点的资源占用率达到最均衡的状态。

## 基本方案制定

本次设计的实验工作将分为两个阶段。

第一个阶段是定量测量典型的边缘应用的资源需求。首先基于StarlingX平台在实验室服务器中搭建一个模拟边缘集群，然后调研开源的典型边缘应用（云游戏、直播），在集群中部署运行，并编写测试程序，测量应用对不同类型资源的需求。

第二个阶段是设计边缘调度系统。系统分为两个模块，一个是集群运行模块，另一个是负载调度器模块。集群运行模块会维护一个模拟边缘集群，集群以周期为单位动态变化，每个周期都会有新的应用部署到集群和旧应用停止运行释放资源，并且运行中的应用占用的资源也会动态变化，负载调度器模块会实时获取集群的节点信息和应用信息，并不断进行负载调度。本次设计的负载调度器将以启发式算法为基础，以在有限的时间内获得一个近似最优解，并且设置两个基准算法（枚举法求最优解策略和NEP使用的最近分配策略），多次进行模拟实验，得到设计的负载调度算法与基准算法之间求得的解的差距和求解时间的差距，论证算法的有效性。

## 关键技术分析

### StarlingX

StarlingX是一个完全集成的边缘云软件堆栈，提供快速部署、可伸缩、高可靠的边缘软件平台基础设施。StarlingX可以适配云端到边缘端的计算场景，实现全系统范围内的自动编排，集中部署和管理边缘云，简化分布式边缘系统的部署。StarlingX集成了Kubenertes服务，Kubenertes是一个用于管理容器化的工作负载和服务的开源平台。

本课题计划在服务上部署StarlingX集群，使用Kubenertes对应用进行管理，以docker容器的方式启动开源边缘应用，并使用测试程序对应用的资源占用情况进行测量。

### 人工鱼群算法

从启发式算法中选择了一个比较适合问题模型的人工鱼群算法作为算法设计的基础。

在一片水域中，鱼往往能自行或尾随其他鱼找到营养物质多的地方，因而鱼生存数目最多的地方一般就是本水域中营养物质最多的地方，人工鱼群算法就是根据这一特点，通过构造人工鱼来模仿鱼群的觅食、聚群及追尾行为，从而实现寻优。

假定求解的问题的状态函数是由两个变量决定，即状态函数为，那么可以给池塘建立一个平面坐标系，每一个点的坐标可以对应状态函数的两个变量，每个点都可以根据状态函数计算出一个状态，并且状态有好坏之分。

向池塘中加入含有若干条人工鱼的鱼群，鱼群具有以下几个参数：鱼的数量m；鱼的步长Step，指鱼一次移动的距离；鱼的视野范围Visual，指鱼可以感知的点和同伴的范围；拥挤度因子δ，指允许邻域内鱼的最大拥挤程度，引入拥挤度因子可以避免鱼群陷入局部最优解；觅食尝试次数Trynumber，指鱼在执行觅食行为时的最大尝试次数；迭代次数Genmax。

算法执行时，人工鱼群进行若干次迭代，每次迭代，对鱼群中的鱼进行一次遍历，每条鱼会试探执行以下三种行为：觅食行为，选择视野范围内一点，如果该点的状态优于当前位置，就向该点移动一步，否则重复选择至多Trynumber次，如果都不优于当前位置，随机选择视野范围内一点并移动一步；群聚行为，探索视野范围内的伙伴位置，计算伙伴的中心位置并向该位置移动一步；追尾行为，探索视野范围内的伙伴，找到处于最优位置的伙伴，并向该位置移动一步。需要注意，如果目标位置的鱼群过于密集，人工鱼则不会执行群聚行为和追尾行为。人工鱼群会实时维护一个公告牌，记录鱼群中处于最优状态的鱼的位置。达到迭代上限或者若干次迭代后公告牌中的状态始终在一个设定的值内变化，算法停止运行，并输出公告牌中的位置作为求得的最优解。

在本研究的问题背景下，池塘可以看成N维的空间，每个维度有M个坐标，池塘中的每个点就是一个分配方式，鱼在池塘中游动并向状态更优的点聚集，因此，可以基于人工鱼群算法来设计负载调度算法。

## 系统可行性分析

通过前文对研究现状的调研可以得知，目前的边缘平台的调度策略，例如NEP平台的以客户为中心的就近调度策略，在负载均衡的方面表现不是很好，可能使部分资源限制，并导致热点问题的出现。随着边缘计算技术的发展，未来将会有更多边缘计算平台的出现，边缘计算节点的规模、边缘应用和用户的数量也会增加，因此研究一种应用可感知的边缘调度系统是很有意义的。

本次设计将涉及两阶段的实验工作，这两阶段工作都可能在时间、资源或技术上受到制约，开发一个负载调度系统是一项复杂的任务，所以需要进行可行性分析，对当下可利用的资源进行分析，从而用尽可能小的代价进行系统的实现。下面将从技术可行性、经济可行性以及社会可行性三个方面进行可行性分析。

### 技术可行性

本次设计第一阶段的工作将利用StarlingX平台开展，StarlingX是一个成熟的边缘云软件堆栈，本次设计主要在实验室的服务器中使用StarlingX平台搭建模拟边缘集群，并使用StarlingX平台集成的Kubenertes服务在集群中以docker容器的方式部署应用，并使用测试程序多次测量应用的资源占用情况，测试程序使用docker提供的用于python编程的SDK来获取应用的资源占用信息。第二阶段将根据下一章提出的系统设计对系统进行开发实现。利用已有的资源和技术，可以按期完成整个系统，因此本设计具有技术可行性。

### 经济可行性

从经济效益方面来看，由于本实验目前只是一个模拟边缘集群负载调度系统，因此不会产生实际的经济效益。从经济成本方面来讲，本设计的第一阶段主要使用开源的StarlingX平台以及两类开源的云应用在实验室的服务器中进行实验，第二阶段主要在本地个人计算机中使用python进行系统开发，并在实验室的服务器中进行测试，因此本设计并不会带来额外的软硬件成本，主要的成本来自电费以及服务器的维护费用。经济成本处于可接受的范围内，本设计具有经济可行性。

### 社会可行性

本设计目前的成果虽然没有用于实际的边缘平台进行负载调度，但是随着边缘平台的发展，用户和应用规模的增大，在满足应用的延迟需求的条件下，避免热点问题的出现成为边缘平台不得不考虑的问题，本设计将为以后的边缘平台负载调度方面的研究提供一些思路，因此本研究具有社会可行性。

## 本章小结

本章首先提出了本次设计的系统需要解决的问题，然后对问题进行数学建模，用一个目标函数表述集群中资源占用的不均衡程度，并用根据分配方式需要满足的条件，构造若干约束函数对模型中的变量进行约束，介绍了希望系统实现的效果以及预期达到的目标，接下来对系统进行需求分析，然后介绍本系统负载调度算法的技术关键，人工鱼群算法的思想，接下来提出实验的基本方案，最后从技术可行性、经济可行性和社会可行性三个方面进行了可行性分析。

# 边缘负载调度系统设计

本章将分析边缘负载调度系统的功能需求，然后提出系统的顶层设计，介绍模块之间相互作用的方式，然后介绍各个模块的设计。

## 功能需求

（1）边缘负载调度系统能够模拟一个有若干应用运行的边缘集群，并使用负载调度器对全局应用负载进行调度。

（2）边缘集群模拟模块能够模拟各种应用和边缘节点，以及他们的各种行为，例如，应用的产生与销毁，节点初始化等。

（3）负载调度器模块能够根据设计的负载调度策略以及当前集群的信息，计算并输出一种分配方式。

（4）边缘负载调度系统能够以周期为单位持续运行，每个周期都会有应用的变动，以及利用负载调度器重新分配各节点的应用负载，并且每个周期的集群状态都会写入到日志文件中，以供查看测试数据。

## 系统总体设计

边缘集群模拟系统能够模拟一个持续运行的边缘集群，边缘集群首先可以进行初始化，设置集群的位置范围并得到边缘集群的初始状态，集群的初始状态含有边缘集群中各个节点的位置以及每个节点能提供的资源总量的信息，初始化既可以通过模板生成若干节点，也可以手动输入或删除节点。

边缘集群能够以周期为单位持续运行，每个周期集群的状态都会发生变化，每个周期集群中都有若干应用生成，若干正在运行的应用停止运行，释放占用的资源，每个正在运行应用的各个资源的占用量也会发生变化，然后负载调度器会根据当前集群的状态，以及选择的调度策略，对全局的应用负载进行调度。

参考文献

1. Xu, Mengwei, Zhe Fu, Xiao Ma, Li Zhang, Yanan Li, Feng Qian, Shangguang Wang, Ke Li, Jingyu Yang, and Xuanzhe Liu. "From cloud to edge: a first look at public edge platforms." In Proceedings of the 21st ACM Internet Measurement Conference, pp. 37-53. 2021.
2. Shea, Ryan, Jiangchuan Liu, Edith C-H. Ngai, and Yong Cui. "Cloud gaming: architecture and performance." IEEE network 27, no. 4 (2013): 16-21.
3. Haouari, Fatima, Emna Baccour, Aiman Erbad, Amr Mohamed, and Mohsen Guizani. "Qoe-aware resource allocation for crowdsourced live streaming: A machine learning approach." In ICC 2019-2019 IEEE International Conference on Communications (ICC), pp. 1-6. IEEE, 2019.
4. Hung, Chien-Chun, Ganesh Ananthanarayanan, Peter Bodik, Leana Golubchik, Minlan Yu, Paramvir Bahl, and Matthai Philipose. "Videoedge: Processing camera streams using hierarchical clusters." In 2018 IEEE/ACM Symposium on Edge Computing (SEC), pp. 115-131. IEEE, 2018.
5. StarlingX. StarlingX Documentation. <https://docs.starlingx.io/>
6. Harchol, Yotam, Aisha Mushtaq, Vivian Fang, James McCauley, Aurojit Panda, and Scott Shenker. "Making edge-computing resilient." In Proceedings of the 11th ACM Symposium on Cloud Computing, pp. 253-266. 2020.
7. Li, Jiapeng, and Hua Huang. "Research Study on Edge Computing." In 2021 IEEE 6th International Conference on Smart Cloud (SmartCloud), pp. 26-32. IEEE, 2021.
8. Tong, Liang, Yong Li, and Wei Gao. "A hierarchical edge cloud architecture for mobile computing." In IEEE INFOCOM 2016-The 35th Annual IEEE International Conference on Computer Communications, pp. 1-9. IEEE, 2016.
9. Bagchi, Saurabh, Muhammad-Bilal Siddiqui, Paul Wood, and Heng Zhang. "Dependability in edge computing." Communications of the ACM 63, no. 1 (2019): 58-66.
10. Beheshti, Zahra, and Siti Mariyam Hj Shamsuddin. "A review of population-based meta-heuristic algorithms." Int. J. Adv. Soft Comput. Appl 5, no. 1 (2013): 1-35
11. Sherif Akoush, Ripduman Sohan, Andrew Rice, Andrew W Moore, and Andy Hopper. Predicting the performance of virtual machine migration. In 2010 IEEE international symposium on modeling, analysis and simulation of computer and telecommunication systems, pages 37–46, 2010.
12. Ghufran Baig, Jian He, Mubashir Adnan Qureshi, Lili Qiu, Guohai Chen, Peng Chen, and Yinliang Hu. Jigsaw: Robust live 4k video streaming. In The 25th Annual International Conference on Mobile Computing and Networking, pages 1–16, 2019.
13. Jacob Benesty, Jingdong Chen, Yiteng Huang, and Israel Cohen. Pearson corre- lation coefficient. In Noise reduction in speech processing, pages 1–4. Springer, 2009.
14. David Breitgand, Gilad Kutiel, and Danny Raz. Cost-aware live migration of services in the cloud. SYSTOR, 10:1815695–1815709, 2010.
15. Rodrigo N Calheiros, Enayat Masoumi, Rajiv Ranjan, and Rajkumar Buyya. Workload prediction using arima model and its impact on cloud applications’ qos. IEEE Transactions on Cloud Computing, 3(4):449–458, 2014.
16. Christopher Canel, Thomas Kim, Giulio Zhou, Conglong Li, Hyeontaek Lim, David G. Andersen, Michael Kaminsky, and Subramanya R. Dulloor. Scaling video analytics on constrained edge nodes. In Proceedings of the 2nd SysML Conference, 2019.
17. Chris Chatfield. The holt-winters forecasting procedure. Journal of the Royal Statistical Society: Series C (Applied Statistics), 27(3):264–279, 1978.
18. David Chou, Tianyin Xu, Kaushik Veeraraghavan, Andrew Newell, Sonia Mar- gulis, Lin Xiao, Pol Mauri Ruiz, Justin Meza, Kiryong Ha, Shruti Padmanabha, et al. Taiji: managing global user traffic for large-scale internet services at the edge. In Proceedings of the 27th ACM Symposium on Operating Systems Principles, pages 430–446, 2019.
19. Franco Cicirelli, Antonio Guerrieri, Giandomenico Spezzano, and Andrea Vinci. An edge-based platform for dynamic smart city applications. Future Generation Computer Systems, 76:106–118, 2017.
20. Christopher Clark, Keir Fraser, Steven Hand, Jacob Gorm Hansen, Eric Jul, Christian Limpach, Ian Pratt, and Andrew Warfield. Live migration of virtual machines. In Proceedings of the 2nd conference on Symposium on Networked Systems Design & Implementation-Volume 2, pages 273–286, 2005.
21. Mark Claypool and Kajal Claypool. Latency and player actions in online games. Communications of the ACM, 49(11):40–45, 2006.
22. Lorenzo Corneo, Maximilian Eder, Nitinder Mohan, Aleksandr Zavodovski, and Suzan BayhanZ. Surrounded by the clouds. In The Web Conference, 2021.
23. Eli Cortez, Anand Bonde, Alexandre Muzio, Mark Russinovich, Marcus Fon- toura, and Ricardo Bianchini. Resource central: Understanding and predicting workloads for improved resource management in large cloud platforms. In Proceedings of the 26th Symposium on Operating Systems Principles, pages 153–167, 2017.
24. Christina Delimitrou and Christos Kozyrakis. Quasar: resource-efficient and qos-aware cluster management. In Rajeev Balasubramonian, Al Davis, and Sarita V. Adve, editors, Architectural Support for Programming Languages and Operating Systems, ASPLOS ’14, Salt Lake City, UT, USA, March 1-5, 2014, pages 127–144. ACM, 2014.
25. Haotian Deng, Chunyi Peng, Ans Fida, Jiayi Meng, and Y Charlie Hu. Mobility support in cellular networks: A measurement study on its configurations and implications. In Proceedings of the Internet Measurement Conference 2018, pages 147–160, 2018.
26. Sheng Di, Derrick Kondo, and Walfredo Cirne. Characterization and comparison of cloud versus grid workloads. In 2012 IEEE International Conference on Cluster Computing, pages 230–238, 2012.