基于智能化用户协作的边缘计算任务卸载与资源分配优化

李贤[1]，毕宿志[1,2]，曾泓儒[1]，林彬[1]，林晓辉[1]
（1. 深圳大学电子与信息工程学院，广东 深圳 518060；2. 鹏城实验室宽带通信研究部，广东 深圳 518066）

摘　要：为了解决移动边缘计算网络中计算资源日益紧缺的问题，设计了一种基于用户协作的边缘计算资源分配机制，充分利用用户之间的空闲计算资源，有效提升系统整体的数据处理性能。以最大化用户的效用函数为目标，将目标优化问题建模为一个关于用户任务卸载决策和本地计算通信资源的联合优化问题，并结合深度学习技术和凸优化理论，提出了一种混合深度学习−优化算法对目标问题进行求解。仿真结果表明，相较于对比算法，所提算法能使用户的效用提升至少 85.4%，并能在亚秒级的时间内实现用户效用的近似最优化。

关键词：移动边缘计算；效用最大化；凸优化；深度学习
中图分类号：TN92
文献标志码：A
doi: 10.11959/j.issn.2096−3750.2022.00303

Collaborative task offloading and resource allocation optimization for intelligent edge devices

LI Xian[1], BI Suzhi[1,2], ZENG Hongru[1], LIN Bin[1], LIN Xiaohui[1]
1. College of Electronics and Information Engineering, Shenzhen University, Shenzhen 518060, China
2. Pengcheng Laboratory, Broadband Communication Research Department, Shenzhen 518066, China

Abstract: In order to deal with the increasingly scarce computing resources, a cooperative edge computing scheme was proposed, which makes full use of the idle resources among users to improve the overall data processing performance. To maximize the user utility, the target problem was formulated as an MINLP (mixed integer non-linear programming), and a learning-optimization-integrated method was proposed to jointly optimize the resource allocation and user offloading decisions. Simulation results show that the proposed scheme can produce a near-optimal solution in sub-second and effectively improve the system utility at least 85.4% compared to the considered benchmark methods.

Key words: mobile edge computing, utility maximization, convex optimization, reinforcement learning

收稿日期：2022−08−12；**修回日期：**2022−09−23
通信作者：毕宿志，bsz@szu.edu.cn
基金项目：国家重点研发计划（No.2019YFB1803305）；国家自然科学基金资助项目（No.61871271，No.62271325）；鹏城实验室宽带通信研究部重点研究计划；广东省教育厅科技重点专项（No.2020ZDZX3050）；广东省基础与应用基础研究基金资助项目（No.2022A1515011219，No.2022A1515010973）；深圳市科创委基础研究项目（No.20220810142637001，No.JCYJ20210324093011030，No.JCYJ20190808120415286）；智慧城市物联网国家重点实验室（澳门大学）开放课题（No.SKL-IoTSC(UM)-2021-2023/ORPF/A03/2022）

Foundation Items: The National Key Research and Development Program (No.2019YFB1803305), The National Natural Science Foundation of China (No.61871271, No.62271325), The Major Key Project of PCL Department of Broadband Communication, The Key Project of Department of Education of Guangdong Province (No.2020ZDZX3050), Guangdong Basic and Applied Basic Research Foundation (No.2022A1515011219, No.2022A1515010973), The Shenzhen Science and Technology Program (No.20220810142637001, No.JCYJ20210324093011030, No.JCYJ20190808120415286), The Open Research Project Programme of the State Key Laboratory of Internet of Things for Smart City (University of Macau) (No.SKL-IoTSC(UM)-2021-2023/ORPF/A03/2022)

0 引言

5G 通信技术的快速发展与应用催生了实时在线游戏、虚拟现实、自动驾驶等一系列新兴智能应用[1-5]。这些应用通常具有数据计算量大、处理时延要求高等特点。然而，受限于制造成本以及自身硬件条件（如计算能力、电池容量、存储容量等）等因素，移动终端设备仅仅依靠自身的计算资源往往难以满足这些新兴智能应用在时延、计算速率等方面的严苛要求。作为提高网络数据处理性能的重要技术手段之一，移动边缘计算（MEC, mobile edge computing）近年来受到了国内外研究机构的热切关注[6-10]。通过将计算、存储、通信等网络资源直接部署至网络边缘，MEC 可在边缘服务器端直接受理用户终端的数据处理请求。相较于传统云计算，MEC 中的终端用户更靠近边缘服务器。因此，MEC 采用较低的发射功率即可覆盖终端用户，其网络频谱可复用程度较高，能有效提升频谱资源的利用率。同时，由于通信距离短，终端用户与边缘服务器间的信号传输路径衰落较少，而且通常采用端到端直接传输的方式，避免了传统云计算系统中复杂的路由转发过程，有效地降低了数据传输时延。频谱资源利用率高和实时性强这两点特性使得边缘计算尤为契合 5G 智能应用对低时延、大连接服务的需求。

在 MEC 系统中，用户终端可根据自身需求，选择性地将数据卸载至边缘服务器进行处理。常用的数据卸载方式有二进制卸载（binary offloading）[11-13]和部分卸载（partial offloading）[14-16]两种。其中，二进制卸载机制适用于数据任务难以分割处理的情况，其要求用户要么在本地完成整个数据任务块的处理；要么将数据上传至边缘端进行边缘处理。而部分卸载则适用于数据任务可分割处理的情况，其允许用户上传部分数据至边缘端，随后整合本地和边缘服务器的输出作为最后的计算结果。

受益于近年来大规模集成电路的快速发展，移动终端设备（如笔记本计算机、平板计算机和智能手机）的性能得到了极大的提升，其数据处理能力甚至可以和 10 年前的微型服务器相媲美。相应地，5G 网络边缘拥有极为丰富的计算资源。然而，5G 用户终端通常具备高度的异构特性：不同用户终端的设备性能不同，在不同时间各终端所需处理的任务量也不一样。部分高性能设备的计算通信等能力较强，可能会出现自身数据处理完之后设备空转的问题，极大地降低了网络的资源利用率。因此，为最大化利用网络资源提升系统的数据处理能力，整合位于网络边缘的空闲计算资源，研究面向协作的移动边缘计算成为近年来国内外研究的焦点[17-26]。例如，对于部分卸载机制，文献[17]研究了面向边缘计算的多用户数据协作计算方法，探索了不同用户间基于数据共享的任务分割卸载方案。针对无线供电应用场景，文献[18]从协作通信的角度出发，研究了中继边缘计算的任务协作传输与资源分配优化。文献[19]则将工作扩展到中继节点同时参与协作通信与任务计算的情况，通过优化用户的传输功率、传输时间以及任务卸载数量等决策，最大化系统的数据处理速率。通过利用协作用户的 CPU 状态预测信息，文献[20]为系统设计了能量有效的任务协作处理离线策略（off-line policy），优化了用户任务数据在多个工作时隙中的分配方法。文献[21]则将非正交多址接入（NOMA, non-orthogonal multiple access）技术引入协作边缘计算系统，并分析了任务卸载中断概率、数据传输吞吐量，以及能量效率等系统性能。在部分实际智能应用中，为了保证任务的顺利处理，用户在处理数据时可能需要保持整段数据的完整性。例如，在图像识别应用中，用户需要在本地对整张图片进行处理，或将其完全卸载至边缘服务器端进行计算。因此，二进制卸载机制下协作计算系统优化设计的相关研究也备受关注。例如，文献[23]研究了边缘服务器协助用户设备进行计算的场景，根据通信信道状态优化了各用户的任务卸载决策和资源分配。文献[24]考虑了多边缘服务器协作处理用户数据的场景，通过优化服务器之间的数据缓存及任务调度方案，在给定任务处理时间和缓存空间约束的条件下，优化了所有用户的任务处理能耗和时延。通过模拟人类活动中的利他行为，文献[25]量化了社会信任值，并将其用于二进制卸载机制下用户间的计算协作方案设计。文献[26]则提出了一种基于拍卖的用户协作机制，通过在边缘服务器端综合评估来自各用户的投标信息（如任务的开始时间、任务的工作负载以及任务的效用）制定用户的协作决策，实现系统整体效用与资源损耗之间差值的最大化。

在二进制卸载机制下，系统的优化设计涉及整型变量（如任务卸载）与连续变量（如资源分配）的联合优化，相应的目标问题往往是 NP 难的组合

非线性规划问题。现有工作大多采用性能次优的启发式算法（如文献[25-26]）或复杂度较高的迭代优化算法（如文献[23-24]）对其进行求解，难以同时兼顾系统决策的有效性和实时性。为解决这个问题，本文考虑了一个包含两个终端用户的边缘计算系统，并提出了一种混合深度学习–优化算法对系统中的任务卸载决策与资源分配进行联合优化，本文的主要贡献如下。

1) 对用户的任务数据处理时延、能耗以及协作计算费用等进行建模，设计了以效用为导向的协作激励机制。并以最大化用户效用为目标，建立了用户任务决策与系统资源分配的联合优化问题模型。

2) 基于目标优化问题的结构特性，综合利用深度学习技术实时性高和凸优化方法求解精确的特点，为目标问题的求解设计了深度学习与凸优化理论相结合的新型优化算法框架，并在给定任务卸载决策的情况下，给出了系统资源优化分配的最优闭式解。

3) 采用数值仿真的方法，在不同系统参数设置条件下，从算法的收敛性、实时性、有效性等多个方面对所提算法的性能进行了全面评估。实验结果显示，相较于对比算法，本文所提算法可在大幅度降低计算时间的同时实现系统效用性能的近似最优化。

1　系统模型

基于用户协作的边缘计算系统模型如图 1 所示。该系统由一个边缘服务器和两类终端用户 EU 和 ER 组成。其中，用户 ER 为在运营商处登记的用户（简称已注册用户），可以直接访问边缘服务器。而用户 EU 未在运营商处登记（简称未注册用户），不具备访问边缘服务器的权限，因此不能与边缘服务器直接通信。不失一般性，本文假设未注册用户 EU 有 $N_{\mathrm{u}}(N_{\mathrm{u}}>0)$ 个任务需要完成，而已注册用户 ER 有 $N_{\mathrm{r}}(N_{\mathrm{r}}>0)$ 个任务需要完成。相应地，系统的总任务数为 $N=N_{\mathrm{u}}+N_{\mathrm{r}}$。为便于描述，记 $\mathcal{N}_{\mathrm{u}}=\{1,2,\cdots,N_{\mathrm{u}}\}$ 为未注册用户的任务集，$\mathcal{N}_{\mathrm{r}}=\{N_{\mathrm{u}}+1,N_{\mathrm{u}}+2,\cdots,N\}$ 为已注册用户的任务集。系统的总任务集则可表示为 $\mathcal{N}=\mathcal{N}_{\mathrm{u}}\cup\mathcal{N}_{\mathrm{r}}=\{1,2,\cdots,N\}$。考虑部分 5G 应用场景中任务（如图像处理应用中的每张图片）的不可分割性，本文采用二进制卸载方式对每个任务进行处理。为充分利用系统的计算资源，对于已注册用户的任务 $i\in\mathcal{N}_{\mathrm{r}}$，用户 ER 可在本地端进行任务处理，也可向运营商支付相应的费用，将任务卸载至边缘服务器进行计算；而对于未注册用户的任务 $i\in\mathcal{N}_{\mathrm{u}}$，用户 EU 可在本地端对其进行处理，也可向已注册用户 ER 提出支付一笔数据协作处理费用，发起任务协作计算请求。在收到请求后，用户 ER 根据当前系统状态（如信道强度、本地计算资源等）决定是否接受任务 $i\in\mathcal{N}_{\mathrm{u}}$ 的协作计算请求，并决定其处理方式，包括：拒绝请求（将任务 $i\in\mathcal{N}_{\mathrm{u}}$ 交由未注册用户 EU 自行处理）、同意请求并在已注册用户 ER 端进行计算或同意请求并卸载至边缘服务器端进行处理。例如，在图 1 中，未注册用户和已注册用户各有 5 个待处理任务（$N_{\mathrm{u}}=N_{\mathrm{r}}=5$）。针对未注册用户发起的 5 个任务（$\mathrm{task}_1$～$\mathrm{task}_5$）的协作计算请求，已注册用户只接受了前 3 项。因此，未注册用户支付相应费用后将任务 task_1、task_2 和 task_3 卸载至已注册用户端，并自行处理任务 task_4 和 task_5。已注册用户端通过向运营商支付相应费用将来自未注册用户的任务 task_1、task_2 以及自身任务 task_6、task_7 和 task_8 卸载至边缘服务器进行计算，而在本地对 task_3、task_9 和 task_{10} 进行处理。

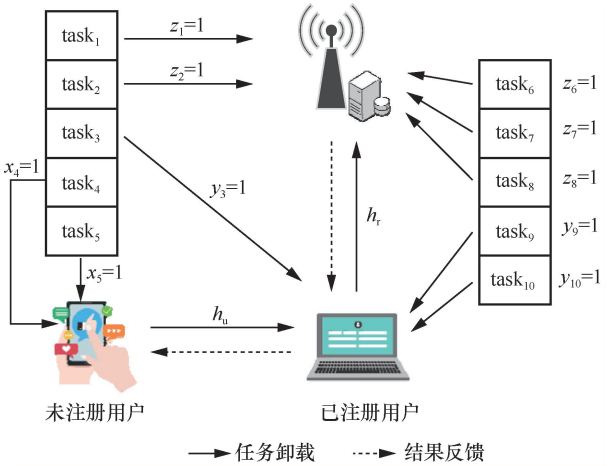

图 1　基于用户协作的边缘计算系统模型

记任务 $i\in\mathcal{N}$ 的二进制卸载指示变量为 $x_i,y_i,z_i\in\{0,1\},\forall i\in\mathcal{N}$。其中，$x_i=1$（或 0）表示任务 i 由（或不由）未注册用户自行处理；$y_i=1$（或 0）表示任务 i 在（或不在）已注册用户端进行计算；$z_i=1$（或 0）表示任务 i 在（或不在）边缘服务器端进行计算。因此，对于未注册用户任务 $i\in\mathcal{N}_{\mathrm{u}}$，

其二进制卸载决策应满足约束

$$x_i + y_i + z_i = 1, \forall i \in \mathcal{N}_{\mathrm{u}} \tag{1}$$

而对于已注册用户端任务 $i \in \mathcal{N}_{\mathrm{r}}$，其卸载决策应满足约束

$$x_i = 0, y_i + z_i = 1, \forall i \in \mathcal{N}_{\mathrm{r}} \tag{2}$$

值得注意的是，本文所考虑的两用户系统可看作多用户系统中的独立子系统。具体而言，考虑一个由 K_{u} 个未注册用户、K_{r} 个已注册用户，以及一个边缘服务器组成的多用户系统。未注册用户根据一定的规则（如就近原则）向已注册用户发起任务处理请求。记 $\mathcal{S}_k$ 为第 k 个已注册用户所对应的未注册用户集合，$k = 1, \cdots, K_{\mathrm{r}}$，则 $\sum_{k=1}^{K_{\mathrm{r}}} |\mathcal{S}_k| = K_{\mathrm{u}}$。在同意请求后，已注册用户 k 接收来自集合 $\mathcal{S}_k$ 中未注册用户的任务数据，并为每个未注册用户建立独立线程，通过本地计算或边缘计算对各用户数据进行并行处理。为避免用户间信道干扰，可采用正交频分多址接入技术实现不同用户任务数据的传输。在此情况下，集合 $\mathcal{S}_k$ 中各用户的数据传输与处理是相互独立的，本文所研究的两用户系统是上述多用户系统中的其中一个独立子系统。下面将对系统中的任务协作计算模型及目标问题进行建模。

1.1 任务协作计算模型

用户协作边缘系统任务执行示意图如图 2 所示。其中，$T_{\mathrm{u}}^{\mathrm{o}}$ 和 $T_{\mathrm{r}}^{\mathrm{o}}$ 分别为未注册用户和已注册用户的任务卸载时长；$T_{\mathrm{u,e}}$ 和 $T_{\mathrm{r,e}}$ 分别为边缘服务器用于计算未注册用户任务和已注册用户任务的时长；$T_{\mathrm{u,r}}$ 和 $T_{\mathrm{r,r}}$ 分别为已注册用户通过本地计算处理未注册用户任务和自身任务的时长；T_{u} 和 T_{r} 分别为已注册用户处理未注册用户任务和自身任务的总时延。在通过协作计算请求后，未注册用户耗时 $T_{\mathrm{u}}^{\mathrm{o}}$ 将相应的任务数据传输至已注册用户端。随后，已注册用户耗时 $T_{\mathrm{r}}^{\mathrm{o}}$ 将进行边缘计算的任务数据卸载至边缘服务器端，同时对剩下的任务数据进行本地处理。考虑未注册用户是付费客户，本文假设已注册用户优先处理未注册用户的任务数据，再处理自身的任务数据。在接收到任务数据后，边缘服务器也将按先后顺序依次处理未注册用户和已注册用户的任务。记任务 $i \in \mathcal{N}$ 的数据量为 D_i，计算工作负载（即计算任务所需的 CPU 周期数）为 L_i。下面对各用户任务处理的时延、能耗以及费用进行建模。

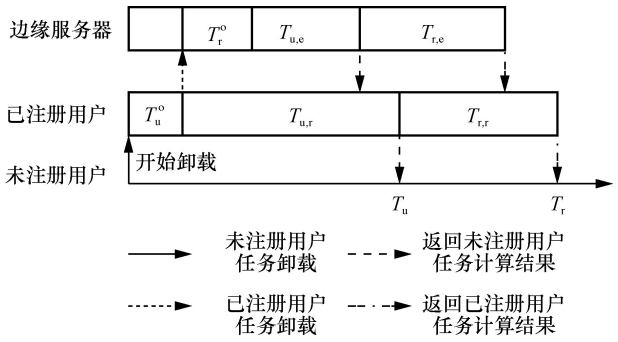

图 2 用户协作边缘系统任务执行示意图

假设系统的通信信道为块衰落信道，即信道增益在单个通信时隙中保持不变，而在不同时隙之间随机变化。令 h_{r} 和 h_{u} 分别表示已注册用户与边缘服务器之间、未注册用户与已注册用户之间的信道增益，B_{r} 和 B_{u} 为系统分别分配给已注册用户和未注册用户的带宽，则未注册用户的任务卸载速率可表示为

$$r_{\mathrm{u}}^{\mathrm{o}} = B_{\mathrm{u}} \mathrm{lb}\left(1 + \frac{P_{\mathrm{u}}^{\mathrm{o}} h_{\mathrm{u}}}{\sigma_{\mathrm{r}}^2}\right) \tag{3}$$

其中，$P_{\mathrm{u}}^{\mathrm{o}}$ 为未注册用户卸载任务的传输功率，σ_{r}^2 为已注册用户端的加性高斯白噪声功率。未注册用户卸载至已注册用户进行协作计算的任务数据量可表示为 $D_{\mathrm{u}} = \sum_{i \in \mathcal{N}_{\mathrm{u}}} (y_i + z_i) D_i$，其相应任务卸载时长为

$$T_{\mathrm{u}}^{\mathrm{o}} = \frac{D_{\mathrm{u}}}{r_{\mathrm{u}}^{\mathrm{o}}} \tag{4}$$

假设未注册服务器为单位工作负载支付的协作计算费用为 μ_{u}，则其向已注册用户支付的总费用为 $C_{\mathrm{u}} = \mu_{\mathrm{u}} L_{\mathrm{u}}$，其中 $L_{\mathrm{u}} = \sum_{i \in \mathcal{N}_{\mathrm{u}}} (y_i + z_i) L_i$。类似地，已注册用户的任务卸载速率为

$$r_{\mathrm{r}}^{\mathrm{o}} = B_{\mathrm{r}} \mathrm{lb}\left(1 + \frac{P_{\mathrm{r}}^{\mathrm{o}} h_{\mathrm{r}}}{\sigma_{\mathrm{e}}^2}\right) \tag{5}$$

其中，$P_{\mathrm{r}}^{\mathrm{o}} \leqslant P_{\mathrm{r}}^{\max}$ 为已注册用户卸载任务的传输功率，$P_{\mathrm{r}}^{\max}$ 为其最大值；σ_{e}^2 为边缘服务器端的噪声功率。已注册用户卸载到边缘服务器的总数据量可表示为 $D_{\mathrm{r}} = \sum_{i \in \mathcal{N}} z_i D_i$，其任务卸载时长为

$$T_{\mathrm{r}}^{\mathrm{o}} = \frac{D_{\mathrm{r}}}{r_{\mathrm{r}}^{\mathrm{o}}} \tag{6}$$

令 μ_{e} 为已注册用户为单位工作负载支付的边缘计算费用，则已注册用户须向运营商支付的总费用为 $C_{\mathrm{r}} = \mu_{\mathrm{e}} L_{\mathrm{r}}$，其中，$L_{\mathrm{r}} = \sum_{i \in \mathcal{N}} z_i L_i$。记函数

$p(x)=\sigma_{\mathrm{e}}\left(2^{\frac{x}{B_{\mathrm{r}}}}-1\right)$，已注册用户卸载功率 $P_{\mathrm{r}}^{\mathrm{o}}$ 可表示为 $P_{\mathrm{r}}^{\mathrm{o}}=\dfrac{1}{h_{\mathrm{r}}}p\left(\dfrac{D_{\mathrm{r}}}{T_{\mathrm{r}}^{\mathrm{o}}}\right)$。相应地，已注册用户用于任务卸载的能耗为

$$e_{\mathrm{r}}^{\mathrm{o}}=P_{\mathrm{r}}^{\mathrm{o}}T_{\mathrm{r}}^{\mathrm{o}}=\frac{T_{\mathrm{r}}^{\mathrm{o}}}{h_{\mathrm{r}}}p\left(\frac{D_{\mathrm{r}}}{T_{\mathrm{r}}^{\mathrm{o}}}\right) \tag{7}$$

假设已注册用户计算未注册用户任务和自身任务所采用的频率分别为 $f_{\mathrm{u,r}}\left(f_{\mathrm{u,r}}\leqslant f_{\mathrm{r}}^{\max}\right)$ 和 $f_{\mathrm{r,r}}\left(f_{\mathrm{r,r}}\leqslant f_{\mathrm{r}}^{\max}\right)$，其中 $f_{\mathrm{r}}^{\max}$ 为其 CPU 最高工作频率。由文献[27-28]可知，任务的计算时间是任务工作负载与 CPU 频率的比值；而其能耗则与任务工作负载（或 CPU 频率）的立方呈线性关系。相应地，已注册用户用于处理未注册用户任务的本地计算时间 $T_{\mathrm{u,r}}$ 和能耗 $e_{\mathrm{u,r}}$ 可分别表示为

$$T_{\mathrm{u,r}}=\frac{L_{\mathrm{u,r}}}{f_{\mathrm{u,r}}} \tag{8}$$

$$e_{\mathrm{u,r}}=\zeta f_{\mathrm{u,r}}^{3}T_{\mathrm{u,r}}=\zeta\frac{L_{\mathrm{u,r}}^{3}}{T_{\mathrm{u,r}}^{2}} \tag{9}$$

其中，$L_{\mathrm{u,r}}=\sum_{i\in\mathcal{N}_{\mathrm{u}}}y_iL_i$ 为在已注册用户端处理的未注册用户任务的总工作负载；ζ 为 CPU 的计算能效系数。已注册用户用于处理自身任务的本地计算时间 $T_{\mathrm{r,r}}$ 和能耗 $e_{\mathrm{r,r}}$ 可分别表示为

$$T_{\mathrm{r,r}}=\frac{L_{\mathrm{r,r}}}{f_{\mathrm{r,r}}} \tag{10}$$

$$e_{\mathrm{r,r}}=\zeta f_{\mathrm{r,r}}^{3}T_{\mathrm{r,r}}=\zeta\frac{L_{\mathrm{r,r}}^{3}}{T_{\mathrm{r,r}}^{2}} \tag{11}$$

其中，$L_{\mathrm{r,r}}=\sum_{i\in\mathcal{N}_{\mathrm{r}}}y_iL_i$ 为采用本地计算的已注册用户自身任务的总工作负载。通过对式(7)～式(11)求和，已注册用户的能量成本 E_{r} 可表示为

$$E_{\mathrm{r}}=e_{\mathrm{r}}^{\mathrm{o}}+e_{\mathrm{u,r}}+e_{\mathrm{r,r}} \tag{12}$$

在接收到来自已注册用户的任务后，边缘服务器采用固定 CPU 频率 f_{e} 对数据进行处理。记 $L_{\mathrm{u,e}}=\sum_{i\in\mathcal{N}_{\mathrm{u}}}z_iL_i$ 和 $L_{\mathrm{r,e}}=\sum_{i\in\mathcal{N}_{\mathrm{r}}}z_iL_i$ 分别为卸载到边缘服务器的未注册用户和已注册用户的总工作负载。则未注册用户和已注册用户任务的边缘处理时长分别为

$$T_{\mathrm{u,e}}=\frac{L_{\mathrm{u,e}}}{f_{\mathrm{e}}} \tag{13}$$

$$T_{\mathrm{r,e}}=\frac{L_{\mathrm{r,e}}}{f_{\mathrm{e}}} \tag{14}$$

综合上述任务处理时延模型并结合图 2 中的系统工作时序可知，已注册用户处理未注册用户任务和自身任务所需的总时延分别为

$$T_{\mathrm{u}}=\max(T_{\mathrm{u}}^{\mathrm{o}}+T_{\mathrm{u,r}},T_{\mathrm{u}}^{\mathrm{o}}+T_{\mathrm{r}}^{\mathrm{o}}+T_{\mathrm{u,e}}) \tag{15}$$

$$T_{\mathrm{r}}=\max(T_{\mathrm{u}}^{\mathrm{o}}+T_{\mathrm{u,r}}+T_{\mathrm{r,r}},T_{\mathrm{u}}^{\mathrm{o}}+T_{\mathrm{r}}^{\mathrm{o}}+T_{\mathrm{u,e}}+T_{\mathrm{r,e}}) \tag{16}$$

1.2　目标问题建模

本文的优化目标在于从已注册用户的角度出发，在保证一定客户（即未注册用户）服务质量的前提下最大化其自身的效用。对于已注册用户而言，其收入来源于未注册用户所支付的任务协作费用 $C_{\mathrm{u}}=\mu_{\mathrm{u}}L_{\mathrm{u}}$，其消耗则包括任务处理时延 T_{r}、能耗 E_{r}，以及其向运营商支付的费用 $C_{\mathrm{r}}=\mu_{\mathrm{e}}L_{\mathrm{r}}$。另外，对于未注册用户而言，其服务质量由其任务计算时延 T_{u} 决定。相应地，已注册用户的效用函数可定义为

$$U=\mu_{\mathrm{u}}L_{\mathrm{u}}-\mu_{\mathrm{e}}L_{\mathrm{r}}-\eta \tag{17}$$

其中，$\eta=\omega_eE_{\mathrm{r}}+\omega_{\mathrm{tu}}T_{\mathrm{u}}+\omega_{\mathrm{tr}}T_{\mathrm{r}}$。$\omega_{\mathrm{e}}$ 和 ω_{tr} 分别为已注册用户所花费的能量和时间成本权重，ω_{tu} 为未注册用户的时间成本权重。为提升自身的效用，已注册用户需要根据系统状态信息（如信道强度、自身数据量以及未注册用户卸载的数据量）对任务的卸载决策以及计算资源分配方案进行调整。考虑实际应用情况中用户之间的独立性，在制定优化决策的过程中，已注册用户将未注册用户的传输功率看作固定值，仅通过优化任务卸载决策 $\pi=\{x_1,y_1,z_1,x_2,y_2,z_2,\cdots,x_N,y_N,z_N\}$、自身的传输功率 $P_{\mathrm{r}}^{\mathrm{o}}$，以及本地计算频率 $F=\{f_{\mathrm{u,r}},f_{\mathrm{r,r}}\}$ 最大化 U，其相应的目标优化问题可描述为

$$\begin{aligned}
\mathbf{P}_1:\quad & \max_{\pi,F,P_{\mathrm{r}}^{\mathrm{o}}}\ U\\
\text{s.t.}\quad & \text{式(1)～式(2)}\\
& \mathrm{C0}:\ x_i,y_i,z_i\in\{0,1\},\forall i\in\mathcal{N}\\
& \mathrm{C1}:\ f_{\mathrm{r,r}}\leqslant f_{\mathrm{r}}^{\max}\\
& \mathrm{C3}:\ f_{\mathrm{u,r}}\leqslant f_{\mathrm{r}}^{\max}\\
& \mathrm{C3}:\ P_{\mathrm{r,o}}\leqslant P_{\mathrm{r}}^{\max}
\end{aligned} \tag{18}$$

其中，C1、C2 和 C3 为本地处理频率和已注册用户传输功率的上下界约束；C0 表示任务卸载决策变量

的取值为二进制；同时，任务卸载决策需满足式(1)和式(2)中的约束。在 $\mathbf{P}_1$ 中，二进制整型任务卸载优化变量 π 与连续资源优化变量 F 和 $P_{\mathrm{r}}^{\mathrm{o}}$ 之间深度耦合，这使得 $\mathbf{P}_1$ 是一个混合整数非线性规划（MINIP, mixed integer nonlinear programming）问题。一般而言，MINLP 问题的求解是 NP 难的[29]。为有效地求解 $\mathbf{P}_1$，本文提出一种将深度学习与传统凸优化理论结合的方法，快速高效地输出任务卸载与资源分配联合优化的策略。

2 混合深度学习–优化算法

一般而言，相较于求解输出为连续变量的回归问题，深度学习更擅长解决输出为离散变量的广义分类问题。而凸优化则刚好相反，更适合变量连续优化的情况。通过分析 $\mathbf{P}_1$ 可知，在给定任务卸载决策 π 的情况下，剩下的资源优化问题是一个凸优化问题，可利用现有凸优化方法（如内点法[30-31]）有效求解。因此，本文提出一种混合深度学习–优化的算法实现任务卸载与资源分配的联合优化。混合深度学习–优化算法框架如图 3 所示，该算法采用了“演员–评论家”（actor-critic）的网络结构，分别基于深度神经网络（DNN, deep neural network）和凸优化理论构建“演员”和“评论家”模块。其中，“演员”模块的目的在于通过优化调整深度神经网络的权重参数，从而根据当前的系统输入状态 I_t 输出任务卸载决策 π；而“评论家”模块的目的则在于通过优化系统的资源分配评估输出决策 π，并将当前得到的优化决策用于更新“演员”模块中 DNN 的参数。随着系统工作时间的增长，“演员”和“评论家”模块迭代运行直至收敛（如第 3 节中仿真结果所示，算法收敛仅需要迭代大约 4 000 个工作时隙）。在实际应用中，为保证算法结果的实时性和有效性，本文对算法进行预训练，在系统运行一段时间后再将算法运用于系统优化中。

2.1 基于 DNN 的任务卸载决策优化

已注册用户在当前时刻观测的系统状态主要包括信道增益 $h=\{h_{\mathrm{r}},h_{\mathrm{u}}\}$、任务数据量 $\mathcal{D}=\{D_i,i\in\mathcal{N}\}$ 以及任务负载 $\mathcal{L}=\{L_i,i\in\mathcal{N}\}$。因此，“演员”模块的输入可定义为 $I_t=\{\mathcal{D},\mathcal{L},h\}$。值得注意的是，在实际训练过程中，为加快网络的收敛速度，通常需将上述输入参数进行归一化处理。

根据当前的输入 I_t，“演员”模块输出相应的任务卸载优化决策。本文采用两层隐藏层构建 DNN，其神经元个数分别为 240 和 160，并同时采用线性整流函数（ReLU, rectified linear unit）作为激活函数，以及均方误差（MSE, mean square error）作为损失函数。相应地，DNN 的输出是实数域中的一个向量 $\bar{\boldsymbol{\pi}}$，其中，$\bar{\boldsymbol{\pi}}$ 中的元素取值为[0,1]区间的实数值。为输出离散的二进制任务卸载决策，本文在“演员”网络中采用了基于 K 近邻（KNN, K-nearest neighbor）算法的量化方法，将 DNN 的输出 $\bar{\boldsymbol{\pi}}$ 量化成距离其最近（以欧氏距离算）的 K 个可行的候选

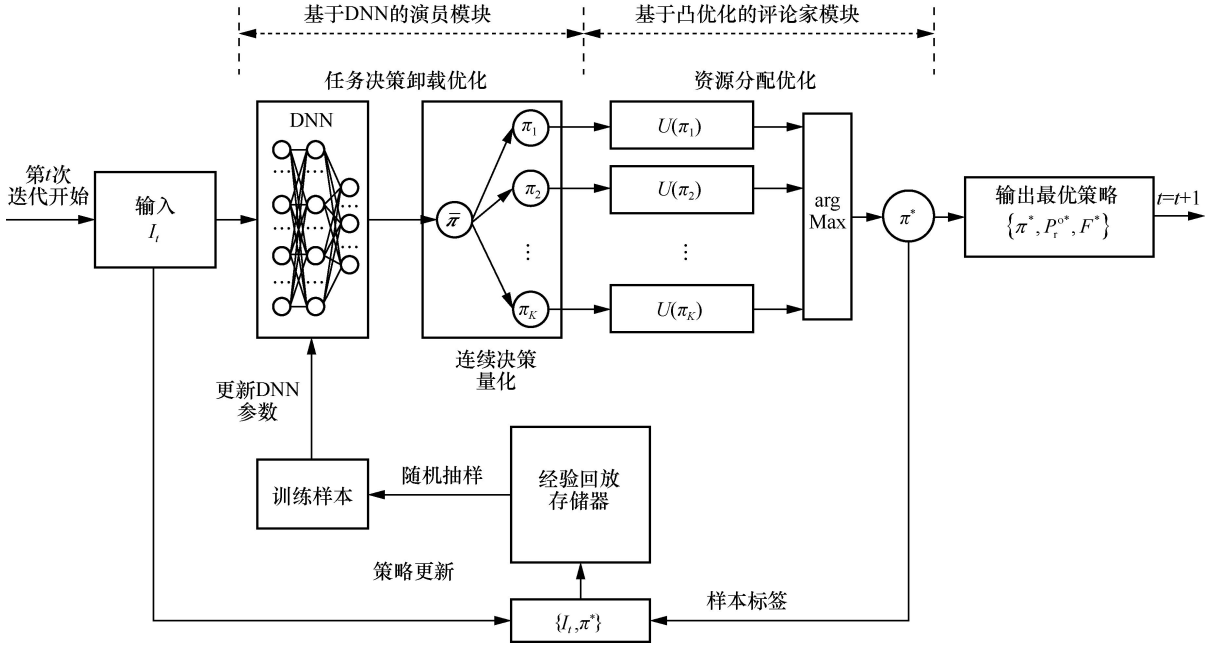

图 3 混合深度学习–优化算法框架

卸载决策方案，即 $\{\pi_1,\cdots,\pi_k,\cdots,\pi_K\}$。由于“评论家”模块需要评估“演员”模块所输出的所有候选决策（详见第 2.2 节），因此 K 值的选取直接影响着算法的性能。若 K 值选取过小，神经网络获取到的知识范围过窄，算法容易收敛到局部最优解；若选取过大，则“评论家”模块需要多次求解连续资源优化问题，导致算法的时间复杂度上升。为平衡算法的性能与复杂度，本文采用了一种 K 值自适应的取值方法[32]，根据式(19)确定第 $t\geqslant 1$ 次迭代（即第 $t\geqslant 1$ 个工作时隙）中 K 的取值

$$K_t=\begin{cases}N, & t=1\\ \min(\max(k_{t-1}^*,\cdots k_{t-\Delta}^*)+1,N), & t \bmod \Delta=0\\ K_{t-1}, & \text{其他}\end{cases} \tag{19}$$

其中，$\Delta=32$ 为 K 值的更新周期；k_t^* 为第 t 次迭代中候选方案 $\{\pi_1,\cdots,\pi_k,\cdots,\pi_K\}$ 中最优方案的序号；mod 为求模函数。

2.2　基于凸优化的资源分配优化

对于“演员”模块所输出的整型任务卸载决策 $\hat{\pi}\in\{\pi_1,\cdots,\pi_K\}$，“评论家”模块需要对其效果进行评估，即在给定任务卸载决策 $\hat{\pi}$ 的情况下通过求解最优资源分配 F 和 $P_{\mathrm{r}}^{\mathrm{o}}$ 评估已注册用户的最大效用。记 $T_{\mathrm{c}}=\{T_{\mathrm{u,r}},T_{\mathrm{r,r}}\}$，利用式(7)～式(11)中 $P_{\mathrm{r}}^{\mathrm{o}}$ 与 $T_{\mathrm{r}}^{\mathrm{o}}$ 以及 T_{c} 与 F 一一对应的关系，可以将原问题等价地转换为如下资源分配优化问题

$$\begin{aligned}\mathbf{P}_2:\ \min_{T_{\mathrm{c}},T_{\mathrm{r}}^{\mathrm{o}},T_{\mathrm{u}},T_{\mathrm{r}}} & U(\hat{\pi})=\omega_{\mathrm{e}}\left[\frac{T_{\mathrm{r}}^{\mathrm{o}}}{h_{\mathrm{r}}}p\left(\frac{D_{\mathrm{r}}}{T_{\mathrm{r}}^{\mathrm{o}}}\right)+\zeta\frac{L_{\mathrm{u,r}}^3}{T_{\mathrm{u,r}}^2}+\zeta\frac{L_{\mathrm{r,r}}^3}{T_{\mathrm{r,r}}^2}\right]\\ \text{s.t.}\quad & \mathrm{C4}:T_{\mathrm{u,r}}\geqslant\frac{L_{\mathrm{u,r}}}{f_{\mathrm{r}}^{\max}}\\ & \mathrm{C5}:T_{\mathrm{r,r}}\geqslant\frac{L_{\mathrm{r,r}}}{f_{\mathrm{r}}^{\max}}\\ & \mathrm{C6}:T_{\mathrm{r}}^{\mathrm{o}}\geqslant\frac{D_{\mathrm{r}}}{B_{\mathrm{r}}\mathrm{lb}\left(1+\dfrac{P_{\mathrm{r}}^{\max}h_{\mathrm{r}}}{\sigma_{\mathrm{e}}^2}\right)}\\ & \mathrm{C7}:T_{\mathrm{u}}\geqslant T_{\mathrm{u}}^{\mathrm{o}}+T_{\mathrm{u,r}}\\ & \mathrm{C8}:T_{\mathrm{u}}\geqslant T_{\mathrm{u}}^{\mathrm{o}}+T_{\mathrm{r}}^{\mathrm{o}}+T_{\mathrm{u,e}}\\ & \mathrm{C9}:T_{\mathrm{r}}\geqslant T_{\mathrm{u}}^{\mathrm{o}}+T_{\mathrm{u,r}}+T_{\mathrm{r,r}}\\ & \mathrm{C10}:T_{\mathrm{r}}\geqslant T_{\mathrm{u}}^{\mathrm{o}}+T_{\mathrm{r}}^{\mathrm{o}}+T_{\mathrm{u,e}}+T_{\mathrm{r,e}}\end{aligned} \tag{20}$$

$\mathbf{P}_2$ 是一个凸优化问题，可通过经典的凸优化算法有效求解。为进一步降低算法的时间复杂度，下面给出其闭式解的结构，并基于此给出低复杂度的求解算法的步骤。

由文献[30]可知，求解凸优化问题 $\mathbf{P}_2$，等价于求解其对偶问题

$$\begin{aligned}\mathbf{P}_2\text{-D}:\ & \max_{\alpha,\beta}\min_{T_{\mathrm{c}},T_{\mathrm{r}}^{\mathrm{o}},T_{\mathrm{u}},T_{\mathrm{r}}} L(T_{\mathrm{c}},T_{\mathrm{r}}^{\mathrm{o}},T_{\mathrm{u}},T_{\mathrm{r}},\alpha,\beta)\\ & \text{s.t. C4}\sim\text{C6},\alpha\geqslant 0,\beta\geqslant 0\end{aligned} \tag{21}$$

其中，$\alpha=\{\alpha_1,\alpha_2\}$ 和 $\beta=\{\beta_1,\beta_2\}$ 为对应约束 C7～C10 的非负拉格朗日乘子。$L(T_{\mathrm{c}},T_{\mathrm{r}}^{\mathrm{o}},T_{\mathrm{u}},T_{\mathrm{r}},\alpha,\beta)$ 为 $\mathbf{P}_2$ 的部分拉格朗日方程，可表示为

$$\begin{aligned}&L(T_{\mathrm{c}},T_{\mathrm{r}}^{\mathrm{o}},T_{\mathrm{u}},T_{\mathrm{r}},\alpha,\beta)=\\ &\omega_{\mathrm{e}}\left[\frac{T_{\mathrm{r}}^{\mathrm{o}}}{h_{\mathrm{r}}}p\left(\frac{D_{\mathrm{r}}}{T_{\mathrm{r}}^{\mathrm{o}}}\right)+\zeta\frac{L_{\mathrm{u,r}}^3}{T_{\mathrm{u,r}}^2}+\zeta\frac{L_{\mathrm{r,r}}^3}{T_{\mathrm{r,r}}^2}\right]+\\ &\omega_{\mathrm{tu}}T_{\mathrm{u}}+\omega_{\mathrm{tr}}T_{\mathrm{r}}+\alpha_1(T_{\mathrm{u}}^{\mathrm{o}}+T_{\mathrm{u,r}}-T_{\mathrm{u}})+\\ &\alpha_2(T_{\mathrm{u}}^{\mathrm{o}}+T_{\mathrm{r}}^{\mathrm{o}}+T_{\mathrm{u,e}}-T_{\mathrm{u}})+\\ &\beta_1(T_{\mathrm{u}}^{\mathrm{o}}+T_{\mathrm{u,r}}+T_{\mathrm{r,r}}-T_{\mathrm{r}})+\\ &\beta_2(T_{\mathrm{u}}^{\mathrm{o}}+T_{\mathrm{r}}^{\mathrm{o}}+T_{\mathrm{u,e}}+T_{\mathrm{r,e}}-T_{\mathrm{r}})\end{aligned} \tag{22}$$

对偶问题 $\mathbf{P}_2$-D 的最优解可通过分别求解其内部子问题和外部主对偶问题获得。在给定拉格朗日乘子的条件下，其内部子问题为

$$\begin{aligned}\mathbf{P}_2\text{-DI}:\ & g(\alpha,\beta)=\min_{T_{\mathrm{c}},T_{\mathrm{r}}^{\mathrm{o}},T_{\mathrm{u}},T_{\mathrm{r}}} L(T_{\mathrm{c}},T_{\mathrm{r}}^{\mathrm{o}},T_{\mathrm{u}},T_{\mathrm{r}},\alpha,\beta)\\ & \text{s.t. C4}\sim\text{C6}\end{aligned} \tag{23}$$

记 $T_{\mathrm{u,r}}^*$、$T_{\mathrm{r,r}}^*$、$T_{\mathrm{r}}^{\mathrm{o}*}$ 为上述内部子问题的最优解。取拉格朗日方程 L 关于 $T_{\mathrm{u,r}}$、$T_{\mathrm{r,r}}$，以及 $T_{\mathrm{r}}^{\mathrm{o}}$ 的一阶导数并令其等于 0，并结合 C4～C6 中的边界约束条件，可以得到 $T_{\mathrm{u,r}}^*$、$T_{\mathrm{r,r}}^*$、$T_{\mathrm{r}}^{\mathrm{o}*}$ 分别具有如下结构

$$T_{\mathrm{u,r}}^*=\begin{cases}\left(\dfrac{2\omega_{\mathrm{e}}\zeta}{\alpha_1+\beta_1}\right)^{\frac{1}{3}}L_{\mathrm{u,r}}, & f_{\mathrm{r}}^{\max}>\left(\dfrac{\alpha_1+\beta_1}{2\omega_{\mathrm{e}}\zeta}\right)^{\frac{1}{3}}\\ \dfrac{L_{\mathrm{u,r}}}{f_{\mathrm{r}}^{\max}}, & \text{其他}\end{cases} \tag{24}$$

$$T_{\mathrm{r,r}}^*=\begin{cases}\left(\dfrac{2\omega_{\mathrm{e}}\zeta}{\beta_1}\right)^{\frac{1}{3}}L_{\mathrm{r,r}}, & f_{\mathrm{r}}^{\max}>\left(\dfrac{\beta_1}{2\omega_{\mathrm{e}}\zeta}\right)^{\frac{1}{3}}\\ \dfrac{L_{\mathrm{r,r}}}{f_{\mathrm{r}}^{\max}}, & \text{其他}\end{cases} \tag{25}$$

$$T_{\mathrm{r}}^{\mathrm{o}*}=\begin{cases}\dfrac{D_{\mathrm{r}}}{B_{\mathrm{r}}\mathrm{lb}\left(1+\dfrac{P_{\mathrm{r}}^{\max}h_{\mathrm{r}}}{\sigma_{\mathrm{e}}^2}\right)}, h_{\mathrm{r}}^{\max}\leqslant\dfrac{\sigma_{\mathrm{e}}^2}{P_{\mathrm{r}}^{\max}}\left(\dfrac{A}{-W(-Ae^{-A})}-1\right)\\ \dfrac{\ln 2D_{\mathrm{r}}}{B_{\mathrm{r}}\left[W\left(e^{-1}\left[\dfrac{(\alpha_2+\beta_2)h_{\mathrm{r}}}{\omega_{\mathrm{e}}\sigma_{\mathrm{e}}^2}-1\right]\right)+1\right]}, \text{其他}\end{cases} \tag{26}$$

其中，$A\triangleq 1+\dfrac{\alpha_2+\beta_2}{\sigma_{\mathrm{e}}^2P_{\mathrm{r}}^{\max}}$；$W(\cdot)$ 为朗伯 W 函数（Lambert-W function）。$\mathbf{P}_2$-D1 中的最优解 T_{u}^* 和 T_{r}^* 可直接根据式(15)和式(16)分别计算得出。而对于外部主对偶问题

$$\mathbf{P}_2\text{-DO}:\quad \min_{\alpha,\beta}\quad g(\alpha,\beta)\\ \text{s.t. }\alpha\geqslant 0,\beta\geqslant 0 \tag{27}$$

本文采用次梯度下降法[32]获得最优拉格朗日乘子 $\{\alpha^*,\beta^*\}$。将 $\{\alpha^*,\beta^*\}$ 代入式(24)～式(26)，并结合式(7)～式(11)，便可得到已注册用户的最优 CPU 工作频率 $f_{\mathrm{u,r}}^*=\dfrac{L_{\mathrm{u,r}}}{T_{\mathrm{u,r}}^*}$ 和 $f_{\mathrm{r,r}}^*=\dfrac{L_{\mathrm{r,r}}}{T_{\mathrm{r,r}}^*}$，以及最优任务卸载功率 $P_{\mathrm{r}}^{\mathrm{o}*}=\dfrac{\sigma_{\mathrm{e}}^2}{h_{\mathrm{r}}}\left(2^{\frac{D_{\mathrm{r}}}{B_{\mathrm{r}}T_{\mathrm{r}}^{\mathrm{o}*}}}-1\right)$。

通过求解 K 个凸优化问题，“评论家”模块得到“演员”模块输出动作 $\hat{\pi}\in\{\pi_1,\pi_2,\cdots,\pi_K\}$ 所对应的效用值 $U(\hat{\pi})=\{U(\pi_1),U(\pi_2),\cdots,U(\pi_K)\}$，并从中选取效用最大的决策 $\{\pi^*\}=\underset{\hat{\pi}\in\{\pi_1,\pi_2,\cdots,\pi_K\}}{\arg\max}\ U(\hat{\pi})$，将其与系统状态一并（即 $\{I_t,\pi^*\}$）作为训练样本存入存储器（replay memory）中。随后，采用经验回放（experience replay）技术（如图 3 所示），通过对存储器中的历史最优状态动作进行随机抽样生成训练样本，从而实现“演员”网络权重的更新。

混合深度学习–优化算法充分利用了深度学习技术适应性强、实时性高的特点，能快速地输出系统的优化决策；与此同时，不同于采用 DNN 构建“评论家”模块的传统深度学习方法，混合深度学习–优化算法充分利用了目标优化问题的结构特性，采用凸优化方法构建“评论家”模块，有效地保证了评估结果的精确性以及算法的收敛性。在随后的仿真实验中可以看到，随着 DNN 的迭代更新，算法的输出将最终收敛至近似最优解。

3 仿真结果与分析

本节在第 1 节系统模型的基础上通过实验仿真评估所提算法的性能。实验中，若无特殊说明，将系统仿真参数设置如下：未注册用户和已注册用户的任务个数分别默认为 $N_{\mathrm{u}}=5$ 和 $N_{\mathrm{r}}=8$。对于任务 $i\in\mathcal{N}$，其数据量 D_i 和工作负载 L_i 为随机数，分别服从 $D_i\sim\mathbb{U}_{\mathrm{D}}(3\text{ Mbit},9\text{ Mbit})$ 和 $L_i\sim\mathbb{U}_{\mathrm{L}}(30\text{ Mcycle},90\text{ Mcycle})$ 的均匀分布。用户协作计算和边缘计算的价格参数为 $\mu_{\mathrm{u}}=0.9\times10^{-8}$，$\mu_{\mathrm{e}}=0.1\times10^{-8}$。已注册用户效用函数中的权重参数为 $\omega_{\mathrm{e}}=3$、$\omega_{\mathrm{tu}}=0.4$ 和 $\omega_{\mathrm{tr}}=0.4$。已注册用户最大传输功率为 $P_{\mathrm{r}}^{\max}=0.1$ W，最大 CPU 频率为 $f_{\mathrm{r}}^{\max}=0.5$ GHz。为模拟网络的随机性，本文假设用户随机部署在网络中。记 $d_{\mathrm{u,r}}$、$d_{\mathrm{r,e}}$ 分别为未注册用户与已注册用户之间、已注册用户和边缘服务器之间的距离，则 $d_{\mathrm{u,r}}$、$d_{\mathrm{r,e}}$ 的值分别服从均匀分布 $d_{\mathrm{u,r}}\sim\mathbb{U}_{\mathrm{d}}(5\text{ m},10\text{ m})$ 和 $d_{\mathrm{r,e}}\sim\mathbb{U}_{\mathrm{d}}(10\text{ m},20\text{ m})$。实验中，采用自由路径损耗模型对未注册用户和已注册用户的信道进行建模。用户的信道增益分别为：$h_{\mathrm{u}}=A_{\mathrm{u}}^{\mathrm{d}}\left(\dfrac{3\times10^8}{4\pi f_{\mathrm{u}}^{\mathrm{c}}d_{\mathrm{u,r}}}\right)^{\gamma}$ 和 $h_{\mathrm{r}}=A_{\mathrm{r}}^{\mathrm{d}}\left(\dfrac{3\times10^8}{4\pi f_{\mathrm{r}}^{\mathrm{c}}d_{\mathrm{r,e}}}\right)^{\gamma}$。其中，下标“u”和“r”分别代表未注册用户和已注册用户；$A_{\mathrm{u}}^{\mathrm{d}}=3$ 和 $A_{\mathrm{r}}^{\mathrm{d}}=4.11$ 为用户的天线增益；$f_{\mathrm{u}}^{\mathrm{c}}=2\,300$ MHz 和 $f_{\mathrm{r}}^{\mathrm{c}}=915$ MHz 为用户任务卸载的载波频率；$\gamma=2.6$ 为路径损失指数。此外，未注册用户的卸载功率为 $P_{\mathrm{u}}^{\mathrm{o}}=0.08$ W；用户信道带宽为 $B_{\mathrm{u}}=2$ MHz 和 $B_{\mathrm{r}}=4$ MHz；CPU 计算能效系数为 $\zeta=10^{-26}$。已注册用户端和边缘服务器端的噪声功率为 $\sigma_{\mathrm{r}}^2=\sigma_{\mathrm{e}}^2=10^{-10}$ W。

3.1 算法收敛性

为研究算法的收敛性，算法训练损失随训练时间的变化趋势如图 4 所示，可以看到，训练损失随时间的变化而上下波动，这主要是训练数据的随机采样造成的。随着时间的增加，所提算法的训练损失迅速降低，仅不到 100 个时隙便从 $t=0$ 的 0.24 降至 0.05，并最终大约在 $t=2\,000$ 的时候稳定收敛至 0.03 左右。除此之外，为了体现该算法在任务数量变化场景下的适用性，本文在 $t=5\,000$ 时将系统的任务数量由 $N_{\mathrm{u}}=5$、$N_{\mathrm{r}}=8$ 增加至 $N_{\mathrm{u}}=7$、$N_{\mathrm{r}}=10$。从图 4 可以看到，在系统

任务数量改变后，本文提出的算法能够快速重新实现收敛。这说明本文算法能够针对任务数量的变化进行快速的自适应调整，能较好地适用于任务数量变化的情况。

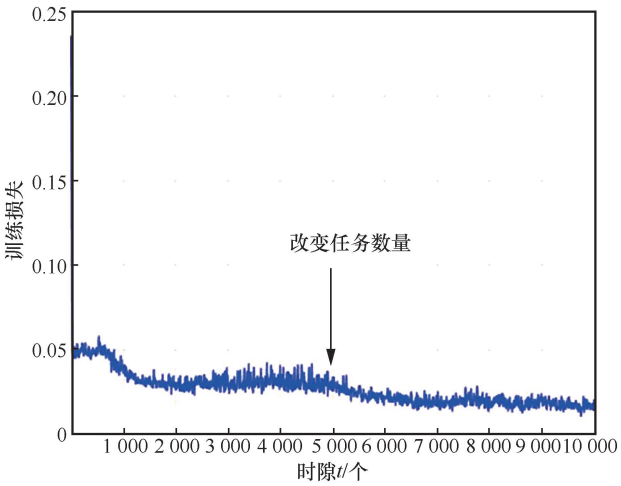

图 4　算法训练损失随训练时间的变化趋势

3.2　算法有效性与实时性

为评估所提算法的性能，本文考虑了以下 3 种对比算法。

1) 仅卸载算法[14]：已注册用户将所有任务卸载到边缘服务器进行处理。

2) 粒子群优化（PSO, particle swarm optimization）算法[22]：PSO 算法迭代搜索任务卸载决策，在每一次迭代搜索过程中，基于凸优化理论优化系统资源分配得到每次搜索结果的性能评估，进而实现系统任务卸载以及资源分配的优化。理论上，当迭代搜索次数足够大时，PSO 算法能得到目标优化问题的近似最优解。

3) 块坐标下降（BCD, block coordinate descent）算法[23]：BCD 算法通过迭代优化任务卸载和资源分配求解目标原问题 $\mathbf{P}_1$。具体而言，与本文提出的算法类似，它采用凸优化方法通过求解 $\mathbf{P}_2$ 实现系统的资源分配优化。而后在给定资源分配的情况下，通过采用块坐标下降法求解如下优化问题得到任务卸载优化决策

$$\mathbf{P}_3:\ \max_{\pi}\ \ \mu_{\mathrm{u}}\sum_{i\in\mathcal{N}_{\mathrm{u}}}(y_i+z_i)L_i-\mu_{\mathrm{e}}\sum_{i\in\mathcal{N}}z_iL_i-\eta^*(\pi)\quad \text{s.t. C0,}\ \ \text{式(1)}\sim\text{式(2)} \tag{28}$$

随着系统状态空间（如用户任务数量）的增大，BCD 算法的时间复杂快速增加，不能适用于系统的实时优化。但尽管如此，它可以近似最优地求解目标优化问题[23,33]。因此，在下面的仿真实验中，将 BCD 算法输出的结果看作最优值，用以评估所提算法的有效性。

考虑边缘服务器强大的运算能力及较为充裕的计算资源，采用仅卸载算法将所有任务卸载至服务器进行计算是实际应用中简单又较为有效的任务数据方法，其得到的结果通常被看作系统性能的下界之一。相应地，将仅卸载算法作为对比算法，不仅可以分析系统性能下界随系统参数的变化，还能同时研究选择性任务卸载（如本文提出的方法）对系统性能提升的重要性。另外，考虑 PSO 算法和 BCD 算法可近似最优地求解目标优化问题的特性，通过将其作为对比算法，不仅可以研究系统参数对系统性能上界的影响，还可以分析本文所提算法的效用性能与系统最优效用性能之间的差距。下面，将在不同的系统参数下对 3 种对比算法以及本文提出的混合深度学习–优化算法的性能进行评估。

首先，本文在不同任务数据量的情况下研究了所提算法的性能。将任务平均数据量 $\bar{D}$ 的取值从 4 Mbit 逐渐增加至 8 Mbit，不同任务数据量情况下用户的效用值如图 5 所示，已注册用户的效用值随着 $\bar{D}$ 的增加而逐渐降低。这是因为当 $\bar{D}$ 变大时，任务数据量相应增大，任务卸载时间能耗成本增加。然而，未注册用户给予的协作费用是关于数据量的常数，已注册用户收到的报酬并不随数据量的增加而增加，从而导致用户效用值的降低。还可以看到，本文提出的混合深度学习–优化算法的性能远胜于以及将任务完全卸载至边缘服务器的方法，并能达到与 PSO 算法、BCD 算法相近的性能。

其次，不同任务工作负载均值下用户的效用值如图 6 所示。实验中，将任务工作负载均值 $\bar{L}$ 的变化范围设置为[40 Mcycle, 80 Mcycle]。随着工作负载的增，已注册用户获得的协作报酬也在增加。可以看出，已注册用的效用值也随工作负载的增大而增大。在不同工作负载的情况下，本文提出的算法均能获得与 PSO 算法、BCD 算法相近的效用性能。在所考虑的 $\bar{L}$ 变化范围中，相较于将任务全卸载至边缘服务器的方法，本文提出的算法可使用户效用值提高至少 85.4%。

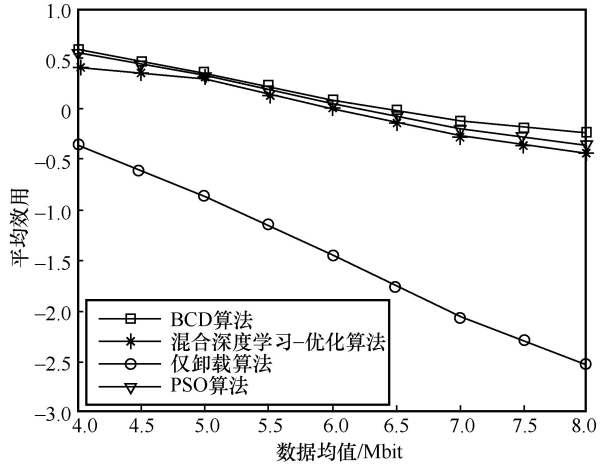

图 5　不同任务数据量情况下用户的效用值

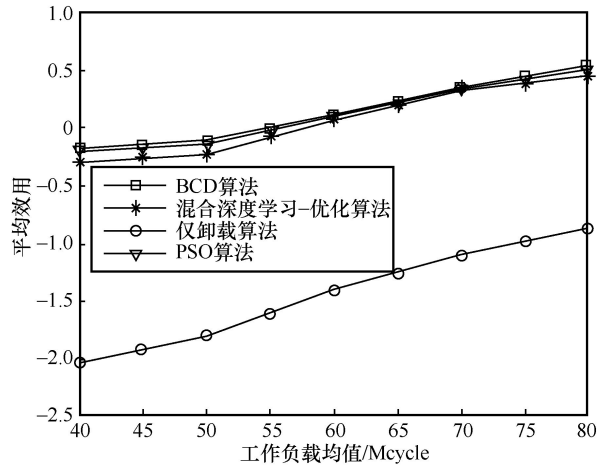

图 6　不同任务工作负载均值下用户的效用值

除此之外，为评估所提算法的实时性，算法计算时间对比见表 1，对比了不同任务数量下所提算法与 PSO 算法和 BCD 算法求解目标优化问题 $\mathbf{P}_1$ 所需要的计算时间。实验中，本文从 $N_r + N_u = 11$ 开始，依次将总任务数量增加至 $N_r + N_u = 15$。如表 1 中结果所示，由于给定当前系统状态输入后，PSO 算法需要迭代上百次才能收敛至目标问题的近似最优解，其求解目标优化问题所需要的时间将近 50 s，极其不适用于网络环境参数快速变化的场景。相较于 PSO 算法，对于给定的系统状态输入，BCD 算法仅需要迭代数次便能优化任务卸载决策和系统资源分配，收敛所需要的时间大幅缩短。但尽管如此，无线信道相干时间通常最长仅为数秒，而 BCD 算法的计算时间接近甚至超过 1 s，开销过大，难以应用于无线环境快速变化的场景。而本文提出的混合深度学习–优化算法采用 DNN 优化任务卸载决策，可根据系统状态直接输出优化结果，具备高效实时的优点。在 DNN 收敛后，针对当前输入的系统状态，所提算法计算目标优化问题所需要的时间大约仅为 BCD 算法计算时延的 1/10，仅需要约花费亚秒级的时间即可输出最终优化决策。

表 1　　算法计算时间对比

总任务数量/个	11	12	13	14	15
PSO 算法计算时间/s	46.6	46.5	46.8	48.4	47.4
BCD 算法计算时间/s	0.74	0.81	1.00	1.07	1.35
混合深度学习–优化算法计算时间/s	0.09	0.09	0.10	0.11	0.14

4　结束语

针对边缘系统中已注册用户协助未注册用户进行任务处理的场景，本文从已注册用户的角度出发，为其建立了效用评价函数，并在此基础上研究了协作边缘系统中用户的任务卸载与资源分配联合优化方法。为最大化已注册用户的效用，本文将目标优化问题建模为一个 MINLP 问题，并提出了一种混合深度学习–优化算法对目标问题进行求解。该算法采用深度学习方法构建“演员”模块确定最优任务卸载决策，并基于凸优化理论设计“评论家”模块求解最优资源分配，充分利用了深度学习技术实时性强和凸优化方法求解问题精确度高的优点。实验结果表明，本文提出的算法具备良好的收敛性和实时性，能有效提升用户的效用性能。

参考文献：

[1] GSMA Intelligence. Understanding 5G: perspectives on future technological advancements in mobile[EB]. 2014.

[2] LIN P, SONG Q Y, YU F R, et al. Wireless virtual reality in beyond 5G systems with the Internet of intelligence[J]. IEEE Wireless Communications, 2021, 28(2): 70-77.

[3] SONKOLY B, SZABÓ R, NÉMETH B, et al. 5G applications from vision to reality: multi-operator orchestration[J]. IEEE Journal on Selected Areas in Communications, 2020, 38(7): 1401-1416.

[4] AGIWAL M, ROY A, SAXENA N. Next generation 5G wireless networks: a comprehensive survey[J]. IEEE Communications Surveys & Tutorials, 2016, 18(3): 1617-1655.

[5] WEN M W, LI Q, KIM K J, et al. Private 5G networks: concepts, architectures, and research landscape[J]. IEEE Journal of Selected Topics in Signal Processing, 2022, 16(1): 7-25.

[6] ZENG J, SUN J Y, WU B W, et al. Mobile edge communications, computing, and caching (MEC3) technology in the maritime communication network[J]. China Communications, 2020, 17(5): 223-234.

[7] MACH P, BECVAR Z. Mobile edge computing: a survey on architec-

ture and computation offloading[J]. IEEE Communications Surveys & Tutorials, 2017, 19(3): 1628-1656.

[8] TRAN T X, HAJISAMI A, PANDEY P, et al. Collaborative mobile edge computing in 5G networks: new paradigms, scenarios, and challenges[J]. IEEE Communications Magazine, 2017, 55(4): 54-61.

[9] LI X, HUANG L, WANG H, et al. An integrated optimization-learning framework for online combinatorial computation offloading in MEC networks[J]. IEEE Wireless Communications, 2022, 29(1): 170-177.

[10] BAI T, PAN C H, HAN C, et al. Reconfigurable intelligent surface aided mobile edge computing[J]. IEEE Wireless Communications, 2021, 28(6): 80-86.

[11] ZHANG W W, WEN Y G, GUAN K, et al. Energy-optimal mobile cloud computing under stochastic wireless channel[J]. IEEE Transactions on Wireless Communications, 2013, 12(9): 4569-4581.

[12] MAO Y Y, ZHANG J, LETAIEF K B. Dynamic computation offloading for mobile-edge computing with energy harvesting devices[J]. IEEE Journal on Selected Areas in Communications, 2016, 34(12): 3590-3605.

[13] YAN J, BI S Z, ZHANG Y J A. Offloading and resource allocation with general task graph in mobile edge computing: a deep reinforcement learning approach[J]. IEEE Transactions on Wireless Communications, 2020, 19(8): 5404-5419.

[14] WANG F, XU J, WANG X, et al. Joint offloading and computing optimization in wireless powered mobile-edge computing systems[J]. IEEE Transactions on Wireless Communications, 2018, 17(3): 1784-1797.

[15] LI X, BI S Z, QUAN Z, et al. Online cognitive data sensing and processing optimization in energy-harvesting edge computing systems[J]. IEEE Transactions on Wireless Communications, 2022, 21(8): 6611-6626.

[16] WANG Y T, SHENG M, WANG X J, et al. Mobile-edge computing: partial computation offloading using dynamic voltage scaling[J]. IEEE Transactions on Communications, 2016, 64(10): 4268-4282.

[17] HE X Y, XING H, CHEN Y, et al. Energy-efficient mobile-edge computation offloading for applications with shared data[C]//Proceedings of 2018 IEEE Global Communications Conference. Piscataway: IEEE Press, 2018: 1-6.

[18] HU X Y, WONG K K, YANG K. Wireless powered cooperation-assisted mobile edge computing[J]. IEEE Transactions on Wireless Communications, 2018, 17(4): 2375-2388.

[19] HE B Q, BI S Z, XING H, et al. Collaborative computation offloading in wireless powered mobile-edge computing systems[C]//Proceedings of 2019 IEEE Globecom Workshops. Piscataway: IEEE Press, 2019: 1-7.

[20] YOU C S, HUANG K B. Exploiting non-causal CPU-state information for energy-efficient mobile cooperative computing[J]. IEEE Transactions on Wireless Communications, 2018, 17(6): 4104-4117.

[21] DONG X Q, LI X H, YUE X W, et al. Performance analysis of cooperative NOMA based intelligent mobile edge computing system[J]. China Communications, 2020, 17(8): 45-57.

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

[23] BI S Z, ZHANG Y J. Computation rate maximization for wireless powered mobile-edge computing with binary computation offloading[J]. IEEE Transactions on Wireless Communications, 2018, 17(6): 4177-4190.

[24] FENG H, GUO S T, YANG L, et al. Collaborative data caching and computation offloading for multi-service mobile edge computing[J]. IEEE Transactions on Vehicular Technology, 2021, 70(9): 9408-9422.

[25] YU S, DAB B, MOVAHEDI Z, et al. A socially-aware hybrid computation offloading framework for multi-access edge computing[J]. IEEE Transactions on Mobile Computing, 2020, 19(6): 1247-1259.

[26] HE J Y, ZHANG D, ZHOU Y Z, et al. A truthful online mechanism for collaborative computation offloading in mobile edge computing[J]. IEEE Transactions on Industrial Informatics, 2020, 16(7): 4832-4841.

[27] MAO Y Y, YOU C S, ZHANG J, et al. A survey on mobile edge computing: the communication perspective[J]. IEEE Communications Surveys & Tutorials, 2017, 19(4): 2322-2358.

[28] NGUYEN T T, HA V N, LE L B, et al. Joint data compression and computation offloading in hierarchical fog-cloud systems[J]. IEEE Transactions on Wireless Communications, 2020, 19(1): 293-309.

[29] BERTHOLD T. Heuristic algorithms in global MINLP solvers[D]. Berlin: Technical University of Berlin, 2014.

[30] BOYD S, VANDENBERGHE L. Convex optimization[M]. Cambridge: Cambridge University Press, 2004.

[31] MEHROTRA S. On the implementation of aprimal-dual interior point method[J]. SIAM Journal on Optimization, 1992, 2(4): 575-601.

[32] HUANG L, BI S Z, ZHANG Y J A. Deep reinforcement learning for online computation offloading in wireless powered mobile-edge computing networks[J]. IEEE Transactions on Mobile Computing, 2020, 19(11): 2581-2593.

[33] LIN B, LIN X H, ZHANG S L, et al. Computation task scheduling and offloading optimization for collaborative mobile edge computing[C]//Proceedings of 2020 IEEE 26th International Conference on Parallel and Distributed Systems. Piscataway: IEEE Press, 2020: 728-734.

[作者简介]

李贤（1988− ），男，博士，深圳大学电子与信息工程学院副研究员，主要研究方向为移动边缘计算、无线供电通信等系统的性能优化设计。

毕宿志（1987– ），男，博士，深圳大学电子与信息工程学院副教授，主要研究方向为无线通信网络资源管理与调度优化。

林彬（1997– ），男，深圳大学电子与信息工程学院硕士生，主要研究方向为移动边缘计算系统的优化设计。

曾泓儒（1996– ），男，深圳大学电子与信息工程学院硕士生，主要研究方向为移动边缘计算系统的优化设计。

林晓辉（1975– ），男，博士，深圳大学电子与信息工程学院教授，主要研究方向为无人机通信网络的优化设计。