计算机科学
COMPUTER SCIENCE

空-天-地一体化移动边缘计算系统的部署优化和计算卸载

郑鸿强, 张建山, 陈星

引用本文

郑鸿强, 张建山, 陈星.空-天-地一体化移动边缘计算系统的部署优化和计算卸载[J]. 计算机科学, 2023, 50(2): 69-79.

ZHENG Hongqiang, ZHANG Jianshan, CHEN Xing. Deployment Optimization and Computing Offloading of Space-Air-Ground Integrated Mobile Edge Computing System [J]. Computer Science, 2023, 50(2): 69-79.

计算机科学 COMPUTER SCIENCE
http://www.jsjkx.com
DOI:10.11896/jsjkx.220600057

空-天-地一体化移动边缘计算系统的部署优化和计算卸载

郑鸿强　张建山　陈　星
福州大学数学与计算机科学学院　福州 350116
福建省网络计算与智能信息处理重点实验室　福州 350116
(zhq921579984@163.com)

摘　要　空-天-地一体化的通信技术作为一种新兴的架构,能够有效提高地面终端的网络服务质量,近年来引起了广泛关注。文中研究了一种空-天-地一体化的移动边缘计算系统,其中多台无人机为地面设备提供低延迟的边缘计算服务,近地轨道卫星为地面设备提供无处不在的云计算服务。由于无人机的部署位置和计算任务的卸载方案是影响系统性能的关键因素,因此需要对无人机的部署位置、地面设备与无人机之间的连接关系以及计算任务的卸载比例进行联合优化,实现系统内系统平均任务响应时延最小化。并且,由于形式化定义的联合优化问题是一个混合非线性规划问题,因此设计了一种双层优化算法,在该算法的上层,提出了一种结合了遗传算法算子的粒子群优化算法来优化无人机的部署位置,并在算法的下层采用贪心算法来实现对计算任务卸载方案的优化。大量的数值仿真实验验证了所提算法的可行性和有效性。结果表明,与其他基准算法相比,所提算法能有效降低系统的任务平均响应时延。

关键词:空-天-地一体化网络;移动边缘计算;无人机部署;计算卸载
中图法分类号　TP393

Deployment Optimization and Computing Offloading of Space-Air-Ground Integrated Mobile Edge Computing System

ZHENG Hongqiang,ZHANG Jianshan and CHEN Xing
College of Mathematics and Computer Science,Fuzhou University,Fuzhou 350116,China
Fujian Provincial Key Laboratory of Networking Computing and Intelligent Information Processing,Fuzhou 350116,China

Abstract　As a new architecture,the space-air-ground integrated communication technology can effectively improve the network service quality of ground terminal,and has attracted widespread attention in recent years. This paper studies a space-air-ground integrated mobile edge computing system,in which multiple UAVs provide low-latency edge computing services for ground devices,and low earth orbit satellites provide ubiquitous cloud computing services for ground devices. Since the deployment position of the UAVs and the scheduling scheme of computing tasks are the key factors affecting the performance of the system,the deployment position of the UAVs,the link relationship between the ground terminal and the UAVs,and the offloading ratio of computing tasks need to be jointly optimized to minimize the average task response delay of the system. Since the formally defined joint optimization problem is a mixed nonlinear programming problem,this paper designs a two-layer optimization algorithm. In the upper layer of the algorithm,a particle swarm optimization algorithm that combines genetic algorithm operators is proposed to optimize the deployment position of the UAVs,and the greedy algorithm is used in the lower layer of the algorithm to optimize the computing task offloading scheme. The extensive simulation experiments verify the feasibility and effectiveness of the proposed method. The results show that the proposed method can achieve lower average task response time,compared to other baseline methods.

Keywords　Space-Air-Ground integrated network,Mobile edge computing,Unmanned aerial vehicle deployment,Computation offloading

到稿日期:2022-06-60　返修日期:2022-11-08
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);国家重点研发计划(2017YFB1002000)
This work was supported by the National Natural Science Foundation of China(62072108),Natural Scienc Foundation of Fujian Province for Distinguished Young Scholars(2020J06014) and National Key R&D Program of China(2017YFB1002000).
通信作者:张建山(zhangjs0512@163.com)

1 引言

物联网以及相关科学技术的快速发展使得各式各样的应用业务不断涌现，给人们的学习、生活和工作带来了极大的便利[1]。然而，新兴的计算密集型应用通常对时延具有较强的敏感性，因此在执行时对终端设备的计算能力提出了很高的要求[2]。由于受到现有的硬件技术水平和便携性等因素的限制，终端设备有限的存储和计算资源难以独立满足上述应用的需求，这大大降低了计算任务在终端设备上执行时的性能表现。

移动边缘计算(Mobile Edge Computing，MEC)技术将服务器部署到网络边缘，被认为是应对上述挑战的一种前瞻性技术手段[3]。在 MEC 中，终端设备可以将待执行的计算任务卸载到附近的服务器上，能够克服传统移动云计算(Mobile Cloud Computing，MCC)中因集中式处理而引发的网络拥塞问题，并且传输距离更短，传输时间和能量消耗更少，进而有效提高了服务质量[4]。然而，MEC 服务器的位置通常是固定的，不能根据移动用户的需求灵活改变，这限制了 MEC 的能力[5]。

近年来，随着无人机(Unmanned Aerial Vehicle，UAV)技术的持续发展，UAV 在无线通信领域受到了广泛关注。UAV 已经在一些通信基础设施受限的场景中得到应用，如灾害响应和战场通信等。目前，已经有许多学者提出将 MEC 服务器部署到 UAV 上，利用 UAV 通信技术的优势进一步提升 MEC 的性能[6-7]。与传统的地面通信相比，集成了 UAV 通信技术的 MEC 具有明显的优势：首先，UAV 灵活且易部署，可以不受地面地形的限制为指定区域提供通信服务[8]；其次，UAV 凭借自身特有的移动性，能够根据服务对象的情况动态调整部署位置，就近提供计算服务；最后，空中的 UAV 能够有效避免因障碍物遮挡引起的信道衰落，提供优质的网络通信[9]。总的来说，UAV 与 MEC 的结合可以为用户提供更好的网络服务。因此，UAV 的使用有望在改善 MEC 性能方面发挥重要作用。

在此基础上，空-天-地一体化的云边端混合 MEC 系统开始受到学术界的关注。一般来说，空-天-地一体化系统由空间层网络、空中层网络和地面网络 3 个部分构成[10]。空中层网络由搭载了 MEC 服务器的 UAVs 组成，能够为一定区域内的地面终端设备(Ground Terminal，GT)提供远程无线接入和低延迟边缘计算服务。空间层网络中的近地轨道(Low Earth Orbit，LEO)卫星能够覆盖整个区域，为所有的 GTs 提供无处不在的云计算服务[11]。目前，已有一些工作研究了空间卫星使能的计算卸载机制，然而，这些工作主要集中在空-地合作上，并没有考虑空-天-地协同；此外，关于 UAVs 使能的 MEC 系统的现有工作通常旨在优化系统能耗、能源效率或者吞吐量[12-13]，对时延，特别是系统整体的平均响应时延的研究还较少涉及。

受到上述研究的启发，本文研究了一种空-天-地一体化的云边端混合 MEC 系统，通过联合优化系统中 UAVs 的部署位置 GTs 与无人机之间的连接关系以及计算任务的卸载比例，实现对系统内所有 GTs 计算任务的平均响应时间最小化。本文的贡献如下：

(1)提出了一种空-天-地一体化的云边端混合 MEC 系统，该系统由空间层、空中层和地面层 3 层架构组成，在空中层部署了一些搭载了 MEC 服务器的 UAVs，能够为服务范围内的 GTs 提供边缘计算服务，同时空间层的 LEO 卫星能为所有 GTs 提供无处不在的云计算服务，允许 GTs 将待执行的计算任务卸载到 UAVs 和 LEO 卫星上执行，从而降低计算任务的完成时延，提高 GTs 的服务质量。

(2)通过优化系统内 UAVs 的部署位置、GTs 与 UAVs 之间的关联关系以及计算任务的卸载率，形式化定义了一个带截止时间和服务范围约束的部署位置和计算卸载联合优化问题，以实现对系统内所有 GTs 计算任务的平均响应时间最小化。

(3)针对所定义的联合优化问题，提出了一种双层嵌套联合优化算法 PG-G，在该算法的上层采用结合了遗传算法(Genetic Algorithm，GA)算子的离散粒子群优化算法(Particle Swarm Optimization algorithm combined with Genetic Algorithm operators，PSO-GA)实现对 UAVs 部署位置的优化，在下层基于贪心算法获得对连接关系和卸载比例的优化；然后，通过循环迭代求解该优化问题。

(4)设计了充分且合理的仿真环境，进行了大量的数值仿真实验验证所提出方法的可行性和有效性。实验结果表明，相比其他基准方法，本文提出的 PG-G 算法的平均任务响应时间更短，有效提高了系统性能。

本文第 2 节回顾了相关工作；第 3 节描述了空-天-地一体化的 MEC 系统模型以及相应的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 PG-G 双层嵌套联合优化算法；第 5 节进行了数值仿真实验验证与分析；最后总结全文。

2 相关工作

近年来，MEC 作为一种新兴的计算范式被提出，其能够为无线接入网络边缘的终端用户就近提供云计算服务[14]。计算卸载是 MEC 的一项关键技术，通过将终端设备的计算任务卸载到边缘服务器上执行，能够减少时延和能量消耗，有效提高服务质量。因此，如何设计高效的计算卸载方案是 MEC 系统的重点问题[15]。

根据性能指标的不同，关于卸载方案的现有研究大致可以分为 3 种类型：面向节能的计算卸载、面向时延优化的计算卸载，以及优化时延、能耗加权和的计算卸载[16-18]。在文献[16]中，作者提出了一种利用非正交多址接入(Non-Orthogonal Multiple Access，NOMA)技术进行任务卸载的 MEC 系统，并通过优化发射功率、传输时间分配和任务卸载来最小化总能耗，但该系统忽略了对时延的优化。文献[17]中研究了一个由移动设备和支持各种无线接入技术的异构边缘服务器组成的 MEC 系统，并根据异构边缘服务器的可用带宽和移动设备的位置，将最优卸载节点选择策略形式化定义为马尔可夫决策过程，采用一种价值迭代算法进行求解，实现时延最小化。文献[18]中提出了一种能量感知的卸载方案，该方案能够在有限的能量和敏感的延迟下共同优化通信和计算资源

分配，实现了能耗和时延的平衡。此外，计算任务的卸载方式也是相关研究的一个重点方向，主要分为完全卸载和部分卸载两种模式。完全卸载指将终端的计算任务完整地卸载到服务器上执行，作为一种传统卸载模式，其在大量的现有工作中得到应用。文献[19]提出了一种由无线电力传输驱动的多用户 MEC 网络，其中每个能量收集终端遵循二进制卸载策略，并提出了一种基于坐标下降法和一种交替方向乘子法的优化算法用于解决所提出的优化问题。文献[20]研究了一种采用二进制卸载策略的无线供电 MEC 网络，并提出了一种基于深度强化学习的在线卸载框架，该框架能够从经验中学习二进制卸载决策。部分卸载指采用代码分解等技术将计算任务进行划分，并将部分任务卸载到服务器上执行的技术。与完全卸载策略相比，部分卸载在性能上更有益[21]。文献[22]首先研究了单用户的卸载问题，随后又提出了支持部分卸载的多用户计算卸载问题，并设计了一种迭代的启发式算法来动态地做出决策。

空-天-地一体化的网络架构能够为物联网设备提供无处不在的网络覆盖，近年来受到了学术界的广泛关注[23-25]。文献[23]针对海上通信业务的复杂性，提出了一种支持卫星、UAV、地面基站以及海上快艇的空-天-地-海一体化网络架构，极大地弥补了海上通信资源的不足。文献[24]提出了一种支持空-天-地一体化的车载网络架构，并解决了空-天-地一体化网络与车载网络之间通信的安全、高效和可靠性问题。文献[25]研究了一种民机增强的空-天-地一体化网络，并提出了一种包括资源分配和资源拍卖在内的公平优化策略。

MEC 与空-天-地一体化网络的结合，有望突破传统地面通信的种种局限，进一步提高用户的服务质量，近年来也受到了部分关注。文献[26]研究了一种多物联网设备协同使用计算资源的空-天-地一体化场景下的计算卸载问题。该项工作考虑了物联网设备生成任务的动态性、无人机的移动性以及各端设备的差异性，制定了一个最小化系统任务总时延的优化问题，并将该问题表述为一个马尔可夫决策过程。然而，该项工作中仅采用一台 UAV 作为边缘节点，由于 UAV 的计算资源有限，提出的模型可能无法适用于用户规模较大的场景。文献[27-28]研究了一种支持单卫星、单 UAV、多地面机制和多地面设备的物联网系统，UAV 作为移动的 MEC 服务器能够在飞行的过程中收集地面设备的计算任务并卸载到基站或卫星上执行。然而，这项工作采用完全卸载策略，同时忽略了终端设备的计算能力，因而无法充分利用系统内各类终端的计算资源。在此类研究中，UAVs 部署和任务卸载的联合优化是研究重点。考虑到节点的高移动性以及频繁变化的网络流量和链路状态，Tang 等[29]提出了一种基于强化学习的流量卸载策略，然而，这类基于强化学习的优化算法需要提供大量的数据用于训练模型。文献[30]提出了一种基于块坐标下降法的联合优化算法，能够快速解决在空-天-地一体化网络中的 UAVs 部署、任务调度和资源分配问题。然而，这种基于凸优化理论的算法复杂度会随着问题规模的增大而呈指数型增长，因此更适用于小规模场景。

综上所述，尽管已有一些工作对空-天-地一体化 MEC 系统的关键技术展开了研究，但其中大部分的工作仅考虑了支持单台 UAV 的场景，无法应用于实际的复杂场景；此外，为了简化模型，大多数工作仅采用了二进制的卸载策略，这将导致系统内各类终端的资源利用率较低。针对上述问题，本文提出了一种包含了多个 GTs、多台 UAVs 和 1 台 LEO 卫星的空-天-地一体化 MEC 系统，每个 GTs 可以按需将部分计算任务卸载至 UAV 和 LEO 卫星上执行，剩余部分在本地执行。此外，由于 UAVs 的服务范围有限，并且 UAVs 与 GTs 间的信道质量受到两者间距离的直接影响，因此，与传统云-边-端融合的 MEC 系统不同，本文通过联合优化 UAVs 的部署位置和计算卸载机制来实现系统平均响应时间最小化。

3 系统模型和问题定义

本文研究了一个如图 1 所示的空-天-地一体化的 MEC 系统，该系统由 3 层网络架构组成，即拥有 M 个 GTs 的地面层、拥有 N 台 UAVs 的空中层以及 1 台 LEO 卫星构成的空间层。为了便于分析，本文将 GTs 和 UAVs 的集合分别表示为 $\mathcal{M}=\{1,2,\cdots,M\}$ 和 $\mathcal{N}=\{1,2,\cdots,N\}$。在地面层，具有有限计算能力的 GTs 分布在通信基础设施缺乏的区域（如山区、海洋和受灾地区），并执行一些具有一定计算要求的任务。为了缓解 GT 执行计算任务的压力，考虑到该区域缺乏地面蜂窝网络的覆盖，空中层的 UAVs 可以作为边缘节点，为 GTs 提供边缘计算服务。此外，在空间层，LEO 卫星可以为其覆盖区域内的所有 GTs 提供集中式云计算服务。尽管 GTs 可以在本地执行计算任务，但由于 GTs 的计算能力有限，可能无法独立完成任务，因此，允许 GTs 将其部分计算任务卸载到 UAVs 和 LEO 卫星上执行。为了便于参考，本文在表 1 中列出了使用的符号。

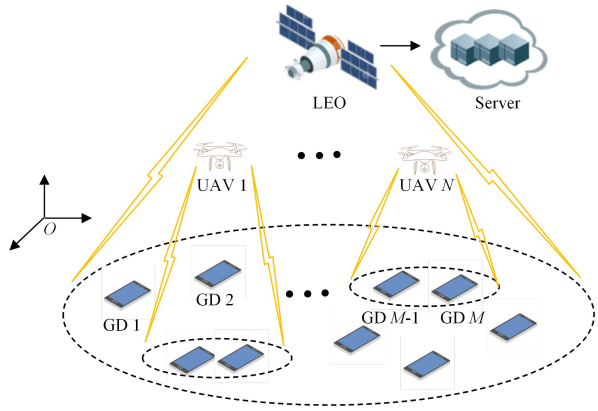

图 1 空-天-地一体化的 MEC 系统

Fig. 1 Apace-Air-Ground integrated MEC system

本文采用三维欧几里德坐标系来表示各地面终端设备和 UAVs 的位置，第 i 个 GT 在当前时隙的位置可以用三维坐标 $u_i^{\mathrm{GT}}=(x_i^{\mathrm{GT}},y_i^{\mathrm{GT}},0)$ 表示，相应地，第 j 个 UAV 的当前位置可以用三维坐标 $u_i^{\mathrm{UAV}}=(x_i^{\mathrm{UAV}},y_i^{\mathrm{UAV}},0)$ 表示。其中，H 表示 UAVs 的飞行高度。需要注意的是，为了简化模型，本文假设在当前时隙内所有的 UAVs 都部署在同一高度。此外，由于本文研究的是准静态场景下的系统性能，这意味着在所考虑的时隙内，所有 UAVs 和 GTs 的位置将保持不变。在本文

中，采用一个四元组 $I_i=\langle D_i,S_i,O_i,E_i\rangle$ 表示第 i 个 GT 在当前时隙内需要执行的计算任务。其中 D_i 表示计算任务 I_i 的输入数据大小(以 bit 为单位)；S_i 表示执行单位计算任务输入量所需的计算资源数(以 cycle/bit 为单位)；O_i 表示任务执行结果的大小(以 bit 为单位)；E_i 表示该计算任务的最大容忍时延(以 second 为单位)。在该定义中，计算任务执行结果 O_i 的大小通常远小于任务的输入量 D_i，因此可以忽略不计，这在许多相关文献中都得到了证明[30]。此外，计算任务 I_i 的完成时延不得超过最大容忍时延 E_i，因为在任务的截止时间之后完成任务是没有意义的。

表 1 符号定义

Table 1 Symbol definitions

符号	定义
$\mathcal{M}$	地面终端的集合
$\mathcal{N}$	无人机的集合
M	移动设备的数量
N	无人机的数量
u^{GT}/u^{UAV}	地面终端/无人机坐标的集合
u_i^{GT}/u_j^{UAV}	第 i 台地面终端/第 j 台无人机坐标
f_i^{GT}/f_j^{UAV}	第 i 台地面终端/第 j 台无人机的计算频率
f^{LEO}	低轨卫星为每个地面终端分配的计算频率
I_i	第 i 台地面终端的计算任务
D_i	计算任务 I_i 的输入数据量
S_i	任务 I_i 中的计算复杂度
O_i	任务 I_i 中的执行结果大小
E_i	任务 I_i 的最大完成时延
H	无人机悬停高度
M_{max}	最大任务并发数
R_{max}	无人机的服务半径
P_i	移动设备 i 的传输功率
h_0	单位距离下的信道增益
B	信道带宽
σ^2	高斯白噪声功率
$h_{i,j}$	第 i 台移动设备和第 j 台无人机间的信道增益
$R_{i,j}$	第 i 台移动设备和第 j 台无人机间的传输速率
T_{avg}	平均任务响应时间

由于 GTs 的计算能力有限，可能无法独立地在 E_i 内完成计算任务，因此需要将计算任务 I_i 卸载到 UAVs 和 LEO 卫星上执行。为充分利用地面层、空中层和空间层各端设备的计算资源，本文采用部分卸载策略，允许 GTs 对其计算任务进行划分并将部分计算任务卸载到 UAVs 和 LEO 卫星上执行，本地执行的部分以及卸载到 UAVs 和 LEO 卫星上执行的子任务所占比例分别用 α_i^{GT}，α_i^{UAV} 和 α_i^{LEO} 表示。与现有的其他工作类似，本文假设采用部分卸载策略不会产生额外的计算任务[19]，因此有：

$$\alpha_i^{GT}+\alpha_i^{UAV}+\alpha_i^{LEO}=1,i\in\mathcal{M} \tag{1}$$

需要注意的是，当第 i 个 GT 不处于任何一台 UAV 的覆盖范围内时，该 GT 不能将部分计算任务卸载到 UAVs 上执行，此时有 $\alpha_i^{UAV}=0$。

根据各个子任务的执行位置不同，本文将原计算任务的执行过程分为 3 种不同模式，即在本地执行的本地计算模式、在 UAV 上执行的边缘计算模式和在 LEO 卫星上执行的云计算模式。由于 3 种计算模式所消耗的计算资源分属于不同的终端，因此这 3 种计算模式可以并行地执行。

(1)在本地计算模式下，定义参数 f_i^{GT} 为第 i 个 GT 本地 CPU 的计算频率，根据计算任务 I_i 以及本地卸载比例 α_i^{GT} 的定义可以得到本地待处理的任务量大小为 $\alpha_i^{GT}D_i$，因此，本地计算所产生的时延可以表示为：

$$T_i^{GT}=\frac{\alpha_i^{GT}D_iS_i}{f_i^{GT}},i\in\mathcal{M} \tag{2}$$

(2)在边缘计算模式下，执行计算任务所产生的时延由 3 个部分构成，即将任务卸载到关联 UAV 上的传输时延、在 UAV 上执行的计算时延，以及将计算结果返回 GT 的回传时延。由于计算结果的大小远小于计算任务的输入量，因此回传时延通常可以忽略不计。

类似于现有工作，本文将第 i 个 GT 与第 j 台 UAV 之间的无线信道建模为视距信道(Line-of-Sigh，LOS)链路模型[26]，该模型的合理性已被高通公司最近的现场实验所验证。因此，第 i 个 GT 和第 j 台 UAV 之间的无线信道功率增益被量化为：

$$h_{ij}=d_{ij}^{-2}h_0,i\in\mathcal{M},j\in\mathcal{N} \tag{3}$$

其中，h_0 表示参考距离 $d_0=1$ m 时无线信道内的信道增益大小，d_{ij} 表示第 i 个 GT 和第 j 台无人机之间的距离，利用欧几里德坐标系可以表示为：

$$d_{ij}=\sqrt{(x_i^{GT}-x_j^{UAV})^2+(y_i^{GT}-y_j^{UAV})^2+H^2},i\in\mathcal{M},j\in\mathcal{N} \tag{4}$$

因此，第 i 个 GT 和第 j 台 UAV 之间的数据传输速率可以进一步表示为：

$$R_{ij}=B\log\left(1+\frac{P_ih_{ij}}{\sigma^2}\right),i\in\mathcal{M},j\in\mathcal{N} \tag{5}$$

其中，B 表示 GTs 与 UAVs 之间的系统带宽，与文献[31]类似，本文假设所有 UAVs 通过频分多址(Frequency Division Multiple Access，FDMA)的方式为所有 GTs 提供同等带宽分配的服务；P_i 表示第 i 个 GT 的传输功率；σ^2 表示信道中的背景噪声功率。

本文定义一个二进制变量 β_{ij} 表示 GTs 与 UAVs 之间的关联关系，其中，$\beta_{ij}=1$ 表示第 i 个 GT 决定将部分任务卸载到第 j 台 UAV 上执行，否则 $\beta_{ij}=0$。为了避免对计算任务过度划分而破坏原任务的完整性，本文规定每个 GT 至多能将部分任务卸载到一台 UAV 上执行，因此有以下约束：

$$\sum_{j=1}^{N}\beta_{ij}\leqslant 1,i\in\mathcal{M} \tag{6}$$

此外，由于 UAVs 的并发能力有限，可能无法同时响应过多的卸载请求，为了避免过多的请求造成的网络拥塞和排队时延，本文规定了在当前时隙内每台 UAV 的并发数量上限，因此有以下约束：

$$\sum_{i=1}^{M}\beta_{ij}\leqslant M_{max},j\in\mathcal{N} \tag{7}$$

综合上述分析，对于第 i 个 GT，其边缘计算模式下的任务完成时延由传输时延和计算时延两部分构成，具体可以表示为：

$$T_i^{UAV}=\sum_{j=1}^{N}\beta_{ij}\left(\frac{\alpha_i^{UAV}D_i}{R_{ij}}+\frac{\alpha_i^{UAV}D_iS_i}{f_j^{UAV}}\right),i\in\mathcal{M} \tag{8}$$

其中，f_j^{UAV} 表示第 j 台 UAV 分配给关联 GT 的 CPU 计算频率。

(3)在云计算模式下，GTs 与 LEO 卫星将采用地面-空间直接传输技术将部分计算任务卸载到远端执行，这种地面与

空间卫星直接传输技术的可行性已在现有的工作中得到论证。与文献[30]类似，本文将所有 GTs 与 LEO 卫星之间的数据传输速率统一定义为 R_s，并且，由于 GTs 与 LEO 卫星之间的距离较远，该数值通常小于 GTs 与 UAVs 之间的传输速率。此外，定义参数 f_j^{LEO} 表示 LEO 卫星分配给各 GT 的 CPU 计算频率，因此，其云计算模式下的任务完成时延可以表示为：

$$T_i^{\mathrm{LEO}}=\left(\frac{\alpha_i^{\mathrm{LEO}}D_i}{R_s}+\frac{\alpha_i^{\mathrm{LEO}}D_iS_i}{f^{\mathrm{LEO}}}\right),i\in\mathcal{M} \tag{9}$$

综合上述分析，由于本地计算模式、边缘计算模式和云计算模式可以并行地执行，因此，对于第 i 个 GT，其计算任务的完成时延 T_i 表示为：

$$T_i=\max(T_i^{\mathrm{GT}},T_i^{\mathrm{UAV}},T_i^{\mathrm{LEO}}),i\in\mathcal{M} \tag{10}$$

系统平均响应时延定义为系统中所有计算任务完成时延的平均值，具体表示如下：

$$T(u^{\mathrm{UAV}},\alpha,\beta)=\frac{1}{M}\sum_{i=1}^{M}T_i \tag{11}$$

考虑到该系统中同时包含多个 GTs、多台 UAVs 和一个 LEO 卫星，需要联合优化 UAVs 的部署$\{u^{\mathrm{UAV}}\}$和任务的调度，包括 GTs 与 UAVs 之间的关联关系$\{\alpha\}$和任务的卸载率$\{\beta\}$，以实现系统平均响应时延最小化。联合部署和计算卸载的优化问题可以表述为：

$$\begin{aligned}
&\mathrm{P1}:\min_{u^{\mathrm{UAV}},\alpha,\beta}\ T(u^{\mathrm{UAV}},\alpha,\beta)\\
&\text{s.t. }\ \mathrm{C1}:T_i\leqslant E_i,i\in\mathcal{M}\\
&\qquad\mathrm{C2}:\alpha\in\{0,1\}\\
&\qquad\mathrm{C3}:\alpha_i^{\mathrm{GT}}+\alpha_i^{\mathrm{UAV}}+\alpha_i^{\mathrm{LEO}}=1,i\in\mathcal{M}\\
&\qquad\mathrm{C4}:\beta\in[0,1]\\
&\qquad\mathrm{C5}:\sum_{j=1}^{N}\beta_{ij}\leqslant 1,i\in\mathcal{M}\\
&\qquad\mathrm{C6}:\sum_{i=1}^{M}\beta_{ij}\leqslant M_{\max},j\in\mathcal{N}\\
&\qquad\mathrm{C7}:\beta_{ij}d_{ij}\leqslant R_{\max},i\in\mathcal{M},j\in\mathcal{N}
\end{aligned} \tag{12}$$

其中，C1 表示各任务的完成时延不能超过该任务的最大容忍时延；C2 和 C3 表示任务卸载率的约束；C4－C6 表示 GTs 与 UAVs 之间的关联关系约束；C7 表示每台 UAV 仅能为有限范围内 GTs 提供卸载服务。

4 优化问题求解

4.1 问题分析

由于形式化定义的联合优化问题是一个混合整数非线性规划问题，该目标函数为非凸函数，并且约束条件 C4－C7 是非凸约束，这使得该问题难以直接求解。粒子群优化算法(Particle Swarm Optimization，PSO)可以不利用目标函数的梯度信息，其对目标函数的连续性和可导性没有要求，具有求解上述优化问题的潜力。但是，传统的 PSO 算法在解决的过程中存在以下问题：

(1)该优化问题同时涉及连续变量$(u^{\mathrm{UAV}},\alpha)$和离散变量$\beta$，是典型的混合决策变量优化问题，传统的 PSO 算法难以直接用于求解该类问题。

(2)UAVs 的部署和任务的调度之间紧密耦合。一方面，GTs 与 UAVs 之间的关联取决于 UAVs 的部署，这是因为 UAVs 仅能为各自覆盖范围内的 GTs 提供卸载服务；另一方面，为了得到最佳的系统性能，UAVs 的部署方案需要根据相应的计算卸载方案进行调整。

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

基于上述原因，直接采用传统 PSO 算法解决该问题是低效的。为了应对上述问题，本文通过整合 PSO-GA 算法和贪心(Greedy)算法，提出了一种 PG-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，而内层则是通过 Greedy 算法实现对计算卸载的优化。

4.2 基于 PSO-GA 的外层优化算法

PSO-GA 算法是在传统 PSO 算法的基础上引入了 GA 的交叉和变异算子对原算法的粒子更新策略进行改进。该算法继承了 PSO 算法易实现且收敛迅速等优势，同时结合了 GA 具有良好全局搜索能力的特点，能够有效弥补传统 PSO 算法容易陷入局部最优解的缺陷，获得更加优质的解。

由于 PSO 算法是一种基于种群的群智能搜索算法，粒子的编码方式将直接影响到算法的效率和求解质量，然而，传统的编码方式(如二进制编码、整数编码等)难以对问题 P1 的可行解进行表示。由于在所提出的系统模型中，各 UAV 的飞行高度已知且保持一致，在所考虑时隙内各 UAV 的部署位置可由水平方向的坐标决定，因此本文采用以下的编码机制：种群由若干个粒子构成，每个粒子都表示一个 UAVs 集群的部署方案；每个粒子编码为一个包含 N 个二维向量的集合，集合中的每一个二维向量表示一台 UAV 在当前时隙的水平坐标。具体地，在第 t 轮迭代中，种群中的第 k 个粒子可以表示为：

$$U_k^t=(u_{k1}^t,u_{k2}^t,\cdots,u_{kn}^t) \tag{13}$$

其中，$u_{kj}^t(k=1,2,\cdots,K)$表示 t 轮迭代中第 j 台 UAV 在水平方向上的坐标，具体表示为：

$$u_{k1}^t=(x_{k1}^t,y_{k1}^t),j\in\mathcal{N} \tag{14}$$

其中，x_{kj}^t 和 y_{kj}^t 分别表示第 k 个粒子中第 j 台 UAV 部署位置的横、纵坐标。

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 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(123，121)，这表示 1 号 UAV 的部署位置为(123，121，H)。

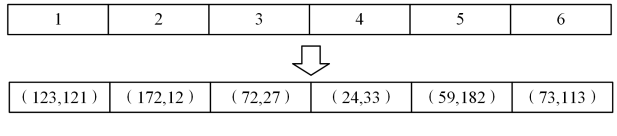

图 2 粒子编码的例子

Fig. 2 Example of particle coding

适应度函数用于评估解的优劣性，本文采用式(11)中定义的系统平均响应时延函数作为 PSO-GA 算法的适应度函数。因此，将 UAVs 部署方案和通过 Greedy 算法得到的计算卸载方案带入式(11)，可得到一个粒子的适应度，Greedy

算法将在下一小节详细介绍。由于本文的优化目标是实现系统的平均任务响应时延最小化，因此适应度函数值越小的粒子展现出的性能更好。

在传统的 PSO 算法中，每个粒子具有两个属性：速度和位置。每个粒子在搜索空间中单独地搜寻最优解，并根据自己找到的当前个体极值和整个粒子群共享的当前全局最优解来调整自己的速度和位置。在这个过程中，粒子的速度和位置的更新方式如下：

$$V_k^{t+1}=wV_k^t+c_1r_1(pBest_k-U_k^t)+c_2r_2(gBest-U_k^t) \quad (15)$$

$$U_k^{t+1}=V_k^{t+1}+U_k^t \quad (16)$$

其中，V_k^t 和 U_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 $gBest$ 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历史最优位置；w 是惯性因子，它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 分别为个体学习因子和社会学习因子，它们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，用于为迭代搜索过程增添随机性。

PSO-GA 算法在 PSO 算法的基础上，通过引入 GA 算法中的交叉和变异算子来改进传统 PSO 算法的粒子更新过程，此时，第 k 个粒子的更新方式变更为：

$$U_k^{t+1}=c_2\oplus C_g(c_1\oplus C_p(w\oplus M_u(U_k^t),pBest_k),gBest) \quad (17)$$

其中，$M_u()$表示 PSO-GA 算法的变异算子，$C_p()$和 $C_g()$表示交叉算子。

在上述的 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：

$$A_k^{t+1}=w\oplus M_u(U_k^t)=\begin{cases}M_u(U_k^t), & r_1<w\\ U_k^t, & \text{其他}\end{cases} \quad (18)$$

其中，r_1 是[0,1]区间内的随机值；$M_u(U_k^t)$表示随机选取粒子 U_k^t 上的某一分位，并在阈值范围内进行随机变异。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 mp_1，并将 mp_1 位上的编码从(72,27)随机变更为(11,162)。

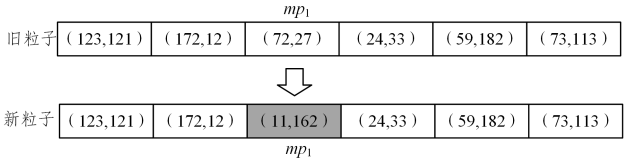

图 3 变异操作

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：

$$B_k^{t+1}=w\oplus C_p(A_k^{t+1},pBest_k)=\begin{cases}C_p(A_k^{t+1},pBest_k), & r_2<c_1\\ A_k^{t+1}, & \text{其他}\end{cases} \quad (19)$$

$$C_k^{t+1}=w\oplus C_g(B_k^{t+1},gBest)=\begin{cases}C_g(B_k^{t+1},gBest), & r_3<c_2\\ B_k^{t+1} & \text{,其他}\end{cases} \quad (20)$$

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 cp_1 和 cp_2，并用 $pBest_k$(或 $gBest$)对应分位之间的 3 个二维向量，即(11,162),(43,26)和(169,24)，进行了替换。

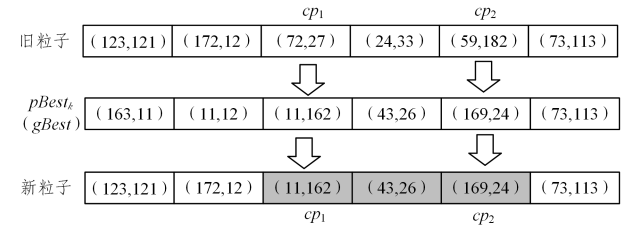

图 4 交叉操作

Fig. 4 Crossover operation

在传统 PSO 算法中，式(17)中的惯性权重因子 w 能够决定算法的收敛速度和搜索能力。当 w 的取值较大时，PSO-GA 算法中粒子发生变异的概率较大，因此算法具有较强的全局搜索能力；当 w 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。由于在算法执行早期更注重问题空间搜索的多样性，并随着搜索的深入，后期应更加注重局部搜索的能力，因此，不同于传统 PSO 算法中使用固定的惯性权重因子，本文采用线性的惯性权重因子调整策略，该策略使得惯性权重因子 w 会随着迭代次数的增加而线性递减，具体的调整策略如下：

$$w=w_{\max}-iters_{\text{cur}}\times\frac{w_{\max}-w_{\min}}{iters_{\max}} \quad (21)$$

其中，$w_{\max}$ 和 $w_{\min}$ 分别是 w 设定的最大值和最小值，$iters_{\max}$ 表示约定的最大迭代次数，$iters_{\text{cur}}$ 表示当前迭代次数。

此外，对于式(17)中的个体学习因子 c_1 和社会学习因子 c_2，本文采用了类似的线性调整策略：

$$c_1=c_{1\_\text{start}}-iters_{\text{cur}}\times\frac{c_{1\_\text{start}}-c_{1\_\text{end}}}{iters_{\max}} \quad (22)$$

$$c_2=c_{2\_\text{start}}+iters_{\text{cur}}\times\frac{c_{2\_\text{end}}-c_{2\_\text{start}}}{iters_{\max}} \quad (23)$$

其中，c_{1_start} 和 c_{2_start} 分别表示参数 c_1 和 c_2 在迭代开始前的初始值，c_{1_end} 和 c_{2_end} 分别表示参数 c_1 和 c_2 在迭代中的最终值。

基于 PSO-GA 的 UAVs 部署优化算法流程如下，伪代码如算法 1 所示。

Step 1 随机初始化种群，包括种群规模大小 pN、最大迭代次数 $iter_{\max}$ 以及粒子 U_i，并初始化每个种群的局部最优解 $pBest_i$ 和全局最优解 $gBest$。

Step 2 在满足算法执行条件下，通过粒子的交叉变异操作更新粒子种群，调用算法 2 获得计算卸载方案，根据式(11)计算每个粒子的适应度，并根据适应度更新局部最优解 $pBest_i$ 和全局最优解 $gBest$。

Step 3 输出最终的全局最优解 $gBest$。

算法 1 PSO-GA 算法

输入：GTs 坐标 u^{GT}、计算任务大小 D_i、计算任务复杂度 S_i、截止时间 E_i

输出：满足条件下的一个全局最优解 gBest

1. 初始化 PSO-GA 参数
2. for each 初始化的粒子 U_i

3. 调用算法 2 获得初始种群粒子 U_i的计算卸载方案
4. 根据式(11)计算粒子 U_i适应度
5. 初始化粒子 U_i的局部最优解 $pBest_i$
6. end for
7. 初始化种群的全局最优解 gBest
8. while not stop
9. for i to pN
10. 对粒子 U_i进行交叉和变异操作
11. 调用算法 2 获得更新后粒子的调度方案
12. 计算粒子的适应度函数值
13. if Fitness(U_i)< Fitness($pBest_i$)
14. 更新 $pBest_i$
15. if Fitness(U_i)< Fitness(gBest)
16. 更新 gBest
17. end for
18. end while
19. 输出 gBest

4.3 基于贪心策略的内层卸载优化算法

本文提出的 PG-G 算法下层需要实现在给定 UAVs 部署方案条件下的计算卸载优化，包括用户关联{α}和卸载率{β}优化两方面，具体如下：

$$\mathrm{P2}:\min_{\alpha,\beta}\frac{1}{M}\sum_{i=1}^{M}T_i \tag{24}$$

s. t. C1－C7

由于云环境下的计算卸载问题通常是 NP Hard 问题，因此用传统算法解决该问题时求解的效率较低。Greedy 算法是一种低复杂度的近似算法，其核心思想是在求解问题的过程中总是选择当前看起来最优的选择，因此能快速获得原问题的近似最优解。在本文提出的模型中，根据式(5)所构建的模型可以得到：当 GT 与所关联的 UAV 距离越近，两者之间的信道增益越大，相应的传输速率就越大。因此，把各个 GT 关联到距离最近的 UAV 上能大大减少计算任务的传输时延。

然而，由于各 UAV 的服务范围是有限的，处于 UAVs 覆盖范围外的 GTs 无法将任务卸载到 UAVs 上；同时，由于 UAVs 的并行处理能力是有限的，无法同时响应过多请求，因此，本文根据 GTs 的分布情况将所有 GTs 分成 3 种类型：

(1)该 GT 不在任何一台 UAV 的覆盖范围内；

(2)该 GT 仅在一台 UAV 的覆盖范围内；

(3)该 GT 在多台 UAVs 的覆盖范围内。

随后，我们按照降序给出上述 3 种类别的优先级。第一类 GTs 拥有最高的优先级，因为这类 GTs 无法关联到 UAVs 上，不占用 UAVs 的计算资源，因此不会影响另外两类 GT 的卸载策略；第二类 GTs 拥有第二级的优先级，因为此类 GTs 仅能关联到最近的 UAV 上；第三类 GTs 可以在多台 UAVs 中选择合适的 UAV 进行关联，因此具有最低的优先级。

然后，我们决策 GTs 的任务卸载率。对于上述第一类 GTs，由于无法关联到 UAVs 上，即 $\alpha_i^{\mathrm{UAV}}=0$，此时原计算任务仅被划分为本地执行部分和 LEO 卫星执行部分；对于第二类和第三类 GTs，原任务被划分为本地执行部分、UAV 执行部分以及 LEO 卫星执行部分。由于 UAVs 的部署位置以及 GTs 和 UAVs 之间的关联关系已经确定，因此任务卸载率优化问题是一个标准的线性规划问题，可以直接采用现有的工具包(如 CVXPY)进行求解。

基于上述分析，本文提出如下的贪心策略：

(1)对于第一类 GTs，不关联 UAVs。

(2)将第二类 GTs 关联到最近的 UAV 上，若当前已有 $M_{\max}$个 GTs 关联到该 UAV，则选择已关联到当前 UAV 上所有 GTs 中时延最大的 GT，使其放弃关联到 UAV 上，并调整卸载率。

(3)对于第三类 GTs，总是尝试将其关联到最近的 UAV 上，若当前已有 $M_{\max}$个 GTs 关联到该 UAV，则选择已关联到当前 UAV 上所有第三类 GTs 中时延最大的 GT，将其卸载到其他最近的 UAV 上；若无法卸载到其他 UAVs 上，则使其放弃关联到 UAV 上，并调整卸载率。

基于 Greedy 的计算卸载优化算法流程如下，伪代码如算法 2 所示。

Step 1 根据 GTs 与 UAVs 之间的位置将 GTs 分成 3 个集合。

Step 2 对第一个集合里的 GTs，直接计算卸载率；对于第二个和第三个集合里的 GTs，先确定其和 UAVs 的关联关系，再计算卸载率。

Step 3 输出最终关联性集合 α 和卸载率集合 β。

算法 2 Greedy 算法

输入：GTs 坐标 u^{GT}，UAVs 的坐标 u^{UAV}，计算任务大小 D_i，计算任务复杂度 S_i，截止时间 E_i

输出：关联性集合 α，卸载率集合 β

1. for i to N
2. 计算 GT_i到各 UAV 之间的距离
3. if GT_i到最近 UAV 的距离大于 UAVs 的服务半径
4. GT_i加入集合 1
5. else if 只有最近的 UAV 到 GT_i的距离小于服务半径
6. GT_i加入集合 2
7. else
8. GT_i加入集合 3
9. end for
10. for i to size(集合 1)
11. GT_i和 UAVs 的关联变量 $\alpha_{ij}=0$
12. 计算 GT_i卸载率
13. end for
14. for i to size(集合 2)
15. GT_i与最近的 UAV_j的关联变量 $\alpha_{ij}=1$
16. 计算 GT_i卸载率
17. if UAV_j的关联数量＞N_{max}
18. 计算所有关联到 UAV_j的 GTs 的完成时延
19. 选择时延最大的 GT_k，改变关联变量 $\alpha_{kj}=0$
20. 重新计算 GT_k卸载率
21. end for
22. for i to size(集合 3)
23. GT_i与最近的 UAV_j的关联变量 $\alpha_{ij}=1$
24. 计算 GT_i卸载率
25. if UAV_j的关联数量＞N_{max}
26. 计算所有关联到 UAV_j的集合 3 中的 GTs 的完成时延

27. 选择时延最大的 GT_k，计算其下一个最近的 UAV_l
28. 改变关联变量 $\alpha_{kj}=0$，$\alpha_{kl}=1$，计算 GT_k 卸载率
29. end for
30. 输出关联性集合 α 和卸载率集合 β

4.4 复杂度分析

本文提出的 PG-G 联合优化算法，其计算复杂度由外层的 PSO-GA 部署优化算法和内层的 Greedy 卸载优化算法共同决定。本小节将分别对上述两个方面的计算复杂度进行分析。

PSO-GA 算法是一种基于粒子群的迭代算法，粒子群的规模 pN 和算法的迭代次数是影响该算法计算复杂度的关键因素；此外，待决策变量的规模（即 UAVs 的数量 N）是影响算法复杂度的另一个要素。为便于分析，用 It 表示算法的实际迭代次数。PSO-GA 算法的主要操作包括种群初始化、获取适应度、粒子交叉和变异。种群初始化的复杂度由粒子群规模 pN 和 UAVs 的数量 N 决定，需要注意的是，由于每台 UAV 的位置由两个未知变量决定，因此其计算复杂度为 $O(pN*N*2)$。适应度计算需要调用内层的 Greedy 算法，其计算复杂度由 Greedy 算法和粒子群规模决定。粒子的交叉需要对每个粒子进行遍历，计算复杂度由粒子群规模 pN 以及实际迭代次数决定，为 $O(pN*It)$。粒子的变异与交叉类似。

Greedy 算法中需要计算各 GTs 和 UAVs 之间的距离，因此 Greedy 算法的计算复杂度主要由 GTs 和 UAVs 的数量决定。首先，Greedy 算法需要按距离将 GTs 分成 3 个集合，其计算复杂度为 $O(M*N)$；其次，Greedy 算法确定集合 1 和集合 2 中的 GTs 的连接决策和卸载率，这部分的计算复杂度由集合中的 GTs 以及求解线性规划的计算复杂度决定，本文采用 CVXPY 工具包求解线性规划问题，因底层调用了内点法的求解过程，故计算复杂度表示为 $O(\log_2(1/\varepsilon)*M^3)$，其中 ε 表示内点法收敛的阈值。在确定集合 3 中 GTs 的连接关系和卸载率时，可能会需要遍历 UAVs 的连接列表，找出需要调整的 GT，并对其重新决策，该计算复杂度可表示为 $O(M_{\max}*N*\log_2(1/\varepsilon)*M^3)$。

综合上述分析，可以得出，Greedy 算法的复杂度约为 $O(MN+M^3(1+M_{\max}N)\log_2(1/\varepsilon))$，而所提出的 PG-G 算法的复杂度为 $O(2N*pN+pN(MN+M^3(1+M_{\max}N)\log_2(1/\varepsilon))+3pN*It)$。

5 数值仿真实验

为了验证所提出的 PG-G 算法的可行性和有效性，本节中，针对以下研究问题（Research Question，RQ）展开仿真实验。

RQ1：PG-G 在求解系统平均响应时间时是否收敛？

RQ2：PG-G 上层的 PSO-GA 算法是否有效？

RQ3：PG-G 下层的 Greedy 算法是否有效？

5.1 实验设置

在本实验中，我们考虑一个 $200\times200\ m^2$ 的矩形区域，分布在该区域的 GTs 的 CPU 计算频率为 0.8～1.2 GHz[32]，GTs 的计算任务输入数据量服从 10 M～15 Mbist 范围内的随机分布[32]，计算任务的复杂度设定为 80～120 cycles/bit[32]，并且任务的最大容忍时延设定为 1 s[30,32]。为了向 GTs 提供网络服务，在该区域内部署了 $N=4$ 台 UAVs 作为边缘服务器，UAVs 的 CPU 计算频率为 2 GHz；此外，一台 LEO 卫星被用作云服务器，LEO 卫星为每个 GTs 分配的计算频率为 3 GHz，卫星与 GTs 之间的数据传输速率设定为 20 Mbits/s。其他系统参数如表 2 所列[30]。

表 2 系统参数设定

Table 2 System parameters

系统参数	值
H/m	40
B/MHz	10
P/W	1
σ^2/dBm	−130
h_0/dB	−30
M/台	20～40
$M_{\max}$/台	8
$R_{\max}$/m	{40,80}

此外，参考我们之前的工作[33]和相关工作[34]，本文提出的 PG-G 联合优化算法的参数设置如表 3 所列。

表 3 算法参数设定

Table 3 Algorithm parameter

算法参数	值
$w_{\max}$	0.8
$w_{\min}$	0.2
c_{1_start}	0.9
c_{1_end}	0.2
c_{2_start}	0.4
c_{2_end}	0.9
$iter_{\max}$	1000

5.2 PG-G 的收敛性

本节中给出了所提出的 PG-G 算法在不同场景下的收敛性曲线。在图 5 中，我们可以看到，所提出的 PG-G 算法总能在 500 次迭代之内收敛到一个稳定值。此外，我们还可以看到，当 UAVs 服务范围为 40 m 时，收敛的速度相对较慢，而当 UAVs 的服务范围为 80 m 时算法的收敛速度较快。这是由于当 UAVs 的服务范围为 40 m 时，有较多的 GTs 无法被 UAVs 所覆盖。

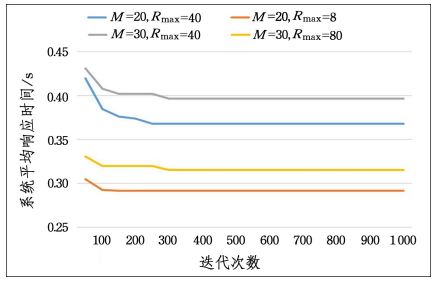

图 5 PG-G 迭代曲线

Fig. 5 Iterative curve of PG-G

为了向更多的 GTs 提供服务，需要对 UAVs 部署有更严格的要求，因此在使用 PG-G 搜索时需要花费更多的时间。而当 UAVs 的服务范围为 80 m 时，几乎所有的 GTs 都能被

UAVs 所覆盖，因此 PG-G 在搜索时所需要的时间相对较短。另外，我们可以看到，当用户数量为 20 时，PG-G 的收敛速度相对较快，而当用户数量增加时，该算法的收敛速度随之降低。

5.3 PG-G 上层 PSO-GA 算法的有效性

在本小节中，为了验证所提出 PG-G 算法上层 PSO-GA 算法的有效性，需要引入其他算法进行对比。然而，由于所形式化的问题是一个混合非线性的优化问题，无法依靠常规手段获得该问题的理想方案和最优解；此外，由于现有的对于空-天-地一体化 MEC 系统的研究较少，并且本文的研究与现有工作存在许多不同之处，因此无法直接采用现有工作中的方法作为本文的对比算法。因此，本文改写了相关研究中常用的两种基准算法，引入了对比算法 RAN-G 和 DE-G。

(1)RAN-G：RAN-G 算法是采用随机算法来替代 PG-G 上层的 PSO-GA 算法作为 UAVs 的部署算法，下层则采用本文所提出的 Greedy 算法作为计算卸载算法。由于随机算法具有很强的不确定性，在 RAN-G 算法中，采用重复 1 000 次实验求平均的方式作为最终结果。

(2)DE-G：DE-G 算法是采用差分进化算法作为搜索引擎，代替本文的 PSO-GA 算法作为 UAVs 的部署算法，下层则采用本文提出的 Greedy 算法作为计算卸载算法。

在本小节中对各个不同场景进行了重复实验，通过对比各算法的平均值和最优值来证明所提算法的有效性。图 6 和图 7 中分别给出了当 UAVs 的服务半径为 40m 和 80m 时各算法的系统平均响应时间。可以看到，在各种场景下，本文提出的 PG-G 算法在平均值和最优值方面均明显优于 RAN-G，并且在服务半径为 80m 时效果显著。这是因为当 UAVs 的服务半径较大时，有效的部署方案能够覆盖更多的 GTs，使更多的 GTs 将部分任务卸载到 UAVs 上执行，从而充分利用了 UAVs 的计算资源。而不好的部署方案，即使 UAVs 的服务范围足够大，也会使部分 GTs 无法关联到 UAVs，从而降低了资源利用率。另外，PG-G 算法性能在不同条件下和 DE-G 对比有好有坏，从结果上看，GTs 数量较少时 PG-G 在最优值和平均值上表现较好，随着 GTs 数量的增加，DE-G 展现出了更好的优化性能。原因是 DE 算法作为一种典型的进化算法拥有更强的全局搜索能力，相比 PSO-GA 更容易跳出局部最优，因此在问题规模更大时更具有优势；PSO-GA 作为一种基于粒子群的群智能搜索算法拥有更快的收敛速度，但是容易收敛到局部最优，因此适用于小规模问题。

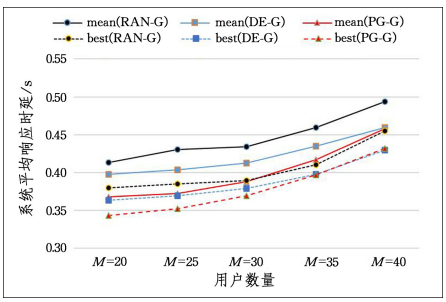

图 6 $R_{max}=40$ m 时不同算法的系统平均响应时延

Fig. 6 System average response delay of different algorithms when $R_{max}=40$ m

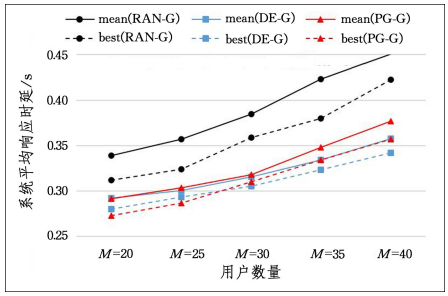

图 7 $R_{max}=80$ m 时不同算法的系统平均响应时延

Fig. 7 System average response delay of different algorithms when $R_{max}=80$ m

5.4 PG-G 下层算法的有效性

在本节中，为了验证 PG-G 算法下层的 Greedy 算法的有效性，引入了对比算法 PG-RAN 和 PG-AVG。

(1)PG-RAN：PG-RAN 采用 PSO-GA 算法作为 UAVs 的部署算法，在算法的下层采用随机的方式关联到可连接的 UAV，然后将计算任务随机划分并卸载到各端执行。

(2)PG-AVG：PG-AVG 采用本文提出的 PSO-GA 算法作为 UAVs 的部署算法，在算法的下层就近关联到可连接的 UAV，然后根据连接情况将计算任务平均分配给各端执行。

本节对所提出的 PG-G 算法和上述对比算法在不同的场景进行了重复实验，对比各算法的平均值和最优值证明了所提算法的有效性。图 8 和图 9 分别给出了当 UAVs 的服务半径为 40m 和 80m 时各算法系统的平均响应时间。

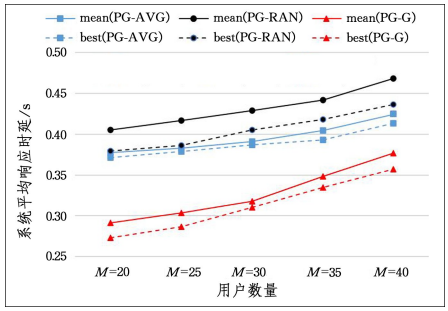

图 8 $R_{max}=40$ m 时不同算法的系统平均响应时延

Fig. 8 System average response delay of different algorithms when $R_{max}=40$ m

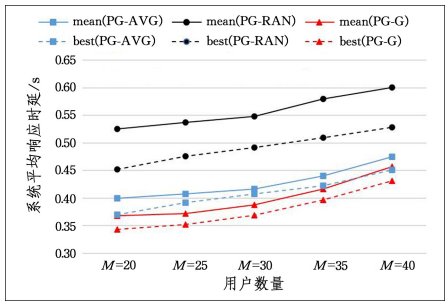

图 9 $R_{max}=80$ m 时不同算法的系统平均响应时延

Fig. 9 System average response delay of different algorithms when $R_{max}=80$ m

从图 8 和图 9 可以看到，所提出的 PG-G 算法在各种场景下均能获得最好的平均值和最优值，PG-AVG 其次，

PG-RAN 最差。这是因为相比基于随机的计算卸载策略，本文提出的 Greedy 算法在用户关联和卸载率决策两个方面都进行了优化，因此性能提升明显；而相对于 PG-AVG，本文提出的 Greedy 算法在卸载率优化方面明显优于平均分配的方案，因此，在 UAVs 服务范围较大时，本文提出的 Greedy 算法能取得更好的优化效果。

结束语 本文研究了一个空-天-地一体化的 MEC 系统，部署多台 UAVs 作为边缘服务器，LEO 卫星被用作云服务器，并通过联合部署优化和计算卸载实现系统平均响应时延最小化。为求解该联合优化问题，本文提出了一种 PG-G 联合优化算法，该算法上层采用 PSO-GA 实现 UAVs 的部署优化，下层采用 Greedy 算法实现计算卸载优化。数值仿真结果验证了所提算法的有效性，并且和其他基准算法相比，PG-G 算法的优化效果非常明显。

目前我们正在研究空-天-地一体化 MEC 系统中的资源分配问题，包括系统信道带宽分配和传输功率控制等，这将进一步完善本文提出的系统模型。此外，UAVs 的覆盖范围是影响系统性能的关键因素，而 UAVs 的覆盖范围通常受到地理环境以及飞行高度的影响，因此，UAVs 的三维坐标也是我们后续研究的方向。

参 考 文 献

[1] XIAO H, HU Z, YANG K, et al. 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] FENG J, YU F R, PEI Q, et al. Cooperative Computation Offloading and Resource Allocation for Blockchain-Enabled Mobile-Edge Computing: A Deep Reinforcement Learning Approach [J]. IEEE Internet of Things Journal, 2020, 7(7): 6214-6228.

[7] LIU L, CHEN C, PEI Q, et al. Vehicular Edge Computing and Networking: A Survey[J]. Mobile Networks and Applications, 2021, 26: 1145-1168.

[8] LI L, WEN X, LU Z, JING W. An Energy Efficient Design of Computation Offloading Enabled by UAV[J]. Sensors, 2020, 20(12): 3363.

[9] LI B, FEI Z, ZHANG Y. UAV Communications for 5G and Beyond: Recent Advances and Future Trends[J]. IEEE Internet of Things Journal, 2019, 6(2): 2241-2263.

[10] LIU J, SHI Y, FADLULLAH Z M, et al. Space-Air-Ground Integrated Network: A Survey[J]. IEEE Communications Surveys & Tutorials, 2018, 20(4): 2714-2741.

[11] CHENG N, LYU F, QUAN W, et al. Space Aerial-Assisted Computing Offloading for IoT Applications: A Learning-Based Approach[J]. IEEE Journal on Selected Areas in Communications, 2019, 37(5): 1117-1129.

[12] MEI H, YANG K, LIU Q, et al. Joint Trajectory-Resource Optimization in UAV-Enabled Edge-Cloud System With Virtualized Mobile Clone[J]. IEEE Internet of Things Journal, 2020, 7(7): 5906-5921.

[13] MEHRABI M, YOU D, LATZKO V, et al. Device-Enhanced MEC: Multi-Access Edge Computing (MEC) Aided by End Device Computation and Caching: A Survey[J]. IEEE Access, 2019, 7(99): 166079-166108

[14] PAVEL M, ZDENEK B. Mobile edge computing: A survey on architecture and computation offloading[J]. IEEE Communications Surveys Tutorials, 2017, 19(3): 1628-1656.

[15] 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.

[16] PEN Y J, CHEN M, YANG Z H, et al. Energy-Efficient NOMA-Based Mobile Edge Computing Offloading[J]. IEEE Communications Letters, 2019, 23(2): 310-313.

[17] YANG G, HOU L, HE X, et al. Offloading Time Optimization via Markov Decision Process in Mobile Edge Computing[J]. IEEE Internet of Things Journal, 2021, 8(4): 2483-2493.

[18] ZHANG J, HU X, NING Z, et al. Energy-Latency Tradeoff for Energy-Aware Offloading in Mobile Edge Computing Networks [J]. IEEE Internet of Things Journal, 2017, 5(4): 2633-2645.

[19] 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.

[20] HUANG L, BI S, ZHANG Y. Deep Reinforcement Learning for Online Computation Offloading in Wireless Powered Mobile-Edge Computing Networks[J]. IEEE Transactions on Mobile Computing, 2020, 19(11): 2581-2593.

[21] CHOUHAN S. Energy Optimal Partial Computation Offloading Framework for Mobile Devices in Multi-access Edge Computing [C]// 2019 International Conference on Software, Telecommunications and Computer Networks(SoftCOM). 2019.

[22] 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.

[23] PANG Y, WANG D, WANG D, et al. A Space-Air-Ground Integrated Network Assisted Maritime Communication Network Based on Mobile Edge Computing[C]// 2020 IEEE World Congress on Services(SERVICES). IEEE, 2020: 269-274.

[24] NIU Z, SHEN X S, ZHANG Q, et al. Space-air-ground integra-

ted vehicular network for connected and automated vehicles: Challenges and solutions[J]. Intelligent and Converged Networks,2020,1(2):142-168.

[25] CHEN Q,MENG W,HAN S,et al. Service-Oriented Fair Resource Allocation and Auction for Civil Aircrafts Augmented Space-Air-Ground Integrated Networks[J]. IEEE Transactions on Vehicular Technology,2020,69(11):13658-13672.

[26] WANG Z,YU H,ZHU S,et al. Curriculum Reinforcement Learning-Based Computation Offloading Approach in Space-Air-Ground Integrated Network[C]// 2021 13th International Conference on Wireless Communications and Signal Processing (WCSP). IEEE,2021:1-6.

[27] ZHOU C,WU W,HE H,et al. Delay-aware IoT task scheduling in space-air-ground integrated network[C]//2019 IEEE Global Communications Conference(GLOBECOM). IEEE,2019:1-6.

[28] ZHOU C,WU W,HE H,et al. Deep reinforcement learning for delay-oriented IoT task scheduling in SAGIN[J]. IEEE Transactions on Wireless Communications,2020,20(2):911-925.

[29] TANG F,HANS H,NEI K,et al. A Deep Reinforcement Learning-Based Dynamic Traffic Offloading in Space-Air-Ground Integrated Networks (SAGIN)[J]. IEEE Journal on Selected Areas in Communications,2022,40(1):276-289.

[30] MAO S,HE S,WU J. Joint UAV Position Optimization and Resource Scheduling in Space-Air-Ground Integrated Networks With Mixed Cloud-Edge Computing[J]. IEEE Systems Journal,2021,15(3):3992-4002.

[31] 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.

[32] CHEN Z,ZHENG H,ZHANG J,et al. Joint computation offloading and deployment optimization in multi-UAV-enabled MEC systems[J]. Peer-to-Peer Networking and Applications,2022,15(1):194-205.

[33] CHEN X,ZHANG J,LIN B,et al. Energy-efficient offloading for DNN-based smart IoT systems in cloud-edge environments [J]. IEEE Transactions on Parallel and Distributed Systems,2021,33(3):683-697.

[34] SHI Y,EBERHART R. A modified particle swarm optimizer [C]// 1998 IEEE International Conference on Evolutionary Computation Proceedings. IEEE,1998:69-73.

ZHENG Hongqiang, born in 1994, postgraduate. His main research interests include edge computing and resource allocation.

ZHANG Jianshan, born in 1995, Ph.D, professor. His main research interests include cloud/edge computing and intelligent computing.

(责任编辑:何杨)