密级：　　保密期限：

北京郵電大學

硕士学位论文

题目：QoS 感知的服务资源调度与优化研究

姓　　名：李松远

专　　业：计算机科学与技术

导　　师：程渤

学　　院：计算机学院（国家示范性软件学院）

2021 年 5 月 31 日

Confidentiality Level: **Secrecy Period:**

BEIJING UNIVERSITY OF POSTS AND TELECOMMUNICATIONS

Thesis for Master Degree

Title: **QOS-AWARE SERVICE RESOURCE SCHEDULING AND OPTIMIZATION**

Candidate: **Li, Songyuan**

Subject: **Computer Science and Technology**

Supervisor: **Cheng, Bo**

Institute: **School of Computer Science**

(National Pilot Software Engineering School)

May 31st, 2021

独创性（或创新性）声明

本人声明所呈交的论文是本人在导师指导下进行的研究工作及取得的研究成果。尽我所知，除了文中特别加以标注和致谢中所罗列的内容以外，论文中不包含其他人已经发表或撰写过的研究成果，也不包含为获得北京邮电大学或其他教育机构的学位或证书而使用过的材料。与我一同工作的同志对本研究所做的任何贡献均已在论文中作了明确的说明并表示了谢意。

申请学位论文与资料若有不实之处，本人承担一切相关责任。

本人签名：李松远　　日期：2021.5.31

关于论文使用授权的说明

本人完全了解并同意北京邮电大学有关保留、使用学位论文的规定，即：北京邮电大学拥有以下关于学位论文的无偿使用权，具体包括：学校有权保留并向国家有关部门或机构送交学位论文，有权允许学位论文被查阅和借阅；学校可以公布学位论文的全部或部分内容，有权允许采用影印、缩印或其它复制手段保存、汇编学位论文，将学位论文的全部或部分内容编入有关数据库进行检索。（保密的学位论文在解密后遵守此规定）

本人签名：李松远　　日期：2021.5.31

导师签名：　　日期：2021.5.31

QoS 感知的服务资源调度与优化研究

摘　要

服务计算是一种多学科交叉融合的新兴计算模式。它通过设计和运用计算和信息技术，对各类信息服务与商业服务进行设计、操作、管理和优化，被广泛应用于各个领域。随着服务计算系统的不断扩张和演化，系统中服务供应商和用户规模急剧扩大，服务计算系统的组成结构愈发复杂，系统逐渐呈现生态化趋势。在生态化的服务计算系统中，多用户/服务供应商共同参与到服务计算系统的运行，并形成相互竞争的态势。对于用户而言，需要面对来自其他用户的服务资源竞争而争取得到最优的目标服务/服务资源分配方案，从而获得优质的服务质量（Quality of Service，QoS）。对于服务供应商而言，需要采取适当的服务激励策略以吸引大量用户来购买其服务资源，同时尽可能地满足最多用户的 QoS 需求，从而在竞争型服务市场中占据较大的市场份额。对于服务计算系统而言，则需要提供一种满足多用户公平性的服务资源竞争平台，较好地兼顾服务公平与 QoS 优化，从而促进生态化服务计算系统中的良序竞争与博弈，有助于服务计算系统的可持续发展。

本文针对生态化的服务计算环境，聚焦管理多用户或服务供应商之间的竞争态势，从服务选择、服务资源定价、服务资源管理三个方面出发，建立对应的 QoS 优化模型，并采用相应的优化方法求解，开展 QoS 感知的服务资源调度与优化研究。本文主要的创新性工作如下：

1. 研究基于公平性指标的多用户服务选择方法。首先，构建 QoS 感知的多用户并发服务选择模型。其次，通过引入最大最小公平性的概念，给出实现最大最小公平性的字典序优化问题描述。然后，分析服务公平优化问题的特有特征和结构，进而设计出基于线性规划迭代的服务公平优化算法（Fairness-aware Concurrent Service Selection，FASS）。在理论上，证明了 FASS 算法的最优性，并且在仿真实验中，充分验证和评估了 FASS 算法的有效性和效率。

2. 研究面向市场的云资源定价机制。首先，对云资源拍卖市场进行建模描述。其次，根据 QoS 感知的用户效用模型，定义基于用户个体理性的云资源购买策略。然后，设计出最大化用户激励的云资源拍卖机制（Price-Incentive Resource Auction，PIRA），其旨在以满足云服务供应商最低利润率γ要求为前提下，激励出最大用户数量来购买云服务资源。与此同时，在理论上证明出 PIRA 机制满足预算可行性、激励相容性、无妒性，这些重要性质有利于增强云资源拍卖机制 PIRA 的鲁棒性。在仿真实验中，PIRA 机制的实际性能被得到充分的验证。
3. 研究体验质量（Quality of Experience, QoE）感知的分布式边缘任务调度和资源管理方法。首先，建立边缘计算系统的服务资源分配模型，并量化分析 QoE 指标与 QoS 指标间的关联关系。其次，给出最大化系统 QoE 水平的优化问题描述。为了解决该优化问题求解的诸多难点与挑战，引入潜在博弈模型，使得多用户的边缘资源分配方案以去中心化的形式被求解得到。然后，基于抢占式 QoE 改进机制、面向多用户合作的消息传递机制，设计了相应的 QoE 感知分布式边缘资源分配算法（QoE-Aware Decentralized Edge Resource Allocation，QoE-DEER），以得到具有最高系统 QoE 水平的纳什均衡解。通过理论证明和仿真实验，充分地验证了分布式 QoE-DEER 算法的性能和收敛性。

关键词： 服务质量（QoS） 体验质量（QoE） 资源管理 资源定价 公平服务选择

QoS-Aware Service Resource Scheduling and Optimization

ABSTRACT

Services computing is a burgeoning interdisciplinary computing paradigm. It effectively devises, operates, manages and optimizes various information services and business services, through the ingenious design and utilization of computing resources and information technologies, which has been widely practiced in diverse fields. With the continuous expansion and evolution of services computing system, the number of service providers and users increases rapidly. Accordingly, the services computing system is gradually with more complicated compositions, which presents an ecological trend. Multiple users/service providers are collectively involved into the operation of the services computing system, and form a competitive situation. For users, they need to encounter the competition from other users for service resources, whose purpose is to win the optimal target service/resource allocation scheme with the high Quality-of-Service (QoS) level obtained. For service providers, they need to take an appropriate incentive strategy to stimulate the maximum users who purchase service resources, in order to meet the QoS demand for the maximum users. It helps to consolidate the maximum market share within the competitive services marketplace. For the services computing system, it necessitates a fair competitive environment for services resources, where service quality and service fairness should be jointly addressed. In this way, a benign competition order is built up within the ecological services computing system, which contributes to the systematic sustainability.

To address these issues, this dissertation primarily focuses on the management of competitive situation between multiple users and service providers, within the ecological services computing environment. From three aspects (i.e., service selection, resource pricing and resource management), the QoS-aware service scheduling and optimization is extensively investigated, where the specific QoS optimization models and solutions are proposed. The dissertation is summarized as follows.

1. Fairness-aware multi-user service selection method is studied. In details, a QoS-aware multi-user concurrent service selection model with constraints is firstly formulated. Then, by means of introducing the Max-Min Fairness (MMF), the lexicographical optimization problem which achieves the MMF is hereby given. According to the uniqueness of MMF optimization problem, a Fairness-aware Concurrent Service Selection algorithm named FASS is proposed, which is the iterative Linear Programming. The optimality of FASS algorithm is theoretically proved, while the effectiveness and efficiency of FASS algorithm are sufficiently verified and evaluated through simulating experiments.
2. Market-oriented cloud resource pricing mechanism is studied. Firstly, the cloud resource auction marketplace is formulated. According to the QoS-aware user utility model, the cloud resource purchase strategy based on individual rationality is defined for users. Then, the Price-Incentive Resource Auction mechanism named PIRA is designed to maximize the user incentive. It targets to stimulate the maximum users who purchase cloud resources, on the premise of the minimum profit-rate requirement by the cloud service provider. In addition, the PIRA mechanism is theoretically proved to satisfy the budget feasibility, incentive compatibility, and envy-freeness. These essential properties can enhance the robustness of the cloud resource auction mechanism PIRA. The actual performance of the proposed PIRA mechanism is extensively verified through simulating experiments.
3. Distributed Quality-of-Experience (QoE)-aware task scheduling and resource management method is studied under the edge computing paradigm. Firstly, the edge resource allocation model is demonstrated and formulated, and then the correlative between QoS and QoE metrics is analyzed quantitatively. The Edge Resource Allocation problem named ERA is given with the objective of maximizing the systematic QoE level across multiple IoT users. To attack the research challenges of solving out the ERA problem effectively, the potential game model is introduced to make the ERA problem optimized in a decentralized manner. On the basis of the preemption-based QoE improvement mechanism and cooperative message-passing mechanism, the QoE-Aware Decentralized Edge Resource Allocation Algorithm named

QoE-DEER is designed to obtain the Nash equilibrium with the highest systematic QoE state. Finally, the performance and convergence of decentralized QoE-DEER algorithm are theoretically proved and experimentally verified.

KEY WORDS: Quality of Service (QoS), Quality of Experience (QoE), Resource Management, Resource Pricing, Fair Service Selection.

目录

第一章 引言

1.1 研究背景

随着信息技术的蓬勃发展，由软硬件共同组成的传统架构模式逐渐演变为面向服务的设计模式。为了方便用户和上层应用的调用，供应商将所提供的软件或硬件资源以预定义的接口形式发布。软硬件资源被封装成抽象的服务以对外提供，仅在接口上披露用户和上层应用所关注的必要信息，简化了用户对资源的直接使用以及上层应用对资源的二次开发。与此同时，因应用户和上层应用不断变化的、多样的应用需求，供应商转变原有的商业模式，从单一的预约制服务订购模式发展为多元的服务订购机制。其中，新兴的按需服务模式允许弹性地订购服务资源，避免了服务资源预约不足而无法满足当前应用需求，或者服务资源预约过量而产生资源浪费。因此，服务计算（Services Computing）以一种新兴的计算模式应运而生，并被广泛应用在商业服务、金融服务、医疗服务等跨界信息服务领域 [1]。

服务计算系统是面向用户的、旨在实现一定业务功能的计算机系统，并运用信息技术为服务的正常运行提供技术和硬件支撑。近年来，诸如互联网、大数据、云计算、物联网、边缘计算等智能信息技术的涌现促进了服务计算技术的发展，并推动了服务计算系统的扩张和演化。期间，服务计算系统是作为支撑各类智能信息应用运行的基础性平台，维护着各类服务资源的有序运行，以供用户自主选择和购买所需的目标服务，并被分配到适当数量的服务资源，从而满足用户的服务质量（Quality of Service，QoS）需求。服务计算不断地扩展和发展其自身的研究内涵，其中服务计算系统的组成结构愈发复杂，系统逐渐呈现生态化趋势 [2]。自此，多用户/服务供应商将共同参与到服务计算系统的运行过程。随着生态化服务计算系统中服务供应商和用户数量的急剧增加，有关 QoS 感知的服务资源调度与优化研究会产生诸多新的科学问题和挑战，亟需进一步的深入调研和探究。

面对多用户的异构应用需求，服务计算系统的管理模块（Operator Module）会执行服务选择或服务组合操作，将供应商的服务资源分配给各个用户，以满足多用户的应用需求，同时权衡多用户之间应用需求的满足程度来维护多用户之间服务资源分配的公平性 [3]。再者，随着越来越多的供应商参与到生态化服务计算环境中，可供用户使用的服务数量急剧增长，执行合适的服务调度算法和用于支撑服务运行的资源管理算法来最大化服务供应商的收益 [4]，促使更多服务供应商积极地参与构建生态化服务计算环境，提升系统的优质 QoS 提供能力。得益于系统的优质 QoS 提供能力，越来越多的新用户也会主动加入到生态化服务计算环境中去，提出自身的服务请求，并让服务供应商向其提供优质 QoS 水平。伴随着更多用户涌入服务生态，如何合理协调多用户的服务资源分配方案以在总

体上提升每位用户的服务满意度也是十分值得研究的科学问题[5]。总而言之，通过维护多用户间服务资源分配的公平性、最大化服务供应商的收益、提升多用户的服务满意度，服务计算系统的生态可持续性可以得到保证。

随着生态化服务计算环境中服务数量的不断增多，服务供应商通常提供了功能相似的服务，但在非功能性（Non-Functionality）方面表现参差不齐。其中，非功能属性中以 QoS 为主要典型代表。在服务计算的研究领域里，如何配置和调度服务资源以最优化 QoS 提供能力一直是研究的热点问题[6]。而着眼于生态化服务计算环境，对 QoS 感知的服务资源调度与优化提出了更加精细的优化要求，并提出新的研究难点和挑战。多用户在生态化服务计算环境中通常处于竞争和博弈关系，每位用户都期待获取具备最优 QoS 水平的服务，而 QoS 最优的服务仅能供给单一用户[7]。多服务供应商之间也存在相互竞争的态势，每位服务供应商都期待将服务提供给用户以赚取尽可能高的服务收益，同时每位用户也能获得优质的 QoS 水平[8]。再者，随着边缘化服务计算架构的日益流行，生态化服务计算系统中各位终端用户广泛地分布于不同的地理位置[9]，这时亟需提出去中心化的新方式来优化多用户的边缘服务资源分配方案。综上所述，本文将针对生态化服务计算环境中多用户/服务供应商间的竞争态势进行有效管理，聚焦有关 QoS 感知的服务资源调度与优化的诸多研究难点和挑战，从服务选择、服务资源定价、服务资源管理三个重要方面出发，提出对应的 QoS 感知优化模型和方法。

1.2 研究内容与意义

1.2.1 实现最大最小公平的多用户并发服务选择

随着面向服务架构（Service-Oriented Architecture，SOA 架构）的日益流行和普及，各式各样的 Web 服务成为了服务计算系统的一部分，被交付提供给用户。这时，亟需设计出相应的服务选择算法，从而被选出的目标服务同时兼顾用户和服务提供商的利益需求。虽然许多现有研究工作已经对服务选择问题进行了深入研究，但是针对生态化服务计算环境中多用户并发提出服务请求的情况，这仍旧是一个有待研究的应用场景。此外，根据具体服务选择的上下文情景，服务选择约束可能存在，比如关于 Web 服务所部署地理位置的约束，服务供应商和用户之间所达成的服务合约等。

因此，本文将从服务公平性的角度出发，研究 QoS 感知的带约束多用户并发服务选择问题，其主要优化目标是在多个用户请求的服务选择结果之间满足最大最小公平性（Max-Min Fairness，MMF）。与此同时，每位用户的 QoS 需求也被会得到保障。服务提供商向用户提供尽可能高 QoS 水平的服务，从而赚取更多的服务收益。具体而言，将实现 MMF 的多用户服务选择问题描述为字典序优化问题。鉴于字典序优化问题所面临的求解挑战，一种实现 MMF 的服务选择框

架（Fairness-aware Concurrent Service Selection，FASS）被提出，其基本思想是将原始字典序优化问题转化为多个线性规划（Linear Programming，LP）子问题。最后，基于真实 Web 服务数据集，实验验证和评估 FASS 算法的有效性和实用性。

1.2.2 最大化用户激励的市场化云资源定价机制

对于云服务供应商来说，需要构建一个面向市场的云服务生态化系统来整合现有云用户，并吸引更多潜在的新用户，从而进一步更多地占据云计算市场份额。每位云用户的资源需求量是价格敏感的；根据即时云资源单位价格，每位云用户再综合考虑其自身的服务竞价预算和 QoS 需求，从而最终确定自己的云资源需求量。基于自身的云资源需求量，每位云用户向云服务供应商购买云资源。与此同时，云服务供应商则会根据动态波动的市场环境而更新即时云资源价格，从而避免过高的云资源价格而导致用户流失，并且防止云资源定价过低而难以覆盖云资源的运营成本（如能耗成本）。

根据上述设计思想，本文提出一种面向云计算市场的价格激励资源拍卖机制（Price-Incentive Resource Auction，PIRA）。得益于所精心设计的市场化云资源定价策略，云用户与云服务供应商之间的利益需求得到了折衷权衡。本文的价格激励资源拍卖机制 PIRA 的优化目标是在保证云服务供应商最低利润率 γ 的前提下，最大限度地激励云用户来购买云服务资源，使得她们选择在云端执行其应用程序。其中，反映用户 QoS 需求的用户效用函数 $v_i(\cdot)$ 被定义。与此同时，云资源拍卖机制 PIRA 也提供了预算可行性（Budget Feasibility）、激励相容性（Incentive Compatibility）和无妒性（Envy-freeness）保证。最后，基于实际数据集进行了仿真实验，从而验证和评估了云资源拍卖机制 PIRA 的性能和有效性。

1.2.3 去中心化 QoE 感知的多用户边缘资源管理

随着在物联网（Internet of Things，IoT）服务与设备的日益流行和普及，新兴的边缘计算架构比传统的云计算架构更具有显著的优势。与云计算的单体架构不同，多台边缘服务器通常被部署在各个物联网设备附近，分散在不同的地理位置上。得益于高性能边缘服务器的辅助，物联网设备本身则不再需要较高的硬件配置，也能完成较复杂的物联网计算任务；同时，鉴于边缘服务器的分散部署，相较于从物联网设备到远程云数据中心所需的网络通信延迟而言，边缘计算架构则具备更低的网络通信延迟。近年来，虽然相关研究人员已经对边缘资源管理/物联网任务调度问题进行了广泛的研究，以实现较低的服务响应时间，但鲜有关注边缘资源分配过程中的用户体验质量（Quality of Experience，QoE）优化问题。QoE 指标是传统 QoS 指标（即服务响应时间）的扩展评价标准，被用于量化描述物联网用户的服务满意度。在生态化服务计算环境中，基于 QoE 优化的服务

资源调度将更具有实际应用意义。

鉴于此，本文将研究以最大化系统 QoE 水平为优化目标的多用户边缘资源分配（Edge Resource Allocation, ERA）问题。然而，由于 ERA 优化问题的 NP 难解性，所以本文将提出一种去中心化的 ERA 问题求解方法来得到多用户边缘资源分配的决策方案，从而得出 ERA 优化问题的等价最优解。具体而言，引入博弈论的观点，将原始 ERA 优化问题描述为潜在博弈模型（Edge Resource Allocation Game，ERAGame）。其中，一种边缘资源抢占机制 PRIM 被嵌入到 ERAGame 博弈模型中，从而向 ERAGame 博弈模型提供了关于系统 QoE 水平的最优性保证。最终，基于合作消息传递机制，设计了 QoE 感知的分布式边缘资源分配算法（QoE-Aware Decentralized Edge Resource Allocation，QoE-DEER），使 ERAGame 博弈过程收敛于具有最高系统 QoE 水平的纳什均衡（Nash Equilibrium，NE）解。在理论证明和实验验证方面，分布式 QoE-DEER 算法的性能和效率都被得到了分析和验证。

1.3 研究难点与挑战

1.3.1 兼顾服务公平与服务质量的多目标优化设计

在多用户/服务供应商共同参与的生态化服务计算系统中,维护多用户/服务供应商间的服务公平是十分必要的，将有利于增强生态化系统的可持续性和鲁棒性。一方面，对于多用户而言，每位用户的服务请求需要被服务供应商公平地看待；换言之，每位用户的 QoS 需求应该被服务供应商同时较好地兼顾，被分别得到最大限度的满足，由服务供应商提供同等优质 QoS 水平的服务。另一方面，对于多服务供应商而言，向每位服务供应商提供公平的服务提供机会也是十分重要的；换句话说，每位服务供应商应该具有相对均衡的机会来向用户提供同等优质 QoS 水平的服务，从而拥有相对同等的机会来赚取服务收益。通过维护生态化服务计算环境的服务公平性，将有助于促进多用户/服务供应商之间的良序竞争关系。

面向多用户服务公平性，本文将关注 QoS 感知的并发服务选择优化模型与求解。首先，给出多用户服务公平性指标的定义——最大最小公平性（MMF）；然后，在 MMF 优化过程中考虑如同时兼顾服务公平与 QoS 优化。不仅每位用户需要被服务供应商一视同仁，并且服务供应商也需要最大限度地向每位用户提供较高 QoS 水平的服务，提升优质 QoS 提供能力。从本质上讲，兼顾服务公平与 QoS 优化属于一种多目标优化问题。一般地，多目标优化问题的描述和求解存在诸多难题与挑战。鉴于此，在本文中，将多用户服务公平问题描述为字典序优化问题，其中充分地权衡了服务公平与 QoS 优化。经过严格的理论推导，字典序优化问题可以被等价转化为多个单目标 LP 子问题来求解。基于此，满足多用户

服务公平性的服务选择最优解能够被高效地求解得到。

1.3.2 服务资源定价与用户资源需求分配间的强耦合性

在生态化的服务计算环境中，若从市场化的角度来看，则用户对于服务资源的需求量通常是价格敏感的。虽然每位用户通常具有异构 QoS 需求，但是在多用户的服务资源竞争过程中，每位用户都期待能够被分配到更多的服务资源，从而获得具有更高 QoS 水平的服务。然而，每位用户的服务资源需求量往往受限于其自身的有限服务预算，从而受到即时资源定价结果的实时调控。一方面，当服务资源价格较高时，系统中整体用户的服务资源需求水平会呈现下行趋势。另一方面，当服务资源价格较低时，整体用户的服务资源需求水平则会随之上升。由此可见，在生态化服务计算市场中，服务资源定价结果与用户资源需求量之间存在较强的耦合关系。如何设计和利用动态的市场化服务资源定价策略来实现特定的市场优化目标是具有科学挑战性的。

为了解决上述研究挑战，本文将剖析生态化服务计算市场的动态性特征，并分析服务资源定价与用户资源需求分配之间的耦合关系，然后提出价格激励的服务资源拍卖机制。具体而言，针对云计算服务环境，首先对多用户的云资源拍卖市场进行建模分析，其次构建 QoS 感知的云用户效用模型，然后定义由用户个体理性驱动的云资源购买策略，最后设计出最大化用户激励的云资源拍卖机制 PIRA。通过详细分析云服务资源定价与用户资源需求分配之间的耦合关系，云用户的资源购买过程、云服务供应商的资源定价过程可以被分解决策，以此增强了云服务资源定价与用户资源需求分配之间的耦合灵活性。与此同时，本文所提出的价格激励资源拍卖机制 PIRA 还满足预算可行性、激励相容性、无妒性，将有助于增强云服务资源拍卖机制的鲁棒性。

1.3.3 多用户竞争博弈过程的无序性

在生态化的服务计算环境中，多位用户共同参与到对服务资源的分配与抢占过程。每位用户都希望被分配到更多的服务资源，从而享有更高的 QoS 水平。由于系统服务资源总量是有限的，多用户之间将因此构成相互竞争的态势，彼此争抢去占据更多的服务资源。鉴于此，生态化服务计算系统将出现多用户的服务资源博弈过程，其中往往存在着多用户竞争/博弈过程的无序性问题。这时，亟需设计出一种良性有序的多用户竞争博弈过程，从而促成多用户之间的良序竞争，为整个生态化服务计算系统的演化提供基础。再者，随着物联网服务和设备的日益流行和普及，服务计算架构逐步趋向分布式的边缘计算架构。特别的是，当边缘计算系统具有庞大的物联网用户规模时，传统的集中决策模式难以适用于如此规模庞大的分布式计算系统。因此，这时亟需提出一种高效的、去中心化的边缘服务资源分配方案，从而在服务资源的调度与优化过程中展现出较好的系统可扩展性。期间，构建一种良好有序的多用户竞争博弈过程将会更加尤为重要。

鉴于此，通过引入潜在博弈模型 ERAGame，本文提出一种分布式边缘任务调度和资源管理方法 QoE-DEER。每位物联网用户提出相应的服务请求，而分布式 QoE-DEER 算法负责将其服务请求调度到合适的边缘服务器，并在边缘服务器内为其分配合适的服务计算资源。在分布式 QoE-DEER 算法中，每位物联网用户的边缘资源分配策略是以去中心化的形式被决策确定的。在 ERAGame 博弈模型中，每位物联网用户将具有一定程度的决策自主性。借助于合作消息传递机制，分布式 QoE-DEER 算法会协调跨用户的边缘资源分配决策过程，以保证分布式 QoE-DEER 算法（ERAGame 博弈过程）收敛于具有最高系统 QoE 水平状态的纳什均衡解。综上所述，上述基于博弈论的研究方法将较好地解决多用户竞争博弈过程的无序性问题。

1.4 论文组织

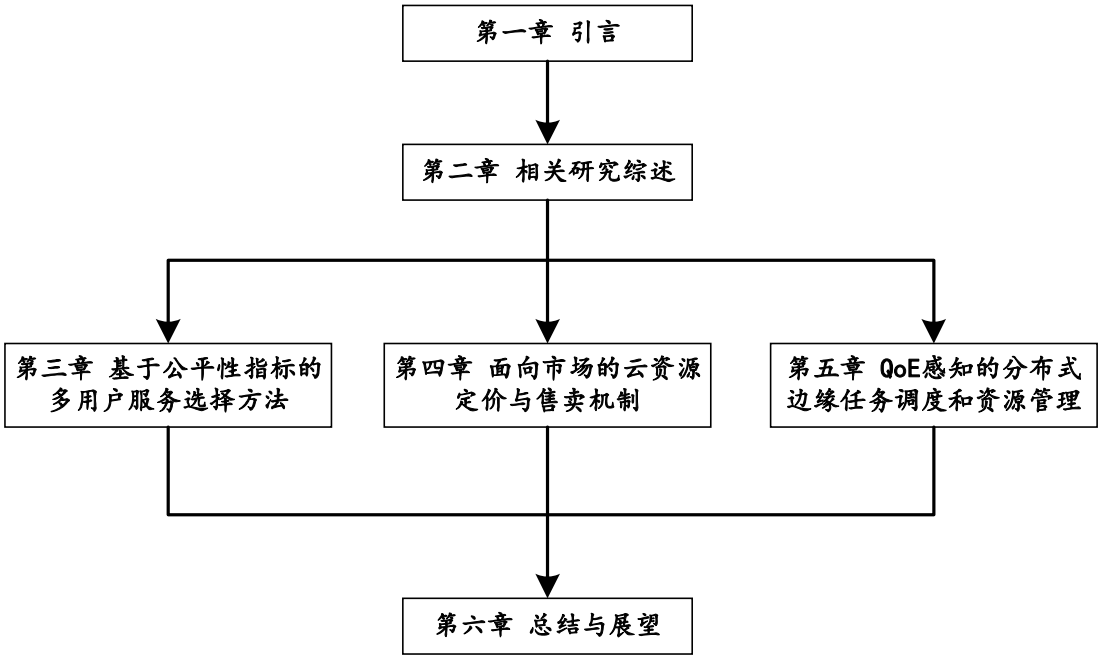


图 1-1. 论文组织结构

本文内容共分为六章，全文组织结构图如图 1-1 所示，主要内容如下：

第一章，引言，即本章。简明阐述了本文的研究背景、研究内容与意义、研究内容与挑战、全文章节组成。

第二章，相关研究综述。针对生态化的服务计算环境，概括性地介绍了 QoS 感知的服务资源调度与优化研究中的主要研究方向，包括服务选择和服务推荐、多用户间竞争与合作、服务资源的定价策略、边缘化的服务计算架构，并且指出了现有研究成果的不足之处，以待进一步探索和研究。

第三章，基于公平性指标的多用户服务选择方法。本章提出了一种实现最大

最小公平性的多用户并发服务选择算法 FASS。

第四章，面向市场的云资源定价与售卖机制。本章提出了一种最大化用户激励的云服务资源拍卖机制 PIRA。

第五章，QoE 感知的分布式边缘任务调度和资源管理。本章提出了一种基于潜在博弈模型 ERAGame 的、去中心化的多用户边缘资源分配算法 QoE-DEER。

第六章，总结与展望。本章对全文的主要研究贡献进行了概括性总结，并且指出了未来工作的探索和研究方向。

第二章 相关研究综述

服务计算的生态化趋势是服务计算领域内新兴的热门研究课题，而 QoS 感知的服务资源调度与优化则一直是服务计算领域备受关注的一项具体研究内容。面向生态化的服务计算环境，有关 QoS 感知服务资源调度与优化的研究可以从服务选择和服务推荐、多用户竞争与博弈、服务资源的定价策略、边缘化的服务计算架构四个方面做相关研究综述。

2.1 服务选择和服务推荐

服务选择和服务推荐是演进和优化服务生态环境的重要研究议题。在云计算和微服务等新型计算架构技术飞速发展的当下，大量 Web 服务被不同的服务供应商在互联网上发布，在 QoS 提供方面良莠不齐。再者，面对动态时变的生态化服务计算环境，单一服务供应商所提供的服务难以持续地向用户提供优质 QoS 水平，存在因服务失效、网路失效而影响用户获取高 QoS 服务的可能性。因此，选择和推荐合适的服务以向用户持续提供优质 QoS 水平，提升用户对生态化服务计算环境的满意度，有利于促进服务计算系统的生态可持续发展。文献[10]设计出一种基于分布式领域知识的服务演进模型来鼓励服务供应商提供优质稳定的 QoS 水平，并淘汰 QoS 表现差且不稳定的服务，最终促成优质服务供应商间协同合作来为用户服务。文献[11]设计并实现了一种基于平台即服务（Platform as a Service，PaaS）的 Web 服务平台 RuCAS，平台上多位服务供应商协作感知用户服务环境的实时变化，并自适应地执行 Web 服务的优化操作，从而最大限度地满足用户的服务需求。针对生态化的服务计算环境，文献[12]运用进化博弈理论来探究服务供应商之间的合作与共享收益关系，并提出服务供应商间形成合作关系的稳态理论。文献[13]分析了生态化服务计算环境中服务与服务、服务与供应商、服务（服务组合）与用户间的动态关联关系，并提出了三层服务网络模型来表征服务生态环境中的动态性和关联性特征，为相关服务选择与推荐操作提供指导。文献[14]从生态化服务计算系统演进的角度出发，运用潜在狄利克雷分布模型和时间序列预测模型，综合考虑网络结构、服务内容描述、服务使用记录来执行时间感知的服务推荐，促使多服务供应商相互协作来持续向用户提供优质 QoS。文献[15]给出合并、分支、并发、产生四种服务组合演化模式，并提出服务组合演化模型的相似度计算方法，最终设计出服务组合推荐框架 DC-SeCo-LDA，其与基准推荐算法相比，在服务推荐性能上有所提高。

为了促进服务计算系统的生态可持续性发展，需要吸引更多的服务供应商加入到服务生态来提供优质服务，促进服务供应商间的良序竞争，在相互竞争中提升服务供应商的优质 QoS 提供能力，最终实现优质 QoS 提供能力的系统级提升。

因此，服务选择与推荐领域中的服务冷启动问题是促进生态化服务计算系统持续发展的重要议题。冷启动服务是指服务计算系统中无使用记录的服务，或者新发布/新创建的服务。文献[16]通过对冷启动服务与非冷启动服务作相似性比较，挖掘出优质的冷启动服务并向用户推荐，进而提升生态化服务计算环境的元素多样性和系统鲁棒性。针对服务推荐的结果，文献[17]给出公平性和可信赖性指标的量化方法，并提出推荐公平性和可信赖性保证的服务推荐机制，赋予冷启动服务相对同等的机会来参与服务推荐过程。文献[18]针对移动服务，运用组合优化技术而设计出一种公平的服务推荐机制，它合理地折衷权衡了服务推荐的准确性和公平性，使得冷启动服务能够合理参与到移动服务生态系统。文献[19]考虑到服务失效、服务新增等动态要素，将生态化服务计算环境中的服务分为新服务、保护服务、普通服务，并分别设计出对应的服务推荐策略，从而提升服务多样性，并减少服务失效的发生概率。文献[20]构建“服务-标签”二部图来量化服务单元之间的竞争关系，并进一步预测可能消亡的服务单元个体，为服务组合的长期可用性提供保障，同时为冷启动服务参与到服务计算系统的运行过程提供空间。

2.2 多用户间竞争与合作

随着生态化服务计算系统内用户数量的急剧增加，系统规模不断地扩大。多用户之间共享着相对有限的服务资源，因而彼此间的服务资源竞争关系也将不断地加剧。鉴于此，面向生态化服务计算环境的多用户间竞争与合作是有关服务资源调度与优化研究中的一项重要研究议题。针对不同用户的异构服务需求和竞价，亟需提出高效科学的服务资源调度与优化方法，以管理多用户间的服务资源竞争关系，促成多用户间的合作行为，进而最终提高生态化服务计算系统的整体优质QoS 提供能力。

面向多用户并发提出的服务请求,文献[21]设计了一种随机的服务资源拍卖机制，用于决策下游云服务供应商的上游云资源购买行为，并统筹调度云用户的服务请求、 对云用户的服务请求进行服务定价，所提出的云资源拍卖机制提供了可信性、高效性、个体理性和社会效用近似最优性保证。文献[22]借鉴“按比例分配资源”的观点，进一步提出了在线服务资源拍卖机制，以实时动态的方式调度多用户的服务请求，在线拍卖算法提供了多项式时间的算法复杂度、有界的社会福利最优化近似比。文献[7]引入博弈论的观点，将多用户服务定价的过程看作纳什均衡的求解过程，提出了多用户服务定价的分布式算法，最终实现最大化服务供应商收益的优化效果。文献[23]旨在维护生态化服务计算系统的有序运行，运用博弈论的观点来优化多用户的服务效用，进而增加云服务供应商的相应效益，为整个系统的持续运行提供效益支持。针对生态化服务计算环境，文献[24]折衷和兼顾用户和服务供应商的利益，提出了一种贯序博弈模型，既满足用户对响应时间的需求，又能提高服务资源的利用率，所提出的方法切实可行。文献[25]着

眼于多用户的整体 QoE 水平，设计了一种面向多用户的服务资源分配算法，该方法能够有效提升系统中用户的平均 QoE 水平。

在生态化服务计算环境中，多用户在相互竞争的同时也会达成合作关系，实现双赢局面，提升服务计算系统的整体优质 QoS 提供水平。为了持续保障用户的 QoS 需求，用户通常倾向于购买足量的服务资源。然而，在动态时变的服务计算环境中，用户的 QoS 需求水平会随时间变化而波动不定，存在所购买的服务资源未被充分利用的可能性。通过将空闲服务资源转售给有额外服务资源需求的其他用户，不仅减少了因订购过量资源所带来的损失，而且还提高了其他用户的 QoS 水平。文献[26]针对用户持有的空闲云资源实例，提出了一种在线空闲资源拍卖机制，对参与空闲资源拍卖过程的竞价者提供了可信性和个体理性保证，所提出的在线空闲资源拍卖算法也具有最优化近似比保障。基于转售 Amazon EC2 云资源实例的真实应用场景，文献[27]设计了一种在线 Amazon EC2 预留实例转售算法，通过预测 Amazon EC2 预留实例的未来资源使用情况来决定是否转售以及转售预留实例的数量，该在线算法提供了相应的最优化近似保证。在文献[27]的基础上，文献[28]进一步改进原有方法，允许 Amazon EC2 预留实例在任意时刻决策是否转售，取消资源转售的时间槽限制，该方法更具有普适应用性。通过跨用户迁移和转售空闲服务资源，资源转售方和资源买入方都受益，有利于促成跨用户间的合作，并进一步提升生态化服务计算系统的优质 QoS 提供能力。

2.3 服务资源的定价策略

服务资源的定价策略是以云计算市场中所采用的各种云资源定价方法为典型代表的。近年来，云计算服务与应用的普遍流行促进了云计算市场中各种云资源定价策略的涌现，包括最早期的基于资源订阅的方法（Subscription-based）[29]、当前流行的按需付费模式（Pay-as-you-go）[30]，以及新兴的市场化定价策略（Market-oriented approach）[31]。

最早期的云资源定价方法可大致追溯至由 Salesforce.com 网站所提供的 SaaS 云订阅定价模型 [29]。在技术实现方面，基于订阅的云资源定价方法是以多租户软件（Software Multitenancy）作为基础技术支撑的。具体而言，每位云用户共同分担公共的云资源运营成本，但仍有各自的私有空间来托管运行自己的应用程序。正因为如此，基于订阅的云资源定价方法提供了相对较低的云资源售卖价格。云用户可以指定某一固定时间段（如 1 年、1 个月）作为云资源订阅时间段，然后以相应的统一定价标准来支付对应的云资源订阅费用。云资源订阅价格会根据不同的 CPU/内存/磁盘配置情况而有所不同，而云用户可以按照其自身 QoS 需求而选择合适的云硬件配置方案。

为了进一步激励更多云用户来选择云计算的商业模式，一种更具用户吸引力的云资源定价策略应运而生，即按需付费模式 [30]。在按需付费模式下，云用户

可以按需购买云资源实例，并且仅按照云资源实例的实际使用时长来收取费用（如以分钟/小时为最小收费单元）。因此，按需付费模式给云用户带来了更高的经济效益，防止云资源实例的过度订阅。截至目前，包括 Microsoft Azure [32]、Google Cloud [33]、Amazon EC2 [34]在内的主流云服务供应商都向云用户提供了按需付费的选项。然而，按需付费的云资源实例通常也是以统一固定的云资源定价标准而配置的，所以该模式并不能充分地反映实时用户需求 [35]和动态云资源运营成本（如能耗成本）[36]。若采用按需付费模式，则云服务供应商难以准确实时地收回云资源运营成本，并且也缺乏从云用户处赚取较高服务收益的能力。

鉴于上述云资源定价方法的局限性，一种更加灵活的云资源定价方法被提出，即市场化的云资源定价策略 [31]。根据实时动态的云资源供需关系，云服务供应商调节即时云资源定价结果，以反映云计算市场的动态实时信息。对于市场化的云资源定价策略而言，最典型的商业案例则是 Amazon EC2 的 Spot 类云资源实例，其采用了基于市场动态性和拍卖过程的云资源定价机制 [37]。Amazon EC2 Spot 类云资源实例充分利用和转售空闲的普通 Amazon EC2 云资源实例；与采用按需付费模式的云资源实例相比，Amazon EC2 Spot 类云资源实例通常有相当大的云资源价格折扣（最高达 90% 的折扣）。从技术实现上讲，Amazon EC2 Spot 类云资源实例的价格是根据云资源供需水平的长期趋势走向而进行动态调整的。借助于市场化的云资源定价策略，云服务供应商和云用户都得到了经济效益上的激励。一方面，得益于更低廉的云资源价格，云用户会主动调整购买行为，订购更多的云服务资源，从而获得更高 QoS 水平；另一方面，云服务供应商可以采取合适的经济激励策略，在市场化的云资源定价策略下从云用户中赚取最多的服务收益 [38]，并保持云计算市场资源供需水平的持续平稳。

2.4 边缘化的服务计算架构

近年来，物联网技术蓬勃发展，应用场景愈加丰富多样。随着物联网设备数目的急剧增加，物联网设备被广泛分布于不同的地理位置。它们被用于收集各项传感数据，并在设备本地进行简单传感数据加工和集成，再上传至服务器远端来执行复杂数据处理操作。再者，诸如手机、行车记录仪等物联网设备呈现移动性特征，在采集传感数据的同时会发生位置移动。因此，一种新型的移动边缘计算架构应运而生，并成为一种新的边缘化服务计算架构，形成新型的生态化服务计算模型。移动边缘计算架构将云服务迁移至邻近物联网设备的边缘服务器；物联网设备将传感数据上传至邻近的边缘服务器之后，再执行复杂的数据处理操作，上述过程加快了传感数据的处理效率，并大幅节省了网络通信带宽。因此，如何高效地向边缘端迁移服务是一项重要的科研议题。文献[39]全面综述了移动边缘计算架构中计算迁移、边缘缓存和服务编排这三个关键研究问题，并从功能增强、QoS 保障和安全可用性三个方面展望了移动边缘计算架构的研究挑战和发展趋

势。文献[40]通过运用约束优化框架，设计了关于边缘数据缓存的优化算法，所设计的启发式缓存迁移算法使得算法复杂度大幅降低，并且适用于实际的大规模物联网服务系统。文献[41]综合考虑低阶马尔可夫模型在不同时刻内存状态之间的相互影响，提出了关于物理内存负载的预测算法，并结合服务迁移收益与动态规划算法，设计了最佳的边缘-云服务迁移策略。

不同于传统的集中式云服务模式，物联网用户所发出的服务请求将在移动边缘计算架构下被调度到不同的边缘服务器。由于服务请求从用户端到边缘服务器端的服务调度过程需要执行跨层数据传输，所以针对移动边缘计算架构，服务请求调度策略需要考虑异步的跨层数据传输过程，这为边缘服务请求调度算法的设计带来了难点和挑战。文献[42]充分考虑了服务迁移和用户移动性，从而设计了物联网用户在边缘服务器间的在线服务调度算法，经过基于实际物联网数据集的仿真实验，验证了所设计方法的性能和效率。文献[43]利用李雅普诺夫优化技术，设计了一种能源高效的、跨边缘端的、在线的服务请求调度算法，该方法兼顾了所产生的服务成本和服务超时惩罚。文献[44]鉴于多用户、多服务供应商参与的边缘化服务生态环境，借鉴博弈论的观点而设计出一种基于分布式合作的多用户任务调度算法，该方法旨在最大化边缘供应商的服务资源效用。文献[45]提出了“端-边-云”融合的服务计算模型，并以多跳模型为基础，设计出基于博弈与合作的物联网任务调度算法，该方法折衷优化了服务响应时间与能源成本。文献[46]在边缘化服务计算架构下给出了关于用户、边缘资源供应商、服务供应商的角色定义，并依此设计出基于三方序贯博弈的任务调度和边缘资源分配算法，该方法旨在最大化边缘资源供应商和服务供应商的服务收益以及物联网用户所获得的QoS 水平。

2.5 讨论与总结

在 QoS 感知的服务资源调度与优化研究领域中，虽然既有研究工作对服务选择和服务推荐、多用户竞争与合作、服务资源的定价策略、边缘化的服务计算架构进行了初步探究，但所提出方法的 QoS 优化目标通常仅为响应时间、能源效率、服务成本等传统优化目标。截至目前，鲜有将 QoS 指标与服务付费、服务定价相结合设计的 QoS/QoE 优化方法，并且尚未见到相关的系统性研究工作，对多用户间的竞争博弈关系进行管理，并研究促成多用户合作的条件和激励方法。鉴于此，本文将从理论角度出发，针对生态化的服务计算环境，进一步深入探索QoS 感知的服务资源调度与优化。本文的研究工作将对生态化服务计算系统的优化和演进具有重要的指导意义。

第三章 基于公平性指标的多用户服务选择方法

3.1 本章引论

随着近年来服务计算技术的普遍流行，服务选择已经成为面向服务架构的重要功能模块。它是一种从海量候选服务中选取目标服务的过程，旨在满足被用户提出的功能性和非功能性需求。伴随着 Web 服务规模的不断扩大，具有同等功能性的多个候选服务能够同时提供给用户选择，但其非功能属性值会参差不齐[47]。非功能属性用于评价候选服务为用户所提供的服务质量（QoS）水平。

QoS 感知的服务选择旨在选出具有最优服务质量的目标服务，其本质上是关于 QoS 优化问题的建模和求解[48]。目前，已经有大量的研究工作提出了有效的服务选择方案，尤其是对于 Web 服务[49, 50]和云计算系统[51, 52]。然而，既有研究文献 [49-52] 主要关注单一用户执行服务选择的应用场景；鲜有相关工作[3]针对多用户并发执行服务请求的情景而展开研究。携有异构 QoS 需求的不同用户分别提出自身的服务选择请求，并在服务平台上为多用户同时执行服务选择操作。换而言之，每位用户请求的服务选择过程应该同步地进行，并且每位用户请求的 QoS 需求都应当被同时兼顾考虑。

在多用户并发服务选择的情景中，不同用户的服务请求也会存在各种不同的选择约束。例如，当移动通信用户请求建立与基站的通信链接时，通常存在着基站选择限制（例如，位置感知[53]）以指定可用基站的有限集合。在内容分发领域，对被多个内容分发节点交叉覆盖的内容用户也有严格的可访问节点限制[54]，即内容用户只能访问已付费的内容分发节点。由此可见，特别是在多用户并发服务选择的情景下，应当充分考虑每位用户的服务选择约束。此外，多用户的服务选择请求共享着功能性相同但服务质量水平各异的候选服务集合。为了获得拥有更高 QoS 水平的目标服务，多个服务选择请求会自发地相互抢占 QoS 最优的候选服务，构成一种多用户竞争态势。因此，亟需一种公平感知的多用户服务选择机制，以协调多用户间的竞争关系。

本章旨在提出一种基于公平性指标的多用户服务选择方案，以解决带约束的 QoS 感知多用户服务选择问题。本章所提出的服务选择方法是基于服务生态化的角度[55]出发，以自顶而下的观点而精心设计的。一方面，用户通常愿意以较合理的价格获得具有较高 QoS 的目标服务。本章所提出的方法同时较好兼顾了每位用户的 QoS 需求，并激励每位用户能够相对公平地获得较高 QoS 的目标服务。另一方面，以不降低其他用户请求的 QoS 水平为前提，每位用户尽可能地获得具有更高 QoS 的目标服务，从而确保了多用户并发服务选择的公平性。借助公平感知算法，不仅有助于生态化服务计算环境增强既有用户群体的黏性，而且还

可以吸引更多的新用户加入到生态化服务计算环境中去。随着服务生态化环境中用户规模的不断增长，服务提供商能够赚取更多服务收益，并进一步激励去开发出具有更高 QoS 的优质服务。基于此，支持生态化服务计算环境可持续发展的服务演进循环圈被完整地建立。

具体而言，本章首先构建了带约束的多用户并发服务选择模型，并将最大最小公平（MMF）的优化目标表述为字典序优化问题。然而，鉴于字典序优化问题的多优化目标和离散优化特征，求得问题精确解通常是 NP 难解的。通过探究本章字典序优化问题的独特结构，发现了两个有益的性质，分别是可分解的凸优化目标、完全幺模线性约束。借助这两个性质，将原始的字典序优化问题等价转换为基于线性规划（LP）的优化问题，从而高效地求解实现最大最小公平地服务选择方案。最后，通过基于真实 Web 服务数据集的仿真实验，验证了本章所提出方法的有效性和实用性。

本章余下部分的内容安排如下：在 3.2 小节中，对多用户并发服务选择进行建模，并形式化描述实现最大最小公平的字典序优化问题。在 3.3 小节中，概述基于迭代的 MMF 优化框架，再给出线性规划等价转换的详细过程。然后，在 3.4 小节中，设计基于真实数据集的仿真实验，以进一步验证本章所提出算法的效果与效率。最后，3.5 小节总结本章的整体研究工作。

3.2 问题描述

本小节首先给出带约束多用户并发服务选择的模型描述，并在此基础上形式化描述基于字典序优化的、实现最大最小公平的服务选择优化问题。

3.2.1 带约束的多用户并发服务选择模型

如图 3-1 所示，多个用户服务请求 $\mathcal{N}=\{1,2,...,N\}$ 被用户提交给 Web 服务平台，以便并发执行服务选择、为多用户同时提供服务。在 Web 服务平台上，多位服务供应商发布了大量的候选服务，以供用户选择。基于每个用户请求的差异化 QoS 需求，服务代理（Service Agent）负责从众多候选服务中拣选出符合个性化 QoS 需求的目标服务。

不失一般地，假定由 M 位服务供应商提供的候选服务集合表示为 $\mathcal{M}=\{C_1, C_2,...,C_M\}$。候选服务集合 C_i 包括众多候选服务 $j\in C_i$，其中 $1\le i\le M$。如本章引论（即 3.1 小节）所述，候选服务集合在多个用户服务请求之间是带约束可共享的。用户请求 n（$1\le n\le N$）的服务选择约束被形式化描述为约束集合 S_n，其中 $i\in S_n$ 表示用户请求 n 可选的候选服务类型（即候选服务集合）。从服务供应商的角度上，服务供应商 i（$1\le i\le M$）的授权用户集合是 $\mathcal{N}_i$，，即仅有用户 $n\in\mathcal{N}_i$

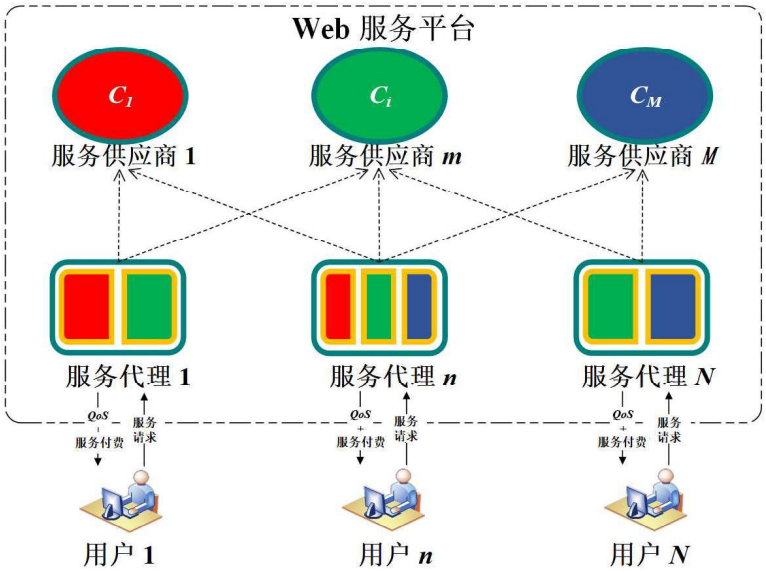


图 3-1. 多用户并发服务选择模型

被允许向服务供应商i发出服务请求以执行服务选择。

服务执行时间（也称作响应时间，Response Time）是服务计算领域中最具代表性的 QoS 评价指标之一。因此，将服务执行时间用作本章的 QoS 评价指标。对于每个候选服务 $j \in C_i$，其服务执行时间被测量为$Q_{i,j}$。

表征服务选择的决策变量由二进制变量 $x_{i,j}^n$ 表示。其中，1 表示候选服务集合 C_i 的第 j 个候选服务被用户请求 n 选定为目标服务；而 0 表示候选服务集合C_i 的第 j 个候选服务未被用户请求 n 选中。所有关联用户请求 n 的决策变量由决策向量 $\boldsymbol{x}_n = \left\{ x_{i,j}^n \mid i \in S_n, j \in C_i \right\}$ 表示，并且所有决策变量 $x_{i,j}^n$ 的可行域组成了决策空间 Θ。基于以上决策模型描述，用户服务请求 n 的执行时间 τ_n 表示为式(3-1)。

$$\tau_n = \sum_{i \in S_n} \sum_{j \in C_i} Q_{i,j} \cdot x_{i,j}^n \qquad \text{式（3-1）}$$

考虑到不同用户请求的个性化 QoS 需求，需要为每位用户量身定制各自的服务等级协议（Service Level Agreement，SLA）[56]，进而给出弹性灵活的服务选择方案，使每位用户的个性化 QoS 需求被满足。具体地说，服务等级协议明确规定了服务供应商应向用户提供的 QoS 水平，以及用户相应需承担的服务费用。在本章中，假定用户以按需付费的模式支付服务费用。按需付费的支付模型在服务计算 [57]、云计算 [32]领域已被广泛地接受和认可。通常地，用户都希望获得更高 QoS 水平的优质服务，即使在合理预算范围内被要求支付更多的服务费用。在按需付费的支付模式下，用户 n 的服务费用由两部分组成。其中一部分是使基本用户 QoS 需求被满足的基础服务费用 a_n，而另一部分是用于提供更高 QoS 水

表 3-1. 主要符号及其定义

符号	定义
$n,N,\mathcal{N}$	用户服务请求的索引值、数量、有限集合
$i,M,\mathcal{M}$	服务供应商的索引值、数量、有限集合
C_i	服务供应商 i 所提供的候选服务集合
j	候选服务集合 C_i 中第 j 个候选服务的索引值
S_n	用户服务请求 n 可供从中选择目标服务的服务供应商集合
$\mathcal{N}_i$,	服务供应商 i 所允许从中选择目标服务的用户服务请求集合
$x_{i,j}^n$	表示用户服务请求 n 是否从候选服务集合 C_i 中选择第 j 个候选服务的二元决策变量，若是则 $x_{i,j}^n=1$，否则 $x_{i,j}^n=0$
Θ	由所有决策变量 $\forall n\in\mathcal{N},i\in S_n,j\in C_i,\ x_{i,j}^n$ 组成的决策空间
$Q_{i,j}$	候选服务集合 C_i 中第 j 个候选服务的 QoS 值
$Q_n^{(ref)}$	用户服务请求 n 的基本 QoS 需求值
τ_n	用户服务请求 n 的实际服务执行时间
$\pi_{i,j}^n$	当用户服务请求 n 从候选服务集合 C_i 中选择第 j 个候选服务时所需支付的服务费用
π_n	用户服务请求 n 所需支付的实际服务费用
a_n	用户服务请求 n 所需支付的基础服务费用
b_n	用户服务请求 n 所需支付的最高额外奖金
ϖ	由所有 $\forall n\in\mathcal{N},i\in S_n,j\in C_i,\ \pi_{i,j}^n$ 项组成的服务支付向量
k,K,ϖ_k	服务支付向量中元素的索引值、数量，第 k 个向量元素
$\lambda_{i,j}^0,\lambda_{i,j}^1$	在等价线性规划变换中依据 λ-技术[58]所引入的辅助实变量

平的最大额外奖金 b_n。在此处，用户 n 的基本 QoS 需求表示为 $Q_n^{(ref)}$。如果用户 n 所获得的实际 QoS 优于 $Q_n^{(ref)}$，则应支付基础服务费用 a_n 和额外奖金。而如果仅用户 n 的基本 QoS 需求 $Q_n^{(ref)}$ 被满足，则仅需支付基础服务费用 a_n。否则，用户 n 的基本 QoS 需求 $Q_n^{(ref)}$ 未被满足，从而不需要支付任何服务费用。当用户 n 从候选服务集合 C_i 中选择第 j 个候选服务作为目标服务时，其服务费用 $\pi_{i,j}^n$ 可以由式（3-2）计算得出。

$$\pi_{i,j}^n = a_n + b_n \cdot \left(1 - \frac{Q_{i,j}}{Q_n^{(ref)}} \cdot x_{i,j}^n\right) \qquad \text{式（3-2）}$$

基础服务费用a_n和额外奖金b_n体现了经用户和服务供应商间协商后所达成的服务定价协议。随着所获得 QoS 水平的提高，用户所需支付的额外奖金呈线性增加[59]；而基础服务费用a_n与基本 QoS 需求的高低（即 $Q_n^{(ref)}$）呈现正相关趋势，即基本 QoS 需求越高，基础服务费用a_n的定价也随之越高。结合式（3-1）和式（3-2），用户n的服务费用可由式（3-3）表示。

$$\pi_n = a_n + b_n \cdot \left(1 - \sum_{i \in S_n} \sum_{j \in C_i} \frac{Q_{i,j}}{Q_n^{(ref)}} \cdot x_{i,j}^n\right) \qquad \text{式（3-3）}$$

3.2.2 实现最大最小公平（MMF）的字典序优化问题

由于服务供应商所发布的众多候选服务是被多用户服务选择请求所共享的，所以本章应当同时考虑每位用户的服务请求，并激发所有用户请求所获得的目标服务在 QoS 方面是具有较高水平的且相对公平的。具体而言，本章的多用户服务选择方案为多个并发用户请求应用了最大最小公平（MMF）的概念，据此提供公平性的保障。关于最大最小公平性的定义给出如下。

定义 1（最大最小公平性，Max-Min Fairnes，MMF）：当剔除了服务支付额高于第n最低服务支付额的这些用户请求后，也无法进一步提高对应第n最低支付额用户的服务付费金额，$n=1,...,N$。这样的多用户服务选择方案符合最大最小公平性。 □

依据最大最小公平性的定义，首先在N个用户请求中最大化最低服务付费，然后在不影响最低服务付费的前提下优化次最低服务付费，依此类推。直到所有用户请求都得到优化，将得到符合最大最小公平性的多用户服务选择方案而终止服务选择的过程。

在多目标优化的研究领域，字典序优化技术[60]将最重要的优化目标赋予最高的优化优先级，恰好与最大最小公平性的定义相匹配。 因此，可以将基于最大最小公平性的服务选择过程严格地形式化描述为字典序优化问题，即以式（3-4）为目标函数，并且以式（3-5）~ 式（3-7）作为约束条件。其中，式（3-5）是用户约束，用于确保每位用户请求仅可从其可用候选服务中选择一个服务；式（3-6）是服务供应商约束，规定不同用户选择不同的目标服务。式（3-7）指定了决策变量$x_{i,j}^n$的定义域，即$\{0,1\}$。

$$lex \max_{x_{i,j}^n \in \Theta} \boldsymbol{\pi} = (\pi_1, \pi_2, ..., \pi_N) \qquad \text{式（3-4）}$$

$$s.t. \quad \sum_{i \in S_n} \sum_{j \in C_i} x_{i,j}^n = 1, \quad \forall n \in \mathcal{N} \qquad \text{式（3-5）}$$

$$\sum_{n \in \mathcal{N}_i} x_{i,j}^n \le 1, \quad \forall i \in \mathcal{M}, \forall j \in C_i \qquad \text{式（3-6）}$$

$$x_{i,j}^{n} \in \{0,1\},\quad \forall n \in \mathcal{N}, \forall i \in \mathcal{M}, \forall j \in C_i \qquad \text{式（3-7）}$$

上述字典序优化问题的优化对象是服务付费向量$\boldsymbol{\pi} \in \mathbb{R}^N$，其每个向量元素$\pi_i$代表用户 n 所需支付的服务费用。最优服务付费向量$\boldsymbol{\pi}^*$在字典序上不小于任何其他可行向量$\boldsymbol{\pi}$。换句话说，$\boldsymbol{\pi}^*$中最小向量元素（即 N 个用户请求中的最低服务付费）应大于所有其他可行向量$\boldsymbol{\pi}$中对应元素值。若所有可行向量$\boldsymbol{\pi}$的最低服务付费都相等，则将$\boldsymbol{\pi}^*$中的次最低服务付费执行最大化操作，使得$\boldsymbol{\pi}^*$中次最低服务付费大于所有其他可行向量$\boldsymbol{\pi}$中对应元素值。依此类推，通过迭代求解字典序优化问题，从而最终获得符合最大最小公平性的服务选择方案，并且实现每个用户请求所需服务费用的最大化。

3.3 多用户公平服务选择方案的求解算法

本小节研究和设计求解最大最小公平服务选择方案的高效算法。 通过引入λ-技术[58]和线性松弛，对原始字典序优化问题进行等效的线性规划（LP）变换，极大地提高了算法求解效率。

3.3.1 基于迭代的 MMF 优化框架

在给出详细服务选择算法前，本部分首先提出一个基于迭代的、实现最大最小公平的服务选择优化框架 FASS。每个用户请求的服务付费参数（即$\mathcal{A}=\{a_n\}_{n=1}^{N}$和$\mathcal{B}=\{b_n\}_{n=1}^{N}$）和 QoS 需求信息（即$\mathcal{Q}^{(ref)}=\{Q_n^{(ref)}\}_{n=1}^{N}$）被作为服务选择优化框架 FASS 的输入参数。在制定多用户服务选择方案时，每位用户所选定的目标服务根据服务支付费用的非递减顺序而被依此确定。例如，在第一轮迭代中，支付服务费用最低的用户请求n^*以最高优先级为其执行服务选择操作，并最大化其服务费用。每个用户请求的服务选择和服务费用最大化操作会被作为一个线性规划子问题来求解。线性规划子问题的推导过程在 3.3.2 小节被详细给出。

一旦用户请求n^*的目标服务被选定后，则可以准备下一轮的迭代优化过程（即为下一个用户请求执行服务选择操作）。在执行下一轮的迭代优化之前，需要“冻结”用户请求n^*的服务选择结果，并且剔除有关决策变量。一方面，关联用户请求n^*的所有决策变量$\{x_{i,j}^{n} \mid n=n^*\}$需要被确定和记录；同时，决策空间$\Theta$也将被剔除关联用户请求$n^*$的维度，即$\Theta \leftarrow \Theta \backslash \{x_{i,j}^{n} \mid n=n^*\}$。另一方面，与用户请求$n^*$所选中目标服务有关的所有决策变量$\{x_{i,j}^{n} \mid x_{i,j}^{n^*}=1, i \in S_n, j \in C_i\}$也需要从决策空间$\Theta$中被剔除。完成如上准备性工作后，可以开始下一轮的迭代优化过程。在完成每一轮的迭代优化后，都需要如上述类似的准备性工作。

Algorithm 1: 实现最大最小公平的服务选择框架(FASS)

Input: 每个用户请求的服务付费参数 $\mathcal{A}$ 和 $\mathcal{B}$, QoS需求信息 $\mathcal{Q}^{(ref)}$.

Output: 服务选择方案 $x_{i,j}^{n}, \forall n \in \mathcal{N}, i \in S_n, j \in C_i$.

初始化 $\widetilde{\mathcal{N}} \leftarrow \mathcal{N}$;
while $\widetilde{\mathcal{N}} \neq \emptyset$ **do**
  $\boldsymbol{x} \leftarrow LP(\mathcal{A}, \mathcal{B}, \mathcal{Q}^{(ref)}, \mathcal{Q}, \Theta)$;
  $\boldsymbol{x}_{n^*} \leftarrow \underset{n \in \mathcal{N}}{\operatorname{argmin}}\ \pi_n$;
  “冻结”用户请求n^*的服务选择结果$\boldsymbol{x}_{n^*}$;
  对于任何 $n \neq n^*$, 设定 $x_{i,j}^{n} \leftarrow 0$;
  $\Theta \leftarrow \Theta \backslash \{x_{i,j}^{n} \mid n = n^*\}$;
  $\Theta \leftarrow \Theta \cap \{x_{i,j}^{n} \mid x_{i,j}^{n^*} = 1, i \in S_{n^*}, j \in C_i\}$;
  $\widetilde{\mathcal{N}} \leftarrow \widetilde{\mathcal{N}} \backslash \{n^*\}$;
return $x_{i,j}^{n}, \forall n \in \mathcal{N}, i \in S_n, j \in C_i$;

重复上述迭代过程，直到所有用户请求都已完成目标服务的选定为止。这时，满足最大最小公平性的多用户服务选择方案被求解得到。值得注意的是，多用户服务选择方案的确定是通过确定的有限次迭代而最终得到的。算法 3-1 是 MMF 迭代优化框架（FASS）的伪代码。

至此，本部分已经详尽陈述 MMF 迭代优化框架的工作流程。在 3.3.2 小节中，通过等价线性规划变换，原始字典序优化问题（即式（3-4）~式（3-7））的多优化目标和离散优化难题会有效解决。

3.3.2 对最大化最低服务费用问题的等价线性规划变换

原始字典序优化问题（即式（3-4）~式（3-7））是基于多目标优化的整数规划问题，通常被认为是 NP 难解问题。为了解决该问题的求解挑战，本部分将原始字典序优化问题分解为 N 个最大化服务付费的单优化目标子问题。具体地，单优化目标子问题的优化目标方程由式（3-8）表示。

$$\max_{x_{i,j}^{n} \in \Theta} \min_{n \in \mathcal{N}} \left(a_n + b_n \times \left(1 - \frac{\tau_n}{Q_n^{(ref)}} \right) \right) \qquad \text{式（3-8）}$$

鉴于决策变量 $x_{i,j}^{n} \in \{0,1\}$，用户请求 n 的服务执行时间 τ_n 也可以由式(3-9)表示。

$$\tau_n = \max_{i \in S_n, j \in C_i} x_{i,j}^{n} \cdot Q_{i,j} \qquad \text{式（3-9）}$$

通过将式（3-9）代入优化目标方程式（3-8），单优化目标子问题能以另一种非线性形式表示，即以式（3-10）为优化目标，并受式（3-5）~ 式（3-7）的约束。

$$\max_{x_{i,j}^{n} \in \Theta} \min_{n \in \mathcal{N},\ j \in S_n, j \in C_i} a_n + b_n \cdot \left(1 - \frac{Q_{i,j}}{Q_n^{(ref)}} \cdot x_{i,j}^{n} \right) \qquad \text{式（3-10）}$$

整数最优解保证：如果线性规划问题的系数矩阵满足完全幺模（Totally Unimodular，TU）特性，则线性规划问题的最优解一定是整数解[61]。在本章所定义的服务选择优化问题中，研究由约束式（3-5）和式（3-6）组成的系数矩阵，以验证系数矩阵是否满足 TU 特性。若是的话，则可以剔除整型定义域约束式（3-7），完全不影响本章服务选择优化问题的正常求解。

引理 1：由约束式（3-5）和式（3-6）组成的系数矩阵符合 TU 特性。

证明：用矩阵 $\mathbf{A}_{s\times t}$ 代表由约束式（3-5）和式（3-6）组成的系数矩阵，其中矩阵 $s=N+\sum_{i\in\mathcal{M}}|C_i|$ 表示用户约束式和服务供应商约束式的数量，而矩阵列数 $t=\sum_{n\in\mathcal{N}}\sum_{i\in S_n}|C_i|$ 表征决策变量向量 $\boldsymbol{x}$ 的维数。

若系数矩阵 $\mathbf{A}_{s\times t}$ 满足如下两个必要条件，则可充分地证明 $\mathbf{A}_{s\times t}$ 符合 TU 特性。首先，鉴于决策变量 $x_{i,j}^{n}\in\{0,1\}$，直接验证 $\mathbf{A}_{s\times t}$ 满足第一个必要条件（即系数矩阵的元素值是 0 或者 ±1）。其次，关于证明 TU 特性的第二个必要条件。具体地，将系数矩阵 $\mathbf{A}_{s\times t}$ 按行划分为两个独立部分 R_1 和 R_2。属于行子集 $\{1,2,\ldots,N\}$ 的部分系数矩阵组成了 R_1，而剩余矩阵部分组成了 R_2，即 $R_1\cap R_2=\varnothing$。根据约束式（3-5），对 R_1 的按列求和结果是维度为 $1\times t$ 的、每个元素值都为 1 的向量。相似地，根据约束式（3-6），对 R_2 的按列求和结果也是维度为 $1\times t$ 的、每个元素值都为 1 的向量。因此，可以推导出结论：$\forall j\in\{1,2,\ldots,t\}$，$\sum_{i_1\in R_1}a_{i_1,j}\le1$ 和 $\sum_{i_2\in R_2}a_{i_2,j}\le1$ 成立。这进一步说明 $\mathbf{A}_{s\times t}$ 满足证明 TU 特性的第二条必要条件，即 $\forall j\in\{1,2,\ldots,t\}$，$\left|\sum_{i_1\in R_1}a_{i_1,j}-\sum_{i_2\in R_2}a_{i_2,j}\right|\le1$。综上所述，系数矩阵 $\mathbf{A}_{s\times t}$ 符合 TU 特性。 □

从引理 1 可以得出，只要服务选择优化问题存在最优解，就一定是整数最优解。鉴于此，可以对整型定义域约束式（3-7）进行定义域松弛。经过整数定义域松弛，决策变量 $x_{i,j}^{n}$ 的定义域变为

$$x_{i,j}^{n}\in\mathbb{R}^{+},\forall n\in\mathcal{N},i\in S_n,j\in C_i \qquad \text{式（3-11）}$$

等价的凸优化目标：根据字典序优化的定义，能够通过求解式（3-12）所定义的字典序优化问题来等价获得式（3-10）所定义的最优服务选择方案。式（3-10）和式（3-12）的共同优化目标是最大化 n 个用户请求中的最低服务付款，即最大化向量 ϖ 中的最小元素。

$$\underset{x_{i,j}^{n}\in\Theta}{\operatorname{lexmax}}\quad \varpi=\left(\pi_{i,j}^{n}\,|\,n\in\mathcal{N},i\in S_n,j\in C_i\right) \qquad \text{式（3-12）}$$

为了得到线性优化目标方程，首先针对性设计出可分解的凸优化目标方程 $\xi(\varpi)$，作为从非线性优化目标到线性优化目标转化的中间数学表达形式，如式

（3-13）所示。向量ϖ的第 k 个元素用ϖ_k表示。

$$\xi(\varpi)=\sum_{k=1}^{|\varpi|}|\varpi|^{-\varpi_k}=\sum_{k=1}^{K}K^{-\varpi_k} \qquad \text{式（3-13）}$$

引理 2： $\xi(\cdot)$ 颠倒了原始字典序优化问题的优化优先级（$\succeq$），即

$$\varpi\left(\mathbf{x}^*\right)\succeq\varpi(\mathbf{x})\Leftrightarrow\xi\left(\varpi\left(\mathbf{x}^*\right)\right)\leq\xi\left(\varpi(\mathbf{x})\right)$$

证明：因为在文献[62]中已经给出类似问题的详细证明过程，所以此处仅给出简化的证明过程。不妨设定向量$\mathbf{g}$ 和$\boldsymbol{\rho}$ 满足条件$\mathbf{g}\prec\boldsymbol{\rho}$，并设$\tilde{k}$ 为$\langle\boldsymbol{\rho}\rangle-\langle\mathbf{g}\rangle$向量差的首个非零元素的索引。这表明着，对于$\forall k\in\{1,\ldots,\tilde{k}-1\}$则$\langle\mathbf{g}\rangle_k=\langle\boldsymbol{\rho}\rangle_k$，而$\langle\mathbf{g}\rangle_{\tilde{k}}<\langle\boldsymbol{\rho}\rangle_{\tilde{k}}$。基于此，设$\langle\mathbf{g}\rangle_{\tilde{k}}=k$，从而$\langle\boldsymbol{\rho}\rangle_{\tilde{k}}\geq k+1$。

$$\begin{aligned}\xi(\mathbf{g})&=\sum_{k=1}^{\tilde{k}-1}K^{-\langle\mathbf{g}\rangle_k}+K^{-\langle\mathbf{g}\rangle_{\tilde{k}}}+\sum_{k=\tilde{k}+1}^{K}K^{-\langle\mathbf{g}\rangle_k}>\sum_{k=1}^{\tilde{k}-1}K^{-\langle\mathbf{g}\rangle_k}+K^{-\langle\mathbf{g}\rangle_{\tilde{k}}}+(K-\tilde{k})\cdot 0\\&=\sum_{k=1}^{\tilde{k}-1}K^{-\langle\mathbf{g}\rangle_k}+K^{-k}\end{aligned} \qquad \text{式（3-14）}$$

$$\begin{aligned}\xi(\boldsymbol{\rho})&=\sum_{k=1}^{\tilde{k}-1}K^{-\langle\boldsymbol{\rho}\rangle_k}+K^{-\langle\boldsymbol{\rho}\rangle_{\tilde{k}}}+\sum_{k=\tilde{k}+1}^{K}K^{-\langle\boldsymbol{\rho}\rangle_k}\leq\sum_{k=1}^{\tilde{k}-1}K^{-\langle\boldsymbol{\rho}\rangle_k}+(K-\tilde{k}+1)\cdot K^{-\langle\boldsymbol{\rho}\rangle_{\tilde{k}}}\\&\leq\sum_{k=1}^{\tilde{k}-1}K^{-\langle\boldsymbol{\rho}\rangle_k}+K^{-k}\end{aligned} \qquad \text{式（3-15）}$$

鉴于$\sum_{k=1}^{\tilde{k}-1}K^{-\langle\mathbf{g}\rangle_k}=\sum_{k=1}^{\tilde{k}-1}K^{-\langle\boldsymbol{\rho}\rangle_k}$，可以推导出$\xi(\mathbf{g})>\xi(\boldsymbol{\rho})$。同时，显而易见得出$\mathbf{g}=\boldsymbol{\rho}\Rightarrow\xi(\mathbf{g})=\xi(\boldsymbol{\rho})$。综上，$\boldsymbol{\rho}\succeq\mathbf{g}\Rightarrow\xi(\boldsymbol{\rho})\leq\xi(\mathbf{g})$ 成立。更进一步地，通过求逆反命题 [62]，可以得出 $\xi(\boldsymbol{\rho})\leq\xi(\mathbf{g})\Rightarrow\boldsymbol{\rho}\succeq\mathbf{g}$ 成立。 □

从引理 2 可以得出，基于字典序优化（i.e., 式（3-12））的优化目标方程等价于可分解的凸优化目标方程$\xi(\varpi)$，如式（3-16）所示。

$$\text{lexmax}_{x_{i,j}^n\in\Theta}\,\varpi\Leftrightarrow\min_{x_{i,j}^n\in\Theta}\xi(\varpi)=\sum_{n\in\mathcal{N}}\sum_{i\in S_n}\sum_{j\in C_i}K^{-\pi_{i,j}^n} \qquad \text{式（3-16）}$$

其中$\xi(\varpi)$为对所有求和项$K^{-\pi_{i,j}^n}$的累加和函数，是关于决策变量$x_{i,j}^n$的凸函数。通过求解优化目标方程式（3-13），可等价得到由式（3-10）所定义优化问题的优化解。将式（3-3）代入式（3-13），得到式（3-13）的数学展开式，如下。

$$\min_{x_{i,j}^n\in\Theta}\sum_{n\in\mathcal{N}}\sum_{i\in S_n}\sum_{j\in C_i}K^{-\left[a_n+b_n\times\left(1-\frac{Q_{i,j}}{Q_n^{(ref)}}\cdot x_{i,j}^n\right)\right]} \qquad \text{式（3-17）}$$

等价的线性规划变换：为了高效求解得到多用户服务选择方案，本章引入 λ-技术 [58] 来执行等价线性规划（LP）变换，最终得到线性优化目标方程式（3-21）。

具体而言，每个凸函数项 $K^{-\pi_{i,j}^n}$ 被应用 λ-技术而转换成另一数学形式 $\psi_{i,j}^n\left(x_{i,j}^n\right)$，其定义如下。

$$\psi_{i,j}^n\left(x_{i,j}^n\right)=\sum_{p\in\{0,1\}} K^{-\left[a_n+b_n\times\left(1-\frac{Q_{i,j}}{Q_n^{(ref)}}\cdot p\right)\right]}\lambda_{i,j}^{n,p}=K^{-(a_n+b_n)}\lambda_{i,j}^{n,0}+K^{-\left[a_n+b_n\times\left(1-\frac{Q_{i,j}}{Q_n^{(ref)}}\right)\right]}\lambda_{i,j}^{n,1} \quad \text{式（3-18）}$$

得益于整数最优解保证和新引入的权重变量 $\lambda_{i,j}^{n,0},\lambda_{i,j}^{n,1}\in\mathbb{R}^+$，决策变量 $x_{i,j}^n$ 的定义域可以由离散域 $\{0,1\}$ 扩展为连续正实数域 $\mathbb{R}^+$。$x_{i,j}^n,\lambda_{i,j}^{n,0},\lambda_{i,j}^{n,1}$ 被式（3-19）~ 式（3-20）约束。

$$\sum_{p\in\{0,1\}}\lambda_{i,j}^{n,p}=\lambda_{i,j}^{n,0}+\lambda_{i,j}^{n,1}=1 \quad \text{式（3-19）}$$

$$x_{i,j}^n=\sum_{p\in\{0,1\}} p\cdot\lambda_{i,j}^{n,p}=\lambda_{i,j}^{n,1} \quad \text{式（3-20）}$$

综上所述，可以得到等价于由式（3-10）所定义优化问题的 LP 优化问题，其线性优化目标方程式为式（3-21）。

$$\min_{\boldsymbol{x},\boldsymbol{\lambda}}\sum_{n\in\mathcal{N}}\sum_{i\in S_n}\sum_{j\in C_i}K_0\cdot\lambda_{i,j}^{n,0}+K_1\cdot\lambda_{i,j}^{n,1} \quad \text{式（3-21）}$$

$$\text{s.t.}\ \sum_{i\in S_n}\sum_{j\in C_i}x_{i,j}^n=1,\forall n\in\mathcal{N}$$

$$\sum_{n\in\mathcal{N}_i}x_{i,j}^n\le 1\forall i\in\mathcal{M},\forall j\in C_i$$

$$x_{i,j}^n=\lambda_{i,j}^{n,1}\quad\forall n\in\mathcal{N},i\in S_n,j\in C_i$$

$$\lambda_{i,j}^{n,0}+\lambda_{i,j}^{n,1}=1\quad\forall n\in\mathcal{N},i\in S_n,j\in C_i$$

$$x_{i,j}^n,\lambda_{i,j}^{n,0},\lambda_{i,j}^{n,1}\in\mathbb{R}^+\forall n\in\mathcal{N},i\in S_n,j\in C_i$$

$$K_0=K^{-(a_n+b_n)},K_1=K^{-\left[a_n+b_n\times\left(1-\frac{Q_{i,j}}{Q_n^{(ref)}}\right)\right]}$$

定理 1：通过迭代求解每个用户请求对应的 LP 优化问题（即式（3-21）），可以得到与原始字典序优化问题（即式（3-4））最优解等价的多用户服务选择方案。

证明：根据引理 1，具有 TU 特性的系数矩阵保障了由式（3-17）所定义优化问题的整数最优性，这也保证了 LP 优化问题（即式（3-21））的整数最优性。此外，从由式（3-17）所定义优化问题而获得的最优服务选择结果也与单目标优化子问题（即式（3-10））的最优服务选择结果一致。通过迭代求解 n 个用户请求的单目标优化子问题，可以得到原始字典序优化问题（即式（3-4））的等价最优解，即实现最大最小公平的多用户服务选择方案。 □

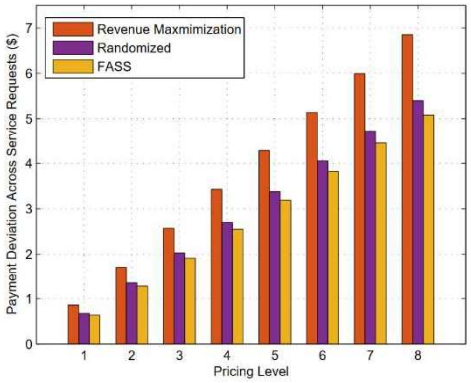


图 3-2. 不同算法下的多用户服务付费偏差

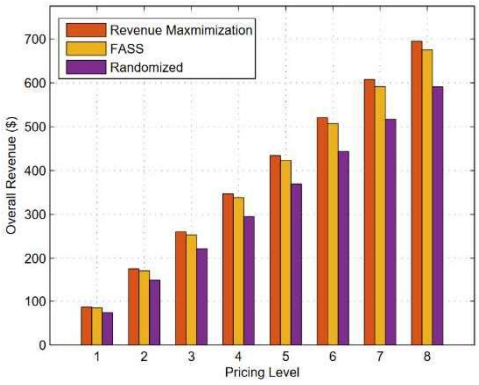


图 3-3. 不同算法下的服务供应商总收益

根据定理 1，通过高效的 LP 算法（如单纯形法，内点法等）和求解器（如 MOSEK [63]，CPLEX [64] 等），能够被高效地求解得到实现最大最小公平的多用户服务选择方案。

3.4 实验验证

3.3.1 实验设置

在本章的实验设置中，分别将服务供应商和用户请求数量设置为 9 和 10，即 $\mathcal{M}=9$ 和 $\mathcal{N}=10$。具体而言，共有 9 个服务供应商各自维护自身的候选服务集合，并且共有 10 位用户同时向服务平台发出服务选择请求。服务供应商所维护候选服务的 QoS 值源自 WSDream 数据集 [65]，该数据集测量了来自不同地理位置的 5825 种真实 Web 服务的响应时间。在 5825 种 Web 服务中，有 9 种被随机选择作为服务供应商，而来自不同地理位置的同种 Web 服务作为候选服务集合。在用户端，有 10 位用户同时发出服务选择请求，其中服务选择约束限定了可选择的服务供应商，并且每位用户对应一个服务选择代理来协调和管理服务选择过程。

本章实验使用 C ++编程语言进行仿真实验，并调用 IBM CPLEX 求解器 [64] 来求解 LP 优化问题。实验仿真在 Ubuntu 18.04 LTS 操作系统上进行；用于实验的处理器为 AMD Ryzen7 2700U 3.8GHz 64 位，物理内存大小为 8.0 GB。

3.3.2 实验结果

为了全面验证本章所提出服务选择算法的最大最小公平性和最优性，设置不同级别的服务供应商定价策略（即 a_n+b_n）以分别评估多用户请求的服务付费偏差、服务供应商的总服务收益，如图 3-2 和图 3-3 所示。服务供应商的定价策略被分别设定为从 1 到 8 的八个定价级别，其中 1 表示最低的服务定价策略，而 8

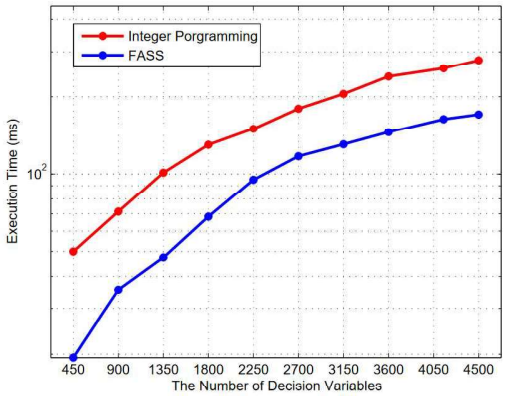


图 3-4. 不同优化问题规模下的算法执行时间

则表示最高的服务定价策略。较高的服务定价级别意味着候选服务被更高地定价。

本章所提出的 FASS 算法与服务收益最大化算法、随机化算法这两个基准算法作比较。服务收益最大化算法仅旨在最大化多用户请求的总服务收益，而不考虑不同用户请求间的服务付费或 QoS 指标偏差；在随机化算法中，每个用户请求从其候选服务集合中随机选择一个候选服务作为目标服务。在本章中，有关随机化算法的实验结果是经过 1000 次重复执行随机化算法而得到的。

一方面，多用户请求间的较低服务付费偏差体现了多用户并发服务选择过程的公平性。鉴于 FASS 算法引入了最大最小公平性的概念，FASS 算法相对于服务收益最大化算法、随机化算法而言具有最低的服务付费偏差，如图 3-2 所示。在服务付费偏差方面，随机化算法排在次低位，而服务收益最大化算法具有最高的服务付费偏差，表明其没有提供任何公平性保证。另一方面，获得更高收益的服务供应商往往能够赚取更高的服务利润。在评估服务供应商收益方面，服务收益最大化算法被用作最佳基准算法。随机化算法，由于盲目的服务选择行为而所获得的服务收益最少。虽然 FASS 算法并没有排在服务供应商收益的首位，但是与服务收益最大化算法之间仅存微小差距，该微小差距在可接受的数值范围内。由此可见，FASS 算法尽管牺牲了部分服务收益，但在多用户并发服务选择的应用场景中提供了公平性保证。

此外，通过测量不同优化问题规模下的算法执行时间来验证 FASS 算法的实用性。由于解决原始字典序优化问题（即式（3-4））非常耗费计算资源和算法求解时间，因此本部分以求解整数规划问题（即 $x_{i,j}^n \in \{0,1\}$ ）的算法执行时间作为比较对象，分析 FASS 算法在不同优化问题规模下的算法执行时间，如图 3-4 所示。优化问题规模由决策变量总数衡量，从 450 个决策变量（即 $x_{i,j}^n$ ）到 4500 个决策变量；图 3-4 中的每个数据点是经过 20 次重复执行算法，然后求均值得到。在优化问题规模的不断增长下，两种算法的执行时间都保持接近线性增长。然而，FASS 算法比整数规划算法快了 153％至 258％。具体地，FASS 算法可以在几十或几百毫秒内完成，远远少于 1 秒。由此可见，本章所提出的 FASS 算法是实际

有效、现实有用的。

3.5 本章小结

当多位用户共享生态化服务计算环境中的多个候选服务时，公平性是服务选择问题中的重要考虑指标。从多用户公平的角度，本章研究了 QoS 感知的多用户服务选择问题，其中还充分考虑了服务选择约束。为了实现最大最小公平性，我们将多用户服务选择问题描述为字典序优化问题。然后，通过引入 λ 技术和线性松弛理论，设计了基于线性规划的高效迭代算法 FASS，能够以较低的计算开销解出实现最大最小公平的服务选择方案。最后，基于真实 Web 数据集的仿真实验，本章所提出的 FASS 算法得到充分的实验验证。

第四章 面向市场的云资源定价与售卖机制

4.1 本章引论

鉴于云计算所独特具备的高性能、低成本服务优势，云计算在近几年获得了蓬勃的发展机会，并且吸引了大量用户选择在云端托管执行他们应用程序 [66]。随着云用户规模的不断扩大，云资源管理已经成为云计算领域中的一项重要研究课题。为了不断地向更多云用户提供较高的 QoS，云服务供应商通常将部署更多云资源以供云用户使用。然而，对于云服务供应商自身来说，过度的云资源扩张部署会导致资源运营成本的大幅增加，从而降低了云服务供应商所获得的利润水平。由此可见，云服务供应商亟需实现一种成本高效的云资源管理方法，而不仅仅是纯粹的云资源容量扩容。

凭借一种高效的云资源管理方法，云用户和服务供应商的服务/利润需求都应该被满足，但是双方需求往往是矛盾冲突的 [21]。具体而言，云用户更愿意以较低廉的价格来购买更多的云资源，从而享有更高的 QoS 水平；云服务供应商则则希望从云用户处赚取更多的服务收益，进而获得更多的服务利润。在一定程度上，由于难以忽视的云资源运营成本或者有限的云资源容量，云用户和云服务供应商的需求目标是相互冲突的。云服务供应商不可能完全忽略自身的云资源运营成本或者有限的云资源容量，任意为云用户索取大量云计算资源，以满足云用户过高的 QoS 需求。

面向市场的云资源定价策略是一种解决云用户与服务供应商间需求目标冲突的有效方法 [67]。云服务供应商拥有对云资源的定价权，旨在实现既定的盈利目标；而云用户会受最新资源价格的激励，在其有限购买能力（即竞价预算）内购买一定数量的云资源，从而实现自身用户效用的最大化，获得自己最期望的 QoS 水平 [68]。云用户的资源购买需求会伴随着云资源价格的波动而不断变化。如果云服务供应商配置较低的资源价格，则所有云用户的资源需求水平会提高。相反地，云用户的资源需求水平会随之紧缩。综合上述分析而知，本章需要制定出面向市场的弹性资源定价策略，以在云用户和服务供应商的矛盾需求目标之间找到一个平衡点。

大多数云服务供应商（如 Amazon EC2[32]、Google Cloud[33]、Microsoft Azure[34]）通常向云用户提供了各式各样的资源计费和付费方式，并且给出各种预配置的虚拟机以满足不同云用户的异构服务需。然而，目前云计算市场通常采用的资源定价方案是基于固定单位价格的，如按期订阅 [29]、按按需付费 [30] 等。选择按期订阅模式的云用户是在预先指定的时间期限内租赁和使用云资源；而选择按需付费的云用户则可根据其实际使用时间和资源需求量而购买对应数量的云资源，云资

源付费按照实际使用时间和资源使用量收取。这两种定价方法都是基于固定定价的，没有考虑时变的市场动态（如用户资源需求、云资源运营成本随时间变化等）。相反地，面向市场的云资源定价策略体现了较灵活的市场弹性，以感知即时市场波动。

本章研究面向市场的云资源定价和售卖策略，并设计了两种基于多用户的云资源拍卖机制，分别实现了云用户激励、云服务供应商收益的最大化。云用户激励的最大化，指使得最大数量的用户被服务于云计算平台，这有利于云服务供应商加强自身的市场份额，在云计算市场中获得竞争优势；而最大化云服务供应商的服务收益，可以进一步提高云服务供应商的利润水平。在多用户的云服务环境里，每个云用户会随着时间推移而动态提出服务竞价请求，而云服务供应商也会相应调整云资源价格，以调节云用户的总体资源需求水平。云服务供应商以保障最低利润率要求为前提来执行市场化的云资源定价策略，从而诱导出最大化的用户激励，或者从云用户处赚取最多的服务收益。对最低利润率要求的保障将有助于收回云资源运营成本（如能耗成本），并从云用户处获得丰厚的服务利润。

不同于文献 [69-71] 采用次高价投标拍卖模型（Second-price Auction）设计云资源拍卖过程，本章所设计的云资源拍卖过程是基于首价密封拍卖模型（First-price Auction）的。在次高价投标拍卖模型中，针对不同竞拍者（即云用户）会执行不同的单位资源定价标准；而在首价密封拍卖模型中，对每个竞拍者（即云用户）一视同仁，采用统一的单位资源定价标准。相比较而言，首价密封拍卖模型与“相同商品应被同等定价”的通识观点保持了一致，可以被广泛地接受。除此之外，本章所提出的云资源拍卖机制还满足了几项重要性质，包括预算可行性（Budget Feasibility）、激励相容性（Incentive Compatibility）、无妒性（Envy-freeness）。这有利于提高云资源拍卖机制的可持续性和鲁棒性。

本章余下部分的内容安排如下：在 4.2 小节中，提出面向多用户的云资源拍拍市场，并对云用户效用进行建模分析。在 4.3 小节中，给出基于用户个体理性的云资源购买策略（Rational Resource Demand Allocation，RARD），云用户根据自身资源需求购买云资源。然后，在 4.4 小节中，设计出最大化用户激的云资源拍卖机制 PIRA，其中给出最大化服务收益的云资源定价算法（Revenue-Optimal Resource Pricing，RORP），并分析和证明云资源拍卖机制 PIRA 所满足的重要性质。在 4.5 小节中，通过进行基于实际数据集的仿真实验，实验验证和评估了云资源拍卖机制 PIRA 的性能和有效性。最后，4.6 小节对本章研究工作进行总结。

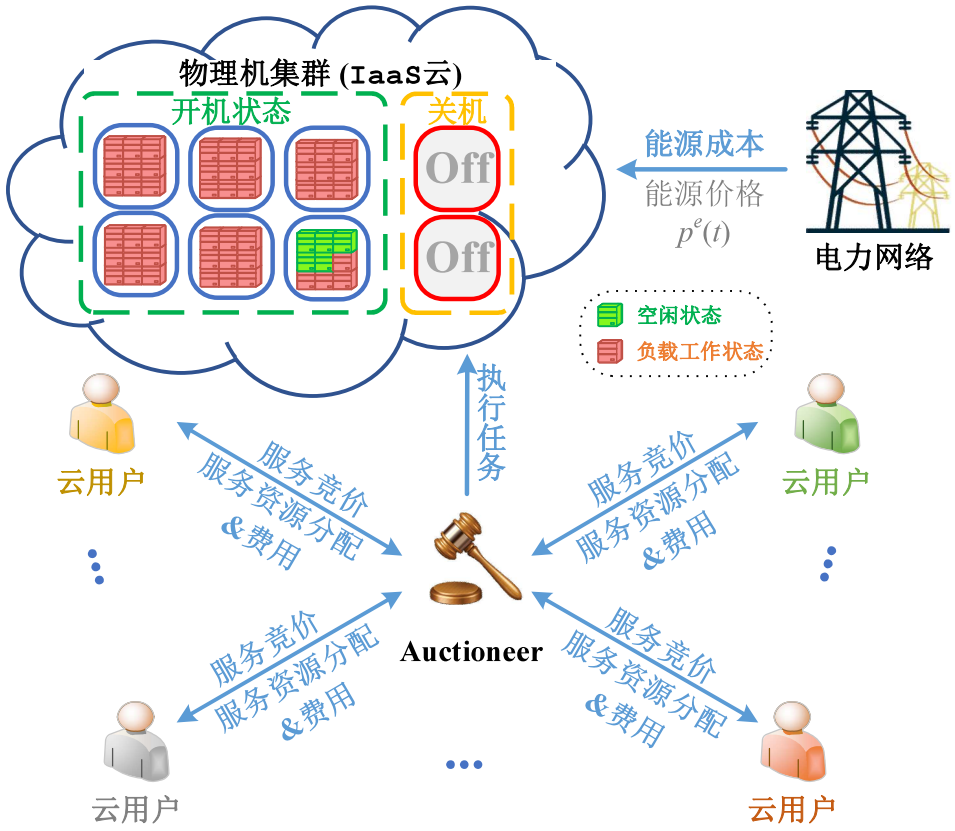


图 4-1. 面向多用户的云资源拍卖市场

4.2 模型设计

4.2.1 云资源拍卖市场

系统概述：图 4-1 展示了面向多用户的云资源拍卖市场，其基于时间槽动态操作拍卖过程。每个时间槽的时间长度为 ，用符号 索引。在每个时间槽 ，各位云用户决定是否发起服务竞价请求；云用户之间形成竞争态势，以希冀获取更多云资源、拥有更高的 QoS 水平。云服务供应商则会在时间槽 t 初端收集所有最新的服务竞价请求，并基于用户竞价信息制定出即时云资源价格、给出当前时间槽的多用户云资源分配方案 $A(t)$。如果云用户 u_i 的服务竞价请求被接受，则云用户 u_i 会被分配单位量为 $a_i(t)$ 的云资源。值得注意的是，单位量为 $a_i(t)$ 的云资源可以被打包为基于资源需求量的定制化 VM [72]；云用户 u_i 在定制化 VM 里执行云托管应用程序。每位云用户 u_i 会按照 $p(t)\cdot a_i(t)$ 收取云服务费用。

云用户：共有 N 位云用户跨时间槽发起服务竞价请求。在时间槽 t 期间，则有 $N(t)\le N$ 个云用户（用符号 $\mathcal{U}(t)$ 标记）发起服务竞价请求。N 位云用户组成了云用户集合 $\mathcal{U}=\{u_1,\ldots,u_N\}$，并且 $\mathcal{U}(t)\subset\mathcal{U}$。不失一般性地，假定每个云用户 u_i 在时间槽 t 期间仅最多发起一条服务竞价请求；在时间槽 t 期间，发起多条服务竞价请求的云用户可被视作一个多用户子集合。

表 4-1. 主要符号及其定义

符号	定义
τ, t	时间槽的时间长度、索引值
$u_i, \mathcal{U}, N$	参与云资源拍卖过程的的用户索引值、有限集合、数量
$\mathcal{U}(t), N(t)$	在时间槽 t 期间发出服务竞价请求的用户集合、数量，其中 $\mathcal{U}(t) \subset \mathcal{U}, M(t) \le N$
r, $\tilde{c}$	每台云物理机器的资源容量、资源运营成本（即每时间槽 t 的平均能耗成本）
$m(t)$	在时间槽 t 期间处于开启状态的云物理机器数量
$p^e(t)$	在时间槽 t 期间的即时电力价格
$p(t)$	在时间槽 t 期间的即时云资源单位价格
$\beta_i(t)$	在时间槽 t 期间由用户 u_i 所提出的服务竞价请求，其中 $\beta_i(t) = \left(s_i, b_i, \chi_i, t_i^-, t_i^+\right)$
s_i	服务竞价请求 $\beta_i(t)$ 的服务类型
b_i	服务竞价请求 $\beta_i(t)$ 的服务竞价预算
χ_i	服务竞价请求 $\beta_i(t)$ 的云托管任务大小
$\left[t_i^-, t_i^+\right]$	服务竞价请求 $\beta_i(t)$ 的 QoS 需求范围
$\left[a_i^-, a_i^+\right]$	服务竞价请求 $\beta_i(t)$ 的云资源需求量范围
$a_i(t)$	在时间槽 t 期间用户 u_i 所购买的云资源单位量
$A(t)$	在时间槽 t 期间的多用户云资源分配方案，其中 $A(t) \leftarrow a_i(t)_{u_i \in \mathcal{U}(t)}$
$\rho_i\left(a_i(t)\right)$	当用户 u_i 被分配到单位量为 $a_i(t)$ 的云资源时，所获得的 QoS 增益率
$g_i\left(a_i(t)\right)$	当用户 u_i 被分配到单位量为 $a_i(t)$ 的云资源时，所获得的“幸福”程度
$v_i\left(p(t), a_i(t)\right)$	用户 u_i 的效用函数
$d_i\left(p_i(t)\right)\right)$	用户 u_i 的云资源需求量（由用户效用最大化问题决定）
γ	云服务供应商所要求的最低服务利润率，即“利润/成本”
$\pi(t)$	在时间槽 t 期间由云服务供应商所赚取的服务收益

在每个时间槽 t 期间，各个云用户 $u_i \in \mathcal{U}(t)$ 向云服务供应商提交服务竞价请求 $\beta_i(t) = \left(s_i, b_i, \chi_i, t_i^-, t_i^+\right)$，进而与云服务供应商协商云资源的定价和分配结果。服务竞价请求 $\beta_i(t) = \left(s_i, b_i, \chi_i, t_i^-, t_i^+\right)$ 揭示了服务竞价信息，包括服务类型 s_i、服务竞价预算 b_i、云托管任务大小 χ_i、QoS 需求范围 $\left[t_i^-, t_i^+\right]$。具体地，服务类型 s_i 指明了云用户 u_i 托管在云端的应用类型；服务竞价预算 b_i 说明了云用户 u_i 在时间槽 t

期间可承担的最大服务付费支出；云托管任务大小 χ_i 估计了云用户 u_i 在云上托管的每个计算任务大小；云用户 u_i 的 QoS 需求是弹性灵活的，以服务响应时间的弹性需求范围（即 $\left[t_i^-, t_i^+\right]$）表示。

在每个时间槽 t 内，各个云用户 u_i 被分配单位量为 $a_i(t)$ 的云资源。被分配的云资源可以是多维度的，其表征不同类型的云资源（如 CPU 单元，运行内存，磁盘存储，网络带宽等）。启发于 Hadoop/Spark 框架所采用的并行度（Degree of Parallelism）概念 [73]，云资源单位量 $a_i(t)$ 可以对应于被分配的计算时隙（Compute Slot）数量，而一个计算时隙指定了一份固定配比的多维云资源。

云服务供应商：与相关文献 [74, 75] 一样，本章将研究关注点放置于资源容量同构的云物理机群上。在云服务供应商的管理运营下，每台物理机器配备有单位量为 r 的计算资源。此外，由于云服务供应商通常提供了足够的计算资源，并且最近的研究文献[76, 77]也表明大型云数据中心的资源利用率往往低于 50%，因此假设云服务供应商被配置以无限的云资源容量。为了减少不必要的运营成本和提高资源盈利能力，云服务供应商只会开启有计算负载的物理机器，而将空闲的物理机器切换到休眠状态。在每个时间槽 t，云服务供应商收集来自不同云用户 $u_i \in \mathcal{U}(t)$ 的服务竞价请求，并分别向每个云用户分配单位量为 $a_i(t)$ 的云资源。如文献[78]所述，在时间槽 t 期间所需开启的物理机器数 $m(t)$ 可由式（4-1）估计。

$$m(t) = \left\lceil \frac{\sum_{u_i \in \mathcal{U}(t)} a_i(t)}{r} \right\rceil \qquad \text{式（4-1）}$$

云服务供应商的资源运营成本主要源于云硬件设施所产生的能耗成本[36]。设定每个时间槽 t 期间运行一台物理机器的平均能耗成本是 $\tilde{c}$，则在该时间槽 t 内运行 $m(t)$ 台物理机器所需的总能耗成本为 $m(t)\cdot\tilde{c}$。同时，即时电力价格 $p^e(t)$ 会随时间而波动不定[79]。因此，在时间槽 t 期间，云服务供应商应该按照 $p^e(t)\cdot\ \tilde{c}\cdot m(t)$ 去支付能源账单。

在每个时间槽 t，云服务供应商还会确定即时云资源单位价格 $p(t)$。基于即时云资源价格 $p(t)$，每个云用户 $u_i \in \mathcal{U}(t)$ 会被收取 $p(t)\cdot a_i(t)$ 的服务费用。同时，云服务供应商不仅能从用户处收回能耗成本，而且能够保障最低为 γ 的利润率。因此，具有云服务供应商利润约束，如式（4-2）所示。

$$p(t)\cdot a_i(t) \geq (1+\gamma)\cdot p^e(t)\cdot\ \tilde{c}\cdot m(t) \qquad \text{式（4-2）}$$

4.2.2 用户效用模型

本小节建立用户效用模型，以反映云用户对不同云资源分配结果的偏好程度。对于被分配到 $a_i(t)$ 云资源量的用户 u_i，其用户效用依赖于实际获得的 QoS 水平

以及 相应所需支付的服务费用 $p(t)\cdot a_i(t)$。

设定对于云用户 u_i，其 QoS 增益（即响应时间 $t_i\left(a_i(t)\right)$ 的增益程度）映射了云用户 u_i 对不同云资源分配结果 $a_i(t)$ 的“幸福”程度。当被分配到更多数量的云资源，云用户 u_i 则会拥有更高的 QoS 水平。依据文献 [80] 中给出的实验依据和文献 [81] 所提供的回归分析结果，云用户 u_i 获得的服务响应时间 $t_i\left(a_i(t)\right)$ 可以由式（4-3）估计得到，其中 $\theta_{s_i,0}$、$\theta_{s_i,1}$、$\theta_{s_i,2}$ 和 θ_{3,s_i} 是关于服务类型 s_i 的回归参数。

$$t_i\left(a_i(t)\right)=\theta_{s_i,0}+\theta_{s_i,1}\cdot\frac{\chi_i}{a_i(t)}+\theta_{s_i,2}\cdot a_i(t)+\theta_{3,s_i}\cdot\log\left(a_i(t)\right) \qquad \text{式（4-3）}$$

在式（4-3）中，$\theta_{s_i,0}$ 表示反映串行计算部分的固定时间成本；$\theta_{s_i,1}\cdot\chi_i/a_i(t)$ 表示在云上托管的计算任务大小对响应时间的影响，即计算任务越大，服务响应时间越长；$\theta_{s_i,2}\cdot a_i(t)$ 表示“多对一（All-to-One）”通信成本 [82]，在多个计算时隙（即 $a_i(t)$）间进行调度/序列化操作会产生一定的时间成本；$\theta_{3,s_i}\cdot\log\left(a_i(t)\right)$ 则反映了跨计算时隙的某些通信模型（如聚合树模型 [83]）中产生的通信成本。

如 4.2.1 小节所述，云用户 u_i 提出了一种弹性灵活的 QoS 需求域 $\left[t_i^-,t_i^+\right]$。根据式（4-3），可以计算出对应于 QoS 需求域 $\left[t_i^-,t_i^+\right]$ 的云资源需求范围 $\left[a_i^-,a_i^+\right]$，如式（4-4）所示。分配给云用户 u_i 单位量为 $a_i(t)>a_i^+$ 的云资源就超出了用户服务需求范围，是无意义的；也就是说，云用户 u_i 会获得比 t_i^- 更短的服务响应时间。然而，分配给云用户 u_i 单位量为 $a_i(t)<a_i^-$ 的云资源则会使云用户 u_i 的服务响应时间长于 t_i^+，没有满足到用户 u_i 的最低服务需求（即 t_i^+ 或者 a_i^-）。

$$a_i^-=t_i^{-1}\left(t_i^+\right),\quad a_i^+=t_i^{-1}\left(t_i^-\right) \qquad \text{式（4-4）}$$

设定用 $g_i\left(a_i(t)\right)$ 表示云用户 u_i 从云资源分配结果 $a_i(t)$ 中获得的“幸福”程度，由式（4-5）计算得到。当 $a_i(t)\in\left[a_i^-,a_i^+\right]$ 时，采用 QoS 增益率 $\rho_i\left(a_i(t)\right)$ 来评估 $g_i\left(a_i(t)\right)$。QoS 增益率 $\rho_i\left(a_i(t)\right)$ 被定义为服务响应时间 $t_i\left(a_i(t)\right)$ 相对于 t_i^+ 的比率定义，如式（4-6）所示。注意，QoS 增益率 $\rho_i\left(a_i(t)\right)$ 不与云资源分配量 $a_i(t)$ 成比例关系，而是具有较强的非凸性 [80]。

$$g_i\left(a_i(t)\right)=\begin{cases}\rho_i\left(a_i^+\right) & \text{if } a_i(t)\in\left(a_i^+,+\infty\right)\\ \rho_i\left(a_i(t)\right) & \text{if } a_i(t)\in\left[a_i^-,a_i^+\right]\\ 0 & \text{if } a_i(t)\in\left[0,a_i^-\right)\end{cases} \qquad \text{式（4-5）}$$

$$\rho_i\left(a_i(t)\right)=\frac{t_i^+}{t_i\left(a_i(t)\right)} \qquad \text{式（4-6）}$$

当云用户 u_i 获得 QoS 增益率 $\rho_i\left(a_i(t)\right)$ 时，云用户 u_i 同时也必须向云服务供应商支付服务费用 $p(t)\cdot a_i(t)$。不同于 QoS 增益率 $\rho_i\left(a_i(t)\right)$，服务费用 $p(t)\cdot a_i(t)$ 对用户效应。通过综合 QoS 增益率 $\rho_i\left(a_i(t)\right)$ 和服务费用 $p(t)\cdot a_i(t)$，用户效用函数 $v_i\left(p(t),a_i(t)\right)$ 被表示为式（4-7）。由于服务费 $p(t)\cdot a_i(t)$ 不能超出服务竞价预算 b_i，所以在式（4-7）中服务费用 $p(t)\cdot a_i(t)$ 由 b_i 作归一化处理。

$$v_i\left(p(t),a_i(t)\right)=\begin{cases}\rho_i\left(a_i^+\right)-\dfrac{p(t)\cdot a_i(t)}{b_i} & \text{if } a_i(t)\in\left(a_i^+,+\infty\right)\\ \rho_i\left(a_i(t)\right)-\dfrac{p(t)\cdot a_i(t)}{b_i} & \text{if } a_i(t)\in\left[a_i^-,a_i^+\right]\\ -\dfrac{p(t)\cdot a_i(t)}{b_i} & \text{if } a_i(t)\in\left[0,a_i^-\right)\end{cases} \qquad \text{式（4-7）}$$

4.3 基于用户个体理性的云资源购买策略

基于用户效用模型，本小节将重点研究在不同的即时云资源价格 $p(t)$ 下每个云用户 $u_i\in\mathcal{U}(t)$ 的云资源需求量。当即时云资源单位价格被云服务供应商确定为 $p(t)$ 时，每个云用户 $u_i\in\mathcal{U}(t)$ 通过求解自身用户效用最大化问题，从而基于个体理性得到自身云资源需求量 $d_i(p(t))$。换言之，

$$d_i(p(t))\triangleq\arg\max_{a_i(t)} v_i\left(p(t),a_i(t)\right) \qquad \text{式（4-8）}$$

然而，如式（4-7）所示，用户效用函数 $v_i(\cdot)$ 的非连续性和非凸性阻碍了云资源需求量 $d_i(p(t))$ 的高效求解。鉴于此，具体地分为两种情况以详细讨论。

- **情况 1：**云用户 u_i 的服务竞价预算 b_i 不足以承担其购买单位量为 $a_i(t)>a_i^-$ 的云资源。在这种情况下，云用户 u_i 的 QoS 需求 $\left[t_i^-,t_i^+\right]$ 不能得到满足。若云用户 u_i 决定购买云资源，则反而会产生负用户效用。因此，基于个体理性，云用户 u_i 会放弃购买云资源（即 $d_i(p(t))=0$）。
- **情况 2：**云用户 u_i 有充足的服务竞价预算 b_i 来购买单位量为 $a_i(t)>a_i^-$ 的云资源。在这种情况下，基于个体理性，云用户 u_i 不会购买单位量为 $a_i(t)>a_i^+$ 的云资源，因为其 QoS 需求 $\left[t_i^-,t_i^+\right]$ 会被过度满足，徒增额外的服务费用。因此，最大化用户效用的云资源需求 $d_i(p(t))$ 应该在 $a_i(t)\in\left[a_i^-,a_i^+\right]$ 范围内搜索。

Algorithm 1: 云资源需求分配算法(RARD)

Input: 即时云资源单位价格$p(t)$; 在时间槽t期间的多用户服务竞价请求$\mathcal{U}(t)$.

Output: 多用户云资源分配方案$A(t)$.

for 每个$u_i \in \mathcal{U}(t)$ **do**
  if $p(t) \cdot a_i^- > b_i$ **then**
    $a_i(t) \leftarrow 0$; ▷ 情况1
  else
    通过求解优化问题（即式（4-9））的KKT方程式，得到最大化云用户u_i效用$v_i(a_i(t))$的云资源需求分配结果$a_i(t) = d_i(p(t))$; ▷ 情况2
return $A(t) \leftarrow \langle a_i(t) \rangle_{i=1}^{N(t)}$;

本小节的后续分析将讨论如何求解情况 2 中的云资源需求量$d_i(p(t))$。根据对情况 2 的上述分析，给出由式（4-9）定义的用户效用最大化问题，即在$\left[a_i^-, a_i^+\right]$范围内求解最大化用户效用的云资源分配量$a_i(t)$。约束式（4-10）体现了预算可行性要求 [84]，表明服务费用$p(t)\cdot a_i(t)$不能超出服务竞价预算b_i。

$$\max_{a_i(t)} \ \rho_i\left(a_i(t)\right) - \frac{p(t)\cdot a_i(t)}{b_i} \qquad \text{式 （4-9）}$$

$$\text{s.t. } p(t)\cdot a_i(t) \le b_i \qquad \text{式（4-10）}$$

$$a_i^- \le a_i(t) \le a_i^+ \qquad \text{式（4-11）}$$

根据凸优化的定义 [85]，由式（4-9）描述的用户效用最大化问题属于凸优化问题。鉴于此，可以求解凸优化问题对应的 KKT 条件方程式（Karush-Kuhn-Tucker Conditions）[85] 来搜索出用户效用最大化问题（如式（4-9））的最优解。求解 KKT 方程式的计算复杂度伴随着凸优化问题中不等式约束的数目呈指数增长 [86]；在优化问题（如式（4-9））中，共有 K=3 个不等式约束（即式（4-10）的上界、式（4-11）的上界和下界），所以求解用户效用最大化问题（如式（4-9））的计算复杂度是$\mathcal{O}\left(2^K\right)=\mathcal{O}(1)$。

综上所述，总结概括情况 1 和情况 2，并给出基于用户合理的云资源需求分配算法（RARD），如算法 1 所示。每位用户$u_i \in \mathcal{U}(t)$根据自身云资源需求$d_i(p(t))$购买单位量为$a_i(t)$的云资源。当即时云资源单位价格被确定为$p(t)$时，依次确定每位云用户$u_i \in \mathcal{U}(t)$的云资源需求量；这时，串行计算复杂度为$\mathcal{O}(N(t))$。此外，因为每位云用户u_i的用户效用最大化问题（如式（4-9））是相互独立的，所以每位用户$u_i \in \mathcal{U}(t)$的资源需求量也可以并行求解；这时，并行计算复杂度低至$\mathcal{O}(1)$。

4.4 最大化用户激励的云资源拍卖机制 PIRA

4.4.1 优化问题描述

本小节将从云服务供应商的角度出发，并兼顾考虑云服务供应商和云用户的利益，提出最大化用户激励的优化问题。也就是说，以保证最低为γ的利润率的为前提，激励最大数量的用户群选择在云端托管应用程序。

如 4.3.1 小节所述，每个云用户的资源需求量$d_i(p(t))$被定义。根据$d_i(p(t))$的定义，最大化用户激励的优化问题被进一步给出。即时云资源单位价格$p(t)$和云资源分配方案$A(t)=\langle a_i(t)\rangle_{u_i\in\mathcal{U}(t)}$在每个时间槽$t$被动态地求解；其中，云资源分配方案$A(t)$是价格敏感的，即在某一特定云资源价格$p(t)$下基于用户需求而求解得到。

$$\max_{a_i(t),p(t)} \sum_{u_i\in\mathcal{U}(t)} I_{\{a_i(t)>0\}} \qquad \text{式（4-12）}$$

s.t. 式（4-2）

$$a_i(t)=d_i(p(t)) \quad \forall u_i\in\mathcal{U}(t) \qquad \text{式（4-13）}$$

最大化用户激励的优化问题是以式（4-12）为优化目标方程，并受式（4-2）、式（4-13）的约束。式（4-12）旨在最大化选择云端服务的用户数量；其中，$I_{\{a_i(t)>0\}}$是指示函数，当$a_i>0$函数返回 1 否则返回 0。式（4-2）保障了云服务供应商的最低利润率要求γ。式（4-13）表示每个云用户$u_i\in\mathcal{U}(t)$在自愿的基础上购买云资源；云资源购买量$a_i(t)$等于云用户u_i的资源需求量$d_i(p(t))$。

值得注意的是，高效求解由式（4-12）定义的优化问题并非易事，具体体现在以下三个方面：

- 由式（4-12）定义的优化问题属于装箱问题，是 NP 难解的。云端部署的物理机器可视作箱子，而云用户所需求的云资源可被看作要装入箱子的物品。由于装箱问题是 NP 难解的，所以由式（4-12）定义的优化问题也是 NP 难解的。
- 另一个问题求解难点来自于用户效用函数$v_i(\cdot)$的非连续性和非凸性。在由式（4-12）定义的优化问题中，每个云用户u_i的资源购买量$a_i(t)$是基于其自身用户效用最大化策略来确定的。这会使得优化问题的解决变得更加复杂。
- 最后一个问题求解难点体现在决策变量$p(t)$和$A(t)$之间的复杂关联关系。对云

资源价格$p(t)$的决策进一步影响了云资源分配方案$A(t)=\langle a_i(t)\rangle_{u_i\in\mathcal{U}(t)}$。与此同时，多用户云资源分配方案$A(t)$又决定了优化结果$\sum_{u_i\in\mathcal{U}(t)} I_{\{a_i(t)>0\}}$，并与云服

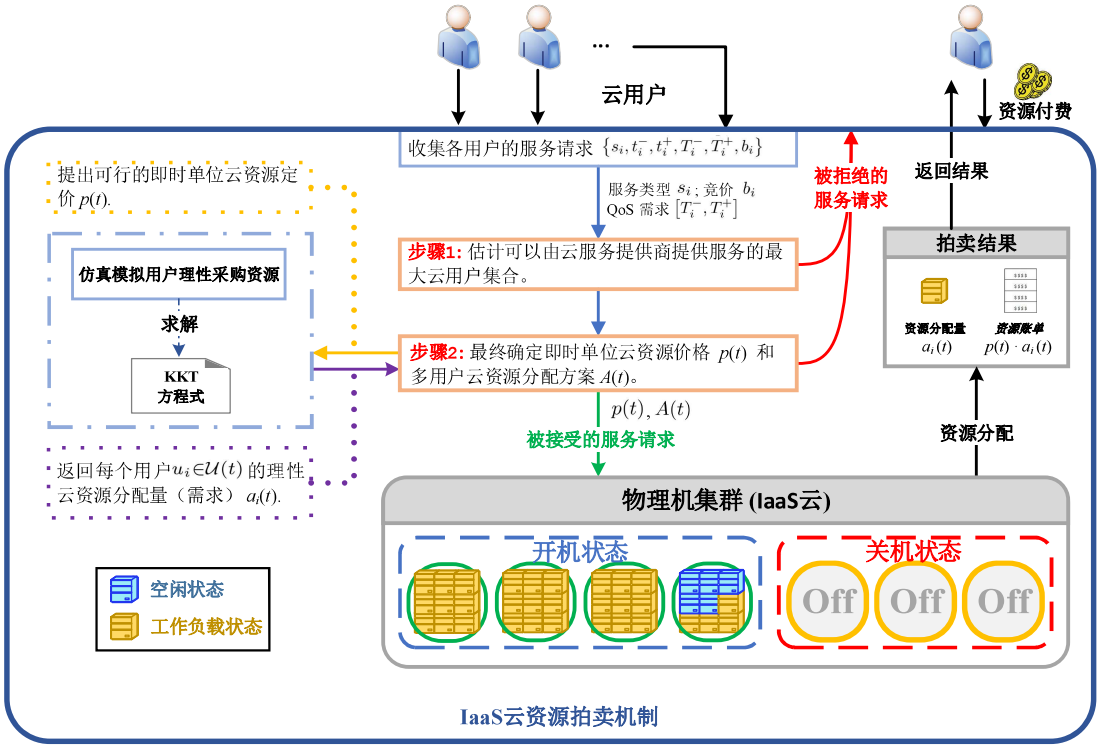


图 4-2. 最大化用户激励的 PIRA 拍卖机制

务供应商所要求的最低利润率 γ 是否被满足密切相关。鉴于上述三个问题求解难点，亟需设计出一种高效的资源拍卖机制。

4.4.2 PIRA 拍卖机制

- **PIRA 拍卖机制概述**

PIRA 云资源拍卖机制旨在最大化用户激励（即激励出最多的云用户），同时为云服务供应商保证最低的利润率 γ。图 4-2 概述了 PIRA 云资源拍卖机制的流程框架。PIRA 云资源拍卖机制是基于价格激励的。云服务供应商对云资源进行定价；而云用户具有价格敏感的云资源需求量，即基于不同的云定价结果，每个云用户根据即时资源需求量来购买云资源，使自身用户效用最大化。上述基于用户个体理性的云资源购买策略已经在 4.3 小节中被详细给出。简而言之，PIRA 云资源拍卖机制动态地感知云市场环境（包括云用户的资源需求水平，云资源的运营成本等），然后通过制定出合适的云资源价格来调节云用户的资源需求总量，以实现用户激励最大化。PIRA 云资源拍卖算法由两个步骤组成，以便对云资源定价和分配结果进行即时决策，具体如下：

- **步骤 1：** 以较粗略而高效的方式，粗估最大云用户集合 $\hat{\mathcal{U}}(t)$

  ☞ 云服务供应商应该为最多的云用户满足其 QoS 需求 $\left[t_i^-, t_i^+\right]$，而不打破价格下限 $\lfloor p \rfloor$（稍后详述）。以此为基础，估计出最大云用户集合 $\hat{\mathcal{U}}(t)$。

- **步骤 2：**最终确定即时云资源单位价格 $p(t)$ 和云资源分配方案 $A(t)$
  - ☞ 根据在步骤 1 中得到的粗估结果 $\widehat{\mathcal{U}}(t)$，云服务供应商进一步最终确定最大云用户集合 $\widetilde{\mathcal{U}}(t)$。当云用户 $u_i \in \widetilde{\mathcal{U}}(t)$ 被云服务供应商服务时，最低利润率 γ 能够得到切实的保障。
  - ☞ 同时，即时云资源单位价格 $p(t)$ 被优化求解，使得云服务供应商尽可能赚取更多的服务收益。基于云资源单位价格 $p(t)$，通过模拟每个云用户 $u_i \in \widetilde{\mathcal{U}}(t)$ 的理性云资源购买过程，得到相应的云资源分配方案 $A(t)=$ $=\langle a_i(t)\rangle_{u_i\in\widetilde{\mathcal{U}}(t)}$。

在本小节余下部分里，将详细介绍步骤 1、步骤 2 的云资源拍卖工作流程。

- **步骤 1 - 粗估最大云用户集合 $\widehat{\mathcal{U}}(t)$**

在每个时间槽 t 期间，云服务供应商收集所有云用户 $u_i \in \mathcal{U}(t)$ 的服务竞价请求。每个云用户 $u_i \in \mathcal{U}(t)$ 拥有异构的服务竞价预算 b_i 和云资源需求 $\left[a_i^-, a_i^+\right]$。基于 b_i 和 $\left[a_i^-, a_i^+\right]$，得到云用户 u_i 最高可接受的云资源单位价格 p_i^+，可由式（4-14）计算得出。如果云资源单位价格 $p(t)$ 高于 p_i^+，则云用户 u_i 的服务竞价预算 b_i 将不足以购买最低单位量为 a_i^- 的云资源，从而其 QoS 需求不能得到满足。根据基于用户个体理性的云资源购买策略（详见 4.3 小节），云用户 u_i 这时将会放弃购买云资源（即 $a_i(t)=0$）。

$$p_i^+ = \frac{b_i}{a_i^-}, \quad \text{对于云用户 } u_i \in \mathcal{U}(t) \qquad \text{式（4-14）}$$

基于云用户最高可接受的云资源单位价格 p_i^+，以递增的排序顺序对云用户 $u_i \in \mathcal{U}(t)$ 进行排序。对云用户 $u_i \in \mathcal{U}(t)$ 的排序结果被表示为 $\Theta = \langle u_{[1]}, u_{[2]}, \ldots, u_{[N(t)]} \rangle$，其中 $u_{[i]}$ 是在序列 Θ 中排在第 i 位的云用户。如果 $p(t) \le p_{[i]}^+$，则云用户 $u_{[j]}\ (j \ge i)$ 具有充足的服务竞价预算 $b_{[j]}$ 来购买足够单位量的云资源，使得其 QoS 需求被满足。相反地，如果 $p(t) > p_{[i]}^+$，则云用户 $u_{[j]}\ (j \le i)$ 将不具备充足的服务竞价预算 $b_{[j]}$ 来购买最低单位量为 $a_{[j]}^-$ 的云资源，从而其最低的 QoS 需求 $t_{[j]}^+$ 也不能被满足。这时，云用户 $u_{[j]}\ (j \le i)$ 会放弃购买云资源（即 $a_{[j]}(t)=0$）。

为了估计在云端能被服务的最大云用户集合，须首先考虑到云服务供应商所要求的最小利润率 γ 能否被满足。基于此，提出“云资源基价”$\lfloor p \rfloor$ 的概念，即 $\lfloor p \rfloor = \tilde{c} \cdot p^e(t) \cdot (1+\gamma)/r$。云资源基价 $\lfloor p \rfloor$ 表示平分一台云物理机底价

$\tilde{c}\cdot p^{e}(t)\cdot(1+\gamma)$（即能耗成本×$(1+\gamma)$）的单位云资源价格。每个云用户$u_i\in\mathcal{U}(t)$最高可接受的云资源价格$p_i^{+}$都将与云资源基价$\lfloor p\rfloor$作比较。对于云用户$u_{[i]}$，如果$p_{[i]}^{+}<\lfloor p\rfloor$则云服务供应商将不会接受云用户$u_{[i]}$在时间槽 t 期间的服务竞价请求。在步骤 1 里，需要在序列Θ中剔除掉所有符合$p_{[i]}^{+}<\lfloor p\rfloor$条件的云用户$u_{[i]}$。因此，在序列$\Theta$中关键云用户$u_{[i'']}$需要被找出，其中云用户$u_{[i'']}$应该满足

$$p_{[i'']}^{+}\geq\lfloor p\rfloor \text{ 且 } p_{[j]}^{+}<\lfloor p\rfloor \ \left(j<i''\right) \qquad \text{式（4-15）}$$

换句话说，所有云用户$u_{[j]}\left(j<i''\right)$的服务竞价请求会被拒绝。根据以上分析过程，粗估的最大云用户集合$\widehat{\mathcal{U}}(t)$可以表示为

$$\widehat{\mathcal{U}}(t)=\left\langle u_{[i'']},u_{[i''+1]},\ldots,u_{[N(t)]}\right\rangle \qquad \text{式（4-16）}$$

- **步骤 2 - 最终确定云资源单位价格$p(t)$和云资源分配方案$A(t)$**

以步骤 1 里所粗估的最大云用户集合$\widehat{\mathcal{U}}(t)$为基础，将进一步最终确定最大云用户集合$\widetilde{\mathcal{U}}(t)$；与此同时，即时云资源单位价格$p(t)$、多用户云资源分配方案$A(t)$也会被最终确定。在陈述上述具体算法实现过程之前，云资源价格范围$[p(t)^{-},\ p(t)^{+}]$的概念被定义。不同的云资源价格范围$[p(t)^{-},\ p(t)^{+}]$将激励规模各异的云用户集合。不妨假定，$[p(t)^{-},\ p(t)^{+}]$是恰好激励出云用户集合$\left\{u_{[i]},u_{[i+1]},\ldots,u_{[N(t)]}\right\}$的云资源价格范围。基于式（4-14），$p_{[i]}^{+}$表示云用户$u_{[i]}$最高可接受的云资源价格。如果云资源单位价格$p(t)\leq p_{[i]}^{+}$，则云用户$u_{[j]}(j=i,i+1,\ldots,N(t))$能够购买足够单位量的云资源，使其自身 QoS 需求被满足。因此，云资源价格范围$[p(t)^{-},\ p(t)^{+}]$由式（4-17）~ 式（4-18）定义。

$$p(t)^{+}=p_{[i]}^{+} \qquad \text{式（4-17）}$$

$$p(t)^{-}=\begin{cases}\dfrac{\tilde{c}\cdot p^{e}(t)\cdot(1+\gamma)}{r} & \text{if } i=1\\ p_{[i-1]}^{+}+\sigma & \text{otherwise}\end{cases} \qquad \text{式（4-18）}$$

其中$\sigma>0$是一个极小正实数，恰好使$p_{[i-1]}^{+}+\sigma>p_{[i-1]}^{+}$成立；同时，当$i=1$时$p(t)^{-}=\lfloor p\rfloor$。值得注意的是，步骤 1 里所定义的云资源基价$\lfloor p\rfloor$只是保障最小利润率$\gamma$的必要不充分条件。不妨考虑此云服务情景：$\sum_{u_i\in\mathcal{U}(t)}a_i(t)/r=50.3$则表明$m(t)=51$。此时，云用户需要额外地分担由 0.7 单位量的空闲云资源所产生的能耗成本和相应的最小利润量（即 0.7 单位量云资源的能耗成本×$(1+\gamma)$）。由此可

见，云资源基价$\lfloor p \rfloor$不足以充分保障云服务供应商的最小利润率γ要求。

因此，步骤 2 有必要在步骤 1 估计结果$\widehat{\mathcal{U}}(t)$的基础上而进一步最终确认得到最大云用户集合$\widetilde{\mathcal{U}}(t)$。简而言之，其基本方法思路是从$\widehat{\mathcal{U}}(t)$为起始点，逐步缩小云用户集合的规模，直到云用户供应商所要求的最小利润率γ被切实保障为止。步骤 2 的具体算法流程如下：

- 按照步骤 1 的估计结果$\widehat{\mathcal{U}}(t)$，将云用户集合暂时设定为$\left\langle u_{[i'']}, u_{[i''+1]}, \ldots, u_{[N(t)]} \right\rangle$；
- 如果在云用户集合$\widehat{\mathcal{U}}(t)$对应的云资源价格范围$[p(t)^-,\ p(t)^+]$之间存在满足最小利润率γ的云资源价格方案，则步骤 1 所估计的最大云用户集合$\widehat{\mathcal{U}}(t)$就是经过最终确认的最大云用户子集$\widetilde{\mathcal{U}}(t)$。
- 否则，拒绝云用户$u_{[i'']}$的服务竞价请求来缩小云用户集合的规模。然后，按照类似上一步的方法，验证在缩小过的云用户集合$\widehat{\mathcal{U}}(t)-\left\{u_{[i'']}\right\}$是否存在云资源价格方案，使得最小利润率$\gamma$被满足。
- 如果存在符合最小利润率γ条件的云资源价格方案，则经过最终确认的最大云用户子集$\widetilde{\mathcal{U}}(t)$便是$\widehat{\mathcal{U}}(t)-\left\{u_{[i'']}\right\}$。否则，将进一步缩小云用户集合的规模（即拒绝云用户$u_{[i''+1]}$的服务竞价请求），然后按照之前的方法再验证在当前云用户规模下是否存在合适的云资源价格方案。
- 循环上述算法过程，直到最大云用户子集$\widetilde{\mathcal{U}}(t)$得到最终确认 / 所有云用户的服务竞价请求都被拒绝为止。

当在云资源价格范围$[p(t)^-,\ p(t)^+]$之间求解满足最小利润率γ的云资源价格方案时，找出实现服务收益最大化的云资源价格方案$p(t)$是符合云服务供应商利益的，有助于让云服务供应商赚取更多利润。云服务供应商的收益π最大化问题由式（4-19）定义，受式（4-2）、式（4-13）、式（4-20）的条件约束。式（4-20）指明了决策变量$p(t)$的可行域，即$p(t)\in[p(t)^-,\ p(t)^+]$。

$$\max_{a_i(t),p(t)} \ \pi(t)=p(t)\cdot\sum_{u_i\in\mathcal{U}(t)} a_i(t) \qquad \text{式（4-19）}$$

$$s.t.\ \ p(t)^- \le p(t) \le p(t)^+ \qquad \text{式（4-20）}$$

式（4-2）、式（4-13）

鉴于式（4-2）和式（4-13），用户效用函数$v_i(\cdot)$的非凸性和非连续性使服务收益最大化问题的优化过程变得复杂难解。然而，通过详细研究由式（4-19）定义优化问题的特征和结构，本章提出一种简单高效的最大化服务收益云资源定价算法 RORP。该算法可求解得到一个服务收益最大化的近似解，近似比为$(1+\epsilon)$。

Algorithm 2: 服务收益最大化的云资源定价算法(RORP)

Input: 云资源价格范围$[p(t)^-, p(t)^+]$.
Output: 即时云资源单位价格$p(t)$;
多用户云资源分配方案$A(t)$.

1 初始化$\pi \leftarrow 0$, $p(t) \leftarrow \texttt{null}$, $A(t) \leftarrow \texttt{null}$;
2 按照式（4-21），计算候选云资源价格数量X;
3 将连续的云资源价格范围$[p(t)^-, p(t)^+]$ 离散采样为X个候选价格，其中$\hat{p}_x \leftarrow p(t)^- \cdot (1+\epsilon)^{x-1}, x = 1, 2, ..., X$;
4 **for** 每个$x = \{1, 2, ..., X\}$ **do**
5 $\quad A(t) \leftarrow \texttt{RARD}(\hat{p}_x, \mathcal{U}(t))$;
6 $\quad$ **if** $\frac{\hat{p}_x \cdot \sum_{u_j \in \mathcal{U}(t)} a_i(t)}{(1+\gamma)} \geq \widetilde{c} \cdot p^e(t) \cdot \left\lceil \frac{\sum_{u_j \in \mathcal{U}(t)} a_i(t)}{r} \right\rceil$ **and** $\hat{p}_x \cdot \sum_{u_i \in \mathcal{U}(t)} a_i(t) > \pi$ **then**
7 $\quad\quad \pi \leftarrow \hat{p}_x \cdot \sum_{u_i \in \mathcal{U}(t)} a_i(t)$;
8 $\quad\quad p(t) \leftarrow \hat{p}_x$;
9 $\quad\quad A(t) \leftarrow \langle a_i(t) \rangle_{u_i \in \mathcal{U}(t)}$;
10 **return** $p(t)$ and $A(t)$;

表 4-2. 近似比 v.s. RORP 算法复杂度

近似优化比$1+\epsilon$	1.02	1.03	1.04	1.05	1.10	1.20
候选价格数量X	117	48	59	48	25	13

* 以 $p(t)^+ / p(t)^- = 10$ 为例

RORP 算法伪码如算法 2 所示，主要由以下两步组成：

1. 将云资源价格范围$\left[p(t)^-, p(t)^+\right]$离散采样为 X 个候选价格 $\hat{p}_x$， $x = 1, 2, \ldots, X$（第 1-3 行）；
2. 在 X 个候选价格之间，找出云资源价格的近似最优解 $\hat{p}(t) = \hat{p}_x$（$x \in \{1, 2, \ldots, X\}$）。当即时云资源价格 $p(t)$ 被定为 $\hat{p}(t)$ 时，云服务供应商可赚取$(1+\epsilon)$-近似比的最高服务收益（第 4-10 行）。

具体地讲，云资源价格范围$\left[p(t)^-, p(t)^+\right]$被离散采样为 X 个候选价格，其中 X 可基于式（4-21）计算得到。

$$X = \left\lfloor \frac{\log\left(p(t)^+ / p(t)^-\right)}{\log(1+\epsilon)} \right\rfloor + 1 \qquad \text{式（4-21）}$$

设定 $\hat{p}_x$ 表示第 x 个候选价格，可由式（4-22）计算得到。值得注意的是，$\epsilon > 0$ 是用于折衷 RORP 算法效率和近似优化比的常量参数。较小的 ϵ 值则表明 RORP 算法具有更好的近似优化比，但同时也需要对云资源价格范围$\left[p(t)^-, p(t)^+\right]$采用

更高的离散采样密度（即更大数值的 X），从而导致更高的算法复杂度（将在定理 1 种稍后详述）。

$$\hat{p}_x = p(t)^+ \cdot (1+\epsilon)^{1-x} \qquad \text{式（4-22）}$$

如算法 2 所示，RORP 算法将分别计算在每个候选价格 $\hat{p}_x$ 下的服务收益 π。然后，赚取最大服务收益的云资源定价候选方案 $\hat{p}(t) = \hat{p}_x$ 被最终选为即时云资源单位价格 $p(t)$。与此同时，在云资源定价结果 $p(t)$ 之下，分别模拟每个云用户的理性资源购买过程，从而得到多用户云资源分配方案 $A(t) = \langle a_i(t) \rangle_{u_i \in \mathcal{U}(t)}$。

定理 1： RORP 算法提供了 $(1+\epsilon)$-近似的服务收益最大化。当 $N(t)$ 名云用户并发地操作云资源购买过程时（详见 RARD 算法），RORP 算法的并行计算复杂度为 $\mathcal{O}(X)$。

证明： 不妨假设，实现服务收益最大化的云资源单位价格是 $p^*(t)$，而云服务供应商因此赚取最高的服务收益 $\pi^*(t)$。此时，必定存在正整数 $y \in [1, X]$ 使得 $p(t)^+ \cdot (1+\epsilon)^{-y} \le p^*(t) \le p(t)^+ \cdot (1+\epsilon)^{1-y}$ 成立。根据 RORP 算法，$\hat{p}(t)$ 从 X 个候选价格中被选定为即时云资源单位价格 $p(t)$；在 $\hat{p}(t)$ 之下，云服务供应商能够获得服务收益 $\hat{\pi}(t)$。鉴于上述设定和分析，可以推导出

$$\begin{aligned}
\pi^*(t) &= p^*(t) \times \sum_{u_i \in \mathcal{U}(t)} d_i\left(p^*(t)\right) \\
&\le \left(p(t)^+ \cdot (1+\epsilon)^{1-y}\right) \times \sum_{u_i \in \mathcal{U}(t)} d_i\left(p^*(t)\right) \\
&\le (1+\epsilon) \times \left(p(t)^+ \cdot (1+\epsilon)^{-y}\right) \times \sum_{u_i \in \mathcal{U}(t)} d_i\left(p(t)^+ \cdot (1+\epsilon)^{-y}\right) \qquad \text{式（4-23）} \\
&\le (1+\epsilon) \times \hat{\pi}(t)
\end{aligned}$$

其中式（4-23）是基于云用户资源需求随云资源价格上升而降低的事实而推导得出的。根据上述公式推导，可以证明得出 RORP 算法的优化近似比为 $(1+\epsilon)$。

在 RORP 算法里，通过模拟每个云用户的理性资源购买过程，分别计算出在 X 个候选价格下的服务收益。在某一候选云资源价格下，通过并发地模拟 $N(t)$ 名云用户的理性资源购买过程，以确定在该云资源定价方案下的服务收益；这需要 $\mathcal{O}(1)$ 的计算复杂度。总而言之，RORP 算法的计算复杂度为 $\mathcal{O}(X)$。 □

RORP 算法将非凸连续优化问题转化为离散优化过程，从而设计出一个计算高效的、$(1+\epsilon)$-近似优化算法。正如定理 1 所证，常量参数 ϵ 被用于在计算复杂度、近似比之间做出权衡；具体以表 4-2 为例证（即 $p(t)^+ / p(t)^- = 10$）。借助于 RORP 算法，时间槽 t 期间的云资源单位价格 $p(t)$ 和云资源分配方案 $A(t)$ 在步骤 2 里被求解得出。

Algorithm 3: 价格激励的云资源拍卖算法(PIRA)

Input: 在时间槽t期间的多用户服务竞价请求$\mathcal{U}(t)$.
Output: 即时云资源单位价格$p(t)$;
多用户云资源分配方案$A(t)$.

// 步骤 1
for 每个$i=\{1,2,...,N(t)\}$ **do**
  找出云用户序列Θ 中排在第i位的云用户$u_{[i]}$;
  if $p^{+}_{[i]} \geq \widetilde{c}\cdot p^{e}(t)\cdot(1+\gamma)/r$ **then**
    $i'' \leftarrow i$;
    break;
  else
    拒绝云用户$u_{[i]}$ 的服务竞价请求;
// 步骤 2
for 每个$i=\{i'',i''+1,...,N(t)\}$ **do**
  $p(t)^{+} \leftarrow p^{+}_{[i]}$;
  if $i=i''$ **then**
    $p(t)^{-} \leftarrow \widetilde{c}\cdot p^{e}(t)\cdot(1+\gamma)/r$;
  else if $i \geq i''+1$ **then**
    $p(t)^{-} \leftarrow p^{+}_{[i-1]}$;
  $p(t), A(t) \leftarrow \texttt{RORP}(p(t)^{-}, p(t)^{+})$;
  if $p(t)$ =null **and** $A(t)$ =null **then**
    拒绝云用户$u_{[i]}$ 的服务竞价请求;
  else
    接受云用户$\{u_{[j]}\}_{j=i}^{N(t)}$ 的服务竞价请求;
    return $p(t)$ and $A(t)$;
return null;

综上所述，本章最大化用户激励的云资源拍卖机制 PIRA 被分为**步骤 1** 和**步骤 2**。在算法 3 中，由步骤 1、步骤 2 组成的 PIRA 拍卖机制流程被概括给出。PIRA 算法在每个时间槽 t 期间被执行。

PIRA 算法复杂度分析：在步骤 1 里，假定 $N_1(t)$ 位云用户经过粗估过程被排除以提供云服务，其中 $N_1(t)\leq N(t)$。在判定是否拒绝云用户 u_i 的服务竞价请求时，需要验证云用户 u_i 对应的 $p^{+}_{[i]}$ 是否不低于$\lfloor p \rfloor$。因此，步骤 1 所需的计算复杂度是$O\left(N_1(t)\right)$。在步骤 2 里，不妨假设需要遍历 $\mathcal{X}$ 个候选云资源价格下的服务收益，从而确定即时云资源单位价格 $p(t)$。在每个候选云资源价格之下，并行地操作每个云用户 u_i 的资源购买过程；这需要$\mathcal{O}(1)$的并行计算复杂度。因此，步骤 2 的计算复杂度是$\mathcal{O}(\mathcal{X})$。综上所述，PIRA 算法的总计算复杂度是$\mathcal{O}\left(N_1(t)+\mathcal{X}\right)$。

4.4.3 重要性质分析

本章的云资源拍卖机制 PIRA 将满足如下三个重要性质，具体如下：

- **预算可行性（Budget Fesibility）：**规定拍卖者（云用户）的服务付费不应该超出服务竞价预算 [87]，这通常被视为拍卖机制的基本要求。
- **激励相容性（Incentive Compatibility）：**禁止拍卖者（云用户）的机会主义行为 [88]。无论其他竞拍者（云用户）采用何种竞价策略，拍卖者（云用户）都会上报真实的服务竞价预算，没有动机谎报。
- **无妒性（Envy-freeness）：**保障多用户拍卖过程的公平性。每个竞拍者（云用户）都可以得到最大化其用户效用的云资源分配量 [89]。这样的话，每个竞拍者（云用户）不会嫉妒其他竞拍者（云用户）的云资源分配结果，体现了多用户云资源分配方案的公平性。

定理 2（预算可行性）：云用户 u_i 的服务竞价预算 b_i 始终足以覆盖购买云资源 $a_i(t)=d_i(p(t))$ 所需的服务费用。

证明：如 4.3.1 小节所述，基于用户个体理性的云资源购买策略提供了预算可行性保障。具体被划分为两个方面讨论：

- 一方面，只有云用户 u_i 的服务竞价预算 b_i 能够至少满足其最低服务需求（即 t_i^+ 或者 a_i^-），云服务供应商才会接受云用户 u_i 的服务竞价请求并满足云用户 u_i 的 QoS 需求。云用户 u_i 被分配到单位量为 $a_i(t)$ 的云资源，受约束式（4-10）的限制以保障预算可行性。
- 另一方面，当云用户 u_i 的服务竞价预算 b_i 不足以满足其最低服务需求（即 t_i^+ 或者 a_i^-）时，云用户会放弃购买云资源（即 $d_i(p(t))=0$），即不会产生云服务费用。显而易见地，这时满足了预算可行性。

总而言之，云用户 u_i 的服务竞价预算 b_i 始终足以覆盖购买云资源 $a_i(t)=d_i(p(t))$ 所需的服务费用，即预算可行性被满足。 □

定理 3（激励相容性）：云用户 u_i 不会向云服务供应商谎报服务竞价预算 b_i。

证明：为了证明激励兼容性，就需要说明云用户 u_i 不能通过谎报自己的服务竞价预算 b_i 来提高自己用户效用值。基于该证明思路，将详细分析所有可能的情况。

- **情况 1：**当云用户 u_i 上报真实的服务竞价预算 b_i 时，其服务竞价请求被拒绝。
  - ☞ 一方面，如果云用户 u_i 谎报其服务竞价预算 $b_i' < b_i$，则显然其服务竞价请求依旧会被拒绝。与上报真实服务竞价预算相比，谎报 $b_i' < b_i$ 的云用户 u_i 所获得的用户效用值没有改变（依旧 $v_i\left(p(t),0\right)=0$）。
  - ☞ 另一方面，云用户 u_i 试图夸大自己的投标预算以分配到云资源。当上报真

实的服务竞价预算b_i时，云用户u_i被拒绝，用户效用为0。这是因为云用户u_i的服务竞价预算b_i不足以购买单位量为a_i^-的云资源，以满足其最低的QoS需求t_i^+。不妨假设，云用户u_i谎报其服务竞价预算$b_i' > b_i$。然而，即使云服务供应商会因此分配给云用户u_i单位量为$a_i(t) \geq a_i^-$的云资源，但是云用户u_i最终也不能用其实际的服务竞价预算$b_i < b_i'$来支付单位量为$a_i(t) \geq a_i^-$的云资源服务费用。因此，对于云用户u_i来说，夸大其服务竞价预算是无用的。

- **情况 2**：当云用户u_i上报真实的服务竞价预算b_i时，其服务竞价请求被接受。

  ☞ 根据式（4-8），云用户u_i购买单位量为$a_i(t) = d_i(p(t))$的云资源，使其自身用户效用最大化。如用户效用函数$v_i(\cdot)$表达式（4-7）所示，实现云用户u_i效用最大化的最优云资源分配量$a_i(t)$与其服务竞价预算b_i相关。无论云用户u_i低估还是夸大了其服务竞价预算，其最优云资源分配量$a_i(t)$的值都是不同的。因此，依据谎报的服务竞价预算（即$b_i' < b_i$或$b_i' > b_i$）得到的最优云资源分配量$a_i(t)$并不能使实际上的用户效用最大化。因此，云用户u_i没有动机去谎报其服务竞价预算（即$b_i' < b_i$或$b_i' > b_i$），而是如实上报她的服务竞价预算b_i。 □

定理 4（无妒性）：云用户u_i更偏爱分配给自己的云资源结果$a_i(t) = d_i(p(t))$，而不是嫉妒其他云用户的云资源分配量。

证明：如 4.3.1 小节所述，基于用户个体理性的云资源购买策略提供了符合无妒性的多用户云资源分配。根据式（4-8），每个云用户u_i都会购买单位量为$a_i(t) = d_i(p(t))$的云资源以最大化自身用户效用。由于每个云用户u_i都获得了自身最大用户效用$v_i(p(t), a_i(t))$，这体现了云用户的个体理性，更进一步地，每个云用户u_i不会嫉妒其他云用户的资源分配结果，这意味着多用户云资源分配方案符合无妒性。 □

4.5 实验验证

4.5.1 实验设置

在本小节里，将执行仿真实验以验证云资源拍卖机制 PIRA 的有效性。具体而言，通过仿真构建开放的 IaaS 云资源环境，多个云用户同时向云服务供应商提出服务竞价请求，从而在云端执行他们的应用程序。发起服务竞价请求的云

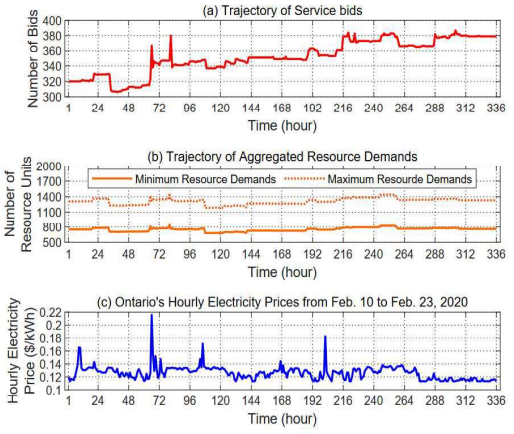


图 4-3. 数据集概览

表 4-3. 实验参数设置

符号	值
τ	1 小时 [76]
$\tilde{c}$	600 瓦特 [77]
γ	0.2
ϵ	0.02
r	10
b_i	$((a_i^- + a_i^+)/2)\cdot\mathcal{N}(0.02, 0.015^2)$ 美元

用户都指定了其服务类型s_i；在实验设置中，每个服务竞价请求的服务类型s_i是从八种典型数据分析应用中随机抽取的，包括 Classification、朴素贝叶斯、 数据回归、K 均值聚类、Spearman 关联分析、主成分分析、交替最小二乘法、概括性数据统计。关于上述八种数据分析应用在式（4-3）中的回归参数$\theta_{s_i,0}$、$\theta_{s_i,1}$、$\theta_{s_i,2}$和θ_{3,s_i}已经在文献 [81] 中被明确给出。

在基于实际数据的仿真实验中，为期 14 天（即 2016 年 11 月 16 日至 2017 年 2 月 16 日）的 Microsoft Azure 集群虚拟机服务轨迹被摘取使用 [90] 。具体而言，每条 Azure VM 订阅请求在仿真实验中被视作一条服务竞价请求β_i；其中，最低云资源需求量a_i^-是基于该条 Azure VM 订阅请求所发起的 vCPU 核心需求量而设置的，而最高云资源需求量a_i^+则根据$1.75\times a_i^-$而设置的。为了简化实验设置，从整个 Azure 集群数据集中随机选择 100 位 Azure VM 订阅用户。图 4-3(a) 展示了 100 位订阅用户在 14 天内所发起的服务竞价请求规模，其中每小时有 300 到 400 条服务竞价请求被同时提；图 4-3(b) 则给出了 100 位订阅用户总云资源需求量的时间波动轨迹。再者，即时电力价格$p^e(t)$是从加拿大安大略省电力系统运营商 IESO 处而收集得到的。从 2020 年 2 月 10 日至 2 月 23 日的即时电力价格轨迹如图 4-3(c)所示。

此外，本小节的关键实验参数数值在表 4-3 中被列出。鉴于 Microsoft Azure 所采用的最小付费时间单元 [76] ，时间槽长度τ被设置为 1 小时。至于服务竞价预算b_i，则采纳文献 [22] 所做出的假设，即根据正态分布$((a_i^- + a_i^+)/2)\cdot\mathcal{N}(0.02, 0.015^2)$（单位：美元）为每条云服务竞价请求$\beta_i$生成预算$b_i$（除非另有规定）。本小节实验均在 Windows 10 计算机上执行，实验用处理器为 Intel Core i7-5500U（双核 CPU，2.4 GHz），运行内存为 12.0 GB。

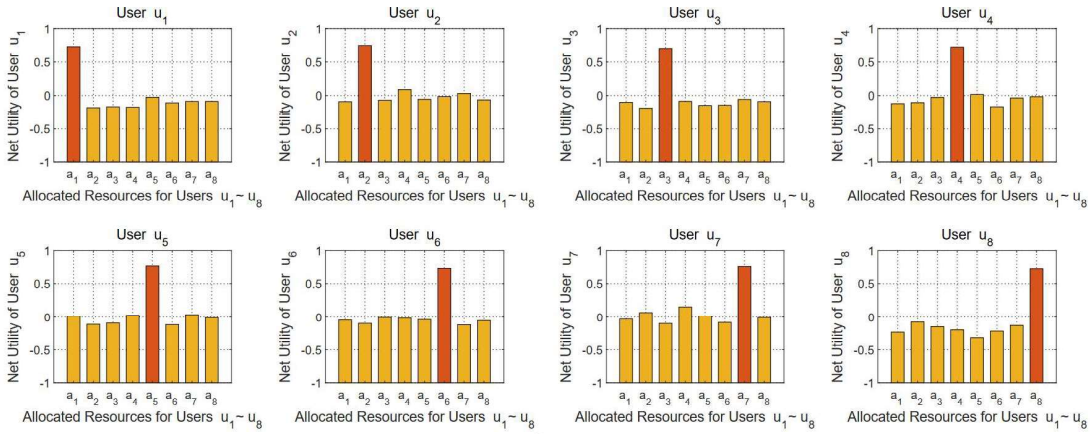


图 4-4. 云用户 $u_1 \sim u_8$ 对不同云资源分配结果 $a_1 \sim a_8$ 的净效用比较

4.5.2 数值实验

为了更清楚地展示数值实验结果，本小节仅在单一时间槽内执行云资源拍卖 PIRA 算法。数值实验结果是经过多轮重复算法执行之后取平均值而得到的，从而充分反映云资源拍卖的随机性。

1）重要性质验证

通过对数值实验结果的分析，实验验证了云资源拍卖机制 PIRA 的无妒性和预算可行性。在随后基于实际数据的仿真实验里，满足无妒性/预算可行性的实验结果也能够类似得出。在本部分，每项数值实验结果都是经过 200 次 PIRA 算法重复执行后取平均值而得到的。

- **无妒性（Envy-Freeness）**：为了更好地验证云资源拍卖机制 PIRA 的无妒性，实验场景被简化为云用户 $u_1 \sim u_8$ 同时向云服务供应商发起服务竞价请求。通过执行 PIRA 算法，每个云用户 u_i 会被分配到单位量为 a_i 的云资源，并基于其云资源分配量 a_i 而获得用户效用 $v_i(a_i)$。如图 4-4 所示，将云用户 u_i 由其自身云资源分配量 a_i 产生的用户效用 $v_i(a_i)$ 与根据其他云用户资源分配量 a_j（$j \neq i$）而产生的用户效用 $v_i(a_j)$ 进行比较。由图 4-4 的实验结果可知，对于云用户 $u_1 \sim u_8$ 而言，她对自身云资源分配量 a_i 而产生的用户效用 $v_i(a_i)$ 总是最高的。该实验结果体现了云资源拍卖机制 PIRA 的无妒性。

- **预算可行性（Budget Fesibility）**：在单一时间槽内设置有 100 位云用户同时向云服务供应商发起服务竞价请求，以实验验证云资源拍卖机制 PIRA 的预算可行性。图 4-5 在 PIRA 算法向每位云用户 u_i 确定的服务费用以及其服务竞价预算 b_i 之间作比较。依据图 4-5 所展示的实验结果，100 位云用户的服务竞价预算 b_i 都充分覆盖了各自的服务费用；平均而言，每位云用户 64.71%的服务竞价预算

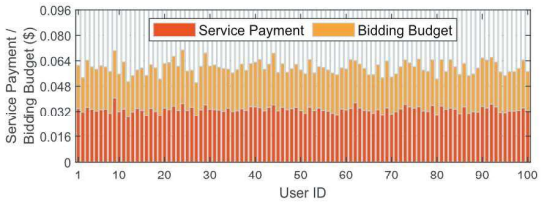


图 4-5. 服务费用 v.s. 服务竞价预算

图 4-6. 在不同最低利润率 γ 要求下的云资源拍卖结果

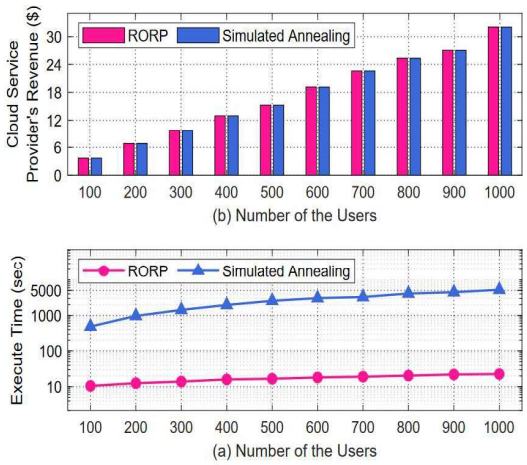


图 4-7. RORP 算法 v.s.模拟退火算法[58]

被用于支付云服务。这充分验证云资源拍卖机制 PIRA 的预算可行性。

2）最低利润率要求 γ 的影响

本部分将研究最低利润率要求 γ 对云资源拍卖结果的影响（即被接受的云服务竞价请求数量、云服务供应商的服务收益）。在此处，实验场景被设置为 1000 位云用户同时向云服务供应商同时发起服务竞价请求。图 4-6 展示了在不同最低利润率 γ 要求下的云资源拍卖结果，其中每项实验结果是经过 200 次 PIRA 算法重复执行后取平均值而得到的。当最低利润率 γ 被设为较高的值时，较少数量的服务竞价请求被云服务供应商接受；这是因为更多具有有限预算的服务竞价请求被拒绝，被拒绝的服务竞价请求无法满足云服务供应商所设定的较高利润目标 γ。然而，由于设置较高 γ 值的云服务供应商倾向于将云资源售卖给具有较高预算的竞拍者（即云用户），所以云服务供应商的服务收益会随着 γ 值的升高而增加。综合上述分析可得出，云服务供应商所制定的最低利润率 γ 起到了权衡云用户、云服务供应商间利益的作用。云资源拍卖 PIRA 机制在满足最低利润率 γ 的前提下，最大化云用户激励（激励出最多的云用户）。

3）RORP 近似算法性能评价

RORP 算法在最大化用户激励的云资源拍卖机制 PIRA 中起着十分重要的作用。当最大的云用户集合被最终确定之后，RORP 算法则负责求解出实现近似最大化服务收益的云资源定价方案 $p(t)$。鉴于 RORP 算法的重要性，本部分将实验评估其算法有效性和执行效率，其中模拟退火算法（Simulated Annealing Algorithm）[91] 被用作基准优化方法。作为先进的非线性优化方法，模拟退火算法能够充分逼近最优云资源单位价格，从而赚得近似最大话的服务收益。

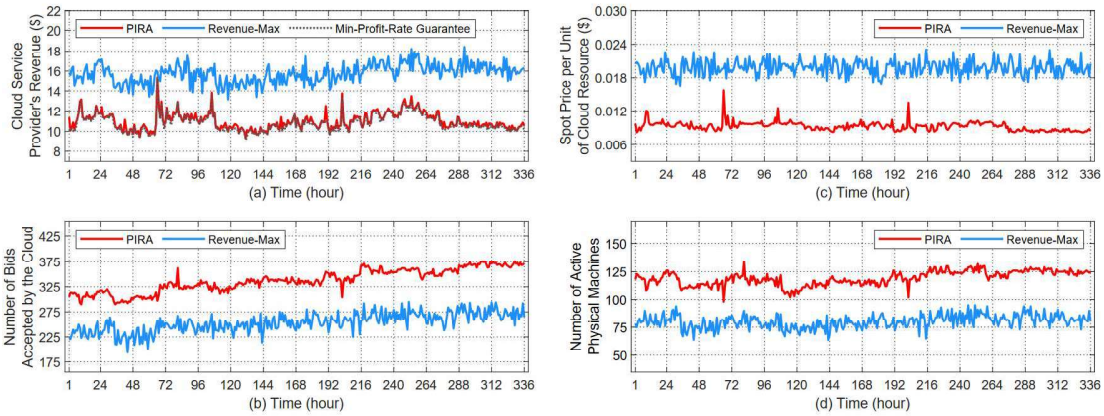


图 4-8. 基于实际轨迹数据的对比仿真结果

图 4-7 展示了 RORP 算法和模拟退火算法之间的对比结果。其中，关于 RORP 算法的实验结果是经过 200 次算法重复执行后取平均值而得到的，而关于模拟退火算法的实验结果是经历 10 次算法重复执行而取平均值得到的（由于模拟退火算法的执行时间过长）。在算法有效性方面，对比了这两个算法所实现的云服务供应商收益结果。在不同的竞拍者（云用户）规模下，RORP 算法和模拟退火算法之间的服务收益差异极其微小。然而，在算法执行效率方面，RORP 算法和模拟退火算法执行时间的差距极大，两者之间存在着显著数量差异。在图 4-7(b)中，算法执行时间被测量以表征其算法执行成本；在确定的云资源定价方案下，每名云用户的资源需求量被依次（串行）地求解得到。RORP 算法的执行时间远远小于模拟退火算法，但 RORP 算法所获得的服务收益与模拟退火算法相差不大。该实验对比结果表明了 RORP 算法的先进性。更进一步地，当每名云用户的资源需求量被并行地求解时，预计 RORP 算法将具有更少的算法执行时间。

4.5.3 基于实际轨迹数据的仿真实验

基于实际轨迹数据，本小节将进一步深入研究云资源拍卖机制 PIRA 的有效性。具体而言，将分别从云服务供应商、云用户的角度验证 PIRA 算法的性能。一方面，对于云服务供应商，将验证 PIRA 算法能否动态保证最低利润率要求 γ。另一方面，对于云用户，将通过检验被云服务供应商接受的服务竞价请求数量来评估 PIRA 算法所赋予的云服务承载能力。除此之外，被称为 Revenue-Max 的基准算法将与本章所提出的 PIRA 算法作比较。Revenue-Max 算法是一种单边主义算法，完全不考虑云用户利益（效用）而执行云服务供应商的收益最大化操作。其中，Revenue-Max 算法采用了与 RORP 算法类似的近似优化技术，从而执行服务收益最大化操作。

图 4-8(a) 展示了云服务供应商在 336 个时间槽内借助 PIRA 算法所获得的服务收益，并与保证最低利润率 γ 的收益基线进行了比较。最低利润率 γ 的值是由云服务供应商根据其盈利目标而预先配置的。在图 4-8(a)中，保证最低利润率 γ

的收益基线随着即时电力价格的变化而往复波动；而云服务供应商借助 PIRA 算法所赚取的服务收益始终高于保证最低利润率γ的收益基线，这符合对本章云资源拍卖机制 PIRA 的预期（即至少保证服务利润率γ）。此外，在 336 个时间槽内，PIRA 算法所实现的服务收益仅略高于最低利润率γ基线，使得最低利润率γ被恰好满足。按照 PIRA 算法的预期设定，只要最低利润率要求γ能被满足，云服务供应商则倾向于发布具有市场竞争力的较低云资源价格；通过放弃与 Revnue-Max 算法相差的部分服务收益，尽可能地刺激出最多数量的云用户（即云用户激励最大化）。

图 4-8(b) 分别展示了云服务供应商在 PIRA 和 Revenue-Max 算法下接受的服务竞价请求数量。被接受的服务竞价请求数量代表被服务的云用户数。不同于本章所提出的 PIRA 算法，Revnue-Max 基准算法只是单方面地最大化云服务供应商的服务收益。因此，由图 4-8(b)可以看出，PIRA 算法比 Revenue-Max 算法能够容纳更多的云用户/服务竞价请求；这有利于云服务供应商提升自身的云计算市场份额。总而言之，此实验结果与对云资源拍卖机制 PIRA 的设想保持一致，即最大限度地提高云用户激励（激励出最多的云用户）。

此外，图 4-8(c)和图 4-8(d)对 PIRA 算法、Revenue-Max 算法进行更详细的比较分析。Revenue-Max 算法忽视了云用户效用，而仅着眼于最大化云服务收益。因此，如图 4-8(c) 所示，Revenue-Max 算法在各时间槽所确定的云资源价格$p(t)$远高于 PIRA 算法。但反过来说，由于 PIRA 算法所制定的云资源价格$p(t)$更低、更具备市场竞争力，所以本章所提出云资源拍卖机制 PIRA 能吸引更多的云用户。正因如此，PIRA 算法需要开启更多的活跃云物理机器，用于为云用户提供服务，如图 4-8(d) 所示。

4.6 本章小结

本章从面向市场的角度，研究云资源定价与售卖机制。具体而言，对面向多用户的云资源拍卖市场进行建模，其中每位云用户可以跨时间槽向云服务供应商提出服务竞价请求。给定单位云资源的即时定价方案，每个云用户根据自己的资源需求量而购买云资源，使得自身用户效用最大化。为了实现云用户激励的最大化，本章为云服务供应商设计出一种价格激励的云资源拍卖机制 PIRA，以对云资源定价与分配进行高效的即时决策。除此之外，本章所设计的 PIRA 机制还提供了预算可行性、激励相容性、无妒性保障。综上所述，本章的研究工作有望为云计算市场环境下的云资源定价与分配问题提供一个高质量的技术解决方案。

第五章 QoE 感知的分布式边缘任务调度和资源管理

5.1 本章引论

随着近年来物联网（IoT）技术的日益普及，各种各样的物联网设备（包括移动电话、可穿戴设备、传感器等）已经融入到人类日常生活的方方面面。物联网设备数量的快速增长促进了相关软件服务的发展，特别是需要与远端服务器交互的在线服务[92, 93]。在远端服务器的算力加持下，对物联网设备本身的硬件性能要求大大降低；计算处理能力相对受限的物联网设备也可以较好地完成具有高复杂度的计算任务。尽管如此, 与远端服务器交互所引起的网络延迟是一个影响 QoS 的重要威胁和瓶颈。如果总服务延迟（包括计算延迟、网络延迟）超过了 100 毫秒，则物联网用户会明显感知到服务响应延迟[94]，对于延迟敏感的物联网服务甚至可能导致负面后果[95]。然而，由于相当一部分云用户的总服务延迟超过了 100 毫秒 [96, 97]，所以传统的云计算架构难以满足物联网服务的低延迟需求。这时，亟需提出一种新兴的计算架构来支持物联网的在线低延迟服务场景。

边缘计算是为物联网服务环境而设计的新兴计算架构 [98, 99] 。借助于被广泛地理分布的边缘服务器，物联网设备被边缘服务器的强大算力加持，其计算处理能力得到增强 [100]。边缘服务器被广泛地在不同地理位置上部署，以靠近物联网终端设备；而服务供应商将在线服务配置在边缘服务器上，以便与物联网设备实时交互。边缘服务器比远端云服务器更靠近物联网设备，所以与云计算架构相比，物联网设备和服务器之间的网络延迟大幅降低。鉴于网络延迟的下降，总服务延迟（包括计算延迟、网络延迟）也进一步降低，从而物联网用户在边缘计算架构下得到更高的 QoS 水平。

边缘资源管理是在边缘计算领域里的重要研究议题 [101, 102] ，需要充分考虑跨边缘服务器的任务请求调度和多任务资源分配。一方面，相邻边缘服务器的基站信号覆盖区域通常会有部分重叠以避免未覆盖的通信区域 [103] 。因此，任务请求调度策略需要被精心地设计；特别地，处于信号重叠区域的用户存在多个可访问的边缘服务器以供选择。另一方面，与云数据中心的远端服务器相比，边缘服务器通常具有相对受限的计算资源。因此，需要构建更复杂细化的边缘资源分配策略。综上所述，在任务请求调度和边缘资源分配策略的共同有效作用下，物联网用户所享有的 QoS 水平会得到较好提升。

体验质量（QoE）是一项评价用户满意度的重要指标。通常地，QoE 指标用于评估终端用户在体验软件服务时的满意度 [104] ，这尤其对于在线交互式网络服务（如大型多人在线游戏 [105] ）具有重要指标性意义。尽管已经有许多边缘计算领域的研究文献，并涉及到服务延迟最小化 [106] 、能源效率优化 [107] 、服务成

本优化[44, 108]等研究话题，但是基于QoE优化的边缘资源分配问题仍旧是有待深入研究的科学问题。因此，在边缘计算环境下，对QoE优化方法的研究将是本章的研究重点。

本章将研究以最大化系统QoE水平为优化目标的多用户边缘资源分配（ERA）问题。多个物联网用户同时发起边缘服务请求，并且相互争夺有限的边缘服务资源。ERA问题被描述为装箱问题，以决定每个边缘服务请求被调度到的边缘服务器、被分配到的边缘服务资源量。由于装箱问题通常是NP难解的[109]，并且QoE指标携有内在的非线性特征，因此很难以集中决策的中心化方式高效地求解ERA问题。

为了应对上述研究挑战，本章将采用一种基于博弈论的方法，被称作ERAGame。具体而言，ERAGame以去中心化的方式来寻找ERA问题的最优解。在博弈模型中，每个物联网用户都被作为策略参与者，以最大化自身QoE水平为目标，对自己的边缘资源分配方案做出独立的决策。通过允许每个物联网用户独立地决定自己的边缘资源分配方案，ERAGame可以大幅减轻由中心化决策带来的复杂度，并且借助抢占式QoE改进机制（Preemption-Based QoE Improvement，PRIM）来协调多用户实现系统QoE水平最大化的共同优化目标。此外，ERAGame允许每个物联网用户具有异构QoE函数。最后，得益于ERAGame的去中心化特性，设计分布式边缘资源分配算法（QoE-DEER）以求出ERAGame的纳什均衡解（NE）。分布式QoE-DEER算法求出的NE解被证明是ERA问题的最优解，即得到了最大化系统QoE水平的边缘资源分配方案。

本章余下部分的内容安排如下：在5.2小节中，描述ERA优化问题，其中具体进行系统建模，量化分析QoE指标与QoS指标间的关联性。在5.3小节中，形式化描述ERA博弈模型（ERAGame），并且提出了协调多用户博弈过程的抢占式PRIM机制。然后，在5.4小节中，设计出分布式边缘资源分配算法（QoE-DEER）；在技术上，给出了实现QoE-DEER算法的合作消息传递机制。在4.5小节中，运行基于真实边缘计算数据集的仿真实验，以验证分布式QoE-DEER算法的性能和效率。最后，在5.6小节中，总结本章的整体研究工作。

5.2 问题描述

5.2.1 系统模型

系统概述：如图5-1所示，设定由N个物联网终端用户、M台边缘服务器组成的边缘计算系统。由N个物联网终端用户组成的有限集合表示为$\mathcal{U}=\{u_1,\ldots,u_N\}$，而由$M$台边缘服务器组成的有限集合则表示为$\mathcal{E}=\{e_1,\ldots,e_M\}$。携有物联网设备的多位终端用户同时向边缘服务器端发起服务请求，以期被其

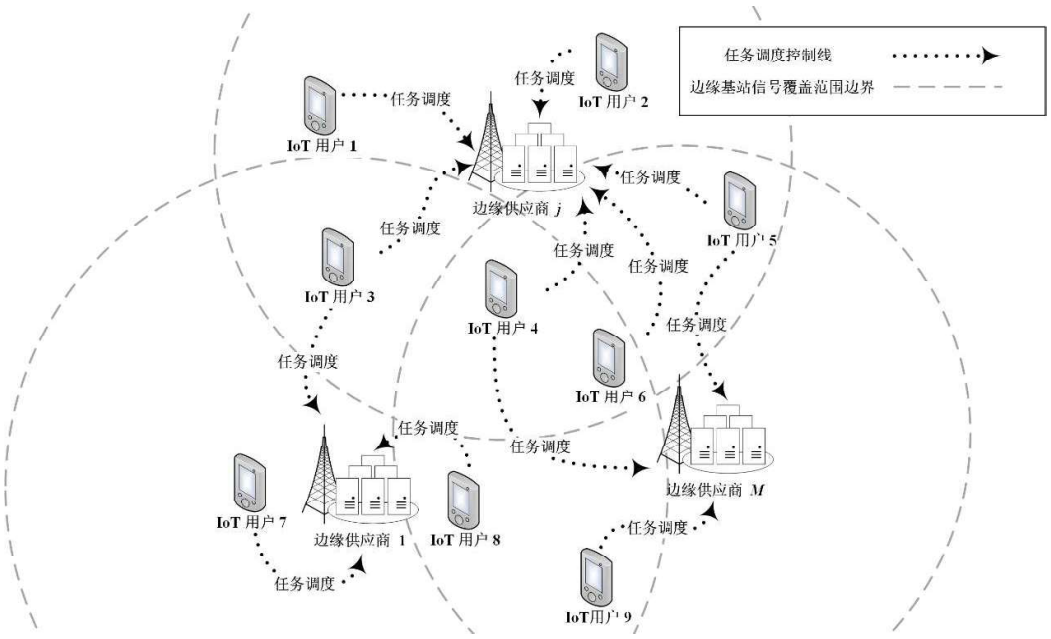


图 5-1. 边缘计算架构下的物联网任务调度和资源管理

附近的边缘服务器处理。通常地，边缘服务器是根据物联网用户的地理分布规律而跨地域部署的；并且，每台边缘服务器具有一定的通信信号覆盖区域，从而覆盖一组物联网用户/设备。在本章设定的系统场景里，每个用户请求在物联网设备上被本地处理，或者被其附近的边缘服务器作远端处理。

物联网服务请求：不失一般性地，假设每位物联网用户 u_i 只会提出单一的服务请求，而提出多个服务请求的物联网用户则可视作一组提出单一服务请求的用户。设定需要 h_i 个 CPU 周期去处理物联网用户 u_i 的服务请求；与此同时，如果物联网用户 u_i 的服务请求被调度到边缘服务器端处理，则从物联网设备 u_i 到边缘服务器端需传送数据量为 η_i 的网络数据包。鉴于边缘服务器的有限信号覆盖范围，用 $\mathcal{E}_i$ 表示物联网用户 u_i 可访问的边缘服务器集合，而用 $\mathcal{U}_j$ 表示边缘服务器 e_j 所覆盖的物联网用户集合。

物联网设备/边缘服务器：设定每台物联网设备被配置有限数量的计算资源，其本地计算处理能力（即 CPU 处理器频率）为 f_j^L。至于每台边缘服务器 e_j，则被配备有 c_j 个 CPU 处理单元；而 f_j^E 表示边缘服务器 e_j 每个 CPU 处理单元的计算处理能力（即 CPU 处理器频率）。在未来研究工作中，更复杂的多维度边缘资源管理方法将进一步被探究。

边缘网络：借助通信带宽为 B 的无线通信网络，物联网设备与边缘服务器端执行通信和交互操作。因此，使用香农公式（Shannon's Formula）来评估物联网设备与边缘服务器之间的无线数据传输率 $r_{i,j}$。同时，物联网设备/用户 u_i 和边缘

表 5-1. 主要符号及其定义

符号	定义
$\mathcal{U}$	M 位物联网用户/设备的有限集合，即 $\mathcal{U}=\{u_1,\ldots,u_i,\ldots,u_N\}$
$\mathcal{E}$	N 台边缘服务器的有限集合，即 $\mathcal{E}=\{e_1,\ldots,e_j,\ldots,e_M\}$
$\mathcal{E}_i$	物联网用户 u_i 可访问的边缘服务器集合
$\mathcal{U}_j^E$	边缘服务器 e_j 所覆盖的物联网用户集合
h_i	处理物联网用户 u_i 服务请求所需的 CPU 周期数
η_i	从物联网设备 u_i 到边缘服务器端所需传送的网络数据包大小
B	无线通信网络带宽
c_j	边缘服务器 e_j 所配置的 CPU 处理单元数
$d_{i,j}$	物联网设备 u_i 和边缘服务器 e_j 间的无线通信距离
$r_{i,j}$	物联网设备 u_i 与边缘服务器 e_j 之间的无线数据传输率
f_i^L	物联网设备 u_i 的本地计算处理能力（即 CPU 处理器频率）
f_j^E	边缘服务器 e_j 每个 CPU 处理单元的计算处理能力
x_i	将物联网用户 u_i 服务请求调度到的边缘服务器索引值
a_i	为物联网用户 u_i 服务请求分配的边缘服务器资源量
s_i	物联网用户 u_i 的 ERAGame 博弈策略，即 $\mathrm{s}_i=(x_i,a_i)$
S_i	物联网用户 u_i 的博弈策略集合
s	物联网用户 $\mathcal{U}$ 的策略组合，即 $\mathrm{s}=(\mathrm{s}_1,\ldots,\mathrm{s}_N)$
QoE_i^L	当物联网用户 u_i 在本地设备上处理其服务请求时，所获得的自身 QoE 水平值
QoE_i^E	当物联网用户 u_i 在边缘服务器上处理其服务请求时，所获得的自身 QoE 水平值
π_i	物联网用户 u_i 根据博弈策略 s_i 而获得的自身 QoE 水平值

服务器 e_j 间的无线通信距离为 $d_{i,j}$，并且物联网用户 u_i 每单位无线通信距离的接收信噪比为 λ_i^0。综上所述，根据香农公式，用式（5-1）[110] 计算从物联网设备/用户 u_i 到边缘服务器 e_j 的无线数据传输率 $r_{i,j}$。

$$r_{i,j}=B\log_2\left(1+\frac{\lambda_i^0}{d_{i,j}^2}\right) \qquad \text{式（5-1）}$$

注意 $\lambda_i^0=\gamma_0\cdot P_i/\sigma^2$，其中 γ_0 是关于单位无线通信距离的信道功率，P_i 是每个物

联网设备/用户 u_i 的信号传输功率，σ^2 表示无线通信噪声功率。

5.2.2 体验质量（QoE）模型

- **QoS/QoE 指标间的关联性分析**

伴随着物联网终端用户数量的不断增加，每位物联网用户会有异构的 QoS 需求。一方面，来自强时延敏感性用户的服务请求（如在线网络视频流传输服务）通常具有较高的 QoS 需求，因此往往需消耗较多的计算资源来满足该类用户的 QoS 需求。另一方面，来自弱时延敏感性用户的服务请求则仅有较温和的 QoS 需求，从而不需要大量计算资源来使其 QoS 需求被满足。具体以在线视频分析服务 [95] 为例，大多数服务请求只需要不超过一秒的服务时间延迟，而其他服务请求则严格要求 100 毫秒以内的服务时间延迟。因此，本章将采用体验质量（QoE）指标作为反映物联网用户服务满意度的评价标准；每位物联网用户因其异构 QoS 需求而具有不同的 QoE-QoS 关系曲线。

已经存在研究 QoE 指标评估方法的相关文献 [111-113]，并尝试分析 QoS 指标与 QoE 指标之间的量化关联关系。如图 5-2 所示，用户 QoE 水平通常不与其所获得的 QoS 水平（即服务时间延迟）呈现正比关系，而是体现一定程度的非线性关系。换句话说，虽然通过向用户提供较低的服务时间延迟，其 QoE 水平能够获得较大幅度的提升，但是其 QoS 改进幅度往往会从某一收敛临界点（如图 5-2 中 P2 点）逐渐趋弱。经过临界点 P2 之后，即使提供更低的服务时间延迟，也不能显著提升用户的 QoE 水平；这是因为，到达临界点 P2 已经足够邻近最高 QoE 水平（即 100%），进一步提升 QoE 水平的空间十分有限。一般而言，随着服务时间延迟的降低，用户 QoE 水平起初仅有较慢的提升速度，直到图 5-2 中 P1 点；然后，用户 QoE 水平会以相对稳定的速度不断提升，直到收敛临界点 P2；最后，用户 QoE 水平的改进幅度则会逐渐消弱，以无限邻近最高 QoE 水平（即 100%）。

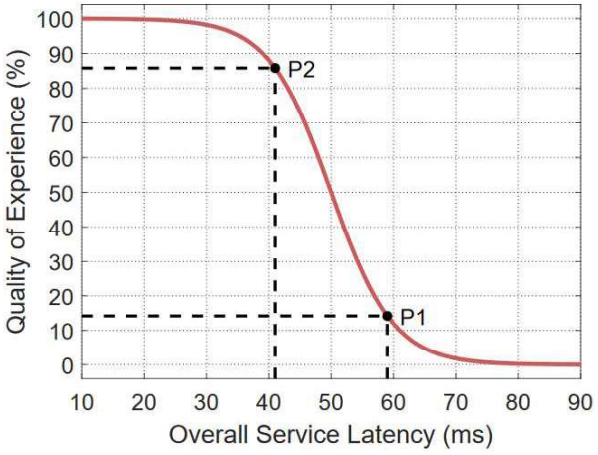


图 5-2. QoS/QoE 指标间的关联性

· 服务质量（QoS）指标

鉴于上述分析所指出的 QoS/QoE 指标间的关联性，本部分首先分析和评估物联网用户的 QoS 指标。因为服务时间延迟通常被视作最典型的 QoS 指标，所以本章将服务时间延迟作为衡量 QoS 指标的量化标准。具体而言，用$\mathbf{s}_i=(x_i,a_i)$表示物联网用户u_i的边缘服务器资源分配策略，其中x_i指将物联网用户u_i的服务请求调度到边缘服务器e_{x_i}，而a_i是为物联网用户u_i分配出的边缘服务器资源量。若物联网用户u_i决定在本地物联网设备上处理服务请求，则相应的边缘服务器资源分配策略(x_i,a_i)被定义为$(0,0)$。

当物联网用户在本地物联网设备上处理服务请求（即$x_i=0$和$a_i=0$）时，则相应的服务时间延迟t_i^L可由式（5-2）计算得到。

$$t_i^L=\frac{h_i}{f_i^L} \qquad \text{式（5-2）}$$

当物联网用户u_i按照策略$\mathbf{s}_i=(x_i,a_i)$将服务请求置于边缘服务器处理时，物联网用户u_i的服务请求在边缘服务器e_{x_i}被处理，并且被分配以单位量为a_i的边缘计算资源。这时，相应的计算延迟$t_{i,x_i}^{E,cmp}(a_i)$可由式（5-3）计算得到，其中边缘服务器e_{x_i}单位量为a_i的计算资源处理能力是$(a_i\cdot f_{x_i}^E)$。注意，单位量为a_i的计算资源处理能力的量化方法遵从对单位资源计算处理能力$f_{x_i}^E$的累加性原则；在技术实现方面，可由轮循 CPU 调度技术（Round Robin CPU Scheduling）来实现单位资源计算处理能力的累加。

$$t_{i,x_i}^{E,cmp}(a_i)=\frac{h_i}{a_i\cdot f_{x_i}^E} \qquad \text{式（5-3）}$$

此外，从物联网用户/设备u_i到边缘服务器e_{x_i}的通信延迟$t_{i,x_i}^{E,off}$可由式（5-4）表示。

$$t_{i,x_i}^{E,off}=\frac{\eta_i}{r_{i,x_i}} \qquad \text{式（5-4）}$$

通过综合式（5-3）和式（5-4），物联网用户u_i依据$\mathbf{s}_i=(x_i,a_i)$在边缘服务器e_{x_i}处理服务请求的总服务时间延迟$t_{i,x_i}^E(a_i)$可由式（5-5）计算得到。

$$t_{i,x_i}^E(a_i)=t_{i,x_i}^{E,off}+t_{i,x_i}^{E,cmp}(a_i) \qquad \text{式（5-5）}$$

- **体验质量（QoE）指标**

正如上述分析，QoE 指标与 QoS 指标之间是非线性相关的。特别的是，相关研究文献 [114-116] 应用 Sigmoid 函数来量化 QoE 指标与 QoS 指标之间的关联性。基于此，本章则选用 Sigmoid 函数的扩展式——Logistic 函数来定量描述 QoE/QoS 指标间的关联性。在 Sigmoid 函数的基础上，Logistic 函数进一步增强对 QoE 指标的数学描述能力（包括 QoE 提升率α_i、基本 QoE 需求β_i）。

鉴于 QoE 指标通常以等级评分的方式被评估，本章使用百分比的量化标准来评价物联网用户的 QoE 指标。每位物联网用户u_i最高获得 100 %的 QoE 水平。当物联网用户u_i在本地设备上处理服务请求（即$x_i=0$和$a_i=0$）时，将根据式（5-6）[5] 获得相应的 QoE 水平QoE_i^L。

$$QoE_i^L=\frac{1}{1+e^{\alpha_i\left(t_i^L-\beta_i\right)}} \qquad \text{式（5-6）}$$

其中参数α_i表示物联网用户u_i的 QoE 提升率，而参数β_i表示物联网用户u_i的 QoE 方程中间点。从物理意义上看，参数β_i指明了物联网用户u_i所需要达到的 QoS 水平，从而获得 50 %的 QoE 水平。在实际应用中，QoE 方程中间点可以代表物联网用户u_i的基本 QoE 需求。

同理，当物联网用户u_i按照策略$\mathbf{s}_i=\left(x_i,a_i\right)$将服务请求置于边缘服务器处理时，物联网用户$u_i$可以获得相应的 QoE 指标$QoE_i^E\left(\mathbf{s}_i\right)$，由式（5-7）表示。

$$QoE_i^E\left(\mathbf{s}_i\right)=\frac{1}{1+e^{\alpha_i\left(t_{i,s_i}^E\left(a_i\right)-\beta_i\right)}} \qquad \text{式（5-7）}$$

至于本章 QoE 模型在现实物联网场景中的实用性，它与在线视频流服务[117]和许多其他应用场合非常吻合。在实际应用里，QoE 指标通常使用平均主观意见分（Mean Opinion Score，MOS）来衡量。较高的 MOS 分数表明了优秀的用户体验/满意度，而较低的 MOS 分数则代表了较差的用户体验/满意度。如一项华为公司科技报告 [118] 所述，网络视频流质量与 MOS 分数间的映射关系与本章所采用的 QoE 曲线类型（即式（5-6）和式（5-7））非常契合。

5.2.3 优化问题描述

在本章的优化问题中，优化目标为对多个物联网用户的总 QoE 水平最大化。首先，以 QoE 函数（即式（5-6）和式（5-7））为基础，描述物联网用户效用$\pi_i\left(s_i\right)$。

一方面，当物联网用户u_i在本地设备上处理服务请求（即$x_i=0$和$a_i=0$）时，物联网用户u_i则根据式（5-8）而获得用户效用$\pi_i\left(s_i\right)$。此时，用户效用$\pi_i\left(s_i\right)$代表着物联网用户u_i在本地设备上处理服务请求时所获得的 QoE 水平。

$$\pi_i\left(s_i\right)=QoE_i^L \qquad \text{式（5-8）}$$

另一方面，当物联网用户u_i按照策略$\mathbf{s}_i=\left(x_i,a_i\right)$将服务请求置于边缘服务器处理时，物联网用户$u_i$则按照式（5-9）而获得相应的用户效用$\pi_i\left(s_i\right)$。

$$\pi_i\left(s_i\right)=\begin{cases}QoE_i^E & \text{if } QoE_i^E>QoE_i^L\\ QoE_i^L & \text{otherwise}\end{cases} \qquad \text{式（5-9）}$$

式（5-9）体现了物联网用户在本地设备与边缘服务器之间进行选择，从而确定在本地设备/边缘服务器上处理其服务请求。具体分为两种情况：

- 根据策略$\mathbf{s}_i=\left(x_i,a_i\right)$，如果在边缘服务器能获得比本地设备处理服务请求更高的 QoE 水平，那么物联网用户u_i将决定在边缘服务器e_{x_i}处理服务请求，从而获得相应的用户效用$\pi_i\left(s_i\right)=QoE_i^E$（表示在边缘端所获得的 QoE 水平）。
- 否则，物联网用户u_i将在本地设备上处理服务请求（即$x_i=0$和$a_i=0$），从而获得用户效用$\pi_i\left(s_i\right)=QoE_i^L$（表示在本地物联网设备上所获得的QoE水平）。

根据物联网用户效用$\pi_i\left(s_i\right)$的定义，本章的多用户边缘资源分配优化问题（简写 ERA 问题）被给出，其优化目标是最大化系统 QoE 水平。具体而言，以式（5-10）为优化目标方程，并且服从式（5-11）~式（5-12）的条件约束。注意，$I_{\{条件式\}}$是指示函数，当条件式为真时返回 1，否则返回 0。

$$\max_{x_i,a_i}\ \sum_{i\in\mathcal{U}}\pi_i\left(s_i\right) \qquad \text{式（5-10）}$$

$$\sum_{i\in\mathcal{U}_j}a_i\cdot I_{\{x_i=j\}}\le c_j \qquad \forall j\in\mathcal{E} \qquad \text{式（5-11）}$$

$$x_i\in\{0\}\cup\mathcal{E}_i,\ \ a_i\ge 0 \qquad \forall i\in\mathcal{U} \qquad \text{式（5-12）}$$

式（5-11）是边缘服务器的资源约束式，用于确保每台边缘服务器e_j不能向物联网用户分配出高于其资源容量c_j的边缘计算资源。式（5-12）用于约束物联网用户的服务请求；每位物联网用户u_i的服务请求要么被调度到一台边缘服务器$e_j\in\mathcal{E}_i$来进一步处理，要么在本地物联网设备上作计算处理。

显而易见地，ERA 优化问题属于装箱问题。每台边缘服务器可被视作资源容量有限的箱子，而 ERA 问题的求解目标是确定多用户的边缘资源分配方案，使得系统 QoE 水平最大化。如装箱问题一样，以集中决策的中心化方式求解 ERA 问题是 NP 难解的 [109] 。除此之外，由于 QoE 函数（即式（5-6）~式（5-7））是非线性的，所以求解 ERA 优化问题的难度也会随之增加。综上所述，亟需一种高效的求解方法来应对 ERA 问题的上述求解挑战。

5.3 多用户边缘资源博弈（ERAGame）

5.3.1 ERAGame 博弈模型描述

针对 ERA 优化问题的诸多求解挑战，本章引入博弈论来降低集中决策的中心化程度，并允许每位物联网用户拥有决策自主性。以提升自身 QoE 水平为个体优化目标，每位物联网用户 u_i 各自决策其边缘资源分配结果。基于此，ERA 优化问题以去中心化的形式来被高效求解。多用户边缘资源博弈模型（ERAGame）在定义 1 中被给出。

定义 1（ERAGame）：策略博弈 $\mathcal{G}$ 被用于描述 N 位物联网用户之间对有限边缘服务器资源的竞争态势，由三元组 $\left\langle \mathcal{U},(\boldsymbol{S}_i)_{u_i\in\mathcal{U}},(\pi_i)_{u_i\in\mathcal{U}} \right\rangle$ 定义。

- $\mathcal{U}$ 是参与策略博弈 $\mathcal{G}$ 的局中人（Player）集合；具体指边缘计算系统中的 N 位物联网用户。多位局中人（物联网用户）之前相互竞争，以期被分配到更多边缘服务器资源，从而获得更高的 QoE 水平。
- $\boldsymbol{S}_i$ 是物联网用户 u_i 的博弈策略集合；每位物联网用户 u_i 的所有可行策略 $\mathbf{s}_i=(x_i,a_i)$ 组成了其博弈策略集合 $\boldsymbol{S}_i$。物联网用户 u_i 的博弈策略 $\boldsymbol{s}_i\in\boldsymbol{S}_i$ 指定了其服务请求被调度到的边缘服务器 e_{x_i}，以及相应被分配的边缘服务器资源量。
- π_i 是物联网用户 u_i 的效用函数，由式（5-8）或者式（5-9）表示。物联网用户 u_i 效用 π_i 被用于评估博弈策略 $\mathbf{s}_i=(x_i,a_i)$ 所获得的 QoE 水平。 □

在 ERAGame 博弈中，每位物联网用户 u_i 都希望被分配到更多的边缘计算资源，从而获得更高的 QoE 水平。然而，由于有限边缘服务器资源容量的约束，所以每位物联网用户 u_i 必须相互竞争去争夺有限的边缘服务器资源，并确定其博弈策略 $\boldsymbol{s}_i\in\boldsymbol{S}_i$ 以最大化其自身 QoE 水平。每位物联网用户 u_i 所选择的策略 $\boldsymbol{s}_i$ 组成了 ERAGame 的策略组合（Strategy Profile），即 $\mathbf{s}=(\mathbf{s}_1,\ldots,\mathbf{s}_N)$。在博弈期间，假定物联网用户 u_i 起初选定了策略 $\mathbf{s}_i$，但是又发现另一个可获得更高 QoE 水平的可行策略 $\mathbf{s}_i'$。在发现到更优博弈策略 $\mathbf{s}_i'$ 的激励下，物联网用户 u_i 会理所当然地更新其博弈策略 $\mathbf{s}_i'$。

然而，由于边缘服务器的资源容量有限，物联网用户之间可能发生竞争性冲突。每位物联网用户 u_i 都希望垄断性获取到最多数量的边缘服务器资源，从而拥有最高的 QoE 水平；因此，物联网用户之间的边缘资源竞争态势将难以避免。为了缓解物联网用户之间的竞争性冲突，纳什均衡 [119]（NE）的概念被引入，从而便于有效管理物联网用户之间的竞争行为。在定义 2 中，详细给出了纳什均衡的定义。

定义 2（纳什均衡，NE）：ERAGame 博弈 $\mathcal{G}=\left\langle \mathcal{U},(\boldsymbol{S}_i)_{u_i\in\mathcal{U}},(\pi_i)_{u_i\in\mathcal{U}}\right\rangle$ 的纳什均衡是指策略组合**s***，其满足对于每位局中人（物联网用户）$u_i\in\mathcal{U}$，

$$\pi_i\left(\boldsymbol{s}_i^*,\boldsymbol{s}_{-i}^*\right)\geq\pi_i\left(\boldsymbol{s}_i,\boldsymbol{s}_{-i}^*\right),\qquad\forall\boldsymbol{s}_i\in\boldsymbol{S}_i \tag{式（5-13）}$$

其中 $\mathbf{s}_{-i}$ 表示除物联网用户 u_i 以外其他用户所组成的策略组合，即 $\mathbf{s}_{-i}=\mathbf{s}-\{\mathbf{s}_i\}$；而 π_i 从原有 $\pi_i(\mathbf{s}_i)$ 扩展为 $\pi_i(\mathbf{s}_i,\mathbf{s}_{-i})$，被用于描述不同物联网用户间的边缘资源竞争态势。 □

值得注意的是，鉴于有限的边缘服务器资源容量，每位物联网用户 u_i 不能无视其他竞争者的边缘资源使用情况，而随意增加其自身边缘资源分配量。此外，$\pi_{-i}(\mathbf{s}_i,\mathbf{s}_{-i})$ 被定义为由物联网用户（除 u_i 以外）组成的策略组合 $\mathbf{s}_{-i}$ 所获得的 QoE 水平，即 $\pi_{-i}(\mathbf{s}_i,\mathbf{s}_{-i})=\sum_{u_j\in\mathcal{U}-\{u_i\}}\pi_j$。

5.3.2 面向 QoE 改进的多用户博弈协调机制

· 基于抢占的多用户博弈协调机制

考虑到纳什均衡解的非唯一性 [119]，本章提出了一种用于协调多用户博弈策略的抢占式机制，使所获得的纳什均衡解最终收敛到较高的系统 QoE 状态。

具体而言，如果存在空闲的边缘服务器资源，那么物联网用户 u_i 可以决定去占用空闲的边缘计算资源，从而获得更高的 QoE 水平。然而，若物联网用户 u_i 所有可访问的边缘服务器资源都已经被其他物联网用户占用，则就没有多余的边缘计算资源来分配给物联网用户 u_i。这时，需要设计出一种边缘资源抢占机制来决定是否将已被其他物联网用户占用的边缘计算资源让给物联网用户 u_i，其目的是提升多用户的系统 QoE 水平。

不妨假定，物联网用户 u_i 试图从其他物联网用户处抢占边缘服务器资源。具体而言，$\triangle a_i$ 表示物联网用户 u_i 试图抢占的边缘服务器资源量，而 $p(u_i)$ 表示被物联网用户 u_i 抢占的其他物联网用户集合。通过从其他物联网用户 $u_k\in p(u_i)$ 抢占单位量为 $\triangle a_i$ 的边缘服务器资源，物联网用户 u_i 可获得的 QoE 增量为 $\triangle\pi_i$；与此同时，对于被抢占的物联网用户 $u_k\in p(u_i)$，其 QoE 总损失量至少为 $\triangle\pi_{-i}$。根据上述分析与描述，边缘资源抢占机制在定义 3 中被严格地给出。当完成一轮边缘资源强占之后，除 $\{u_i\}\cup p(u_i)$ 以外其他物联网用户的博弈策略没有改变。

定义 3（边缘资源抢占机制）：通过本轮边缘资源抢占，如果系统 QoE 水平得到提升，即

$$\triangle \pi_i > \triangle \pi_{-i} \quad \text{式（5-14）}$$

则物联网用户 u_i 会抢占其他物联网用户 $u_k \in p(u_i)$ 所占用的部分边缘服务器资源，从而提升系统 QoE 水平。 □

式（5-14）是物联网用户 u_i 从其他用户 $u_k \in p(u_i)$ 处抢占边缘服务器资源的触发条件。经过对边缘服务器资源的抢占之后，物联网用户 u_i 所获得的 QoE 增量 $\triangle \pi_i$ 要足以抵消被抢占用户 $u_k \in p(u_i)$ 的 QoE 损失量 $\triangle \pi_{-i}$。此外，在边缘资源抢占期间，因为除 $\{u_i\} \cup p(u_i)$ 以外其他物联网用户的博弈策略保持不变，所以其他物联网用户 $\mathcal{U} - (\{u_i\} \cup p(u_i))$ 所获得的 QoE 不受影响。也就是说，只有物联网用户 $u_k \in p(u_i)$ 的 QoE 水平被降低。综合上述分析可知，经过边缘资源抢占之后，整体上系统 QoE 水平得到提升，从而使 ERAGame 博弈过程收敛于具有较高系统 QoE 水平的纳什均衡状态。

· 抢占式 QoE 改进算法 PRIM

本部分设计出抢占式 QoE 改进算法（PRIM），从而实现前文所定义的边缘资源抢占机制，如算法 1 所示。按照边缘资源抢占机制，PRIM 算法从两方面进行设计。一方面，如果物联网用户 u_i 可采用新策略 $\mathbf{s}_i'$ 来进一步占用当前空闲可用的边缘服务器资源，那么物联网用户 u_i 会实施新策略 $\mathbf{s}_i'$ 以提升其自身 QoE 水平。另一方面，如果对于物联网用户 u_i 而言，当前没有空闲可用的额外边缘资源以供其实施占用更多边缘资源的新策略 $\mathbf{s}_i'$，那么需要计算出可能被抢占用户 $p(u_i) = \{u_k \in \mathcal{U} : x_k = x_i' \text{ and } k \neq i\}$ 的最低 QoE 损失量 $\triangle \pi_{-i}$，并与物联网用户 u_i 因抢占边缘资源而产生的 QoE 增量 $\triangle \pi_i$ 进行比较。据此，判定物联网用户 u_i 是否执行对边缘服务器资源的抢占操作。

值得注意的是，被抢占用户 $p(u_i)$ 的最低 QoE 损失量 $\triangle \pi_{-i}$ 可以在有限可数的迭代轮次内较高效地被计算得出。具体而言，在第一轮迭代里，首先按照 $\hat{\pi}_k$ 对占满用户 u_i 可访问边缘资源的物联网用户 $u_k \in p(u_i)$ 进行非降序排序，其经过排序的用户序列为 P。此处，$\hat{\pi}_k$ 是指当物联网用户 u_k 被抢占单位量为 1 的边缘服务器资源时所导致的自身最低 QoE 损失量，用式（5-15）表示。上述对物联网用户 $u_k \in p(u_i)$ 的排序过程可以借助快速排序算法实现；该快速排序过程的算法复杂度为 $\mathcal{O}(m \log m)$，其中 $m = |p(u_i)|$ 通常远小于系统中物联网用户总数 N。

Algorithm 1: 抢占式QoE改进算法(PRIM)

Input: 物联网用户u_i所提出的改进策略$\mathbf{s}'_i$,
当前策略组合$\mathbf{s}$.
Output: 经过更新后的策略组合$\mathbf{s}'$.

if $\pi_i(\boldsymbol{s}'_i) > \pi_i(\boldsymbol{s}_i)$ **then**
  if $a'_i + \sum_{u_k \in \mathcal{U}: x_k = x'_i, k \neq i} a_k \leq c_{x'_i}$ **then**
    return $\boldsymbol{s}' \leftarrow \{\boldsymbol{s}'_i, \boldsymbol{s}_{-i}\}$;
  else
    $\triangle a_i \leftarrow a'_i - c_{x'_i} + \sum_{u_k \in \mathcal{U}: x_k = x'_i, k \neq i} a_k$;
    按照$\hat{\pi}_k$对物联网用户$u_k \in \mathcal{U} : x_k = x'_i \ and \ k \neq i$进行非降序排序, 其经过排序的用户排列为$P$;
    初始化$\mathbf{s}'_{-i} \leftarrow \mathbf{s}_{-i}$;
    while $\triangle a_i > 0$ **do**
      选取用户排列P中的头部用户u_v;
      尝试更新物联网用户u_v的策略$\mathbf{s}'_v \leftarrow (x_v, a_v - 1)$;
      重新按照$\hat{\pi}_k$对物联网用户$u_k \in \mathcal{U} : x_k = x'_i \ and \ k \neq i$进行非降序排序;
      $\triangle a_i \leftarrow \triangle a_i - 1$;
    $\triangle \pi_i \leftarrow \pi_i(\mathbf{s}'_i, \mathbf{s}'_{-i}) - \pi_i(\mathbf{s}_i, \mathbf{s}_{-i})$;
    $\triangle \pi_{-i} \leftarrow \pi_{-i}(\mathbf{s}_i, \mathbf{s}_{-i}) - \pi_{-i}(\mathbf{s}'_i, \mathbf{s}'_{-i})$;
    if $\triangle \pi_i > \triangle \pi_{-i}$ **then**
      return $\boldsymbol{s}' \leftarrow \{\boldsymbol{s}'_i, \boldsymbol{s}'_{-i}\}$;
    else
      return $\boldsymbol{s}' \leftarrow \boldsymbol{s}$;
else
  return $\boldsymbol{s}' \leftarrow \boldsymbol{s}$;

在完成对所有物联网用户 $u_k \in p(u_i)$ 的排序之后，从用户序列 P 中拣选出具有最低 QoE 损失量 $\hat{\pi}_v$ 的物联网用户 u_v；然后，临时从物联网用户 u_v 处释放出单位量为 1 的边缘服务器资源，以供物联网用户 u_i 抢占其边缘资源。

$$\hat{\pi}_k = \begin{cases} \pi_k(\mathbf{s}_k) - \pi_i(x_k, a_k - 1) & \text{if } a_i \geq 2 \\ \pi_k(\mathbf{s}_k) - \pi_i(0,0) & \text{otherwise} \end{cases} \qquad \text{（式 5-15）}$$

接下来，开始下一轮的迭代过程。再次根据 $\hat{\pi}_k$ 对物联网用户 $u_k \in p(u_i)$ 进行非降序排序；其中，对于边缘资源使用情况被改动过的物联网用户 u_v 而言，其 $\hat{\pi}_v$ 值在本轮迭代中需要被重新计算，用于本轮迭代的物联网用户 $p(u_i)$ 排序。对于本轮对物联网用户 $u_k \in p(u_i)$ 的重新排序过程，仅需在上一轮用户排序结果 P 的基础上完成；也就是说，只需要将物联网用户 u_v 插入到既有用户序列 P 中的适当位置。得益于此，本轮物联网用户的重新排序过程仅以 $\mathcal{O}(m)$ 的算法复杂度完成。经过对物联网用户 $u_k \in p(u_i)$ 的重新排序之后，执行与上一轮迭代中类似的步骤，从而从 $p(u_i)$ 中拣选出一位物联网用户，并临时让出单位量为 1 的边缘服务器资

源给物联网用户 u_i 。

上述迭代过程不断重复，直到物联网用户 $p(u_i)$ 释放出单位量为 $\triangle a_i$ 的边缘服务器资源。当多轮迭代过程终止时，可以得到物联网用户 $p(u_i)$ 的最低 QoE 损失量 $\triangle \pi_{-i}$，用于验证是否满足边缘资源抢占的触发条件（即式（5-14））。如果满足该触发条件，则最终确定将预先临时释放的边缘服务器资源 $\triangle a_i$ 交由物联网用户 u_i 抢占。鉴于多轮迭代过程中排序操作贡献了大部分 PRIM 算法复杂度，PRIM 算法复杂度为 $\mathcal{O}\left(m\left(\log m+\Delta a_i-1\right)\right)$。

5.4 基于 ERAGame 的分布式算法

5.4.1 QoE-DEER 算法设计

在 PRIM 算法的基础上，本小节进一步设计出 QoE 感知的分布式边缘资源分配算法（QoE-DEER），从而找出 ERAGame 具有较高系统 QoE 水平的纳什均衡解。该纳什均衡解确定了系统中多用户边缘资源分配方案。为了使多用户边缘资源分配过程以去中心化的形式进行，一种基于合作的消息传递机制被提出，如图 5-3 所示；在此基础上，分布式 QoE-DEER 算法被给出。具体而言，基于合作的消息传递机制涉及 5 种消息类型，用于维持物联网用户与边缘服务器之间的协调通信。

- **开始消息（Begin Message，BM）**：只要存在请求更新的策略，则 ERAGame 博弈就未达到纳什均衡状态。这时，每个边缘服务器 e_j 所属的物联网用户 $u_i \in \mathcal{U}_j$ 广播 BM 消息，表示继续 ERAGame 博弈。
- **信息消息（Information Message，IM）**：在每个边缘服务器 e_j 向物联网用户广播 BM 消息的同时，也通过发送 IM 消息来向物联网用户通知当前的边缘服务器资源分配状态。
- **策略消息（Strategy Message，SM）**：收到 IM 消息后，每个物联网用户 u_i 会决定是否请求更新当前的边缘资源分配策略。如果需要更新，则物联网用户 u_i 发送 SM 消息给有关的边缘服务器，以期获得策略更新的许可。
- **允许消息（Allow Message，AM）**：在 ERAGame 博弈模型中，每个时间槽 t 里仅能有一位用户的 SM 消息所披露的策略更新请求可以被允许。因此，所有边缘服务器将互相协商，然后仅发送出一条 AM 消息给被允许执行策略更新的 AM 请求用户。该协商过程是基于全局 QoE 改进的。
- **更新消息（Update Message，UM）**：一旦确定被批准的策略更新请求后，所有边缘服务器也应该向其边缘资源被抢占的物联网用户发送 UM 消息，从而

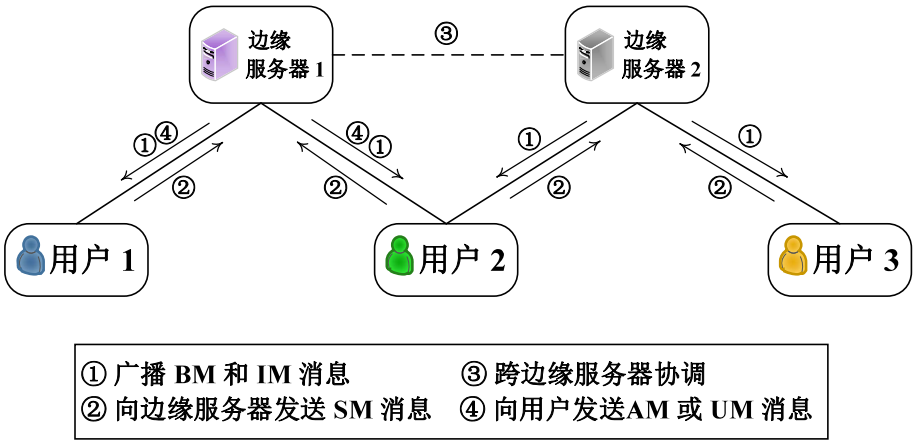


图 5-3. 每个时间槽t决策中传递的消息流图

告知新实施的博弈策略。

在上述消息传递机制的基础上，分布式 QoE-DEER 算法被详细提出，如算法 2 所示。值得注意的是，$\triangle t$ 被预定义为边缘服务器与物联网用户之前消息传递所需要的最长等待时间；它将被用于分析 QoE-DEER 算法效率。分布式 QoE-DEER 算法将在每个时间槽t被执行，直到当前时间槽的 ERAGame 博弈过程结束。具体分为如下三个步骤：

步骤 1（第 1-3 行）：如果 ERAGame 博弈尚未达到纳什均衡状态，则各个边缘服务器将向所属的物联网用户广播 BM 消息，表明“继续博弈”。与此同时，每个边缘服务器都会更新记录当前的边缘资源分配状态，并将其打包为 IM 消息、再发送到每个边缘服务器所下属的物联网用户，为各用户提供策略更新的参考。

步骤 2（第 4-10 行）：收到各边缘服务器发来的 IM 消息后，每位物联网用户u_i将查找是否存在一个 QoE 改进的策略s'。如果存在，则物联网用户u_i将发送 SM 消息到s'指定的边缘服务器。

步骤 3（第 11-18 行）：如果有物联网用户请求策略更新，则边缘服务器将不会晚于$\triangle t$收到 SM 消息。在等待$\triangle t$时间后，边缘服务器应接收到所有 SM 消息；然后，开始协商并决定哪个博弈策略更新请求被允许，AM 消息被作为允许策略更新的消息响应。相反，如果在$\triangle t$时间范围内未收到任何 SM 消息，则表明 ERAGame 博弈达到纳什均衡状态，从而 QoE-DEER 算法终止。

在实际的物联网应用场景中，每个时间槽t决策执行的分布式 QoE-DEER 算法可在最长 $3\triangle t$ 时间内近似完成。在步骤 1 里，每台边缘服务器e_j向所属物联网用户$u_i \in \mathcal{U}_j$同时发送 BM 消息和 IM 消息，最长需要$\triangle t$时间。然后，在步骤 2 里，如果存在物联网用户u_i请求更新博弈策略，则物联网用户u_i会向边缘服务器发送 SM 消息；通常地，边缘服务器不会晚于$\triangle t$时间内收到 SM 消息。在步骤 3 里，收到 SM 消息的边缘服务器会决定哪个博弈策略更新请求被允许，并向

Algorithm 2: QoE感知分布式边缘资源分配算法(QoE-DEER)

Input: 当前时间槽t的策略组合$\mathbf{s}(t)$.
Output: 下一时间槽$t+1$的策略组合$\mathbf{s}(t+1)$.

初始化设置$\mathbf{s}(t+1) \leftarrow \mathbf{s}(t)$;
步骤1: 继续博弈
每台边缘服务器向所属物联网用户广播BM和IM消息;
步骤2: 获取QoE改进的新博弈策略
for 每位物联网用户$i \in \mathcal{U}$ **do**
  设置临时变量$\mathbf{s}' \leftarrow \mathbf{s}(t)$;
  for 每个可行策略$\hat{s}_i \in S_i$ **do**
    $\mathbf{s}' \leftarrow PRIM(\hat{\mathbf{s}}_i, \mathbf{s}')$;
  if $s' \neq s(t)$ **then**
    向新策略$\mathbf{s}'_i$所指明的边缘服务器发送SM消息;
步骤3: 选出并实施一条QoE改进的新策略组合
if 任一边缘服务器在$\triangle t$时间内收到SM消息 **then**
  拣选出系统QoE提升幅度最大的新策略组合$\mathbf{s}^*$;
  采用新策略组合$\mathbf{s}(t+1) \leftarrow \mathbf{s}^*$;
  向请求实施新策略组合$\mathbf{s}^*$的物联网用户发送AM消息;
  向被抢占的物联网用户发送UM消息;
else
  return $s(t+1)$; ▷ 达到ERAGame纳什均衡状态.

相关物联网用户发送 AM 消息/UM 消息；在最晚$\triangle t$时间内，物联网用户会收到由边缘服务器发出的 AM 消息/UM 消息。综上所述，分布式 QoE-DEER 算法可以在最长 3$\triangle t$ 时间内近似完成。

5.4.2 QoE-DEER 算法分析

1）算法收敛性分析

本部分将研究 QoE-DEER 算法是否能够在有限次数的博弈回合内收敛得到一个纳什均衡解。因为分布式 QoE-DEER 算法是 ERAGame 博弈的去中心化实现方案，所以仅需要验证 ERAGame 博弈是否能够在有限的博弈回合次数内达到纳什均衡状态。根据文献 [120]，潜在博弈具有重要性质——有限改进（Finite Improvement），表明潜在博弈可以在有限次博弈回合内收敛到纳什均衡状态。因此，如果 ERAGame 博弈被证明是潜在博弈，那么 QoE-DEER 算法的收敛性可得到保证。在定义 4 中，潜在博弈被严格地定义。

定义 4（潜在博弈，Potential Game）：如果存在潜在函数$\Phi: S \to \mathbb{R}$，使得对于每位局中人$u_i \in \mathcal{U}$，从$\boldsymbol{s} = (\boldsymbol{s}_i, \boldsymbol{s}_{-i})$到$\boldsymbol{s}' = (\boldsymbol{s}'_i, \boldsymbol{s}'_{-i})$的博弈策略改进，满足

$$\pi_i(\boldsymbol{s}'_i) > \pi_i(\boldsymbol{s}_i) \Rightarrow \Phi(s') > \Phi(s) \qquad \text{式（5-16）}$$

则该博弈模型$\mathcal{G}$属于潜在博弈。 □

由于边缘资源抢占机制确保 ERAGame 博弈可以达到具有较高系统 QoE 水

平的纳什均衡状态，所以从策略组合 $\boldsymbol{s}$ 到 $\boldsymbol{s}'$ 的每次博弈策略改进过程中，多位物联网用户（包括抢占/被抢占的物联网用户）的博弈策略 $\mathbf{s}_i$ 可能同时发生改变。根据定义 4，对 ERAGame 博弈的纳什均衡解 $\pi_i\left(\mathbf{s}_i^*,\mathbf{s}_{-i}^*\right)$ 可给出如下描述

$$\forall u_i \in \mathcal{U}，\ \pi_i\left(\mathbf{s}_i^*,\mathbf{s}_{-i}^*\right)=\max_{\mathbf{s}_i\in\mathcal{S}}\ \pi_i\left(\mathbf{s}_i,\mathbf{s}_{-i}^*\right) 成立$$

由此可知，通过找出潜在函数 $\Phi\left(S\right)$ 的最优解，ERAGame 纳什均衡解也就被找到 [121]。在不断改进博弈策略的同时，相应潜在函数 $\Phi\left(S\right)$ 也随着单调性增长；当潜在函数 $\Phi(\mathbf{s})$ 增长并收敛到其最大值时，则表明当前策略组合 $\mathbf{s}^*$ 达到纳什均衡状态。上述潜在函数 $\Phi(\mathbf{s})$ 随着策略改进而增长（直到收敛）的过程体现了潜在博弈的有限改进性质。在相关研究文献 [44, 45, 46] 中，已经利用有限改进的性质去设计潜在博弈模型，从而准确描述和管理多个实体间的竞争态势。在定理 1 中，ERAGame 被严格地证明为潜在博弈，其中潜在函数 $\Phi(\mathbf{s})$ 在式（5-17）中被给出。

$$\Phi(\mathbf{s})=\sum_{u_i\in\mathcal{U}}\pi_i\left(\mathbf{s}_i\right) \qquad 式（5\text{-}17）$$

定理 1（ERA 潜在博弈）： ERAGame 博弈 $\mathcal{G}=\left\langle \mathcal{U},\left(\boldsymbol{S}_i\right)_{u_i\in\mathcal{U}},\left(\pi_i\right)_{u_i\in\mathcal{U}}\right\rangle$ 属于潜在博弈，其潜在函数 $\Phi(\mathbf{s})$ 由式（5-17）表示。

证明： 假定物联网用户 $u_i\in\mathcal{U}$ 从 $\mathbf{s}_i$ 到 $\mathbf{s}_i'$ 改进其博弈策略，从而实现 $\pi_i(\mathbf{s}_i',\mathbf{s}_{-i}')>\pi_i\left(\mathbf{s}_i,\mathbf{s}_{-i}\right)$。为了证明定理 1，则需由 $\pi_i(\mathbf{s}_i',\mathbf{s}_{-i}')>\pi_i\left(\mathbf{s}_i,\mathbf{s}_{-i}\right)$ 推导出 $\Phi\left(\mathbf{s}'\right)>\Phi(\mathbf{s})$。注意，$\mathbf{s}_i=\left(x_i,a_i\right)$ 和 $\mathbf{s}_i'=(x_i',a_i')$。

根据边缘资源抢占机制，物联网用户 u_i 可能需要执行对边缘服务器资源的抢占操作，从而实施从 $\mathbf{s}_i$ 到 $\mathbf{s}_i'$ 的策略改进。具体而言，当前被其他物联网用户 $\left\{u_k\in\mathcal{U}:x_k=x_i'\text{ and }k\neq i\right\}$ 占用的边缘服务器资源需要被用户 u_i 抢占。用 $p\left(u_i\right)$ 表示可能被 u_i 抢占的物联网用户集合；在完成边缘资源抢占之后，物联网用户 $u_v\in p\left(u_i\right)$ 需要从原有策略 $\mathbf{s}_v$ 变更为 $\mathbf{s}_v'$，并导致相应的 QoE 损失量 $\pi_v\left(\mathbf{s}_v\right)-\pi_v\left(\mathbf{s}_v'\right)$。所有物联网用户 $u_v\in p\left(u_i\right)$ 的总 QoE 损失量为 $\sum_{u_v\in p(u_i)}\left(\pi_v\left(\mathbf{s}_v\right)-\pi_v\left(\mathbf{s}_v'\right)\right)$。

根据边缘资源抢占机制的触发条件（即式（5-14））可知，当且仅当式（5-18）成立时，对边缘服务器资源的抢占操作会被允许。

$$\pi_i\left(\mathbf{s}_i'\right)-\pi_i\left(\mathbf{s}_i\right)>\sum_{u_v\in p(u_i)}\left(\pi_v\left(\mathbf{s}_v\right)-\pi_v\left(\mathbf{s}_v'\right)\right) \qquad 式（5\text{-}18）$$

据此，可以提供如下推导，

$$\Phi(\mathbf{s}')-\Phi(\mathbf{s})=\left(\pi_i(\mathbf{s}_i')+\sum_{u_v\in p(u_i)}\pi_v(\mathbf{s}_v')\right)-\left(\pi_i(\mathbf{s}_i)+\sum_{u_v\in p(u_i)}\pi_v(\mathbf{s}_v)\right)$$
$$=\left(\pi_i(\mathbf{s}_i')-\pi_i(\mathbf{s}_i)\right)+\sum_{u_v\in p(u_i)}\left(\pi_v(\mathbf{s}_v')-\pi_v(\mathbf{s}_v)\right)>0$$

由上述推导与分析过程可知，如果 $\pi_i(\mathbf{s}')-\pi_i(\mathbf{s})>0$ 则 $\Phi(\mathbf{s}')-\Phi(\mathbf{s})>0$。据此可证，ERAGame 博弈 $\mathcal{G}$ 属于潜在博弈，其潜在函数是 $\Phi(\mathbf{s})=\sum_{u_i\in\mathcal{U}}\pi_i(\mathbf{s}_i)$。 □

2）算法最优性分析

从算法最优性的角度，本部分将研究分布式 QoE-DEER 算法的有效性，去判断分布式 QoE-DEER 算法是否能等价地求解原始 ERA 优化问题（由式（5-10）定义）。如本章 5.2.3 小节中所讨论，原始 ERA 优化问题旨在最大化多用户的系统 QoE 水平。鉴于 ERA 优化问题的诸多求解挑战，本章设计出 ERAGame 博弈模型，从而试图以去中心化的方式求解 ERA 优化问题。然而，ERAGame 博弈可能存在非唯一的纳什均衡状态 [119]。因此，借助于分布式 QoE-DEER 算法，ERAGame 博弈是否能够得到最大化系统 QoE 水平的纳什均衡解是十分值得研究的。如果可以的话，原始 ERA 优化问题则能以去中心化的方式被高效地求解。在定理 2 中，给出了相关的严格证明过程。

定理 2：分布式 QoE-DEER 算法收敛于最大化系统 QoE 水平的纳什均衡解。

证明：设定用 $\mathbf{s}=\{\mathbf{s}_1,\ldots,\mathbf{s}_n\}$ 表示由分布式 QoE-DEER 算法解出的策略组合，而用 $\mathbf{s}^*=\{\mathbf{s}_1^*,\ldots,\mathbf{s}_n^*\}$ 表示最大化系统 QoE 水平 $\sum_{u_i\in\mathcal{U}}\pi_i(\mathbf{s}_i^*)$ 的最优策略组合。

简而言之，使用反证法来证明定理 2。不妨假设，由 QoE-DEER 算法解出的策略组合 $\mathbf{s}$ 不能实现系统 QoE 水平的最大化，即 $\sum_{u_i\in\mathcal{U}}\pi_i(\mathbf{s}_i)<\sum_{u_i\in\mathcal{U}}\pi_i(\mathbf{s}_i^*)$。这时，必定存在一部分物联网用户，其在最优策略组合 $\mathbf{s}^*$ 中具有更高的 QoE 水平。基于此，将全体物联网用户 $\mathcal{U}$ 分成两组，即 $\mathcal{G}_1$ 和 $\mathcal{G}_2$。

物联网用户组 $\mathcal{G}_1$ 由在最优策略组合 $\mathbf{s}^*$ 中具有更高 QoE 水平的用户组成，则意味着 $\sum_{u_i\in\mathcal{G}_1}\pi_i(\mathbf{s}_i^*)>\sum_{u_i\in\mathcal{G}_1}\pi_i(\mathbf{s}_i)$。对于 $\mathcal{G}_1$ 中的物联网用户而言，在最优策略组合 $\mathbf{s}^*$ 中相对于策略组合 $\mathbf{s}$ 的 QoE 增量由 $\Delta I=\sum_{u_i\in\mathcal{G}_1}\left(\pi_i(\mathbf{s}_i^*)-\pi_i(\mathbf{s}_i)\right)$ 表示。

物联网用户组 $\mathcal{G}_2$ 则由剩余物联网用户 $\mathcal{U}-\mathcal{G}_1$ 组成。在最坏的情况下，对于物联网用户 $u_i\in\mathcal{G}_2$ 而言，在策略组合 $\mathbf{s}$ 中相对于最优策略组合 $\mathbf{s}^*$ 有部分 QoE 损失，即 $\sum_{u_i\in\mathcal{G}_2}\pi_i(\mathbf{s}_i^*)\le\sum_{u_i\in\mathcal{G}_2}\pi_i(\mathbf{s}_i)$。此时，相应的 QoE 减量由 $\Delta D=\sum_{u_i\in\mathcal{G}_2}\left(\pi_i(\mathbf{s}_i)-\pi_i(\mathbf{s}_i^*)\right)$ 表示。

鉴于$\sum_{u_i\in\mathcal{U}}\pi_i\left(\mathbf{s}_i\right)<\sum_{u_i\in\mathcal{U}}\pi_i\left(\mathbf{s}_i^*\right)$，由物联网用户组$\mathcal{G}_1$带来的 QoE 增量$\triangle I$应该大于由物联网用户组$\mathcal{G}_2$带来的 QoE 减量$\triangle D$，即$\triangle I>\triangle D$。根据式（5-14），这将触发对边缘服务器资源的抢占操作，从而进一步提升多用户的系统 QoE 水平。值得注意的是，只有没有物联网用户请求改进其博弈策略（包括执行抢占操作的请求），分布式 QoE-DEER 算法才会终止。换句话说，根据$\sum_{u_i\in\mathcal{U}}\pi_i\left(\mathbf{s}_i\right)<\sum_{u_i\in\mathcal{U}}\pi_i\left(\mathbf{s}_i^*\right)$，策略组合$\mathbf{s}$不可能是 QoE-DEER 算法的最终解。这与由反证法做出的基本假设产生了矛盾。

综上所述，分布式 QoE-DEER 算法提供了最大化系统 QoE 水平的纳什均衡解，从而以去中心化的方式等价地求解了原始 ERA 优化问题。 □

5.5 实验验证

5.5.1 实验设置

本小节使用 MATLAB 搭建边缘计算系统的仿真环境，其中采用了边缘计算 EUA 数据集[122]。边缘计算 EUA 数据集分别记录了澳大利亚墨尔本中央商务区内移动蜂窝基站、移动终端用户的经/纬度位置信息。根据澳大利亚通信媒体管理局 ACMA 所公开的无线电通信许可数据，移动蜂窝基站的位置信息从中被解析得到。亚太互联网络信息中心 APNIC 提供了分配给澳大利亚地区的各 IP 地址区段；移动终端用户的位置信息则借助 IP 地址检索服务 http://ip-api.com/ 将用户 IP 地址转换为相应的经/纬度位置信息。由于边缘服务器通常在移动蜂窝基站处被部署，因此在本小节实验设置中将移动蜂窝基站的位置信息作为边缘服务器的部署位置。与此同时，将 EUA 数据集中移动终端用户的位置信息也作为仿真环境中物联网用户的所处位置。

对于每台边缘服务器e_j，其无线通信信号的覆盖半径在 450 米至 750 米之间随机设置。每单位边缘服务器资源的 CPU 频率设置为 10 GHz，而在不同的具体实验设置里，每台边缘服务器的资源容量c_j在 6~26 之间设置。

对于每位物联网用户u_i，若在本地物联网设备上处理其服务请求，则本地物联网设备的 CPU 频率根据 [1.5, 2] GHz 的范围而随机设置。同时，每位物联网用户u_i的 QoE 指标参数α_i和β_i则分别根据正态分布$\mathcal{N}\left(1.5,0.25^2\right)$和$\mathcal{N}\left(75,10^2\right)$而设置。再者，每位物联网用户$u_i$需向边缘服务器上传的数据量$\eta_i$则根据 [150, 200] KB 的范围而随机设置；[150, 200] KB 的上传数据量范围与一般分辨率的图片大小相匹配。最后，每位物联网用户的服务请求包含待处理的服务数据η_i，则每单位物联网服务数据被处理的所需 CPU 周期数为 5。

至于物联网用户与边缘服务器之间的无线通信环境设置，每台物联网设备/用

户 u_i 的信号传输功率 P_i 为 0.5 W，无线通信带宽 B 为 10 MHz [123, 124]，无线通信噪声功率 σ^2 为-87 dBm [125]，单位无线通信距离的信道功率 γ_0 为-50 dB [110, 126]。

5.5.2 实验结果

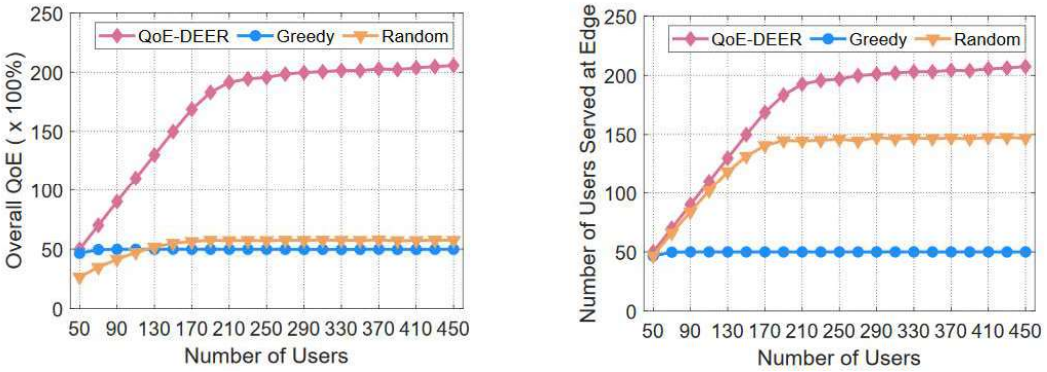
1）纳什均衡解的最优性

为了评估分布式 QoE-DEER 算法的有效性，本部分将研究 QoE-DEER 算法所收敛到纳什均衡解的最优性。结合边缘计算系统环境，纳什均衡解的最优性分别从两个方面进行评估，即多用户的系统 QoE 水平、在边缘服务器端被服务的物联网用户数。除此之外，分布式 QoE-DEER 算法将与贪婪算法（Greedy）、随机化算法（Random）进行比较。

- **贪婪算法（Greedy）：**在文献 [25] 中被提出。具体而言，以物联网用户 QoE 参数 α_i 的非递增排序结果作为各物联网用户服务请求的调度优先级，依此对每位物联网用户 u_i 执行服务调度和边缘资源分配。基于当前空闲可用的边缘服务器资源，每位物联网用户 u_i 将尽可能地被分配到更多的边缘服务器资源，使其获得尽可能最高的自身 QoE 水平。当物联网用户 u_i 在服务调度队列中被排到时，如果没有空闲可用的边缘服务器资源，则在本地物联网设备上处理其服务请求。
- **随机化算法（Random）：**按照随机顺序，对每位物联网用户 u_i 的服务请求执行边缘服务器调度和资源分配。具体地，每位物联网用户 u_i 再以随机的方式被调度到边缘服务器 $e_j \in \mathcal{E}_i$，并被随机分配到单位量为 a_i 的边缘服务器资源。如果物联网用户 u_i 所有可访问的边缘服务器资源都已被其他物联网用户占用，那么该物联网用户将不得不在本地物联网设备上处理其服务请求。

注意，与分布式 QoE-DEER 算法、贪婪算法有关的每项实验结果是经过 60 次重复执行而取平均值得到的，而关于随机化算法的每项实验结果则是经过 720 次重复执行后而取平均值得到的。在每次算法执行期间，依据 EUA 数据集所提供的位置信息，随机分配和指定仿真环境中边缘服务器、物联网用户的不同所处位置。如 5.2.3 小节所述，由于原始 ERA 优化问题是 NP 难解的，所以很难以在有限的合理时间内以集中决策的中心化方式完成 ERA 优化问题的求解（特别当优化问题规模较大时）。鉴于此，本章没有将 ERA 优化问题的中心化求解结果作为评估分布式 QoE-DEER 算法的基准实验结果。

首先，通过设置不同的物联网用户数量 N（即从 50 位 ~ 450 位）来实验验证和评估分布式 QoE-DEER 算法所得到纳什均衡解的最优性。这时，边缘服务器数量 M 和每台边缘服务器的资源容量 c_j 被分别固定设置为 50 和 10。具体实

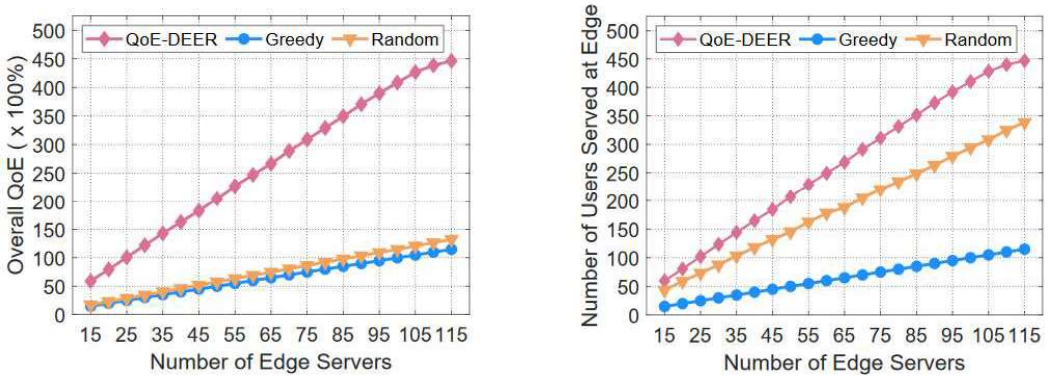


(a) 多用户的系统 QoE 水平　　(b) 在边缘端被服务的物联网用户数

图 5-4. QoE-DEER 纳什均衡解的最优性 v.s. 物联网用户数量 N

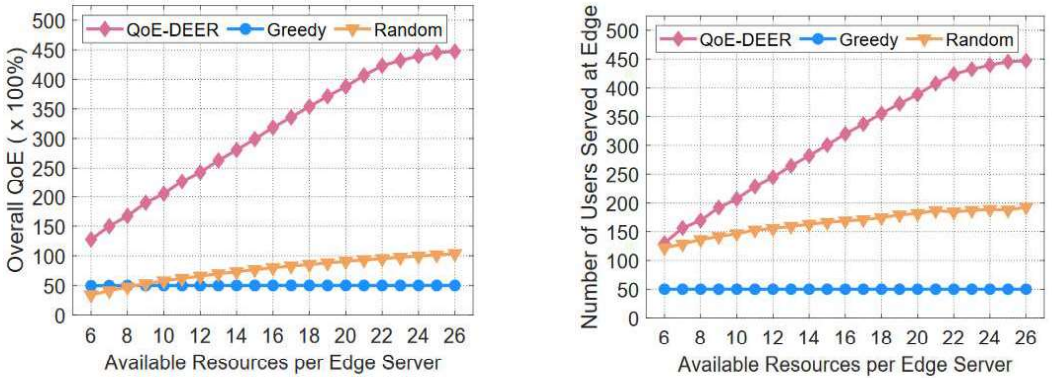
验结果如图 5-4 所示。在不同的物联网用户数量 N 设置下，相较于其他两种对比算法而言，本章所提出的分布式 QoE-DEER 算法始终获得了最高的系统 QoE 水平、最多的在边缘端被服务的物联网用户。除此之外，随着物联网用户规模的增长，分布式 QoE-DEER 算法在纳什均衡解的最优性方面的优势也不断地扩大，直到物联网用户规模发展到 210 位之后。这是因为受限于有限的边缘服务器资源供应，当物联网用户规模发展到一定程度后，分布式 QoE-DEER 算法无法在边缘端服务更多的物联网用户，从而系统 QoE 水平也不会随之再有显著提升。值得注意的是，在不同的物联网用户数 N 设置下，贪婪算法所获得的系统 QoE 水平几乎保持不变。这是因为，在贪婪算法中，一旦物联网用户 u_i 被调度到边缘服务器，则将垄断占用整台边缘服务器的全部计算资源，从而获得尽可能最高的自身 QoE 水平。然而，系统中边缘服务器数量 M 通常远小于物联网用户数量 N。这样的话，在图 5-4 中物联网用户的初始规模（即 50 位物联网用户）之下，所有边缘服务器资源已经被当前的物联网用户所占用；尽管物联网用户规模 N 继续扩大，也没有进一步提升系统 QoE 水平的空间。

然后，通过设置不同的边缘服务器数量 M（即从 15 台 ~115 台）来进一步验证和评估分布式 QoE-DEER 算法所得到纳什均衡解的最优性。这时，物联网用户数量 N 和每台边缘服务器的资源容量 c_j 被分别固定设置为 450 和 10。在不同边缘服务器数量 M 下的具体实验结果如图 5-5 所示。本章所设计的分布式 QoE-DEER 算法以及其他两种对比算法所获得的系统 QoE 水平、边缘端服务的物联网用户数，都随着边缘服务器数量 M 的增大而不断增加。与其他两种对比算法相比，分布式 QoE-DEER 算法具有最高的系统 QoE 水平、最多的在边缘端被服务的物联网用户。更多台边缘服务器被部署加入边缘计算系统则意味着更强的边缘端服务承载能力，所以在边缘服务器端被服务的物联网用户数不断地增加，如图 5-5(b) 所示。相较于在本地物联网设备上处理服务请求，物联网用户 u_i 可以在边缘服务器获得更高的 QoE 水平。因此，越来越多的物联网用户被调度到边



(a) 多用户的系统 QoE 水平　　(b) 在边缘端被服务的物联网用户数

图 5-5. QoE-DEER 纳什均衡解的最优性 v.s. 边缘服务器数量 M

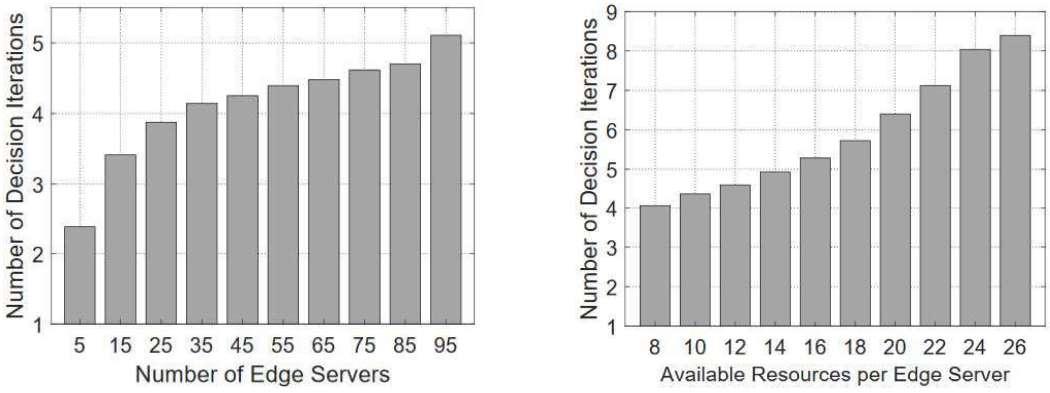


(a) 多用户的系统 QoE 水平　　(b) 在边缘端被服务的物联网用户数

图 5-6. QoE-DEER 纳什均衡解的最优性 v.s. 边缘服务器资源容量 c_j

缘服务器来处理其服务请求，则整体系统 QoE 水平也会随之得到提升，如图 5-5(a)所示。当边缘服务器数量 M 被部署超过 105 台时，分布式 QoE-DEER 算法所获得的系统 QoE 水平提升幅度逐渐趋于缓慢。这是因为这时几乎所有 $N=450$ 位物联网用户都可以在边缘服务器端处理其服务请求，进而继续扩大边缘服务器的部署也不能大幅度提升多用户的系统 QoE 水平。

最后，将研究不同边缘服务器资源容量 c_j（即从 6～26）对分布式 QoE-DEER 算法所得到纳什均衡解最优性的影响，其实验结果如 5-6 所示。这时，边缘服务器数量 M 、物联网用户数量 N 被分别固定设置为 450 台和 500 位。在不同边缘服务器资源容量 c_j 的设置下，本章所设计的分布式 QoE-DEER 算法在系统 QoE 水平、在边缘端被服务的物联网用户数都优于其他两种对比算法。同时，随着边缘服务器资源容量 c_j 的增加，分布式 QoE-DEER 算法相对于其他两种对比算法的优越性将更加明显，特别是相较于贪婪算法而言。在贪婪算法里，无论边缘服务器资源容量 c_j 被如何设置，如果物联网用户 u_i 的服务请求被调度到边缘服务器端来处理，那么她会独占整台边缘服务器的全部计算资源，而不会为其他物联



(a) QoE-DEER 算法收敛性 v.s. M　　(b) QoE-DEER 算法收敛性 v.s. c_j

图 5-7. 分布式 QoE-DEER 算法收敛性的实验结果

网用户保留任何空闲资源。因此，根据贪婪算法，最多仅有 M 位物联网用户会选择在边缘服务器处理其服务请求，其中通常 $M << N$。再者，每位物联网用户 u_i 只能最高获得 100%的 QoE 水平，故而在贪婪算法里，对边缘服务器资源容量 c_j 的提高将不能够大幅度地提升系统 QoE 水平。注意，当 $c_j \geq 22$ 时，分布式 QoE-DEER 算法所获得的系统 QoE 水平提升幅度逐步趋弱。得益于边缘服务器资源容量 c_j 的提高，当 $c_j = 22$ 时边缘服务器已经足够完全为 450 位物联网用户提供较高 QoE 水平的服务。

2）QoE-DEER 算法收敛性

通过仿真实验，本部分将评估分布式 QoE-DEER 算法的收敛性。从理论上，定理 1 已经严格地证明出分布式 QoE-DEER 算法的收敛性。在仿真实验中，采用分布式 QoE-DEER 算法收敛于纳什均衡解所需的博弈回合次数来衡量其算法收敛性。类似于实验评估纳什均衡解的最优性，本部分在不同的实验设置下验证分布式 QoE-DEER 算法的收敛性。具体实验结果如图 5-7 和表 5-2 所示，其中每项实验结果是经过重复 1000 次 QoE-DEER 算法执行后而取平均值得到的。在每次 QoE-DEER 算法执行中，仿真环境中物联网用户和边缘服务器的位置信息都不相同，从 EUA 数据集中随机拣选出指定数量的不同边缘服务器和物联网用户。

图 5-7(a) 展示了随着系统中边缘服务器数量 M 的增加，分布式 QoE-DEER 算法所需的博弈回合次数。由于更多台边缘服务器参与到 ERAGame 博弈过程中，因此每位物联网用户的策略决策空间也相应扩大，从而降低了分布式 QoE- DEER 算法的收敛速度。如图 5-6(a) 所示，ERAGame 博弈回合次数随着边缘服务器数量 M 的增加而缓慢增加。

图 5-7(b) 展示了在不同边缘服务器资源容量 c_j 的设置下，分布式 QoE-DEER 算法所需的博弈回合次数。随着边缘服务器资源容量 c_j 的增加，ERAGame 博弈

表 5-2. 分布式 QoE-DEER 算法收敛性 v.s. 物联网用户数量 N

物联网用户数量 N	50	130	210	290	380	450
博弈回合次数	4.629	6.419	5.646	4.643	4.455	4.337

回合次数也同步地增长。当 $c_j = 8$ 时，平均博弈回合次数是 4.053 次；当 $c_j = 26$ 时，平均博弈回合次数是 8.388 次。在不同的边缘服务器资源容量 c_j 之下，分布式 QoE-DEER 算法所需的博弈回合次数总是可以接受的。这样的话，本章所提出的分布式 QoE-DEER 算法在实际物联网应用中切实可用。

表 5-2 列出了在不同物联网用户数量 N 的设置下，分布式 QoE-DEER 算法所需的博弈回合次数。随着不同的物联网用户数量 N，ERAGame 博弈回合次数的变化趋势并不甚明显，但是平均博弈回合次数是 5.0215 次，相对较低。由此可见，在大规模边缘计算系统中，分布式 QoE-DEER 算法总是保持着高效性。

5.6 本章小结

从博弈论的观点出发，本章提出了多用户边缘资源分配问题（简写 ERA 问题）的高效求解方案，从而最大限度地提升系统 QoE 水平。针对求解 ERA 优化问题的诸多难题与挑战，使用潜在博弈模型 ERAGame 来描述原始的 ERA 优化问题，从而给出一种去中中心化的求解方案。基于对自身 QoE 水平的提升需求，每位物联网用户自主地决定其边缘资源分配方案。为了使 ERAGame 收敛于具有较高系统 QoE 水平的纳什均衡状态，一种边缘资源抢占机制 PRIM 被引入以协调多用户的边缘资源博弈过程。最后，基于合作消息传递机制，设计了 QoE 感知的分布式边缘资源分配算法 QoE-DEER，以去中心化的方式求出原始 ERA 优化问题的等价最优解。经过严格理论分析和仿真实验验证，分布式 QoE-DEER 算法的有效性和收敛性被充分地评估和验证。

第六章 总结与展望

6.1 论文总结

面向生态化的服务计算环境，本文围绕着如何执行 QoS 感知的服务资源调度与优化来进行深入研究。近年来，随着服务计算技术的日益流行和蓬勃发展，服务计算系统趋于生态化。从服务生态化的观点出发，通过来考虑 QoS 感知的服务资源调度与优化，将有效地促进服务计算系统的生态化演进和优化，最终形成由多服务供应商/用户共同参与的生态化服务计算系统。其中，服务生态实体（即多服务供应商/用户）之间会相互竞争与博弈，其目的在于赚取较高服务收益/获得具有较高 QoS 水平的服务。因此，如何有效地管理服务生态实体之间的竞争态势是执行服务资源调度过程中所需重点关注的研究议题，从而实现公平的服务提供过程、有效的用户激励策略、多用户之间的良序博弈。这样的话，生态化服务计算系统的发展与演化过程就具备了可持续性。具体而言，本文的创新性工作和研究贡献如下：

1. 从生态化服务计算系统的角度出发，设计了一种实现最大最小公平性的多用户服务选择算法（FASS），从而为用户提供了一个符合用户公平性的服务提供平台。首先，建立 QoS 感知的多用户并发服务选择模型。然后，引入最大最小公平性的概念，并且给出基于字典序优化的公平优化问题描述。最终，设计出基于线性规划迭代的服务公平优化框架 FASS。在理论上，证明了 FASS 服务选择算法的最优性；与此同时，在仿真实验里，实验验证和评估了 FASS 算法的有效性和执行效率。FASS 服务公平优化框架较好地同时权衡兼顾了服务公平与 QoS 优化。不仅每位用户被一视同仁，而且服务供应商也尽可能地向每位用户提供了较高 QoS 水平的服务。
2. 从服务供应商的角度出发，设计了市场化的用户服务激励机制（PIRA），其旨在吸引大量用户来购买云服务资源，从而帮助服务供应商占据较大的云计算市场份额、赚取较高的服务收益。首先，对云资源拍卖市场进行建模。其次，构建 QoS 感知的用户效用模型，并且给出由用户个体理性驱动的云资源购买策略，使得云用户按照其自身 QoS 需求购买云资源。然后，设计出最大化用户激励的云资源拍卖机制 PIRA，其优化目标是激励出最大云用户数量来购买云服务资源。其中，服务收益最大化的云资源定价算法 RORP 被调用，用于帮助云服务供应商尽可能赚取最高的服务收益。在理论上，证明了 PIRA 机制满足预算可行性、激励相容性、无妒性；在仿真实验里，充分地实验验证和评估了 PIRA 机制的实际性能。
3. 从用户的角度出发，设计了 QoE 感知的去中心化边缘资源分配博弈算法（QoE-DEER），从而有效地管理了多用户之间的竞博弈关系，促成了生态化服务计算系统的良序竞争态势。首先，构建了针对边缘计算架构的系统模型，并且

详细分析了 QoE 指标与 QoS 指标之间的量化关联关系。其次，给出最大化系统 QoE 水平的 ERA 优化问题描述，其旨在得出系统 QoE 水平最优的多用户边缘资源分配方案。鉴于求解 ERA 优化问题的难点与挑战，引入潜在博弈模型 ERAGame，使得多用户的边缘资源分配方案能够以去中心化的方式被求解得到。最后，在边缘资源抢占机制 PRIM、面向多用户合作的消息传递机制的基础上，设计了分布式 QoE-DEER 算法以得到具有最高系统 QoE 水平的纳什均衡解。分布式 QoE-DEER 算法性能和收敛性都经过了理论证明和仿真实验的验证。综上所述，面对多用户的服务资源竞争与博弈时，分布式 QoE-DEER 算法为生态化服务计算系统提供了一种去中心化的服务资源管理方案。

6.2 未来研究展望

针对生态化的服务计算环境，本文在 QoS 感知的服务资源调度与优化方面上取得了一定的研究成果，但是仍然存在着值得进一步探索的研究问题，具体为如下几处：

1. 本文主要考虑了多用户之间的服务公平性，提出了基于最大最小公平性的多用户服务选择算法 FASS。然而，在多服务供应商/用户共同参与的生态化服务计算系统中，如何实现多服务供应商之间的服务公平性是同样重要的科研议题。具体而言，不同服务供应商都应该具有相对均衡的机会来向用户提供优质 QoS 水平的服务。进一步地，每位服务供应商将会拥有同等机会来赚取服务收益，从而实现跨服务供应商的公平服务提供机制。这将有利于促进服务供应商之间的良序竞争态势，从而服务供应商能够不断地向用户提供更加优质 QoS 水平的服务。
2. 本文所提出的价格激励资源拍卖机制 PIRA 是基于首价密封拍卖模型的，即针对不同竞拍者（用户）采用统一的资源定价标准。正因为如此，PIRA 拍卖机制缺乏次高价投标拍卖模型所具备的价格弹性。通常地，次高价投标拍卖模型具有更好的价格激励能力；它能够赋予云服务供应商更高的赚取服务收益能力。然而，在本质上，基于次高价投标拍卖模型的资源分配机制通常对竞拍者（用户）带有歧视性，即不能提供无妒性保证。因此，在未来的研究工作中，将针对生态化服务计算环境（如云计算环境），设计出无用户歧视性的次高价投标拍卖机制。它不仅会具备自身的价格激励优势，而且还会提供无妒性保证，所得到的服务资源分配方案可以满足多用户服务公平性。
3. 本文所提出的分布式 QoE-DEER 算法是基于潜在博弈模型 ERAGame 的。值得注意的是，潜在博弈模型 ERAGame 属于静态博弈模型，其缺乏较好的动态建模分析能力。然而，面向生态化服务计算环境的服务资源调度过程应该注重有关服务供应商/用户的实时动态系统变化，比如，当前时刻的服务资源

使用情况、当前用户的服务资源需求水平等。鉴于此，在未来的研究工作中，将考虑使用斯塔克尔伯格博弈、马尔可夫博弈等动态博弈模型来分析服务计算系统中的各类动态操作与变化。基于动态博弈模型，关于服务资源调度的决策将更加精准有力，相应的 QoS 优化效果将更佳。

参考文献

[1] L. J. Zhang, J. Zhang, H. Cai. Services Computing. Beijing: Springer and Tsinghua University Press, 2007.

[2] X. Xue, Z. Feng, S. Chen, Z. Zhou, C. Qin, B. Li, Z. Wang, B. Hu, H. Wu, S. Wang, L. Zhang, “Service Ecosystem: A Lens of Smart Society,” 2020, Online Available: https://arxiv.org/abs/2008.03418.

[3] K. Bessai, S. Youcef, A. Oulamara, C. Godart, “Bi-Criteria Strategies for Business Processes Scheduling in Cloud Environments with Fairness Metrics,” in IEEE International Conference on Research Challenges in Information Science (RCIS), 2013, pp. 1-10.

[4] C. Qiu, H. Shen, “Dynamic Demand Prediction and Allocation in Cloud Service Brokerage,” IEEE Transactions on Cloud Computing, 2019, DOI: 10.1109/TCC.2019.2913419.

[5] P. Lai, Q. He, G. Cui, F. Chen, M. Abdelrazek, J. Grundy, J. Hosking, Y. Yang, “Quality of Experience-Aware User Allocation in Edge Computing Systems: A Potential Game,” in IEEE International Conference on Distributed Computing Systems (ICDCS), 2020, pp. 223-233.

[6] L. J. Zhang, “EIC Editorial: Introduction to the Knowledge Areas of Services Computing,” IEEE Transactions on Services Computing, 2008, 1(2), pp. 62–74.

[7] L. Pan, B. An, S. Liu, L. Cui, “Nash Equilibrium and Decentralized Pricing for QoS Aware Service Composition in Cloud Computing Environments,” in IEEE International Conference on Web Services (ICWS), 2017, pp. 154-163.

[8] B. Jedari, M. D. Francesco, “Auction-based Cache Trading for Scalable Videos in Multi-Provider Heterogeneous Networks,” in IEEE Conference on Computer Communications (INFOCOM), 2019, pp. 1864-1872.

[9] F. Arena, G. Pau, “When edge computing meets IoT systems: Analysis of Case Studies,” China Communications, 2020, 17 (10), pp. 50-63.

[10] X. Wang, Z. Feng, S. Chen, K. Huang, “DKEM: A Distributed Knowledge Based Evolution Model for Service Ecosystem,” in 2018 IEEE International Conference on Web Services (ICWS), 2018, pp. 1-8.

[11] H. Takatsuka, S. Saiki, S. Matsumoto, M. Nakamura, “Developing Service Platform for Web Context-Aware Services Towards Self-Managing Ecosystem,” in International Conference on Service-Oriented Computing (ICSOC), 2014, pp. 270-280.

[12] H. Wang，J. Chen，N. Jiang, “A Study on Cooperation Evolution of Stakeholders in Service Ecosystem,” in 2019 3rd International Conference on Management Engineering, Software Engineering and Service Sciences (ICMSS), 2019, pp. 235-239.

[13] Y. Liu, Y. Fan, K. Huang, “Service Ecosystem Evolution and Controlling: A

Research Framework for the Effects of Dynamic Services," in 2013 International Conference on Service Sciences (ICSS), 2013, pp. 28-33.

[14] Z. Yong, Y. Fan, K. Huang, W. Tan, J. Zhang, "Time-Aware Service Recommendation for Mashup Creation in an Evolving Service Ecosystem," in 2014 IEEE International Conference on Web Services (ICWS), 2014, pp. 25-32.

[15] Z. Gao, Y. Fan, C. Wu, W. Tan, J. Zhang, "Service Recommendation from the Evolution of Composition Patterns," in 2017 IEEE International Conference on Services Computing (SCC), 2017, pp. 108-115.

[16] 郝予实，范玉顺,"服务系统中冷启动服务协作关系挖掘与预测," 清华大学学报（自然科学版），2019，59(11)，pp. 917-924.

[17] K. Huang, Y. Liu, S. Nepal, Y. Fan, S. Chen, W. Tan, "A Novel Equitable Trustworthy Mechanism for Service Recommendation in the Evolving Service Ecosystem，" in International Conference on Service-Oriented Computing (ICSOC), 2014, pp. 510-571.

[18] Q. Zhu, A. Zhou, Q. Sun, S. Wang, F. Yang, "FMSR: A Fairness-aware Mobile Service Recommendation Method Recommender Services," in IEEE International Conference on Web Services (ICWS), 2018, pp.171-178.

[19] 刘轶，范玉顺，黄科满，"全局视角下的 Web 服务系统模型及推荐策略," 计算机集成制造系统，2016，22 (1)，pp. 133-143.

[20] 夏博飞，范玉顺，黄科满，"Web 服务生态系统中消亡服务的预测方法," 计算机集成制造系统，2014，20 (8)， pp. 2060-2070.

[21] B. Zheng, L. Pan, D. Yuan, S. Liu, Y. Shi, L. Wang, "A Truthful Mechanism for Optimally Purchasing IaaS Instances and Scheduling Parallel Jobs in Service Clouds," in International Conference on Service-Oriented Computing (ICSOC)，2018，pp. 651-659.

[22] B. Zheng, L. Pan, S. Liu, L. Wang, "An Online Mechanism for Purchasing IaaS Instances and Scheduling Pleasingly Parallel Jobs in Cloud Computing Environments," in IEEE International Conference on Distributed Computing Systems (ICDCS), 2019, pp. 35-45.

[23] 刘楚波，"云用户效益优化的博弈论方法研究，" 湖南大学博士学位论文，2016.

[24] 李振宇，"云计算资源调度中的序列博弈模型，" 重庆大学硕士学位论文，2017.

[25] P. Lai, Q. He, G. Cui, X. Xia, M. Abdelrazek, F. Chen, J. Hosking, J. Grundy, Y. Yang, "Edge User Allocation with Dynamic Quality of Service," in International Conference on Service-Oriented Computing (ICSOC)，2019, pp. 86-101.

[26] Y. Zhu, S. Fu, J. Liu, Y. Cui, "Truthful Online Auction for Cloud Instance Subletting," in IEEE International Conference on Distributed Computing Systems (ICDCS), 2017, pp. 2466-2471.

[27] S. Yang, L. Pan, Q. Wang, S. Liu, “To Sell or Not To Sell: Trading Your Reserved Instances in Amazon EC2 Marketplace,” in IEEE International Conference on Distributed Computing Systems (ICDCS), 2018, pp. 939-948.

[28] S. Yang, L. Pan, S. Liu, “An Online Algorithm for Selling Your Reserved IaaS Instances in Amazon EC2 Marketplace,” in IEEE International Conference on Web Services (ICWS), 2019, pp. 296-303.

[29] M. Benioff, C. Adler, Behind the Cloud: The Untold Story of How Salesforce.com Went from Idea to Billion-Dollar Company-and Revolutionized an Industry. Jossey-Bass, San Francisco, CA, pp. 103-105, 2009.

[30] J. Weinman, “The Economics of Pay-per-use Pricing,” IEEE Cloud Computing, 5(5), pp. 99-107, 2018.

[31] M. Khodak, L. Zheng, A. S. Lan, C. Joe-Wong, M. Chiang,“Learning Cloud Dynamics to Optimize Spot Instance Bidding Strategies,” in IEEE Conference on Computer Communications (INFOCOM), 2018, pp. 2762-2770.

[32] “Amazon Elastic Compute Cloud (Amazon EC2),” Online Available: https://aws.amazon.com/ec2/.

[33] “Google Cloud Computing Services,” Online Available: https://cloud.google.com.

[34] “Microsoft Azure Cloud Computing Services,” Online Available: https://azure.microsoft.com/en-us.

[35] L. Zheng, C. Joe-Wong, C. W. Tan, M. Chiang, X. Wang, “How to Bid the Cloud”, in ACM Conference on Special Interest Group on Data Communication (SIGCOMM), 2015, pp. 71-84.

[36] S. Hou, W. Ni, S. Chen, S. Zhao, B. Cheng, J. Chen, “Real-time Optimization of Dynamic Speed Scaling for Distributed Data Centers,” IEEE Transactions on Network Science and Engineering, 7 (3), pp. 2090-2103, 2020.

[37] T. Pham, S. Ristov, T. Fahringer, “Performance and Behavior Characterization of Amazon EC2 Spot Instances,” in IEEE International Conference on Cloud Computing (CLOUD), 2018, pp. 73–81.

[38] H. Xu and B. Li, “Dynamic Cloud Pricing for Revenue Maximization,” IEEE Transactions on Cloud Computing, 1 (2), pp. 158-171, 2013.

[39] 张开元，桂小林，任德旺，李敬，吴杰，任东胜，“移动边缘网络中计算迁移与内容缓存研究综述，” 软件学报， 2019， 30 (8)，pp. 2491-2516.

[40] X. Xia, F. Chen, Q. He, G. Cui, P. Lai, M. Abdelrazek, J. Grundy, H. Jin, “Graph-Based Optimal Data Caching in Edge Computing,” in International Conference on Service-Oriented Computing (ICSOC)，2019, pp. 477-493.

[41] 徐意，“面向移动应用的边缘云服务迁移策略研究，” 华中科技大学硕士学位论文，2019.

[42] Q. Peng, Y. Xia, Z. Feng, J. Lee, C. Wu, X. Luo, W. Zheng, S. Pang, H. Liu, Y. Qin, P. Chen, “Mobility-Aware and Migration-Enabled Online Edge User Allocation in Mobile Edge Computing,” in IEEE International Conference on

Web Services (ICWS), 2019, pp. 91-98.

[43] H. Zhao, S. Deng, C. Zhang, W. Du, Q. He, J. Yin, “A Mobility-Aware Cross-edge Computation Offloading Framework for Partitionable Applications,” in IEEE International Conference on Web Services (ICWS), 2019, pp. 193-200.

[44] Q. He, G. Cui, X. Zhang, F. Chen, S. Deng, H. Jin, Y. Li, Y. Yang, “A Game-Theoretical Approach for User Allocation in Edge Computing Environment,” IEEE Transactions on Parallel and Distributed Systems, 21 (3), pp. 515-529, 2020.

[45] Z. Hong, W. Chen, H. Huang, S. Guo, Z. Zheng, “Multi-hop Cooperative Computation Offloading for Industrial IoT-Edge-Cloud Computing Environments,” IEEE Transactions on Parallel and Distributed Systems, 2019, 30(12), pp. 2759-2774.

[46] S. Ma, S. Guo, K. Wang, W. Jia, M. Guo, “A Cyclic Game for Joint Cooperation and Competition of Edge Resource Allocation,” in IEEE International Conference on Distributed Computing Systems (ICDCS), 2019, pp. 503-513.

[47] L. Qi, Y. Tang, W. Dou, J. Chen, “Combining Local Optimization and Enumeration for QoS-aware Web Service Composition,” in IEEE International Conference on Web Services (ICWS), 2010, pp. 34-41.

[48] L. Zeng, B. Benatallah, A. H. H. Ngu, M. Dumas, J. Kalagnanam, H. Chang, “QoS-aware Middleware for Web Services Composition,” IEEE Transactions on Software Engineering, 30 (5), pp. 311-327, 2004.

[49] J. Xu, Z. Zheng, M. R. Lyu, “Web Service Personalized Quality of Service Prediction via Reputation-based Matrix Factorization,” IEEE Transactions on Reliability, 65 (1), pp. 28–37, 2016.

[50] M. S. Saleem, C. Ding, X. Liu, C. H. Chi, “Personalized Decision-strategy based Web Service Selection using a Learning-to-Rank Algorithm,” IEEE Transactions on Services Computing, 8 (5), pp. 727-739, 2017.

[51] H. Wu, S. Deng, W. Li, M. Fu, J. Yin, A. Y. Zomaya,“Service Selection for Composition in Mobile Edge Computing Systems,” in IEEE International Conference on Web Services (ICWS), 2018, pp. 355-358.

[52] C. Powell, K. Miura, M. Munetomo, “Optimal Cloud Resource Selection Method Considering Hard and Soft Constraintsand Multiple Conflicting Objectives,” in IEEE International Conference on Cloud Computing (CLOUD), 2018, pp. 831-835.

[53] G. Chen, L. Xiao, J. Li, L. Zhou, “Dynamic Base Station Selection Strategies based on Location Fingerprinting,” in IEEE Asia-Pacific Conference on Antennas and Propagation (APCAP), 2015, pp. 474-477.

[54] V. K. Adhikari, Y. Guo, F. Hao, M. Varvello, “Unreeling Netflix: Understanding and Improving Multi-CDN Movie Delivery,” in IEEE Conference on Computer Communications (INFOCOM), 2012, pp. 1620-1628.

[55] G. Castelli, M. Mamei, A. Rosi, F. Zambonelli, “Engineering Pervasive Service Ecosystems: The Sapere Approach,” ACM Transactions on Autonomous and

Adaptive Systems, 10 (1), pp. 1-27, 2015.

[56] J. Yan, R. Kowalczyk, J. Lin, M. B. Chhetri, S. K. Goh, J. Zhang, “Autonomous Service Level Agreement Negotiation for Service Composition Provision,” Future Generation Computer Systems, 23 (6), pp. 748-759, 2007

[57] K. Kofler, I. U. Haq, and E. Schikuta, “User-Centric, Heuristic Optimization of Service Composition in Clouds,” in International Euro-Par Conference on Parallel Processing (EuroPar), 2010, pp.405–417.

[58] R. R. Meyer, “A Class of Nonlinear Integer Programs Solvable by a Single Linear Program,” SIAM Journal on Control and Optimization (SICON), 15 (6), pp. 935-946, 1976.

[59] D. E. Irwin, L. E. Grit, J. S. Chase, “Balancing Risk and Reward in a Market-based Task Service,” in IEEE International Symposium on High Performance Distributed Computing (HPDC), 2004, pp. 160-169.

[60] A. V. Zykina, “A Lexicographic Optimization Algorithm,” Automation and Remote Control, 65 (5), pp. 363–368, 2004.

[61] B. H. Korte, J. Vygen, “Combinatorial Optimization: Theory and Algorithms,” Algorithms and Combinatorics, 21 (5), 2006.

[62] L. Chen, S. Liu, B. Li, B. Li, “Scheduling Jobs across Geo-distributed Datacenters with Max-min Fairness,” in IEEE Conference on Computer Communications (INFOCOM), 2017, pp. 1-9.

[63] E. D. Andersen, K. D. Andersen, “The MOSEK Interior Point Optimizer for Linear Programming: An Implementation of the Homogeneous Algorithm,” Applied Optimization, vol. 33, pp. 197-232, 2000.

[64] “IBM Cplex Optimizer: High-Performance Mathematical Programming Solver,” Online Available: https://www-01.ibm.com/software/commerce/optimization/cplex-optimizer/.

[65] Z. Zheng, Y. Zhang, M. R. Lyu, “Investigating QoS of Real-World Web Services,” IEEE Transactions on Services Computing, vol. 7, no. 1, pp. 32-39, 2014.

[66] D. S. Linthicum, “The Evolution of Cloud Service Governance,” IEEE Cloud Computing, 2 (6), pp. 86-89, 2015.

[67] C. Wu, R. Buyya, K. Ramamohanarao, “Cloud Pricing Models: Taxonomy, Survey, and Interdisciplinary Challenges,” ACM Computing Surveys, 52 (6), pp. 108:1-108:36, 2019.

[68] S. Li, J. Huang, B. Cheng, “A Price-Incentive Resource Auction Mechanism Balancing the Interests between Users and Cloud Service Provider,” IEEE Transactions on Network and Service Management, 2020, DOI: 10.1109/TNSM.2020.3036989.

[69] A. Jin, W. Song, W. Zhuang, “Auction-based Resource Allocation for Sharing Cloudlets in Mobile Cloud Computing,” IEEE Transactionson Emerging Topics in Computing, 6 (1), pp. 45-57, 2018.

[70] S. Hosseinalipour, H. Dai, “A Two-stage Auction Mechanism for Cloud Resource Allocation,” IEEE Transactions on Cloud Computing, 2019, DOI: 10.1109/TCC.2019.2901785.

[71] L. Lu, J. Yu, Y. Zhu, M. Li, “A Double Auction Mechanism to Bridge Users’ Task Requirements and Providers’ Resources in Two-sided Cloud Markets,” IEEE Transactions on Parallel and Distributed Systems, 29 (4), pp. 720-733, 2018.

[72] X. Zhang, Z. Huang, C. Wu, Z. Li, F. C. M. Lau, “Online Auctions in IaaS Clouds: Welfare and Profit Maximization with Server Costs,” IEEE/ACM Transactions on Networking, 25 (2), pp. 1034-1047, 2017.

[73] C. Chen, W. Wang, B. Li, “Speculative Slot Reservation: Enforcing Service Isolation for Dependent Data-Parallel Computations,” in IEEE International Conference on Distributed Computing Systems (ICDCS), 2017, pp. 549-559.

[74] H. Liang, T. Xing, L. X. Cai, D. Huang, D. Peng, Y. Liu,“Adaptive Computing Resource Allocation for Mobile Cloud Computing,” International Journal of Distributed Sensor Networks, 9 (4), article no. 181426, pp. 1-14, 2013.

[75] C. Qiu, H. Shen, L. Chen, “Towards Green Cloud Computing: Demand Allocation and Pricing Policies for Cloud Service Brokerage,” IEEE Transactions on Big Data, 5 (2), pp. 238-251, 2019.

[76] “Pricing calculator - Configure and Estimate the Costs for Azure Products,” Online Available: https://azure.microsoft.com/en-us/pricing/calculator/.

[77] C. Wang, N. Nasiriani, G. Kesidis, B. Urgaonkar, Q. Wang, L. Y. Chen,A. Gupta, R. Birke, “Recouping Energy Costs from Cloud Tenants: Tenant Demand Response-aware Pricing Design,” in ACM InternationalConference on Future Energy Systems (e-Energy), 2015, pp. 141-150.

[78] C. Wang, B. Urgaonkar, G. Kesidis, A. Gupta, L. Y. Chen, R. Birke, “Effective Capacity Modulation as an Explicit Control Knob for Public Cloud Profitability,” ACM Transactions on Autonomous and AdaptiveSystems, 13 (1), pp. 2:1–2:25, 2018.

[79] M. Aldossary, K. Djemame, I. Alzamil, A. Kostopoulos, A. Dimakis, E. Agiatzidou, “Energy-aware Cost Prediction and Pricing of Virtual Machines in Cloud Computing Environments,” Future Generation Computer Systems, vol. 93 (4), pp. 442 - 459, 2019.

[80] C. Chen, W. Wang, B. Li, “Performance-aware Fair Scheduling: Exploiting Demand Elasticity of Data Analytics Kobs,” in IEEE International Conference on Computer Communications (INFOCOM), 2018, pp. 504-512.

[81] S. Venkataraman, Z. Yang, M. Franklin, B. Recht, I. Stoica, “Ernest: Efficient Performance Prediction for Large-scale Advanced Analytics,” in USENIX Symposium on Networked Systems Design and Implementation (NSDI), 2016, pp. 363-378.

[82] J. Bruck, Ching-Tien Ho, S. Kipnis, E. Upfal, D. Weathersby,“Efficient Algorithms for All-to-all Communications in Multiport Message-passing

Systems," IEEE Transactions on Parallel and Distributed Systems, 8 (11), pp. 114-1156, 1997.

[83] B. Moon, I. Fernando Vega Lopez, V. Immanuel, "Efficient Algorithms for Large-scale Temporal Aggregation," IEEE Transactions on Knowledge and Data Engineering, 15 (3), pp. 744-759, 2003.

[84] X. Zhang, Z. Huang, C. Wu, Z. Li, F. C. M. Lau, "Online Auctionsin IaaS Clouds: Welfare and Profit Maximization with Server Costs," IEEE/ACM Transactions on Networking, 25 (2), pp. 1034-1047, 2017.

[85] S. Boyd, L. Vandenberghe, Convex Optimization. Cambridge University Press, 2004.

[86] J. C. Gilbert, Numerical Optimization: Theoretical and Practical Aspects (Universitext), 2006.

[87] D. Zhao, X. Li, H. Ma, "Budget-Feasible Online Incentive Mechanisms for Crowdsourcing Tasks Truthfully," IEEE/ACM Transactions on Networking, 24 (2), pp. 647-661, 2016.

[88] S. Dobzinski, N. Nisan, M. Schapira, "Truthful Randomized Mechanisms for Combinatorial Auctions," in ACM Symposium on Theory of Computing (STOC), 2006, pp. 644-652.

[89] V. Guruswami, J. D. Hartline, A. R. Karlin, D. Kempe, C. Kenyon, F. McSherry, "On Profit-maximizing Envy-free Pricing," in Annual ACM-SIAM Symposium on Discrete Algorithms (SODA), 2005, pp. 1164-1173.

[90] E. Cortez, A. Bonde, A. Muzio, M. Russinovich, M. Fontoura, R. Bianchini, "Resource Central: Understanding and Predicting Workloads for Improved Resource Management in Large Cloud Platforms," in ACM Symposium on Operating Systems Principles (SOSP), 2017, pp.153-167.

[91] S. Kirkpatrick, C. Gelatt, M. Vecchi, "Optimization by Simulated Annealing," 1983, 220 (4598), pp. 671-680.

[92] M. Raja, V. Ghaderi, S. Sigg, "WiBot! In-vehicle Behaviour and Gesture Recognition using Wireless Network Edge," in IEEE International Conference on Distributed Computing Systems (ICDCS), 2018, pp. 376-387.

[93] T. Braud, F. H. Bijarbooneh, D. Chatzopoulos, P. Hui, "Future Networking Challenges: The Case of Mobile Augmented Reality," in IEEE International Conference on Distributed Computing Systems (ICDCS), 2017, pp. 1796-1807.

[94] M. Jarschel, D. Schlosser, S. Scheuring, T. Hobfeld, "An Evaluation of QoE in Cloud Gaming based on Subjective Tests," in International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing (IMIS), 2011, pp. 330-335.

[95] G. Ananthanarayanan, P. Bahl, P. Bodłk, K. Chintalapudi, M. Philipose,L. Ravindranath, S. Sinha, "Real-time Video Analytics: The Killerapp for Edge Computing," Computer, 50 (10), pp. 58-67, 2017.

[96] S. Choy, B. Wong, G. Simon, C. Rosenberg, "The Brewing Storm in Cloud

Gaming: A Measurement Study on Cloud to End-user Latency," in Annual Workshop on Network and Systems Support for Games (NetGames), 2012, pp. 1-6.

[97] A. Li, X. Yang, S. Kandula, M. Zhang, "CloudCmp: Comparing Public Cloud Providers," in ACM SIGCOMM Conference on Internet Measurement (IMC), 2010, pp. 1-14.

[98] Y. Chen, N. Zhang, Y. Zhang, X. Chen, W. Wu, X. S. Shen,"Energy Efficient Dynamic Offloading in Mobile Edge Computing for Internet of Things," IEEE Transactions on Cloud Computing, 2019, DOI:10.1109/TCC.2019.2898657.

[99] J. Huang, S. Li, Y. Chen, "Revenue-optimal Task Scheduling and Resource Management for IoT Batch Jobs in Mobile Edge Computing," Peer-to-Peer Networking and Applications, 13 (5), pp. 1776-1787, 2020.

[100] W. Shi, G. Pallis, Z. Xu, "Edge Computing," Proceedings of the IEEE, 107 (8), pp. 1474-1481, 2019.

[101] K. M. Sim, "Intelligent Resource Management in Intercloud, Fog, and Edge: Tutorial and New Directions," IEEE Transactions on Services Computing, 2020, DOI: 10.1109/TSC.2020.2975168.

[102] A. Galanopoulos, T. Salonidis, G. Iosifidis, "Cooperative Edge Computing of Data Analytics for the Internet of Things," IEEE Transactions on Cognitive Communications and Networking, 6 (4), pp. 1166-1179, 2020.

[103] T. X. Tran, A. Hajisami, P. Pandey, D. Pompili, "Collaborative Mobile Edge Computing in 5G Networks: New Paradigms, Scenarios, and Challenges," IEEE Communications Magazine, 55 (4), pp. 54-61,2017.

[104] Y. Wang, P. Li, L. Jiao, Z. Su, N. Cheng, X. S. Shen, P. Zhang, "A Data-driven Architecture for Personalized QoE Management in 5G Wireless Networks," IEEE Wireless Communications, 24 (1), pp.102-110, 2017.

[105] Y. Lin, H. Shen, "CloudFog: Leveraging Fog to Extend Cloud Gaming for Thin-client MMOG with High Quality of Service," IEEE Transactionson Parallel and Distributed Systems, 28 (2), pp. 431-445, 2017.

[106] X. Ma, A. Zhou, S. Zhang, S. Wang, "Cooperative Service Caching and Workload Scheduling in Mobile Edge Computing," in IEEE International Conference on Computer Communications (INFOCOM), 2020, pp. 2076-2085.

[107] W. Chen, D. Wang, K. Li, "Multi-user Multi-task Computation Offloading in Green Mobile Edge Cloud Computing," IEEE Transactionson Services Computing, 12 (5), pp. 726-738, 2019.

[108] L. Kuang, T. Gong, S. OuYang, H. Gao, S. Deng, "Offloading Decision Methods for Multiple Users with Structured Tasks in Edge Computing for Smart Cities," Future Generation Computer Systems, 105 (4), pp. 717-729.

[109] M. R. Garey, D. S. Johnson, "Computers and Intractability: A Guide to the Theory of NP-Completeness," Computers and Intractability, vol. 340, 1979.

[110] Y. Zeng, R. Zhang, "Energy-efficient UAV Communication with Trajectory Optimization," IEEE Transactions on Wireless Communications, 16 (6), pp.

3747-3760, 2017.

[111] R. Sun, Y. Wang, N. Cheng, L. Lyu, S. Zhang, H. Zhou, X. Shen, “QoE-driven Transmission-aware Cache Placement and Cooperative Beamforming Design in Cloud-RANs,” IEEE Transactions on Vehicular Technology, 69 (1), pp. 636-650, 2020.

[112] M. Alreshoodi, J. Woods, “Survey on QoE/QoS Correlation Models for Multimedia Services,” International Journal of Distributed and Parallel Systems, 4 (3), pp. 53-72, 2013.

[113] M. Fiedler, T. Hossfeld, P. TranGia, “A Generic Quantitative Relationship between Quality of Experience and Quality of Service,” IEEE Network, 24 (2), pp. 36-41, 2010.

[114] M. Hemmati, B. McCormick, S. Shirmohammadi, “QoE-aware Bandwidth Allocation for Video Traffic using Sigmoidal Programming,” IEEE MultiMedia, 24 (4), pp. 80-90, 2017.

[115] P. Hande, S. Zhang, M. Chiang, “Distributed Rate Allocation for Inelastic Flows,” IEEE/ACM Transactions on Networking, 15 (6), pp. 1240-1253, 2007.

[116] S. Shenker, “Fundamental Design Issues for the Future Internet,” IEEE Journal on Selected Areas in Communications, 13 (7), pp. 1176-1188, 1995.

[117] C. Liang, Y. He, F. R. Yu, N. Zhao, “Enhancing Video Rate Adaptation with Mobile Edge Computing and Caching in Software-Defined Mobile Networks,” IEEE Transactions on Wireless Communications, 17 (10), pp. 7013-7026, 2018

[118] S. Daryl, “Mobile Video Requires Performance and Measurement Standards,” Huawei Global Mobile Broadband Forum, 2015, Online Available: www. huawei.com/minisite/hwmbbf15/img/mvponline.pdf.

[119] M. J. Osborne, A. Rubinstein, “A Course in Game Theory,” MIT Press, 1994.

[120] D. Monderer, L. S. Shapley, “Potential Games,” Games & Economic Behavior, 14 (1), pp. 124-143, 1996.

[121] J. R. Marden, G. Arslan, J. S. Shamma, “Cooperative Control and Potential Games,” IEEE Transactions on Systems, Man and Cybernetics - Part B (Cybernetics), 39 (6), pp. 1393-1407, 2009.

[122] P. Lai, Q. He, M. Abdelrazek, F. Chen, J. Hosking, J. Grundy, Y. Yang, “Optimal Edge User Allocation in Edge computing with Variable Sized Vector Bin Packing,” in International Conference on Service-Oriented Computing (ICSOC), 2018, pp. 230-245.

[123] X. Chen, “Decentralized Computation Offloading Game for Mobile Cloud Computing,” IEEE Transactions on Parallel and Distributed Systems, 26 (4), pp. 974-983, 2015.

[124] X. Chen, L. Jiao, W. Li, X. Fu, “Efficient Multi-user Computation Offloading for Mobile-edge Cloud Computing,” IEEE/ACM Transactions on Networking, 24 (5), pp. 2795-2808, 2016.

[125] E. Tanghe, W. Joseph, L. Verloock, L. Martens, H. Capoen, K. V. Her-wegen,

W. Vantomme, “The Industrial Indoor Channel: Large-scale and Temporal Fading at 900, 2400, and 5200 MHz,” IEEE Transactions on Wireless Communications, 7 (7), pp. 2740-2751, 2008.

[126] H. Yao, C. Bai, M. Xiong, D. Zeng, Z. Fu, “Heterogeneous Cloudlet Deployment and User-Cloudlet Association toward Cost Effective Fog Computing,” Concurrency & Computation Practice & Experience, 2016, DOI: 10.1002/cpe. 3975.

致谢

时间如白驹过隙，我已经在北京邮电大学的校园里度过了七年的春秋。特别的是，三年硕士研究生生涯中的成长与收获将成为我心中的难忘记忆和宝贵精神财富。期间，无论是学业成就的进步，抑或是人生阅历的积淀，都离不开各位恩师们的悉心指导和关怀，漫漫学途，师恩永记！与此同时，我也十分感激于各位同窗好友、家人的一路支持和鼓励；正是您们的陪伴与激励，使得我始终保持充沛的信心和勇气来不断地进步和前进。

首先，衷心感谢我的硕士研究生导师程渤教授，在科学研究方面为我创造了良好宽松的科研学习环境，始终支持我发掘自身的科研兴趣、发挥学术专长。在为期三年的硕士研究生经历中，我所取得的多项学术成果都离不开程渤教授的鼓励和支持。作为导师，程老师具有崇高的师德，始终向学生提供无微不至的指导和帮助，无私地向我传授其从教以来所积累的丰富科研经验和知识，这将成为我从事科研道路上的宝贵财富。程渤教授为人正直，治学严谨，待人宽和，是我不断学习与效仿的榜样和楷模。

其次，特别感谢现中国石油大学（北京）计算机科学与技术系主任黄霁崴教授！从 2016 年起，我便在网络与交换技术国家重点实验室（北京邮电大学），在黄霁崴教授的指导下参与科研项目。在加入中国石油大学（北京）后，黄老师一如既往地关心我的科研工作和生活，在科学研究方面极为耐心地启发我，向我提供十分细致的指导意见。每当我在科研中遇到瓶颈时，黄老师总会尽心尽力地引导、为我答疑解惑，展现出他广博的视野和学识。一路走来，我所取得的一项项学术成果与奖励都凝聚着黄霁崴教授的辛苦付出，师恩如海，令我不胜感激！黄霁崴教授是我从事科学研究的启蒙导师，引领我步入科学研究的神圣殿堂，他严格的治学操守、敏锐的洞察能力、严谨的思维逻辑都在不经意间深刻地影响着我，使我终生受益。

然后，十分感谢山东大学软件学院的刘士军教授、潘丽副教授，您们团队的研究方向与我的科研兴趣十分相近，您们所取得的科研成果为我自身展开科研工作提供了很有价值的启发。每次与您们交流科研工作，总是令我受益匪浅！

同时，十分感谢中国石油大学（华东）计算机科学与技术学院的徐九韵教授。在第十届中国计算机学会服务计算学术会议（NCSC 2019）期间，我有幸与您结识，您向我讲授自身多年积累下来的科研经验，解开了我在科研工作中所遇到的瓶颈和困惑，由衷感谢！

感谢美国弗尼吉亚大学 Haiying (Helen) Shen 副教授。在 IEEE 世界服务大会（IEEE SERVICES 2019）上，我与您结下良好的情谊；期间，您耐心地解答我的科研疑惑，向我介绍您的多年科研经验，鼓励我做出更高水平的科研成果。从您团队所发表的学术论文中，我得到了诸多启发，并学习到了很多科研方法和问

题剖析技巧。再者，也衷心地感谢美国韦恩州立大学 Shiyong Lv 教授，英国埃克塞特大学 Geyong Min 教授和 Jia Hu 博士，感谢您们无私地与我分享了很多科研工作经验！

十分感谢同实验室的刘轩、王盟、牛梦、侯守璐、王昭宁等博士师兄师姐，是他（她）们一直在科研道路上为我答疑解惑、向我提供宝贵的科研经验。再者，很感谢国家重点研发计划（No.2018YFB1003804）项目组下的赵亚茹、韩佳乐两位同窗好友，同你们一起交流讨论,让我积累了诸多科研项目经验！也十分感谢湖南大学的杨浩、李昊两位博士师兄，有幸在 IEEE ISPA 2017 学术会议上与你们结识，在为期三年的硕士研究生经历中，我也时常得到你们的提点和帮助，让我在科研工作中少了些许疑惑和困顿！

特别地，由衷地感激于我的父母和家人！一直以来，您们给予我无微不至的关心和爱，始终坚定地支持我在科学研究的海洋中驰骋。您们一直是我能股全力投入的最坚强后盾，永远是我前进的动力和不断奋斗的源泉！

最后，真诚地感谢各位评委能够在百忙中抽出宝贵的时间来评阅我的硕士学位论文，并且提供宝贵的指导意见！

李松远

2021 年 5 月 31 日于北京

个人简历、在学期间发表的学术论文与研究成果

个人简历

2014 年 9 月考入北京邮电大学计算机学院，专业计算机科学与技术，2018 年 6 月本科毕业并获得工学学士学位。

2018 年 9 月免试进入北京邮电大学网络与交换技术国家重点实验室攻读工学硕士学位至今，专业计算机科学与技术。

攻读学位期间发表及在审的学术论文

[1] **Songyuan Li**, Jiwei Huang, Bo Cheng. Resource Pricing and Demand Allocation for Revenue Maximization in IaaS Clouds: A Market-Oriented Approach. ***IEEE Transactions on Network and Service Management (TNSM)***.（**CCF 推荐 C 类期刊；中科院 2 区、小类 1 区**，近三年影响因子：3.949；SCI、EI 刊源，已录用，DOI：10.1109/TNSM.2021.3085519）

[2] **Songyuan Li**, Jiwei Huang, Bo Cheng. A Price-Incentive Resource Auction Mechanism Balancing the Interests Between Users and Cloud Service Provider. ***IEEE Transactions on Network and Service Management (TNSM)***.（**CCF 推荐 C 类期刊；中科院 2 区、小类 1 区**，近三年影响因子：3.949；EI 收录，检索号：20204709518052；SCI 刊源，已录用，DOI：10.1109/TNSM.2020.3036989）

[3] **Songyuan Li**, Jiwei Huang, Bo Cheng, Lizhen Cui, Yuliang Shi. FASS: A Fairness-Aware Approach for Concurrent Service Selection with Constraints. ***The 2019 IEEE International Conference on Web Services (IEEE ICWS 2019)***, Milan, Italy, pp. 255-259, July 2019.（**CCF 推荐 B 类会议**，EI 收录，检索号：20194007491243）

[4] Jiwei Huang, **Songyuan Li**, Ying Chen. Revenue-Optimal Task Scheduling and Resource Management for IoT Batch Jobs in Mobile Edge Computing. ***Peer-to-Peer Networking and Applications (PPNA)***, vol. 13, no. 5, pp. 1776–1787, September 2020. （**CCF 推荐 C 类期刊；中科院 3 区**，近三年影响因子：2.235；EI 收录，检索号：20201208313908；SCI 收录，检索号：000562295700002）

[5] **Songyuan Li**, Jiwei Huang, Bo Cheng. QoE-DEER: A QoE-Aware Decentralized Resource Allocation Scheme for Edge Computing. ***IEEE Transactions on***

Cognitive Communications and Networking (TCCN).（**中科院 2 区**，近三年影响因子：4.574；SCI、EI 刊源，初审）

专利申请情况

[1] 程渤，赵帅，**李松远**，陈俊亮. QoE 感知的分布式边缘任务调度和资源管理方法及系统. 专利申请号：2021105114155.

参与的科研项目

[1] 国家重点研发计划“面向服务的群智化生态化软件开发方法与环境”，课题 4：“人-机-服务”协同学习的软件生态系统自适应机制研究（No.2018YFB1003804）

[2] 国家自然科学基金面上项目“基于边缘计算架构的物联网服务系统性能评价与优化”（No. 61972414）

[3] 国家自然科学基金青年项目“面向大规模动态服务环境的 QoS 评价方法研究”（No. 61502043）

[4] 北京市自然科学基金面上项目“农产品质量追溯的形式化建模理论和方法研究”（No. 4202066）

校外学术兼职

[1] 国际会议程序委员会委员：

- 2021 International Conference on Computer Engineering and Artificial Intelligence (ICCEAI 2021)

[2] 国际期刊论文审稿人：

- IEEE Access,
- Scientific Programming,
- Behaviour & Information Technology

[3] 国际会议论文审稿人：

- 2020 IEEE 92nd Vehicular Technology Conference (IEEE VTC2020-Fall),
- 2020 EAI 16th International Conference on Collaborative Computing (EAI CollaborateCom 2020)

获奖情况

[1] 国家奖学金，中华人民共和国教育部，2019 年
[2] 北京市优秀研究生毕业生，北京市教育委员会，2021 年
[3] 北京邮电大学优秀研究生毕业生，北京邮电大学，2021 年
[4] 一等学业奖学金，北京邮电大学，2018 年、2019 年、2020 年
[5] 网络与交换技术国家重点实验室优秀研究生，北京邮电大学，2019 年