中国科技期刊卓越行动计划项目入选期刊

控制与决策

CONTROL AND DECISION

不确定性感知的边缘计算任务调度算法

尹璐, 周俊龙, 孙晋, 吴泽彬

引用本文:
尹璐,周俊龙,孙晋,吴泽彬. 不确定性感知的边缘计算任务调度算法[J]. 控制与决策, 2024, 39(7): 2405−2413.

在线阅读 View online: https://doi.org/10.13195/j.kzyjc.2023.1055

您可能感兴趣的其他文章

Articles you may be interested in

不确定性感知的边缘计算任务调度算法

尹 璐, 周俊龙, 孙 晋†, 吴泽彬

(南京理工大学 计算机科学与工程学院，南京 210094)

摘 要: 任务执行时长的不确定性是设计任务调度算法时的一个重要问题,关系到调度方案能否满足任务的截止时间要求. 鉴于此,研究不确定性感知的边缘计算任务调度问题,以最小化边缘提供商开销为优化目标建立任务调度问题的优化模型. 该模型将任务执行时长建模为随机变量并推导出任务完成时间的完整概率分布,引入关于任务截止时间的概率约束,以可调节的概率阈值保证任务按时完成. 为求解该问题,进一步提出基于蝙蝠算法搜索策略的元启发式算法,包含两个关键的算法组件,映射算子实现蝙蝠空间与调度解空间的关联,评估算子实现候选解可行性的判定和优化目标值的计算. 基于对比实验的仿真结果表明,所提出算法能够得到高质量的任务调度方案.

关键词: 边缘计算；任务调度；不确定性；概率约束；随机优化；蝙蝠算法

中图分类号: TP273 **文献标志码:** A

DOI: 10.13195/j.kzyjc.2023.1055

引用格式: 尹璐,周俊龙,孙晋,等. 不确定性感知的边缘计算任务调度算法[J]. 控制与决策, 2024, 39(7): 2405-2413.

Uncertainty-aware task scheduling algorithm in edge computing environments

YIN Lu, ZHOU Jun-long, SUN Jin†, WU Ze-bin

(School of Computer Science and Engineering，Nanjing University of Science and Technology，Nanjing 210094，China)

Abstract: When designing task scheduling algorithms, the uncertainty of task execution duration is an important concern, which impacts whether the task completion time can meet its deadline constraint. In this paper, we study the uncertainty-aware task scheduling problem in edge computing systems and formulate a scheduling optimization model with the objective of minimizing the edge provider's monetary cost. By modeling the task execution duration as a random variable and deriving the complete probability distribution of the task completion time, the formulated model introduces a probabilistic constraint on the task's deadline to guarantee the task is completed on time with an adjustable probability threshold. To solve this problem, this paper further proposes a metaheuristic algorithm that is built upon the bat algorithm's search strategy. The proposed algorithm contains two key algorithmic components. The mapping operator enables the connection between the bat space and the scheduling solution space, and the evaluation operator enables the determination of the candidate solution's feasibility and the calculation of the optimization objective value. Simulation results based on comparative experiments demonstrate that the proposed algorithm is capable of obtaining high-quality task scheduling solutions.

Keywords: edge computing；task scheduling；uncertainty；probabilistic constraint；stochastic optimization；bat algorithm

0 引 言

作为一种在网络边缘部署计算资源和服务的计算架构,边缘计算成为一种缓解移动设备计算负担、降低服务延迟的有效解决方案[1]. 考虑到计算资源有限的边缘服务器可能无法承担计算密集型应用的快速处理,边缘提供商可以将部分任务提交到云计算中心,以按需付费的方式租用云平台的计算资源[2-4]. 从边缘提供商的角度,为了在保证服务质量的前提下降低租赁开销,高效的任务调度算法至关重要.

目前,边缘计算系统中的任务调度问题已经引起

收稿日期: 2023-07-27；**录用日期:** 2023-12-23.

基金项目: 江苏省重点研发计划项目 (BE2022065-2)；江苏省创新支撑计划项目 (BZ2023046)；教育部产学研创新基金项目 (2020ITA03002, 2021ITA01004)；国家自然科学基金项目 (62172224, U23B2006)；江苏省自然科学基金项目 (BK20220138).

责任编委: 邓庆绪.

†通讯作者. E-mail: sunj@njust.edu.cn.

国内外学者们极大的研究兴趣[5]. 在静态调度的研究范畴,现有多数调度方法将任务的执行时长视为一个可预先获知的固定值,但是完全准确地预判任务的执行时长是具有挑战性的[6-7]. 除了程序执行时间固有的不确定性,还有以下两方面的原因:一方面,任务程序通常涉及条件指令和循环操作. 受条件指令的影响,不同的输入会导致不同的程序分支和不同的循环次数,从而导致任务程序的计算量不同. 因此,在不同的输入条件下,任务持续时间会有所不同[8]. 另一方面,边缘或云服务器虚拟机实例的处理速度会发生波动. 这是由于在同一物理服务器上部署了多个虚拟机,这些虚拟机同步使用物理机的计算和存储资源,由此导致任务执行时间的不确定性[9]. 在任务执行时长不确定的影响下,若调度算法将其视为固定值,则容易造成任务截止时间约束的违背或计算资源超额分配过于保守的调度方案.

针对以上问题,本文研究边缘计算环境中不确定性感知的任务调度问题,从边缘提供商的角度建立最小化云服务器租用开销的任务调度优化模型. 该模型将任务调度所涉及的多个变量建模为随机变量,并基于数学推导得到任务完成时间的概率分布. 通过引入关于任务截止时间的概率约束,模型在概率的意义上保证具有不确定特征的任务按时完成. 为求解此问题,提出一个基于蝙蝠算法搜索策略的元启发式任务调度算法,包含两个重要的算法模块:映射模块为蝙蝠个体生成调度解,实现蝙蝠空间与调度解空间的关联;评估模块判定候选解的可行性并计算优化目标值,实现个体适应度值的确定. 实验结果表明,在不确定性任务执行时长的影响下,所提出算法能够得到高质量的调度方案,有效降低边缘提供商开销.

1 相关工作

近年来,国内外学者针对边缘计算环境中的任务调度问题开展了大量的研究工作. 在现有文献中,从系统架构的角度看,部分方法研究端-边两层架构的边缘计算系统[10],部分方法研究端-边-云三层架构的边缘计算系统[11]. 从优化目标的角度看,部分方法致力于能耗优化[12],部分方法关注时延最小化[13],部分方法着重增加利润或降低成本[14],还有部分方法考虑了多目标优化问题[15]. 杨子轩等[16]针对多用户多边缘服务器的边缘计算系统中的任务调度问题展开研究,设计了一种基于果蝇优化算法的计算卸载方法,以优化任务时延和系统能耗. Meng等[17]设计了任务与计算资源之间的3层安全模型,并提出一种异构边缘云平台上安全性感知的任务调度方法,该方法基于粒子群优化算法设计了参数设置方案和分布式搜索策略. 雷雪梅等[18]研究多类型边缘节点的任务调度问题,以加权和的形式优化调度延迟、能耗及总成本,并提出了基于线性规划松弛的三级卸载决策算法. Lin等[19]针对深度神经网络应用在端-边-云计算系统中的调度问题,提出了一种混合元启发式算法. 该算法应用遗传算法中的突变算子来增加粒子群算法中的种群多样性从而抑制算法的过早收敛. 上述方法都是基于任务执行时间是确定值这一假设,然而,考虑到物理服务器上虚拟机性能波动和任务程序的复杂逻辑,任务的准确执行时间会无法准确预测,因此确定性调度算法不适用于求解具有不确定性特征的任务调度问题.

针对不确定性任务的调度问题,一部分研究方法致力于对任务执行时间进行更准确地估算和刻画. Novak等[20]研究了以完工时间为优化目标的任务执行时间不确定的调度问题,其中每个任务都包含一组备选的执行时间. Jia等[21]根据虚拟机设置和实际工作流的历史数据,开发了一种任务执行时间估算模型. 基于该估算模型,进一步提出了一种自适应蚁群优化算法,用于调度有截止日期约束的工作流应用. 另外,从随机优化的角度进行建模和求解是研究方向之一. 为了调度网格计算系统中具有随机执行时间的任务,Tang等[22]根据执行时间的均值和方差建立一个随机模型,并提出了一种基于启发式的随机异构最早完成时间算法. Yang等[23]针对异构计算系统中具有随机执行时间的任务调度问题研究,引入均衡时间与能量的重要性比值作为性能指标,并提出了一个随机算法优化该指标以提高调度质量. 相较于现有的不确定性感知的研究方法,本文的显著特点体现在两个方面,分别是由数学分析方法驱动的调度决策高效性以及基于完整概率分布信息和可调节概率阈值的高适用性. 首先,基于数学推导获得不确定性变量的分布信息,相比基于抽样的方法具有更高的计算效率,对于大规模调度问题或者需要快速响应的应用场景至关重要;其次,引入概率约束施加于已知完整概率分布信息的不确定性变量上,在保证截止时间约束有效性的同时实现对概率分布信息的充分利用,在方法论上更契合随机优化的本质. 另外,本文模型在概率约束中设置一个可调节的概率阈值,进一步实现了具体应用场景下具有不同截止时间敏感度应用任务的可调度性.

2 问题模型

2.1 边缘计算系统模型

如图1所示,该系统由终端设备、边缘服务器(edge servers, ESs)和云计算中心构成. 由于边缘服务器的计算资源有限,边缘提供商需要决定将来自终端设备上的计算任务卸载到某一个边缘服务器上执行,还是进一步传输到云计算中心执行. 设系统中边缘服务器的集合为$\mathrm{ES}=\{\mathrm{es}_1,\mathrm{es}_2,\ldots,\mathrm{es}_S\}$,其中边缘服务器总数为$S$,$\mathrm{es}_s(s=1,2,\ldots,S)$为第$s$个边缘服务器. 云计算中心的大规模计算资源能够保证提交上来的任务得到快速处理,并以按需付费的方式向边缘提供商收取费用,单位时间的价格为p.

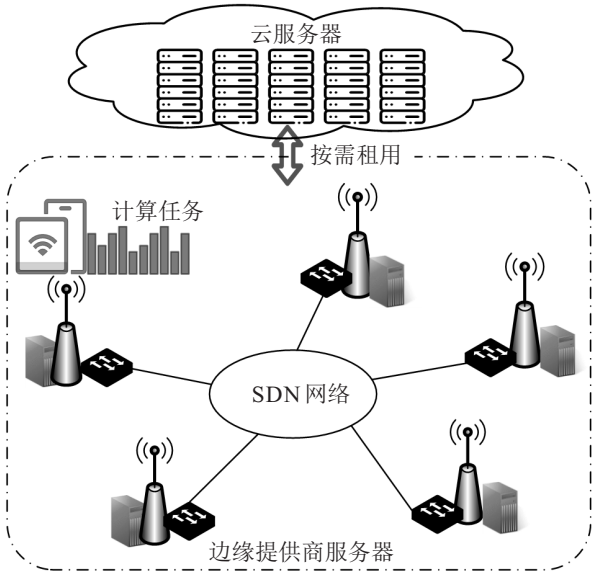


图1　一个边缘计算系统的示例

2.2 计算任务模型

设待调度的任务为计算密集型的众包任务(bag-of-tasks, BoT)应用程序(简称众包应用),是由许多独立任务组成的可高度并行处理的应用类型,在数据挖掘、计算机视觉以及参数扫描等领域广泛存在[24]. 一个众包应用中的各任务之间没有依赖关系,可以在没有同步或通信的情况下并行处理或按照任何顺序执行. 设众包应用有N个,构成的集合为$A=\{a_1,a_2,\ldots,a_N\}$. 对于$a_i(i=1,2,\ldots,N)$,其特征可由二元组表示为$a_i=\langle\lambda_i,\mathrm{dl}_i\rangle$. 其中:$\lambda_i$为$a_i$所包含的任务数,$\mathrm{dl}_i$为截止时间.

众包应用a_i中的所有任务可以表示为$a_i=\{t_{i1},t_{i2},\ldots,t_{i\lambda_i}\}$. 对于每个任务$t_{ij}$,其特征可由三元组表示为

$$t_{ij}=\langle\widetilde{\mathrm{et}}_{ij}^{\mathrm{ES}},\widetilde{\mathrm{et}}_{ij}^{\mathrm{C}},\mathrm{tt}_{ij}^{\mathrm{ES}\Rightarrow\mathrm{C}}\rangle.$$

其中:$\widetilde{\mathrm{et}}_{ij}^{\mathrm{ES}}$表示该任务在边缘服务器上的执行时间,$\widetilde{\mathrm{et}}_{ij}^{\mathrm{C}}$表示该任务在云服务器上的执行时间,$\mathrm{tt}_{ij}^{\mathrm{ES}\Rightarrow\mathrm{C}}$表示该任务从边缘服务器传输到云服务器的时间. $\widetilde{\mathrm{et}}_{ij}^{\mathrm{ES}}$和$\widetilde{\mathrm{et}}_{ij}^{\mathrm{C}}$是服从正态分布的随机变量,可以用均值和方差来表示,即$\widetilde{\mathrm{et}}_{ij}^{\mathrm{ES}}\sim N(\mu_{ij}^{\mathrm{ES}},\sigma_{ij}^{\mathrm{ES}\,2})$,$\widetilde{\mathrm{et}}_{ij}^{\mathrm{C}}\sim N(\mu_{ij}^{\mathrm{C}},\sigma_{ij}^{\mathrm{C}\,2})$. 基于任务执行结果的数据量远小于其本身的数据量这一假设,在此忽略传输结果所涉及的能耗和时间开销.

2.3 调度模型

设$x_{ij,k}(k=1,2,\ldots,S+1)$为本调度问题模型的决策变量,即任务被分配到哪一个计算设备上执行. $x_{ij,k}=1\,(k\leqslant S)$表示任务$t_{ij}$在边缘提供商的第$k$个服务器上执行,若$x_{ij,S+1}=1$则表示$t_{ij}$被外包到云服务器上执行. 由于一个任务仅能被分配到一个计算设备上执行,有

$$\sum_{k=1}^{S+1}x_{ij,k}=1,\ \forall i,j. \tag{1}$$

分配到云服务器上的任务能够在数据传输完成后立刻开始执行,被分配到同一边缘服务器上的任务需要按序依次执行,因此任务的预期执行开始时间$\widetilde{\mathrm{st}}_{ij}$为

$$\widetilde{\mathrm{st}}_{ij}=\begin{cases}\displaystyle\sum_{t_{uv}\in\mathrm{pre}(t_{ij})}\widetilde{\mathrm{et}}_{uv}^{\mathrm{ES}},\ x_{ij,S+1}\neq 1;\\ \mathrm{tt}_{ij}^{\mathrm{ES}\Rightarrow\mathrm{C}},\ x_{ij,S+1}=1.\end{cases} \tag{2}$$

对于分配到边缘服务器上的任务,其预期开始执行时间满足

$$\widetilde{\mathrm{st}}_{ij}\sim N\Big(\sum_{t_{uv}\in\mathrm{pre}(t_{ij})}\mu_{uv}^{\mathrm{ES}},\sum_{t_{uv}\in\mathrm{pre}(t_{ij})}\sigma_{uv}^{\mathrm{ES}\,2}\Big),\quad \forall i,j,x_{ij,S+1}\neq 1. \tag{3}$$

那么任务的预期结束时间$\widetilde{\mathrm{ft}}_{ij}$可计算为

$$\widetilde{\mathrm{ft}}_{ij}=\begin{cases}\widetilde{\mathrm{st}}_{ij}+\widetilde{\mathrm{et}}_{ij}^{\mathrm{ES}},\ x_{ij,S+1}\neq 1;\\ \mathrm{tt}_{ij}^{\mathrm{ES}\Rightarrow\mathrm{C}}+\widetilde{\mathrm{et}}_{ij}^{\mathrm{C}},\ x_{ij,S+1}=1.\end{cases} \tag{4}$$

$\widetilde{\mathrm{ft}}_{ij}$满足如下分布:

$$\widetilde{\mathrm{ft}}_{ij}\sim\begin{cases}N\Big(\displaystyle\sum_{t_{uv}\in\mathrm{pre}(t_{ij})}\mu_{uv}^{\mathrm{ES}}+\mu_{ij}^{\mathrm{ES}},\sum_{t_{uv}\in\mathrm{pre}(t_{ij})}\sigma_{uv}^{\mathrm{ES}\,2}+\sigma_{ij}^{\mathrm{ES}\,2}\Big),\\ \qquad x_{ij,S+1}\neq 1;\\ N(\mathrm{tt}_{ij}^{\mathrm{ES}\Rightarrow\mathrm{C}}+\mu_{ij}^{\mathrm{C}},\sigma_{ij}^{\mathrm{C}\,2}),\ x_{ij,S+1}=1.\end{cases} \tag{5}$$

对于众包应用a_i,其完成时间$\widetilde{\mathcal{C}}_i$是它所包含所有任务完成时间的最大值,即

$$\widetilde{\mathcal{C}}_i=\max_{j=1,2,\ldots,\lambda_i}\widetilde{\mathrm{ft}}_{ij},\ \forall i. \tag{6}$$

由于各个任务的完成时间$\widetilde{\mathrm{ft}}_{ij}$是服从正态分布的随机变量,众包应用的完成时间$\widetilde{\mathcal{C}}_i$是多个正态分

布变量的最大值. 对于两个正态分布随机变量的最大值,文献[25]认为其依旧服从正态分布,且给出了Clark公式用于计算该最大值的均值和方差. 本文以迭代的方式使用Clark公式,进而获得众包应用完成时间$\widetilde{\mathcal{C}}_i$的概率分布信息. 另外,因为众包应用中的各个任务彼此独立,所以任意两个任务完成时间的相关系数为零,即$\rho\left(\widetilde{\mathrm{ft}_{iu}},\widetilde{\mathrm{ft}_{iv}}\right)=0, 1\leqslant u\leqslant v$.

以t_{i1}和t_{i2}为例,$\max\{\widetilde{\mathrm{ft}_{i1}},\widetilde{\mathrm{ft}_{i2}}\}$的均值和方差计算如下:

$$\begin{aligned}&E[\max\{\widetilde{\mathrm{ft}_{i1}},\widetilde{\mathrm{ft}_{i2}}\}]=\\&E[\widetilde{\mathrm{ft}_{i1}}]\phi(\xi_{1,2})+E[\widetilde{\mathrm{ft}_{i2}}]\phi(-\xi_{1,2})+\varepsilon_{1,2}\psi(\xi_{1,2}),\end{aligned}\tag{7}$$

$$\begin{aligned}&\mathrm{Var}[\max\{\widetilde{\mathrm{ft}_{i1}},\widetilde{\mathrm{ft}_{i2}}\}]=\\&E[\max\{\widetilde{\mathrm{ft}_{i1}},\widetilde{\mathrm{ft}_{i2}}\}^2]-E^2[\max\{\widetilde{\mathrm{ft}_{i1}},\widetilde{\mathrm{ft}_{i2}}\}]=\\&(E^2[\widetilde{\mathrm{ft}_{i1}}]+\mathrm{Var}[\widetilde{\mathrm{ft}_{i1}}])\phi(\xi_{1,2})+\\&(E^2[\widetilde{\mathrm{ft}_{i2}}]+\mathrm{Var}[\widetilde{\mathrm{ft}_{i2}}])\phi(-\xi_{1,2})+\\&(E[\widetilde{\mathrm{ft}_{i1}}]+E[\widetilde{\mathrm{ft}_{i2}}])\varepsilon_{1,2}\psi(\xi_{1,2})-\\&E^2[\max\{\widetilde{\mathrm{ft}_{i1}},\widetilde{\mathrm{ft}_{i2}}\}].\end{aligned}\tag{8}$$

其中

$$\begin{aligned}&\xi_{1,2}=E([\widetilde{\mathrm{ft}_{i1}}]-E[\widetilde{\mathrm{ft}_{i2}}])/\varepsilon_{1,2},\\&\varepsilon_{1,2}=\sqrt{\mathrm{Var}[\widetilde{\mathrm{ft}_{i1}}]+[\widetilde{\mathrm{ft}_{i2}}]},\\&\psi(t)=\frac{1}{\sqrt{2\pi}}\mathrm{e}^{-\frac{t^2}{2}},\ \phi(x)=\int_{-\infty}^{x}\psi(t)\mathrm{d}t.\end{aligned}$$

基于式(7)和(8)可知,多个任务预期完成时间的分布能够通过迭代计算的方式获得. 以t_{i1}、t_{i2}和t_{i3}为例,$\max\{\widetilde{\mathrm{ft}_{i1}},\widetilde{\mathrm{ft}_{i2}},\widetilde{\mathrm{ft}_{i3}}\}$的均值和方差计算如下:

$$\begin{aligned}&E[\max\{\widetilde{\mathrm{ft}_{i1}},\widetilde{\mathrm{ft}_{i2}},\widetilde{\mathrm{ft}_{i3}}\}]=\\&E[\widetilde{\mathrm{ft}_{i1}},\widetilde{\mathrm{ft}_{i2}}]\phi(\xi_{1,2,3})+\\&E[\widetilde{\mathrm{ft}_{i3}}]\phi(-\xi_{1,2,3})+\varepsilon_{1,2,3}\psi(\xi_{1,2,3}),\end{aligned}\tag{9}$$

$$\begin{aligned}&\mathrm{Var}[\max\{\widetilde{\mathrm{ft}_{i1}},\widetilde{\mathrm{ft}_{i2}},\widetilde{\mathrm{ft}_{i3}}\}]=\\&E[\max\{\widetilde{\mathrm{ft}_{i1}},\widetilde{\mathrm{ft}_{i2}},\widetilde{\mathrm{ft}_{i3}}\}^2]-E^2[\max\{\widetilde{\mathrm{ft}_{i1}},\widetilde{\mathrm{ft}_{i2}},\widetilde{\mathrm{ft}_{i3}}\}]=\\&(E^2[\max\{\widetilde{\mathrm{ft}_{i1}},\widetilde{\mathrm{ft}_{i2}}\}]+\mathrm{Var}[\max\{\widetilde{\mathrm{ft}_{i1}},\widetilde{\mathrm{ft}_{i2}}\}])\phi(\xi_{1,2,3})+\\&(E^2[\widetilde{\mathrm{ft}_{i3}}]+\mathrm{Var}[\widetilde{\mathrm{ft}_{i3}}])\phi(-\xi_{1,2,3})+\\&(E[\max\{\widetilde{\mathrm{ft}_{i1}},\widetilde{\mathrm{ft}_{i2}}\}]+E[\widetilde{\mathrm{ft}_{i3}}])\varepsilon_{1,2,3}\psi(\xi_{1,2,3})-\\&E^2[\max\{\widetilde{\mathrm{ft}_{i1}},\widetilde{\mathrm{ft}_{i2}},\widetilde{\mathrm{ft}_{i3}}\}].\end{aligned}\tag{10}$$

其中

$$\begin{aligned}&\varepsilon_{1,2,3}=\sqrt{\mathrm{Var}[\max\{\widetilde{\mathrm{ft}_{i1}},\widetilde{\mathrm{ft}_{i2}}\}]+\mathrm{Var}[\widetilde{\mathrm{ft}_{i3}}]},\\&\xi_{1,2,3}=(E[\max\{\widetilde{\mathrm{ft}_{i1}},\widetilde{\mathrm{ft}_{i2}}\}]-E[\widetilde{\mathrm{ft}_{i3}}])/\varepsilon_{1,2,3}.\end{aligned}$$

根据上述方法,对于包含λ_i个任务的众包应用a_i,其完成时间$\widetilde{\mathcal{C}}_i$的均值$E[\widetilde{\mathcal{C}}_i]$和方差$\mathrm{Var}[\widetilde{\mathcal{C}}_i]$可以通过迭代地使用$\lambda_i-1$次Clark公式得到. 基于$\widetilde{\mathcal{C}}_i$的概率分布,本模型引入一个关于众包应用截止时间的概率约束

$$\mathrm{prob}\{\widetilde{\mathcal{C}}_i<d_i\}=\alpha,\ \forall i.\tag{11}$$

其中:$\widetilde{\mathcal{C}}_i\sim N(E[\widetilde{\mathcal{C}}_i],\mathrm{Var}[\widetilde{\mathcal{C}}_i])$;$\alpha$为可定义的概率阈值,表示众包应用按时完成的概率要求,α越高表示应用对截止时间的要求越为严格. 考虑到任务执行时间的随机性特征,式(11)基于其完成时间的概率分布信息对其按时完成的概率进行约束,从随机优化的角度上满足每个众包应用的截止时间要求.

假设边缘提供商给云平台的通讯费用为包月付费,因此在不考虑通讯支出的情况下,为最小化边缘提供商的云服务器租用开销,本模型的优化目标为

$$E[\mathrm{cost}]=\sum_{i=1}^{N}\sum_{j=1}^{\lambda_i}x_{ij,S+1}p\mu_{ij}^{\mathrm{C}}.\tag{12}$$

综上所述,本任务调度优化模型为

$$\begin{aligned}&\min\ E[\mathrm{cost}].\\&\text{s.t. }\mathrm{prob}\{\widetilde{\mathcal{C}}_i<d_i\}=\alpha,\ \forall i;\\&\sum_{k=1}^{S+1}x_{ij,k}=1,\ \forall i,j;\\&x_{ij,k}\in\{0,1\},\ \forall\, i,j,k.\end{aligned}\tag{13}$$

3 算法描述

为求解上述任务调度问题,设计基于蝙蝠算法搜索策略的不确定性感知任务调度算法(uncertainty-aware task scheduling algorithm, UATS). 首先在算法初始化阶段,给出详细的种群初始化方法和初始调度解生成规则;然后进入迭代搜索过程,对于种群中的蝙蝠个体,通过频率和速度等参数的更新得到新的个体位置,为了将蝙蝠空间与调度解空间联系起来,设计一种基于当前最优解的映射算子;为了评估个体的适应度值,评估算子为任务分配计算资源,进而判断候选解的可行性并计算租赁开销;最后在到达最大迭代次数时算法结束,并返回最优任务调度方案. 具体如下.

3.1 算法初始化

UATS初始化包括种群中所有个体的初始化和初始调度解的生成. 设蝙蝠种群为$B=\{b_1,b_2,\ldots,b_\Lambda\}$. 其中:$\Lambda$表示种群规模,$b_m(m=1,2,\ldots,\Lambda)$表示第$m$个蝙蝠个体. 对于任意一个蝙蝠个体$b_m$,其频率$f_m$、速度$v_m$、响度$A_m$、脉冲发射率$r_m$和位置$\mathrm{pos}_m$属性在一定取值区间内随机生成.

在本文中,调度解的表达方式为由所有待调度

任务组成的任务序列,也称调度解序列或调度解. 以蝙蝠个体b_m为例,其在解空间对应的调度解序列为$\Pi_m = \{\pi_m^1, \pi_m^2, \ldots, \pi_m^{\sum_{i=1}^{N}\lambda_i}\}$,其中$\sum_{i=1}^{N}\lambda_i$为所有众包应用的任务总数. Π_m中每个元素为一个任务,即$\pi_m^n \in T = \{t_{ij}|i=1,2,\ldots,N, j=1,2,\ldots,\lambda_i\}$. 为了给算法提供一个良好的初始调度解,本文考虑到任务在其截止时间之前完成的紧迫性,为每个任务t_{ij}定义截止时间违背概率(deadline violation probability, DVP),计算方式如下:

$$\mathrm{DVP}_{ij} = \mathrm{prob}\{\widetilde{\mathrm{et}_{ij}^{\mathrm{ES}}} > \mathrm{dl}_i\} = \int_{\mathrm{dl}_i}^{+\infty} \widetilde{\mathrm{et}_{ij}^{\mathrm{ES}}}(t)\mathrm{d}t. \quad (14)$$

按照DVP值以非升序的方式对所有任务进行排序生成的初始调度解,使得在计算资源分配时,截止时间较为紧迫的任务将被赋予更高的优先级.

3.2 调度解的更新

在调度解更新前,算法首先进行个体位置的更新. 以第t次迭代过程中的蝙蝠b_m为例,UATS算法更新其频率、速度和位置如下:

$$f_m^t = f_{\min} + (f_{\max} - f_{\min})\beta, \quad (15)$$

$$v_m^t = v_m^{t-1} + (\mathrm{pos}_m^{t-1} - \mathrm{pos}^*)f_m^t, \quad (16)$$

$$\mathrm{pos}_m^t = \mathrm{pos}_m^{t-1} + v_m^t. \quad (17)$$

其中:$\beta \in [0,1]$为均匀分布的随机数,$f_{\min}$和$f_{\max}$为频率的最大值和最小值,pos_m^{t-1}和pos_m^t分别为蝙蝠b_m在第$t-1$次和第t次迭代中的位置,pos^*为当前最佳的蝙蝠个体位置,v_m^{t-1}和v_m^t分别为蝙蝠b_m在第$t-1$次和第t次迭代中的速度. 按照下式进行局部搜索:

$$\mathrm{pos}_{\mathrm{new}} = \mathrm{pos}_{\mathrm{old}} + \varsigma A_m^t, \quad (18)$$

其中$\varsigma \in (-1,1)$为均匀分布的随机数. 该过程产生两个随机数rand_1和rand_2. 若$\mathrm{rand}_1 > r_m$,则该蝙蝠在最优解附近随机飞行,即$\mathrm{pos}_{\mathrm{old}}$为当前最优解$\mathrm{pos}^*$,否则$\mathrm{pos}_{\mathrm{old}}$为$\mathrm{pos}_m^t$. 若$\mathrm{rand}_2 < A_i$且$\mathrm{pos}_{\mathrm{new}}$的适应度值优于当前最优解的适应度值,则接受$\mathrm{pos}_{\mathrm{new}}$作为蝙蝠$b_m$的新位置,并更新其响度与脉冲发射率为

$$A_m^t = \kappa A_m^{t-1}, \quad (19)$$

$$r_m^t = r_m^0[1 - \mathrm{e}^{-\gamma t}]. \quad (20)$$

其中:$\kappa \in (0,1)$为声波衰减系数,$\gamma > 0$为脉冲频率增强系数,A_m^{t-1}和A_m^t分别为蝙蝠b_m在第$t-1$次和第t次迭代中响度,r_m^t和r_m^0分别为蝙蝠b_m在第t次和初始的脉冲发射率.

完成种群中个体位置的更新后,进行个体在解空间的调度解序列更新. 为此,设计一种基于距离和最优解的映射算子,具体过程如下. 设当前最佳的调度解序列为Π^*,将Π^*各个任务以一定概率置于Π_m中的相同位置,概率计算方式为

$$\mathrm{prob} = \mathrm{e}^{-d_m}, \quad (21)$$

其中d_m表示b_m与当前最佳蝙蝠的距离,即$d_m = |\mathrm{pos}_m - \mathrm{pos}^*|$. 因为$d_m \geqslant 0$,所以$\mathrm{prob} \in (0,1]$且$d_m$越小prob越大. 对于未能置于相同位置的剩余任务,以随机的方式置于Π_m的空位置中. 因此,距离当前最优蝙蝠越近的蝙蝠个体,在生成调度解时,有越大的概率继承当前最优调度解中的任务排序. 根据上述步骤,即可得到蝙蝠个体的完整调度解序列,并随着迭代过程不断更新,实现对调度解空间的搜索.

3.3 调度解的评估

UATS算法中的评估算子如算法1所示. 基于贪婪思想,评估算子根据输入的调度解序列,依次将各任务分配到边缘或云的计算资源上执行,具体如下.

算法1 评估算子.

输入: 调度解序列Π_m;

输出: 个体适应度值obj_m(即优化目标值).

step 1: 初始化相关参数: $\mathrm{obj}_m = 0, x_{ij,k} = 0$.

step 2: 对于每一个任务$\pi_m^n \in \Pi_m$,其属性集为$\{\mathrm{dl}_i, \widetilde{\mathrm{et}_{ij}^{\mathrm{ES}}}, \widetilde{\mathrm{et}_{ij}^{\mathrm{C}}}, \mathrm{tt}_{ij}^{\mathrm{ES}\Rightarrow\mathrm{C}}\}$. 由式(22)计算服务器分配时长$\mathcal{D}$,然后在$t \in (0, \mathrm{dl}_i]$内为其寻找具有空闲时段$\mathcal{D}$的边缘服务器$\mathrm{es}_s$. 若能够找到,则$x_{ij,s} = 1$;否则将该任务提交到云服务器上处理,$x_{ij,S+1} = 1$.

step 3: 由式(7)~(10)计算所有众包应用的预期完成时间,若满足式(11)中的概率约束,则根据式(12)计算obj_m,否则Π_m为不可行解.

在算法1中,考虑到任务的执行时长具有随机性,在计算服务器分配时长$\mathcal{D}$时,根据任务执行时长的概率分布$\widetilde{\mathrm{et}_{ij}} \sim N(\mu_{ij}, {\sigma_{ij}}^2)$和可定义的概率约束阈值$\alpha$来确定其取值. 由于$\widetilde{\mathrm{et}_{ij}}$为非标准正态分布变量,本文推导了非标准正态分布变量的某一特定分位点的计算过程.

定理1 已知非标准正态分布随机变量服从$\widetilde{\mathrm{et}_{ij}} \sim N(\mu_{ij}, {\sigma_{ij}}^2)$,那么其$\alpha$分位点取值$\mathrm{et}_{ij}^{\alpha}$为

$$\mathrm{et}_{ij}^{\alpha} = \mathrm{norminv}(\alpha, 0, 1) \cdot \sigma_{ij} + \mu_{ij} = \phi^{-1}(\alpha) \cdot \sigma_{ij} + \mu_{ij}, \quad (22)$$

其中$\mathrm{norminv}(\alpha, 0, 1)$表示标准正态分布$N(0, 1^2)$的累积分布函数(cumulative distribution function, CDF)的逆函数,具体取值由CDF表易得. 为简化表达,用$\phi(x)$表示标准正态分布的CDF,$\phi^{-1}(\alpha)$表示标准正态分布变量的α分位点.

证明 对于标准正态分布$\widetilde{X}\sim N(0,1^2)$,其CDF为

$$\phi(x)=\int_{-\infty}^{x}\varphi(t)\mathrm{d}t=\int_{-\infty}^{x}\frac{1}{\sqrt{2\pi}}\mathrm{e}^{-\frac{t^2}{2}}\mathrm{d}t. \tag{23}$$

对于非标准正态分布变量$\widetilde{\mathrm{et}_{ij}}$,有

$$\begin{aligned}\mathrm{prob}\{\widetilde{\mathrm{et}_{ij}}<\mathrm{et}_{ij}^{\alpha}\}&=\mathrm{prob}\left\{\frac{\widetilde{\mathrm{et}_{ij}}-\mu_{ij}}{\sigma_{ij}}<\frac{\mathrm{et}_{ij}^{\alpha}-\mu_{ij}}{\sigma_{ij}}\right\}=\\&\mathrm{prob}\left\{\widetilde{X}<\frac{\mathrm{et}_{ij}^{\alpha}-\mu_{ij}}{\sigma_{ij}}\right\}=\alpha.\end{aligned} \tag{24}$$

因此$\dfrac{\mathrm{et}_{ij}^{\alpha}-\mu_{ij}}{\sigma_{ij}}=\phi^{-1}(\alpha)$成立,进而有$\mathrm{et}_{ij}^{\alpha}=\phi^{-1}(\alpha)\sigma_{ij}+\mu_{ij}$. □

基于定理1得到的随机任务执行时长的α分位点$\mathrm{et}_{ij}^{\alpha}$即为算法1中服务器分配时长$\mathcal{D}$.

4 实验结果

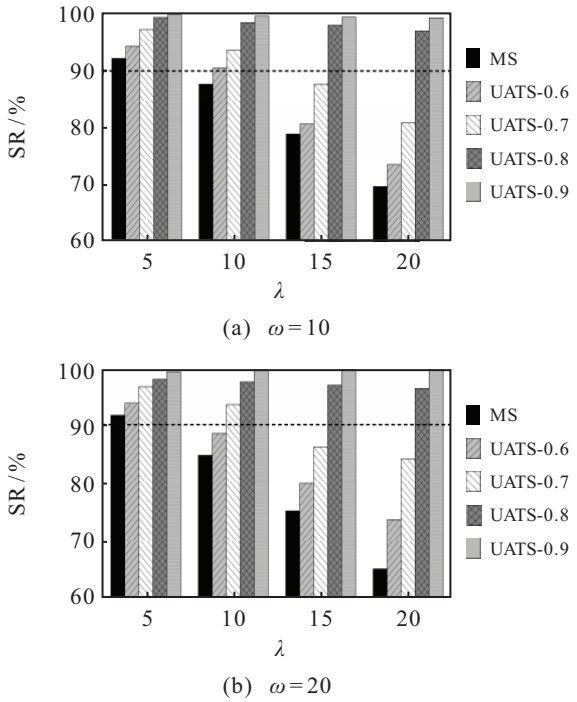
本文算法在Python3.10.5编程环境下得以实现并加以验证,实验过程在一台配置为十核3.4 GHz处理器32 GB内存的计算机上完成. 考虑众包任务执行时长的不确定性,设计一个众包应用测试实例集. 测试实例集中所有任务的执行时长建模为服从正态分布的随机变量,以min为单位,满足$E[\widetilde{\mathrm{et}_{ij}}]\in[1,30]$, $\mathrm{Var}[\widetilde{\mathrm{et}_{ij}}]\in[0.1\times E[\widetilde{\mathrm{et}_{ij}}],0.3\times E[\widetilde{\mathrm{et}_{ij}}]]$. 为了比较不同问题规模下的算法性能,众包应用的数量设置为$\omega=\{10,20\}$,每个众包应用中的任务个数设置为$\omega\times\lambda=\{10\times5,10\times10,10\times15,10\times20,20\times5,20\times10,20\times15,20\times20\}$. 每组测试任务集包含5个不同的测试实例,测试实例集中共有40个测试实例. 该系统中边缘提供商配备的边缘服务器有6台,云平台收费标准参考阿里云华北2区(北京)的云服务器按需租用计费标准,规格为计算型ecs.c5.3xlarge,12核,24 GB的价格为3.73元/h/台,其计费周期为2 min,即0.124 3元/min/台[26]. 算法种群规模设置为30,迭代次数为500. 蝙蝠个体的频率、速度等属性取值范围为$f_m\in(0,1),v_m\in(-2,2),A_m\in(0.7,1),r_m\in(0,0.4),\mathrm{pos}_m\in(0,10)$,参数$\kappa=\gamma=0.98$. 为了评估所设计的UATS算法产生的调度解质量,将任务持续时间变化影响下的两种确定性调度策略作为对比算法: 1) mean-case调度(mean-case scheduling, MS),也称均值调度,该方案以随机任务执行时长$\widetilde{\mathrm{et}_{ij}}$的均值$\mu_{ij}$生成任务调度解. 2) worst-case调度(worst-case scheduling, WS),也称最坏情况调度,该方案以随机任务执行时长$\widetilde{\mathrm{et}_{ij}}$的$\mu_{ij}+3\sigma_{ij}$生成任务调度解. 为实现公平的比较,上述两种对比算法的迭代寻优过程和参数设置均与UATS保持一致.

首先,本文考察MS算法和不同概率阈值下的UATS算法所产生的调度解成功率. 对于每个测试实例,根据任务执行时长的均值和方差,随机生成ϱ组($\varrho=10\,000$)测试样本. 算法的调度成功率(success rate, SR)定义如下:

$$\mathrm{SR}=\frac{\displaystyle\sum_{i=1}^{\varrho}\frac{\omega_s}{\omega}}{\varrho}, \tag{25}$$

其中ω_s为按时完成的众包应用数量. 只有众包应用中的所有任务都在截止日期之前完成,才认为该众包应用成功完成. 根据式(25), SR则为所有测试样本的成功率平均值.

实验结果如图2所示. 可以发现,随着概率阈值的提高,UATS算法的调度成功率逐渐提高,且均高于MS调度方案. 当众包应用中的任务数量增加时,算法的调度成功率均有一定程度的下降,这种现象在MS算法以及概率阈值为0.6和0.7的UATS算法中表现得尤为明显. 在不确定性的影响下,对于不同规模的任务量,UATS算法都能保证调度成功率不低于其设定的概率约束阈值. 由此可见,基于用户可定义的概率要求,所提出的UATS算法能够有效保证具有随机任务执行时长的众包应用按时完成.



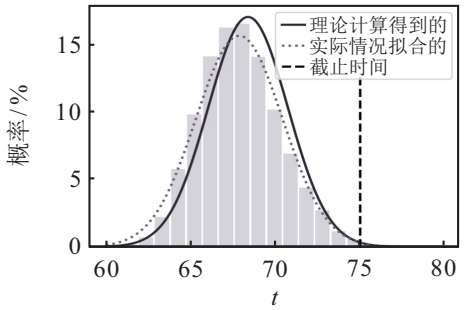
(a) $\omega=10$

(b) $\omega=20$

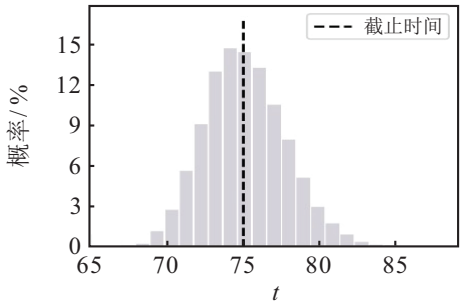
图2 MS算法和UATS算法的调度成功率

为了更形象地展示不同算法下众包应用的完成时间,以任务规模$\omega\times\lambda=20\times20$的某一个众包应用为例,随机生成20 000组测试样本,得到MS、UATS和WS三种算法下该众包应用完成时间的实际分布情况. 如图3所示,MS算法中近半数测试样本未能在

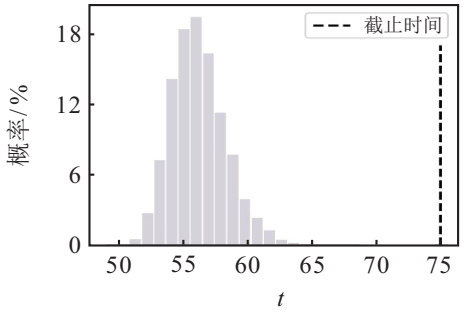
该截止时间之前完成,而UATS算法在概率阈值设置为0.8时,确保了近98%的测试样本按时完成. 图3(a)展示了理论方式计算得到的和实际得到的众包应用完成时间$\widetilde{C}_i$的概率分布曲线,可见,所提出的针对$\widetilde{C}_i$的概率约束是可行且有效的. 而对于WS算法,其所有测试样本的完成时间均在截止时间之前,这是因为该方案采取了过于保守的调度方案,默认所有任务均在只有极小可能性出现的最长执行时长完成. 但WS算法的资金开销是不可忽视的,为此,下述实验展示了不同任务规模下WS算法和UATS算法的租赁开销对比.



(a) UATS



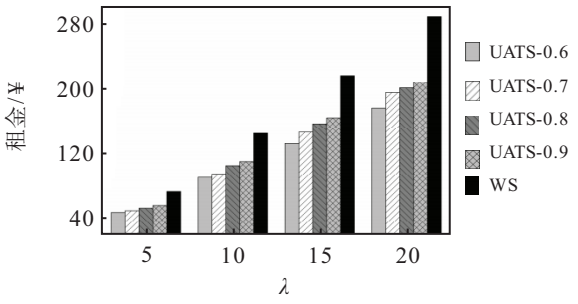
(b) MS



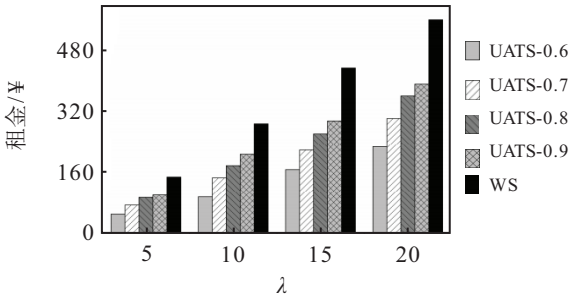
(c) WS

图 3　各算法获得的众包应用完成时间的概率分布

如图4所示,WS算法产生的租赁开销显著高于UATS算法,且随着问题规模的增加越发明显. 以$\omega \times \lambda = 20 \times 20$为例,结合图2所示的UATS算法的调度成功率,概率阈值为0.9的UATS算法实现了与WS算法无显著差别的调度成功率,但其产生的租赁开销相比WS降低了30.22%,由此可见,UATS算法具有将调度成功率和租赁开销两者兼顾的调度能力.



(a) $\omega=10$



(b) $\omega=20$

图 4　WS算法和UATS算法的边缘提供商租赁开销

为了进一步评估UATS算法与其他不确定性感知任务调度算法的性能差异,选取两个经典的种群智能优化算法遗传算法(genetic algorithm, GA)和萤火虫算法(firefly algorithm, FA)作为对照试验,旨在展示蝙蝠算法搜索策略在求解本文问题时的性能优势[27]. 以上3种算法的共有参数(种群规模和迭代次数)设置相同,非共有参数设置参考相关文献[28,11].

由于种群智能优化算法是一类具有随机性的元启发式算法,本文采用方差分析方法(analysis of variance, ANOVA)评估算法性能. 针对每一个测试实例, ANOVA需要对每种待评估算法进行多轮重复,并记录所有轮次中获得的优化目标值. 为量化各算法的求解质量,评价指标(relative percentage deviation, RPD)定义如下:

$$\mathrm{RPD} = \frac{1}{K}\left(\sum_{k=1}^{K}\frac{\mathrm{obj}_k - \mathrm{obj}^*}{\mathrm{obj}^*}\right) \times 100\%. \tag{26}$$

其中: obj_k为当前算法第k轮时获得的目标函数值, obj^*为所有算法所有轮次中最佳的目标函数值. 由此可见, RPD值越小则当前算法所获得的解的质量越好. 实验中将K的值设置为5,图5显示了在测试实例集上执行上述3种不确定性感知任务调度算法的ANOVA分析结果,包括RPD的均值和95%最小显著性差异(least-significant difference, LSD)区间. 对比结果表明,本文方法相比GA和FA方法能够获得更低的RPD值,即在调度解质量方面表现更为优秀. 此外,本文方法与GA方法的LSD区间并未重叠,表明本文方法相比于GA方法具有显著性优势.

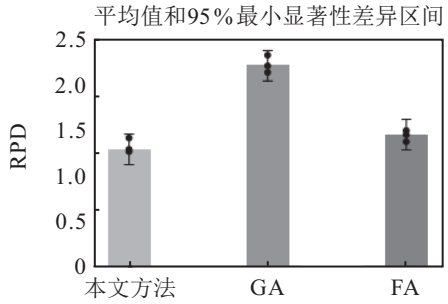


图 5 多种不确定性感知任务调度算法的求解质量对比

此外, 本文引入一个标准化效率 (normalized efficiency, NE) 的度量指标, 从定量的角度评价不同算法的计算时间效率. NE定义如下:

$$\mathrm{NE}=\frac{T}{T_{\mathrm{baseline}}}. \tag{27}$$

其中: T 表示当前算法的运行时间, T_{baseline} 表示基准算法的运行时间. 实验中选择本文方法作为基准算法, 由于每种算法都要在每个测试实例上重复 R 轮, T 和 T_{baseline} 均为所有测试实例经过 R 轮重复得到的平均值. 根据式(27), NE值越小算法计算效率越高. 表1列出了各算法在测试实例集上获得的平均运行时间和NE值, 可知本文方法的计算效率优于FA方法. 由于本文方法中蝙蝠个体的更新步骤较为复杂, 涉及频率、响度和位置等更新计算, 本文方法在计算效率上未能超越GA方法. 但另一方面, 正是由于这种复杂的寻优过程, 本文方法能够在求解质量的对比中优于其他方法. 如何在保证本文算法优秀寻优能力的基础上, 进一步提高算法的计算效率也是未来一个重要的研究方向.

表 1 多种不确定性感知任务调度算法的效率对比

	运行时间/ms	NE
本文方法	347.324	1.000
GA	279.459	0.805
FA	689.387	1.985

5 结 论

本文研究了边缘计算系统中不确定性感知的任务调度问题, 建立了具有随机任务执行时长的调度优化问题模型. 基于模型中关于任务截止时间的概率约束, 刻画了任务完成时间的完整概率分布信息, 从概率的角度保证众包应用按时完成. 为求解该问题, 提出了一种基于蝙蝠算法的任务调度算法, 用于在截止时间的概率约束和边缘提供商的计算资源约束下, 搜索租赁开销最小的调度决策. 实验结果验证了所提出算法的可行性和有效性.

参考文献(References)

[1] 赵璞, 肖人彬. 基于自组织劳动分工的边云协同任务调度与资源缓存算法[J]. 控制与决策, 2023, 38(5): 1352-1362.
(Zhao P, Xiao R B. Edge-cloud collaborative task scheduling and resource cache algorithm based on self-organizing division of labor[J]. Control and Decision, 2023, 38(5): 1352-1362.)

[2] Zakarya M, Gillam L, Ali H, et al. epcAware: A game-based, energy, performance and cost-efficient resource management technique for multi-access edge computing[J]. IEEE Transactions on Services Computing, 2022, 15(3): 1634-1648.

[3] 柴天佑, 程思宇, 李平, 等. 端边云协同的复杂工业过程运行控制智能系统[J]. 控制与决策, 2023, 38(8): 2051-2062.
(Chai T Y, Cheng S Y, Li P, et al. Intelligent system for operational control of complex industrial process based on end-edge-cloud collaboration[J]. Control and Decision, 2023, 38(8): 2051-2062.)

[4] 郑莹莹, 周俊龙, 申钰凡, 等. 时间和能量敏感的端-边-云车路协同系统资源调度优化方法[J]. 计算机研究与发展, 2023, 60(5): 1037-1052.
(Zheng Y Y, Zhou J L, Shen Y F, et al. Time and energy-sensitive end-edge-cloud resource provisioning optimization method for collaborative vehicle-road systems[J]. Journal of Computer Research and Development, 2023, 60(5): 1037-1052.)

[5] Li K. Scheduling precedence constrained tasks for mobile applications in fog computing[J]. IEEE Transactions on Services Computing, 2023, 16(3): 2153-2164.

[6] Zhang Y, Zhou J L, Sun J. Scheduling bag-of-tasks applications on hybrid clouds under due date constraints[J]. Journal of Systems Architecture, 2019, 101: 101654.

[7] Yin L, Zhou J, Sun J. A stochastic algorithm for scheduling bag-of-tasks applications on hybrid clouds under task duration variations[J]. Journal of Systems and Software, 2022, 184: 111123.

[8] Li K L, Tang X Y, Veeravalli B, et al. Scheduling precedence constrained stochastic tasks on heterogeneous cluster systems[J]. IEEE Transactions on Computers, 2015, 64(1): 191-204.

[9] Cao K, Li L Y, Cui Y G, et al. Exploring placement of heterogeneous edge servers for response time minimization in mobile edge-cloud computing[J]. IEEE Transactions on Industrial Informatics, 2021, 17(1): 494-503.

[10] Dinh T Q, Tang J, La Q D, et al. Offloading in mobile edge computing: Task allocation and computational frequency scaling[J]. IEEE Transactions on Communications, 2017, 65(8): 3571-3584.

[11] Yin L, Sun J, Zhou J, et al. ECFA: An efficient convergent firefly algorithm for solving task scheduling problems in cloud-edge computing[J]. IEEE Transactions on Services

Computing, 2023, 16(5): 3280-3293.

[12] Li S, Sun W, Sun Y, et al. Energy-efficient task offloading using dynamic voltage scaling in mobile edge computing[J]. IEEE Transactions on Network Science and Engineering, 2020, 8(1): 588-598.

[13] Mukherjee M, Kumar V, Maity D, et al. Delay-sensitive and priority-aware task offloading for edge computing-assisted healthcare services[C]. IEEE Global Communications Conference. Taipei, 2020: 1-5.

[14] 吕灵灵, 杨志鹏, 张磊. 基于合约设计的移动边缘计算任务卸载策略研究[J]. 控制与决策, 2019, 34(11): 2366-2374.
(Lyu L L, Yang Z P, Zhang L. Contract theory based task off loading strategy of mobile edge computing[J]. Control and Decision, 2019, 34(11): 2366-2374.)

[15] Li J, Shang Y, Qin M, et al. Multi objective oriented task scheduling in heterogeneous mobile edge computing networks[J]. IEEE Transactions on Vehicular Technology, 2022, 71(8): 8955-8966.

[16] 杨子轩, 张文柱, 程鹏, 等. 基于混合果蝇算法的计算卸载方法[J]. 小型微型计算机系统, 2023, 44(6): 1290-1296.
(Yang Z X, Zhang W Z, Cheng P, et al. Computation offloading decision strategy based on hybrid fruit fly optimization algorithm[J]. Journal of Chinese Computer Systems, 2023, 44(6): 1290-1296.)

[17] Meng S M, Huang W J, Yin X C, et al. Security-aware dynamic scheduling for real-time optimization in cloud-based industrial applications[J]. IEEE Transactions on Industrial Informatics, 2021, 17(6): 4219-4228.

[18] 雷雪梅, 刘丽, 王倩. 基于线性规划松弛的移动边缘计算卸载模型[J]. 计算机科学, 2023, 50(S1): 626-630.
(Lei X M, Liu L, Wang Q. MEC offloading model based on linear programming relaxation[J]. Computer Science, 2023, 50(S1): 626-630.)

[19] Lin B, Huang Y, Zhang J, et al. Cost-driven off-loading for DNN-based applications over cloud, edge, andend devices[J].IEEE Transactions on Industrial Informatics, 2019, 16(8): 5456-5466.

[20] Novak A, Sucha P, Hanzalek Z. Scheduling with uncertain processing times in mixed-criticality systems[J]. European Journal of Operational Research, 2019, 279(3): 687-703.

[21] Jia Y H, Chen W N, Yuan H, et al. An intelligent cloud workflows cheduling system with time estimation and adaptive ant colony optimization[J]. IEEE Transactions on Systems, Man, and Cybernetics: Systems, 2018, 51(1): 634-649.

[22] Tang X Y, Li K L, Liao G P, et al. A stochastic scheduling algorithm for precedence constrained tasks on Grid[J]. Future Generation Computer Systems, 2011, 27(8): 1083-1091.

[23] Yang Y Q, Lu X Q, Jin H, et al. A stochastic task scheduling algorithm based on importance-ratio of makespan to energy for heterogeneous parallel systems[C]. IEEE 17th International Conference on High Performance Computing and Communications. New York, 2015: 390-396.

[24] Varshney P, Simmhan Y. AutoBoT: Resilient and cost-effective scheduling of a bag of tasks on spot VMs[J]. IEEE Transactions on Parallel and Distributed Systems, 2019, 30(7): 1512-1527.

[25] Clark C E. The greatest of a finite set of random variables[J]. Operations Research, 1961, 9(2): 145-162.

[26] Aliyun. Alibaba cloud billing methods[EB/OL]. (2023-03-25) [2023-07-27]. https://help.aliyun.com/document_detail/40653.html.

[27] 李佳磊, 顾幸生. 双种群混合遗传算法求解具有预防性维护的分布式柔性作业车间调度问题[J]. 控制与决策, 2023, 38(2): 475-482.
(Li J L, Gu X S. Two-population hybrid genetic algorithm for distributed flexible job-shop scheduling problem with preventive maintenance[J]. Control and Decision, 2023, 38(2): 475-482.)

[28] Abbasi M, Yaghoobikia M, Rafiee M, et al. Optimal distribution of workloads in cloud-fog architecture in intelligent vehicular networks[J]. IEEE Transactions on Intelligent Transportation Systems, 2021, 22(7): 4706-4715.

作者简介

尹璐 (1996−), 女, 博士生, 从事移动边缘计算、调度优化理论等研究, E-mail: ylu@njust.edu.cn;

周俊龙 (1988−), 男, 副教授, 博士, 从事边缘计算、嵌入式系统、计算机体系结构等研究, E-mail: jlzhou@njust.edu.cn;

孙晋 (1983−), 男, 教授, 博士, 从事计算机体系结构、高性能计算等研究, E-mail: sunj@njust.edu.cn;

吴泽彬 (1981−), 男, 教授, 博士, 从事遥感图像处理、高性能计算等研究, E-mail: wuzb@njust.edu.cn.