浙江工业大学

本科毕业设计（论文、创作）

题目：基于非正交多址接入的边缘计算算法设计与实现

作者姓名 蔡嘉丽

指导教师 吴远 教授

专业班级 通信 1504

学　　院 信息工程学院

提交日期 2019 年 6 月 10 日

Dissertation Submitted to Zhejiang University of Technology for the Degree of Bachelor

Design and implementation of edge computing algorithm based on non-orthogonal multiple access

Student: Jiali Cai

Advisor: Yuan Wu

College of Information Engineering

Zhejiang University of Technology

June 2019

浙江工业大学

本科生毕业设计(论文、创作)诚信承诺书

本人慎重承诺和声明：

1. 本人在毕业设计（论文、创作）撰写过程中，严格遵守学校有关规定，恪守学术规范，所呈交的毕业设计（论文、创作）是在指导教师指导下独立完成的；

2. 毕业设计（论文、创作）中无抄袭、剽窃或不正当引用他人学术观点、思想和学术成果，无虚构、篡改试验结果、统计资料、伪造数据和运算程序等情况；

3. 若有违反学术纪律的行为，本人愿意承担一切责任，并接受学校按有关规定给予的处理。

学生（签名）：

年　　月　　日

浙 江 工 业 大 学

本科生毕业设计（论文、创作）任务书

专 业 通信工程 班 级 通信 1504 班 学生姓名/学号 蔡嘉丽/201503090401

一、设计（论文、创作）题目： **基于非正交多址接入的边缘计算算法设计与实现**

二、主要任务与目标：

基于非正交多址接入的边缘计算系统具有高性能、低延迟和高带宽电信级服务环境等优势。通过系统分析，利用优化非正交多址接入的传输资源和移动边缘计算任务量分配，实现充分利用系统资源以及提升服务质量。

三、主要内容与基本要求：

1.完成系统模型建立，建模优化问题 2.针对优化问题，完成算法设计和算法的软件实现 3. 撰写毕业论文和提交相关算法设计、仿真代码与仿真图形等

四、计划进度：

2019.1.1~3.1 收集相关资料文献，学习相关通信、问题优化基础知识；完成外文翻译、文献综述；熟悉课题，做好开题准备，有初步设计方案；

2019.3.2~3.15 完成开题报告，参加开题交流；

2019.3.16~4.30 完成系统建模，有初步解决方案，接受中期检查；

2019.5.1~5.31 算法仿真与优化，有完整的解决方案，撰写毕业论文初稿；

2019.6.1~6.20 论文修改，毕业答辩，提交相关文档资料。

五、主要参考文献：

[1]施巍松等，“边缘计算：现状与展望”，计算机研究与发展，DOI：10.7544/issn1000-1239.2019. [2]N. Abbs, et. al. “Mobile Edge Computing: A Survey,” IEEE Internet of Things Journal, vol. 5, no. 1, pp. 450-465, Feb. 2018.[3] Y. Wu, te. al., “NOMA Assisted Multi-Access Mobile Edge Computing: A Joint Optimization of Computation Offloading and Time Allocation,” IEEE Trans. on Vehicular Tech., vol. 67, no. 12, pp. 12244-12258, Dec. 2018.[4] Y Zhang, et al. “Energy-efficient transmission design in non-orthogonal multiple access,” IEEE Trans. on Vehicular Tech., vol. 66, no. 3, pp. 2852-2857, Mar. 2017.

任务书下发日期 2018 年 12 月 20 日

设计（论文、创作）工作自 2018 年 12 月 20 日 至 2019 年 6 月 20 日

设计（论文、创作）指导教师 _____

_____

系主任（专业负责人）_____

主管院长 _____

基于非正交多址接入的边缘计算算法设计与实现

摘　要

随着移动通信技术的发展，延迟高、安全漏洞、数据传输之后等问题愈加凸显。移动边缘计算通过将云计算能力部署到无线接入网的边缘，成为解决上述问题的关键技术。本文旨在利用非正交多址技术（NOMA），优化移动用户的任务量和非正交多址接入传输时间，实现移动用户将任务迁移到边缘服务器的能量消耗最小。同时，在多任务多服务器的场景下，找到分配任务（或者说分配服务器）的方案，从而进一步最小化总能量消耗。

本文首先假设不同任务分配到不同服务器的分配方案已经给定，在这情况下，联合优化移动用户的任务量和非正交多址接入传输时间，计算出用户完成所有任务所需的最小能耗。其次，本文采用基于交换的模拟退火算法（SA 算法），找到总能量消耗最小的分配方案。最后，通过 MATLAB 仿真以及全排列的大量数值结果验证了本文提出算法的有效性。

与传统云计算相比，边缘计算能减少完成计算任务的延迟，提高资源利用效率，满足用户体验。且基于非正交多址接入的计算迁移相较于传统的频分多址（FDMA）可以减少整体延迟。

关键词：非正交多址技术，边缘计算，计算迁移，最佳匹配

Design and implementation of edge computing algorithm based on non-orthogonal multiple access

ABSTRACT

With the development of mobile communication technology, problems such as high delay, security vulnerability and data transmission become more and more prominent. Mobile edge computing becomes the key technology to solve the above problem by deploying cloud computing power to the edge of wireless access network. This paper aims to optimize the task amount and NOMA transmission time of mobile users by using non-orthogonal multiple access technology (NOMA), and realize the minimum energy consumption of mobile users to migrate the task to the edge server. At the same time, in the multi-task multi-server scenario, we find the solution of allocating tasks (or allocating servers) to further minimize the total energy consumption.

This paper first assumes that the allocation scheme of different tasks assigned to different servers has been given. In this case, the mobile user's task amount and NOMA transmission time are jointly optimized to calculate the minimum energy consumption required for the user to complete all tasks. Secondly, based on the switching, simulated annealing algorithm (SA algorithm) is adopted in this paper to find the allocation scheme with the minimum total energy consumption. Finally, MATLAB simulation and a large number of numerical results of full array verify the effectiveness of the proposed algorithm.

Compared with traditional cloud computing, edge computing can reduce the delay of completing computing tasks, improve the efficiency of resource utilization and meet the user experience. And noma-based computing migration can reduce overall latency compared to traditional frequency division multiple access (FDMA).

Key Words: Non-orthogonal multiple access technology, edge computing, computational migration, optimal matching

目　　录

第 1 章　绪　论

1.1　研究意义

1.1.1　课题背景

如今，物联网（Internet of Things，IoT）快速发展，人与人之间的通信方式也发生了极大的变化，移动通信网络逐渐步入 5G 时代。在日常生活中，终端设备输了增加，不仅会产生庞大的数据量，而且新兴的大量应用程序，对数据处理的实时性、安全性，以及传输速度也提出了极高要求。

随着物联网的发展，传统的集中式云计算技术，远离终端设备，已经难以应付需求，尤其是处理计算密集型任务。网络带宽有限，逐渐成为云计算的缺陷，而且云计算的计算性能，使得传送、处理数据需要一定时间，造成极大延迟，且设备产生的数据传输，由数据中心集中保存，存在隐私数据泄露，易受攻击、易丢失等隐患，此外，云数据中心的能耗问题也是巨大隐患。云计算的固有缺陷，使得移动云计算（Mobile Cloud Computing ，MCC）发展受限，难以满足用户体验。而移动边缘计算（Mobile Edge Computing ，MEC）通过将云计算能力部署到网络边缘，接近终端设备，是解决上述问题的最有潜力的方案[1]。

正如表 1-1 所示，边缘计算系统与云计算系统在计算服务器、与终端用户的距离、延迟等方面存在显著差异。

表 1-1　MEC 和 MCC 对比

	MEC	MCC
服务器	小型数据中心	大型数据中心
层次结构	2 层	3 层
部署	需要轻量级配置和规划	需要复杂的配置和规划
与终端用户距离	近	远
应用程序	延迟敏感和计算密集型	容忍延迟和计算密集型

边缘计算可以衍生为三个层次结构：云、边缘服务器和移动设备，如图 1-1 所示。移动用户通过适当地将部分计算任务，迁移到附近的边缘服务器，能减少完成计算任务的延迟，降低总能量消耗。

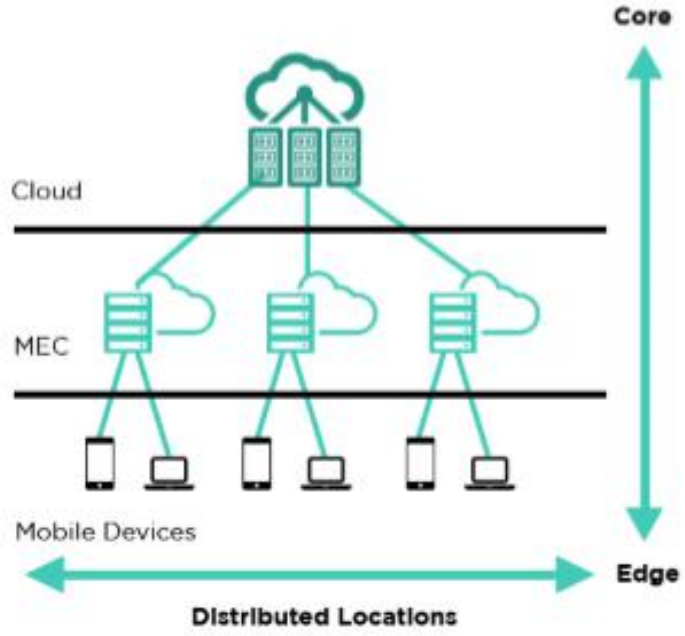

图 1-1　边缘计算三层架构

来源：参考文献[1]

1.1.2　边缘计算优势

与云计算相比，边缘计算具有延迟低、移动设备节能、支持上下文感知计算、增强移动应用程序隐私和安全性等明显优势。

1）低延迟：通常，延迟取决于三个因素，传播距离、计算能力和数据速率。对于边缘计算来说，边缘服务器接近终端用户，因此传播距离短。且现代移动设备功能强大，足以应付计算需求。数据只在服务器和移动设备之间传输，也减少了流量控制、路由等其他通信延迟。

2）节能：移动设备能量有限，相较于本地处理，通过将计算密集型任务迁移到边缘服务器，能大大降低能耗。

3）上下文感知：边缘计算能利用边缘设备的邻近性，实时追踪和预测终端用户的行为、位置和环境等信息。

4）隐私性和安全性：相较于传统云计算集中式数据存储，边缘计算的边缘设备可以是私有的，不易于数据泄露或丢失，且分布式存储也极大的增强了信息安全。

1.1.3 应用场景

由于边缘计算存在的天然优势，边缘计算成为了 5G 系统的关键技术之一[2]，已经在许多场景得到应用，例如物联网[3]-[4]、车辆网络[5,6]、软件定义网络[7,8]，此外，边缘计算也将应用与许多新兴场景，如视频分析[9,10]、医疗保健[11,12]，以及环境异常监测[13]等科学研究方面。

具体来说，边缘计算可以应用于以下方面：

1）视频分析

如今，为了建设智慧城市、平安城市，大量传感器安装在各个角落。共享单车、滴滴打车等产业也发展迅速，为了保障公共安全，部署了大量摄像头，视频分析技术变得尤为关键。这也导致了数以万计的数据需要处理，显然，大部分摄像头难以满足计算功能，而基于边缘计算的视频分析系统，能够靠近视频产生源，对视频内容分析判断，在追踪嫌犯时，甚至能和周围其他边缘设备协作，实时处理。基于边缘计算，能大大提高系统效率。

2）车辆联网

随着物联网、机器学习等技术的发展，车辆不再局限于代步的交通工具，逐渐变得智能。而边缘计算的兴起，更加推动了智能车辆的发展。通过边缘计算技术，车辆信息能够及时发送到边缘服务器，实现实时定位、诊断车辆异常行为，甚至有助于自动驾驶技术。且车与车之间的信息反馈，能够实时更新路况等。

3）虚拟现实/增强现实

应用边缘计算的虚拟现实（Virtual Reality，VR)和增强现实 (Augment Reality，AR)技术的应用，极大的满足了用户体验。例如图片渲染、人脸识别等，将 VR/AR 的计算任务迁移到边缘服务器，能够降低时延，满足其实时性、容量限制等要求。

4）智能家居

提升了安全性与舒适性，智能家居受到了大众追捧。随着边缘计算的发展，智能家居系统也得到进一步发展。其部署的大量湿度、温度传感器，照明、安防系统等等能利用边缘计算系统，实时监控检测异常，并且达到节能效果。此外，边缘计算能将家庭数据传输到内部网关处理，保障了用户隐私性。

5）海洋监测

科学家们正在研究如何应对任何海洋灾难性事件，并提前了解气候变化。这有助于

迅速做出反应并减轻压力，以防止出现任何灾难性情况。部署在海洋某处的传感器传输大量的数据，这需要大量的计算资源。数据由云处理，可能会导致传输延迟，难以实时预测。在这种情况下，边缘计算可以起到至关重要的作用，防止传感器数据传输中的任何数据丢失或延迟。

边缘计算前途广阔，除上述所列之外，在其他应用程序方面也有无限可能，因此边缘计算是很有潜力的技术，5G 时代来临，研究边缘计算也是顺应时代的结果。

1.2 边缘计算研究现状与成果

1.2.1 国内外研究现状

现阶段，对于边缘计算的研究如火如荼地开展，由于其相较于云计算的巨大优势，边缘计算已经在许多应用在许多场景。边缘计算源于 1998 年，阿卡迈公司提出的内容分发网络[14]（Content Delivery Network，CDN）。在 2009 年，Satyanarayanan 等人提出了 “微云”（Cloudlet） [15]的概念。上述技术旨在将云计算的功能下行到边缘设备来降低延迟，之后，为了解决计算负载和数据传输带宽的问题，业内研究人员开始在边缘设备增加数据处理的功能，即雾计算和移动边缘计算。2013 年， “edge computing”一词，由美国学者 Ryan LaMothe 首次提出[16]。由于其优势，迅速引起了学术界和工业界的密切关注，各种国际会议开始组织探讨边缘计算，2014 年，欧洲电信标准化协会（ETSI）提出移动边缘计算术语的标准化，致力于将边缘计算研究变得更加规范。2016 年，ACM 和 IEEE 联合举办了全球首个以边缘计算为主题的科研学术会议，之后，边缘计算的相关国际会议络绎不绝。在 2017 年 3 月，移动边缘计算行业规范工作组正式更名为多接入边缘计算 （multi-access edge computing, MEC），此后，对边缘计算的研究变得更加规范、标准。

边缘计算在国内也得到了迅速发展。2016 年 11 月，华为、中国科学院沈阳自动化研究所、中国信息通信研究院、英特尔、ARM 等在北京成立了边缘计算产业联盟（edge computing consortium），致力于推动“政产学研用” 各方产业资源合作，引领边缘计算产业的健康可持续发展。2017 年 5 月，首届中国边缘计算技术研讨会在合肥开幕。8 月，成立了中国自动化学会边缘计算专委会，这也预示着边缘计算的发展已经得到了专业学会的认可和推动。之后，边缘计算开始进入大众视线，被人熟知，成为了移动通信关键技术。2018 年，边缘计算进入了大众视野，10 月 30 日，中国移动成立了

边缘计算开放实验室，促进了各行业对边缘计算的合作研究。如今，研究边缘计算的参与者数量剧增，成果丰硕，边缘计算也就此成为了热门话题。

1.2.2 研究方向

本节介绍典型边缘计算系统的计算/通信的系统模型，为边缘计算的研究提供了诸多方向。

1) 计算任务模型

虽然为计算任务建立精确的模型是非常复杂的，但是也有一些合理简单的模型分别为二进制和部分计算迁移。这两种模型在已有的边缘计算文献中得到了广泛的应用

(1)二进制迁移任务模型：适用于数据难以区分或者相对简单的任务，可以选择整体在本地移动设备执行，或者迁移到边缘服务器。

(2)部分迁移任务模型：适用于多过程的计算任务，程序分为两部分，一部分在本地移动设备执行，另一部分迁移到边缘服务器执行 。

2) 通信模型

(1)无线比特管模型：为了便于处理，具有恒定速率或随机速率的比特管模型较为粗糙，而边缘计算需要尽可能的减少延迟，还需考虑无线传输特性以及高效的空中接口

(2)无线信道衰落模型：边缘计算在任务迁移时，需要考虑信道增益对传输延迟的影响，需要选择在有助于减少延迟或质量较好的信道进行负载迁移，因此需要对联合考虑迁移和无线传输来设计模型。

3) 回程链路

在边缘计算系统中，通信通常发生在移动设备和边缘服务器之间，而边缘计算系统属于小型数据中心，通过回程链路，可以访问远程数据中心来进一步将计算任务迁移至其他边缘服务器或者大型数据中心。

1.2.3 相关成果

近年来，边缘计算已经引起了业界和学术界的重视，诸多学者对边缘计算的含义、体系结构[17] 、计算平台，运用的关键技术、计算模型等进行了大量理论研究。如今，边缘计算技术尚未成熟，需要研究的方向丰富，当下，本文仅从边缘计算的任务迁移方

法考虑，以最小化总能量消耗的角度，设计算法，旨在实现单用户不同任务与不同服务器的最佳任务迁移方案。本文基于非正交多址接入传输，研究边缘计算算法。具体的，与本文研究密切相关学术成果的回顾如下。

Chen 等人研究了多用户迁移问题，并将分布式迁移决策问题作为多用户计算迁移博弈[18]。Wang 等人考虑到用户在执行迁移计算任务时的协同信道，提出了一个迁移计算和干扰管理的集成框架[19]。Tan 等人研究了一种具有边缘计算和缓存功能的支持虚拟全双工的小单元网络[20]。此外也有学者提出了一种能量有效的方案[21]，用于对延迟敏感的任务迁移。Chen 等人考虑了一个通用的多用户移动云计算系统[22]，每个用户都有多个独立的任务，旨在最大限度地降低边缘计算中所有用户的能源、计算和延迟的总成本。He 等人利用深度强化学习方法研究联网车辆的集成网络、缓存和计算[23]。Chen 等人研究了一个具有代表性的用户对多个基站的计算迁移[24]，提出了一种基于深度 Q 网络的迁移优化算法。也有对于反荷载方案的能量效率计算[25,26] 的研究。基于拍卖理论，作者提出了一种基于激励的迁移计算方法[27]。利用最新的先进多址边缘计算，可以进一步提高 MUs 的计算能力，提高计算迁移的效率[28]。

边缘计算已经引起了大量的研究，其中，基于非正交多址技术（NOMA）的多址边缘计算，能进一步减少延迟，满足计算要求。非正交多址接入（NOMA）[29-31]，允许一组 MUs 同时使用相同的资源块（RB），并进一步使用连续干扰消除（SIC）来减轻 MUs 的共信道干扰，已被设想为第五代（5g）蜂窝系统的一种使能空中接口技术。要从非正交多址接入技术获益，需要进行适当的资源管理，以减轻共享同一资源块时的干扰。

对非正交多址接入研究的相关成果回顾如下，Zhu 等人研究了不同性能标准下，如总和率和公平性等[32]，非正交多址接入技术的不同功率分配策略。为了提高非正交多址接入的能源效率，也有学者提出了不同的资源分配方案[33,34]。对于非正交多址接入，作者提出了一个联合用户调度和功率分配的方案[35]，旨在最大限度地减少总能耗。考虑到多信道情况，一些研究对非正交多址接入网络的联合子信道分配和用户调度进行了研究[36–38]。张等人提出了一种联合子载波分配和功率分配的方法来实现非正交多址接入的安全传输[39]。Wu 等人研究了非正交多址接入中继辅助网络的联合功率分配和业务调度问题[40]。此外，也有学者提出了一种将正交频分多址接入和多载波码分多址接入相结合的混合非正交多址接入传输方案[41]。特别的，最近一些人在研究如何利用非正交多址接入开发边缘计算，分析表明了利用非正交多址接入技术降低边缘计算延迟和能耗的好处[42]。Wang 等人研究了用户通过非正交多址接入和波束形成将其计算任务分配给多个可

信的帮助用户的协同迁移计算[43]。

多址边缘计算能实现边缘计算的部分计算迁移，即使得一个移动用户将需要完成的计算任务划分多个部分，同时利用多个无线接入，将部分工作负载同时迁移到多个不同的边缘服务器，能进一步提高计算效率，减少计算总延迟以及能量消耗。

1.2.4 本文成果

对于支持非正交多址接入的边缘计算，能极大地减少移动用户完成计算任务的延迟，因此需要考虑用户传输时间以及分配的迁移负载工作量来最大化非正交多址接入的效益，在本文中，主要工作如下，首先本文研究了单用户多任务的最小化能量消耗的方案，对优化任务迁移量和非正交多址接入传输时间进行优化，目标是最小化移动用户完成任务计算的总能量，虽然联合优化问题具有非凸性，但本文通过两步来实现上述算法。第一步优化移动用户迁移工作负载，第二步利用最优的迁移工作负载，确定最佳传输时间。通过较小的步长线性改变非正交多址接入传输时间，来实现完成计算要求的最小能量消耗。

接下来，通过上述算法，本文研究基于交换的模拟退火算法，找到不同任务与不同服务器之间的最佳分配方案。第一步，初始化一组分配方案，利用上述算法，求得当前情况下的最小能耗。第二步，随机选择两个任务，交换对应服务器，得到一组新的分配方案以及最小能耗。通过比较两者结果，在一定条件下，选择是否接受新的分配方案，通过重复上述步骤，不断迭代至满足终止条件，最终能输出最优解，即总能量消耗最小的分配方案。

最后，为了验证算法的正确性，本文通过 MATLAB 进行算法仿真，得到大量数值结果，也充分证明了其优势。

1.3 本章小结

在本章节中，首先第一小节，介绍了选题的意义。本文结合了课题的背景意义，分析了边缘计算的主要优势：有延迟低、移动设备节能、支持上下文感知计算、增强移动应用程序隐私和安全性等，且边缘计算的应用场景丰富，是一项非常有潜力的技术[44,45]，因此结合个人兴趣，选择了这个课题研究。

其次，为接下来具体的研究内容进行了铺垫。在第二小节中，介绍边缘计算的发展历程，虽然其潜力无限，但完全成熟尚需时日，从而引出边缘计算系统诸多研究方向中

与本文相关的模型设计。之后，介绍了前辈们发表的与本文相关的理论成果。在后一小节中，简单介绍了本文研究内容。

本文剩余内容安排如下。

第二章，本文将具体提出单用户多任务多服务器的边缘计算算法思路，以及系统模型建立过程，之后，下一小节中的如何找到分配任务（或者说分配服务器）的方案，从而进一步最小化总能量消耗。

第三章，将理论应用于实践，通过 MATLAB 仿真，实现算法以及验证其有效性。

第四章，总结。首先总结了全文的主要工作，并给出结论。同时，也针对本文存在的不足之处，对未来的发展方向提出展望。

第 2 章　系统建模与算法设计

2.1　系统建模与问题优化

在这一章中，将详细介绍本文的算法，对基于非正交多址接入传输的单用户多址计算进行研究，目标是最小化移动用户（MU）完成计算任务的总能量，从计算最小传输功率来看，与三个因素有关，因此把问题分为三个，即移动用户如何决定：1）每个任务 i 的分流计算量 s_i；2）将计算任务量 $\{s_1,s_2,...,s_I\}$ 传送到各个边缘服务器的传输时间 t；3）移动用户的任务如何一对一匹配服务器。

2.1.1　系统建模

为了实现算法，本文用分步解决上述问题。为了便于建模，首先考虑任务和边缘服务器的对应关系给定的情况，解决问题 1）和 2），即一个移动用户，通过优化 s_i 和传输时间 t，最大限度地减少移动用户完成所有任务的总能量。本文假设移动用户设备上需要执行 I 个应用程序（即 I 个任务），将其表示为 $\mathrm{I}=\{1,2,...,\mathrm{I}\}$，其最大计算要求由 S_i^{tot} 表示。为了减少本地计算延迟，本文将移动用户的计算任务 i 分成两部分：一部分是通过非正交多址接入，迁移到边缘服务器的计算任务 s_i，以及另一部分，本地完成的计算任务 $S_i^{tot}-s_i$。

用户采用非正交多址传输方式，将任务 $\mathrm{I}=\{1,2,...,\mathrm{I}\}$ 的计算任务量 $\{s_1,s_2,...,s_I\}$ 同时迁移到边缘服务器组，如下图 2-1 所示。使用 g_i 表示从用户到边缘服务器 i 的信道功率增益。为了便于演示，本文假设 I 个边缘服务器的排列按序是

$$\mathrm{g}_1>\mathrm{g}_2>...>\mathrm{g}_\mathrm{I} \tag{2-1}$$

基于式（2-1）和 SIC，用户向边缘服务器 i 发送 s_i 的最小总传输功率（持续时间为 t）为[40]：

$$P^{tot}(\{s_i\}_{\forall i\in I},t)=Wn_0\sum_{i=1}^{I}(\frac{1}{g_i}-\frac{1}{g_{i-1}})2^{\frac{1}{t}\frac{1}{W}\sum_{m=i}^{I}s_m}-\frac{Wn_0}{g_I} \tag{2-2}$$

式（2-2）中，参数 W 表示信道带宽，参数 n_0 表示背景噪声的功率密度。其中预先设置 $\frac{1}{g_0}=0$，（即表示 g_0 无穷大）。

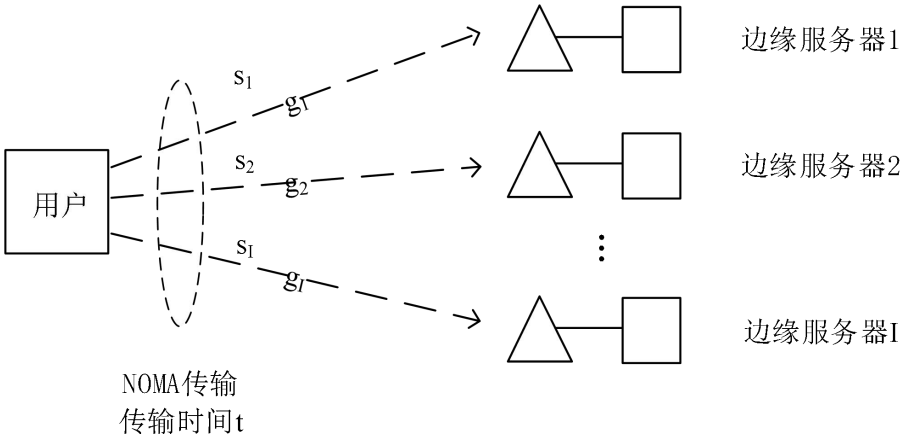

图 2-1　用户任务迁移示意图

2.1.2　问题优化

在上述场景下，本文需要研究的问题是，移动用户如何决定：1）每个任务 i 的分流计算量 s_i；2）将计算任务量 $\{s_1,s_2,...,s_i\}$ 传送到各个边缘服务器的传输时间 t；3）用户在本地为每个任务安排的计算速度 $\mu_{L,i}$，，单位为“量\MU”。

首先，移动用户完成其 S_i^{tot} 的总体延迟能表示为

$$d_i^{ove}=\max\{\frac{S_i^{tot}-s_i}{\mu_{L,i}},t+\frac{s_i}{\mu_i}\} \tag{2-3}$$

其中 $\frac{S_i^{tot}-s_i}{\mu_{L,i}}$ 表示移动用户执行剩余负载 $S_i^{tot}-s_i$ 的本地计算延迟。$t+\frac{s_i}{\mu_i}$ 表示 MU 迁移其工作负载 s_i 到边缘服务器的总延迟。使用式（2-3）来量化移动用户的总延迟，并建模以下问题，目的是最小化用户完成所有任务的总能量（“EM”表示“能量最小化”）[46,47]：

（EM-i）：　$\min\ tP^{tot}(\{s_i\}_{\forall i\in I},t)+\sum_{i\in I}\frac{S_i^{tot}-s_i}{\mu_{L,i}}\rho(\mu_{L,i})^3$

$$=\min\ tP^{tot}(\{s_i\}_{\forall i\in I},t)+\sum_{i\in I}(S_i^{tot}-s_i)\rho(\mu_{L,i})^2 \tag{2-4}$$

约 束:　$P^{tot}(\{s_i\}_{\forall i\in I},t)\le \mathrm{P}^{\max},$ (2-5)

$$d_i^{ove}=\max\{\frac{S_i^{tot}-s_i}{\mu_{L,i}},t+\frac{s_i}{\mu_i}\}\le T_i^{\max},\forall i\in I, \tag{2-6}$$

$$\sum_{i\in \mathrm{I}}\mu_{L,i}\le \mu_{\mathrm{L}}^{\max}, \tag{2-7}$$

变 量: $\{s_i\}_{\forall i\in I}, t$ 和 $\{\mu_{L,i}\}_{\forall i\in I}$

其中 $tP^{tot}(\{s_i\}_{\forall i\in I}, t)$ 表示 MU 迁移负载能耗，$\sum_{i\in \mathrm{I}} \frac{S_i^{tot}-s_i}{\mu_{L,i}}\rho(\mu_{L,i})^3$ 表示本地 CPU 计算能耗（基于动态电压调整原理，本地 CPU 的计算功率是计算速度 $\mu_{L,i}$ 的三次方，ρ 是系数，由芯片结构决定）。约束（2-5）指的是非正交多址接入传输功率不能超过移动用户的功率预算 $\mathrm{P}^{\max}$。约束（2-6）表示 MU 的总延迟不能超过最大延迟 $T_i^{\max}$。约束（2-7）表示移动用户本地的总 CPU 计算速度 $\mu_{L,i}$ 不能超过最大计算速度 $\mu_{\mathrm{L}}^{\max}$。

由式（2-6）可得，$\frac{S_i^{tot}-s_i}{\mu_{L,i}} \le T_i^{\max}, \forall i \in I,$，即

$$\frac{S_i^{tot}-s_i}{T_i^{\max}} \le \mu_{L,i}, \forall i \in \mathrm{I}, \tag{2-8}$$

表示移动用户在本地执行任务 i 所需计算速度的最小值。此外，$t+\frac{s_i}{\mu_i} \le T_i^{\max}, \forall i \in I,$，即

$$s_i \le (T_i^{\max}-t)\mu_i, \forall i \in \mathrm{I}, \tag{2-9}$$

将式（2-8）带入目标（2-4），可将优化问题（EM-i）转化为

$$\min\ tP^{tot}(\{s_i\}_{\forall i\in I}, t)+\sum_{i\in I}(S_i^{tot}-s_i)\rho(\frac{S_i^{tot}-s_i}{T_i^{\max}})^2$$

$$=\min\ tP^{tot}(\{s_i\}_{\forall i\in I}, t)+\rho\sum_{i\in I}(\frac{1}{T_i^{\max}})^2(S_i^{tot}-s_i)^3 \tag{2-10}$$

约 束: $$P^{tot}(\{s_i\}_{\forall i\in I}, t) \le \mathrm{P}^{\max}, \tag{2-11}$$

$$0 \le s_i \le (T_i^{\max}-t)\mu_i, \forall i \in \mathrm{I}, \tag{2-12}$$

$$0 \le \sum_{i\in \mathrm{I}} \frac{S_i^{tot}-s_i}{T_i^{\max}} \le \mu_{\mathrm{L}}^{\max}, \tag{2-13}$$

变 量: t 和 $\{s_i\}_{\forall i\in I}$

从上述约束条件可知，本文可以通过仅优化 t 和 $\{s_i\}_{\forall i\in I}$ 来实现能量最小化。此外，由式(2-4)可知，在给定 t 下，优化问题是一个凸优化问题，因此分两步解决问题（EM-i）。

1）第一步：在给定 t 下，优化迁移负载 s_i。首先考虑给出移动用户的传输时间 t 来优化迁移负载 s_i，来尽量减少移动用户消耗的总能量，因此问题求解如下：

$$V_t = \min\ tP^{tot}(\{s_i\}_{\forall i\in I}, t) + \rho\sum_{i\in I}(\frac{1}{T_i^{\max}})^2(S_i^{tot} - s_i)^3 \tag{2-14}$$

约 束:
$$P^{tot}(\{s_i\}_{\forall i\in I}, t) \le \mathrm{P}^{\max}, \tag{2-15}$$

$$0 \le s_i \le (T_i^{\max} - t)\mu_i, \forall i \in \mathrm{I}, \tag{2-16}$$

$$0 \le \sum_{i\in \mathrm{I}} \frac{S_i^{tot} - s_i}{T_i^{\max}} \le \mu_{\mathrm{L}}^{\max}, \tag{2-17}$$

变 量:
$$\{s_i\}_{\forall i\in I}$$

2）第二步：优化 t。由第一步可得出每个给定 t 下的 V_t,因此，通过调整 $t \in [0, \min_{i\in \mathrm{I}}\{T_i^{\max}\}]$ 来最小化 V_t，即求解问题：

$$\min V_t$$

约 束:
$$0 \le t \le \min_{i\in \mathrm{I}}\{T_i^{\max}\}, \tag{2-18}$$

变 量:
$$t$$

上述分步计算，能够解决问题（EM-i）。首先，给定 t，优化 $\{s_i\}_{\forall i\in I}$ 求解当前情况下的最小总能量，而传输时间 t 是有范围的，只需以较小的步长线性改变 t，依次求出相应的最小能量，即可求出所有情况下，单用户实现任务迁移的最小能耗，等价地解决问题（2-4）。

2.2 算法设计

在第一步中，本文假设了将不同任务分配到不同服务器的方案已经给定，在这个方案给定情况下，设计上述方法，可以计算出一个用户完成所有任务的所需要的最小能量消耗。而不同任务和不同服务器的情况下，用户将计算任务分配到边缘服务器的方式有多种，因此，本文算法第二步是，要找到分配任务（或者说分配服务器）的方案，从而进一步最小化总能量消耗。

2.2.1 系统建模

接下来，要找到任务和服务器的匹配方案。最直接的方法就是将I个任务排列组合，计算所有方案对应的最小能耗，找出能量消耗最小的组合情况。但这个方法缺点明显，一旦任务和服务器的数量较多，程序会变得十分复杂，而且计算量大。

因此，为了方便且充分利用上述算法，本文采用基于交换的模拟退火算法（Simulated

Annealing Algorithm, SA-Algorithm），具体思路如下。

基于纳什均衡的排序算法，首先将边缘服务器组表示为$\mathrm{I}=\{1,2,...,\mathrm{I}\}$，将任务集表示为$\kappa=\{1,2,...,\mathrm{K}\}$。假设$\mathrm{I}\geq\mathrm{K}$，即边缘服务器的数量≥需要完成的任务数量。为了使得一个任务迁移到对应的一个边缘服务器，一个边缘服务器只承载一个任务，本文额外设置 I-K 个任务量$\mathrm{S}^{\mathrm{tot}}=0$的虚拟任务，不影响算法运行，又能保证任务和服务器一一对应的关系，即表示为$\phi(k)=i$，意味着任务 k 迁移到边缘服务器（ES）i，如图 2-2 中的虚线部分，即补充的虚拟任务，相反，$\widetilde{\phi}(i)=k$，即表示边缘服务器 i 要承载第 k 个任务的分流计算任务。

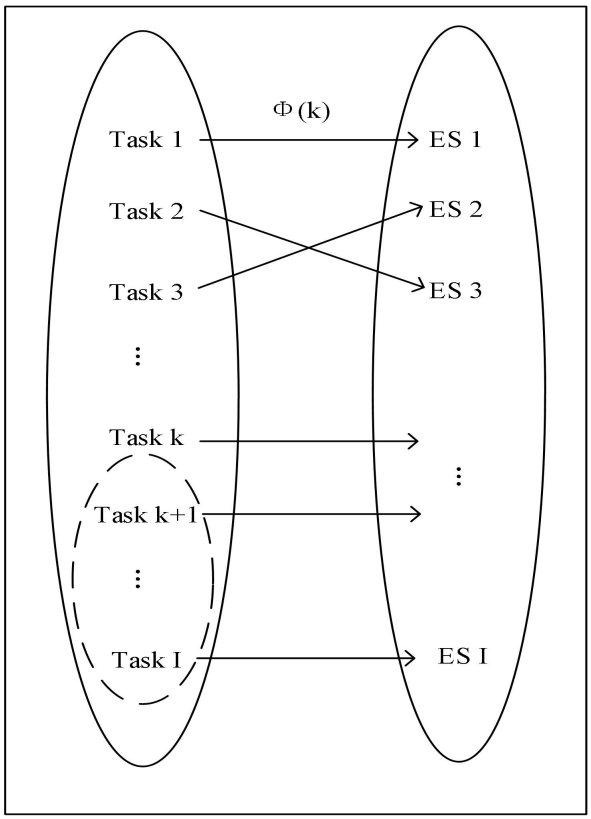

图 2-2　任务和服务器之间映射

在上述条件下，用户向边缘服务器发送$\{s_k\}_{k\in\kappa}$的最小总传输功率（持续时间为 t）转化为：

$$P^{tot}(\{s_k\}_{\forall k\in\kappa},t,\phi(k))=Wn_0\sum_{i=1}^{I}(\frac{1}{g_i}-\frac{1}{g_{i-1}})2^{\frac{1}{t}\frac{1}{W}\sum_{m=i}^{I}s_m}-\frac{Wn_0}{g_I} \tag{2-19}$$

其中，如果$\phi(k)=m$，则$s_m=s_k$，即边缘服务器 m 承载的计算量s_m等于任务 k 的分流计算量s_k。

同理，也可表示为：

$$P^{tot}(\{s_k\}_{\forall k\in\kappa},t,\widetilde{\phi}(i))=Wn_0\sum_{i=1}^{I}(\frac{1}{g_i}-\frac{1}{g_{i-1}})2^{\frac{1}{t}\frac{1}{W}\sum_{m=i}^{I}s_{\widetilde{\phi}(m)}}-\frac{Wn_0}{g_I} \tag{2-20}$$

$\widetilde{\phi}(m)$ 表示服务器 m 所服务的任务。

由此，移动用户完成其 S_k^{tot} 的总体延迟为

$$d_k^{ove}=\max\{\frac{S_k^{tot}-s_k}{\mu_{L,k}},t+\frac{s_k}{\mu_{\phi(k)}}\} \tag{2-21}$$

优化问题（EM-i）更新为

（EM-i）：
$$\min\ tP^{tot}(\{s_k\}_{\forall k\in\kappa},t,\widetilde{\phi}(i))+\sum_{i\in\mathrm{I}}\frac{S_k^{tot}-s_k}{\mu_{L,k}}\rho(\mu_{L,k})^3$$

$$=\min\ tP^{tot}(\{s_k\}_{\forall k\in\kappa},t,\widetilde{\phi}(i))+\sum_{k\in\kappa}(S_k^{tot}-s_k)\rho(\mu_{L,k})^2 \tag{2-22}$$

约 束:
$$P^{tot}(\{s_k\}_{\forall k\in\kappa},t,\widetilde{\phi}(i))\le \mathrm{P}^{\max}, \tag{2-23}$$

$$d_k^{ove}=\max\{\frac{S_k^{tot}-s_k}{\mu_{L,k}},t+\frac{s_k}{\mu_{\phi(k)}}\}\le T_k^{\max},\forall k\in\kappa \tag{2-24}$$

$$\sum_{k\in\kappa}\mu_{L,k}\le \mu_{\mathrm{L}}^{\max}, \tag{2-25}$$

变 量:
$\{s_k\}_{\forall k\in\kappa},t,\widetilde{\phi}(i)$ 和 $\{\mu_{L,k}\}_{\forall k\in\kappa}$

在上述条件下，先考虑寻找使得总能量消耗最小的组合方案的整体思路，如图 2-3，首先初始化一组对应关系 $\phi(k)$，利用 EM-i 算法，优化迁移任务量和传输时间，得到相应的最小能耗 $E_{\phi(k)}^{\min}$，再根据当前的 $E_{\phi(k)}^{\min}$，重新寻找新的对应关系，这样通过反馈不断循环，直至得出最佳的匹配方案，使得 $E_{\phi(k)}^{\min}$ 最小。

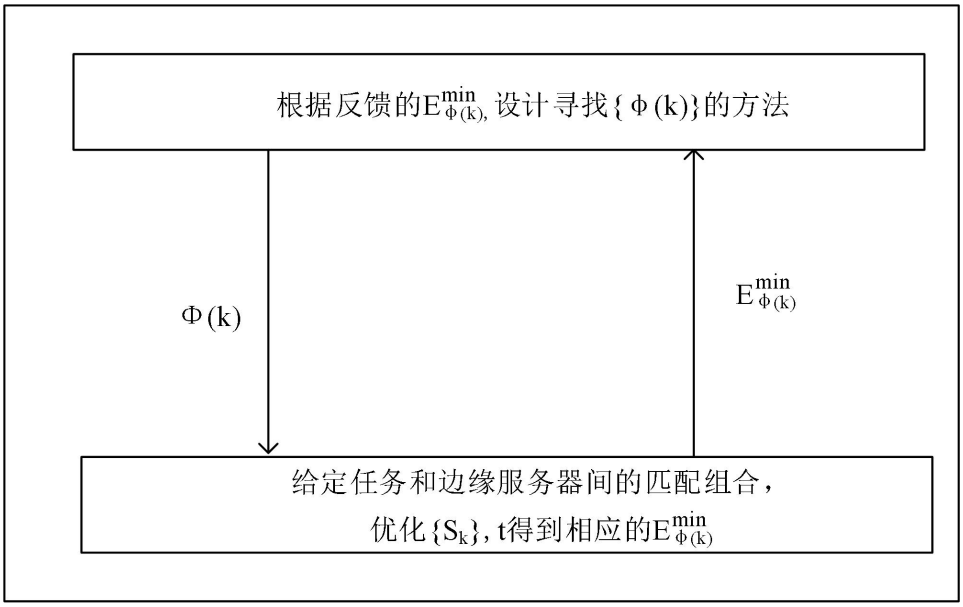

图 2-3　匹配思路框图

得到上述思路之后，经过研究，本文决定采用基于交换的 SA 算法。其中，寻找 $\phi(k)$ 的方法采用交换的思想，如图 2-4。

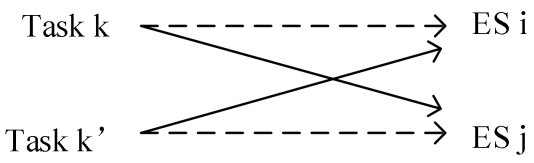

图 2-4　随机交换

SA 算法类似于最优匹配（Optimal Matching）[48]，但区别是，最优匹配的两个点集是独立的，而本文采用的算法中每对任务-边缘服务器的匹配关系并不是独立的，需要确定整体的分配方案才能计算目标函数值，即任务和服务器一一对应之后，才能计算用户完成所有计算任务的最小总能量。因此，这严格意义上是受外部影响的最优匹配（Optimal Matching with Externality）。

在普通的求解最优解的算法中，如果初始解选择的不好，可能会陷入局部最优解的情况，即无法判断是否是全局最优解而终止算法，如图 2-5，而 SA 算法在更新迭代中引入了随机因素，即以一定概率接受不优于当前解的新解，从而继续进行迭代判断，因此有可能会跳出这个局部的最优解，达到全局的最优解，如图 2-6。

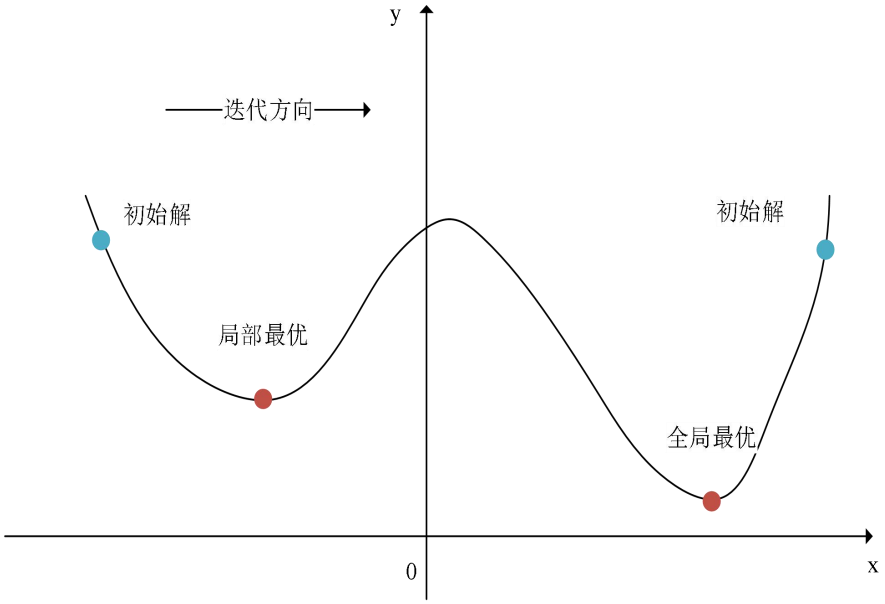

图 2-5　求最优解

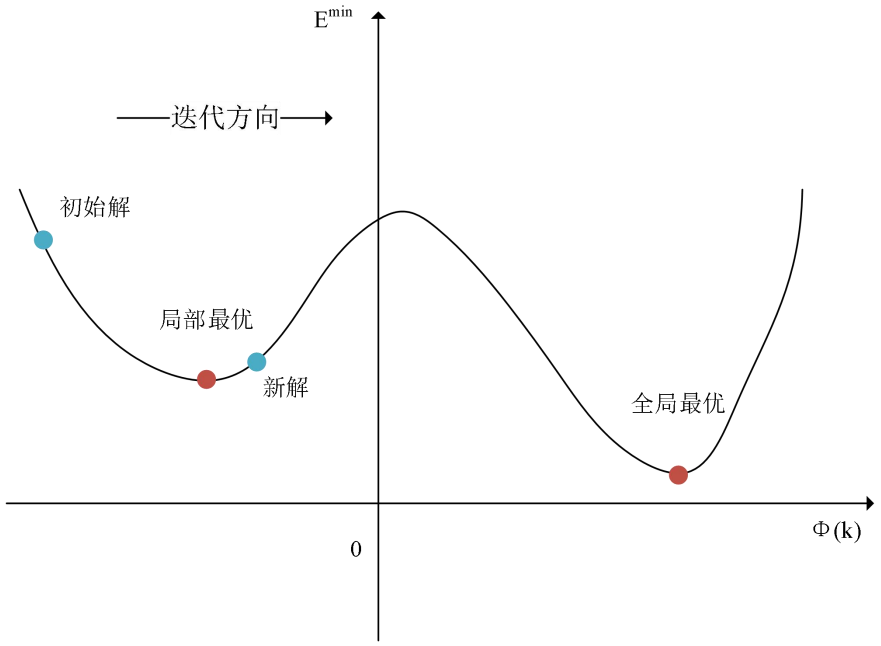

图 2-6　SA 求最优解

具体来说，模拟退火算法利用了金属退火的原理，首先以任意点为初始解，通过不断降低温度，并在每个温度下迭代，达到局部最优解时，以一定的概率接受新解，从而跳出局部最优解。如此不断迭代，寻找最优值，直至温度降到最低，即接受新解的概率为 0。这里的“一定的概率”的计算参考了金属冶炼的退火过程，即 $\rho = \exp(\Delta/kT)$，其中 Δ 表示当前解情况下目标值与新解所得目标值的绝对差值。

2.2.2　SA 算法具体步骤

将该算法代入边缘计算的场景，具体步骤如下：

1) 设定初始温度 T（充分大），初始化迭代次数 i，随机建立任务和服务器的初始始映射关系 $\{\phi(m)\}_{m\in\kappa}$，作为当前解，确定任务和边缘服务器的分配方案，利用 2.2 节中的 EM-i 算法，求解当前情况下的最小能耗 $E^{\min}_{\{\phi(m)\}_{m\in\kappa}}$；

2) 随机选定两个任务 $\phi(k)=i$ 和 $\phi(k')=j$，并交换服务器，如图 2-4，即 $\phi(k)^{\text{upd}}=j$ 且 $\phi(k')^{\text{upd}}=i$,则新的映射关系表示为 $\{\{\phi(m)\}_{m\in\kappa,m\neq k,k'},\phi(k)^{upd}=j,\phi(k')^{upd}=i\}_{m\in\kappa}$，具体实例如图 2-7 所示，即产生新解 $\{\phi(m)^{\text{upd}}\}_{m\in\kappa}$，并重新计算新情况下的最小能耗 $E^{\min}_{\{\phi(m)^{\text{upd}}\}_{m\in\kappa}}$，迭代次数 i=i+1；

3) 比较 $E^{\min}_{\{\phi(m)^{\mathrm{upd}}\}_{m\in\kappa}}$ 和 $E^{\min}_{\{\phi(m)\}_{m\in\kappa}}$，若 $E^{\min}_{\{\phi(m)^{\mathrm{upd}}\}_{m\in\kappa}} < E^{\min}_{\{\phi(m)\}_{m\in\kappa}}$，即 $\Delta<0$，则接受 $\{\phi(m)^{\mathrm{upd}}\}_{m\in\kappa}$，将当前映射关系由 $\{\phi(m)\}_{m\in\kappa}$ 更新为 $\{\phi(m)^{\mathrm{upd}}\}_{m\in\kappa}$；反之，若 $E^{\min}_{\{\phi(m)^{\mathrm{upd}}\}_{m\in\kappa}} > E^{\min}_{\{\phi(m)\}_{m\in\kappa}}$，即 $\Delta>0$，则以一定概率 $\rho=\exp(-\Delta/T)$ 接受 $\{\phi(m)^{\mathrm{upd}}\}_{m\in\kappa}$ 作为新的当前解，其中 $\Delta = E^{\min}_{\{\phi(m)^{\mathrm{upd}}\}_{m\in\kappa}} - E^{\min}_{\{\phi(m)\}_{m\in\kappa}}$，迭代次数 *i*=*i*+1，逐渐减少 *T*，重复步骤 2)和步骤 3)；

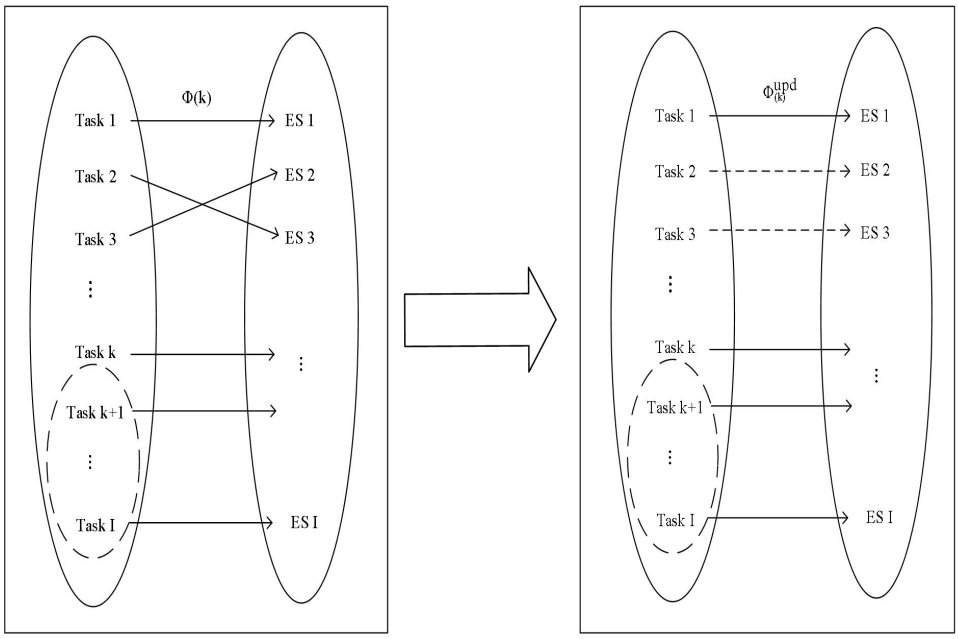

图 2-7 实例：任务 2 由服务器 3 承载，任务 3 由服务器 2 承载交换为任务 2 由服务器 2 承载，任务 3 由服务器 3 承载，其余映射不变

4) 若满足终止条件：*T* 趋近于 0；连续若干个 $\{\phi(m)^{\mathrm{upd}}\}_{m\in\kappa}$ 未被接受或连续若干次迭代得到的最优解不变，则输出最优解，终止算法；

算法的流程如图 2-8。

2.3 本章小结

在本章中，详细介绍了本文算法，具体分为两种：EM-i 算法和 SA 算法。为了便于建模，采用自下而上的思路求解问题。而整体运行思路如下：

首先初始化一组任务-服务器的对应关系作为当前解，将 EM-i 算法用于联合优化传输时间和迁移任务量，求解当前情况下的最小能耗。之后随机交换其中两个元素，即其中两个任务交换对应服务器，得到新解，再运行 EM-i 算法，得到新解对应的最小能耗，比较前后两者结果，以一定概率接受新解作为当前解，重复上述步骤不断迭代至满足终止条件，返回当前解，即得到的最优解，以及对应的总能量最小值，从而实现了基于交

换的 SA 算法。

本章基于理论，分析介绍了算法。为了判断所提出算法是否可行，本文也采用 MATLAB 进行仿真。在下一章中，将展示具体的数值结果，来验证上述算法是否能得到最优解。

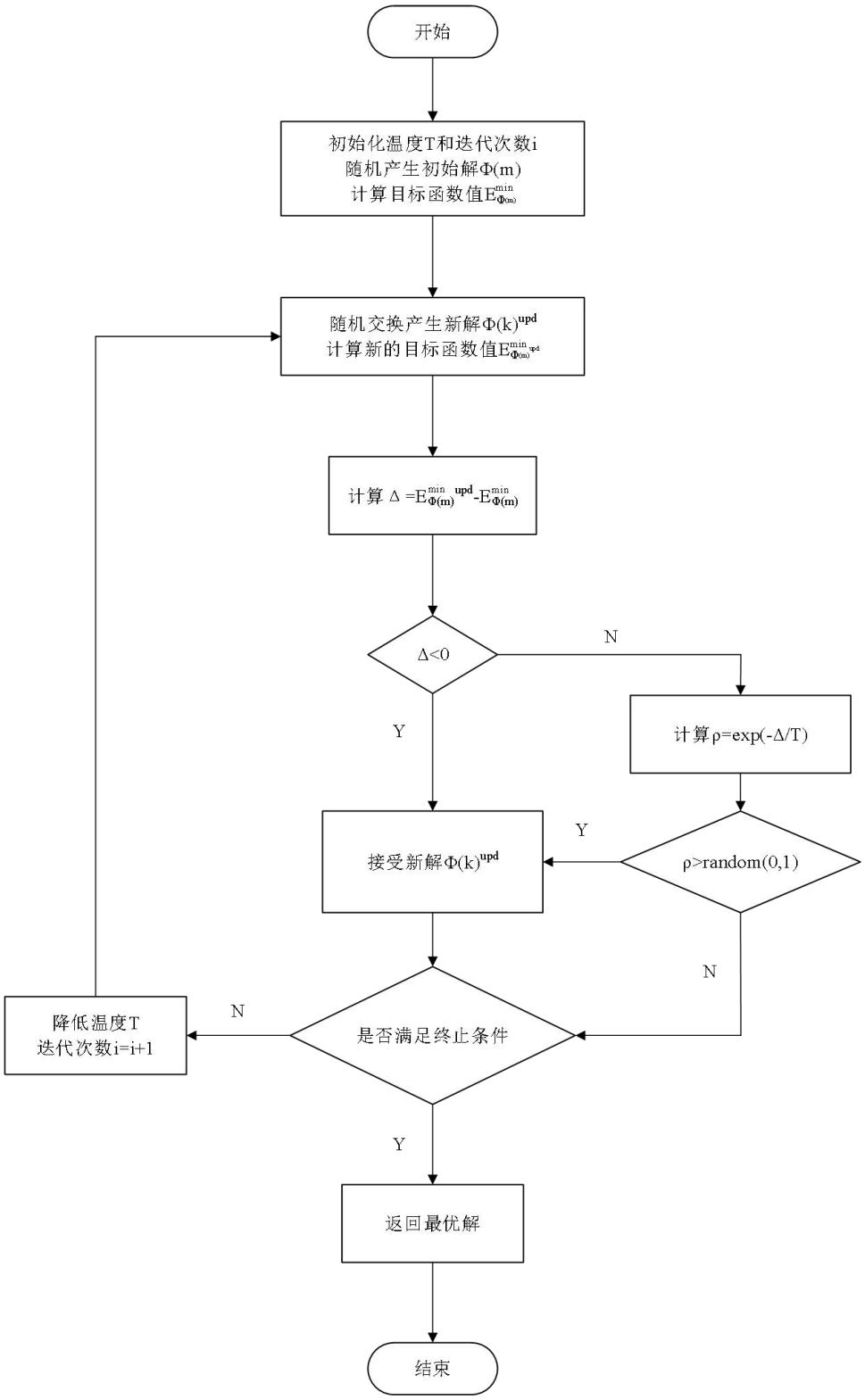

图 2-8　SA 算法流程图

第 3 章　算法实现

3.1　EM-i 算法

本文首先将验证提出的 EM-i 优化模型在单用户情况下的性能。先建立了一个 3-BS 场景，其中三个基站分别位于(500，0)、(-500，0)和(0，500)。同时，移动用户随机分布在圆心为（0,0）且半径为 100 的圆内，根据距离模型，生成从用户到基站的连续信道功率增益。根据上述设置，本小节中具体使用的随机信道功率增益为 $\{g_i\}_{i\in \mathrm{I}}=\{1.8185*10^{-7},1.7793*10^{-7},1.7599*10^{-7}\}$，信道带宽 $W_i=16\mathrm{MHz}$，背景噪声的功率密度 $n_0=10^{-8}$。本文设置服务器计算速度为 $\mu_i=15\mathrm{Mbits/s}$。此外，为移动用户设置功率预算 $\mathrm{P}^{\max}=0.5\mathrm{Joul/s}$，总任务量 $S_i^{tot}=4\mathrm{Mbits}$，最大传输时间 $T_i^{\max}=4$ 秒，以及本地 CPU 最大计算速度 $\mu_{\mathrm{L}}^{\max}=8\mathrm{Mbits/s}$。所有模拟都在 Intel（R）Core（TM）i5-6300HQ CPU@2.30 GHz 的 PC 上执行。

首先第一步，在给定传输时间 t = 2s 下，求解 $V_t=0.3624\mathrm{Joul}$，相应 $\{s_i\}_{\forall i\in \mathrm{I}}=\{2.2840\mathrm{Mbits},2.2661\mathrm{Mbits},2.2574\mathrm{Mbits}\}$。

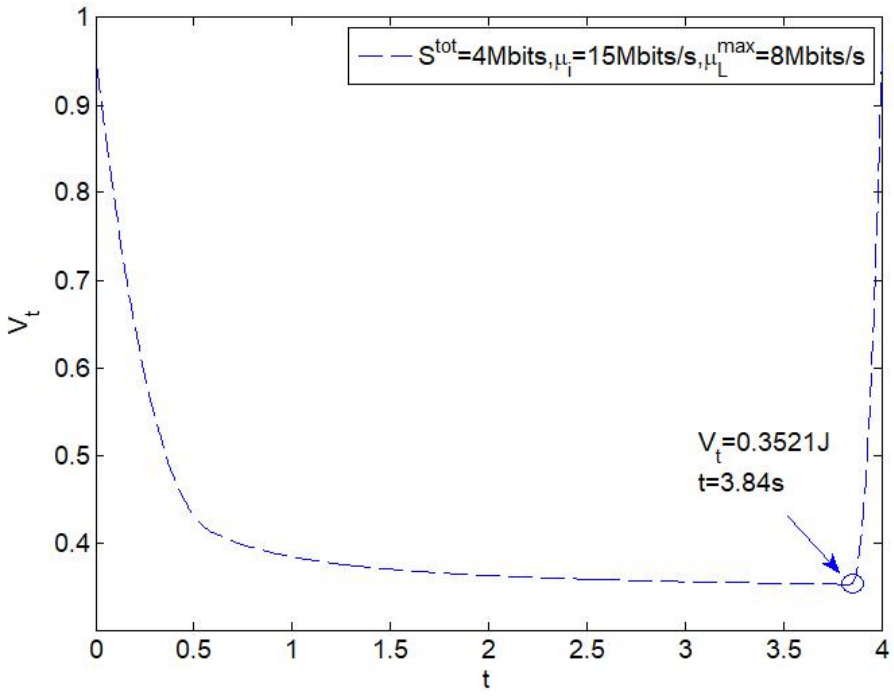

图 3-1　线性改变 t，V_t的变化情况

第二步，以 0.01 的步长线性改变$t \in [0, T_i^{\max}]$,求解$\min V_t$，如图 3-1 所示。得到结果如下，最小能量消耗 $\min V_t$=0.3521J，传输时间$t = 3.84\text{s}$，迁移计算任务量$\{s_i\}_{\forall i \in \mathrm{I}} = \{2.3419\text{Mbits}, 2.3242\text{Mbits}, 2.3155\text{Mbits}\}$。

由图 3-1 可见，EM-i 算法能顺利通过联合优化迁移负载和传输时间，求解出最小能耗。

为了进一步验证，接下来测试了$\{S_i^{tot}\} = \{4\text{Mbits}, 6\text{Mbits}, 8\text{Mbits}\}$三种不同情况，算法的数据结果，如图 3-2 所示。

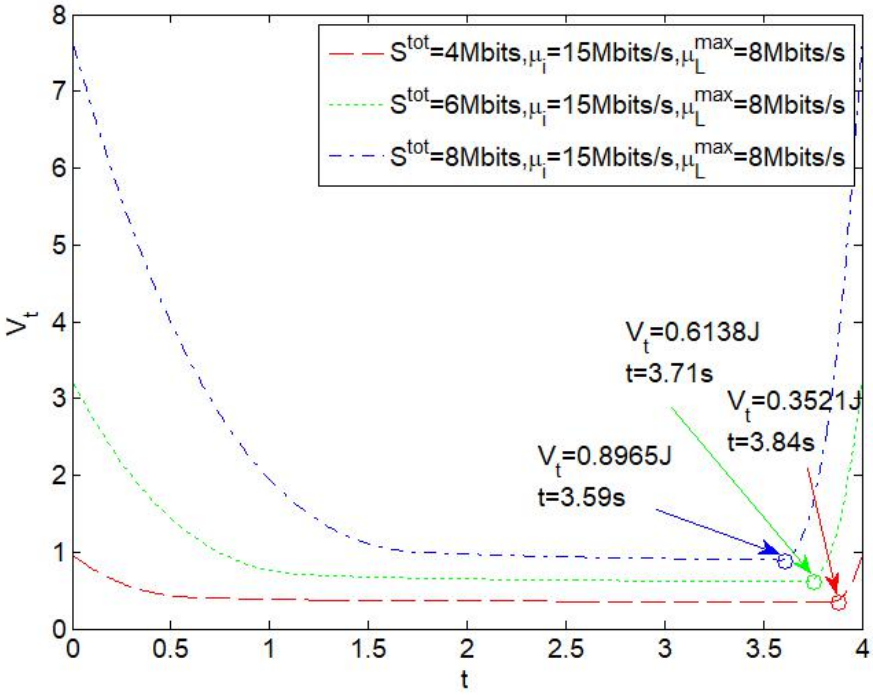

图 3-2　3 个场景下运行 EM-i 算法结果

图 3-2 展示了本文提出的 EM-i 算法的有效性。为了充分验证算法的性能，本文考虑了三个总任务量不同场景。每个场景中，一个用户将三个计算任务迁移至三个边缘服务器。其中，总任务量S_i^{tot}分别为 4Mbits, 6Mbits,8Mbits。运行结果得出，总任务量S_i^{tot}为 8Mbits，传输时间 t=3.59s 的时候，各计算任务向边缘服务器迁移的分流量为$\{s_i\}_{\forall i \in \mathrm{I}} = \{6.1500\text{Mbits}, 6.1500\text{Mbits}, 6.1500\text{Mbits}\}$；总任务量$S_i^{tot}$为 6Mbits，传输时间 t=3.71s 时，各计算任务向边缘服务器迁移的分流量为$\{s_i\}_{\forall i \in \mathrm{I}} = \{4.2822\text{Mbits}, 4.2643\text{Mbits}, 4.2557\text{Mbits}\}$；总任务量$S_i^{tot}$为 4Mbits，传输时间 t=3.84s 时，各分流计算量为$\{s_i\}_{\forall i \in \mathrm{I}} = \{2.3419\text{Mbits}, 2.3242\text{Mbits}, 2.3155\text{Mbits}\}$。

由图 3-2 以及运行结果可知，随着应用程序任务量S_i^{tot}的增加，$\min V_t$增加，

相应的 t 反而减小。这是由于任务量 S_i^{tot} 的增加，导致相应分流任务量 $\{s_i\}_{\forall i\in \mathrm{I}}$ 增加，在边缘服务器计算速度 μ_i，最大延迟 $T_i^{\max}$ 一定的条件下，由约束（2-16）可知，相应的最优传输时间减小。分析可知，这是为了减少能量，充分利用延迟，通过牺牲一定的传输功率能耗 $tP^{tot}(\{s_i\}_{\forall i\in I},t)$，来减小本地计算能耗 $\rho\sum_{i\in I}(\frac{1}{T_i^{\max}})^2(S_i^{tot}-s_i)^3$，从而使得总能耗 V_t 达到最优。

三种情况下的传输能耗、服务器计算能耗、全部本地计算能耗如下表：

表 3-1　3-BS 情况下任务迁移与全部的本地计算对比

任务量（Mbits）	传输总能耗（Joul）	本地 CPU 计算能耗（Joul）	用户总能耗（Joul）	全部本地计算能耗（Joul）	改进效率	所有任务分流量/总计算量
$S_i^{tot}=4$	0.2819	0.0702	0.3521	0.9600	63.3%	0.5818
$S_i^{tot}=4$	0.5357	0.0780	0.6138	3.2400	81.06%	0.7112
$S_i^{tot}=4$	0.8016	0.0950	0.8965	7.6800	88.33%	0.7688

表 3-1 中数据直观地显示，在三种情况下，用户将部分任务迁移到边缘服务器与全部任务本地处理相比，所需能耗减小，效率增加，处理计算密集型任务时，优势更加明显。由此也证明了边缘计算的有效性，将任务迁移到边缘服务器能大大降低能耗。

3.2　SA 算法

接下来，验证算法的第二步。找到分配任务的方案，使得 MU 消耗的总能量最小。建立一个 7-BS，5-task 的场景，运行 SA 算法，允许的最大延迟分别为 $T_k^{\max}=[2s,3s,4s,5s,6s]$，测试对应任务量分别为 $S_k^{tot}=[4M,5M,6M,7M,8M]$、$S_k^{tot}=[6M,7M,8M,9M,10M]$、$S_k^{tot}=[8M,9M,10M,11M,12M]$ 三种情况下，系统能够接受的最小能耗。此外我们设置初始温度值 T=300，迭代次数 i=0，运行算法得到结果如图 3-3。

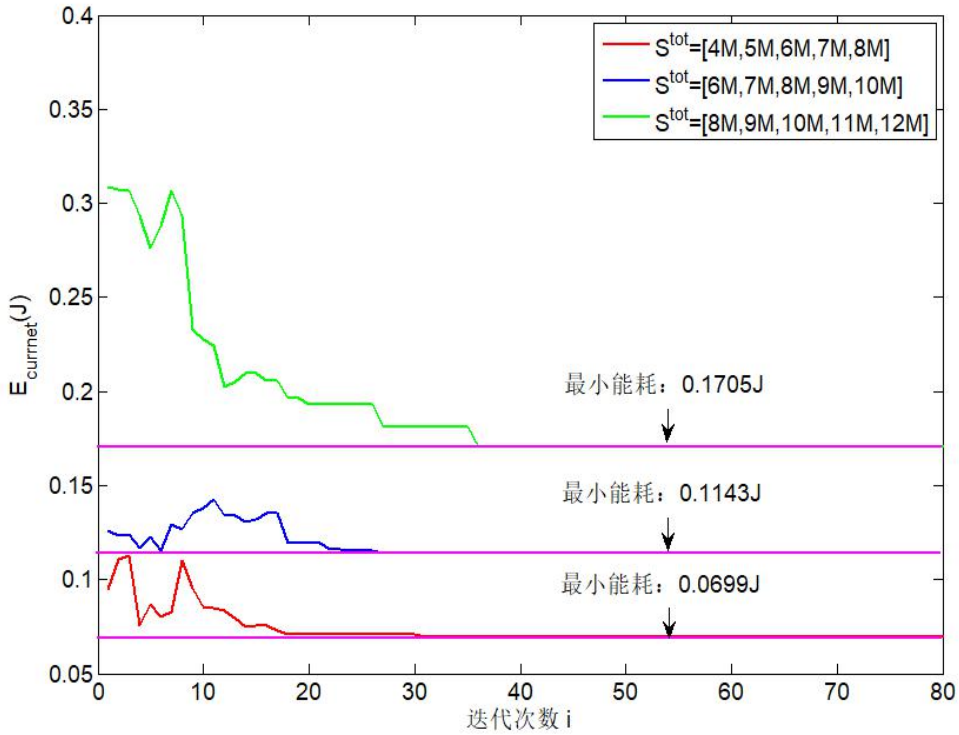

图 3-3 $E^{\min}_{\{\phi(m)\}_{m\in\kappa}}$-i 曲线

由图 3-3 可知，三种情况下，随着总任务量增加，最小能耗逐渐增加，而迭代次数达到一定值时，算法收敛，都能输出最优解，即找到最小能耗对应的任务与服务器分配方案。

接下来，将分配方案进行全排列，并计算相应的 $E^{\min}_{\{\phi(m)\}_{m\in\kappa}}$，将其与 SA 算法的运行结果比较，从另一角度证明本文的 SA 算法是否正确。分别建立 3 个任务、4 个任务、5 个任务的场景，记录每次的排列方案，即不同服务器与承载不同任务的对应方案，与相应的最小能量，单位 Joul。其中，3 个任务场景下，单用户计算任务量分别为 $S_k^{tot}=[5\text{M},6\text{M},7\text{M}]$，相应允许的最大传输延迟 $T_k^{\max}=[2\text{s},3\text{s},4\text{s}]$。用户向边缘服务器迁移计算任务时，信道增益分别为 $g_i=[9.1831*10^{-6},5.4139*10^{-6},3.2571*10^{-6}]$，边缘服务器的计算速度 $\mu_i=[12\text{Mbits/s},15\text{Mbits/s},11\text{Mbits/s}]$。4 个任务场景下，单用户计算任务量分别为 $S_k^{tot}=[5\text{M},6\text{M},7\text{M},8\text{M}]$，相应允许的最大传输延迟 $T_k^{\max}=[2\text{s},3\text{s},4\text{s},5\text{s}]$。向边缘服务器迁移时，信道增益 $g_i=[9.1831*10^{-6},5.4139*10^{-6},3.2571*10^{-6},2.4311*10^{-6}]$，边缘服务器的计算速度 $\mu_i=[12\text{Mbits/s},15\text{Mbits/s},11\text{Mbits/s},10\text{Mbits/s}]$。5 个任务场景下，单用户计算任务量分别为 $S_k^{tot}=[5\text{M},6\text{M},7\text{M},8\text{M},9\text{M}]$，相应允许的最大传

输延迟 $T_k^{\max}=[2s,3s,4s,5s,6s]$。向边缘服务器进行任务迁移时，对应的信道增益分别为 $g_i=[9.1831*10^{-6},5.4139*10^{-6},3.2571*10^{-6},2.4311*10^{-6},2.2623*10^{-6}]$，边缘服务器的计算速度 $\mu_i=[12Mbits/s,15Mbits/s,11Mbits/s,10Mbits/s,9Mbits/s]$。

运行结果输出 SA 算法与全排列所得最佳匹配方案与相应的最小能耗，具体对比如下表，其中假定边缘服务器有固定顺序，与 μ_i 一一对应，即分配方案 $\{S_k^{tot}\}$ 的索引，其值表示边缘服务器所承载任务的总任务量。

表 3-2　多任务场景下全排列与 SA 算法结果

	输出结果	3 个计算任务	4 个计算任务	5 个计算任务
SA 算法	分配方案 $\{S_k^{tot}\}$	[7 6 5]	[8 7 6 5]	[9 8 7 5 6]
	最小能耗 $E^{\min}$	0.0281J	0.0538	0.0904
全排列	分配方案 $\{S_k^{tot}\}$	[7 6 5]	[8 7 6 5]	[9 8 7 5 6]
	最小能耗 $E^{\min}$	0.0281J	0.0538	0.0904

其中，完成 3 个任务迁移时，SA 算法迭代 66 次，运行时间$1.5685*10^3$s；完成 4 个任务迁移时，SA 算法迭代 95 次，运行时间 $2.5831*10^3$s；完成 3 个任务迁移时，SA 算法迭代 72 次，运行时间 $1.8935*10^3$s。由上表以及数据结果表明，3 个任务、4 个任务、5 个任务的排列组合和 SA 算法分别运行，得到的结果一致，由此进一步验证了 SA 算法的可行性，且尤其适用于任务数量多、边缘服务器多的场景。

3.3　本章小结

本章展示了两种算法在 MATLAB 仿真下的数值结果，首先介绍了仿真背景，参数设置，之后展示仿真结果。图 3-1 和 3-2 表明了通过优化传输时间和任务迁移量，可以求解用户完成计算任务的最小能耗。表 3-1 表明了 EM-i 算法相较于全部本地计算的优势。图 3-3 也证明了 SA 算法的收敛性，能通过不断迭代，返回全局最优解。接下来，本文通过将 SA 算法与全排列的结果比较，即表 3-2，进一步证明了 SA 算法的有效性，能适用于任务数多、服务器多的场景。

第 4 章　总结与展望

4.1　全文总结

本文主要研究基于非正交多址的边缘计算算法，旨在为边缘计算的研究提供一种思路。全文首先介绍了选题的意义，在当前时代背景下，边缘计算是对传统云计算的弥补，能很好地解决数据量大、延迟低、带宽受限等问题。且边缘计算优势明显，能应用于诸多新兴应用程序，促进公共安全，推进智慧家居、智慧城市建设等[49]。其次，介绍了相关理论成果，从边缘计算的计算/通信模型引出本文算法内容——计算任务迁移模型。之后详细介绍了算法的实现过程。本文设计了单用户最小化总能量消耗的算法（EM-i 算法），算法分为两步，第一步优化迁移任务量，第二步以较小步长线性改变传输时间，得出最小能量。同时，通过 MATLAB 仿真，绘制了 3 个场景下，不同传输时间所对应的总能量曲线图，均可验证算法的有效性，得出任务迁移的最小能量。此外，本文通过上述算法以及基于交换的模拟退火算法，求解单用户多任务与多服务器的匹配方案，来进一步最小化总的能量消耗。首先随机初始化一组任务与服务器的对应关系，利用 EM-i 算法，得出当前情况下的最小能量，其次随机交换两个任务-服务器的对应关系，再得出新的最小能量，通过一定条件更新任务-服务器的对应关系，不断迭代，直至得出最小的能量以及所对应的匹配方案。接下来，本文也通过 MATLAB 进行仿真，数值结果验证了所提出算法的可行性，并与全排列的结果对比，给出的数值对比也证明了算法的优势与可行性。

4.2　未来展望

本文仅从单用户最小化总能量消耗的角度，为边缘计算的任务迁移提供一种算法思路，然而对于边缘计算技术的研究尚未停止，未来仍有其他研究方向。

本文的算法设计针对了单用户的情况，然而实际应用需要考虑多用户的情况，如何安排多个用户各自传输工作负载到边缘服务器的调度方案以及资源分配也接下来的研究方向之一。对于计算任务与边缘服务器的匹配方案，本文采用了基于交换的模拟退火算法，为了更加精确且高效的完成任务迁移，可以结合深度加强学习等的机器学习算法来考虑。边缘计算还需要综合考虑最小延迟等其他因素，这也将是未来可能的工作方向。

参考文献

[1] Abbas N , Zhang Y , Taherkordi A , et al. Mobile Edge Computing: A Survey[J]. IEEE Internet of Things Journal, 2017, AA(BB):1-17.

[2] 张平, 陶运铮, 张治. 5G 若干关键技术评述[J]. 通信学报, 2016, 37(7):15-29.

[3] 施巍松, 孙辉, 曹杰, 等. 边缘计算:万物互联时代新型计算模型[J]. 计算机研究与发展, 2017, 54(5): 907-924.

[4] Verma S , Kawamoto Y , Fadlullah Z , et al. A Survey on Network Methodologies for Real-Time Analytics of Massive IoT Data and Open Research Issues[J]. IEEE Communications Surveys & Tutorials, 2017, 19(3):1457-1477.

[5] Zhang K , Mao Y , Leng S , et al. Predictive Offloading in Cloud-Driven Vehicles: Using Mobile-Edge Computing for a Promising Network Paradigm[J]. IEEE Vehicular Technology Magazine, 2017, 24(6):55-63.

[6] Zhang K , Mao Y , Leng S , et al. Predictive Offloading in Cloud-Driven Vehicles: Using Mobile-Edge Computing for a Promising Network Paradigm[J]. IEEE Vehicular Technology Magazine, 2017,12(3):36-44.

[7] Rodrigues T G , Suto K , Nishiyama H , et al. Hybrid Method for Minimizing Service Delay in Edge Cloud Computing Through VM Migration and Transmission Power Control[J]. IEEE Transactions on Computers, 2017, 66(5):810-819.

[8] Tang F , Fadlullah Z M , Mao B , et al. An Intelligent Traffic Load Prediction Based Adaptive Channel Assignment Algorithm in SDN-IoT: A Deep Learning Approach[J]. IEEE Internet of Things Journal, 2018, 5(6):5141-5154.

[9] Hong K , Lillethun D , Ramachandran U , et al. Mobile Fog: A Programming Model for Large-Scale Applications on the Internet of Things[C]. Proceedings of the Second ACM SIGCOMM Workshop on Mobile Cloud Computing. New York, NY, USA: ACM, 2013, 15-20.

[10] 吕华章, 陈丹, 范斌, et al. 边缘计算标准化进展与案例分析[J]. 计算机研究与发展, 2018, 55(3): 487-511.

[11] Cao Y , Chen S , Hou P , et al. FAST: A fog computing assisted distributed analytics system to monitor fall for stroke mitigation[C]. IEEE International Conference on Networking. IEEE Computer Society, 2015, 2–11.

[12] A. S. Go, D. Mozaffarian, V. L. Roger, E. J. Benjamin, J. D. Berry, M. J. Blaha,

S. Dai, E. S. Ford, C. S. Fox, S. Franco et al. Heart disease and stroke statistics-2014 update. Circulation, 2014, 129(3):e28-e292.

[13]于天琪, 朱咏絮, 王现斌. 基于边缘计算的物联网监测系统中利用自编码神经网络实现的异常检测[J]. 物联网学报, 2018, 2(04):18-25.

[14]Y. Jararweh, A. Doulat, O. AlQudah, E. Ahmed, M. Al-Ayyoub, and E. Benkhelifa, The future of mobile cloud computing: Integrating cloudlets and mobile edge computing[C], in 2016 23rd International Conference on Telecommunications (ICT), 2016: 1–5.

[15]T. Beck, M. Werner, S. Feld, and S. Schimper, Mobile edge computing: A taxonomy[C], in Proc of the Sixth International Conference on Advances in Future Internet, 2014:1-7.

[16]K. Kai, W. Cong, and L. Tao, Fog computing for vehicular ad-hoc networks: paradigms, scenarios, and issues[J], The Journal of China Universities of Posts and Telecommunications, 2016, 23(3):56-96.

[17]施巍松, 张星洲, 王一帆, et al. 边缘计算：现状与展望[J]. 计算机研究与发展, 2019, 56(1):69-89.

[18]X. Chen, L. Jiao, W. Li, and X. Fu. Efficient multi-user computation offloading for mobile-edge cloud computing. IEEE/ACM Trans. Netw, 2016, 24(5):2795-2808.

[19]Wang C, Yu F R, Liang C, et al. Joint Computation Offloading and Interference Management in Wireless Cellular Networks With Mobile Edge Computing[J]. IEEE Transactions on Vehicular Technology, 2017, 66(8):7432–7445.

[20]Tan Z, Yu F R, Li X, et al. Virtual Resource Allocation for Heterogeneous Services in Full Duplex-enabled SCNs with Mobile Edge Computing and Caching[J]. IEEE Transactions on Vehicular Technology, 2017, 67(2): 1794–1808.

[21]Lyu X , Tian H , Ni W , et al. Energy-Efficient Admission of Delay-Sensitive Tasks for Mobile Edge Computing[J]. IEEE Transactions on Communications, 2018, 66(6): 2603–2616.

[22]Chen M.H, Liang B, and Dong M. Multi-user multi-task offloading and resource allocation in mobile cloud systems," unpublished paper, 2018.

[23]He Y, Zhao N, Yin H. Integrated Networking, Caching, and Computing for Connected Vehicles: A Deep Reinforcement Learning Approach[J]. IEEE Transactions on Vehicular Technology, 2018, 67(1):44-55.

[24]Chen X, Zhang H, Wu C, Mao S, Ji Y, and Bennis M, "Optimized computation

offloading performance in virtual edge computing systems via deep reinforcement learning," unpublished paper, 2018. [Online]. Available: https://arxiv.org/abs/1805.06146v1.

[25] Chen L, Zhou S, and Xu J, "Energy efficient mobile edge computing in dense cellular networks," unpublished paper, 2018. [Online]. Available: https://arxiv.org/abs/1701.07405.

[26] You C, Huang K, Chae H and Kim B. H. Energy-efficient resource allocation for mobile-edge computation offloading. IEEE Trans.Wireless Commun, 2017, 66(3): 1397–1411.

[27] Jin A L , Song W , Zhuang W . Auction-Based Resource Allocation for Sharing Cloudlets in Mobile Cloud Computing[J]. IEEE Transactions on Emerging Topics in Computing, 2018, 6(1):45–57.

[28] ETSI, "Multi-access edge computing," 2018.

[29] Ding Z , Liu Y , Choi J , et al. Application of Non-Orthogonal Multiple Access in LTE and 5G Networks[J]. IEEE Communications Magazine, 2017, 55(2):185-191.

[30] Huang H , Xiong J , Yang J , et al. Rate Region Analysis in a Full-Duplex-Aided Cooperative Non-Orthogonal Multiple Access System[J]. IEEE Access, 2017, 5:17869–17880.

[31] Gui Guan, Huang Hongji, Song Yiwei, et al. Deep Learning for An Effective Non-Orthogonal Multiple Access Scheme[J]. IEEE Transactions on Vehicular Technology, 2018, 67(9):8440–8450.

[32] Zhu J , Wang J , Huang Y , et al. On Optimal Power Allocation for Downlink Non-Orthogonal Multiple Access Systems[J]. IEEE Journal on Selected Areas in Communications, 2017, 35(12): 2744–2757.

[33] Zhang Y , Wang H M , Zheng T X , et al. Energy-efficient transmission design in non-orthogonal multiple access [arXiv][J]. arXiv, 2017, 66(3):2852-2857.

[34] Fang F , Zhang H , Cheng J , et al. Energy-Efficient Resource Allocation for Downlink Non-Orthogonal Multiple Access (NOMA) Network[J]. IEEE Transactions on Communications, 2016, 64(9):3722-3723.

[35] Zhai D, Zhang R, Cai L, Li B, and Jiang Y. Energy-Efficient user scheduling and power allocation for NOMA based wireless networks with massive IoT devices. IEEE Int. Things J., 2018, 5(3):1857–1868.

[36] Di Boya, Song Lingyang, and Li Yonghui. Sub-Channel assignment, power allocati

on, and user scheduling for non-orthogonal multiple access networks. IEEE Trans. Wireless Commun., 2016, 15(11):7686–7698.

[37]Cheng Y , Li K H , Teh K C , et al. Joint User Pairing and Subchannel Allocation for Multi-Subchannel Multi-User Non-Orthogonal Multiple Access Systems[J]. IEEE Transactions on Vehicular Technology, 2018, 67(9): 8238–8248.

[38]Qian L P , Wu Y , Zhou H , et al. Dynamic Cell Association for Non-Orthogonal Multiple-Access V2S Networks[J]. IEEE Journal on Selected Areas in Communications, 2017, 35(10):2342–2356.

[39]Zhang H, Yang H, Long K, Pan M, Karagiannidis G, and Leung V. Secure communications in NOMA system: Subcarrier assignment and power allocation. unpublished paper, 2018. [Online].Available https://arxiv.org/abs/1801.04441

[40]Wu Y , Qian L P , Mao H , et al. Optimal Power Allocation and Scheduling for Non-Orthogonal Multiple Access Relay-Assisted Networks[J]. IEEE Transactions on Mobile Computing, 2018, 17(11):2591–2606.

[41]Maatouk A , Caliskan E , Koca M , et al. Frequency-Domain NOMA with Two Sets of Orthogonal Signal Waveforms[J]. IEEE Communications Letters, 2018, 22(5):906–909.

[42]Ding Z, Fan P, and Poor H. Impact of non-orthogonal multiple access on the offloading of mobile edge computing. unpublished paper, 2018. [Online]. Available: https://arxiv.org/abs/1804.06712

[43]Li W , Mengling G , Yutong A , et al. Beamforming Aided NOMA Expedites Collaborative Multiuser Computational Off-loading[J]. IEEE Transactions on Vehicular Technology, 2018, 67(10):10027–10032.

[44]谢人超, 廉晓飞, 贾庆民, et al. 移动边缘计算卸载技术综述[J]. 通信学报, 2018, 39(11):142-159.

[45]李子姝 , 谢人超 , 孙礼 , et al. 移动边缘计算综述[J]. 电信科学, 2018, 34(01),87-101.

[46]Wang Yanting, Sheng Min, Wang Xijun, Wang Liang, and Li Jiandong. Mobile-Edge Computing: Partial Computation Offloading Using Dynamic Voltage Scaling[J]. IEEE Transactions on Communications, 2016, 64(10):4268-4282.

[47]Wu Y , Ni K , Zhang C , et al. NOMA Assisted Multi-Access Mobile Edge Computing: A Joint Optimization of Computation Offloading and Time Allocation[J]. IEEE Transactions on Vehicular Technology, 2018, 67(12):12244-12258.

[48] Massin R , Le Martret C J , Ciblat P . A Coalition Formation Game for Distributed Node Clustering in Mobile Ad Hoc Networks[J]. IEEE Transactions on Wireless Communications, 2017, 16(6):3940-3952.

[49] 祁兵,夏琰,李彬，et al. 基于边缘计算的家庭能源管理系统:架构、关键技术及实现方式[J]. 电力建设，2018,39(03),33-41

附录

缩写、符号清单、术语表

5G	Fifth Generation
AR	Augment reality
BS	Base Station
CDN	Content Delivery Network
D2D	Device to Device
EM	Energy Minimum
ES	Edge Server
IoT	Internet of Things
MCC	Mobile Cloud Computing
MEC	Mobile Edge Computing
MU	Mobile User
NOMA	Non-orthogonal Multiple Access
VR	Virtual Reality
SA-algorithm	Simulated Annealing-algorithm

致谢

光阴似箭，本科大学四年的时光转瞬即逝，这四年来，有坎坷有迷茫，也充满了欢声笑语，我由衷的感谢这段路上一直陪伴、教诲我的人们。

首先，我衷心感谢我的导师吴远教授在我完成毕设期间的精心指导，在此期间，一直是我坚强的后盾，并始终给予我强烈的支持和意见。他乐观严谨的态度，对教学和研究的认真和负有责任感，以及平易近人的人格魅力时刻感染着我，提高了我对学术研究的热情，这让我受益匪浅。在老师的积极帮助，指导和介绍下，使我在科研规划和研究方向上，获得了更丰富的学术资源，并进一步拓宽了学术视野。

感谢研究生学长的殷切帮助，他们对于学习的认真，对通信的热情关爱也成为了我学习的榜样，这些学习使我获益良多。

感谢我的同学、室友，在我难过、迷茫时一直陪伴着我，我们一同学习，共同进步，一起创造了很多美好回忆。她们乐观开朗的性格，时刻提醒着我保持积极的态度。

最后，我要感谢我的父母，将我养育长大，引导我树立正确的人生观价值观，鼓励我追求进步，好好学习。我衷心地感谢你们的出现，让我的生活精彩纷呈。