



# 计算机科学

COMPUTER SCIENCE

## 多无人机使能移动边缘计算系统中的计算卸载与部署优化

刘漳辉, 郑鸿强, 张建山, 陈哲毅

### 引用本文

刘漳辉, 郑鸿强, 张建山, 陈哲毅. 多无人机使能移动边缘计算系统中的计算卸载与部署优化[J]. 计算机科学, 2022, 49(6A): 619-627.

LIU Zhang-hui, ZHENG Hong-qiang, ZHANG Jian-shan, CHEN Zhe-yi. [Computation Offloading and Deployment Optimization in Multi-UAV-Enabled Mobile Edge Computing Systems](#)[J]. Computer Science, 2022, 49(6A): 619-627.

---

### 相似文章推荐 (请使用火狐或 IE 浏览器查看文章)

Similar articles recommended (Please use Firefox or IE to view the article)

#### [D2D 辅助移动边缘计算下的卸载策略优化](#)

Optimization of Offloading Decisions in D2D-assisted MEC Networks

计算机科学, 2022, 49(6A): 601-605. <https://doi.org/10.11896/jsjcx.210200114>

#### [空中智能反射面辅助边缘计算中基于 PPO 的任务卸载方案](#)

PPO Based Task Offloading Scheme in Aerial Reconfigurable Intelligent Surface-assisted Edge Computing

计算机科学, 2022, 49(6): 3-11. <https://doi.org/10.11896/jsjcx.220100249>

#### [超密集物联网中多任务多步计算卸载算法研究](#)

Multi-Task and Multi-Step Computation Offloading in Ultra-dense IoT Networks

计算机科学, 2022, 49(6): 12-18. <https://doi.org/10.11896/jsjcx.211200147>

#### [视频缓存策略中 QoE 和能量效率的公平联合优化](#)

Fair Joint Optimization of QoE and Energy Efficiency in Caching Strategy for Videos

计算机科学, 2022, 49(4): 312-320. <https://doi.org/10.11896/jsjcx.210800027>

#### [基于 NOMA-MEC 的车联网任务卸载、迁移与缓存策略](#)

Task Offloading, Migration and Caching Strategy in Internet of Vehicles Based on NOMA-MEC

计算机科学, 2022, 49(2): 304-311. <https://doi.org/10.11896/jsjcx.210100157>

# 多无人机使能移动边缘计算系统中的计算卸载与部署优化

刘漳辉<sup>1,2</sup> 郑鸿强<sup>1,2</sup> 张建山<sup>1,2</sup> 陈哲毅<sup>3</sup>

1 福州大学数学与计算机科学学院 福州 350116

2 福建省网络计算与智能信息处理重点实验室 福州 350116

3 英国埃克塞特大学计算机科学系 埃克塞特 EX4 4QF

(lzh@fzu.edu.cn)

**摘要** 无人机与移动边缘计算技术的结合突破了传统地面通信的局限性。无人机所提供的有效视距信道可大大改善边缘服务器与移动设备之间的通信质量。为了进一步提升移动边缘计算系统的服务质量,设计了一种多无人机使能的移动边缘计算系统模型。在该系统中,无人机作为边缘服务器为移动设备提供计算服务,通过联合优化无人机部署与计算卸载策略实现平均任务响应时间的最小化。基于问题定义,提出了一种 PSO-GA-G 双层嵌套联合优化方法,该方法的外层采用了结合遗传算法算子的离散粒子群优化算法(Discrete Particle Swarm Optimization Algorithm Combined with Genetic Algorithm Operators, PSO-GA),实现了对无人机部署位置的优化;而该方法的内层则是采用了贪心算法(Greedy Algorithm),实现了对计算卸载策略的优化。大量仿真实验验证了所提方法的可行性和有效性。实验结果表明,相比其他基准方法,所提出方法可以实现更短的平均任务响应时间。

**关键词**: 移动边缘计算; 无人机部署; 计算卸载; 离散粒子群优化算法; 贪心算法

**中图分类号** TP393

## Computation Offloading and Deployment Optimization in Multi-UAV-Enabled Mobile Edge Computing Systems

LIU Zhang-hui<sup>1,2</sup>, ZHENG Hong-qiang<sup>1,2</sup>, ZHANG Jian-shan<sup>1,2</sup> and CHEN Zhe-yi<sup>3</sup>

1 College of Mathematics and Computer Science, Fuzhou University, Fuzhou 350116, China

2 Fujian Provincial Key Laboratory of Networking Computing and Intelligent Information Processing, Fuzhou 350116, China

3 Department of Computer Science, University of Exeter, Exeter EX4 4QF, United Kingdom

**Abstract** The combination of unmanned aerial vehicles(UAVs) and mobile edge computing(MEC) technology breaks the limitations of traditional terrestrial communications. The effective line-of-sight channel provided by UAVs can greatly improve the communication quality between edge servers and mobile devices(MDs). To further enhance the quality-of-service(QoS) of MEC systems, a multi-UAV-enabled MEC system model is designed. In the proposed model, UAVs are regarded as edge servers to offer computing services for MDs, aiming to minimize the average task response time by jointly optimizing UAV deployment and computation offloading. Based on the problem definition, a two-layer joint optimization method(PSO-GA-G) is proposed. On one hand, the outer layer of the proposed method utilizes a discrete particle swarm optimization algorithm combined with genetic algorithm operators(PSO-GA) to optimize the UAV deployment. On the other hand, the inner layer of the proposed method adopts a greedy algorithm to optimize the computation offloading. Extensive simulation experiments verify the feasibility and effectiveness of the proposed method. The results show that the proposed method can achieve shorter average task response time, compared to other baseline methods.

**Keywords** Mobile edge computing, Unmanned aerial vehicle deployment, Computation offloading, Discrete particle swarm optimization algorithm, Greedy algorithm

### 1 引言

随着物联网的不断发展和第五代移动通信技术的大规模商用,越来越多的新兴移动应用融入到了人们的日常生活中,如无人驾驶、虚拟现实和人脸识别等<sup>[1]</sup>。这类计算密集型的

应用通常对时延具有较强的敏感性,因此在执行时对移动设备(Mobile Devices, MDs)的计算能力提出了很高的要求<sup>[2]</sup>。然而,考虑到现有硬件技术水平和便携性等因素,MDs的计算能力通常是很有限制的,这大大降低了上述应用在MDs上执行时的性能表现,并且无法很好地满足用户需求。近年来,

基金项目:国家自然科学基金(62072108);福建省自然科学基金杰青项目(2020J06014)

This work was supported by the National Natural Science Foundation of China(62072108) and Natural Science Foundation of Fujian Province for Distinguished Young Scholars(2020J06014).

通信作者:陈哲毅(zc300@exeter.ac.uk)

移动边缘计算(Mobile Edge Computing, MEC)被视为一种解决上述问题的前瞻性技术手段<sup>[3]</sup>。MEC通过将边缘服务器部署在网络边缘(如蜂窝网络基站或WiFi接入点)的方式,解决了传统云计算技术<sup>[4]</sup>集中式处理而引发的网络拥塞问题。因此,MEC能够以较小的时间开销为用户提供计算服务,进而有效提高服务质量(Quality-of-Service, QoS)。

传统的移动通信通常依赖于地面通信基础设施,但是在一些偏远区域(如山区和海洋)以及一些紧急情况(如灾难救援和军事演习)下,有限的地面通信基础设施可能会对移动通信的效率产生巨大的影响<sup>[5]</sup>。无人机(Unmanned Aerial Vehicles, UAVs)因其灵活度高、移动性强和部署成本低等特点,逐渐被应用到应急通信领域。UAVs可以作为移动通信基站为地面的MDs提供通信服务,从而构筑地-空一体的通信网络<sup>[6]</sup>。使用UAVs充当通信基站(边缘服务器)具有两个方面的优势:一是UAVs与MDs构成的有效视距(Line-of-Sight, LOS)信道能够避免障碍物遮挡导致的信号衰减和穿透损耗问题<sup>[7]</sup>;二是UAVs能够灵活地改变其在网络中部署的位置,当MDs或网络状态发生改变时,UAVs能够以更低的成本与更快的速度进行自适应地调整。

但是,将UAVs应用于MEC系统还存在着诸多挑战。一方面,在传统的MEC系统中,边缘服务器的部署通常依赖于地面的基础设施,但是使用UAVs作为边缘服务器时,需要更加灵活的UAV部署策略,因此传统的边缘服务器部署策略无法很好地适用于UAVs辅助的MEC系统。UAVs部署的位置将直接影响MDs与UAVs之间的通信时延和能耗,因此UAV部署策略至关重要。另一方面,传统的计算卸载策略通常是将来自MDs的计算任务全部卸载到边缘服务器上执行,但是UAVs并没有配备与传统边缘服务器相当的计算资源,因此UAVs可能无法为其覆盖范围内所有的MDs提供计算服务。计算卸载策略将直接影响到UAVs的计算效率,因此有效的计算卸载策略不可或缺。

针对上述挑战,本文对多无人机使能移动边缘计算系统中的计算卸载与部署优化问题展开研究。该问题的优化目标为,在一个多无人机使能的MEC系统中,通过优化UAVs的部署位置和计算卸载策略,最小化所有计算任务的平均响应时间。本文的主要贡献如下:

(1)提出了一种多无人机使能的移动边缘计算系统模型。基于该模型,定义了带有服务数量约束的平均响应时间优化问题。

(2)针对所定义的优化问题,提出了一种PSO-GA-G双层嵌套联合优化方法。该方法的外层采用了结合遗传算法算子的离散粒子群优化算法(PSO-GA)得到UAV部署方案,内层则是通过贪心算法得到计算卸载方案。最后,通过循环迭代的方式求解该优化问题。

(3)设计了完备合理的仿真环境,并进行了大量的仿真实验,验证了所提方法的可行性和有效性。实验结果表明,相比其他基准方法,所提方法可以实现更短的平均任务响应时间。

本文第2节回顾了相关工作;第3节描述了多无人机使能的移动边缘计算系统模型以及相应计算卸载与部署优化问题的定义;第4节详细介绍了本文提出的PSO-GA-G双层嵌套联合优化方法;第5节进行了仿真实验验证与分析;最后总结全文。

## 2 相关工作

近年来,MEC在工业界和学术界都受到了广泛的关注<sup>[8]</sup>。MEC通过将远程服务器的计算资源下沉到网络边缘的方式,有效减少了MDs与服务器之间的通信时延,从而提升用户体验<sup>[9]</sup>。

计算卸载作为MEC的关键技术之一<sup>[10]</sup>,通过合理的卸载决策和资源分配将MDs上运行的计算任务卸载到边缘服务器上,进而利用其充足的计算和存储资源完成任务。传统的计算卸载技术允许终端采用粗粒度卸载方式将计算任务整体卸载到边缘服务器上<sup>[11]</sup>,或者对任务进行细粒度的划分并将其部分卸载到边缘服务器上<sup>[12]</sup>。随着5G网络的发展,也出现了一种MEC和D2D协同的卸载技术<sup>[13]</sup>。相关研究表明,MEC的计算卸载技术能够有效地缩短任务的完成时间<sup>[14]</sup>、降低设备的能耗<sup>[15]</sup>并减少系统的通信代价<sup>[16]</sup>。

将UAVs和MEC相结合,有望提供一种更可靠、时延更低和覆盖范围更广的网络服务<sup>[17]</sup>,可被视为MEC领域的一项新技术。当前针对UAVs使能的MEC系统的研究主要集中在计算卸载方面。Chen等<sup>[18]</sup>引入了排队论中的先来先服务(First Come First Served, FCFS)算法,并将优化问题定义为卸载博弈,进而设计了一种基于并发最佳响应的分布式卸载算法。Zhang等<sup>[19]</sup>设计了一种考虑卸载决策和资源竞争约束的UAVs使能MEC系统,并提出了一种基于博弈论的解决方案来证明纳什均衡的存在。Kim等<sup>[20]</sup>提出了一种基于机器学习的计算卸载方法,实现了对系统能耗和时延的优化。Wang等<sup>[21]</sup>提出了一种基于强化学习的任务卸载和UAVs资源分配的优化算法,并在大规模场景中展现出了良好的优化效果。Seid等<sup>[22]</sup>提出了基于无模型深度强化学习的协作计算卸载和资源分配方案,该方案可以控制连续的动作空间,每个代理独立地在网络中学习高效的卸载策略,并通过Jain公平指数检查无人机的状态。

此外,由于UAVs和MDs之间的相对位置将直接影响传输信道状态,合适的UAVs部署方案能够显著提升网络的通信质量,因此UAVs部署优化是UAVs通信领域的另一个研究热点。Yao等<sup>[23]</sup>分析了UAVs的分布特性,提出了一种随机最优响应算法来解决UAVs部署问题,目标是最小化系统能耗。Yang等<sup>[24-25]</sup>设计了一种基于多UAVs辅助MEC的物联网(Internet-of-Things, IoT)架构,并提出了一种基于差分进化算法的UAVs部署策略,实现了多个UAVs之间的负载均衡。Zhang等<sup>[26]</sup>提出了一种基于离散粒子群算法的UAVs部署算法,以实现UAVs覆盖范围的最大化。Huang等<sup>[27]</sup>提出了一种基于进化算法的UAVs部署优化方法,该方法通过优化UAVs停靠点的数量和位置来最小化系统能耗。

综上所述,UAVs使能的MEC系统已成为研究前沿,然而当前工作主要集中在系统能耗的优化方面,对于系统响应时延的优化还有待深入研究。此外,当前针对时延的研究都是从优化系统的最大响应时延这一角度开展的。与上述工作不同,本文研究了一个多UAVs使能的MEC系统,在所考虑的时隙内,通过联合优化UAVs的部署位置和MDs的卸载策略,实现对系统内各用户的平均时延的优化,从而达到对系统整体性能的优化。

### 3 系统模型和问题定义

如图 1 所示,考虑一个由多台 UAVs 使能的 MEC 系统。在该系统中,地面上不均匀地分布了  $m$  个 MDs,记为  $\mathcal{M} = \{MD_1, MD_2, \dots, MD_m\}$ ,部署在空中的  $n$  台 UAVs,记为  $\mathcal{N} = \{UAV_1, UAV_2, \dots, UAV_n\}$ ,为地面上的 MDs 提供计算服务。

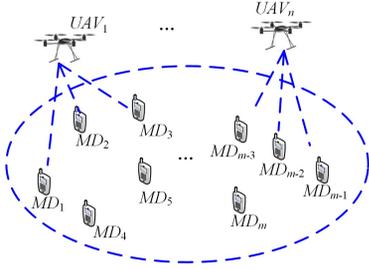


图 1 多无人机使能的 MEC 系统

Fig. 1 Multi-UAV-Enabled MEC system

假定  $MD_i$  有一项待执行的计算任务,记为  $Task_i = (D_i, S_i)$ ,其中  $D_i$  表示该任务输入数据量的大小(以 bit 为单位), $S_i$  表示该任务每 bit 的数据量所需的计算资源量(以 CPU 的计算周期数为单位)。此外,MDs 与 UAVs 之间通过无线信道进行数据交换,MDs 可以将其计算任务全部卸载到 UAVs 上执行。本节中主要涉及的符号及其定义如表 1 所列。

表 1 符号定义

Table 1 Symbol definitions

符号	定义
$\mathcal{M}$	移动设备的集合
$\mathcal{N}$	无人机的集合
$MD_i$	移动设备 $i$
$UAV_j$	无人机 $j$
$m$	移动设备的数量
$n$	无人机的数量
$p^{MD}$	移动设备位置的集合
$p_i^{MD}$	移动设备 $i$ 的坐标
$p^{UAV}$	无人机位置的集合
$p_j^{UAV}$	无人机 $j$ 的坐标
$f_i^{MD}$	移动设备 $i$ 的 CPU 计算频率
$f_j^{UAV}$	无人机 $j$ 的 CPU 计算频率
$A$	任务卸载矩阵
$Task_i$	移动设备 $i$ 的计算任务
$D_i$	计算任务 $i$ 的数据量
$S_i$	任务 $i$ 中每 bit 需要的计算资源量
$H$	无人机悬停高度
$N_{max}$	最大任务并发数
$P_i$	移动设备 $i$ 的传输功率
$g_0$	单位距离下的信道增益
$B$	信道带宽
$\sigma^2$	高斯白噪声功率
$h_{i,j}$	移动设备 $i$ 和无人机 $j$ 之间的信道增益
$R_{i,j}$	移动设备 $i$ 和无人机 $j$ 之间的传输速率
$T_{avg}$	平均任务响应时间

#### 3.1 通信模型

MDs 和 UAVs 的位置采用三维笛卡尔坐标的形式表示, $MD_i$  的位置表示为  $p_i^{MD} = (x_i^{MD}, y_i^{MD}, 0)$ , $UAV_j$  的位置表示为  $p_j^{UAV} = (x_j^{UAV}, y_j^{UAV}, H)$ ,其中  $H$  表示系统中该 UAV 的飞行高度。为了简化模型,假定在所考虑的时隙中,所有 UAVs 都部署在同一高度<sup>[28]</sup>,并且部署后的位置保持不变。

由于 UAVs 所处的高度远超 MDs,UAVs 通信链路的视距信道比其他信道损耗(如小尺度损耗和阴影损耗)更显著。

因此, $MD_i$  与  $UAV_j$  之间的无线信道可利用自由空间路径损耗(Free Space Path Loss, FSPL)模型进行评估<sup>[7]</sup>,即它们之间无线信道链路的功率增益为:

$$h_{i,j} = g_0 d_{i,j}^{-2} = \frac{g_0}{H^2 + \|p_i^{MD} - p_j^{UAV}\|^2} \quad (1)$$

其中, $g_0$  表示单位空间距离的信道功率增益, $d_{i,j}$  表示  $MD_i$  与  $UAV_j$  的空间距离。

为了减少传输时延, $MD_i$  始终以其最大传输功率  $P_i$  进行数据传输以改善通信链路的信噪比。因此, $MD_i$  与  $UAV_j$  之间的数据传输速率为:

$$R_{i,j} = B \log_2 \left( 1 + \frac{P_i h_{i,j}}{\sigma^2} \right) \quad (2)$$

其中, $\sigma^2$  表示通信链路中的高斯白噪声功率, $B$  表示信道带宽。

#### 3.2 计算模型

考虑到所有的 UAVs 和 MDs 都可提供计算服务,MDs 上的计算任务可卸载到 UAVs 上执行,也可在本地执行。MDs 和 UAVs 之间的任务卸载矩阵  $A \in \{0, 1\}^{m \times (n+1)}$  表示如下:

$$A = \begin{bmatrix} \alpha_{1,1} & \dots & \alpha_{1,n+1} \\ \vdots & \ddots & \vdots \\ \alpha_{m,1} & \dots & \alpha_{m,n+1} \end{bmatrix}$$

其中, $\alpha_{i,j} \in \{0, 1\}$ 。当  $MD_i$  将计算任务卸载到  $UAV_j$  上执行时, $\alpha_{i,j} = 1$ ,否则  $\alpha_{i,j} = 0$ 。特别地,当  $\alpha_{i,n+1} = 1$  时,计算任务在  $MD_i$  上执行。

假设所有 MDs 均采用完全卸载策略,那么每个 MD 上的计算任务有两种可选的执行方式:

(1)本地执行。当  $\alpha_{i,n+1} = 1$  时,计算任务在  $MD_i$  上执行。根据  $Task_i$  的定义,执行该任务所需的总计算周期数为  $S_i \times D_i$ 。因此,在  $MD_i$  上执行任务所需的时间可以表示为:

$$T_i^{loc} = \frac{S_i \times D_i}{f_i^{MD}} \quad (3)$$

其中, $f_i^{MD}$  表示  $MD_i$  的 CPU 计算频率。

(2)卸载到一台 UAV 上执行。当  $\alpha_{i,j} = 1$  时, $MD_i$  将计算任务卸载到  $UAV_j$  上执行。因此,计算任务的完成时间由 3 个部分构成,包括数据传输时间  $T_{i,j}^{tran}$ 、UAV 执行任务所需的时间  $T_{i,j}^{UAV}$  以及结果回传时间  $T_{i,j}^{back}$ 。在这个过程中, $MD_i$  向  $UAV_j$  传输数据所需的时间为:

$$T_{i,j}^{tran} = \frac{D_i}{R_{i,j}} \quad (4)$$

计算任务在  $UAV_j$  上的执行时间为:

$$T_{i,j}^{UAV} = \frac{S_i \times D_i}{f_j^{UAV}} \quad (5)$$

其中, $f_j^{UAV}$  表示  $UAV_j$  的 CPU 计算频率。

当一个任务执行完成后,执行结果需要被回传到相应的 MD 上。由于回传的数据量非常小,该传输时间  $T_{i,j}^{back}$  通常可以忽略不计。

综上所述,当  $MD_i$  将任务卸载到  $UAV_j$  上执行时,任务完成时间可表示为:

$$T_{i,j}^{off} = T_{i,j}^{tran} + T_{i,j}^{UAV} \quad (6)$$

#### 3.3 问题定义

基于上述分析,为了最小化多无人机使能的 MEC 系统内所有 MDs 的平均任务响应时间,将优化问题定义如下:

$$\begin{aligned}
P1: \min & \frac{1}{n} \sum_{i=1}^m (\alpha_{i,n+1} T_i^{\text{loc}} + \sum_{j=1}^n \alpha_{i,j} T_{i,j}^{\text{off}}), \forall i \in [1, m], \\
& \forall j \in [1, n] \\
\text{s. t. } & C1: 0 \leq X_j^{\text{UAV}} \leq X_{\max}, \forall j \in [1, n] \\
& C2: 0 \leq Y_j^{\text{UAV}} \leq Y_{\max}, \forall j \in [1, n] \\
& C3: \alpha_{i,j} \in \{0, 1\}, \forall i \in [1, m], \forall j \in [1, n+1] \\
& C4: \sum_{j=1}^{n+1} \alpha_{i,j} = 1, \forall i \in [1, m] \\
& C5: 0 \leq \sum_{j=1}^m \alpha_{i,j} \leq N_{\max}, \forall j \in [1, n]
\end{aligned} \quad (7)$$

其中, C1 和 C2 表示 UAVs 的部署范围约束, C3 表示任务采用完全卸载策略, C4 表示每个 MD 只能将任务卸载到一台 UAV 上或在本地执行, C5 表示每台 UAV 可执行的任务数量不得超过最大任务并发数。

## 4 双层嵌套联合优化方法

### 4.1 问题分析

上述定义的优化问题是一个混合非线性规划问题, 其目标函数和约束条件 C3 具有非凸性, 这使得该问题难以直接求解。PSO 算法作为一种常用的群智能搜索算法, 不利用函数的梯度信息。因此, PSO 算法对函数的连续性和可导性没有要求, 可用于求解上述优化问题。但是, 传统的 PSO 算法在解决过程中存在以下问题:

(1) 该问题涉及 UAVs 部署和计算卸载这两方面的优化, 但传统的 PSO 算法难以求解此类多参数耦合的优化问题。

(2) 传统的 PSO 算法容易陷入局部最优, 因此可能无法得到最优或者接近最优的解。

为了应对上述问题, 本文通过整合 PSO-GA 算法和贪心算法, 提出了一种 PSO-GA-G 双层嵌套联合优化方法。在每一次迭代中, 该方法的外层通过 PSO-GA 算法实现对 UAVs 部署位置的优化, 而该方法的内层则是通过贪心算法实现对计算卸载的优化。

### 4.2 基于 PSO-GA 算法的 UAVs 部署优化

PSO-GA 是一种引入遗传算法 (Genetic Algorithm, GA) 更新算子的改进后的粒子群优化算法。该算法继承了 PSO 算法易实现、精度高和收敛快等特点, 并通过利用 GA 算法的更新算子, 使得该算法能够有效避免局部最优, 从而获得更优的解。

#### 4.2.1 粒子编码

一种好的编码方式可以提高算法的执行效率, 但是传统的编码方式 (如二进制编码和整数编码) 很难对问题 P1 的可行解进行表示。在所提模型中, UAVs 的部署位置由其  $x$  轴和  $y$  轴坐标表示, 因此采用了以下粒子编码机制: 每个粒子代表系统中所有 UAVs 的一种部署方案, 粒子的每个分位由一个二维向量进行编码, 表示一台 UAV 在水平方向上的部署位置。在  $t$  时刻, 粒子  $k$  的位置可表示为:

$$X_k^t = (x_{k1}^t, x_{k2}^t, \dots, x_{kn}^t) \quad (8)$$

其中,  $x_{kj}^t$  ( $k=1, 2, \dots, n$ ) 表示 UAV $_j$  在  $t$  时刻的水平坐标, 定义如下:

$$x_{kj}^t = (X_{kj}^t, Y_{kj}^t) \quad (9)$$

其中,  $X_{kj}^t$  和  $Y_{kj}^t$  分别表示在  $t$  时刻 UAV $_j$  的  $x$  轴和  $y$  轴坐标。

图 2 给出了一个粒子编码的例子。该粒子由 6 个分位构成, 分别代表了 6 台 UAVs 的部署位置。其中, 每个分位

由一个二维向量构成, 表示某台 UAV 在水平方向上的坐标。例如, 1 号分位的编码为 (123, 643), 这表示 1 号 UAV 的部署位置为 (123, 643,  $H$ )。

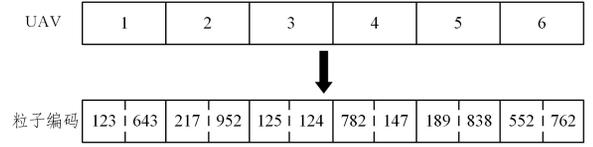


图 2 粒子编码的例子

Fig. 2 Example of particle coding

#### 4.2.2 适应度函数

适应度函数是用来评估粒子的优劣。在求解上述优化问题时, 式(7)中的目标函数将作为 PSO-GA 算法的适应度函数。因此, 将 UAVs 部署方案和计算卸载方案代入式(7)中的目标函数, 可得到一个粒子的适应度函数的值。由于本文的优化目标是 minimized 平均任务响应时间, 因此适应度函数值越小的粒子展现出更好的性能。

#### 4.2.3 粒子更新策略

在传统的 PSO 算法中, 每个粒子以一定的方向和速度在整个问题空间范围内移动。粒子是由自身的位置和速度决定的, 它们会根据周围粒子和自身经验在问题搜索空间中不断迭代更新调整自己的位置和速度。在这个过程中, 粒子的速度和位置的更新方式如下:

$$V_k^{t+1} = \omega V_k^t + c_1 r_1 (pBest_k - X_k^t) + c_2 r_2 (gBest - X_k^t) \quad (10)$$

$$X_k^{t+1} = V_k^{t+1} + X_k^t \quad (11)$$

其中,  $t$  代表当前迭代次数,  $V_k^t$  和  $X_k^t$  分别表示第  $k$  个粒子在第  $t$  次迭代时粒子的速度和位置,  $pBest_k$  和  $gBest$  分别表示经过  $t$  轮迭代后第  $k$  个粒子的历史最优位置和整个种群的最优位置。 $\omega$  是惯性因子, 决定了前一次迭代的速度对当前代移动速度的影响;  $c_1$  和  $c_2$  分别称为个体学习因子和社会学习因子, 反映了对自身历史最优值和种群历史最优值的学习能力。 $r_1$  和  $r_2$  是  $[0, 1]$  区间内的两个随机值, 为迭代搜索过程增添随机性。

PSO-GA 算法在 PSO 算法的基础上引入了 GA 算法中的交叉和变异算子。相应地, 粒子  $k$  的更新方式变更为:

$$X_k^{t+1} = c_2 \oplus C_g (c_1 \oplus C_p (\omega \oplus M_u (X_k^t), pBest_k), gBest) \quad (12)$$

其中,  $M_u()$  表示变异算子,  $C_p()$  和  $C_g()$  表示交叉算子。

在 PSO-GA 算法中, 传统 PSO 算法的惯性部分将与 GA 算法的变异思想相结合, 粒子的惯性部分更新方式为:

$$A_k^t = \omega \oplus M_u (X_k^{t-1}) = \begin{cases} M_u (X_k^{t-1}), & r_1 < \omega \\ X_k^{t-1}, & \text{其他} \end{cases} \quad (13)$$

其中,  $r_1$  是  $[0, 1]$  区间内的随机值。  $M_u (X_k^{t-1})$  表示随机选取粒子  $X_k^{t-1}$  上的某一分位, 并在阈值范围内随机变异。图 3 给出了对粒子执行变异算子的结果。在这个过程中, 变异算子随机选择了一个分位 mp1, 并将 mp1 位上的编码从 (217, 952) 随机变更为 (345, 996)。

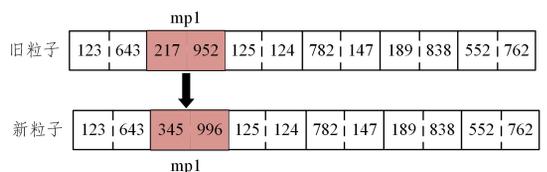


图 3 变异操作

Fig. 3 Mutation operation

在 PSO-GA 算法中,传统的个体学习部分与社会学习部分和 GA 算法的交叉算子相结合,相应的更新方式分别为:

$$B_k^i = \omega \oplus C_p(A_k^i, pBest_k) = \begin{cases} C_p(A_k^i, pBest_k), & r_2 < c_1 \\ A_k^i, & \text{其他} \end{cases} \quad (14)$$

$$C_k^i = \omega \oplus C_g(B_k^i, gBest) = \begin{cases} C_g(B_k^i, gBest), & r_3 < c_2 \\ B_k^i, & \text{其他} \end{cases} \quad (15)$$

其中,  $r_2$  和  $r_3$  是  $[0, 1]$  区间内的两个随机值。交叉算子随机选取粒子上的两个分位,并用  $pBest_k$  或  $gBest$  上对应分位之间的数值进行替换。

图 4 给出了对粒子执行交叉算子的结果。在这个过程中,交叉算子在旧粒子上随机选择了两个分位 cp1 和 cp2,并用  $pBest_k$ (或  $gBest$ )对应分位之间的 3 个二维向量,即 (342, 432), (121, 241) 和 (720, 390), 进行替换。

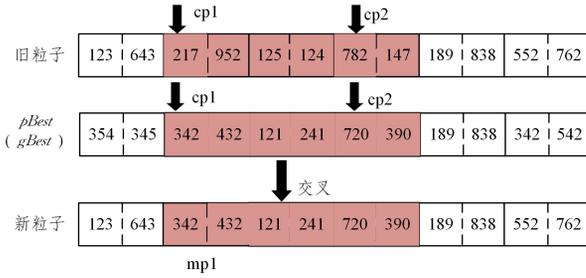


图 4 交叉操作

Fig. 4 Crossover operation

#### 4.2.4 参数设置

类似于传统 PSO 算法,式(12)中的惯性权重因子  $\omega$  能够决定算法的收敛速度和搜索能力。当  $\omega$  的取值较大时,PSO-GA 算法中粒子以较大概率发生变异,因此算法具有较强的全局搜索能力;当  $\omega$  的取值较小时,PSO-GA 算法发生变异的概率较小,此时算法具有较好的局部搜索能力。

算法执行的早期更注重问题空间搜索的多样性,随着搜索的深入,后期应更加注重局部搜索的能力。因此,算法的惯性权重因子  $\omega$  应该随着迭代次数的增加而逐渐减小。本文采用的惯性权重因子调整策略为:

$$\omega = \omega_{\max} - iters_{\text{cur}} \times \frac{\omega_{\max} - \omega_{\min}}{iters_{\max}} \quad (16)$$

其中,  $\omega_{\max}$  和  $\omega_{\min}$  分别是  $\omega$  设定的最大值和最小值,  $iters_{\max}$  表示最大迭代次数,  $iters_{\text{cur}}$  表示当前迭代次数,因此  $\omega$  表现为随着迭代次数的增加而线性递减。

此外,对于式(12)中的个体学习因子  $c_1$  和社会学习因子  $c_2$ ,本文采用了如下类似的线性调整策略:

$$c_1 = c_{1_{\text{start}}} - iters_{\text{cur}} \times \frac{c_{1_{\text{start}}} - c_{1_{\text{end}}}}{iters_{\max}} \quad (17)$$

$$c_2 = c_{2_{\text{start}}} - iters_{\text{cur}} \times \frac{c_{2_{\text{start}}} - c_{2_{\text{end}}}}{iters_{\max}} \quad (18)$$

其中,  $c_{1_{\text{start}}}$  和  $c_{2_{\text{start}}}$  分别表示参数  $c_1$  和  $c_2$  在迭代开始前的初始值,  $c_{1_{\text{end}}}$  和  $c_{2_{\text{end}}}$  分别表示参数  $c_1$  和  $c_2$  在迭代中的最终值。

#### 4.2.5 算法实现

基于 PSO-GA 的外层部署优化算法如算法 1 所示。

##### 算法 1 PSO-GA 算法

/\* 基于 PSO-GA 算法寻找最优部署方案 \*/

输入:  $(P^{\text{MD}}, S_i, D_i, I_i^{\text{MD}}, I_i^{\text{UAV}}, n)$

输出:  $(P^{\text{UAV}}, T_{\text{avg}})$

1. 初始化初代种群  $\text{Pop}^0$
2. 根据算法 2 获得任务卸载矩阵  $\mathbf{A}$
3. 根据适应度函数计算  $\text{Pop}^0$ . Fitness
4. /\* 初始化历史最优值 \*/
5.  $\min \leftarrow \text{Pop}^0$ . Fitness
6. for  $i \leftarrow 0$  to Size:
7.  $pBest_i \leftarrow \text{Pop}_i^0$
8. if  $pBest_i$ . Fitness  $<$  min:
9.  $\min \leftarrow pBest_i$ . Fitness
10.  $gBest \leftarrow pBest_i$
11. end if
12. end for
13. /\* 迭代求解 \*/
14. for  $iters \leftarrow 0$  to  $iters_{\max}$ :
15. 更新参数  $\omega, c_1, c_2$
16. for  $i \leftarrow 0$  to Size:
17. 粒子  $\text{Pop}_i^{\text{iters}}$  执行变异
18. 粒子  $\text{Pop}_i^{\text{iters}}$  执行交叉
19. 根据算法 2 获得卸载矩阵  $\mathbf{A}$
20. 计算  $\text{Pop}_i^{\text{iters}}$ . Fitness
21. if  $\text{Pop}_i^{\text{iters}}$ . Fitness  $<$   $pBest_i$ . Fitness:
22.  $pBest_i \leftarrow \text{Pop}_i^{\text{iters}}$
23. end if
24. if  $\text{Pop}_i^{\text{iters}}$ . Fitness  $<$   $gBest$ . Fitness:
25.  $gBest \leftarrow \text{Pop}_i^{\text{iters}}$
26. end if
27. end for
28. end for
29. /\* 输出最优解和平均响应时间 \*/
30.  $T_{\text{avg}} \leftarrow gBest$ . Fitness
31.  $P^{\text{UAV}} \leftarrow gBest$

根据算法 1,下面给出该算法的具体步骤。

第 1 步 初始化初代种群  $\text{Pop}^0$ , 包括种群规模  $\text{Size}$ 、初代粒子的局部最优解  $pBest_i$  和种群全局最优解  $gBest$ , 如算法第 1—12 行。

第 2 步 迭代求解,通过交叉和变异操作,对各粒子进行变异,然后求解粒子的适应度函数值  $\text{Fitness}$ ,根据  $\text{Fitness}$  更新  $pBest_i$  和  $gBest$ ,如算法第 13—28 行。

第 3 步 输出最优解和系统平均响应时间,如算法第 29—31 行。

#### 4.3 基于贪心算法的计算卸载优化

4.2 节介绍了基于 PSO-GA 的 UAVs 部署算法。为了获得问题 P1 的完整解,针对 PSO-GA 得到的部署方案,还需要对 MDs 的计算任务进行合理的卸载决策。现有研究表明,计算卸载问题是 NP 难的。本节介绍一种基于贪心的卸载算法,可以快速地实现对计算卸载的优化。

贪心算法的思想是,在求解问题时,总是做出当前看起来最优的选择。根据 3.2 节介绍的信道增益模型,可以得一个结论:当 UAVs 和 MDs 之间的距离越少时,通信链路之间的信道增益功率越大,任务卸载所需的时间越少。因此,把 MDs 的计算任务卸载到距离最近的 UAV 上能大大减少卸载时间。

然而,由于 UAV 的并行计算能力是有限的,只能为一定数量的 MDs 提供卸载服务;同时,对于一些零散的 MDs 来

说,即使是卸载到最近的 UAV 所需时间也相对较长。因此,本文的提出的贪心策略表述如下:

(1)总是尝试将计算任务卸载到最近的 UAV 上;

(2)对于  $MD_i$ ,如果该计算任务在本地的执行时间少于卸载到最近 UAV 上的执行时间,该任务在本地执行;

(3)当  $MD_i$ 尝试将计算任务卸载到 UAV<sub>j</sub>上执行时,若当前已有  $N_{\max}$ 个 MDs 尝试将任务卸载到该 UAV 上时,则拒绝这些 MDs 中距离该 UAV 最远 MD 的卸载请求并使其在本地执行。

基于贪心策略的外层卸载优化算法如算法 2 所示。

## 算法 2 贪心算法

/\* 基于贪心策略求解卸载方案 \*/

输入:  $(P^{UAV}, P^{MD}, S_i, D_i, f_i^{MD}, f_j^{UAV})$

输出: 卸载矩阵  $A$

```

1. 初始化  $\alpha_{i,j} \leftarrow 0$ 
2. 计算  $MD_i$ 和  $UAV_j$ 之间的距离  $d_{i,j}$ 
3. for  $i \leftarrow 0$  to  $M$ :
4.  $d_i \leftarrow \inf$ 
5.  $index \leftarrow 0$ 
6. for  $j \leftarrow 0$  to  $N$ :
7. if  $d_i > d_{i,j}$ :
8.  $d_i \leftarrow d_{i,j}$ 
9.  $index \leftarrow j$ 
10. end if
11. end for
12. 根据式(3)计算  $T_i^{loc}$ 
13. 根据式(6)计算  $T_i^{off}$ 
14. if  $T_i^{loc} \leq T_i^{off}$ :
15.  $\alpha_{i,N+1} \leftarrow 1$ 
16. else:
17.  $\alpha_{i,index} \leftarrow 1$ 
18. if  $\text{Sum}(A[index]) > N_{\max}$ :
19. 搜索距离  $UAV_{index}$ 最远且  $\alpha_{k,index} = 1$ 的  $MD_k$ 
20.  $\alpha_{k,index} \leftarrow 0$ 
21.  $\alpha_{k,N+1} \leftarrow 1$ 
22. end if
23. end if
24. end for

```

根据算法 2,下面给出算法的具体步骤。

第 1 步 初始化卸载矩阵,如算法第 1 行。

第 2 步 计算  $MD_i$ 到各 UAV 的距离,并记录距离各 MD 最近的 UAV,如算法第 2—11 行。

第 3 步 计算  $MD_i$ 在本地执行和卸载到最近 UAV 所花费的时间,判断卸载是否有利,如算法第 12—15 行。

第 4 步 判断 UAV 是否达到服务上限,如果还没达到上限,则可以继续为 MD 提供服务,如果达到上限,则搜索距离最远的 MD 并使其在本地执行,如算法第 16—22 行。

## 5 实验仿真与评估

本节设置了 4 种环境不同的实验场景,对所提 PSO-GA-G 算法进行评估。

### 5.1 实验设置

在本实验中,设定实验区域为  $1000 \times 1000 \text{ m}^2$ 的矩形区域,在该区域内共有  $M=100$ 个 MDs 以某种形式不均匀地

分布<sup>[29]</sup>,地面用户的计算任务量服从  $10 \sim 20 \text{ Mbit}$  范围内的随机分布,MDs 的本地计算频率为  $1 \text{ GHz}$ 。为了向区域内的 MDs 提供网络服务,在该区域内部署了  $N=10$  台异构多核的 UAVs 作为边缘服务器,允许各 UAV 并行地为多个 MDs 提供卸载服务。各 UAV 的 CPU 计算频率服从  $2.5 \sim 3.5 \text{ GHz}$  范围内的随机分布。PSO-GA 算法的相关参数参考文献<sup>[30]</sup>所设置。其他的系统参数设定如表 2 所列<sup>[29]</sup>。

表 2 系统参数设定

Table 2 System parameters

系统参数	值
$H/\text{m}$	20
$B/\text{MHz}$	10
$P/\text{W}$	1
$\sigma^2/\text{dBm}$	$-1 \times 10^{-5}$
$g_0/\text{dB}$	-20
$D/\text{cycle}$	100
$N_{\max}$	10

在 4 种场景中,以上的设置保持一致,各场景间的区别在于区域内 MDs 的分布规律不同。

场景 1 是一个典型的 MDs 分布场景。在该场景中,整个实验区域内存在一个 MDs 大量聚集的局部区域(如超市、医院等),整个环境内 90% 的 MDs 都集中分布在该局部区域,仅有 10% 的 MDs 随机地分布在其他区域。该场景下 MDs 的分布状况如图 5(a)所示。

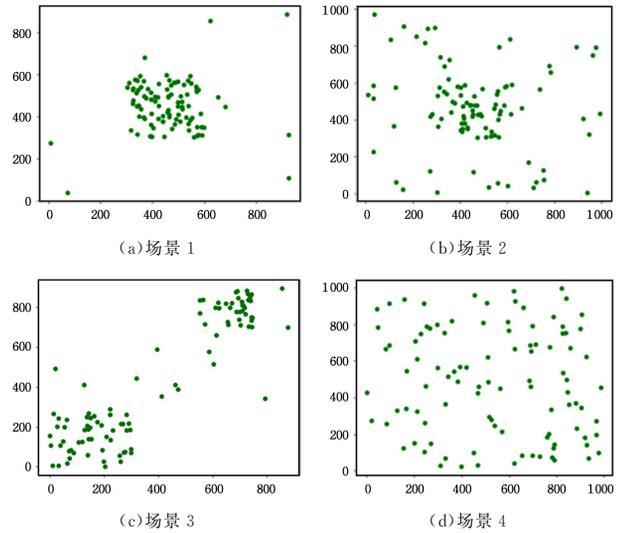


图 5 MDs 分布示意图

Fig. 5 Mobile devices distribution

在场景 2 中,实验区域内同样存在一个 MDs 聚集的局部区域,与场景 1 不同的是,有 50% 的 MDs 集中分布在该区域,剩下 50% 的 MDs 随机分布在其他区域,即聚集的程度较场景 1 更低。该场景下 MDs 的分布状况如图 5(b)所示。

在场景 3 中,实验区域内存在两个 MDs 中等程度聚集的区域(如工厂中的两个生产车间),区域 1 中聚集了 50% 的 MDs,区域 2 中聚集了 35% 的 MDs,剩余的 15% 在其他的区域随机分布。该场景下 MDs 的分布状况如图 5(c)所示。

在场景 4 中不存在任何局部聚集区域,所有的 MDs 均随机地分布在整个实验区域内(如公园、步行街)。该场景下 MDs 的分布状况如图 5(d)所示。

上述 4 种场景间的对比如下:

(1)场景1和场景2的区别在于,场景1的MDs局部聚集程度更高。

(2)场景1和场景3总体上聚集的MDs数量相差不大,主要的差别在于,场景1中仅有一个聚集区,而场景3中存在两个聚集区域,每个区域的聚集程度较场景1更低。

(3)场景2和场景3的区别在于,场景2仅有一个聚集区,且聚集的总人数较少。

(4)与其他场景不同,场景4中不存在MDs集中的局部区域,MDs的分布情况相对均匀。

## 5.2 对比算法

由于本文所考虑的MEC系统是一个多UAV多MD的中等规模通信网络,并且UAVs的部署范围为一个连续空间,因此无法使用穷举之类的方法获得原问题的最优解。为了验证所提出的PSO-GA-G算法的有效性,通过改写一种基于随机算法Random、文献[31]提出的聚类算法K-means和文献[26]提出的离散粒子群优化算法PSO,使其适用于本文所提出的问题,作为PSO-GA算法的对比算法。

随机算法是相关研究中最常用的基准算法之一。改写后的随机算法记为RAN-G,在UAVs的部署优化方面采用随机部署的策略,在任务卸载方面则采用和所提算法一致的贪心策略。该算法的特点是:将UAVs部署的计算卸载看作两个完全独立的过程,且在部署UAVs时忽略了MDs的位置和计算任务大小对结果的影响。

以K-means为代表的聚类算法是一种常用的UAVs部署方案。改写后的K-means-G算法中,在UAVs部署优化方面采用K-means算法,在任务卸载方面采用和所提算法一致的贪心策略。该算法的特点是:将UAVs部署和计算卸载看作两个完全独立的过程,在部署UAVs时忽略了用户的计算任务对系统响应时间的影响。

以PSO为代表的群智能算法同样是一种常用的UAVs部署方案。改写后的PSO-G算法中,在UAVs部署方面采用传统的PSO算法实现,在任务卸载方面采用和所提算法一致的贪心策略。该算法的特点是:综合考虑UAVs部署和任务卸载两部分,在每一轮迭代中,先通过PSO获得部署方案,接着通过贪心算法获得卸载策略,然后更新粒子执行下一轮迭代。

## 5.3 结果评价

为了测试所提出的PSO-GA-G算法、基准算法RAN-G、K-means-G和PSO-G算法的性能,针对每一个场景各进行了50次重复实验,并采用求取平均值的方式减小实验中各波动的随机变量对结果的影响,结果如图6所示。

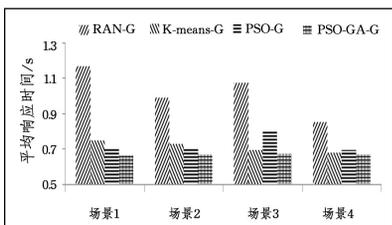


图6 系统平均响应时间

Fig. 6 Average system response time

从实验结果来看,在本文提出的4种不同的MDs分布场景中,所提出的PSO-GA-G联合优化算法均表现出最好的

优化效果。在典型的场景1中,所提出的PSO-GA-G算法明显优于另外3种基准算法,PSO-G算法次之,K-means-G算法再次之,RAN-G算法效果最差。造成这种现象的原因在于,基于K-means的优化算法总是先根据区域内MDs的分布状况部署UAVs,然后再根据部署方案决定卸载策略,因而在部署UAVs的过程中该方法忽略了计算任务对系统响应时间的影响,无法对给定的部署方案进行及时的评估;PSO-G和PSO-GA-G算法作为嵌套的联合优化算法,在每一轮迭代中综合考虑了UAVs部署位置和MDs卸载方案对系统响应时间的影响,因此拥有更好的性能。该场景充分证明了嵌套的联合算法的优越性。

在场景2中,PSO-GA-G的优化效果依然表现最好,同时K-means-G和PSO-G的优化效果与PSO-GA-G的差距相对场景1有所减小。出现这种现象是因为在场景1中,MDs的聚集程度相对较高,基于K-means的优化算法由于总是将区域内的MDs划分为 $n$ 个簇并在质心处放置UAVs,因此在周边零散点处K-means-G算法同样会部署UAVs来为MDs提供卸载机会,这就导致在零散点处UAVs的资源利用率不高,而在MDs密集区超出了UAVs的服务能力;基于PSO-GA和PSO的联合优化算法则会避免在零散点处部署UAVs,因此资源利用率更高。而对于场景2,由于MDs聚集的程度降低,K-means-G算法得到的方案在资源利用率方面相对提高,该算法的性能有了明显的提升。

在场景3中,本文提出的PSO-GA-G算法表现最优,K-means-G算法次之,且PSO-GA-G和K-means-G明显优于另外两种算法。相比前两种场景,PSO-GA-G算法得到的系统平均响应时间基本保持不变,而PSO-G算法得到的系统平均响应时间明显增大,这是由于在场景3中存在两个MDs密集区,基于PSO的优化方案容易陷入局部最优,而基于PSO-GA的方案由于引入了遗传算法中的变异交叉算子,有效地跳出局部最优解。场景3充分说明了引入了遗传算法算子的PSO-GA算法较传统的PSO算法具有更强的全局搜索能力。

在场景4中,本文提出的PSO-GA-G算法、K-means-G算法以及PSO-G算法都有较好的优化效果。

上述4种优化策略在各场景中的系统平均响应时间如表3所列。可以看到,当各景内MDs和UAVs的数量保持一致时,本文提出的PSO-GA-G算法在不同的MDs分布场景中不仅拥有最好的优化效果,同时在不同的用户分布场景中性能基本保持不变,具有通用性;基于PSO的优化算法在有二个或者多个MD聚集区的场景中容易收敛到局部最优解;基于K-means的优化算法在MDs分布较为均匀的场景中拥有较好的性能,但是随着区域内MDs聚集程度的提升,该优化算法的性能随之降低,因而也不具有通用性。

表3 系统平均响应时间

Table 3 Average system response time

(单位:s)

算法	场景1	场景2	场景3	场景4
RAN-G	1.1698	0.9857	1.0735	0.8537
K-means-G	0.7486	0.7275	0.6932	0.6810
PSO-G	0.7027	0.7038	0.7981	0.6929
PSO-GA-G	0.6666	0.6703	0.6683	0.6756

从图7可以看到,随着部署的UAVs数量的增加,各种算法下系统的平均响应时间都随之降低,这是因为部署的

UAVs 数量越多, MDs 的计算任务卸载的机会就越大, 同时 MDs 和提供服务的 UAVs 之间的距离也会减小, 因此任务的卸载时间也相应地减少。

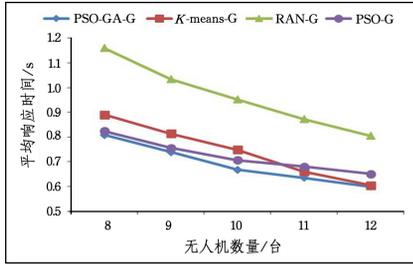


图7 无人机数量对系统平均响应时间的影响

Fig. 7 Average system response time with different number of UAVs

另外, 可以看到, 本文提出的 PSO-GA-G 算法在以上条件下拥有较好的性能, 而 K-means-G 算法在部署的 UAVs 数量较少时效果较差, 但是随着所部署的 UAVs 数量的增加, 其性能有大幅度改善。此外, 对于所提出的 PSO-GA-G 算法, 系统响应时间的减少和无人机数量的增加并非线性关系, 这是因为当  $N < 10$  时, UAVs 的计算资源相对不足, 增加 UAVs 可以明显提高系统性能; 当  $N > 10$  时, UAVs 的计算资源相对过剩, 再增加 UAVs 并不能明显改善系统性能。另外, 相关研究表明, 增加部署的 UAVs 数量会提高系统能耗<sup>[28]</sup>, 因此需要对系统内的 UAVs 数量进行合理规划。

最后, 图 8 给出了在最大迭代次数为 1000 时, 不同无人机数量下, 所提出的 PSO-GA-G 算法的收敛性。从图中可以看到, 在部署了不同数量无人机的条件下, 所提出的 PSO-GA-G 算法均能有效地收敛到稳定值。另外, 从图中还可以看到, 当部署的 UAVs 数量较少时, 所提出的 PSO-GA-G 算法收敛的速度较快; 当部署的 UAVs 数量较多时, 算法需要更多的迭代次数才能收敛到稳定值。

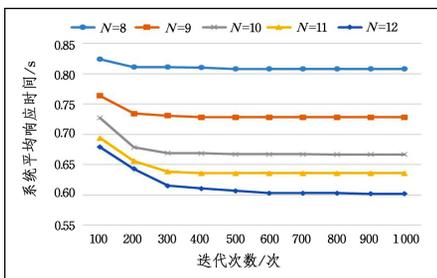


图8 不同 UAVs 数量下 PSO-GA-G 算法的收敛性

Fig. 8 Convergence of PSO-GA-G with different numbers of UAVs

**结束语** 在多无人机使能的移动边缘计算系统中, 通过部署多台无人机作为边缘服务器, 为地面用户提供计算卸载机会, 可以有效减少地面用户计算任务的响应时间。本文提出了一种基于 PSO-GA 算法和贪心策略的双层嵌套联合优化算法 PSO-GA-G, 可以同时解决无人机部署和计算卸载问题, 并实现系统内各用户的平均响应时间最小化。然后, 通过将 PSO-GA-G 算法和常用的 3 种基准算法进行比较, 验证了所提算法的性能。实验结果表明, 所提算法在不同的场景中具有通用性, 且表现出的性能优于其他 3 种基准算法。

尽管本文提出的算法能够有效地实现对系统平均响应时间的优化, 但是由于算法的局限性, 所提算法在执行时间上

略显不足, 且贪心算法不能得到问题的最优解。未来的研究方向主要有两点: 1) 针对算法的不足之处继续研究和改进算法, 提高算法的性能; 2) 结合对系统能耗方面的研究工作, 对该系统模型中的系统能耗、时延和资源分配的优化问题做进一步的探讨。

## 参考文献

- [1] XIAO H, HU Z, YANG K, et al. An Energy-Aware Joint Routing and Task Allocation Algorithm in MEC Systems Assisted by Multiple UAVs[C]// 2020 International Wireless Communications and Mobile Computing(IWCMC). 2020: 1654-1659.
- [2] PORAMBAGE P, OKWUIBE J, LIYANAGE M, et al. Survey on multi-access edge computing for Internet of things realization [J]. IEEE Communications Surveys & Tutorials, 2018, 20(4): 2961-2991.
- [3] GUO H Z, LIU J J, ZHANG J. Computation Offloading for Multi-Access Mobile Edge Computing in Ultra-Dense Networks [J]. IEEE Communications Magazine, 2018, 56(8): 14-19.
- [4] HUANG G, MA Y, LIU X, et al. Model-Based Automated Navigation and Composition of Complex Service Mashups[J]. IEEE Transactions on Services Computing, 2015, 8(3): 494-506.
- [5] ZHANG T K, XU Y, LOO J, et al. Joint Computation and Communication Design for UAV-Assisted Mobile Edge Computing in IoT[J]. IEEE Transactions on Industrial Informatics, 2020, 16(8): 5505-5516.
- [6] NGUYEN V, KHANH T T, VAN NAM P, et al. Towards Flying Mobile Edge Computing[C]// 2020 International Conference on Information Networking(ICOIN). 2020: 723-725.
- [7] XU J, ZENG Y, ZHANG R. UAV-Enabled Wireless Power Transfer: Trajectory Design and Energy Optimization[J]. IEEE Transactions on Wireless Communications, 2018, 17(8): 5092-5106.
- [8] SPINELLI F, MANCUSO V. Towards Enabled Industrial Verticals in 5G: A Survey on MEC-Based Approaches to Provisioning and Flexibility[J]. IEEE Communications Surveys & Tutorials, 2021, 23(1): 596-630.
- [9] SHI W, JIE C, QUAN Z, et al. Edge Computing: Vision and Challenges[J]. Internet of Things Journal, IEEE, 2016, 3(5): 637-646.
- [10] FLORES H, HUI P, TARKOMA S, et al. Mobile code offloading: from concept to practice and beyond[J]. IEEE Communications Magazine, 2015, 53(3): 80-88.
- [11] BI S, ZHANG Y J. Computation Rate Maximization for Wireless Powered Mobile-Edge Computing with Binary Computation Offloading[J]. IEEE Transactions on Wireless Communications, 2018, 17(6): 4177-4190.
- [12] NING Z, DONG P, KONG X, et al. A Cooperative Partial Computation Offloading Scheme for Mobile Edge Computing Enabled Internet of Things[J]. IEEE Internet of Things Journal, 2019, 6(3): 4804-4814.
- [13] LI Y, XU G, GE J, et al. Jointly Optimizing Helpers Selection and Resource Allocation in D2D Mobile Edge Computing[C]// 2020 IEEE Wireless Communications and Networking Conference(WCNC). IEEE, 2020: 1-6.
- [14] SALEEM U, LIU Y, JANGSHER S, et al. Latency Minimization for D2D-Enabled Partial Computation Offloading in Mobile Edge

- Computing[J]. IEEE Transactions on Vehicular Technology, 2020, 69(99):4472-4486.
- [15] CAO X, WANG F, XU J, et al. Joint Computation and Communication Cooperation for Energy-Efficient Mobile Edge Computing[J]. IEEE Internet of Things Journal, 2019, 6(3):4188-4200.
- [16] HAN Y, ZHAO Z, MO J, et al. Efficient Task Offloading with Dependency Guarantees in Ultra-Dense Edge Networks[C]// 2019 IEEE Global Communications Conference(GLOBECOM). IEEE, 2020.
- [17] ZHANG W, LI L, ZHANG N, et al. Air-Ground Integrated Mobile Edge Networks: A Survey[J]. IEEE Access, 2020, 8:125998-126018.
- [18] CHEN R, CUI L, ZHANG Y, et al. Delay Optimization with FCFS Queuing Model in Mobile Edge Computing-Assisted UAV Swarms: A Game-Theoretic Learning Approach[C]// 2020 International Conference on Wireless Communications and Signal Processing(WCSP). 2020.
- [19] ZHANG K, GUI X, REN D, et al. Energy-Latency Tradeoff for Computation Offloading in UAV-assisted Multi-Access Edge Computing System[J]. IEEE Internet of Things Journal, 2021, 8(8):6709-6719.
- [20] KIM K, YU M P, HONG C S. Machine Learning Based Edge-Assisted UAV Computation Offloading for Data Analyzing[C]// 2020 International Conference on Information Networking (ICOIN). 2020:117-120.
- [21] WANG L, HUANG P, WANG K, et al. RL-Based User Association and Resource Allocation for Multi-UAV enabled MEC[C]// 2019 15th International Wireless Communications and Mobile Computing Conference(IWCMC). IEEE, 2019:741-746.
- [22] SEID A M, BOATENG G O, ANOKYE S, et al. Collaborative Computation Offloading and Resource Allocation in Multi-UAV Assisted IoT Networks: A Deep Reinforcement Learning Approach[J]. IEEE Internet of Things Journal, 2021, 8(15):12203-12218.
- [23] YAO K, XU Y, CHEN J, et al. Distributed Joint Optimization of Deployment, Computation Offloading and Resource Allocation in Coalition-based UAV Swarms[C]// 2020 International Conference on Wireless Communications and Signal Processing(WCSP). 2020:207-212.
- [24] YANG L, YAO H, ZHANG X, et al. Multi-UAV Deployment for MEC Enhanced IoT Networks[C]// 2020 IEEE/CIC International Conference on Communications in China(ICC). IEEE, 2020:436-441.
- [25] YANG L, YAO H, WANG J, et al. Multi-UAV Enabled Load-Balance Mobile Edge Computing for IoT Networks [J]. IEEE Internet of Things Journal, 2020, 7(8):6898-6908.
- [26] ZHANG Y, ZHANG L, LIU C. 3-D Deployment Optimization of UAVs Based on Particle Swarm Algorithm[C]// 2019 IEEE 19th International Conference on Communication Technology (ICCT). IEEE, 2019:954-957.
- [27] HUANG P Q, WANG Y, WANG K, et al. Differential Evolution With a Variable Population Size for Deployment Optimization in a UAV-Assisted IoT Data Collection System[J]. IEEE Transactions on Emerging Topics in Computational Intelligence, 2019, 4(3):324-335.
- [28] ZHANG X, ZHANG J, XIONG J, et al. Energy Efficient Multi-UAV-Enabled Multi-Access Edge Computing Incorporating NOMA[J]. IEEE Internet of Things Journal, 2020, 7(6):5613-5627.
- [29] WANG Y, RU Z Y, WANG K, et al. Joint Deployment and Task Scheduling Optimization for Large-Scale Mobile Users in Multi-UAV-Enabled Mobile Edge Computing[J]. IEEE Transactions on Cybernetics, 2020, 50(9):3984-3997.
- [30] LIN B, GUO W Z, CHEN G L. Scheduling strategy for science workflow with deadline constraint on multi-cloud[J]. Journal on Communications, 2018, 39(1):56-69.
- [31] QU H, ZHANG W, ZHAO J, et al. Rapid Deployment of UAVs Based on Bandwidth Resources in Emergency Scenarios[C]// 2020 Information Communication Technologies Conference(ICTC). 2020:86-90.



**LIU Zhang-hui**, born in 1971, Ph.D, professor, Ph.D supervisor, is a member of China Computer Federation. His main research interests include big data technology and intelligence computation.



**CHEN Zhe-yi**, born in 1991, Ph.D. His main research interests include cloud/edge computing, resource optimization, deep learning, and reinforcement learning.