异构边缘云架构下的多任务卸载算法

尼俊红[1,2], 臧云[1]
(1. 华北电力大学(保定) 电子与通信工程系,河北 保定 071003; 2. 华北电力大学 河北省电力物联网技术重点实验室,河北 保定 071003)

摘 要:为在资源有限的终端设备上运行计算密集型与时延敏感型应用,同时降低系统时延和能耗,构建边缘云异构网络模型。本文提出了一种 H-PSOGA 多任务卸载优化算法,并通过无人机、路边单元、车辆等边缘设备以及边缘云服务器进行多任务计算卸载。该算法以先串行再并行的方式将粒子群和遗传算法结合在一起,通过适应度值排序、种群选择、多点交叉、反向变异等操作,利用遗传算法对粒子群进行优选,弥补粒子群算法早熟收敛、陷入局部最优的缺陷。6 种标准测试函数的测试分析以及与基线方案进行仿真对比的结果表明:在用户数较多时,混合优化算法的系统平均开销可降低 26%~43%,可以有效提高收敛精度。

关键词:移动边缘计算; 异构网络; 边缘节点; 任务卸载; 粒子群算法; 遗传算法; 多目标优化; 标准测试函数
DOI:10. 11990/jheu. 202111007
网络出版地址:https://link. cnki. net/kcms/detail/23. 1390. U. 20240131. 0932. 002
中图分类号:TN929. 5 **文献标志码**:A **文章编号**:1006-7043(2024)04-0800-08

Multitask offloading algorithm under heterogeneous edge cloud architecture

NI Junhong[1,2], ZANG Yun[1]
(1. Department of Electronic and Communication Engineering, North China Electric Power University, Baoding 071003, China; 2. Hebei Key Laboratory of Power Internet of Things Technology, North China Electric Power University, Baoding 071003, China)

Abstract: To run computing-intensive and delay-sensitive applications on terminal devices with limited resources and to reduce system time delay and energy consumption, an edge cloud heterogeneous network model is constructed. In addition, a hierarchical particle swarm optimization with genetic algorithm (H-PSOGA) for multitasking offloading optimization is proposed for multitasking computational offloading through edge devices, such as drones, roadside units, and vehicles, as well as edge cloud servers. The H-PSOGA combines particle swarm and genetic algorithms in a serial and then parallel manner. The genetic algorithm is used to optimize the particle swarm through operations such as fitness value sequencing calculation, population selection, multipoint crossover, and reverse mutation to compensate for the defects in the particle swarm algorithm. These issues include premature convergence and local optima. The test analysis of six standard test functions and the result of simulation comparison with the baseline scheme show that with a large number of users, the average cost can be reduced by 26% to 43%, H-PSOGA can effectively improve convergence accuracy reduce system overhead.

Keywords: mobile edge computing; heterogeneous network; edge node; task offloading; particle swarm algorithm; genetic algorithm; multiobjective optimization; standard test function

随着移动通信技术的发展和智能终端的普及,增强现实(augmented reality,AR)、虚拟现实(virtual reality,VR)、无人驾驶等各种应用不断涌现,用户对服务质量(quality of service,QoS)和体验质量有了更高的要求。预计到 2030 年,移动数据流量将出现爆炸式增长,全球移动终端数将接近 1 000 亿,中国可能达到 200 亿[1]。在终端设备上运行这些新兴的数据量较大的应用程序,需要大量的计算资源、存储资源以及较高的能耗,而移动设备的计算能力、资源存储和电池电量往往是有限的,无法满足这些需求。

欧洲电信标准化协会在 2014 年成立了移动边缘计算(mobile edge computing,MEC)规范工作组,

收稿日期:2021-11-04. **网络出版日期**:2024-01-31.
基金项目:国家自然科学基金项目(61771195);河北省自然科学基金项目(F2018502047).
作者简介:尼俊红,女,副教授,硕士生导师.
通信作者:尼俊红,E-mail: 2267295749@ qq. com.

促进对移动边缘计算的研究[2]。移动边缘计算是指在移动网络边缘部署计算和存储资源,为移动网络提供 IT 服务环境和云计算能力,从而为用户提供超低时延和高带宽的网络服务解决方案[3-4]。MEC 的关键技术为计算卸载,为应用制定适合的卸载决策[5]是近年来的研究热点。

研究人员为了研究任务卸载决策的相关问题做了许多工作,通过建立车辆和无人机协助下的 MEC 模型,将不同的任务按需卸载到不同的边缘节点,并设计了分布式匹配-贪婪算法[6]。Gu 等[7]研究了车辆网络移动边缘计算系统,提出了 2 种独立的启发式匹配算法解决移动中的任务卸载问题。Du 等[8]通过联合优化卸载决策和计算资源、无线电资源的分配,解决混合云/雾系统中的计算卸载问题。Wang 等[9]提出了基于智能体的任务卸载体系架构,协同无人机和边缘服务器上的计算资源为用户提供最优的任务卸载决策。Yu 等[10]建立了深度多标签分类模型,并将模型分为离线演示生成、离线模型训练和在线决策 3 个阶段,提出了基于深度模仿学习的 MEC 任务卸载策略。Ke 等[11]考虑车辆移动场景下信道的时变性,建立了异构车载网络任务计算卸载模型,提出了一种基于深度强化学习的自适应计算卸载方法,通过权衡能耗、带宽分配和缓冲队列的成本来选择任务最佳卸载策略。

除了以上这些方法,还有一些研究者通过群体智能算法解决卸载问题,例如遗传算法(genetic algorithm,GA)、粒子群算法(particle swarm optimization,PSO)等。王妍等[12]提出云辅助移动边缘计算的计算卸载策略,在 GA 算法的基础上,设计了一种基于改进 GA 算法的计算卸载算法。罗斌等[13]针对时延敏感型的应用,提出了一种基于 PSO 算法的计算卸载策略,在能耗约束条件下最小化时延,使用惩罚函数来平衡时延与能耗。Wu 等[14]考虑延迟和能量消耗因素,提出了一种边缘云协同多任务计算卸载模型,采用非线性指数惯性权重 PSO 算法进行求解。Wei 等[15]通过建立异构网络模型以及任务依赖关系模型,设计了结合贪婪策略的 PSO 和动态 PSO 2 种算法研究任务卸载策略。Adhikari 等[16]研究了分布式雾设备和集中式云数据中心的协作,使用加速 PSO 技术为分层雾云环境设计了一种多目标卸载策略,在有效利用计算资源的同时,最小化总成本和延迟。

上述研究工作的共同特点是边缘节点类型单一或者在构建系统模型时未考虑不同计算任务的差异性。而在实际环境中,往往是复杂异构的边缘网络,而用户侧无论是任务的类型还是任务的数量都不是单一的。本文考虑构建混合云/雾的异构边缘网络模型,将多任务卸载到车辆、无人机、路边单元等边缘节点或者通过边缘节点卸载到边缘云,满足用户的服务质量和体验质量。

1　异构边缘云系统模型

1.1　网络模型

如图 1 所示,本文考虑由 N 个用户,M 个边缘节点,1 个边缘云组成的异构边缘云系统,其中无人机、路边单元、车辆作为边缘节点,数量分别为 I、J、K,$I+J+K=M$。为了满足用户的服务质量和体验质量,用户可以在本地执行任务,也可以将任务通过 D2D 方式卸载到车辆,或通过蜂窝无线网络卸载到无人机、路边单元(road side unit,RSU)等边缘节点。由于边缘节点的计算能力和能源有限,当其无法满足用户需求时,可将任务通过边缘节点中继到边缘云服务器。边缘云部署在服务小区的基站上,无人机和车辆通过无线方式连接到边缘云,路边单元通过有线链路连接到边缘云。

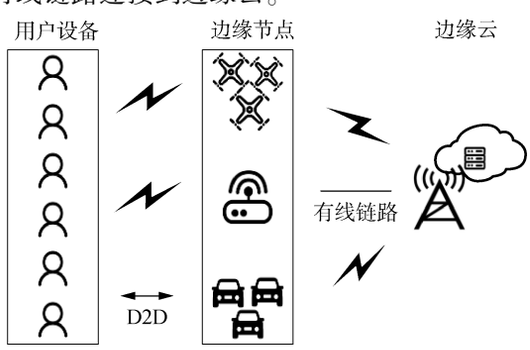


图 1　异构边缘云系统网络模型

Fig. 1　Network model of heterogeneous edge cloud system

用户卸载决策的约束条件为:

$$\begin{cases} X_n + Y_n^{\mathrm{dro}} + Y_n^{\mathrm{rsu}} + Y_n^{\mathrm{veh}} + Z_n^{\mathrm{dro}} + Z_n^{\mathrm{rsu}} + Z_n^{\mathrm{veh}} = 1 \\ X_n, Y_n^{\mathrm{dro}}, Y_n^{\mathrm{rsu}}, Y_n^{\mathrm{veh}}, Z_n^{\mathrm{dro}}, Z_n^{\mathrm{rsu}}, Z_n^{\mathrm{veh}} \in \{0,1\} \\ \forall n \in N, \forall i \in I, \forall j \in J, \forall k \in K \end{cases} \tag{1}$$

式中:$X_n = 1$、$Y_n^{\mathrm{dro}} = 1$、$Y_n^{\mathrm{rsu}} = 1$、$Y_n^{\mathrm{veh}} = 1$、$Z_n^{\mathrm{dro}} = 1$、$Z_n^{\mathrm{rsu}} = 1$、$Z_n^{\mathrm{veh}} = 1$,分别表示计算任务在本地、无人机、路边单元、车辆及分别通过 3 种中继方式卸载至边缘云服务器执行。式(1)变量中有且只有一个变量取值为 1,即任务进行完全卸载,只能在一个服务器执行。

假设无线信道总带宽为 B,每个用户设备(user equipment,UE)有 Q 种类型的计算任务要执行,定义 $J_n^q \triangleq \{B_n^q, C_n^q, D_n^q, \tau^{\max q}, e^{\max q}\}, \forall n \in N, \forall q \in Q$ 表示用户 n 第 q 种类型的计算任务,其中 B_n^q 表示输入数据大小,C_n^q 表示完成计算任务所需的 CPU 周期,D_n^q 表示输出数据大小,$\tau^{\max q}$、$e^{\max q}$ 分别表示 q 类型计算任务的容忍时延和容忍能耗。在不影响分析结果的前提下,为了简化符号的书写,省略上标 q,

将用户 n 的计算任务采用 $J_n \triangleq \{B_n, C_n, D_n, \tau^{\max}, e^{\max}\}, \forall n \in N$ 表示。

1.2 计算模型

为了便于分析,假定边缘节点和边缘云仅在接收到所有输入数据之后,开始执行任务。

1.2.1 本地执行

在本地执行任务的时延 T_n^{loc} 和能耗 E_n^{loc} 分别为:

$$T_n^{\text{loc}} = C_n / f^{\text{loc}} \tag{2}$$

$$E_n^{\text{loc}} = P_{\text{com}}^{\text{loc}} C_n / f^{\text{loc}} \tag{3}$$

式中:C_n 表示用户 n 完成计算任务所需的 CPU 周期;f^{loc} 表示 UE 本地计算能力;$P_{\text{com}}^{\text{loc}}$ 表示 UE 本地执行功耗。

1.2.2 边缘节点执行

本文中的边缘节点包含具有一定计算能力的无人机、路边单元和车辆。用户任务卸载到边缘节点执行的时延 T_n^{dro} 由任务上传时延 $T_{n,e_{i,j}}^{\text{up}}$、边缘节点执行时延 $T_{n,e_{i,j}}^{\text{com}}$ 和任务执行结果回传时延 $T_{n,e_{i,j}}^{\text{down}}$ 构成:

$$T_n^{\text{dro}} = T_{n,e_{i,j}}^{\text{up}} + T_{n,e_{i,j}}^{\text{com}} + T_{n,e_{i,j}}^{\text{down}} = B_n / R_{n,e_{i,j}} + C_n / f^{e_{i,j}} + D_n / R_{e_{i,j},n} \tag{4}$$

式中:$e_{i,j}$ 的第 1 个下标 i 表示边缘节点类型,$i=1,2,3$ 分别对应无人机、路边单元和车辆;j 表示边缘节点的序号;B_n、$R_{n,e_{i,j}}$ 分别表示用户 n 上传数据量大小以及用户 n 和边缘节点 $e_{i,j}$ 之间通信链路的上行传输速率;$f^{e_{i,j}}$ 表示边缘节点 $e_{i,j}$ 的计算能力;$R_{e_{i,j},n}$ 表示边缘节点 $e_{i,j}$ 与用户 n 之间的下行传输速率。假定通信链路的上、下行信道条件相同,且分配带宽相同,则上、下行传输速率相等,即,$R_{n,e_{i,j}} = R_{e_{i,j},n}$。

用户 n 将任务卸载到边缘节点执行的能耗为:

$$E_n^{\text{dro}} = P_{\text{tra}}^{\text{loc}} \frac{B_n}{R_{n,e_{i,j}}} + P_{\text{com}}^{e_{i,j}} \frac{C_n}{f^{e_{i,j}}} + P_{\text{tra}}^{e_{i,j}} \frac{D_n}{R_{e_{i,j},n}} + P^{\text{lid}} \frac{C_n}{f^{e_{i,j}}} \tag{5}$$

式(5)中的 4 项分别对应任务上传能耗、边缘节点 $e_{i,j}$ 执行能耗、执行结果回传能耗以及 UE 空载状态下的能耗;$P_{\text{tra}}^{\text{loc}}$、$P^{\text{lid}}$ 分别表示 UE 的发射功率和空闲状态下的功耗;$P_{\text{com}}^{e_{i,j}}$ 和 $P_{\text{tra}}^{e_{i,j}}$ 分别表示边缘节点 $e_{i,j}$ 的计算功耗和发射功率。

1.2.3 边缘云执行

除了以上几种任务执行方式之外,还可以将任务通过边缘节点中继到部署在基站的边缘云服务器执行。

任务通过边缘节点中继到边缘云卸载时,时延的构成与边缘节点执行类似,不同点在于,上传时延 $T_n^{c_{\text{up}}}$ 增加了从边缘节点到边缘云的上传时间 $(D_n/R_{e_{i,j},c})$,其中 $R_{e_{i,j},c}$ 为边缘节点 $e_{i,j}$ 与边缘云之间的传输速率;回传时延 $T_n^{c_{\text{down}}}$ 中增加了边缘云到边缘节点的时延 $(D_n/R_{c,e_{i,j}})$,其中 $R_{c,e_{i,j}}$ 为边缘节云与边缘节点之间的传输速率。

因此,任务通过边缘节点中继到边缘云服务器卸载的时延为:

$$T_n^{c_{\text{dro}}} = \frac{B_n}{R_{n,e_{i,j}}} + \frac{B_n}{R_{e_{i,j},c}} + \frac{C_n}{f^{\text{clo}}} + \frac{D_n}{R_{c,e_{i,j}}} + \frac{D_n}{R_{e_{i,j},n}} \tag{6}$$

式中:前 2 项为上传时延 $T_n^{c_{\text{up}}}$;第 3 项为边缘云执行时延 $T_n^{c_{\text{com}}}$,其中 f^{clo} 表示边缘云的计算能力;后 2 项为回传时延 $T_n^{c_{\text{down}}}$。

能耗主要包括任务上传阶段能耗 $E_n^{c_{\text{up}}}$、任务执行能耗 $E_n^{c_{\text{com}}}$、任务回传阶段能耗 $E_n^{c_{\text{down}}}$ 以及 UE 在空载状态下的能耗 $E_n^{c_{\text{lid}}}$。$E_n^{c_{\text{up}}}$ 又包括 UE 发射能耗和边缘节点发射能耗:

$$E_n^{c_{\text{up}}} = (P_{\text{tra}}^{\text{loc}} D_n / R_{n,e_{i,j}}) + (P_{\text{tra}}^{d_i} D_n / R_{e_{i,j},c}) \tag{7}$$

本文 $P_{\text{com}}^{\text{clo}}$ 为边缘云服务器的执行功耗,则任务在边缘云服务器执行的能耗为:

$$E_n^{c_{\text{com}}} = P_{\text{com}}^{\text{clo}} C_n / f^{\text{clo}} \tag{8}$$

本文 $P_{\text{tra}}^{\text{clo}}$ 为部署边缘云服务器的基站的发射功率,则数据回传能耗 $E_n^{c_{\text{down}}}$ 为:

$$E_n^{c_{\text{down}}} = P_{\text{tra}}^{\text{clo}} D_n / R_{c,e_{i,j}} + P_{\text{tra}}^{e_{i,j}} D_n / R_{e_{i,j},n} \tag{9}$$

此外,数据在无人机和边缘云服务器之间传输阶段和任务执行阶段,UE 处于空载状态,此时的能耗为:

$$E_n^{c_{\text{lid}}} = P^{\text{lid}} \left(\frac{D_n}{R_{e_{i,j},c}} + \frac{C_n}{f^{\text{clo}}} + \frac{D_n}{R_{c,e_{i,j}}} \right) \tag{10}$$

1.3 问题表述

用户 n 任务执行时,系统的时延 T_n 和能耗 E_n 可以分别表示为:

$$T_n = T_n^{\text{loc}} X_n + T_n^{\text{dro}} Y_n^{\text{dro}} + T_n^{\text{rsu}} Y_n^{\text{rsu}} + T_n^{\text{veh}} Y_n^{\text{veh}} + T_n^{c_{\text{dro}}} Z_n^{\text{dro}} + T_n^{c_{\text{rsu}}} Z_n^{\text{rsu}} + T_n^{c_{\text{veh}}} Z_n^{\text{veh}} \tag{11}$$

$$E_n = E_n^{\text{loc}} X_n + E_n^{\text{dro}} Y_n^{\text{dro}} + E_n^{\text{rsu}} Y_n^{\text{rsu}} + E_n^{\text{veh}} Y_n^{\text{veh}} + E_n^{c_{\text{dro}}} Z_n^{\text{dro}} + E_n^{c_{\text{rsu}}} Z_n^{\text{rsu}} + E_n^{c_{\text{veh}}} Z_n^{\text{veh}} \tag{12}$$

式(11)、(12)是只考虑执行单一任务时系统的时延和能耗,执行多任务时系统的时延和能耗可以分别用 T_n^q、E_n^q 表示,q 为任务类型。任务卸载问题可以通过对目标函数求最优解来实现,为此定义本文的优化目标:

$$\begin{cases} \min \sum_{n=1}^{N} \sum_{q=1}^{Q} Y_n^q \\ \text{s.t.} \quad C_1: \lambda^t, \lambda^e \in [0,1], \\ C_2: \lambda^t + \lambda^e = 1, \\ C_3: T_n^q \leqslant \tau^{\max q}, n \in N, q \in Q, \\ C_4: E_n^q \leqslant e^{\max q}, n \in N, q \in Q \end{cases} \tag{13}$$

式中:$Y_n^q = \lambda^t T_n^q + \lambda^e E_n^q$ 为系统总开销;$\lambda^t + \lambda^e = 1$,$\lambda^t, \lambda^e \in [0,1]$,$\lambda^t$、$\lambda^e$ 分别为时延和能耗的权重系数,表示用户对能耗和时延的偏好,分别由任务完成紧迫性和剩余电池电量决定;C_1、C_2 表示权重约束;C_3 表示时延约束;C_4 表示能耗约束。

2　H-PSOGA 任务卸载算法

PSO 和 GA 算法都是经典的群体智能算法。PSO 算法原理简单、搜索速度快,但是容易陷入局部最优;GA 算法具有很强的全局搜索能力,但是收敛精度不高。本文采用先串行后并行的方式,提出了一种基于 PSO 算法和 GA 算法的混合优化算法(hybrid particle swarm optimization and genetic algorithm, H-PSOGA)。

假设粒子群规模为 S,整个粒子群的位置集合表示为 $X = \{x_1, x_2, \cdots, x_S\}$,速度集合表示为 $V = \{v_1, v_2, \cdots, v_S\}$。

粒子编码采用整数编码,每个粒子的位置可表示为 $\boldsymbol{x}_s = [x_s^{n,q}]_{N\times Q}, x_s^{n,q}, s \in S, n \in N, q \in Q$ 为整数并在更新的过程中进行取整,表示粒子 s 执行用户 n 的 q 类型任务的位置。若 $x_s^{n,q} = 0$,则任务本地执行;若 $x_s^{n,q} = j$,则任务在编号为 j 的服务器执行。类似地,定义每个粒子的速度 $\boldsymbol{v}_s = [v_s^{n,q}]_{N\times Q}, v_s^{n,q}, s \in S, n \in N, q \in Q$ 为整数并在更新的过程中进行取整。

H-PSOGA 算法首先进行 PSO 进化,计算每个个体的适应度值并进行排序,然后利用 PSO 融合 GA 进行子代的选择、多点交叉、反向变异,形成新的子代;子代再进行 PSO 进化,并重复上述过程再次形成新的子代;循环往复,直至得到最优解。H-PSOGA 算法优化过程如图 2 所示,m 为粒子群初始规模,算法实现具体步骤为:

1)随机初始化每个个体的速度、位置以及相关参数,对不满足约束条件的个体重新初始化;

2)将个体当前位置设置为个体最优任务分配方案 G_{best},并将适应度值最大的个体的位置设置为群体最优分配方案 H_{best},计算每个个体的适应度值为:

$$F = \frac{1}{\sum_{n=1}^{N}\sum_{q=1}^{Q}\left(\frac{\lambda^t T_n^q}{\tau^{\max q}} + \frac{\lambda^e E_n^q}{e^{\max q}}\right)} \tag{14}$$

式中:λ^t 和 λ^e 为时延和能耗的权重系数;$\tau^{\max q}$ 和 $e^{\max q}$ 分别为 q 类型任务的容忍时延和容忍能耗。

3)根据式(15)、(16)更新个体每一维度的速度与位置并取整:

$$v(t+1) = \omega \cdot v(t) + c_1 \cdot m_{\text{rand}} \cdot (G_{\text{best}} - x(t)) + c_2 \cdot m_{\text{rand}} \cdot (H_{\text{best}} - x(t)) \tag{15}$$

$$x(t+1) = x(t) + v(t+1) \tag{16}$$

式中:ω 为惯性权重;c_1、c_2 为学习因子;m_{rand} 为分布在区间[0,1]的随机数。

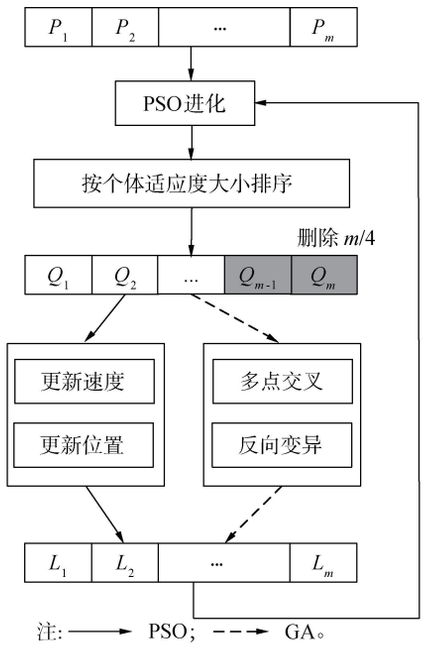


图 2　H-PSOGA 算法优化过程

Fig. 2　H-PSOGA algorithm optimization process

权重 ω 随着适应度值的变化而变化:

$$\omega = \begin{cases} \omega_{\min} - \dfrac{(\omega_{\max} - \omega_{\min}) \cdot (F - F_{\min})}{F_{\text{avg}} - F_{\min}}, & F \leqslant F_{\text{avg}} \\ \omega_{\max}, & F > F_{\text{avg}} \end{cases} \tag{17}$$

式中:F 表示目前的适应度值;F_{avg}、$F_{\min}$ 分别表示目前所有粒子适应度的平均值和最小值。当粒子目标值比较分散时,减小惯性权重,粒子目标值比较集中时,增加惯性权重;

4)判断是否满足 PSO 进化的迭代次数或者收敛精度,若满足条件,则执行步骤 5),否则执行步骤 2),直到满足条件为止;

5)计算每个个体的适应度值并按照从大到小的顺序进行排序;

6)种群选择,将排序后的种群 Q 平均分成 4 个子种群,淘汰适应度值最小的子种群 4,如图 3 所示;

7)多点交叉和反向变异:保留的 3 个子种群进行交叉、变异操作,对每一维度进行取整;

定义平均距离的概念,通过判断交叉父代同一维度的距离与平均距离的大小,本文设计了一种交叉方式。假设 2 个个体分别为 x_1 和 x_2,定义 2 个个体的平均距离为:

$$\Theta \triangleq \frac{1}{N \times Q}\sqrt{\sum_{n=1}^{N}\sum_{q}^{Q}(x_1^{n,q} - x_2^{n,q})^2} \tag{18}$$

若 2 个个体的第(n,q)维距离 $|x_1^{n,q} - x_2^{n,q}|$ 不

小于 2 个个体平均距离 Θ，则第(n,q)维进行交叉，即第(n,q)维的 2 个基因交换位置。交叉方式如图 4 所示。

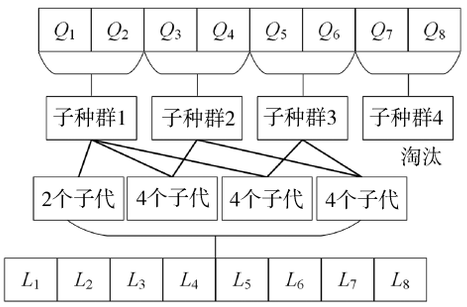
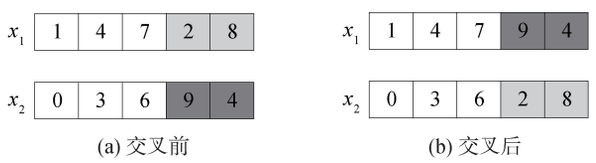


图 3 选择操作示意

Fig. 3 Schematic diagram of select operation



(a) 交叉前 (b) 交叉后

图 4 交叉操作示意

Fig. 4 Schematic diagram of cross operation

由图 4(a)的 2 个个体的基因序列，根据式(18)计算 2 个个体的平均距离为 1.37，图 4(a)中第 4 维和第 5 维的距离大于平均距离，则第 4 维和第 5 维对应的基因分别交换位置，得到图 4(b)的子代。

在实数编码时，等位基因的类型不再是简单的布尔型，故变异不能简单的取反。本文采用基于时变概率的反向学习方式进行均匀变异，使个体跳出局部最优。设个体目前适应度值为 F_s，目前种群的平均适应度值为 F_{avg}，适应度值方差 ψ^2 为：

$$\psi^2 = \frac{1}{S}\sum_{s=1}^{S}(F_s - F_{\text{avg}})^2 \tag{19}$$

当 ψ^2 值小于等于给定的密集度阈值且迭代次数小于迭代阈值时，对个体每一个维度进行反向变异操作。在不影响分析结果的前提下，为了简化符号，省略表达式中的任务维度 Q。假设个体 s 的位置为 $\boldsymbol{x}_s = (x_s^1, x_s^2, \cdots, x_s^N)$，变异后的个体 s 的第 k 维搜索空间表示为：

$$x_s^k(t+1) = m_{\text{rand}} \cdot (a_s^k + b_s^k) - x_s^k(t) \tag{20}$$

式中：m_{rand} 为分布在区间[0,1]的随机数；a_s^k 和 b_s^k 分别为个体 s 的第 k 维搜索空间的下边界和上边界。若反向变异之后个体 s 的第 k 维搜索空间超出搜索边界，则采用随机生成的方法重置：

$$x_s^k(t+1) = m_{\text{rand}}(a_s^k, b_s^k) \tag{21}$$

式中 $m_{\text{rand}}(a_s^k, b_s^k)$ 为在区间 (a_s^k, b_s^k) 上生成随机数。

8)适应度值最大的子种群直接进行 PSO 速度和位置更新生成子代；

9)计算所有子代的适应度值并按照从大到小的顺序进行排序；

10)选择适应度值较大且数量与初始种群相同的子代重新组成新的种群；

11)判断总的迭代次数或者收敛精度是否满足条件，若满足条件，则输出最优解，即最优卸载方案，否则执行步骤 2)。

3 H-PSOGA 算法仿真与结果分析

3.1 仿真参数设置

本文使用 Matlab 2020a 软件对 H-PSOGA 算法性能进行仿真验证。首先用 6 种标准测试函数对 H-PSOGA 算法的性能进行测试，为了便于观察，采用测试函数值的相反数进行绘图。然后将 H-PSOGA 算法应用到系统模型中，通过与基线算法的对比，验证本文算法在解决异构边缘云架构下的多任务卸载中的适用性。表 1 为仿真模型参数。

表 1 仿真参数

Table 1 Simulation parameter

参数	参数取值
任务大小/kB	300~2 000
所需 CPU 周期/GHz	100~1 200
系统带宽/MHz	20
用户发射功率/dBm	20
无人机发射功率/dBm	30
RSU 发射功率/dBm	50
车辆发射功率/dBm	30
环境特征参数 α	9.61
环境特征参数 β	0.16
系统带宽/MHz	20
噪声功率谱密度/(dBm/Hz)	−174
时延权重系数 λ^t	0.5
能耗权重系数 λ^e	0.5

表 2 为 6 种标准测试函数。算法控制参数：粒子群规模 $S=40$，粒子维度 $N=100$，迭代次数 $D=1\,000$，学习因子 $c_1=1.5$，$c_2=2.5$，惯性权重的最大值和最小值为 $\omega_{\max}=0.8$，$\omega_{\min}=0.4$，标准测试函数最优解为 0。因为是 n 维粒子空间，$(0)^n$ 为每个维度的粒子的最优位置都是 0。

3.2 仿真结果分析

惯性权重 ω 可以控制粒子的搜索范围，增大 ω 的值可以提高算法的全局搜索能力，减小 ω 的值可以提高算法的局部搜索能力。对多种 PSO 改进方案进行对比实验，如图 5 所示，随着迭代次数的增加，几种改进 PSO 的方案以收敛速度为代价提高了收敛精度，非线性递减改进权重方案的收敛精度最高，故采用该方案与本文 H-PSOGA 算法进行对比实验。

表 2　标准测试函数

Table 2　Classic test function

函数名	函数表达式	搜索范围	最优点
Sphere	$f=\sum_{i}^{n}x_i^2$	$[-5.12,5.12]$	$(0)^n$
Sum Square	$f=\sum_{i}^{n}ix_i^2$	$[-10,10]$	$(0)^n$
Rosenbrock	$f=\sum_{i}^{n-1}(100(x_{i+1}-x_i^2)^2+(x_i-1)^2)$	$[-5,10]$	$(1)^n$
Rastrigrin	$f=\sum_{i}^{n}(x_i^2-10\cos(2\pi x_i)+10)$	$[-5.12,5.12]$	$(0)^n$
Ackley	$f=-20\exp(-0.2\sqrt{\frac{1}{n}\sum_{i}^{n}x_i^2})-\exp(\frac{1}{n}\sum_{i}^{n}\cos(2\pi x_i))+20+e$	$[-30,30]$	$(0)^n$
Griewank	$f=\sum_{i}^{n}(\frac{x_i}{4\,000})^2-\prod_{i}^{n}\cos(\frac{x_i}{\sqrt{i}})+1$	$[-600,600]$	$(0)^n$

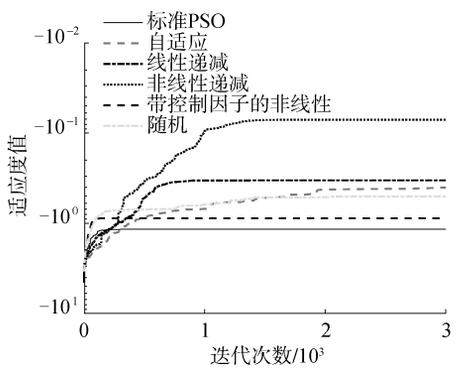
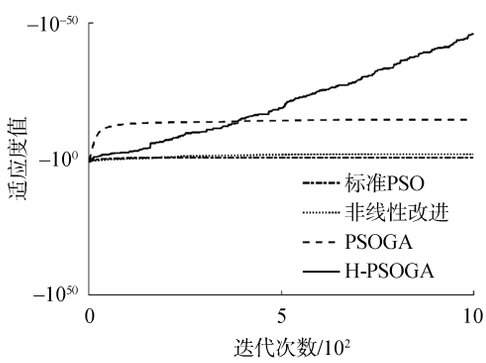


图 5　不同惯性权重适应度值曲线

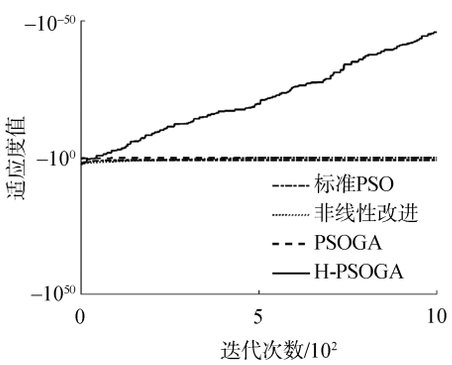
Fig. 5　Different inertia weight fitness value curves

为了更加清晰直观地说明 H-PSOGA 算法的性能,通过 6 种标准测试函数比较标准 PSO、非线性改进权重方案、PSOGA 算法以及本文 H-PSOGA 算法的收敛情况,如图 6 所示。

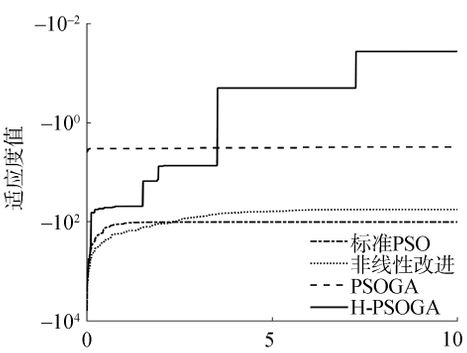
从图 6 可以看出,6 种算法的适应度值均随着迭代次数的增加而逐渐增加。对于任意标准测试函数,标准 PSO、非线性改进权重方案以及 PSOGA 算法的适应度值增加到定值达到稳定,即收敛状态。而对于 H-PSOGA 算法来说,对于不同的标准测试函数,H-PSOGA 算法的适应度值表现出不同的收敛效果。



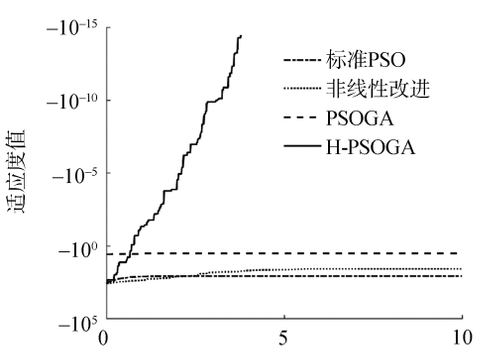
(a) Sphere函数



(b) Sum square函数



(c) Rosenbrock函数



(d) Rastrigin函数

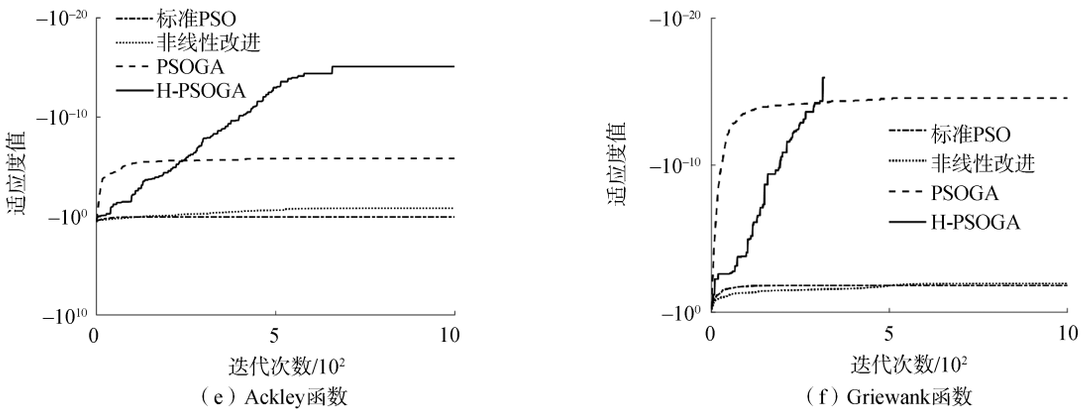


图 6　6 种标准函数测试结果对比

Fig. 6　Comparison of test results of the six standard functions

图 6(a)和(b)中 H-PSOGA 算法适应度值整体呈现增加的趋势,还未达到收敛的状态;图 6(c)和(e)中 H-PSOGA 算法的适应度值收敛到定值,且收敛精度高于其他算法。图 6(d)和(f)中 H-PSOGA 算法在迭代到一定次数时停止迭代,此时得到最优解。

总体来说,H-PSOGA 算法的收敛精度更高,用多种测试函数也证明了 H-PSOGA 算法的普适性。将 H-PSOGA 算法应用到系统网络模型,进行对比实验,如图 7 所示,收敛后的适应度值比标准 PSO 和 PSOGA 算法分别提升 36.5%和 16%。

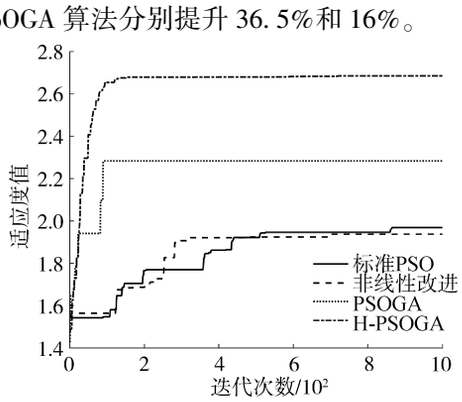


图 7　适应度函数收敛曲线

Fig. 7　Fitness function convergence curve

从图 7 可以看出,随着迭代次数的增加,标准 PSO 算法、非线性改进权重 PSO 算法、PSOGA 算法的适应度值都会增加,最后收敛到相对稳定的状态。相对于基线方案,H-PSOGA 算法的适应度值较大,随着迭代次数的增加收敛速度较快,收敛精度也较高,证明了 H-PSOGA 算法适用于本文提出的异构网络模型,同时也验证了 H-PSOGA 算法的性能。

图 8 所示为不同类型业务的卸载决策。图中数字 1~5 为 5 种具有不同任务复杂度、容忍时延、容忍能耗的计算任务,任务类型序号越大,任务复杂度越高。任务的复杂度越大,本地执行任务的概率越小,卸载到无人机、路边单元、车辆以及边缘云的概率越大,这是因为本地无法满足用户业务的需求,需要将任务卸载到计算能力更大的边缘节点或边缘云服务器。

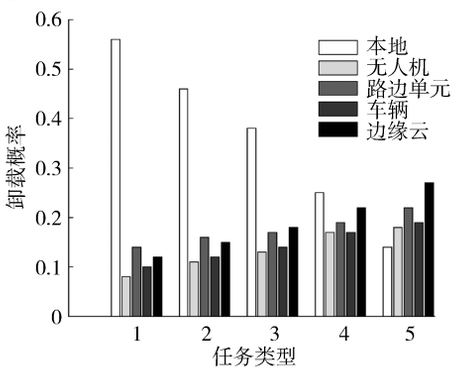


图 8　不同类型任务卸载方案概率分布

Fig. 8　Probability distribution of different types of task offloading schemes

图 9 为系统平均开销与用户数量的关系曲线。从图 9 中可以看出,随着用户数量的增加,系统的平均开销逐渐增大。

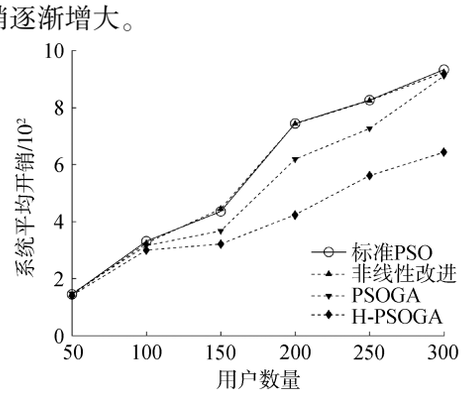


图 9　系统平均开销与用户数量关系

Fig. 9　The relationship between the average system overhead and the number of users

在用户数量相同的情况下,与对照方案相比,H-PSOGA 算法的系统平均开销最小。在用户数较少时,几种方案的性能较为接近,H-PSOGA 算法的性能优势不是很明显;当用户数超过 150 时,系统性能提升较为明显,例如在用户数为 200 时,相比标

准 PSO、非线性改进算法和 PSOGA 算法的性能提升分别为 43.06%、42.98% 和 31.52%;在用户数为 300 时,相对上述 3 种对照方案的性能提升分别为 31.08%、30.38%和 29.4%。

4　结论

1)提出混合优化算法 H-PSOGA 实现异构边缘云架构下的多任务卸载,有效提高了收敛精度,降低了系统总开销。

2)本文在研究任务卸载时将通信资源和计算资源设为固定值,而在实际情况中,需要联合任务卸载和资源分配进行优化,将是后续研究重点解决的问题。

参考文献:

[1] IMT-2020(5G)推进组. 5G 愿景与需求白皮书[EB/OL]. 北京: IMT-2020(5G)推进组, 2014[2021-10-22]. http://www.imt2020.org.cn/zh/documents/1.
IMT-2020(5G) Promotion Group. 5G Vision and Requirements White Paper[EB/OL]. Beijing: IMT-2020(5G) Promotion Group. [2021-10-22]. http://www.imt2020.org.cn/zh/documents/1.

[2] HU Y C, PATEL M, SABELLA D, et al. Mobile edge computing—A key technology towards 5G[J]. ETSI white paper, 2015, 11(11): 1-16.

[3] MACH P, BECVAR Z. Mobile edge computing: a survey on architecture and computation offloading[J]. IEEE communications surveys & tutorials, 2017, 19(3): 1628-1656.

[4] 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.

[5] 谢人超, 廉晓飞, 贾庆民, 等. 移动边缘计算卸载技术综述[J]. 通信学报, 2018, 39(11): 138-155.
XIE Renchao, LIAN Xiaofei, JIA Qingmin, et al. Survey on computation offloading in mobile edge computing[J]. Journal on communications, 2018, 39(11): 138-155.

[6] 尼俊红, 吕梦楠. 无人机与车辆协助下的分布式多任务边缘计算卸载算法[J]. 科学技术与工程, 2021, 21(3): 1045-1051.
NI Junhong, LYU Mengnan. Distributed multi-task edge computation offloading algorithm assisted by unmanned aerial vehicle and vehicle[J]. Science technology and engineering, 2021, 21(3): 1045-1051.

[7] GU Bo, ZHOU Zhenyu. Task offloading in vehicular mobile edge computing: a matching-theoretic framework[J]. IEEE vehicular technology magazine, 2019, 14(3): 100-106.

[8] DU Jianbo, ZHAO Liqiang, FENG Jie, et al. Computation offloading and resource allocation in mixed fog/cloud computing systems with Min-max fairness guarantee[J]. IEEE transactions on communications, 2018, 66(4): 1594-1608.

[9] WANG Rui, CAO Yong, NOOR A, et al. Agent-enabled task offloading in UAV-aided mobile edge computing[J]. Computer communications, 2020, 149: 324-331.

[10] YU Shuai, CHEN Xu, YANG Lei, et al. Intelligent edge: leveraging deep imitation learning for mobile edge computation offloading[J]. IEEE wireless communications, 2020, 27(1): 92-99.

[11] KE Hongchang, WANG Jian, DENG Lingyue, et al. Deep reinforcement learning-based adaptive computation offloading for MEC in heterogeneous vehicular networks[J]. IEEE transactions on vehicular technology, 2020, 69(7): 7916-7929.

[12] 王妍, 葛海波, 冯安琪. 云辅助移动边缘计算中的计算卸载策略[J]. 计算机工程, 2020, 46(8): 27-34.
WANG Yan, GE Haibo, FENG Anqi. Computation offloading strategy in cloud-assisted mobile edge computing[J]. Computer engineering, 2020, 46(8): 27-34.

[13] 罗斌, 于波. 移动边缘计算中基于粒子群优化的计算卸载策略[J]. 计算机应用, 2020, 40(8): 2293-2298.
LUO Bin, YU Bo. Computation offloading strategy based on particle swarm optimization in mobile edge computing[J]. Journal of computer applications, 2020, 40(8): 2293-2298.

[14] WU Jinze, CAO Zhiying, ZHANG Yingjun, et al. Edge-cloud collaborative computation offloading model based on improved partical swarm optimization in MEC[C]//2019 IEEE 25th International Conference on Parallel and Distributed Systems (ICPADS). Piscataway, NJ: IEEE, 2019: 959-962.

[15] WEI Qiuyue, LIU Li'ang, WEI Fansi, et al. Computational offloading Strategy based on Dynamic Particle Swarm for Multi-User Mobile Edge Computing[C]//2019 IEEE Symposium Series on Computational Intelligence (SSCI). Piscataway, NJ: IEEE, 2019: 2890-2896.

[16] ADHIKARI M, SRIRAMA S N, AMGOTH T. Application offloading strategy for hierarchical fog environment through swarm optimization[J]. IEEE internet of things journal, 2020, 7(5): 4317-4328.

本文引用格式:

尼俊红,臧云. 异构边缘云架构下的多任务卸载算法[J]. 哈尔滨工程大学学报,2024,45(4):800-807.
NI Junhong, ZANG Yun. Multitask offloading algorithm under heterogeneous edge cloud architecture[J]. Journal of Harbin Engineering University, 2024,45(4):800-807.