区块链赋能的边缘异构计算系统中资源调度研究

张平[1]，李世林[1]，刘宜明[1]，秦晓琦[1]，许晓东[2]
（1. 北京邮电大学网络与交换技术国家重点实验室，北京 100876；2. 北京邮电大学移动网络技术国家工程实验室，北京 100876）

摘 要：在区块链赋能的移动边缘计算（BMEC）系统中，针对各类新型计算任务并行性需求的差异，提出了一种基于异构计算的 BMEC 系统模型，通过调用异构计算架构中并行计算能力不同的处理器，实现区块链业务与用户业务的高效处理。通过综合考虑异构处理器调度、计算资源分配以及带宽资源分配，将通信及计算资源受限下的系统效用最大化问题建模为混合整数非线性问题。为了快速求解该问题，将所提模型进一步解耦为业务驱动的异构处理器调度问题和资源联合分配问题，并提出了基于拉格朗日对偶理论的联合优化算法。仿真结果表明，所提算法可以有效提升 BMEC 系统的系统效用。

关键词：移动边缘计算；区块链；异构计算；计算卸载
中图分类号：TN92
文献标识码：A
doi: 10.11959/j.issn.1000−436x.2020206

Resource management in blockchain-enabled heterogeneous edge computing system

ZHANG Ping[1], LI Shilin[1], LIU Yiming[1], QIN Xiaoqi[1], XU Xiaodong[2]
1. State Key Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications, Beijing 100876, China
2. National Engineering Laboratory for Mobile Network Technologies, Beijing University of Posts and Telecommunications, Beijing 100876, China

Abstract: In blockchain-enabled mobile edge computing (BMEC) systems, a new class of blockchain application related computation tasks was introduced to the system. Due to the differences of parallelism among computation tasks, heterogenous computing framework was introduced to suitably split various computation tasks on processors with vastly different processing power to achieve efficient task execution. Under the limited computation and communication resources, a system-wide utility maximization problem by jointly considering heterogeneous processor scheduling, computation and bandwidth resource allocation was formulated as a mixed-integer nonlinear programming problem. To solve the problem efficiently, the formulated problem was transformed into two sub-problems, namely application-driven heterogeneous processor scheduling and joint resource allocation, and a Lagrange-dual based algorithm was proposed. Simulation results show that the proposed scheme can effectively improve the system-wide utility of the BMEC system.

Key words: mobile edge computing, blockchain, heterogeneous computing, computation offloading

1 引言

随着 5G 的发展和移动终端设备的普及，日益涌现的移动应用及其产生的数据量将急剧增长[1]。其中，人脸识别、交互式在线游戏、虚拟/增强现实、4K/8K 超高清视频等时延敏感型、计算密集型的应用不仅需要消耗大量的计算资源，在用户服务质量方面也提出了超低时延、超高可靠性的要求。围绕 6G 网络的总体愿景，未来移动通信网络将以智能简约作为设计内涵[2]，充分利用边缘泛在的计算能

收稿日期：2020−07−16；**修回日期：**2020−09−21
基金项目：国家重点研发计划基金资助项目（No.2018YFB1800800）；北京市自然科学基金资助项目（No.19L2033）
Foundation Items: The National Key Research and Development Program of China (No.2018YFB1800800), Beijing Natural Science Foundation (No.19L2033)

力，促使通信网络更加扁平化[3-4]。在云计算、移动边缘计算（MEC, mobile edge computing）的发展大趋势下，未来移动用户的周围将部署不同规模的边缘服务器以完成复杂任务处理，并利用通信资源完成计算任务数据的高效传输，逐步形成通信资源赋能“端−边−云”协同计算的新范式[5-8]。

为了满足人脸识别、虚拟/增强现实等计算密集型移动应用超低时延的服务需求，可以在靠近用户侧的边缘服务器上进行计算任务处理。由于单一边缘服务器能力有限，通常需要多个边缘服务器的协同处理，且涉及计算数据迁移、计算结果共享以及多维资源的使用。然而，泛在接入的边缘网络极易受到恶意节点的信息干扰或数据攻击，恶意节点可能篡改甚至伪造计算迁移请求、计算处理结果以及资源调度指令，破坏正常计算迁移、计算结果共享与资源调度进程。因此，在边缘侧环境异构开放、节点互不信任的情况下，如何保障计算卸载服务、计算结果共享和资源调度过程的安全可信与高效协同，是一个亟需解决的关键问题[9]。

目前，学术界和产业界普遍认为区块链技术是助力实现安全可信 MEC 生态的关键技术[10-11]。区块链技术本质上是一种基于分布式对等网络的分布式账本技术，具有去中心化、不可篡改、可追溯、匿名性和透明性五大特征，这些特征为构建安全可信的分布式交易环境提供了良好的契机[12-15]。在区块链赋能的移动边缘计算（BMEC, block-chain-enabled mobile edge computing）系统中，在不需要第三方授权机构或平台背书的情况下，移动终端设备与边缘服务器之间可以自由发起计算卸载和资源调度请求，利用密码学原理，如非对称加密算法和哈希算法，可以对边缘资源调度指令、计算结果、交易记录等敏感信息进行保护与验证，同时利用共识机制，对服务与资源交易记录达成一致确认，进而保障资源与服务交易过程的安全、可信与透明[16-19]。然而，在区块链系统的共识过程中，需要利用计算及通信资源完成区块的生成、验证及传输等关键步骤，这对有限的计算资源和通信资源的高效调度提出了更高的要求。

在 BMEC 系统中，不仅需要考虑移动用户的计算任务卸载请求，还需要考虑来自区块链系统的计算任务。值得注意的是，用户卸载计算任务和区块链系统的计算任务在并行处理方面存在较大差异。具体来说，用户卸载计算任务中的视频转码、人工智能推理、图像识别等高并行性计算任务可以被划分为大量相互独立的子任务进行并行计算，而其他复杂的计算任务如基于有向无环图（DAG, directed acyclic graph）的高斯消元和快速傅里叶变换，往往拥有不同程度的内部关联性，导致其在并行处理方面存在诸多限制。此外，BMEC 系统中的区块生成和验证也是一种高并行性计算任务。针对不同计算任务之间存在的并行性需求差异，众多学者和研究人员提出异构计算架构，联合优化中央处理单元（CPU, central processing unit）和图形处理单元（GPU, graphics processing unit）调度进行加速计算[20]。其中，CPU 一般只包含若干个单线程或多线程内核，仅可以实现粗粒度并行（CP, coarse-grained parallelism），对计算任务兼容性较好；GPU 通过装载数以千计的内核实现细粒度并行（FP, fine-grained parallelism），拥有强大的并行计算能力，但不适用于对复杂度较高的计算任务。针对 BMEC 系统，异构计算架构可以满足不同计算任务的并行性需求，使计算任务的并行性类型与处理器类型相匹配，充分利用各种计算资源进行并行计算，达到降低计算任务执行时间的目的[20]。

在本文中，考虑到 BMEC 系统中区块链计算任务和用户卸载计算任务的并行性需求存在差异，引入异构计算架构，优化 CPU-GPU 异构处理器调度策略，联合分配通信和计算资源，实现系统性能的提升。考虑处理器调度约束、计算资源限制、通信资源限制，本文提出了联合优化处理器调度与资源分配（PSRA, processor scheduling and resource allocation）算法。本文的主要工作总结如下：1) 考虑不同计算任务的并行性需求，引入异构计算架构解决计算任务并行性类型与处理器类型之间的匹配问题；2) 考虑用户任务处理效率和系统区块生成效率占系统整体性能的权重，通过计算资源和带宽资源的合理分配，提升用户任务处理效率和系统区块生成效率。

2 相关工作

与本文相关的研究工作的介绍从以下三方面展开：区块链在 MEC 系统中的应用、CPU-GPU 异构计算架构与应用、基于 GPU 的区块链应用。

文献[16-19]主要针对 BMEC 场景展开研究。文献[16]考虑了区块链系统区块最终生成时延与 MEC 系统能耗之间的平衡问题，以两者加权和最小

化为目标，将计算卸载决策、传输速率分配、区块生成调度、计算资源分配的联合优化问题建模为混合整数非线性规划（MINLP, mixed-integer nonlinear programming）问题，在解耦优化变量的基础上提出了迭代算法用于解决卸载决策和功率分配问题，并利用二分法解决了计算资源分配问题。针对传统卸载决策、资源分配、区块生成控制方法无法根据环境变化实时调整策略的缺陷，文献[17-19]将优化问题建模为马尔可夫决策过程（MDP, Markov decision process），并利用深度强化学习（DRL, deep reinforcement learning）算法进行求解。文献[17]采用联合基于股份授权证明（DPoS, delegated proof of stake）和实用拜占庭容错（PBFT, practical Byzantine fault tolerance）的共识机制，对 BMEC 系统的计算卸载请求与处理结果进行记录并存储至区块链，提出了自适应的资源分配和区块生成方案，以提升由区块链系统吞吐量和边缘计算系统用户服务质量决定的系统长期奖励。为了实现长期计算卸载增益最大化并适应环境的高动态性，文献[18]研究了多跳网络中数据处理任务和区块挖掘任务的实时联合卸载控制问题，并提出 DRL 算法对问题进行求解。以区块链系统交易吞吐量和 MEC 系统计算速率最大化为目标，文献[19]对计算卸载决策、传输功率、块大小、块间隔进行联合优化，并针对决策空间同时存在离散决策和连续决策的特征，提出了基于异步优势行动者−评价家（A3C, asynchronous advantage actor-crigic）的 DRL 算法。

考虑 CPU-GPU 协同计算的模式，文献[21]设计了基于 CPU-GPU 异构处理器的云计算架构，利用自适应分层、分层调度、性能估计等方法提升系统计算性能并降低计算和通信开销。文献[22]介绍了一种基于 CPU 与 GPU 的云计算平台 Ankea，其是一个完整的平台即服务（PaaS, platform as a service）框架，包含用于应用程序快速开发的多种编程模型，支持基于分布式异构计算资源的编程模型部署。文献[23]针对视频编码过程中并行度较低的问题，提出了基于 CPU-GPU 异构计算系统的多粒度并行计算方法，提升视频编码过程中并行执行的效率。文献[24]针对远程医疗数据加密时延较高的问题，提出了基于 CPU-GPU 异构计算系统的加密函数调度优化算法。

GPU 加速计算在区块链系统中得到了广泛应用[25-28]。文献[25]基于不同规格的 CPU、GPU 进行了包括比特币、以太坊在内的多类数字加密货币的挖掘测试，测试结果表明在数字货币挖掘的哈希速率方面，GPU 相较于 CPU 有数十倍的速率提升。文献[26]同样利用基于统一计算设备架构（CUDA, compute unified device architecture）的 GPU 提升了区块链挖掘效率，证明 GPU 并行计算的优势只有当并行挖掘的任务数量大于 800 时才得以体现。文献[27]针对区块链系统中大量用户搜索给全节点带来的计算瓶颈问题，利用区块链数据不能删除和更改的特征，引入适合 GPU 并行计算的帕特里夏树（PT, Patricia tree）数组用于存储区块数据，实验结果表明基于 PT 数组和 GPU 的搜索吞吐量是基于 CPU 的搜索吞吐量的 14.1 倍。针对区块链系统中交易数据异常检测过程中的时延敏感、计算密集类任务，文献[28]将需要提取特征信息的交易数据缓存至 GPU 存储单元中，并在 GPU 中进行特征提取和异常检测，实验结果表明 GPU 异常检测速率可以达到 CPU 异常检测速率的 37.1 倍。

与已有工作相比，本文考虑基于 CPU-GPU 异构计算架构的 BMEC 系统，以系统区块生成效率和用户任务处理效率共同决定的系统效用最大化为研究目标，建立异构处理器调度、计算资源分配、带宽资源分配的联合优化问题。本文将系统效用最大化问题建模为 MINLP 问题。考虑到 MINLP 问题的高复杂性，将原优化问题分解为处理器调度优化问题和资源联合分配优化问题，并对应提出了 PSRA 算法。

3　BMEC 系统模型

本文考虑了基于异构计算的 BMEC 系统，在边缘服务器上分别部署了 CPU、GPU 两类处理器。本文将移动应用划分为低并行性普通应用和高并行性特殊应用（后文简称为普通应用和特殊应用），前者仅可以调用 CPU 处理器，后者不仅可以调用 CPU 处理器还可以调用 GPU 处理器。异构计算实现过程如图 1 所示，利用虚拟机（VM, virtual machine）技术，将边缘服务器中的 CPU、GPU 计算资源虚拟化并分配给不同的虚拟机，以处理区块链计算任务和用户卸载计算任务。本文假设任意虚拟机仅可同时调用一类处理器处理计算任务，且两类计算资源可以按照任意比例分配给虚拟机。

3.1　网络模型

在本文中，利用区块链技术，BMEC 系统可以对多方共同参与计算卸载和任务执行信息进行记

录，包括移动用户的计算卸载请求及任务处理结果等信息。值得注意的是，所有交易信息都先经过区块链共识节点的验证与确认，然后才会被存储至所有区块链节点备份中。BMEC 系统网络模型如图 2 所示。BMEC 系统主要由区块链用户、区块生成（BP, block producer）节点构成，BP 节点又分为主节点（PN, primary node）和备份节点（BN, backup node），具体介绍如下。

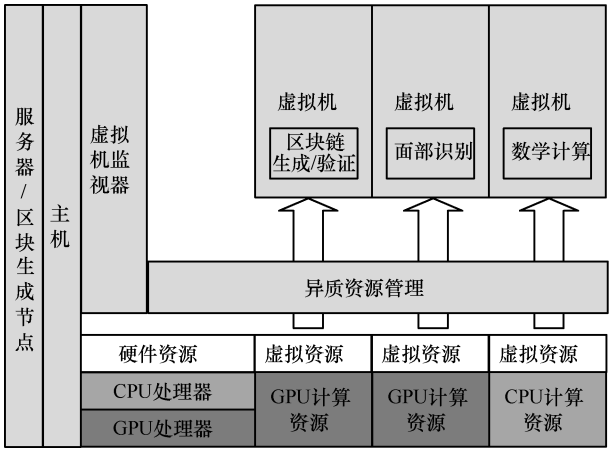

图 1 异构计算实现过程

区块链用户。系统中的移动用户，可以提交计算卸载请求，边缘服务器完成计算任务并将处理结果反馈至移动用户。

区块生成节点。系统中的边缘服务器，不仅提供计算卸载服务，还负责区块的生成和验证。

主节点。在特定时间段从区块生成节点中选出的单个节点，被授权在该时间段生成新的区块。

备份节点。区块生成节点中除主节点以外的所有节点，负责新区块的验证工作。

交易信息。记录移动用户的计算卸载请求及任务处理结果，共享于整个区块链网络，由主节点收集并生成新的区块。

本文采用基于 DPoS-PBFT 的共识机制[17]，假设 M 个边缘服务器（区块生成节点）以 T 为时延间隔轮流生成区块。服务器集合由 $\prod=\{1,2,\cdots,M\}$ 表示，m 表示第 m 个服务器。假设恶意节点数量小于 $\frac{M-1}{3}$。简单来说，在指定时间内，PN 收集交易信息并将新生成的区块广播至所有 BN 进行验证，即共识过程。当所有 BP 节点达成共识后，新的区块才会被添加至区块链，然后被存储至所有区块链节点。

假设服务器 m 可用的 CPU、GPU 计算资源分别为 F_{C}、F_{G}（单位为 cycle/s）。服务器中部署的应用集合表示为 $\psi=\{1,2,\cdots,A\}$，A 表示应用数量，a 表示第 a 个应用。应用类型表示为 $\rho_a\in\{0,1\}$，$\rho_a=0$ 表示 a 为普通应用，$\rho_a=1$ 表示 a 为特殊应用。假设任意服务器 m 服务于 N_m 个移动用户，用户集合表示为 $\Gamma_m=\{1_m,2_m,\cdots,N_m\}$，$n_m$ 表示由服务器 m 服务的第 n 个用户。此外，$n_m=0$ 表示区块链计算任务。用户 n_m 的计算任务可以表示为数组 $\mathbf{CT}_{m,n}\triangleq\{a_{m,n},\alpha_{m,n},\beta_{m,n},\gamma_{m,n},\eta_{m,n}\}$，其中 $a_{m,n}\in\psi$ 表示任务所属的应用，$\alpha_{m,n}$ 表示任务数据量（单位为 bit），$\beta_{m,n}$ 表示利用 CPU 完成计算任务

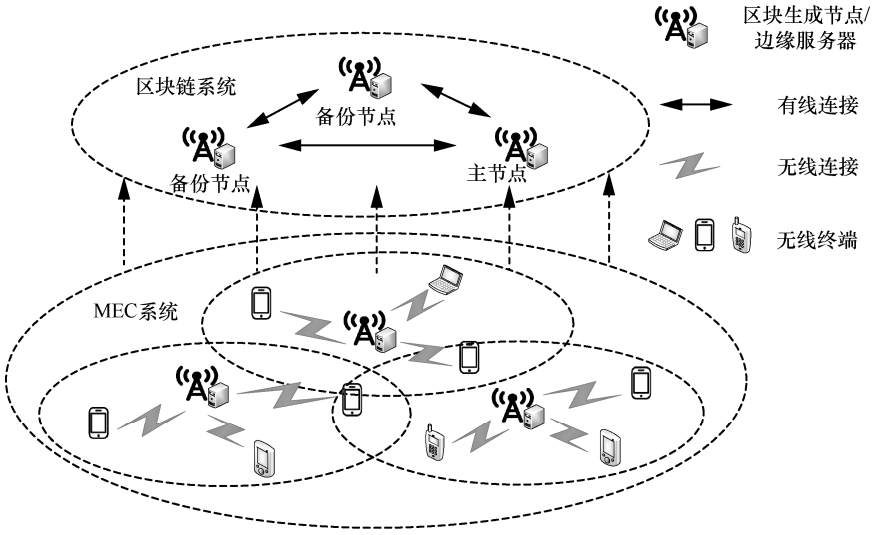

图 2 BMEC 系统网络模型

所需要的计算量，$\gamma_{m,n}$ 表示利用 GPU 完成计算任务所需要的计算量，$\eta_{m,n}\in(0,1]$ 表示计算任务对时延的敏感程度。由于不同种类应用对 GPU 的利用效率不同，本文假设 $\gamma_{m,n}=\dfrac{\beta_{m,n}}{X_{m,n}}$，其中 $X_{m,n}\in(0,1]$ 表示计算任务 n_m 对 GPU 的利用效率，$X_{m,n}$ 由任务类型决定，$X_{m,n}$ 越大表明对 GPU 的利用效率越高。本文考虑采用正交频分多址接入（OFDMA, orthogonal frequency division multiple access）方法，通信带宽资源被划分为数量为 E 的等带宽子信道，每个用户都可以被分配到若干个子信道用于无线传输。详细系统参数表示和相关含义如表 1 所示。

表 1　　系统参数表示和相关含义

参数	含义
M	区块生成节点（边缘服务器）数量
N	移动用户数量
A	移动应用数量
$F_{\mathrm{C}},F_{\mathrm{G}}$	CPU、GPU 计算资源
ρ_a	应用类型
$a_{m,n},\alpha_{m,n}$	任务 n_m 类型与数据量
$\beta_{m,n},\gamma_{m,n}$	任务 n_m 的 CPU、GPU 计算量
$\eta_{m,n}$	任务 n_m 对时延的敏感度
$X_{m,n}$	任务 n_m 的 GPU 利用效率
E	带宽资源数量
$f_{\min}^{\mathrm{C}},f_{\min}^{\mathrm{G}}$	维持虚拟机正常运行的最低计算资源
$r_{m,n}^{0}$	用户 n_m 基于单位子信道的传输速率
$\boldsymbol{\lambda}_{m,n}$	处理调度决策
$\boldsymbol{f}_{m,n}$	计算资源分配方案
$\boldsymbol{W}_m$	带宽资源分配方案
S_B	区块数据量
ϖ	平均交易数据量
ϕ	验证/生成 SIG 需要的 CPU 计算量
φ	验证/生成 MAC 需要的 CPU 计算量
g	有效交易记录比例

考虑到 BMEC 系统中的计算负载分别来源于移动用户卸载的计算任务和区块链系统中区块生成、区块验证、达成共识所需要完成的计算任务，接下来将分别对区块链任务计算和用户卸载任务计算进行建模分析。

3.2　区块链任务计算模型

首先，移动用户向服务器发出卸载请求，服务器接受请求并处理任务；然后，将任务处理结果反馈至移动用户并将相关信息作为交易记录挂起等待 PN 的进一步处理。PN 在收集交易信息并生成区块的时候，需要对交易记录中的用户签名（SIG, signature）和消息认证码（MAC, message authentication code）进行验证。PBFT 共识协议包含以下 5 个阶段[17]：请求、序号分配、序号准备、序号确认、响应。

在请求阶段，PN 需要验证新区块中所有交易记录的 SIG 和 MAC 并生成新的区块，完成交易验证任务和区块生成任务所需要的 CPU 计算量为 $\beta_0^{\mathrm{req}}=\dfrac{S_B(\phi+\varphi)}{g\varpi}$，其中，$S_B$、$\varpi$、$\phi$、$\varphi$、$g$ 分别表示区块数据量、平均交易数据量、验证/生成 SIG 需要的 CPU 计算量、验证/生成 MAC 需要的 CPU 计算量、有效交易记录比例。

在序号分配阶段，PN 为新区块附上 SIG 与 MAC，并广播至所有 BN。BN 收到新区块之后，需要验证新区块及其包含所有交易记录的 SIG 与 MAC。因此，此阶段在任意 BN 处需要的 CPU 计算量为 $\beta_0^{\mathrm{prep}}=\left(\dfrac{S_B}{\varpi}+1\right)(\phi+\varphi)$。

在响应阶段，新区块经过 BN 验证为有效区块后，所有 BN 需要为新区块中的每条交易记录附加新的 SIG 和 MAC，然后将其传回 PN。因此，此阶段在任意 BN 处需要的 CPU 计算量为 $\beta_0^{\mathrm{rep}}=\dfrac{S_B(\phi+\varphi)}{\varpi}$。值得注意的是，序号准备与确认阶段，各 BP 节点的计算需求量相对较小，本文忽略不计。

若服务器 m 为 PN，则完成区块链计算任务的时延为

$$d_{m,0}^{\mathrm{PN}}=\frac{\lambda_{m,0}^{\mathrm{C}}\beta_{m,0}^{\mathrm{PN}}}{f_{m,0}^{\mathrm{C}}}+\frac{\lambda_{m,0}^{\mathrm{G}}\gamma_{m,0}^{\mathrm{PN}}}{f_{m,0}^{\mathrm{G}}}\tag{1}$$

其中，$\boldsymbol{\lambda}_{m,0}=\left[\lambda_{m,0}^{\mathrm{C}},\lambda_{m,0}^{\mathrm{G}}\right]$ 表示区块链应用的处理器调度决策，$\boldsymbol{\lambda}_{m,0}=[0,1]$ 表示调用 GPU，$\boldsymbol{\lambda}_{m,0}=[1,0]$ 表示调用 CPU，本文假设任意虚拟机同一时刻只能调用一类处理器执行计算任务，处理器调度决策需满足 $\lambda_{m,0}^{\mathrm{C}}+\lambda_{m,0}^{\mathrm{G}}=1$；$\boldsymbol{f}_{m,0}=\left[f_{m,0}^{\mathrm{C}},f_{m,0}^{\mathrm{G}}\right]$ 表示计算资源分配方案；$\gamma_{m,0}^{\mathrm{PN}}=\dfrac{\beta_{m,0}^{\mathrm{PN}}}{X_0}$ 表示完成相应任务需要的

GPU 计算量，$\beta_{m,0}^{\mathrm{PN}}=\beta_0^{\mathrm{req}}$，$X_0$ 表示区块链应用对 GPU 的利用效率。若服务器 m 为 BN，则完成区块链计算任务的时延为

$$d_{m,0}^{\mathrm{BN}}=\frac{\lambda_{m,0}^{\mathrm{C}}\beta_{m,0}^{\mathrm{BN}}}{f_{m,0}^{\mathrm{C}}}+\frac{\lambda_{m,0}^{\mathrm{G}}\gamma_{m,0}^{\mathrm{BN}}}{f_{m,0}^{\mathrm{G}}} \tag{2}$$

其中，$\gamma_{m,0}^{\mathrm{BN}}=\dfrac{\beta_{m,0}^{\mathrm{BN}}}{X_0}$ 表示完成相应任务需要的 GPU 计算量，$\beta_{m,0}^{\mathrm{BN}}\in\left\{\beta_0^{\mathrm{prep}},\beta_0^{\mathrm{rep}}\right\}$。

实时类应用往往需要其交易记录快速得到确认，因此，当服务器 m 为 PN 时，系统区块生成效率为

$$G_{m,\mathrm{bc}}^{\mathrm{PN}}=\frac{1}{d_{m,0}^{\mathrm{PN}}} \tag{3}$$

当服务器 m 为 BN 时，系统区块生成效率为

$$G_{m,\mathrm{bc}}^{\mathrm{BN}}=\frac{1}{d_{m,0}^{\mathrm{BN}}} \tag{4}$$

3.3 用户卸载任务计算模型

在用户卸载任务计算模型中，需要考虑移动用户卸载任务的时延和服务器处理计算任务的时延。用户 n_m 的上行传输速率可以表示为 $r_{m,n}=W_{m,n}r_{m,n}^0$，其中，$W_{m,n}$ 表示用户 n_m 分配到的子信道数量，$r_{m,n}^0$ 表示用户 n_m 基于单个子信道的上行传输速率。因此，用户 n_m 上传计算任务产生的时延为

$$d_{m,n}^{\mathrm{up}}=\frac{\alpha_{m,n}}{r_{m,n}} \tag{5}$$

在任务处理阶段，本文考虑每一个虚拟机都可以运行应用集合 ψ 中的任意应用，不同虚拟机可以同时运行相同的应用，但是一个虚拟机一次只能运行一个应用；服务器主机可以根据用户数量划分为数量相同的虚拟机，并将其与任务 n_m 进行关联，然后根据任务需求给不同的虚拟机分配不同的计算资源。由于任意虚拟机只能同时调用一类处理器，且与用户 n_m 关联的虚拟机的处理器调度决策可以表示为向量 $\boldsymbol{\lambda}_{m,n}=\left[\lambda_{m,n}^{\mathrm{C}},\lambda_{m,n}^{\mathrm{G}}\right]$，$\boldsymbol{\lambda}_{m,n}=[0,1]$ 表示调用 GPU，$\boldsymbol{\lambda}_{m,n}=[1,0]$ 表示调用 CPU。令向量 $\boldsymbol{f}_{m,n}=\left[f_{m,n}^{\mathrm{C}},f_{m,n}^{\mathrm{G}}\right]$ 表示计算资源分配策略，其中 $f_{m,n}^{\mathrm{C}}$ 表示与用户 n_m 关联的虚拟机所调用的 CPU 资源数量，$f_{m,n}^{\mathrm{G}}$ 表示与用户 n_m 关联的虚拟机所调用的 GPU 资源数量。则计算任务 n_m 的处理时延为

$$d_{m,n}^{\mathrm{exe}}=\frac{\lambda_{m,n}^{\mathrm{C}}\beta_{m,n}}{f_{m,n}^{\mathrm{C}}}+\frac{\lambda_{m,n}^{\mathrm{G}}\gamma_{m,n}}{f_{m,n}^{\mathrm{G}}} \tag{6}$$

计算任务 n_m 从上传至处理完成的总时延可以表示为 $d_{m,n}=d_{m,n}^{\mathrm{up}}+d_{m,n}^{\mathrm{exe}}$。由于移动应用一般为时延敏感型，其任务完成时延直接决定了用户任务处理效率。因此，本文将用户任务处理效率定义为

$$G_{m,\mathrm{oc}}=\frac{\sum_{n_m=1}^{N_m}\eta_{m,n}}{d_{m,n}} \tag{7}$$

3.4 BMEC 系统计算模型

为了均衡系统区块生成效率和用户任务处理效率，本文将服务器效用函数表示为用户任务处理效率和系统区块生成效率的加权求和。当服务器 m 为 PN 时，服务器效用函数为

$$G_m^{\mathrm{PN}}\left(\boldsymbol{\lambda}_m,\boldsymbol{f}_m,\boldsymbol{W}_m\right)=\theta_{\mathrm{M}}G_{m,\mathrm{oc}}+\theta_{\mathrm{B}}G_{m,\mathrm{bc}}^{\mathrm{PN}} \tag{8}$$

当服务器 m 为 BN 时，服务器效用函数为

$$G_m^{\mathrm{BN}}\left(\boldsymbol{\lambda}_m,\boldsymbol{f}_m,\boldsymbol{W}_m\right)=\theta_{\mathrm{M}}G_{m,\mathrm{oc}}+\theta_{\mathrm{B}}G_{m,\mathrm{bc}}^{\mathrm{BN}} \tag{9}$$

其中，$\boldsymbol{\lambda}_m=[\boldsymbol{\lambda}_{m,n}]$；$\boldsymbol{f}_m=[\boldsymbol{f}_{m,n}]$；$\boldsymbol{W}_m=[W_{m,n}]$；$\theta_{\mathrm{M}},\theta_{\mathrm{B}}\in(0,1)$，且 $\theta_{\mathrm{M}}+\theta_{\mathrm{B}}=1$，$\theta_{\mathrm{M}}$和$\theta_{\mathrm{B}}$分别表示用户任务处理效率与系统区块生成效率在系统效用中所占的比例。基于系统效用的目标函数为

$$G\left(\boldsymbol{\lambda},\boldsymbol{f},\boldsymbol{W}\right)=\sum_{m\in\Pi}z_mG_m^{\mathrm{PN}}+(1-z_m)G_m^{\mathrm{BN}} \tag{10}$$

其中，$z_m\in\{0,1\}$，$z_m=1$ 表示服务器 m 为主节点，Z_m=0 表示 m 为备份节点。本文以系统效用最大化为目标，综合考虑主节点约束、处理器调度约束、计算资源限制、带宽资源限制，将优化问题表示为 P1。

$$\mathrm{P1}:\max_{\boldsymbol{\lambda},\boldsymbol{f},\boldsymbol{W}}G\left(\boldsymbol{\lambda},\boldsymbol{f},\boldsymbol{W}\right)$$

$$\begin{aligned}
\text{s.t. }\ &\mathrm{C1}:\lambda_{m,n}^{\mathrm{C}}+\lambda_{m,n}^{\mathrm{G}}=1,\forall n\in\Gamma_{m,+},m\in\Pi\\
&\mathrm{C2}:\lambda_{m,n}^{\mathrm{G}}=0,\forall n\in\Gamma_{m,0},m\in\Pi\\
&\mathrm{C3}:\sum_{n\in\Gamma_{m,\mathrm{C}+}}f_{m,n}^{\mathrm{C}}\leqslant F_{\mathrm{C}},\forall m\in\Pi\\
&\mathrm{C4}:\sum_{n\in\Gamma_{m,\mathrm{G}+}}f_{m,n}^{\mathrm{G}}\leqslant F_{\mathrm{G}},\forall m\in\Pi\\
&\mathrm{C5}:\sum_{n\in\Gamma_m}W_{m,n}\leqslant E,\forall m\in\Pi\\
&\mathrm{C6}:\lambda_{m,n}^{\mathrm{C}},\lambda_{m,n}^{\mathrm{G}}\in\{0,1\},\forall n\in\Gamma_{m,+},\forall m\in\Pi\\
&\mathrm{C7}:f_{m,n}^{\mathrm{C}}\geqslant f_{\min}^{\mathrm{C}},\forall n\in\Gamma_{m,\mathrm{C}+};\\
&\qquad f_{m,n}^{\mathrm{G}}\geqslant f_{\min}^{\mathrm{G}},\forall n\in\Gamma_{m,\mathrm{G}+}\\
&\mathrm{C8}:W_{m,n}\in\{1,2,\cdots E\},\forall n\in\Gamma_m,\forall m\in\Pi\\
&\mathrm{C9}:\sum_{m\in\Pi}z_m=1
\end{aligned}$$

C1 表示任意虚拟机仅可以同时调用一类处理器处理计算任务。C2 则给出了一般应用不能调用GPU处理计算任务的限制，其中 $\Gamma_{m,+}=\{0\bigcup\Gamma_m\}$，$\Gamma_{m,0}=\left\{n\in\Gamma_{m,+}\middle|\rho_{a_{m,n}}=0\right\}$。C3、C4 分别给出了可分配 CPU、GPU 计算资源的上限。C5 给出了带宽资源限制条件。C6～C8 则分别给出了不同变量的取值范围，$f_{\min}^{\mathrm{C}}$ 和 $f_{\min}^{\mathrm{G}}$ 分别表示维持虚拟机运行所需要的最低 CPU 计算资源和 GPU 计算资源，其中，有

$$\Gamma_{m,\mathrm{C}+}=\left\{n\in\Gamma_{m,+}\middle|\boldsymbol{\lambda}_{m,n}=[1,0]\right\}$$

$$\Gamma_{m,\mathrm{G}+}=\left\{n\in\Gamma_{m,+}\middle|\boldsymbol{\lambda}_{m,n}=[0,1]\right\}$$

C9 表示同时最多只能有一个服务器生成新的区块。由于所有服务器轮流生成新的区块，导致 z_m 的取值是随机的，且当 z_m 确定时，所有服务器的效用最大化问题是相互独立的。因此，C9 被松弛之后，原优化问题可以简化为面向任意服务器的系统效用最大化问题 P2。

$$\mathrm{P2}:\max_{\boldsymbol{\lambda},\boldsymbol{f},\boldsymbol{W}}\left\{G(\boldsymbol{\lambda},\boldsymbol{f},\boldsymbol{W})=\theta_{\mathrm{M}}G_{\mathrm{oc}}+\theta_{\mathrm{B}}G_{\mathrm{bc}}\right\}$$

$$\text{s.t. C1~C8}$$

其中，$G_{\mathrm{bc}}=\dfrac{1}{d_0^{\mathrm{exe}}}$，$d_0^{\mathrm{exe}}=\dfrac{\lambda_0^{\mathrm{C}}\beta_0}{f_0^{C}}+\dfrac{\lambda_0^{\mathrm{G}}\gamma_0}{f_0^{G}}$，$\beta_0$ 表示完成区块链任务需要的 CPU 计算量，$\beta_0\in\left\{\beta_0^{\mathrm{req}},\beta_0^{\mathrm{prep}},\beta_0^{\mathrm{rep}}\right\}$。值得注意的是，P2 及其限制条件中的所有变量及后续涉及的变量均为原问题 P1 中对应且服务器变量 m 被松弛之后的变量。

可以看出，P2 是一个典型的 MINLP 问题，通常情况下此类问题没有有效的解决方法，需要根据具体模型设计解决方案。通过观察分析，原问题可以分解为 2 个子问题，即资源分配优化问题和处理器调度优化问题。其中，在给定任意可行处理器调度策略 $\boldsymbol{\lambda}$ 的情况下，资源分配优化问题可以表示为 P3。

$$\mathrm{P3}:\max_{\boldsymbol{f},\boldsymbol{W}}G(\boldsymbol{f},\boldsymbol{W})$$

$$\text{s.t. C3~C5,C7,C8}$$

令 $G(\boldsymbol{\lambda})=G(\boldsymbol{f}^*,\boldsymbol{W}^*)$，其中 $\boldsymbol{f}^*$ 和 $\boldsymbol{W}^*$ 分别是调度策略为 $\boldsymbol{\lambda}$ 时的最优计算资源分配方案和最优带宽资源分配方案。优化问题 P2 等价于处理器调度优化问题 P4。

$$\mathrm{P4}:\max_{\boldsymbol{\lambda}}G(\boldsymbol{\lambda})$$

$$\text{s.t. C1,C2,C6}$$

4　问题分析与算法设计

本节首先针对资源优化问题 P3 提出基于拉格朗日对偶理论的资源分配算法，然后针对处理器调度优化问题 P4 设计了基于任务处理时延的处理器调度与资源分配联合优化算法，最后对算法复杂度进行了分析。

4.1　资源分配优化问题

给定任意可行处理器调度方案 $\boldsymbol{\lambda}$，问题 P3 的目标函数可以重新表示为

$$G(\boldsymbol{f},\boldsymbol{W})=G_1(\boldsymbol{f},\boldsymbol{W})+G_2(\boldsymbol{f},\boldsymbol{W})+\frac{f_0^{\mathrm{C}}\theta_{\mathrm{B}}}{\beta_0}\tag{11}$$

其中，$G_1=\sum_{n\in\Gamma_{\mathrm{C}}}\dfrac{k_n^{(1)}W_nf_n^{\mathrm{C}}}{\alpha_nf_n^{\mathrm{C}}+k_n^{(2)}W_n}$，$G_2=\sum_{n\in\Gamma_{\mathrm{G}}}\dfrac{k_n^{(1)}W_nf_n^{\mathrm{G}}}{\alpha_nf_n^{\mathrm{G}}+k_n^{(3)}W_n}$，$k_n^{(1)}=\theta_{\mathrm{M}}\eta_nr_n^0$，$k_n^{(2)}=\beta_nr_n^0$，$k_n^{(3)}=\gamma_nr_n^0$，$\Gamma_{\mathrm{C}}=\left\{n\in\Gamma\middle|\boldsymbol{\lambda}_n=[1,0]\right\}$ 表示由 CPU 处理任务的用户集合，$\Gamma_{\mathrm{G}}=\left\{n\in\Gamma\middle|\boldsymbol{\lambda}_n=[0,1]\right\}$ 表示由GPU处理任务的用户集合。本文假设 $\boldsymbol{\lambda}_0=[1,0]$。

给定任意可行的计算资源分配方案 $\boldsymbol{f}$，并将 $\boldsymbol{W}$ 松弛为连续变量 $\boldsymbol{W}'$，$W_n'\geqslant1,\forall n\in\Gamma$，问题 P3 可以转化为问题 P5。

$$\mathrm{P5}:\min_{\boldsymbol{W}'}\left\{-G(\boldsymbol{W}')\right\}$$

$$\text{s.t. C5':}\sum_{n\in\Gamma}W_n'\leqslant E$$

$$\text{C8':}W_n'\geqslant1,\forall n\in\Gamma$$

定理 1　问题 P5 是凸优化问题。

证明　首先，根据限制条件 C5' 和 C8' 可知，目标函数 $-G(\boldsymbol{W}')$ 的可行域为凸集。根据式(12)和式(13)推导可知，目标函数 $-G(\boldsymbol{W}')$ 的海森矩阵为正定矩阵，则 $-G(\boldsymbol{W}')$ 为凸函数，问题 P5 为凸优化问题[29]。

$$\frac{\partial^2\left\{-G(\boldsymbol{W}')\right\}}{\partial{W_n'}^2}=$$

$$\begin{cases}2k_n^{(1)}k_n^{(2)}\alpha_n\left(f_n^{\mathrm{C}}\right)^2\left(\alpha_nf_n^{\mathrm{C}}+k_n^{(2)}W_n'\right)^{-3}>0,\forall n\in\Gamma_{\mathrm{C}}\\ 2k_n^{(1)}k_n^{(3)}\alpha_n\left(f_n^{\mathrm{G}}\right)^2\left(\alpha_nf_n^{\mathrm{G}}+k_n^{(3)}W_n'\right)^{-3}>0,\forall n\in\Gamma_{\mathrm{G}}\end{cases}\tag{12}$$

$$\frac{\partial^2\left\{-G\left(\boldsymbol{W}'\right)\right\}}{\partial W_n'\,\partial W_i'}=0,\forall n\neq i \tag{13}$$

证毕。

因此，问题 P5 可通过拉格朗日对偶法求解，且拉格朗日函数构造如下。

$$L\left(\boldsymbol{W}',v\right)=-G\left(\boldsymbol{W}'\right)+v\left(\sum_{n\in\Gamma}W_n'-E\right) \tag{14}$$

其中，$v\geqslant 0$ 为与 C5' 相关的拉格朗日乘子，限制条件 C8' 被暂时松弛。拉格朗日对偶函数可以表示为

$$L_{\mathrm{D}}\left(v\right)=\min_{\boldsymbol{W}'}L\left(W',v\right)$$

由此引出问题 P5 的对偶问题 $\max\limits_{v}L_{\mathrm{D}}\left(v\right)$。又由于问题 P5 满足 Salter 条件，问题 P5 与对偶问题之间的强对偶性成立。根据 Karush-Kuhn-Tucker 条件 $\dfrac{\partial L}{\partial \boldsymbol{W}'^{*}}=0$，可以求得

$$W_n'^{*}=\begin{cases}f_n^{\mathrm{C}}\dfrac{\left(\sqrt{\dfrac{k_n^{(1)}\alpha_n}{v}}-\alpha_n\right)}{k_n^{(2)}},\forall n\in\Gamma_{\mathrm{C}}\\ f_n^{\mathrm{G}}\dfrac{\left(\sqrt{\dfrac{k_n^{(1)}\alpha_n}{v}}-\alpha_n\right)}{k_n^{(3)}},\forall n\in\Gamma_{\mathrm{G}}\end{cases} \tag{15}$$

将 $\boldsymbol{W}'^{*}$ 代入拉格朗日对偶函数可以获得关于 v 的对偶问题。由于对偶问题也是凸优化问题，可以得到 $\dfrac{\partial L_{\mathrm{D}}}{\partial v^{*}}=0$，由此可以计算出

$$v^{*}=\left(\frac{\sum\limits_{n\in\Gamma_{\mathrm{C}}}\dfrac{f_n^{\mathrm{C}}\sqrt{k_n^{(1)}\alpha_n}}{k_n^{(2)}}+\sum\limits_{n\in\Gamma_{\mathrm{G}}}\dfrac{f_n^{\mathrm{G}}\sqrt{k_n^{(1)}\alpha_n}}{k_n^{(3)}}}{\sum\limits_{n\in\Gamma_{\mathrm{C}}}\dfrac{\alpha_n f_n^{\mathrm{C}}}{k_n^{(2)}}+\sum\limits_{n\in\Gamma_{\mathrm{G}}}\dfrac{\alpha_n f_n^{\mathrm{G}}}{k_n^{(3)}}+E}\right)^2 \tag{16}$$

再将 v^{*} 代回 $\boldsymbol{W}'^{*}$ 即可获得最优带宽资源分配策略。但是，此时并不能保证限制条件 $W_n'\geqslant 1,\forall n\in\Gamma$ 一定成立。根据限制条件 C5' 可知，v 存在下界且下界满足 $v\geqslant v_{\min}=v^{*}$。由于 $-G\left(\boldsymbol{W}'^{*}\right)$ 关于 v 单调递增，为使 $-G\left(\boldsymbol{W}'^{*}\right)$ 最小化，v^{*} 应取满足限制条件的最小值。

$$-G\left(\boldsymbol{W}'^{*}\right)=\sqrt{v}\sum_{n\in\Gamma}\sqrt{\alpha_n k_n^{(1)}}-\sum_{n\in\Gamma_{\mathrm{C}}}\frac{f_n^{\mathrm{C}}k_n^{(1)}}{k_n^{(2)}}-\sum_{n\in\Gamma_{\mathrm{G}}}\frac{f_n^{\mathrm{G}}k_n^{(1)}}{k_n^{(3)}}-\frac{f_0^{\mathrm{C}}\theta_{\mathrm{B}}}{\beta_0}$$

基于上述结论，本文设计了基于拉格朗日对偶的迭代算法对带宽资源分配方案进行优化，如算法 1 所示。

算法 1 带宽资源分配算法

初始化 $\Gamma'=\Gamma$

1) 循环

2) for $n=1:\left|\Gamma'\right|$

3) if $W_n'^{*}<1$

4) 令 $W_n'^{*}=1$，将用户 n 从 Γ' 移除

5) end if

6) end for

7) 重新计算 $v_{\min}$ 并代入 $\boldsymbol{W}'^{*}$，返回步骤 2)

8) until $\Gamma'=\varnothing$ 或 $W_n'^{*}\geqslant 1,\forall n\in\Gamma'$

需要注意的是，带宽资源实际为离散变量，因此本文设计算法 2，将 $\boldsymbol{W}'^{*}$ 的连续变量转变为符合条件的离散变量。

算法 2 带宽资源分配方案转换算法

初始化 设置 $\boldsymbol{W}^{(1)}$，$W_n^{(1)}=W_n'-\left\lfloor W_n'\right\rfloor,\forall n\in\Gamma$；设置 $\boldsymbol{W}^{(2)}$，$W_n^{(2)}=\left\lfloor W_n'\right\rfloor,\forall n\in\Gamma$；设置集合 $\Gamma_{\mathrm{mid}}=\Gamma$

1) 循环

2) $n_{\min}=\operatorname{argmin}_{n\in\Gamma_{\mathrm{mid}}}W_n^{(1)}$

$n_{\max}=\operatorname{argmax}_{n\in\Gamma_{\mathrm{mid}}}W_n^{(1)}$

3) if $W_{n_{\min}}^{(1)}==0$

4) 将 $n_{\min}$ 从 Γ_{mid} 移除，返回步骤 2)

5) end if

6) if $1-W_{n_{\max}}^{(1)}\leqslant W_{n_{\min}}^{(1)}$

7) 令 $W_{n_{\min}}^{(1)}=W_{n_{\min}}^{(1)}-\left(1-W_{n_{\max}}^{(1)}\right)$，$W_{n_{\max}}^{(1)}=1$，将 $n_{\max}$ 从 Γ_{mid} 移除，返回步骤 2)

8) end if

9) if $1-W_{n_{\max}}^{(1)}>W_{n_{\min}}^{(1)}$

10) 令 $W_{n_{\max}}^{(1)}=W_{n_{\max}}^{(1)}+W_{n_{\min}}^{(1)}$，$W_{n_{\min}}^{(1)}=0$，将 $n_{\min}$ 从 Γ_{mid} 移除，返回步骤 2)

11) end if

12) until $\Gamma_{\text{mid}}=\varnothing$

13) 输出带宽资源分配方案 $\boldsymbol{W}=\boldsymbol{W}^{(1)}+\boldsymbol{W}^{(2)}$

给定任意可行的带宽资源分配方案 $\boldsymbol{W}$，由于限制条件 C3、C4 互相独立，优化问题 P3 可以根据处理器调度决策解耦为 2 个子问题 P6、P7，分别对 CPU 计算资源分配和 GPU 计算资源分配进行优化。具体地，CPU 计算资源分配优化问题 P6 可以表示为

$$\text{P6}:\min_{\boldsymbol{f}}\left\{-G_{\text{I}}\left(\boldsymbol{f}^{\text{C}}\right)-\frac{f_0^{\text{C}}\theta_{\text{B}}}{\beta_0}\right\}$$

$$\text{s.t. C3,C7':}f_n^{\text{C}}\geqslant f_{\min}^{\text{C}},\forall n\in\Gamma_{\text{C+}}$$

GPU 计算资源分配优化问题 P7 可以表示为

$$\text{P7}:\min_{\boldsymbol{f}}\left\{-G_2\left(\boldsymbol{f}^{\text{G}}\right)\right\}$$

$$\text{s.t. C4,C7'':}f_n^{\text{G}}\geqslant f_{\min}^{\text{G}},\forall n\in\Gamma_{\text{G+}}$$

定理 2　问题 P6、P7 是凸优化问题。

证明　同定理 1。

根据拉格朗日对偶理论，问题 P6、P7 的拉格朗日函数分别构造为

$$L^{(2)}\left(\boldsymbol{f}^{\text{C}},y_1\right)=-G_{\text{I}}\left(\boldsymbol{f}^{\text{C}}\right)-\frac{f_0^{\text{C}}\theta_{\text{B}}}{\beta_0}+y_1\left(\sum_{n\in\Gamma_{\text{C}}}f_n^{\text{C}}+f_0^{\text{C}}-F_{\text{C}}\right)\tag{17}$$

$$L^{(3)}\left(\boldsymbol{f}^{\text{G}},y_2\right)=-G_2\left(\boldsymbol{f}^{\text{G}}\right)+y_2\left(\sum_{n\in\Gamma_{\text{G}}}f_n^{\text{G}}-F_{\text{G}}\right)\tag{18}$$

与之对应的拉格朗日对偶函数可以分别表示为

$$L_{\text{D}}^{(2)}\left(y_1\right)=\min_{\boldsymbol{f}^{\text{C}}}L^{(2)}\left(\boldsymbol{f}^{\text{C}},y_1\right)$$

$$L_{\text{D}}^{(3)}\left(y_2\right)=\min_{\boldsymbol{f}^{\text{G}}}L^{(3)}\left(\boldsymbol{f}^{\text{G}},y_2\right)$$

由此引出的拉格朗日对偶问题分别表示为 $\max\limits_{y_1}L_{\text{D}}^{(2)}\left(y_1\right)$ 和 $\max\limits_{y_2}L_{\text{D}}^{(3)}\left(y_2\right)$。其中，$y_1$、$y_2$ 分别为与 C3、C4 相关的拉格朗日乘子，且 C7' 和 C7'' 被暂时松弛。

针对问题 P6，根据 $\dfrac{\partial L^{(2)}}{\partial f^{\text{C}*}}=0$，可得 $y_1=\dfrac{\theta_{\text{B}}}{\beta_0}>0$，$f_n^{\text{C}*}=\dfrac{W_n}{\alpha_n}\left(\sqrt{\dfrac{k_n^{(1)}k_n^{(2)}\beta_0}{\theta_{\text{B}}}}-k_n^{(2)}\right)$。根据 Karush-Kuhn-Tucker 条件中的松弛互补条件可得 $f_0^{\text{C}*}=F_{\text{C}}-\sum\limits_{n\in\Gamma_{\text{C}}}f_n^{\text{C}*}$。考虑到限制条件 C7'，可得

$$f_n^{\text{C}*}=\max\left[f_{\min}^{\text{C}},\frac{W_n}{\alpha_n}\left(\sqrt{\frac{k_n^{(1)}k_n^{(2)}\beta_0}{\theta_{\text{B}}}}-k_n^{(2)}\right)\right]$$

$$f_0^{\text{C}*}=\max\left[f_{\min}^{\text{C}},F_{\text{C}}-\sum\nolimits_{n\in\Gamma_{\text{C}}}f_n^{\text{C}*}\right]$$

针对问题 P7，根据 $\dfrac{\partial L^{(3)}}{\partial \boldsymbol{f}^{\text{G}*}}=0$ 可得 $f_n^{\text{G}*}=\dfrac{W_n}{\alpha_n}\left(\sqrt{\dfrac{k_n^{(1)}k_n^{(3)}}{y_2}}-k_n^{(3)}\right)$。将 $\boldsymbol{f}^{\text{G}*}$ 代入 $L^{(3)}$ 可得 $L_{\text{D}}{}^{(3)}$，由于 $\max\limits_{y_2}L_{\text{D}}{}^{(3)}\left(y_2\right)$ 也是凸问题，根据 $\dfrac{\partial L_{\text{D}}^{(3)}}{\partial y_2^*}=0$ 可得

$$y_2^*=\left(\frac{\sum\limits_{n\in\Gamma_{\text{G}}}\dfrac{W_n\sqrt{k_n^{(1)}k_n^{(3)}}}{\alpha_n}}{F_{\text{G}}+\sum\limits_{n\in\Gamma_{\text{G}}}\dfrac{k_n^{(3)}W_n}{\alpha_n}}\right)^2$$

然而，此时并不能保证限制条件 C7'' 一定成立。根据限制条件 C4 可知，y_2 存在下界且满足 $y_2\geqslant y_{2,\min}=y_2^*$。由于 $-G_2\left(\boldsymbol{f}^{\text{G}*}\right)=\sqrt{y_2}\sum\limits_{n\in\Gamma_{\text{G}}}\dfrac{W_n\sqrt{k_n^{(1)}k_n^{(3)}}}{\alpha_n}-\sum\limits_{n\in\Gamma_{\text{G}}}\dfrac{W_nk_n^{(1)}}{\alpha_n}$ 关于 y_2 单调递增，为使 $-G_2\left(\boldsymbol{f}^{\text{G}*}\right)$ 最小化，y_2^* 应取满足限制条件的最小值。基于上述结论，本文设计了基于拉格朗日对偶的迭代算法对计算资源分配方案进行优化，如算法 3 所示。

算法 3　计算资源分配算法

初始化　$\Gamma_{\text{C}}'=\Gamma_{\text{C}}$，$\Gamma_{\text{G}}'=\Gamma_{\text{G}}$

1) for $n=1:\left|\Gamma_{\text{C}}'\right|$

2)　if $f_n^{\text{C}*}<f_{\min}^{\text{C}}$

3)　　令 $f_n^{\text{C}*}=f_{\min}^{\text{C}}$，将 n 从 Γ_{C}' 移除

4)　end if

5) end for

6) if $\sum\limits_{n\in\Gamma_{\text{C}}}f_n^{\text{C}*}>F_{\text{C}}-f_{\min}^{\text{C}}$

7)　令 $f_n^{\text{C}*}=\dfrac{f_n^{\text{C}*}\left[F_{\text{C}}-\left(\left|\Gamma_{\text{C}}\right|-\left|\Gamma_{\text{C}}'\right|+1\right)f_{\min}^{\text{C}}\right]}{\sum\limits_{n\in\Gamma_{\text{C}}'}f_i^{\text{C}*}}$

8) end if

9) 循环

10) for $n=1:\left|\Gamma_{\mathrm{G}}'\right|$

11) if $f_n^{\mathrm{G}*}<f_{\min}^{\mathrm{G}}$

12) 令 $f_n^{\mathrm{G}*}=f_{\min}^{\mathrm{G}}$，并将用户 n 从 Γ_{G}' 移除

13) end if

14) end for

15) 重新计算 $y_{2,\min}$ 并代入 $\boldsymbol{f}^{\mathrm{G}*}$，返回步骤 10)

16) until $\Gamma_{\mathrm{G}}'=\varnothing$ 或 $f_n^{\mathrm{G}*}\geqslant f_{\min}^{\mathrm{G}},\forall n\in\Gamma_{\mathrm{G}}'$

注意到上述获得的计算资源分配方案是基于假设 $\boldsymbol{\lambda}_0=[1,0]$ 的。当 $\boldsymbol{\lambda}_0=[0,1]$ 时，计算资源分配方案可以按照类似的方法进行优化，本文不再详细阐述。基于上述带宽资源分配优化方法与计算资源分配优化方法，本文设计了资源联合分配（JRA, joint resources allocation）算法，如算法 4 所示。

算法 4 资源联合分配算法

初始化

$$\boldsymbol{f}_n(1)=\begin{cases}\left[\dfrac{F_{\mathrm{C}}}{\sum\limits_{n\in N_{\mathrm{C}+}}\lambda_n^{\mathrm{C}}},0\right],\forall n\in\Gamma_{\mathrm{C}+}\\ \left[0,\dfrac{F_{\mathrm{G}}}{\sum\limits_{n\in N_{\mathrm{G}+}}\lambda_n^{\mathrm{G}}}\right],\forall n\in\Gamma_{\mathrm{G}+}\end{cases}$$

其中，$\Gamma_{\mathrm{C}+}=\left\{n\in\Gamma_+\middle|\boldsymbol{\lambda}_n=[1,0]\right\}$，$\Gamma_{\mathrm{G}+}=\{n\in\Gamma_+\left|\boldsymbol{\lambda}_n=[0,1]\right\}$。设置初始迭代次数 $t=1$，迭代精度为 π_1 和 π_2

1) 循环

2) 给定 $\boldsymbol{f}(t)$，根据算法 1 求解带宽资源分配方案 $\boldsymbol{W}'(t)$

3) 给定 $\boldsymbol{W}'(t)$，根据算法 3 求解计算资源分配方案 $\boldsymbol{f}(t+1)$

4) $\boldsymbol{W}'^{*}=\boldsymbol{W}'(t)$，$\boldsymbol{f}^{*}=\boldsymbol{f}(t)$，$t=t+1$

5) until $\Delta\boldsymbol{f}(t)=\left\|\boldsymbol{f}(t)-\boldsymbol{f}(t-1)\right\|_2\leqslant\pi_1$ 或 $\Delta\boldsymbol{W}(t)=\left\|\boldsymbol{W}(t)-\boldsymbol{W}(t-1)\right\|_2\leqslant\pi_2$

6) 输出资源分配方案 $\boldsymbol{W}'^{*}$、$\boldsymbol{f}^{*}$

4.2 处理器调度优化问题

根据限制条件 C2 可知，与普通类型任务相关联的虚拟机只能调用 CPU 处理器处理计算任务，由此可得 $\lambda_n^{*}=[1,0],\forall n\in\Gamma_0$。由于处理器调度策略可直接决定任务处理时延，从而影响系统效用，本文综合考虑任务处理时延和权重因子，设计了基于任务处理时延 T_n 的 PSRA 算法对处理器调度问题（等价于问题 P2）进行优化，如算法 5 所示。具体地，加权任务处理时延定义为

$$T_n=\begin{cases}\theta_{\mathrm{M}}\eta_n d_n^{\mathrm{exe}},\forall n\in\Gamma\\ \theta_{\mathrm{B}}d_0^{\mathrm{exe}},n=0\end{cases}$$

算法 5 处理器调度与资源分配联合优化算法

初始化 随机生成处理器调度方案 $\boldsymbol{\lambda}(1)$，执行 JRA 算法获得 $\boldsymbol{f}(1)$ 和 $\boldsymbol{W}'(1)$，计算相应系统效用 $G(\boldsymbol{\lambda}(1),\boldsymbol{f}(1),\boldsymbol{W}'(1))$；设置初始迭代次数 $t=1$，迭代精度 π_3；$\Gamma_1=\{\Gamma_+\setminus\Gamma_0\}$

1) 循环

2) 根据 $\boldsymbol{\lambda}(t)$、$\boldsymbol{f}(t)$ 计算 $T_n,\forall n\in\Gamma_1$，令 $\Gamma_1'=\Gamma_1$

3) 循环

4) $\boldsymbol{\lambda}=\boldsymbol{\lambda}(t)$，令 $n'=\operatorname{argmax}_{n\in\Gamma_1'}T_n$，$\lambda_{n'}^{\mathrm{C}}=1-\lambda_{n'}^{\mathrm{C}}$，$\lambda_{n'}^{\mathrm{G}}=1-\lambda_{n'}^{\mathrm{G}}$

5) 执行 JRA 算法获得 $\boldsymbol{f}$、$\boldsymbol{W}'$，计算 $G(\boldsymbol{\lambda},\boldsymbol{f},\boldsymbol{W}')$

6) if $G(\boldsymbol{\lambda},\boldsymbol{f},\boldsymbol{W}')>G(\boldsymbol{\lambda}(t),\boldsymbol{f}(t),\boldsymbol{W}'(t))$

7) 令 $t=t+1$，$\boldsymbol{\lambda}(t)=\boldsymbol{\lambda}$，$\boldsymbol{f}(t)=\boldsymbol{f}$，$\boldsymbol{W}'(t)=\boldsymbol{W}'$，返回步骤 2)

8) else

9) 将 n' 从 Γ_1' 移除，返回步骤 4)

10) end if

11) until $\Gamma_1'=\varnothing$

12) until $\Gamma_1'=\varnothing$ 或 $|G(\boldsymbol{\lambda}(t),\boldsymbol{f}(t),\boldsymbol{W}'(t))-G(\boldsymbol{\lambda}(t-1),\boldsymbol{f}(t-1),\boldsymbol{W}'(t-1))|\leqslant\pi_3$

13) 执行算法 2，将 $\boldsymbol{W}'(t)$ 转换为离散变量 $\boldsymbol{W}(t)$

14) 输出 $\boldsymbol{\lambda}^{*}=\boldsymbol{\lambda}(t)$，$\boldsymbol{f}^{*}=\boldsymbol{f}(t)$，$\boldsymbol{W}^{*}=\boldsymbol{W}(t)$

4.3 算法复杂度分析

PSRA 算法的计算复杂度主要来自处理器调度方案的迭代优化，假设迭代次数为 I_1。处理器调度方案每次迭代过程中的计算量主要来自 JRA 算法对带宽资源及计算资源的迭代优化，假设迭代次数为 I_2。算法 1 和算法 3 的最大循环次数分别为 N 和 $N+1$。因此，JRA 算法的计算复杂度可以表

示为 $O\left(I_2\left(2N+1\right)\right)$，处理器调度方案迭代优化的计算复杂度可以表示为 $O\left(I_2\left(I_1+1\right)\left(2N+1\right)\right)$，算法 PSRA 的总计算复杂度则可以表示为 $O\left(I_2\left(I_1+1\right)\left(2N+1\right)+N\right)$。

5　仿真结果与分析

本节对基于异构计算的 BMEC 系统性能进行了仿真测验和分析，并验证分析了所提算法相较于其他基准算法的优越性。首先描述了仿真参数设置，然后相继展示了所提算法的收敛性、优越性以及系统效用与用户数量之间的关系。

5.1　仿真参数设置

本文考虑的 BMEC 系统中包含 5 个 MEC 服务器，每个服务器服务的用户数量为 15，假设所有服务器均为区块生成节点，且每个节点轮流负责新区块的生成。本文考虑移动用户上行信道传输速率为随机变量[17]，假设每个用户在单位子信道上的上行传输速率 r_n^0 均匀分布于 $[10^6, 2\times10^6]$ bit/s。本文假设每个服务器装配有 6 个主频为 2.4 GHz 的 4 核 8 线程 CPU 处理器和一个主频为 1 038 MHz 的 768 核 GPU 处理器[25, 28]，即 $F_{\mathrm{C}}=57.6$ Gcycle/s，$F_{\mathrm{G}}=797$ Gcycle/s。本文考虑每个服务器可处理五类应用，每类应用对应一种计算任务，每个用户随机产生其中一种任务，五类任务对应的应用信息如表 2 所示。此外，区块链应用为特殊应用且 GPU 利用效率为 0.5，根据所有 BP 节点轮流生成新区块的特征，本文假设区块链任务计算量满足概率 $\Pr\left(\beta_0^{\mathrm{req}}\right)=\dfrac{1}{M}$，$\Pr\left(\beta_0^{\mathrm{prep}}\right)=\dfrac{M-1}{2M}$，$\Pr\left(\beta_0^{\mathrm{rep}}\right)=\dfrac{M-1}{2M}$。其他仿真参数设置如表 3 所示。

表 2　　应用信息

序号	类型	任务量/KB	CPU 计算量/Gcycle	GPU 利用效率	时延敏感度
1	0	100	0.25	None	0.5
2	0	100	0.3	None	0.6
3	1	200	1	0.5	0.7
4	1	200	1	0.7	0.8
5	1	300	1.5	0.9	0.9

为了证实所提 PSRA 算法的性能和异构计算架构引入的效果，本文将其与以下基准算法进行比较。

表 3　　仿真参数设置

参数	值
E	50
θ_{M}	0.4
θ_{B}	0.6
$f_{\min}^{\mathrm{C}}$ /(Gcycle·s^{-1})	0.5
$f_{\min}^{\mathrm{G}}$ /(Gcycle·s^{-1})	1
S_B /KB	5 000
ϖ /KB	20
ϕ /Gcycle	0.001
φ /Gcycle	0.01
g	0.9

1) 穷搜算法。列举所有处理器调度方案，找出使系统效用最高的处理器调度方案，资源分配采用 JRA 算法。

2) 贪婪算法。凡属于特殊应用类型的任务，均调用 GPU 处理器执行相关计算，资源分配采用 JRA 算法。

3) 资源平均算法。计算资源和带宽资源平均分配给每个用户，处理器调度策略基于 PSRA 算法进行优化。

由于贪婪算法和资源平均算法未充分挖掘不同计算任务所存在的并行性，不能实现计算任务与处理器之间的匹配优化和异构计算资源分配优化，可视为未引入异构计算的基准算法。

5.2　算法收敛性

图 3 和图 4 分别表示用户数量为 15 时，JRA 算法关于计算资源分配方案和带宽资源分配方案的收敛性能。可以看出，2 种资源分配方案均在有限次迭代以后达到收敛状态。

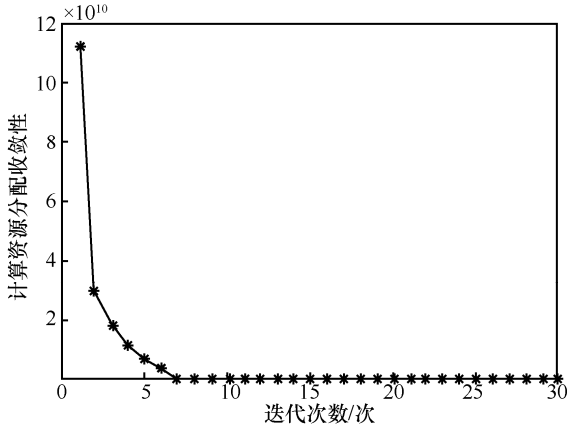

图 3　JRA 算法收敛性（计算资源分配方案）

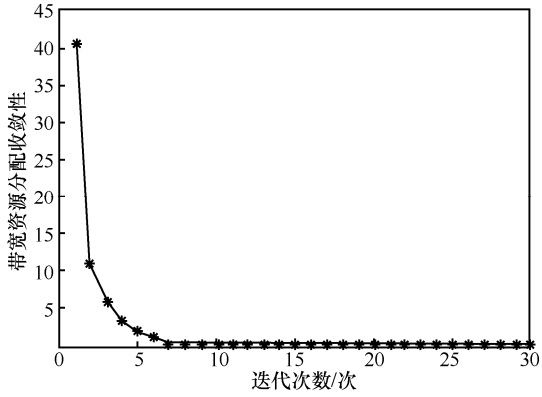

图 4　JRA 算法收敛性（带宽资源分配方案）

图 5 表示 PSRA 算法在不同用户数量下的收敛性能。从图 5 可以看出，随着迭代次数的增加，所提算法在不同用户数量的场景下均逐渐收敛。此外，当用户数量较少时，所提 PSRA 算法的求解空间较小，可以更快收敛，而用户数量较大时，所提 PSRA 算法的收敛速度降低。如图 5 所示，当 $N=5$ 时，所提 PSRA 算法在平均迭代 7 次后达到收敛状态。当 $N=15$ 时，相比于 $N=5$ 的场景，PSRA 算法需要更多迭代轮次达到收敛状态。因此，当平均迭代次数为 7 次时，由于 PSRA 算法在 $N=15$ 的场景下尚未达到收敛，因此该轮次下的系统效用取值并不能反映系统效用随用户数量变化的相对趋势。当平均迭代次数为 12 时，所提 PSRA 算法在 $N=15$ 的场景下达到收敛，此时该场景下的最终可达系统效用值高于 $N=5$ 的场景。

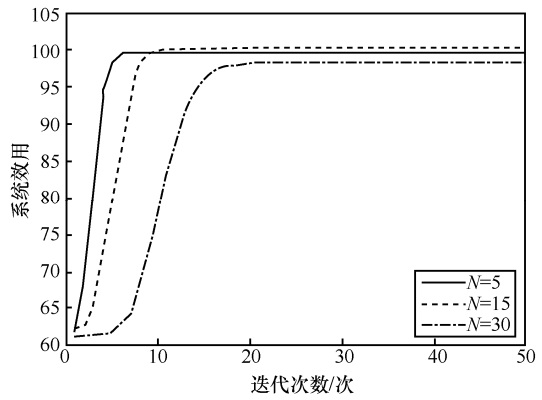

图 5　PSRA 算法收敛性

本文定义的系统效用是由所有用户任务处理效率与系统区块生成效率的加权和决定的，系统效用函数不随用户数量增长呈现单调性。系统效用的取值受到带宽资源、计算资源、用户数量、用户任务处理效率与系统区块生成效率的权值等因素的综合影响。具体来说，随着用户数量的增加，系统效用函数中被累加的任务处理效率的数量随之增加；但是，由于随着用户数量的增加，每个用户可用的平均带宽资源、计算资源减少，区块生成任务可用计算资源也减少，导致单个用户任务处理效率及系统区块生成效率逐渐降低。

5.3　算法性能

图 6 和图 7 分别为 $N=5$ 和 $N=15$ 场景下 PSRA 算法、贪婪算法、资源平均算法的性能曲线。由图 6 和图 7 可以看出，在不同的用户数下，PSRA 算法性能优于贪婪算法和资源平均算法。这主要是因为，针对不同种类计算任务的处理需求，PSRA 算法通过优化处理器调度和资源分配，可以有效提高用户任务处理效率和系统区块生成效率，进而提升系统效用。

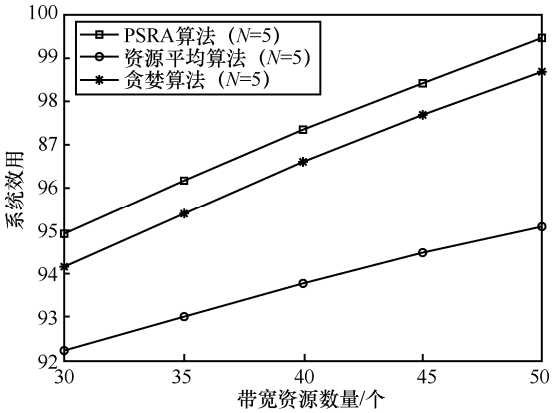

图 6　PSRA 算法、贪婪算法、资源平均算法对比（$N=5$）

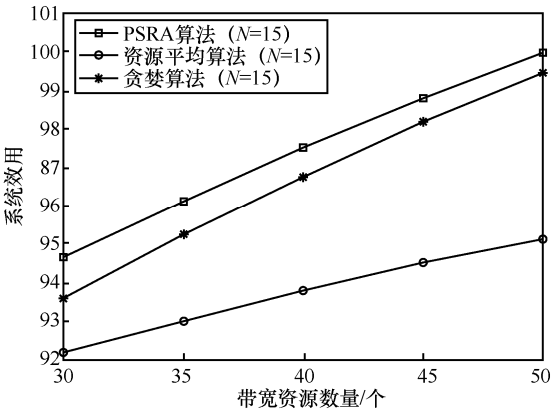

图 7　PSRA 算法、贪婪算法、资源平均算法对比（$N=15$）

贪婪算法的优势在于可以根据任务需求优化带宽资源和计算资源分配方案，但是不能根据任务处理时延优化处理器调度方案；资源平均算法的优势在于可以根据任务处理时延优化处理器调度方

案，但是不能根据任务需求优化带宽资源和计算资源分配方案。当带宽资源较少时，带宽资源平均分配方案与 JRA 算法优化的带宽资源分配方案相对接近，此时带宽资源分配不均衡程度较低，优化资源分配方案所带来的系统效用增益较低。因此，当带宽资源较少时，资源平均算法的性能相对较好。但是随着带宽资源数量的增加，带宽资源平均分配方案导致的资源分配不均衡程度越来越高，通过优化资源分配可以获得较高的系统效用增益。因此，随着带宽资源的增加，资源平均算法与可以优化计算资源和带宽资源分配方案的 PSRA 算法、贪婪算法相比，其算法性能会逐渐变差。

5.4　系统效用与用户数量的关系

图 8 给出了 PSRA 算法在带宽资源为 50 时，系统效用与用户数量之间的变化趋势。由图 8 可以看出，随着用户数量的增加，系统效用函数呈现先上升后下降的趋势。当带宽资源数量为 50，用户任务处理效率和系统区块生成效率权重因子分别为 0.4 和 0.6 的情况下，$N=15$ 时的系统效用高于 $N=5$ 和 $N=30$ 的场景。然而，由于系统效用函数受到多重因素的影响，该相对趋势并不能体现系统的性能特征，系统效用不会随着用户数量的增长呈现出单调性特征。

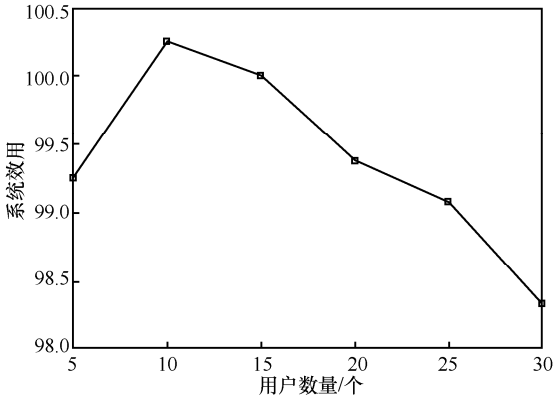

图 8　系统效用与用户数量的关系

6　结束语

本文提出了一种基于异构计算的 BMEC 系统模型，该模型通过联合优化异构处理器调度与资源分配，解决了联合处理移动应用计算任务和区块链计算任务时面临的高时延和资源利用率低下等问题。为了实现不同计算任务高效并行性处理，以系统效用最大化为目标，本文研究了异构处理器调度与资源分配的联合优化问题，并提出了基于拉格朗日对偶理论的处理器调度与资源分配联合优化算法进行求解。仿真结果表明，本文所提的 PSRA 算法优于其他基准算法，可以有效提升基于异构计算的 BMEC 系统效用。

参考文献：

[1] EVANS D. The Internet of things: how the next evolution of the Internet is changing everything[R]. Cisco, (2011-04)[2020-07-16].

[2] 张平, 牛凯, 田辉, 等. 6G 移动通信技术展望[J]. 通信学报, 2019, 40(1): 141-148.
ZHANG P, NIU K, TIAN H, et al. Technology prospect of 6G mobile communications[J]. Journal on Communications, 2019, 40(1):141-148.

[3] ZHANG P, ZHANG J H, QI Q, et al. Ubiquitous-X: constructing the future 6G networks[J]. SCIENTIA SINICA Informationis, 2020, 50(6): 913-930.

[4] ZHANG P, XU X D, QIN X Q, et al. Evolution toward artificial intelligence of things under 6G Ubiquitous-X[J]. Journal of Harbin Institute of Technology, 2020, 27(3): 116-135.

[5] HU Y C, PATEL M, SABELLA D, et al. Mobile edge computing-a key technology towards 5G[R]. ETSI White Paper, (2015-09)[2020-07-16].

[6] PHAM Q, LEANH T, TRAN N H, et al. Decentralized computation offloading and resource allocation for mobile-edge computing: a matching game approach[J]. IEEE Access, 2018, 6: 75868-75885.

[7] TRAN T X, POMPILI D. Joint task offloading and resource allocation for multi-server mobile-edge computing networks[J]. IEEE Transactions on Vehicular Technology, 2019, 68(1): 856-868.

[8] LI S L, TAO Y Z, QIN X Q, et al. Energy-aware mobile edge computation offloading for IoT over heterogenous networks[J]. IEEE Access, 2019, 7: 13092-13105.

[9] YANG K, JIA X H, REN K. Secure and verifiable policy update outsourcing for big data access control in the cloud[J]. IEEE Transactions on Parallel and Distributed Systems, 2015, 26(12): 3461-3470.

[10] RAWAT D B. Fusion of software defined networking, edge computing, and blockchain technology for wireless network virtualization[J]. IEEE Communications Magazine, 2019, 57(10): 50-55.

[11] YANG R Z, YU F R, SI P B, et al. Integrated blockchain and edge computing systems: a survey, some research issues and challenges[J]. IEEE Communications Surveys & Tutorials, 2019, 21(2): 1508-1532.

[12] HERBAUT N, NEGRU N. A model for collaborative blockchain-based video delivery relying on advanced network services chains[J]. IEEE Communications Magazine, 2017, 55(9): 70-76.

[13] KANG J W, YU R, HUANG X Y, et al. Enabling localized peer-to-peer electricity trading among plug-in hybrid electric vehicles using consortium blockchains[J]. IEEE Transactions on Industrial Informatics, 2017, 13(6): 3154-64.

[14] MENDKI P. Blockchain enabled IoT edge computing[C]//Proceedings of the 2019 International Conference on Blockchain Technology. New York: ACM Press, 2019: 66-69.

[15] JIAO Y T, WANG P, NIYTAO D, et al. Auction mechanisms in cloud/fog computing resource allocation for public blockchain networks[J]. IEEE Transactions on Parallel and Distributed Systems, 2019, 30(9): 1975-1989.

[16] FENG J, YU F R, PEI Q, et al. Joint optimization of radio and computational resources allocation in blockchain-enabled mobile edge computing systems[J]. IEEE Transactions on Wireless Communications, 2020, 19(6): 4321-4334.
[17] GUO F X, YU F R, ZHANG H L, et al. Adaptive resource allocation in future wireless networks with blockchain and mobile edge computing[J]. IEEE Transactions on Wireless Communications, 2020, 19(3): 1689-1703.
[18] QIU X Y, LIU L B, CHEN W H, et al. Online deep reinforcement learning for computation offloading in blockchain-empowered mobile edge computing[J]. IEEE Transactions on Vehicular Technology, 2019, 68(8): 8050-8062.
[19] 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.
[20] GASTER B, HOWES L, KAELI D R, et al. Heterogeneous computing with OpenCL: revised OpenCL 1.2 edition[M]. San Francisco: Morgan Kaufmann Press, 2012.
[21] WANG H L, XIAO B, WANG L, et al. CHCF: a cloud-based heterogeneous computing framework for large-scale image retrieval[J]. IEEE Transactions on Circuits and Systems for Video Technology, 2015, 25(12): 1900-1913.
[22] ILAGER S, WANKAR R, KUNE R, et al. GPU PaaS computation model in Aneka cloud computing environment [C]//SmartData-2019. Saarland: DBLP, 2019: 19-40.
[23] XIAO B, WANG H L, WU J, et al. A multi-grained parallel solution for HEVC encoding on heterogeneous platforms[J]. IEEE Transactions on Multimedia, 2019, 21(12): 2997-3009.
[24] LEE W, PHAN R C, GOI B, et al. Parallel and high speed hashing in GPU for telemedicine applications[J]. IEEE Access, 2018, 6: 37991-38002.
[25] IYER S G, DIPAKUMAR P A. GPU and CPU accelerated mining of cryptocurrencies and their financial analysis[C]//2018 2nd International Conference on I-SMAC. Piscataway: IEEE Press, 2018: 599-604.
[26] EKBOTE B, HIRE V, MAHAJAN P, et al. Blockchain based remittances and mining using CUDA[C]//2017 International Conference on Smart Technologies for Smart Nation. Piscataway: IEEE Press, 2017: 908-911.
[27] MORISHIMA S, MATSUTANI H. Accelerating blockchain search of full nodes using GPUs[C]//2018 26th Euromicro International Conference on Parallel, Distributed and Network-based Processing. Piscataway: IEEE Press, 2018: 244-248.
[28] MORISHIMA S, MATSUTANI H. Acceleration of anomaly detection in blockchain using in-GPU cache[C]//2018 IEEE International Conference on Parallel & Distributed Processing with Applications, Ubiquitous Computing & Communications, Big Data & Cloud Computing, Social Computing & Networking, Sustainable Computing & Communications (ISPA/IUCC/BDCloud/SocialCom/SustainCom). Piscataway: IEEE Press, 2018: 244-251.
[29] BOYD S, VANDENBERGHE L. Convex optimization[M]. Cambridge: Cambridge University Press, 2014.

[作者简介]

张平（1959− ），男，陕西汉中人，中国工程院院士，北京邮电大学教授、博士生导师，主要研究方向为先进移动通信系统。

李世林（1992− ），男，河南许昌人，北京邮电大学博士生，主要研究方向为移动边缘计算、区块链技术、通信与计算资源管理与优化。

刘宜明（1993− ），女，河南商丘人，博士，北京邮电大学在站博士后，主要研究方向为下一代无线网络理论与关键技术、基于区块链的智能边缘网络可信协作策略、可信边缘协同计算关键技术等。

秦晓琦（1988− ），女，北京人，博士，北京邮电大学讲师、硕士生导师，主要研究方向为下一代无线网络基础理论及性能分析、基于信息时效性的通信与计算融合机理研究、面向动态场景的主动认知技术等。

许晓东（1980− ），男，山东沂水人，博士，北京邮电大学教授、博士生导师，主要研究方向为无线组网基础理论、通信计算融合和试验系统研发等。