控制与决策
Control and Decision

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

赵璞, 肖人彬

引用本文:
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.

在线阅读 View online: https://doi.org/10.13195/j.kzyjc.2022.0907

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

赵 璞[1,2], 肖人彬[1†]

(1. 华中科技大学 人工智能与自动化学院，武汉 430074；2. 中国北方车辆研究所，北京 100072)

摘 要: 针对边缘计算环境中,边缘设备的计算和存储资源有限的问题,探讨高效的边云协同任务调度和资源缓存策略,研究自组织劳动分工群智能算法模型机理,并以此为基础,提出基于蜂群劳动分工“激发-抑制”模型的边云协同任务调度算法(edge cloud collaborative task scheduling algorithm based on bee colony labor division ‘activator-inhibitor’ model, ECCTS-BCLDAI)和基于蚁群劳动分工“刺激-响应”模型的边云协同资源缓存算法(edge cloud collaborative resource caching algorithm based on ant colony labor division ‘stimulus-response’ model, ECCRC-ACLDSR). 仿真实验结果表明:所提出的ECCTS-BCLDAI任务调度算法在降低平均任务执行时长、减少边云协同费用上相较于传统算法有更好的表现;所提出的ECCRC-ACLDSR资源缓存算法在降低任务平均时长、优化网络带宽占用率、减少边云协同费用上相较于传统算法更具有优越性.

关键词: 边缘计算；自组织劳动分工；任务调度；资源缓存

中图分类号: TP18 **文献标志码:** A

DOI: 10.13195/j.kzyjc.2022.0907

引用格式: 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.

Edge-cloud collaborative task scheduling and resource cache algorithm based on self-organizing division of labor

ZHAO Pu[1,2], *XIAO Ren-bin*[1†]

(1. School of Artificial Intelligence and Automation，Huazhong University of Science and Technology，Wuhan 430074，China；2. China North Vehicle Research Institute，Beijing 100072，China)

Abstract: Aiming at the problem of limited computing and storage resources of edge devices in the edge computing environment, we discuss efficient edge-cloud collaborative task scheduling and resource caching strategies, and study the mechanism of the self-organizing labor division swarm intelligent algorithm model. On this account, the edge cloud collaborative task scheduling algorithm based on bee colony labor division ‘activator-inhibitor’ model (ECCTS-BCLDAI) and the edge cloud collaborative resource caching algorithm based on ant colony labor division ‘stimulus-response’ model (ECCRC-ACLDSR) are proposed. The simulation results show that the proposed task scheduling and resource caching algorithm have better performance than the traditional algorithm in reducing the average task execution time, optimizing network bandwidth usage and reducing the edge-cloud collaboration cost.

Keywords: edge computin；self-organizing labor division；task scheduling；resource caching

0 引 言

在传统云计算架构中,计算和存储均在云计算中心进行集中处理,在面对高流量、低延时的用户需求时,长距网络的延迟问题日益凸显,特别是在实时交互等场景[1]. 随着5G、物联网、无人驾驶[2-4]等技术的进一步推广应用,互联网流量愈加呈现爆炸增长态势,骨干网络不断承受流量增加带来的冲击, 因此边缘计算逐渐兴起. 边缘计算是一种新的计算模型,它将空间距离或网络距离与用户临近的边缘设备进行组织与管理,在近端提供计算和存储能力[5]. 其概念最早可以追溯至1998年阿卡迈(Akamai)公司提出的内容分发网络(content delivery network, CDN),但CDN更强调内容数据的下发[6-7],偏向于存储的边缘化,而边缘计算则兼顾计算和存储能力的边缘化. 因此,研究如何将计算任务在云端和边缘端合理调度,以及如何将资源从云端向边缘端缓存,从而形成一套

收稿日期: 2022-05-22；**录用日期:** 2022-11-10.
基金项目: 科技创新2030——“新一代人工智能”重大项目(2018AAA0101200).
责任编委: 杨涛.
†通讯作者. E-mail: rbxiao@hust.edu.cn.

整体的边云协同策略,成为当下的研究热点[8-12].

鉴于边缘计算网的复杂特性,人工智能算法也被用于边缘计算以实现资源的智能管理[11],其中群智能算法最具代表性. 群智能的概念源于对蚂蚁、蜜蜂等社会性昆虫群体行为的研究,简单的个体通过相互作用可表现出复杂的智能行为[13],是一种有别于连接主义和符号主义的新的研究人工智能的途径,且成为研究热点[14]. 目前,对群智能的研究除了模拟昆虫群体觅食行为的优化算法,对群智能自组织劳动分工的研究也逐渐兴起,并取得了一系列的成果[14-19]. 王英聪等[15]将蚁群劳动分工任务分配应用到卫星舱布局空间分配上; Kim 等[19]将基于蚁群劳动分工响应阈值模型的概率决策机制应用到无人机攻击任务分配上. 肖人彬等[14]从复杂系统和复杂性科学研究的角度,将群定义为通过改变局部环境直接或间接交互的一组个体,这组个体可通过合作分工解决分布式问题. 群体智能是简单个体通过交互,表现出难以预测的宏观智能行为特性,并以此提出了群智能自组织劳动分工的4类模型:激发-抑制、刺激-响应、个体排序和寻觅工作模型.

关于边缘计算的任务调度和资源缓存算法方面,国内外学者已经做了大量的研究,付主木等[20]应用李雅普诺夫随机优化提出一种低复杂度的车辆边缘计算联合任务卸载与资源分配算法; Li 等[21]综合数据块的最佳放置策略和任务调度算法,实现对边缘设备存储的高效利用,减少任务响应时间;还有学者通过引入马尔科夫决策过程、Q学习算法、深度学习算法等,提出任务调度和资源缓存算法[22-25]. 但是,这些研究多考虑将任务尽可能地迁移到边缘服务器上,忽略了边缘服务器相对较弱的性能与较小的存储容量,以及较高的维护费用.

自组织劳动分工模型以其角色可塑性、族群高效性和自组织性,给研究如何实现云端和边缘端的任务调度和资源缓存提供了新的研究思路. 本文结合边缘计算的特点,设计出基于劳动分工的边云协同任务调度和资源缓存算法,并验证两种算法在减少平均任务执行时长, 提高本地缓存资源命中率,降低网络带宽占用率,从而降低整体设备租赁费用方面的作用,对边缘计算的推广应用具有较大研究意义.

1　边云协同任务调度和资源缓存网络模型

图1展示了本文边缘计算的网络示意图,其核心思想是通过云端和边缘端之间的动态任务调度和资源缓存来实现云-边的动态协同.

若干基本变量定义如下:

$V=\{V_1,V_2,\ldots,V_n\}$: 边缘服务器节点;

$U=\{U_1,U_2,\ldots,U_m\}$: 用户终端节点;

S: 云计算中心节点;

CE: 节点的计算单元;

SE: 节点的存储单元;

$F=\{F_1,F_2,\ldots,F_p\}$: 资源文件列表;

$W_i(i=1,2,\ldots,p)$: 资源文件F的大小;

$\text{Job}=\{\text{Job}_1,\text{Job}_2,\ldots,\text{Job}_q\}$: 任务集合,任务由用户终端发起;

$\text{Job}_i(i=1,2,\ldots,q)=\{\text{IF}|\text{rF}_1,\ldots,\text{rF}_k\}$: 执行任务$i$的需要资源集合,IF为该任务用户终端的输入,rF为执行该任务需要的其他资源文件.

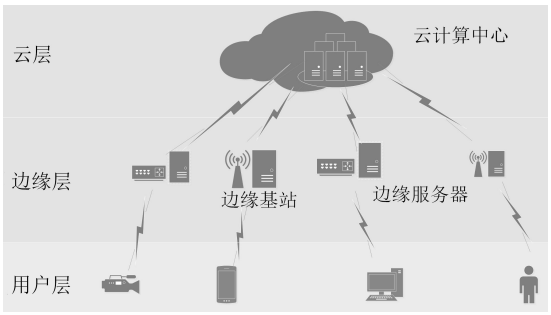

图 1　边缘计算的网络示意图

1.1　任务调度和资源缓存过程

下面,介绍在边缘计算网络模型中进行动态任务调度和资源缓存的过程,具体步骤如下:

step 1: 终端用户U_1向最近的边缘节点V发送任务请求. 边缘节点V根据调度算法决定任务在本地执行(转到step 2)或者转到云中心节点S执行(转到step 5).

step 2: 边缘节点V将任务加入到本地任务列表,若列表为空,则执行任务;若不为空,则排队等待.

step 3: 边缘节点V开始执行任务,首先接收用户的输入,查看任务所需要的资源列表,并与本地的资源缓存列表对比. 若任务所需要的资源在本地缓存中都能找到,则开始计算(转到step 8); 否则,转到step 4.

step 4: 边缘节点V选择恰当的资源所在节点,远程请求资源,根据资源缓存策略决定是否对资源进行缓存,判断任务所需资源是否齐备,若齐备,则开始计算(转到step 8); 否则,重复本步骤.

step 5: 云中心节点S将任务加入到其任务列表,若列表为空,则执行任务;若不为空,则排队等待.

step 6: 云中心节点S首先接收用户输入. 将任务所需资源与云中心资源缓存列表对比. 若所需资源已齐备,则开始计算(转到step 8); 否则,转到step 7.

step 7: 云中心节点S选择恰当的节点远程请求资源,鉴于云计算中心海量弹性资源提供能力,一般对所有任务所需要的程序、数据库等资源都缓存. 若所需资源已齐备,则开始计算;否则,重复本步骤.

step 8: 执行计算,结束后计算任务时间和费用等.

1.2 优化模型建立

下面对边云协同任务调度和资源缓存性能评价指标进行建模,包括边云协同费用模型、任务历史平均延时、网络带宽占用率等.

定义1 (边云协同费用模型) 由云计算中心虚拟服务器、边缘服务器、宽带网络等设备或资源的租赁费用总和构建起的模型,称为边云协同费用模型,其目标函数为

$$\min\left(\sum_{i=1}^{q} C_{\text{Job}_i}+\sum_{j=1}^{p} C_{F_j}\right). \tag{1}$$

C_{Job}表示任务在节点V执行的费用,为

$$C_{\text{Job}}=G_V\times T_{\text{excute}}+\sum_{j=1}^{k} G_{\text{flow}}\times W_{\text{rF}_j}. \tag{2}$$

其中:G_V表示租用节点V上计算资源执行任务每小时所需要的费用,G_{flow}表示边云协同网络上每GB流量所收取的费用,rF_j表示需从其他节点上获取的资源,W_{rF_j}表示资源大小.

C_{F_j}表示资源F_j在节点V的缓存费用,为

$$C_{F_j}=S_V\times W_{F_j}\times \text{storageTime}_{F_j}. \tag{3}$$

其中:S_V表示租用节点V上存储每GB数据每小时所需费用,storageTime_{F_j}表示资源F_j占用的存储时间.

定义2 (任务历史平均延时$\overline{T}$) 各任务排队延时、任务资源获取时间、任务执行时间总和的平均值. 任务历史平均延时是评价优化算法优劣的最重要标准.

$$\overline{T}=\sum_{i=1}^{n} T_i\Big/n. \tag{4}$$

其中:n为运行作业总数,T_i为第i个作业运行时间. T_i定义如下:

$$T_i=T_{\text{queue}}+T_{\text{access}}+T_{\text{excute}}. \tag{5}$$

其中

$$T_{\text{queue}}=\sum_{T_j \text{ in the queue}} T_j, \tag{6}$$

$$T_{\text{access}}=\sum_{F_i \text{ in the remoteSite}} \frac{\text{fileSize}(F_i)}{\text{BW}_{NV}(t)}, \tag{7}$$

$$T_{\text{excute}}=\sum_{F\in \text{ files in the job}} \frac{\text{fileSize}(F)}{\text{WorkerNode}_V}. \tag{8}$$

T_{queue}指任务排队时间,T_{access}指任务资源准备时间,T_{excute}指任务计算时间,$\text{BW}_{NV}(t)$指节点间网络带宽,WorkerNode_V指节点计算能力.

远程访问资源文件需花费时间且占用网络带宽,缓存创建和任务调度应以减少网络流量为基本出发点,因此定义网络带宽占用率如下.

定义3 网络带宽占用率R_{ENU}为

$$R_{\text{ENU}}=\sum \frac{N_{\text{remote accesses}}+N_{\text{file caches}}}{N_{\text{remote accesses}}+N_{\text{local accesses}}}. \tag{9}$$

其中:$N_{\text{remote accesses}}$指从非本地节点访问资源文件次数,$N_{\text{file caches}}$指文件执行本地缓存次数,$N_{\text{local accesses}}$指从本地节点中访问资源缓存次数. 对于一个给定的网络拓扑结构,R_{ENU}越小,说明本地缓存命中率越高,对网络带宽的占用越少.

2 基于蜂群劳动分工“激发-抑制”模型的边云协同任务调度算法

2.1 “激发-抑制”模型

“激发-抑制”模型原理可简要描述如下:激发剂和抑制剂一起决定蜜蜂从年幼向年长的行为发育,且它们之间具有耦合关系,即幼蜂激发剂和抑制剂的含量均少于老蜂. 保幼激素是一种典型的激发剂,它促进幼蜂从巢穴内哺育工作向巢穴外觅食工作发育[13]. 老蜂下颚分泌物是一种典型抑制剂,通过接触传播以阻碍保幼激素发挥作用,从而抑制蜜蜂的行为发育. 当从事觅食的老蜂减少时,抑制剂减弱,幼蜂加速发展为觅食者;当从事觅食的老蜂增多时,抑制作用变强,幼蜂发育会延迟,甚至一些觅食者会返回巢内从事哺育工作.

基于上述模型,Naug[26]和Gadagkar[27]建立了一个计算仿真模型. 群体中的每个个体都含有一种激发剂J和两种抑制剂IR、ER. 其中:激发剂J和抑制剂IR由个体自身产生,激发剂J能够促进自身行为发育,抑制剂IR却不会抑制自身发育,其作用是通过交互传播抑制其他个体行为发育;抑制剂ER不由个体自身产生,而是在交互中从其他个体接收,它会阻碍自身行为发育. 最终,激发剂J和抑制剂IR、ER三者相对水平$J/(\alpha\text{IR}+\text{ER})$决定个体的行为发育是正常速度还是被加速、延迟或逆转.

2.2 问题分析

边缘计算可将用户请求从云端分配到离用户更近的边缘服务器上,但如果面临突发大量请求,且边缘服务器性能不足以满足所有任务快速执行的需求时,简单将任务向边缘分发反而会使任务排队延时变长,甚至造成边缘服务器负载过重而瘫痪,云计算中

心大量计算资源则因任务极不饱和而造成浪费. 同时,由于云计算中心有着更完善的资源整合策略,更合适的地理气候选址,以及更低的电力供应成本,收费相比分布散、维护难的边缘设备往往要低[28]. 因此,边云协同的调度策略不能是一味地将任务从云端迁移向边缘端,而是研究如何使任务更合理地在云端和边缘端调度,在减少任务访问延时、减轻骨干网络压力的同时,减少设备、服务的租赁费用.

蜂群劳动分工“激发-抑制”模型和边云协同的调度算法有着天然的拟合性,蜜蜂从巢内蜂向觅食蜂的发育过程和任务从云端向边缘端迁移的过程有着很大的行为一致性,且蜂群劳动分工“激发-抑制”模型已经在很多任务调度算法中成功应用[14],因此本文在深入研究“激发-抑制”模型原理的基础上,提出基于蜂群劳动分工“激发-抑制”模型的边云协同任务调度算法(edge cloud collaborative task scheduling algorithm based on bee colony labor division ‘activator-inhibitor’ model, ECCTS-BCLDAI).

2.3　算法描述

ECCTS-BCLDAI算法通过激发剂、外部抑制剂、内部抑制剂的合理设置,以及激发抑制的相互作用,可有效地将任务在云端和边缘端合理调度,在减少任务访问延时,减轻骨干网络压力的同时,合理平衡负载,减少设备、服务的租赁费用.

ECCTS-BCLDAI算法描述为: 将某项任务看作某只蜜蜂,任务从云计算中心向边缘服务器迁移看作蜜蜂从巢内蜂向觅食蜂分化现像,边缘服务器负载看作抑制剂,任务历史平均延时看作激发剂. 某项任务的历史平均延时越长,则其激发剂越大,在激发抑制原理作用下,任务越趋向于从云计算中心向离用户更近的边缘服务器迁移(巢内蜂向觅食蜂分化),此时边缘服务器负载增大,相应的外部抑制剂增大,在激发抑制原理作用下,任务会适当地往云计算中心迁移(觅食蜂向巢内蜂逆分化),通过激发剂和抑制剂的变化来自适应调整任务在云计算中心和边缘服务器间的动态分配. 模型中的变量与边云协同问题实际变量之间的映射关系如表1所示.

表 1　“激发-抑制”模型变量与实际变量之间的映射关系

模型变量	实际变量
蜜蜂个体	某类任务
巢内/巢外	云中心/边缘服务器
生理发育	任务分配从云中心节点调度到边缘服务器节点
激发剂J	任务历史平均延时
外部抑制剂ER	边缘服务器节点负载
内部抑制剂IR	任务执行的费用

激发剂J(某类任务历史平均延时)的定义已经在式(4)中给出,具体到某类任务job为

$$J=\overline{T_{\text{job}}}=\sum_{i=1}^{n}T_{\text{job}_i}\Big/n. \tag{10}$$

其中: n为Job执行次数, T_{job_i}为Job第i次执行时间. 任务历史平均延时映射为激发剂,时间越长,越能激发任务从云端向边缘端迁移,利用边缘服务器离用户更近的优势,缩短任务执行延时.

外部抑制剂ER,即边缘点负载定义如下:

$$\text{ER}=R_V=\frac{\text{Num}_{\text{queue}}+\text{Num}_{\text{excute}}}{\text{Max}_{\text{queue}}-\text{Num}_{\text{queue}}}. \tag{11}$$

其中: $\text{Num}_{\text{queue}}$表示边缘服务器正在排队待执行的任务数量; $\text{Num}_{\text{excute}}$表示边缘服务器正在执行的任务数量; $\text{Max}_{\text{queue}}$表示边缘服务器缓存任务队列的最大值; R_V体现出边缘服务器当前负载情况,是任务从云端向边缘端迁移的外部抑制剂. 当排队任务数和执行任务数均为0时,服务器空闲,负载为0,抑制作用为零,任务尽可能分配到边缘服务器上执行; 当排队任务数等于缓存任务队列的最大值时,服务器满负载,负载值为无穷大,抑制作用为正无穷,任务必须分配到云端执行. 对于云端负载,一般云计算中心有着较大弹性计算能力,故其负载可以视为0.

内部抑制剂IR,即任务执行的费用C_{Job},其本身大小不会对当前任务的调度产生影响,但和其他任务一起产生的任务历史平均执行费用则会对新任务向边缘端的分配进行抑制,因为边缘存储设备比云中心费用更高,甚至会促进任务从边缘端逆向迁移回云端,这也与蜂群激发抑制模型中觅食蜂向巢内蜂逆向发育现象契合. 当前任务受到其他任务内部抑制剂的抑制作用表示为

$$\overline{\text{IR}}=\sum_{1}^{n}\text{IR}\Big/n. \tag{12}$$

最终,激发剂J和抑制剂的相对水平表示为激发抑制比H,如下所示:

$$H=\frac{J}{\alpha\overline{\text{IR}}+\beta\text{ER}}. \tag{13}$$

H将决定蜜蜂的行为发育是按照正常速度还是被加速、延迟或逆转,对应于任务按照正常速度从云端迁移到边缘端,还是被加速、延迟或逆转回云端,如下所示:

$$\text{Job}_{\text{调度策略}}=\begin{cases}\text{边缘端执行},\ H>H^{\text{high}};\\ \text{云端执行},\ H<H^{\text{low}};\\ \text{不变},\ \text{otherwise}.\end{cases} \tag{14}$$

基于群智能劳动分工算法,云中心节点和边缘服务器节点之间的任务分配可以实现动态调节,达到自适应地减小任务的平均延时、费用的目的,具有原理简明、易于实现的特点.

2.4 算法步骤

ECCTS-BCLDAI任务调度算法步骤如下.

step 1: 任务初次请求时,初始化激发剂、抑制剂.

step 2: 调度算法收到任务请求,开启调度策略.

step 3: 调度算法将激发剂映射为任务历史平均延时J,外部抑制剂映射为边缘服务器节点负载ER,内部抑制剂映射为任务执行的费用IR.

step 4: 根据式(10)$\sim$(13)计算激发抑制比H,如果H $>$上限阈值,则调度策略变更为将任务调度到边缘服务器上执行;如果H $<$下限阈值,则调度策略变更为将任务调度到云计算中心上执行;否则保持调度策略不变.

step 5: 判断所有任务是否执行完成,如果是,则转到step 6;否则,转至step 2继续执行任务调度.

step 6: 结束,输出边云协同费用、任务历史平均延时、网络带宽占用率等最终指标统计结果.

3 基于蚁群劳动分工“刺激-响应”模型的资源缓存算法

3.1 “刺激-响应”模型

“刺激-响应”模型由Bonabeau等[13]提出,群体中每个个体都有一个其特有的响应阈值,而外界每个任务都有一个刺激强度,当任务刺激强度大于个体响应阈值时,该个体有较大可能执行该任务. 响应阈值θ的意义是决定个体对某项任务的刺激s发生响应,从而执行该任务的可能性. 假设有i个个体可执行同一任务,令S_i为个体i的状态($S_i = 0$对应于待命状态,$S_i = 1$对应于执行状态),θ_i为第i个个体的响应阈值,则一个处于待命状态的个体按照如下概率开始执行任务:

$$P(S_i = 0 \to S_i = 1) = \frac{s^n}{s^n + \theta_i^n}. \tag{15}$$

一个正在执行任务的个体按照固定概率p放弃该任务的执行,成为待命状态. 以上两种个体状态变化均指在单位时间发生的个体状态跃变,刺激强度的变化取决于以下两个方面:任务近期是否被执行(执行后该任务需求得到缓解,刺激强度降低);任务需求随时间的自主增加(任务长时间未执行,其需求会更迫切). 某项任务的刺激强度演化公式为

$$s(t+1) = s(t) + \delta - \partial n_{\text{act}}. \tag{16}$$

其中:δ为某项任务的刺激强度在单位时间内的增量,通常取常数;n_{act}为执行该任务个体的数量;∂为调节因执行而导致该任务刺激强度降低的折扣因子. 这里δ和∂n_{act}的联合是一种负反馈作用,即某任务如果正在被多个个体执行,则该任务需要新个体执行的需求被缓解,刺激强度就会降低,反之亦然,从而使任务的执行情况达到一种均衡态.

一些学者将学习和遗忘因素引入模型,当任务被执行时,与之对应的响应阈值在学习作用下降低. 当任务未被执行时,与之对应的响应阈值在遗忘作用下增加. 令ξ和φ分别表示学习因子和遗忘因子,若在Δt内,个体i执行任务j,响应阈值θ_{ij}按下式进行更新:

$$\theta_{ij} = \theta_{ij} - \xi\Delta t; \tag{17}$$

否则,响应阈值θ_{ij}按下式进行更新:

$$\theta_{ij} = \theta_{ij} + \varphi\Delta t. \tag{18}$$

3.2 问题分析

传统的资源缓存算法有LRU、LFU、ECO等,它们多是贪婪算法的改进,即每次都执行缓存操作,当空间不足时,执行缓存替换操作,缺点是替换策略的不合理可能导致“缓存抖动”现象,另外长期占满的存储单元也会产生不必要的存储费用.

边云协同资源缓存的核心思想是综合考虑资源访问延时和边缘服务器剩余存储容量,将资源在云计算中心和边缘服务器之间尽可能合理地分配缓存,减少任务执行时远程获取资源的次数,增大本地资源命中概率,获得较好的费用优化. 蚁群劳动分工“刺激-响应”模型个体行为柔性好,适用于描述形态行为多型劳动分工[14],边缘服务器对不同资源的缓存决策行为,符合形态行为多型劳动分工的特点,因此“刺激-响应”模型非常适合应用在边云协同资源缓存算法上. 本文深入分析蚁群劳动分工“刺激-响应”模型,在此基础上提出基于蚁群劳动分工“刺激-响应”模型的边云协同资源缓存算法(edge cloud collaborative resource caching algorithm based on Ant colony labor division ‘stimulus-response’ model, ECCRC-ACLDSR).

3.3 算法描述

ECCRC-ACLDSR算法扩展了“刺激-响应”机制中“个体-环境”交互的表现形式,加入影响个体任务的记忆因素,使缓存算法更具自主性和可靠性.

ECCRC-ACLDSR算法的基本思路为:将某个边

缘服务器看作某只蚂蚁,某个资源缓存操作视为蚂蚁执行某项任务,边缘服务器执行不同资源的缓存操作视为劳动分工现象,资源缓存响应阈值增减视为蚂蚁记忆和遗忘,对资源的访问视为外界刺激.

根据边缘服务器执行任务情况,对该资源缓存进行一定奖励或惩罚,并对该缓存的记忆进行更新:若包含该缓存访问的任务执行,则针对该资源执行缓存的响应阈值降低,降低值可以用学习因子定义;若不包含该缓存访问的任务执行,则针对该资源执行缓存的响应阈值升高,升高值可以用遗忘因子定义. 随着年龄(任务执行次数)增加,其经验(是否缓存的决策次数)也会累加,这也与自然界中年长蚂蚁比年轻蚂蚁抗扰动能力更强的现象相符. 根据该思路设计出基于蚁群劳动分工“刺激-响应”模型的边云协同资源缓存算法(ECCRC-ACLDSR). 模型中变量与实际问题变量之间的映射关系如表2所示.

表 2　“刺激-响应”模型变量与实际变量间的映射关系

模型变量	实际变量
蚂蚁个体	边缘服务器
某项任务	某个资源缓存操作
劳动分工	边缘服务器执行不同资源缓存操作
蚂蚁记忆和遗忘	资源缓存响应阈值增减
外界刺激	对资源的访问

资源F从源服务器N传输到边缘服务器V的传输时间$\omega_{NV}^{F}(t)$的预测如下:

$$\omega_{NV}^{F}(t)=\frac{F_{\text{filesize}}}{\text{BW}_{NV}(t)}. \tag{19}$$

其中:F_{filesize}指资源F的大小,$\text{BW}_{NV}(t)$指源服务器N传输到边缘服务器V之间的实时网络带宽.

边缘存储节点V剩余存储容量百分比为

$$L_V=\begin{cases}\dfrac{\text{SE}_{\text{left}}^{V}-F_{\text{filesize}}}{\text{SE}_{\text{full}}^{V}},\ \text{SE}_{\text{left}}^{V}-F_{\text{filesize}}>0;\\ 0,\ \text{SE}_{\text{left}}^{V}-F_{\text{filesize}}<0.\end{cases} \tag{20}$$

其中:$\text{SE}_{\text{left}}^{V}$指边缘存储节点$V$的剩余存储容量,$\text{SE}_{\text{full}}^{V}$指边缘存储节点$V$的全部存储容量.

初始化边缘节点V缓存资源F的响应阈值为

$$\theta_V^F(0)=\frac{1}{L(0)}\times\frac{1}{\omega_{NV}^{F}(0)}. \tag{21}$$

由式(21)可以看出,边缘存储节点V缓存资源F的响应阈值与边缘存储节点V的剩余容量成反比,与资源F从源服务器N传输到边缘服务器V的实时访问代价成正比,即剩余容量越大,资源获取越困难时,执行缓存操作价值越大,对应响应阈值越小. 当剩余存储容量百分比为1时,边缘节点为空,此时关注获取资源的时间,时间越长,阈值越小,即越值得缓存; 当剩余百分比为0,边缘节点存储被占满时,响应阈值为无穷大,即不执行缓存操作.

随着访问次数的增加,可以加入“蚂蚁”(边缘存储节点V)的记忆机制. 当资源F被边缘节点V名下的用户请求时,边缘存储节点V对资源F缓存的响应阈值在学习作用下减小,即缓存的概率增大,是一个正反馈;当用户请求的是非资源F的其他资源时,边缘存储节点V对资源F缓存的响应阈值在遗忘作用下增大,即保持对资源F缓存的概率减小,执行清除该资源的概率增大,是一个负反馈. 令ξ和Φ分别表示学习因子和遗忘因子,对资源F进行请求时,响应阈值θ按下式进行更新:

$$\theta_V^F(t)=\theta_V^F-\frac{1}{L(t)}\times\frac{1}{\omega_{NV}^{F}(t)}\times\xi; \tag{22}$$

若任务不包含对资源F进行请求时,响应阈值θ按下式进行更新:

$$\theta_V^F(t)=\theta_V^F+\Phi. \tag{23}$$

可以看出,当资源F被请求时,响应阈值θ按资源缓存的价值进行减小,当资源F总不被请求时,对资源F进行缓存的响应阈值θ按固定值逐渐增加.

用户访问可以看作是外界环境的刺激动作,而刺激值的大小也与边缘存储节点V的剩余容量和资源F从源服务器N传输到边缘服务器V的实时访问代价有关,将用户访问时边缘存储节点V缓存资源F的刺激值定义为

$$S_V^F(t)=L(t)\times\omega_{NV}(\text{t})\times\lambda, \tag{24}$$

其中λ为刺激值的调节参数. 与此同时,当边缘存储节点V名下的用户访问某一资源时,边缘存储节点V对该资源以及本地所有已有缓存资源计算缓存概率来决定对这些资源进行缓存或不进行缓存.

边缘存储节点V缓存资源F的概率为

$$P(t)=\frac{(S_V^F(t))^n}{(S_V^F(t))^n+(\theta(t))^n},\ n>1. \tag{25}$$

由式(25)可以看出刺激值和响应阈值的互相影响:当剩余存储容量百分比为0时,边缘节点存储空间被缓存占满,响应阈值为无穷大,刺激值为0,$P(t)=0$,即不执行缓存操作;当网络延时无穷大时,表明若有机会,该资源非常值得缓存,此时刺激值无穷大,$P(t)=1$,即如果可能,该资源一定被缓存.

3.4　算法步骤

ECCRC-ACLDSR资源缓存算法步骤如下.

step 1: 每个边缘节点V根据式(21)对每个资源初始化响应阈值.

step 2: 当边缘节点V收到任务请求时,对任务中含有的所有资源文件的请求,根据式(22)计算响应阈值$\theta_V^F(t)$;对本地有缓存,但任务中不含有对其请求的资源时,根据式(23)计算响应阈值$\theta_V^F(t)$.

step 3: 对任务中每个请求的资源和本地缓存的每个资源,根据式(25)计算边缘存储节点V缓存资源F的概率$P(t)$,根据$P(t)$决定缓存策略. 若本地没有该资源缓存,则在任务结束后根据概率决定对该资源执行缓存操作或不执行缓存操作;若本地有该资源的缓存,则在任务结束后根据概率决定继续保有该资源缓存或清除该资源缓存.

step 4: 判断所有任务是否执行完成,是则转到step 5,否则转至step 2继续执行任务调度.

step 5: 输出本地缓存命中率、边云协同费用、任务历史平均延时网络带宽占用率等结果.

4 模拟实验及结果分析

本文在OptorSim模拟器[29]上进行模拟实验,OptorSim模拟器是欧洲粒子物理研究中心为分布式计算而开发的模拟真实网络结构的模拟器. 本文基于“CMS全球数据生产挑战赛”真实网络背景构建网络环境,网络拓扑图如图2所示,共由28个节点组成,其中云计算中心节点1个,连接着5个边缘服务器节点,每个边缘服务器节点连接着4至5个用户节点. 根据典型5G网络环境中目标延迟,设置用户和边缘服务器节点之间网络带宽为1 000 M/s,边缘服务器节点和云计算中心之间网络带宽为500 M/s.

设每个边缘服务器拥有1 000个计算单元,每个计算单元每秒可处理8 M数据,500 G存储容量;云计算中心有海量计算、存储资源和弹性计算能力,但用户一般也是按需租赁,这里假设租赁的云计算中心虚拟服务器性能为边缘服务器性能的5倍,即拥有5 000个计算单元,每个计算单元每秒可处理8 M数据,存储容量为40 G～32 768 G. 根据具有代表性的云服务提供商阿里云的收费规则,设边缘服务器和云计算中心节点间的流量为0.8元/G,一般云计算中心虚拟服务器的租赁费用为2元/小时,存储租赁价格为0.001 05元/(1 G/小时),边缘服务器的租赁费用为3.073元/小时,存储租赁价格为0.004 20元/(1 G/小时).

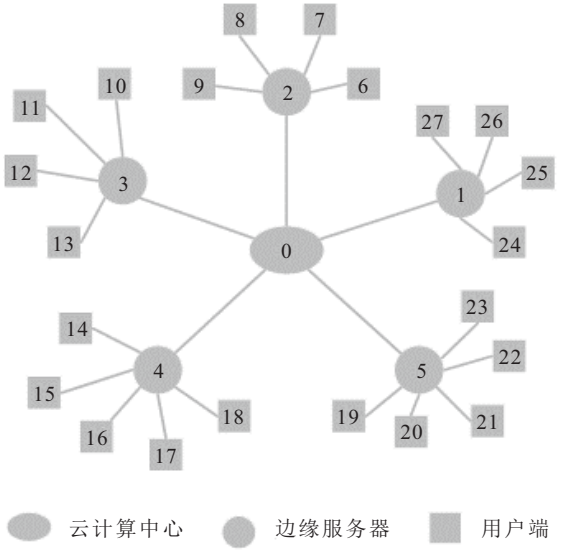

图2 实验网络拓扑结构

以无人驾驶汽车场景为例,车身配置有大量传感器,每秒钟都能生成海量数据[30-31],据此定义5类输入文件,每类大小为400 G;定义44类资源文件,每类资源文件大小为80 G;定义5类任务,每类任务包含一个用户端的输入文件和若干任务需要顺序读取的资源文件,每类任务按照一定比例随机发起,发起间隔遵照“CMS全球数据生产挑战赛”中以下午3点为中心的高斯分布的访问背景. 任务总数量取1 000,具体任务参数见表3.

表3 任务参数表

任务类型	概率/%	输入	任务需要的其他资源
jpsi	50	jpsi0	jpsi1, jpsi2, jpsi3, jpsi4, jpsi5, jpsi6, jpsi7, jpsi8, jpsi9 jpsi10, jpsi11
highptlep	20	highptlep0	highptlep1, incmuon10, jpsi10, jpsi11
incelec	13	incelec0	incelec1, incelec2, incelec3, incelec4, incelec5, incelec6, incelec7
incmuon	10	incmuon0	incmuon1, incmuon2, incmuon3, incmuon4, incmuon5, incmuon6, incmuon7, incmuon8, incmuon9, incmuon10, incmuon11, incmuon12, incmuon13
highptphot	7	highptphot0	highptphot1, highptphot2, highptphot3, highptphot4, highptphot5, highptphot6, highptphot7, highptphot8, highptphot9, highptphot10, highptphot11, highptphot12

4.1 ECCRC-ACLDSR资源缓存算法实验分析

首先验证ECCRC-ACLDSR资源缓存算法的性能. 将任务调度算法设为Random随机分配,参数设置为$\xi = 1\,000, \lambda = 500, n = 2$,进行1 000个任务的模拟仿真,并将ECCRC-ACLDSR资源缓存算法实验结果与文献[32]中提出的Hot_ECO算法,以及No-

cache、LRU、LFU等算法比较,任务平均延时如图3所示,网络带宽占用率变化如图4所示,各边缘服务器存储使用率如图5所示,边云协同费用变化如表4所示.

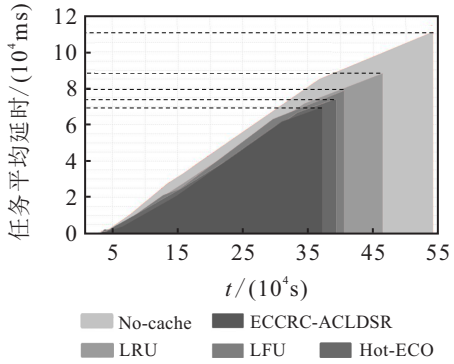

图 3　资源缓存算法任务平均延时

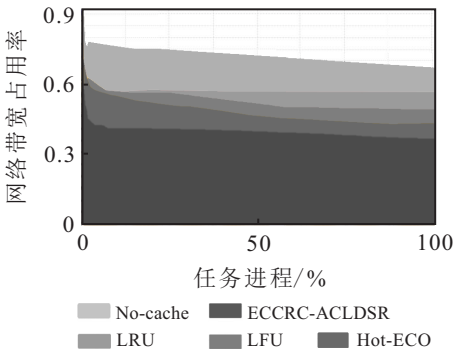

图 4　资源缓存算法网络带宽占用率

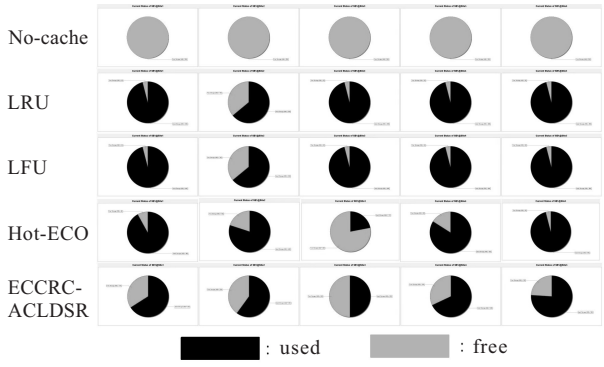

图 5　资源缓存算法各边缘服务器存储使用率

表 4　资源缓存算法边云协同费用

算法	边云协同费用 / 元
No-cache	22 372.04
LRU	21 131.06
LFU	20 029.09
Hot_ECO	16 756.76
ECCRC-ACLDSR	14 514.13

由图3可见: No-cache算法因大部分的资源都由远程访问获得,数据传输导致算法耗时最长; LRU、LFU等算法只是简单执行缓存操作,在存储容量不足时遵照简单的规则删除缓存,以腾出空间缓存新资源,由于没有拒绝缓存的操作,导致缓存很可能出现频繁“抖动”的现象,即有价值的缓存被频繁删除,又被频繁缓存,优化效果一般; Hot_ECO算法基于资源热度和经济模型进行资源价值的预测,存在一定“慢热”的现象,在任务数量不多时,没有充足的资源历史信息,对资源缓存的判断有一定的误差,对算法效果有一定影响; 本文提出的ECCRC-ACLDSR资源缓存算法能够根据当前边缘服务器的剩余存储容量、资源获取的难易程度实时自主决策,具有高效且合理的自治性,实验结果表示,算法在降低任务平均延时上取得了良好的优化效果,比No-cache提高38.2%,比LRU提高22.2%,比LFU提高14.6%,比Hot_ECO提高9.1%.

由图4可见: No-cache算法因大部分的资源都由远程访问获得,频繁的网络传输使网络带宽占用率最高; LRU、LFU等算法由于策略简单而导致资源本地缓存命中率一般,较多的资源依然通过远程访问获得,故网络带宽占用率也较高; Hot_ECO算法由于“慢热”的原因,导致前期网络带宽占用率较高; 本文提出的ECCRC-ACLDSR资源缓存算法能够有效提高资源本地缓存的命中率,从而有效减少骨干网络的传输压力,在优化网络带宽占用率上比No-cache降低49.3%,比LRU降低36.2%,比LFU降低28.1%,比Hot_ECO降低15.3%.

由图5和表4可见: No-cache算法因大部分的资源都由远程访问获得,频繁的网络传输使网络传输费用较大; LRU、LFU等算法由于总是执行缓存操作,使得边云服务器存储始终处于占满状态,从而产生了较大的存储费用; Hot_ECO算法由于前期对资源价值和热度的判断误差,导致前期费用较高,后期虽有较大降低,但在任务量达不到一定量级的情况下仍会产生较多费用; 本文提出的ECCRC-ACLDSR资源缓存算法能够有效提高资源本地缓存的命中率,从而降低网络传输费用,同时边缘服务器能够自主地决定缓存操作和清除操作,使边缘服务器存储容量的占用保持在合理区间,有效降低了存储费用,边云协同费用比No-cache降低35.1%,比LRU降低31.3%,比LFU降低27.5%,比Hot_ECO降低13.4%.

4.2　ECCTS-BCLDAI任务调度算法实验分析

接下来验证ECCTS-BCLDAI任务调度算法的性能. 将资源缓存算法设为本文提出的ECCRC-ACLDSR,参数设置为$\alpha = 1, \beta = 1, H^{\text{high}} = 0.7$, $H^{\text{low}} = 0.63$. 进行1 000个任务的模拟仿真,分别将

ECCTS-BCLDAI任务调度算法与文献[32]中提出的PMC算法,以及Random、Cloud-First、Edge-First等调度方法比较,验证该算法在缩短平均任务执行时长、降低整体设备租赁费用方面的表现.

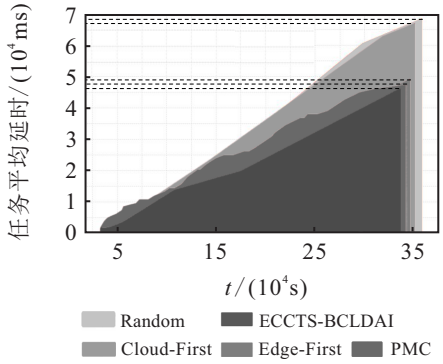

图 6 任务调度算法任务平均延时

由图6可见: Random和Cloud-First调度算法没有考虑云端与用户端之间的网络延时和云端与边缘服务器各节点的负载情况,导致任务平均延时较高; Edge-First算法尽可能地将任务调度到边缘服务器上执行,在优化任务平均延时方面取得了良好效果,但在有大量且集中的爆发式用户访问时可能导致边缘节点负载过大,任务延时在短时间内反而变大; PMC算法以蚁群算法为基础,在算法过程中易陷入局部最优,虽然加入轮盘赌随机选择增加跳出局部最优的概率,但收敛较慢,平均任务延时仍有提升空间; ECCTS-BCLDAI将任务历史平均延时作为激发剂,将边缘服务器节点负载作为内部抑制剂,可以在任务历史平均延时变大时将任务调度到边缘服务器上以减少任务执行时间,同时在边缘服务器负载过大时将任务适当调度到云端执行,缓解边缘服务器负载,从而达到动态自主优化的效果,其任务平均延时最短,比Random降低32.6%,比Cloud-First降低30.4%,比Edge-First降低5.1%,比PMC降低3.7%.

在对5种任务调度算法优化任务平均延时分析的基础上,由表5可以看出: Random算法在降低边云协同费用和优化任务平均延时方面都表现不佳; Cloud-First算法虽然有最小的边云协同费用,但其在优化任务平均延时方面与Edge-First、 ECCTS-BCLDAI有着较大的差距; Edge-First算法虽然在优化任务平均延时方面表现较好,但由于边缘服务器的租赁费用普遍高于云计算中心的租赁费用,算法有着较高的边云协同费用; PMC算法由于信息素更新没有将费用因素考虑进去,同样存在边缘服务器资源占用较高问题,算法有进一步改进的空间. 本文提出的ECCTS-BCLDAI算法将历史任务的边云协同费用作为自己获得的外部抑制剂,能够在边云协同费用过大时,将任务适当调度到云端执行,以获得较小的计算和存储费用,其在有效优化任务平均延时的同时,在降低边云协同费用方面也有着较好的表现,比Random降低3.8%,比Edge-First降低3.7%,比PMC降低1.6%.

表 5 任务调度算法边云协同费用

算法	边云协同费用 / 元
Random	14 514.13
Cloud-First	13 595.37
Edge-First	14 496.23
PMC	14 178.98
ECCTS-BCLDAI	13 953.61

综上,根据模拟仿真结果可以得出,将ECCTS-BCLDAI任务调度算法与ECCRC-ACLDSR资源缓存算法相结合应用于边云协同上,可以有效缩短任务平均执行时长,降低网络带宽占用率,减少边云协同费用,具有良好的动态自主性.

5 结 论

本文探讨高效的边云协同任务调度和资源缓存策略,提出了边云协同任务调度和资源缓存网络模型,并基于上述模型,结合自组织劳动分工群智能算法模型机理和边缘计算中任务调度、资源缓存的特点,提出了基于蜂群劳动分工“激发-抑制”模型的边云协同任务调度算法(ECCTS-BCLDAI)和基于蚁群劳动分工“刺激-响应”模型的边云协同资源缓存算法(ECCRC-ACLDSR). ECCRC-ACLDSR资源缓存算法能够根据当前边缘服务器的剩余存储容量、资源获取的难易程度自主地决定资源的缓存与否,具有高效且合理的自治性; ECCTS-BCLDAI任务调度算法通过激发抑制的相互作用,可在任务历史平均延时变大时将任务调度到边缘服务器上以减少任务执行时间,同时在边缘服务器负载过大时将任务适当调度到云端执行,缓解边缘服务器负载,具有动态自主优化效果.

仿真结果表明,本文提出的ECCTS-BCLDAI任务调度算法在优化平均任务执行时长,降低边云协同费用上与传统的Cloud-First、Edge-First、PMC等算法相比有更优异的整体表现; ECCRC-ACLDSR资源缓存算法在优化任务平均时长,优化网络带宽占用率,优化边云协同费用上比传统的No-cache、LRU、LFU、Hot_ECO等算法均有大幅提升.

参考文献(References)

[1] 周悦芝, 张迪. 近端云计算: 后云计算时代的机遇与挑战[J]. 计算机学报, 2019(4): 677-700.
(Zhou Y Z, Zhang D. Near-end cloud computing: Opportunities and challenges in the post-cloud computing era[J]. Chinese Journal of Computers, 2019(4): 677-700.)

[2] Gkatzios N, Anastasopoulos M, Tzanakaki A, et al. Optimized placement of virtualized resources for 5G services exploiting live migration[J]. Photonic Network Communications, 2020, 40(3): 233-244.

[3] Alnoman A, Sharma S K, Ejaz W, et al. Emerging edge computing technologies for distributed IoT systems[J]. IEEE Network, 2019, 33(6): 140-147.

[4] Zhou Z Y, Liao H J, Wang X Y, et al. When vehicular fog computing meets autonomous driving: Computational resource management and task offloading[J]. IEEE Network, 2020, 34(6): 70-76.

[5] Satyanarayanan M. The emergence of edge computing[J]. Computer, 2017, 50(1): 30-39.

[6] Shi W S, Cao J, Zhang Q, et al. Edge computing: Vision and challenges[J]. IEEE Internet of Things Journal, 2016, 3(5): 637-646.

[7] 孙立峰, 胡文, 马茗, 等. 基于边缘计算的高效视频内容分发关键技术与挑战[J]. 无线电通信技术, 2020, 46(3): 261-270.
(Sun L F, Hu W, Ma M, et al. Edge computing based video content delivery: Challenge and technology[J]. Radio Communications Technology, 2020, 46(3): 261-270.)

[8] 王凌, 吴楚格, 范文慧. 边缘计算资源分配与任务调度优化综述[J]. 系统仿真学报, 2021, 33(3): 509-520.
(Wang L, Wu C G, Fan W H. A survey of edge computing resource allocation and task scheduling optimization[J]. Journal of System Simulation, 2021, 33(3): 509-520.)

[9] 苏命峰, 王国军, 李仁发. 边云协同计算中基于预测的资源部署与任务调度优化[J]. 计算机研究与发展, 2021, 58(11): 2558-2570.
(Su M F, Wang G J, Li R F. Resource deployment with prediction and task scheduling optimization in edge cloud collaborative computing[J]. Journal of Computer Research and Development, 2021, 58(11): 2558-2570.)

[10] 李津, 章雨鹏, 庞玲, 等. 移动边缘计算中的资源分配与任务调度方法[J]. 重庆理工大学学报: 自然科学, 2020, 34(11): 156-163.
(Li J, Zhang Y P, Pang L, et al. Joint resource allocation and task scheduling in mobile edge computing[J]. Journal of Chongqing University of Technology: Natural Science, 2020, 34(11): 156-163.)

[11] 刘雷, 陈晨, 冯杰, 等. 车载边缘计算中任务卸载和服务缓存的联合智能优化[J]. 通信学报, 2021, 42(1): 18-26.
(Liu L, Chen C, Feng J, et al. Joint intelligent optimization of task offloading and service caching for vehicular edge computing[J]. Journal on Communications, 2021, 42(1): 18-26.)

[12] 张帆, 王亚刚, 刘子杰. 基于延迟敏感应用的边云协同方案[J]. 计算机应用研究, 2022, 39(2): 543-547.
(Zhang F, Wang Y G, Liu Z J. Edge cloud collaboration scheme based on delay sensitive application[J]. Application Research of Computers, 2022, 39(2): 543-547.)

[13] Bonabeau E, Dorigo M, Theraulaz G. Swarm intelligence: From natural to artificial systems[M]. New York: Oxford University Press, 1999.

[14] 肖人彬, 王英聪. 群智能自组织劳动分工研究进展[J]. 信息与控制, 2019, 48(2): 129-139.
(Xiao R B, Wang Y C. Research progress of self-organized labor division in swarm intelligence[J]. Information and Control, 2019, 48(2): 129-139.)

[15] 王英聪, 肖人彬. 求解卫星舱布局问题的蚁群劳动分工优化算法[J]. 控制与决策, 2021, 36(7): 1637-1646.
(Wang Y C, Xiao R B. Ant colony labor division optimization algorithm for satellite module layout design[J]. Control and Decision, 2021, 36(7): 1637-1646.)

[16] 王英聪, 刘军辉, 肖人彬. 基于刺激-响应分工机制的人工蜂群算法[J]. 控制与决策, 2022, 37(4): 881-891.
(Wang Y C, Liu J H, Xiao R B. Artificial bee colony algorithm based on stimulus-response labor division[J]. Control and Decision, 2022, 37(4): 881-891.)

[17] Zahadat P, Hahshold S, Thenius R, et al. From honeybees to robots and back: Division of labour based on partitioning social inhibition[J]. Bioinspiration & Biomimetics, 2015, 10(6): 066005.

[18] 杨惠珍, 王强. 基于动态蚁群劳动分工模型的多AUV任务分配方法[J]. 控制与决策, 2021, 36(8): 1911-1919.
(Yang H Z, Wang Q. A multi-AUV dynamic task allocation method based on antcolony labor division model[J]. Control and Decision, 2021, 36(8): 1911-1919.)

[19] Kim M H, Baik H, Lee S. Response threshold model based UAV search planning and task allocation[J]. Journal of Intelligent & Robotic Systems, 2014, 75(3/4): 625-640.

[20] 付主木, 王俊朋, 司鹏举, 等. 基于李雅普诺夫随机优化的车辆边缘计算资源管理[J]. 控制与决策, 2022, 37(3): 721-728.
(Fu Z M, Wang J P, Si P J, et al. Resource management

of vehicle edge computing based on Lyapunov stochastic optimization[J]. Control and Decision, 2022, 37(3): 721-728.)

[21] Li C L. Joint optimization of data placement and scheduling for improving user experience in edge computing[J]. Journal of Parallel and Distributed Computing, 2019, 125: 93-105.

[22] Liu J, Mao Y Y, Zhang J, et al. Delay-optimal computation task scheduling for mobile-edge computing systems[C]. IEEE International Symposium on Information Theory. Barcelona, 2016: 1451-1455.

[23] 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 offloading strategy of mobile edge computing[J]. Control and Decision, 2019, 34(11): 2366-2374.)

[24] Ahlehagh H, Dey S. Video-aware scheduling and caching in the radio access network[J]. IEEE/ACM Transactions on Networking, 2014, 22(5): 1444-1462.

[25] Baştuğ E, Bennis M, Kountouris M, et al. Cache-enabled small cell networks: Modeling and tradeoffs[J]. EURASIP Journal on Wireless Communications and Networking, 2015, 2015: 41.

[26] Naug D. Flexible division of labor mediated by social interactions in an insect colony — A simulation model[J]. Journal of Theoretical Biology, 1999, 197(1): 123-133.

[27] Gadagkar R. Division of labour and organization of work in the primitively eusocial wasp Ropalidia marginata[J]. Journal of Molecular Structure, 2001, 67(6): 397-422.

[28] 阿里云. 云服务器实例价格[EB/OL]. (2022-03-01)[2022-03-01]. https://www.aliyun.com/price/product#/ecs/detail.

[29] 孙新, 李庆洲, 赵璞, 等. 对等网络中一种优化的副本分布方法[J]. 计算机学报, 2014, 37(6): 1424-1434.
(Sun X, Li Q Z, Zhao P, et al. An optimized replica distribution method for peer-to-peer network[J]. Chinese Journal of Computers, 2014, 37(6): 1424-1434.)

[30] Wang N, Shen G X, Bose S K, et al. Zone-based cooperative content caching and delivery for radio access network with mobile edge computing[J]. IEEE Access, 7: 4031-4044.

[31] 李长乐, 张云锋, 张尧, 等. 面向自动协同驾驶的多车编队任务分配策略[J]. 电子与信息学报, 2020, 42(1): 65-73.
(Li C L, Zhang Y F, Zhang Y, et al. Task assignment strategy for platoons in cooperative driving[J]. Journal of Electronics & Information Technology, 2020, 42(1): 65-73.)

[32] Zhao P, Shang J T, Lin J J, et al. A dynamic convergent replication strategy based on distributed hierarchical systems[C]. Proceedings of the International Conference on Network, Communication and Computing. Kunming, 2017: 7-13.

[33] 肖人彬, 冯振辉, 王甲海. 群体智能的概念辨析与研究进展及应用分析[J]. 南昌工程学院学报, 2022, 41(1): 1-21.
(Xiao R B, Feng Z H, Wang J H. Collective intelligence: Conception, research progress and application analysis[J]. Journal of Nanchang Institute of Technology, 2022, 41(1): 1-21.)

作者简介

赵璞(1989–), 男, 副研究员, 博士生, 从事群智能优化、劳动分工等研究, E-mail: zhaojialipu@126.com;

肖人彬(1965–), 男, 教授, 博士生导师, 从事群智能、涌现计算、复杂系统建模与仿真等研究, E-mail: rbxiao@hust.edu.cn.

(责任编辑：齐 霁)