深空环境中基于云边端协同的任务卸载方法

尚玉叶，袁家斌

引用本文

尚玉叶，袁家斌. 深空环境中基于云边端协同的任务卸载方法[J]. 计算机科学, 2023, 50(2): 80-88.

SHANG Yuye, YUAN Jiabin. Task Offloading Method Based on Cloud-Edge-End Cooperation in Deep Space Environment [J]. Computer Science, 2023, 50(2): 80-88.

计算机科学 COMPUTER SCIENCE
http://www.jsjkx.com
DOI:10.11896/jsjkx.220800156

深空环境中基于云边端协同的任务卸载方法

尚玉叶　袁家斌
南京航空航天大学计算机科学与技术学院　南京 211106
(shangyuye351@nuaa.edu.cn)

摘　要　深空探测是当今世界航天任务的重要领域,深空探测自主技术对未来进行大规模的深空探测具有重大意义。由于深空环境复杂且未知,通信时延长,星上计算资源有限,深空探测自主技术面临严峻挑战。针对此问题,提出了一种面向深空探测任务的数字孪生云边端协同框架,通过云边端协同的任务卸载,为深空探测自主技术提供更加高效的资源服务。首先将复杂深空探测任务分解为多个具有依赖关系的子模块,然后在虚拟空间层分别建立环绕器覆盖时间模型、协同计算模型和模块依赖模型,最后基于以上模型构建了相应的目标优化问题。优化目标是在模块依赖、环绕器的有效通信服务时间以及着陆巡视器发射功率控制约束条件下,最小化着陆巡视器完成深空探测任务的能耗和时间。为了解决该优化问题,提出了一种自适应遗传算法,以确定最优的执行策略交由物理空间层的着陆巡视器执行。仿真结果表明,所提出的自适应遗传算法可以有效减少任务完成时间和能耗。此外,将所提的云边端协同计算模式与另外3种计算模式进行了对比,结果表明,在完成相同目标的情况下,所提的云边端协同框架具有更高的资源利用率。

关键词:深空探测;云边端协同;自适应遗传算法;数字孪生;任务卸载;任务依赖
中图法分类号　TP393

Task Offloading Method Based on Cloud-Edge-End Cooperation in Deep Space Environment

SHANG Yuye and YUAN Jiabin
School of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing 211106, China

Abstract　Deep space exploration is a significant area of space missions in the modern world, and future large-scale deep space exploration will be greatly impacted by autonomous deep space exploration technologies. The autonomous technology of deep space exploration faces severe challenges because of the complicated and uncharted deep space environment, lengthy deep space communication time, and constrained on-board computing capacity. To address this issue, a cloud-edge-end cooperation architecture for deep space exploration tasks using digital twins is developed, which can offer more efficient resource services for deep space exploration autonomous technologies. Firstly, the complex deep space exploration task is decomposed into multiple sub-modules with dependencies. Secondly, the orbiter coverage time model, the collaborative computing model, and the task dependency model are established in the virtual space layer. Finally, based on the aforementioned models, the corresponding target optimization problem is proposed. The optimization objective is to minimize the energy consumption and time of the landing rover for completing the deep space exploration mission under the constraints of module dependence, the effective communication service time of the orbiter and the transmit power control of the landing rover. In order to solve this optimization problem, an adaptive genetic algorithm is proposed, so that the optimal execution strategy for the landing rover in the physical space layer can be determined. Simulation results show that the proposed adaptive genetic algorithm can effectively reduce the mission completion time and energy consumption. Additionally, the proposed cloud-edge-end cooperation computing model is contrasted with the other three computing models, and the results reveal that, when it is used to achieve the same objective, the proposed cloud-edge-end cooperation framework has a greater resource utilization.

Keywords　Deep space exploration, Cloud-Edge-End cooperation, Adaptive genetic algorithm, Digital twin, Task offloading, Task dependency

---

到稿日期:2022-08-16　返修日期:2022-11-30
基金项目:国家重点研发计划(2017YFB0802303);国家自然科学基金(62076127)
This work was supported by the National Key Research and Development Program of China(2017YFB0802303) and National Natural Science Foundation of China(62076127).
通信作者:袁家斌(jbyuan@nuaa.edu.cn)

1 引言

深空探测是人类航天技术发展的高级阶段，指对地外天体以及空间进行的探测活动。目前探测任务依靠地面测控系统遥操作探测器完成，而未来有望进行载人行星探测。相较于近地空间，探测器在执行深空任务时面临更为复杂且恶劣的空间环境，如极端温度环境、辐射环境、宇宙尘等，同时，近地空间与深空有着巨大的空间尺度差异。这一系列的因素都导致探测器无法直接采用近地卫星的地面测控系统，需要考虑深空环境带来的一系列额外约束，如指令信号时延、空间天体遮挡信号、数据传输码率、信号衰减等。

由于在深空探测任务的完成过程中，无法采用传统的天地大回路式控制模式对探测器进行实时测控，这使得深空探测器系统必须具备随空间环境变化实时调整自身状态的能力。深空探测器系统技术的目标是构建一套无地面测控系统参与的在轨运行自主管理系统，包括自主导航技术、自主制导技术、自主任务规划技术、自主故障检测技术。目前大部分深空探测器仅仅是实现了部分子系统的自主运行。以火星探测为例，其着陆巡视器计算资源有限，现有的自主控制技术方法难以同时满足高精度、高时效、低能耗与安全性，只能在短时间内不依赖于地面测控系统自主工作。因此，亟须研究一种新的方案辅助着陆巡视器自主管理系统高效完成深空探测任务。

数字孪生(Digital Twin，DT)利用传感器可以全面、准确、动态地反映深空探测器的状态变化，帮助组织监控操作，实施预测性维护和改进过程。针对着陆巡视器资源有限的问题，本文结合数字孪生的优势，提出了一种基于云边端协同的深空任务卸载方法，以辅助着陆巡视器自主管理系统高效完成深空探测任务。

本文的主要贡献如下：

(1)提出了一种面向深空探测任务的数字孪生云边端协同框架，在虚拟空间层利用着陆巡视器、环绕器与地球云中心三端协同的计算资源，决策其最优执行策略返回给物理空间层的着陆巡视器。

(2)考虑到模块依赖、环绕器的服务时间和功率控制的约束，将最小化着陆巡视器的任务完成能量消耗和时间的均衡问题描述为一个混合非线性规划问题，并提出了一种自适应遗传算法求解该问题。

(3)进行了大量仿真实验以验证所提框架的性能。实验结果表明所提出的自适应遗传算法优于传统的4种智能优化算法。此外，将云边端协同计算模式与另外3种计算模式进行对比，验证了所提出的云边端协同计算模式的优越性。

2 相关工作

2.1 数字孪生

近年来，数字孪生技术成为了国内外学术界和工业界的研究热点。数字孪生的雏形概念最早由美国密歇根大学于2003年提出，随后NASA和美国空军合作丰富了其中的概念，并正式提出“数字孪生”概念[1-2]。之后国内外研究者对数字孪生概念进行了广泛的讨论，北京航空航天大学陶飞研究团队提出了数字孪生五维模型并探索了五维模型在10个领域的应用[3]，其中五维模型具体包括物理实体、虚拟实体、孪生数据、连接和接口以及服务[4]。当前工业界主要以数字孪生五维模型作为技术应用的指导。从本质上来说，数字孪生技术是将真实的物理实体和环境实时映射到虚拟空间[5]，在虚拟空间中根据虚拟实体提供的实时数据以及运行智能学习算法帮助物理实体做出更准确、更优的决策，降低用户决策的资源消耗[6]。文献[7]建立了电池的数字孪生模型，打破了常规电池管理的计算能力和存储空间限制。基于此，数字孪生为深空探测任务的模拟分析与规划这一复杂问题提供了新的思路。

已有部分研究工作将数字孪生与边缘计算网络相结合，以构建数字孪生边缘网络(Digital Twin Edge Network，DTEN)。文献[8]提出了一个允许区块链授权的数字孪生无线网络边缘计算联邦学习框架，以缓解终端用户和边缘服务器之间不可靠的长距离通信，从而减少通信负载以及降低数据泄露风险。文献[9]通过边缘服务器的数字孪生来估计边缘服务器的状态，全局系统的数字孪生为深度强化学习提供训练数据并实时监控整个系统的状态，降低用户的卸载时延。

基于上述数字孪生的优势，本文将数字孪生应用到深空探测领域，提出了第一个工作内容：建立着陆巡视器、环绕器、地球云中心以及全局系统的数字孪生，在虚拟空间层内分析与预测着陆巡视器的状态，辅助着陆巡视器自主完成深空探测任务。

2.2 云边端协同计算

云边端协同一般指计算资源丰富的中央云、边缘服务器、移动终端通过协同的方式满足不同类型的应用请求，主要包括数据协同、智能协同、服务协同和资源协同。数据协同，即边缘服务器将终端设备的数据预处理后传送给云中心，云中心进行数据分析与数据挖掘。文献[10]通过终端设备生成公钥和私钥对，采用可搜索的公钥加密技术，在保证数据安全的同时降低了通信开销与计算成本；文献[11]结合边缘计算的实时性和云计算处理复杂问题的能力，通过数据协同、信息协同和知识协同来提高机床的智能化程度；文献[12]提出了一种面向物联网网络中实时数据分析的云边端协同框架，提升了实时数据的处理速度。智能协同，即边缘服务器与云中心分别负责深度学习模型的推理与训练，从而实现分布式智能。文献[13]提出了一种基于卷积神经网络的云边端协作框架，通过共享云端的最小层协助边缘端，减少了服务的平均响应时间，提高了模型的准确率。服务协同，即边缘计算服务与云服务的协同。文献[14]将服务部署问题转化为约束满足问题，通过马尔可夫求得最佳分配策略，提高了资源利用率。资源协同，即边缘服务器与云中心为终端设备提供计算能力、存储资源以及资源管理策略。文献[15]提出了一种基于Docker的云边端协同任务卸载框架，提高了计算任务的执行效率和各设备的资源利用率；文献[16]提出了一种面向云边端协同机制下的计算卸载和服务缓存策略，以实现最大化网络运营商的收益目标；文献[17]考虑了任务依赖需求和任务的完成截止时间，提出的随机映射协同任务计算卸载算法能够有效降低物联网传感器的能耗；文献[18]将移动设备的部分任务卸载

到边缘节点和云服务器上处理，联合优化通信和计算资源分配问题，以最小化所有移动设备的加权和时延。

资源协同在卫星网络方面已有部分研究，考虑到卫星因自身部署的特点，星上资源受限，文献[19-21]利用了一种以用户、卫星网络和地面网络为主的星地协同网络通信架构，对分配决策问题和资源分配问题进行了联合优化，降低了系统时延与能耗。

由于着陆巡视器资源有限，因此本文将云边端协同应用到深空探测领域，提出了本文的第二个工作内容：通过云边端协同计算模式，即着陆巡视器、环绕器、地球云中心资源协同，联合优化计算卸载和功率控制，均衡着陆巡视器的能耗和任务完成时间，以达到用最小化成本完成深空探测任务的目的。

3 系统模型及问题描述

3.1 系统模型

本文基于传统的三维数字孪生模型构建了面向深空探测任务的数字孪生云边端协同框架，如图 1 所示，包括物理空间层、虚拟空间层以及二者之间的连接。物理空间层主要包括执行深空探测任务的着陆巡视器、作为边缘服务器的环绕器以及地球云中心。不同类型的传感器将包括设备电量、存储状态、可用计算资源等在内的各设备状态参数同步到虚拟空间层，同时维护着陆巡视器、环绕器与地球云中心彼此之间的通信链路映射，及时更新三者之间的通信状态。虚拟空间层主要是通过物理空间层提供的设备状态参数来构建设备和全局系统的虚拟模型，预测任务执行过程中对应设备的状态，并使用智能算法模拟获得最优的执行方案，然后返回给物理空间层的着陆巡视器执行。物理空间层各设备在具体的执行过程中将执行情况及时反馈给虚拟模型，虚拟空间层根据具体执行结果和预期结果之间的差别对执行方案进行修正。如此不断迭代，最终实现整个深空探测任务的最优化执行。

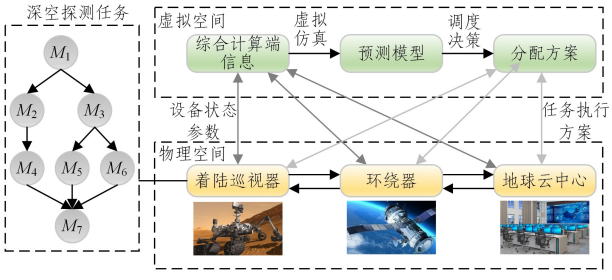

图 1 面向深空探测任务的数字孪生云边端协同框架

Fig. 1 Digital twin cloud-edge-end cooperation framework for deep space exploration task

以着陆巡视器导航点自主高效移动感知的深空任务为例，完成该任务主要包括图像采集、图像预处理、视觉定位、地形重构、中间点定位、障碍物检测和局部感知这 7 个关键步骤，且各步骤之间存在数据依赖。因此，可以将深空探测任务分解为多个可独立执行的子模块分步执行，本文使用 N 个节点构成的 DAG 图 $G=\langle M,E\rangle$ 来表示子模块之间的依赖关系，$M=\{M_1,M_2,\cdots,M_N\}$ 表示可独立执行的子模块集合，N 表示子模块的个数，如图 2 所示。由于部分模块必须在着陆巡视器执行，例如深空任务中的样本采集操作等，因此可以使用变量 isLoad 来判断该子模块是否只能在着陆巡视器本地执行。由于子模块间存在数据依赖关系，即子模块的执行需要其他子模块生成的结果数据，因此将一个可独立执行模块记为 M_i，$i\in N'$，$N'=\{1,2,\cdots,N\}$ 时，子模块 M_i 的计算数据量用 D_i 表示，子模块 M_i 依赖前驱子模块 M_j 的结果数据量表示为 $data_{ji}$，执行子模块 M_i 需要的 CPU 周期数为 X_i。

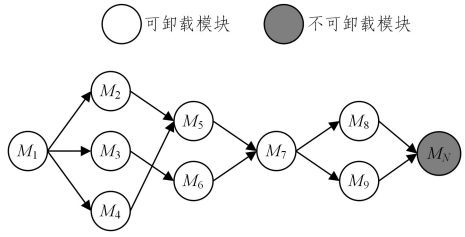

图 2 深空探测任务的模块依赖图

Fig. 2 Module dependency graph for deep space missions

由图 1 可知，DAG 中的 N 个独立子模块根据虚拟空间层的最优决策方案可以选择在着陆巡视器本地计算，或将子模块卸载到环绕器或地球云中心执行。环绕器覆盖时间模型、数字孪生模型、协同计算模型和模块依赖模型的形式化描述如下。

3.1.1 环绕器覆盖时间模型

由于环绕器的移动性，子模块卸载至环绕器或者地球云中心执行，均需在环绕器覆盖时间内完成子模块的数据传输。因此，根据运动物理学理论，环绕器提供有效通信服务的时间 T 计算式如下：

$$T=\frac{L}{v} \tag{1}$$

其中，v 表示环绕器的环绕速度，L 表示环绕器与着陆巡视器可通信的弧长。

如图 3 所示，环绕器与着陆巡视器可通信的弧长 L 与环绕器和着陆器的空间位置关系紧密相关，相关计算式如下：

$$L=2\cdot(R+H)\cdot\gamma \tag{2}$$

$$\gamma=\arccos\left(\frac{R}{R+H}\cdot\cos\alpha\right)-\alpha \tag{3}$$

$$\alpha=\arccos\left(\frac{R+H}{S}\cdot\sin\gamma\right) \tag{4}$$

$$S=\sqrt{R^2+(R+H)^2-2\cdot R\cdot(R+H)\cdot\cos\gamma} \tag{5}$$

其中，γ 表示着陆巡视器与环绕器的地心夹角，α 表示着陆巡视器和环绕器的仰角，R 表示火星的半径，H 表示环绕器距离火星表面的高度，S 表示环绕器与着陆巡视器的距离。

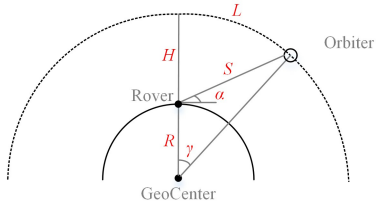

图 3 环绕器与着陆巡视器的空间几何关系

Fig. 3 Space geometric relationship between rover and orbiter

3.1.2 数字孪生模型

由于深空环境中数据传输的距离较远，传感器不能实时地将设备信息与状态反馈到虚拟空间层，即数字孪生的数据更新存在一定延迟。因此，虚拟空间层设备的预估计算能力

与物理空间设备的真实计算能力之间存在偏差。本文使用设备计算能力的偏差 $\hat{f}$ 来描述虚拟空间层模拟预测结果和物理空间层真实结果之间的偏差。本文系统的数字孪生 DT 可表示为：

$$DT=\Theta(f,\hat{f}) \tag{6}$$

其中，f 表示虚拟空间层设备的预估计算能力。

假设设备完成模块 M_i 需要 CPU 的周期数为 X_i，则虚拟空间层的估计时延与真实计算时延的时延偏差计算式如下：

$$\hat{T}=\frac{X_i}{f+\hat{f}}-\frac{X_i}{f}=-\frac{X_i\hat{f}}{f(f+\hat{f})} \tag{7}$$

3.1.3 协同计算模型

基于前文阐述的云边端协同框架，每个子模块 M_i 的执行方案主要有 3 种：着陆巡视器的本地计算方案、卸载至环绕器的边缘服务器计算方案，以及卸载至地球云中心的计算方案。下文分别对模块 M_i 在 3 种不同的执行方案下的时延模型与能耗模型进行描述。

(1)本地计算方案。模块 M_i 在着陆巡视器本地处理的实际时延 T_i^{compL} 由虚拟空间层着陆巡视器计算的估计时延 T_i^L 和着陆巡视器计算的时延偏差 $\hat{T}_i^L$ 组成，其计算式如下：

$$T_i^{TL}=T_i^L+\hat{T}_i^L \tag{8}$$

$$T_i^L=\frac{X_i}{f_i^L} \tag{9}$$

$$\hat{T}_i^L=\frac{-X_i\hat{f}_i^L}{f_i^L(f_i^L+\hat{f}_i^L)} \tag{10}$$

其中，f_i^L 表示在处理模块 M_i 时，虚拟空间层着陆巡视器的预估计算能力，$\hat{f}_i^L$ 表示虚拟空间层着陆巡视器的预估计算能力与物理空间着陆巡视器的真实计算能力的偏差。

模块 M_i 在着陆巡视器计算的能耗计算式如下：

$$E_i^L=\varepsilon\times(f_i^L+\hat{f}_i^L)^2\times X_i \tag{11}$$

其中，ε 表示有效开关电容，其大小取决于芯片架构。

(2)边缘服务器计算方案。该执行方案主要考虑模块 M_i 卸载至环绕器，传输过程中的可达数据率为 R_i，其计算式如下：

$$R_i=B\log 2\left(1+\frac{g_0\ (L_0/S)^\theta P_i}{N_0B}\right) \tag{12}$$

其中，B 表示着陆巡视器与环绕器之间的通信带宽，P_i 表示着陆巡视器卸载模块 M_i 的发射功率，g_0 表示路径损失常数，θ 表示路径损失指数，L_0 表示参考距离，N_0 表示环绕器接收端的噪声功率谱密度。由于着陆巡视器与环绕器距离较远，本文考虑了信号在深空环境下的传播时延。因此，模块 M_i 卸载到环绕器计算的执行时间 T_i^{TE} 由传播时延 T_i^{travel}、传输时延 T_i^{transE} 和计算时延 T_i^{compE} 组成，计算式如下：

$$T_i^{\mathrm{TE}}=T_i^{\mathrm{travel}}+T_i^{\mathrm{transE}}+T_i^{\mathrm{compE}} \tag{13}$$

$$T_i^{\mathrm{travel}}=\frac{S}{c} \tag{14}$$

$$T_i^{\mathrm{transE}}=\frac{D_i}{R_i} \tag{15}$$

$$T_i^{\mathrm{compE}}=\frac{X_i}{f_i^e}+\hat{T}_i^E \tag{16}$$

其中，S 表示着陆巡视器和环绕器的距离，c 表示无线电磁波的传播速度，f_i^e 表示处理模块 M_i 时，虚拟空间层环绕器的预估计算能力，$\hat{T}_i^E$ 是模块 M_i 根据数字孪生模型计算出的环绕器计算时延偏差。

假设环绕器的计算资源充足，那么在边缘服务器计算方案中，着陆巡视器的能耗主要来源于将模块 M_i 卸载到环绕器的通信能耗，计算式如下：

$$E_i^E=P_i\times\frac{D_i}{R_i} \tag{17}$$

(3)云中心计算方案。该执行方案中，着陆巡视器的模块 M_i 卸载到地球云中心，需要通过环绕器的转发，因此模块 M_i 卸载到地球云中心计算的执行时间 T^{TC_i} 由传播时延 T_i^{travel}、着陆巡视器到环绕器的传输时延 T_i^{transE}、环绕器到地球云中心的传输时延 T_i^{transC} 和计算时延 T_i^{compC} 组成，计算式如下：

$$T_i^{\mathrm{TC}}=T_i^{\mathrm{travel}}+T_i^{\mathrm{transE}}+T_i^{\mathrm{transC}}+T_i^{\mathrm{compC}} \tag{18}$$

$$T_i^{\mathrm{transC}}=\frac{D_i}{r} \tag{19}$$

$$T_i^{\mathrm{compC}}=\frac{X_i}{f_i^c}+\hat{T}_i^C \tag{20}$$

其中，r 表示环绕器与地球云中心之间的传输速率，由于环绕器与地球之间的传输速率对模型的建立与求解影响很小，因此，为了降低模型的复杂度，将其设为固定值；f_i^c 表示处理模块 M_i 时，虚拟空间层地球云中心的预估计算能力；$\hat{T}_i^C$ 是模块 M_i 根据数字孪生模型求得的地球云中心计算时延偏差。

此外，可以观察到，无论模块 M_i 卸载到环绕器还是地球云中心，着陆巡视器的能耗都主要来源于将模块 M_i 卸载到环绕器的通信能耗，因此，着陆巡视器的能耗保持不变，云中心计算方案中能耗 E_i^C 为：

$$E_i^C=P_i\times\frac{D_i}{R_i} \tag{21}$$

因此，着陆巡视器完成深空探测任务，所有子模块需要的能耗如式(22)所示：

$$E_N=\sum_{i=1}^{N}(x_iE_i^L+y_iE_i^E+z_iE_i^C) \tag{22}$$

其中，$x_i,y_i,z_i\in\{0,1\}$，$x_i+y_i+z_i=1$，表示每个子模块只能在一个位置执行。

3.1.4 模块依赖模型

模块间的依赖关系主要是由模块的完成时间反映的，为了更好地描述模块之间的依赖关系，本文对模块 M_i 的完成时间模型进行了定义，如式(23)所示：

$$AFT_i=AST_i+B_i \tag{23}$$

其中，AST_i 为模块 M_i 的开始执行时间，AFT_i 为模块 M_i 的完成时间，B_i 表示模块 M_i 自身计算数据 D_i 的执行时间，计算式如下：

$$Bi=xiT_i^{\mathrm{TL}}+yiT_i^{\mathrm{TE}}+ziT_i^{\mathrm{TC}} \tag{24}$$

其中，x_i,y_i,z_i 是二进制变量，且三者的关系为 $x_i+y_i+z_i=1$，代表模块 M_i 只能在一个位置执行，$x_i=1$ 表示模块 M_i 的执行位置是着陆巡视器，$y_i=1$ 表示模块 M_i 的执行位置是环绕器，$z_i=1$ 表示模块 M_i 的执行位置是地球云中心。

模块 M_i 的开始执行时间 AST_i 取决于前驱模块 M_j 的完成时间 AFT_j，两者的关系如下：

$$AST_i=\begin{cases}\max\limits_{j\in pre(i)}\{AFT_j+C_{ji}\} & ,pre(i)\neq\varnothing\\ 0 & ,pre(i)=\varnothing\end{cases} \tag{25}$$

其中，$pre(i)$ 为模块 M_i 的前驱模块集合，$\max\{j\in pre(i):AFT_j+C_{ji}\}$ 为所有的前驱模块执行完成并将计算结果数据传输给模块 M_i 的时间；C_{ji} 为前驱模块 M_j 的数据结果到达模块 M_i 的传输时间，其值与两者之间传输数据量 $data_{ji}$ 以及执行位置有关，具体的计算式如下：

$$C_{ji}=\begin{cases}0, & \boldsymbol{v}_j-\boldsymbol{v}_i=(0,0,0)\\ T_i^{\text{travel}}+\dfrac{data_{ji}}{R_j}, & \boldsymbol{v}_j-\boldsymbol{v}_i=(1,-1,0)\\ T_i^{\text{travel}}+\dfrac{data_{ji}}{R_i}, & \boldsymbol{v}_j-\boldsymbol{v}_i=(-1,1,0)\\ T_i^{\text{travel}}+\dfrac{data_{ji}}{R_j}+\dfrac{data_{ji}}{r}, & \boldsymbol{v}_j-\boldsymbol{v}_i=(1,0,-1)\\ T_i^{\text{travel}}+\dfrac{data_{ji}}{R_i}+\dfrac{data_{ji}}{r}, & \boldsymbol{v}_j-\boldsymbol{v}_i=(-1,0,1)\\ T_i^{\text{travel}}+\dfrac{data_{ji}}{r}, & \boldsymbol{v}_j+\boldsymbol{v}_i=(0,1,1)\end{cases} \tag{26}$$

其中，模块 M_i 与模块 M_j 是前驱后继关系；向量 $\boldsymbol{v}_j$ 和 $\boldsymbol{v}_i$ 分别代表前驱模块 M_j 和当前模块 M_i 的执行位置向量。$\boldsymbol{v}_i$ 由二进制变量 x_i,y_i,z_i 构成，关系为 $\boldsymbol{v}_i=(x_i,y_i,z_i)$，$x_i,y_i,z_i\in\{0,1\}$，$x_i+y_i+z_i=1$，$i\in N'$。前驱模块 M_j 的结果数据到达模块 M_i 的传输时间 C_{ji} 根据模块执行位置可以划分为以下 6 种情况：1)前驱模块 M_j 与当前模块 M_i 在相同的位置执行；2)前驱模块 M_j 在着陆器执行，当前模块 M_i 在环绕器执行；3)前驱模块 M_j 在环绕器执行，当前模块 M_i 在着陆器执行；4)前驱模块 M_j 在着陆器执行，当前模块 M_i 在地球云中心执行；5)前驱模块 M_j 在地球云中心执行，当前模块 M_i 在着陆器执行；6)模块 M_i 与模块 M_j 分别在环绕器与地球云中心完成任务。

3.2 问题描述

在环绕器有效通信服务时间和着陆巡视器发射功率约束条件下，本文通过联合优化模块的卸载决策和着陆巡视器的发射功率来最小化着陆巡视器完成深空探测任务的能耗和时间。具体的优化问题规划如下：

$$\begin{aligned}&\Omega(X,Y,Z,P)=\min_{X,Y,Z,P}\beta E_N+(1-\beta)AFT_N\\ &\text{s.t. C1}:x_i+y_i+z_i=1,\forall i\in N'\\ &\quad\text{C2}:x_i\in\{0,1\},y_i\in\{0,1\},z_i\in\{0,1\}\\ &\quad\text{C3}:0\leqslant P_i\leqslant P_{\max},i\in N'\\ &\quad\text{C4}:AFT_i\leqslant T,i\in N'\\ &\quad\text{C5}:1/N\sum_{i=1}^{N}P_i\leqslant\bar{P}\end{aligned} \tag{27}$$

该问题的优化变量分别是 $X=\{x_1,x_2,\cdots,x_N\}$，$Y=\{y_1,y_2,\cdots,y_N\}$，$Z=\{z_1,z_2,\cdots,z_N\}$，$P=\{P_i,i\in N'\}$。β 表示时延和能耗在目标函数中所占的比重，取值范围为 $[0,1]$；$\bar{P}$ 是着陆巡视器的功率门限。C1 和 C2 表示每个模块只能在一个位置执行；C3 表示着陆巡视器的每个模块的非负发射功率不能大于最大发射功率 $P_{\max}$；C4 表示着陆巡视器的每个模块的完成时间必须在环绕器的通信服务时间 T 内；C5 表示着陆巡视器模块的平均发射功率要低于系统设置的功率门限。

式(27)是非线性混合整数问题，是非凸优化问题，其复杂度随着优化参量规模的增大而增加，通过枚举所有可能解来获得问题的最优解是不切实际且不可行的。因此，本文提出了自适应的遗传算法来解决该非线性混合整数问题。

4 自适应遗传算法

遗传算法(Genetic Algorithm，GA)起源于对生物系统进行的计算机模拟研究，它是一种模拟自然选择和自然遗传中发生的复制、交叉和变异等现象的随机全局搜索优化方法。该方法可以使群体进化到搜索空间中越来越好的区域，最后收敛到一群最适应环境的个体，从而求得问题的优质解。基于传统遗传算法，本文提出了自适应遗传算法(Adaptive Genetic Algorithm，AGA)，该算法主要对传统遗传算法中的交叉算子、变异算子以及精英保留策略进行改进，在提升群体多样性的同时保证算法的收敛性，使得其能更好地解决研究的目标问题。

4.1 参数编码和初始化

在本文的协同计算场景中，我们的目标是寻找模块最优的计算卸载策略以及最优的发射功率，使得着陆巡视器用更少的能耗和时间完成深空探测任务。因此，本文将模块的计算卸载策略和发射功率进行编码后作为自适应的遗传算法中的染色体个体。

每条染色体由 $2N$ 个基因构成，前 N 个基因代表模块的计算卸载策略，后 N 个基因代表着陆巡视器对模块的发射功率。为了简化编码，将模块 M_i 的执行位置向量 $\boldsymbol{v}_i$ 用变量 a_i 表示，其中 $a_i\in\{0,1,2\}$。每个模块经过卸载决策后可以选择着陆巡视器本地计算、传输到环绕器计算或传输到地球云中心计算。因此，前 N 个基因的编码值为 0，1，2；后 N 个基因的编码值为 0 到最大发射功率的实数值。

自适应遗传算法中的适应度用来衡量个体对环境的适应性，在进化的过程中通过适应度函数的值来判断并淘汰适应程度较差的个体，本文定义染色体个体 Ind 的适应度函数 $f(a_i,P_i)$ 为：

$$f(a_i,P_i)=-\Omega(X_i,Y_i,Z_i,P_i) \tag{28}$$

可以看出，染色体个体的适应度值与目标函数值成反比。

4.2 改进的交叉算子与变异算子

遗传算法通过交叉过程来迭代产生新的染色体，由于本文的染色体既包含离散编码又包含连续编码，因此染色体的交叉过程采用先拆分再合并的方式。如图 4 所示，前 N 个离散编码的基因采用两点交叉算子，用两个点切开父代染色体基因后，交换切出来的基因段；后 N 个实数连续编码的基因采用模拟二值交叉算子，子代染色体的基因可以通过以下公式得到：

$$\begin{cases}c^1=0.5*[(1+\lambda)\cdot Ind^1+(1-\lambda)\cdot Ind^2]\\ c^2=0.5*[(1-\lambda)\cdot Ind^1+(1+\lambda)\cdot Ind^2]\end{cases} \tag{29}$$

其中，λ 是分布因子，按照如下公式随机决定：

$$\lambda=\begin{cases}(2u)^{\frac{1}{1+\eta}}, & u\leqslant 0.5\\ \left(\dfrac{1}{2-2u}\right)^{\frac{1}{1+\eta}}, & u>0.5\end{cases} \tag{30}$$

其中，η 是一个自定义的参数，其值越大，产生的子代个体逼近父代个体的概率越大；u 是均匀分布的随机数。交叉概率表示交叉过程的执行概率，用 $CXPB$ 表示。交叉过程可以使得每次迭代进化后留下相对优良的基因，但易陷入局部最优解，因此需要引入变异算子，即每一个基因都按照一定的概率变异，变异概率用 $MUTPB$ 表示。

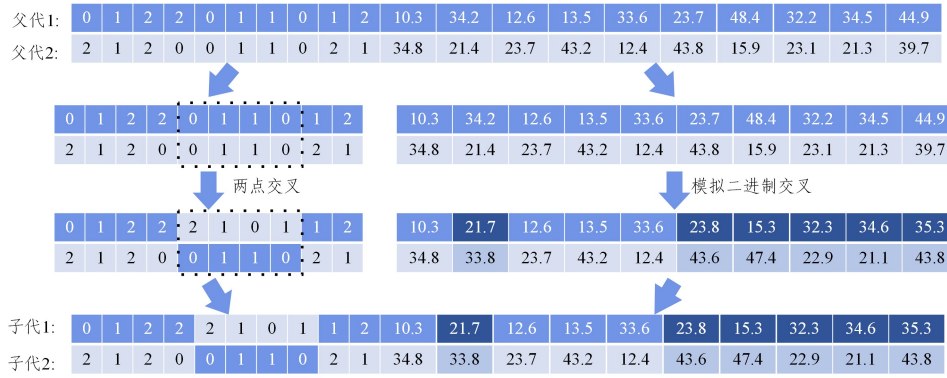

图 4 染色体交叉示意图

Fig. 4 Chromosome crossover diagram

随着算法进化迭代次数的增加，种群的搜索区域越来越小，种群的相似程度逐渐增加，因此需要动态地改变交叉概率与变异概率，使其自适应于算法的进化过程，提高算法的全局搜索能力和局部搜索能力。种群适应度的平均值通过个体适应度的期望 EX 来刻画，离散程度通过标准差 SD 来刻画，种群的相似程度 φ 与种群适应度平均值以及离散程度的关系如下：

$$\varphi=\frac{EX+1}{SD} \tag{31}$$

根据遗传算法交叉算子与变异算子的设计原则和变换规律，改进的自适应遗传算法的交叉算子与变异算子的动态调节公式分别为：

$$CXPB=0.6*\frac{1}{1+e^{-\frac{10}{\varphi}}}+0.3 \tag{32}$$

$$MUTPB=\frac{0.1}{5(1+e^{\frac{1}{\varphi}})} \tag{33}$$

4.3 精英保留策略

为了在提升群体多样性的同时保证算法的收敛性，本文提出的自适应遗传算法采用了精英保留策略，其核心思想是：每一次迭代过程中出现的适应度最高的基因个体不进行交叉变异操作，而直接复制替换下一代中的适应度最低的基因个体。具体操作步骤如下：

步骤 1 计算父代种群中个体适应度值的集合 F_1，并计算经过交叉、变异等操作后的个体适应度值的集合 F_2；

步骤 2 将集合 F_1 和集合 F_2 合并，并按照适应度值由高到低排序；

步骤 3 选取一定比例的较优适应度个体，将这部分的个体替代 F_2 种群中适应度较差的个体，从而形成新的种群个体，进入下一步迭代。

4.4 算法框架

本文算法设计了自适应的交叉算子和变异算子，对选中的个体分别进行两点交叉、模拟二进制交叉和动态变异，以保证算法具有较强的搜索能力。在自适应算法进化的过程中，使用精英保留策略保证算法的收敛性。自适应遗传算法的算法框架如算法 1 所示。

算法 1 自适应遗传算法(AGA)

输入：深空任务的 DAG 图矩阵

输出：种群最优个体 BestInd 的适应度值，模块执行方案和模块发射功率

初始化：最大迭代次数 K，种群规模 popsize，随机产生第一代种群；

1. while 迭代次数<K do
2. 根据式(28)计算父代种群中个体的适应度并排序；
3. while 子代种群个数< popsize do
4. 根据适应度值以比例选择法从父代种群中选择 2 个个体；
5. 根据式(32)计算父代种群的交叉概率 CXPB，对 2 个个体按交叉概率执行交叉操作；
6. 根据式(33)计算父代种群的变异概率 MUTPB，对 2 个个体按变异概率执行变异操作；
7. end while
8. 计算子代种群适应度并排序；
9. 父代种群和子代种群合并排序，适应度较高的个体代替子代种群适应度较低的个体，更新子代种群个体；
10. 迭代次数++；
11. end while
12. 输出最优个体 BestInd 与其对应的适应度值。

5 仿真与评估

5.1 实验设置

本文仿真实验基于 PyCharm 平台，用到的相关工具库包括 Python3.8 和 DEAP 智能优化算法库。相比其他的智能优化算法库，DEAP 智能优化算法库用法灵活，内部函数模块可自定义，适合本文的复杂问题模型求解。基于以上工具，本文模拟了一个云边端协同计算系统，该系统包含一个执行深空任务的着陆巡视器、一个环绕器和地球云中心。针对不同复杂度的深空任务，随机生成了具有 N 个模块节点的有向无环图作为系统的输入。仿真实验涉及的参数默认值参考文献[22-23]，并根据实际深空环境进行了相应的调整，具体如表 1

所列。为了降低结果的随机性，实验结果取 500 次模拟实验的平均值。

表 1 实验参数设定

Table 1 Experimental parameter setting

参数	数值
有效开关电容 ε	10^{-27}
着陆巡视器估计计算能力 f_i^l/GHz	0.1～0.5
环绕器的估计计算能力 f_i^e/GHz	5
地球云中心的估计计算能力 f_i^c/GHz	10
模块所需 CPU 资源 X_i/GHz	100～400
模块数据量 D_i/KB	100～400
路径损失常数 g_0/dB	−40
参考距离 L_0/km	8
环绕器接收端的噪声功率谱密度 N_0/(dBm/Hz)	−174
路径损失指数 θ	4
电磁波传播速度 c/m/s	3×10^8
着陆巡视器的最大发射功率 P_{max}/W	50
环绕器距地面高度 H/km	350
模块间传输数据量 $data_{ji}$/KB	10～40
模块数量 N	10
着陆器与环绕的仰角 α/°	20
环绕器的运行速度 v/(km/s)	7
数字孪生误差率	0.1
交叉概率 $CXPB$ 的初始值	0.8
变异概率 $MUTPB$ 的初始值	0.1
模拟二值交叉算子自定义的参数 η	1

仿真实验主要包括以下几个部分：首先，为确保 AGA 算法能呈现最好的性能，本文进行了多次迭代仿真，以确定自适应遗传算法涉及的关键参数，如种群规模；其次，为评估 AGA 算法针对本文优化问题的求解性能，将 AGA 算法与其他的卸载决策算法进行比较，主要包括传统的遗传算法、麻雀搜索算法（Sparrow Search Algorithm，SSA）[24]、鲸鱼优化算法（Whale Optimization Algorithm，WOA）[25]，以及粒子群优化算法（Particle Swarm Optimization，PSO）[26]；再次，考虑到深空任务的复杂程度不同，研究了深空任务的模块数量对系统性能的影响；然后，研究了在不同计算方案下着陆巡视器的最大发射功率对系统性能的影响，计算方案主要有以下 4 种。

(1)Local：所有子模块在着陆巡视器计算。

(2)Edge：所有子模块在环绕器计算。

(3)Cloud：所有子模块在地球云中心计算。

(4)Joint：所有子模块云边端协同计算。

最后，研究了数字孪生对系统性能的影响。

5.2 结果分析

5.2.1 种群规模对 AGA 算法性能的影响

种群规模的大小决定了算法的收敛速度与计算效率，规模过大则收敛速度慢且易陷入局部最优解，规模过小则会降低计算速度。因此，本文研究了种群规模大小对效用函数 Ω（优化问题的目标函数，即系统的成本）的影响。本文将种群规模分别设置为 10，20，…，100，最大终止迭代次数设置为 100，每个种群分别运行 50 次，并对效用函数取平均值，结果如图 5 所示。图中纵坐标为效用函数 Ω，可以看出，当种群规模为 40 时，效用函数最低。因此，本文将种群规模设置为 40，以使 AGA 算法在后续实验中能保持较好的性能。

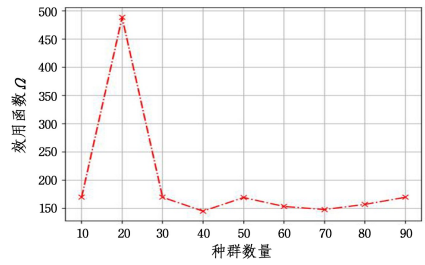

图 5 种群大小对效用函数 Ω 的影响

Fig. 5 Impact of the size of population on utility function Ω

5.2.2 权重因子对系统性能的影响

为研究问题模型中权重因子 β 对效用函数的影响，在实验过程中，系统处理随机生成的具有 10 个模块节点的有向无环图，随着权重因子的变化，效用函数值即优化问题的目标函数值 β 变化结果如图 6 所示。当 $\beta=0$ 时，效用函数代表对深空任务完成时间的目标函数；当 $\beta=1$ 时，效用函数代表对深空任务能耗的目标函数。由图可知，总体上效用函数值呈现先增后减的趋势，$\beta=0.9$ 时，效用函数值最低，此时系统的性能优劣主要取决于完成深空任务的能量消耗。

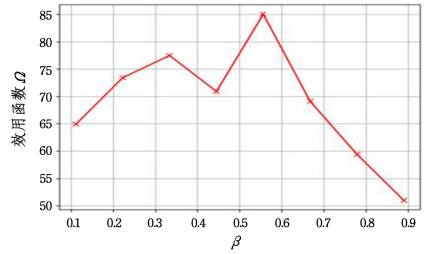

图 6 权重 β 对效用函数 Ω 的影响

Fig. 6 Impact of β on utility function Ω

5.2.3 不同卸载决策算法下系统性能对比

为证明 AGA 算法对本文优化问题具有更优的求解性能，将其与 GA 算法、SSA 算法、WOA 算法和 PSO 算法进行对比，算法收敛性能如图 7 所示。由图可知，PSO 算法过早陷入了局部最优解，SSA 算法过早地收敛，WOA 算法收敛较慢；AGA 算法与 GA 算法收敛速度适中，但 AGA 算法比 GA 算法的效用函数值低 20%左右，AGA 算法的寻优效果较好，求解性能更好。因此，从收敛速度和寻优效果的综合情况来看，AGA 算法在求解本文提出的问题模型方面具有一定的优越性。

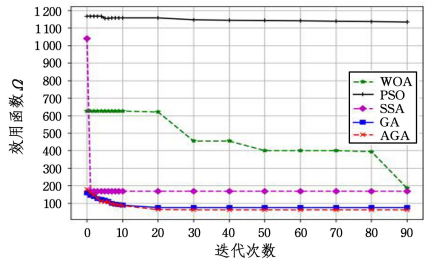

图 7 5 种算法的收敛性对比图

Fig. 7 Comparison of convergence of five algorithms

5.2.4 关键参数对系统性能的影响

首先，为验证不同模块数量的规模，即子任务数对系统

性能的影响，本文考虑动态调整模块数量以模拟不同复杂程度的深空任务。基于此，本文将 AGA 算法与另外 4 种算法进行对比，结果如图 8 所示。由图可知，随着模块数量的增加，模块自身数据与模块间数据的传输成本增加，执行完成所有模块的能耗增加，而系统成本中能耗占比大，因此效用函数值逐步递增，即系统成本缓慢增加。AGA 算法在模块数量相同的情况下，效用函数值一直保持最低的水平。因此，从整个系统处理不同的模块数量角度来说，AGA 算法同样具有一定的优越性，是更适合应用于本文优化问题的群智能优化算法。

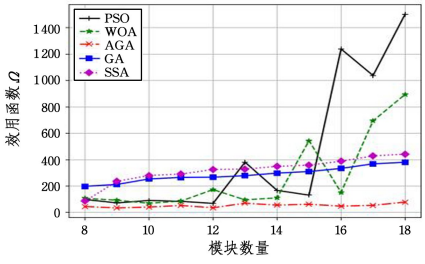

图 8 不同模块数量下 5 种算法的效用函数

Fig. 8 Utility function of five algorithms with different modules

此外，为研究着陆巡视器的最大发射功率对系统性能的影响，以及证明本文提出的云边端协同方案优于 Local 方案、Edge 方案和 Cloud 方案，本文对着陆巡视器的最大发射功率对效用函数的影响进行了分析，结果如图 9 所示。可以看出，随着着陆巡视器的最大发射功率的不断增加，Edge 方案、Cloud 方案和本文提出的云边端协同方案效用函数值缓慢增加，Local 方案的效用函数值一直维持在最高水平。在相同的最大发射功率下，云边端协同方案的效用函数远小于 Local 方案、Edge 方案和 Cloud 方案，消耗的系统成本更少。

随着最大发射功率的增大，着陆巡视器的可用功率范围变大，从而提高数据上传速率，深空任务的完成时间减少，但发射功率的提高同时会引起完成深空任务的能耗增加。这是因为本文效用函数中的权重值 $\beta=0.9$，完成深空任务的能耗占比远大于完成时间，所以系统的性能主要取决于完成深空任务的能量消耗。本文提出的 Joint 方案的效用函数值呈现缓慢增加的趋势，但其值远小于 Local 方案、Edge 方案和 Cloud 方案的效用函数值，验证了本文提出的云边端协同方案优于 Local 方案、Edge 方案和 Cloud 方案。

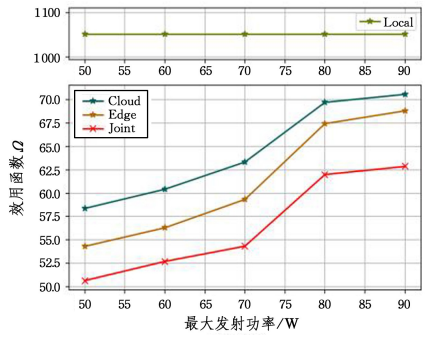

图 9 最大发射功率对效用函数 Ω 的影响

Fig. 9 Impact of the maximum transmitting power on Ω

5.2.5 数字孪生对系统性能的影响

为验证本文提出的面向深空探测任务的数字孪生云边端协同计算方案的优越性，本文将使用数字孪生技术的云边端协同计算方案与未使用数字孪生技术的云边端协同计算方案进行对比，以分析两者的系统成本开销与模块数量的关系，其中数字孪生误差率设置为 0.1，即 $\hat{f}=0.1f$，结果如图 10 所示。可以看出，系统成本随着模块数的增加而增加，且两者之间差距逐渐增大。使用数字孪生技术的云边端协同计算方案的系统成本开销总是小于未使用数字孪生技术的云边端协同计算方案的系统成本开销，即使用数字孪生技术辅助的云边端系统技术能获得更优的系统性能，验证了本文提出的面向深空探测任务的数字孪生云边端协同框架具有一定的优越性。

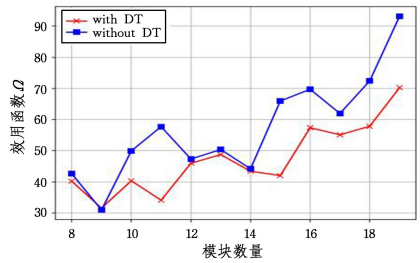

图 10 模块数与系统成本开销的关系

Fig. 10 Relationship between modules and cost

结束语 本文提出了一种数字孪生辅助的面向深空探测任务的云边端协同任务卸载方法。首先将复杂深空探测任务分解为多个具有依赖关系的子模块，然后在虚拟空间层分别建立环绕器覆盖时间模型、协同计算模型和模块依赖模型，最后基于以上模型构建了相应的目标优化问题。优化目标是在模块依赖、环绕器的有效通信服务时间以及着陆巡视器发射功率控制约束条件下，最小化着陆巡视器完成深空探测任务的能耗和时间。为了解决该优化问题，提出了一种自适应遗传算法，以使决策模块的最优执行策略交由物理空间层的着陆巡视器执行。实验结果表明，本文提出的自适应遗传算法优于传统遗传算法、麻雀算法、粒子群优化算法和鲸鱼算法。此外，将云边端协同计算方法与另外 3 种计算模型进行对比，验证了所提出的云边端协同计算方法的优越性。

本文只考虑了单个环绕器的情况，未来将考虑着陆巡视器可恢复能量、多个环绕器构成星间链路等问题，其中可能涉及区块链和深度强化学习技术。

参 考 文 献

[1] GRIEVES M W. Product lifecycle management: the new paradigm for enterprises[J]. International Journal of Product Development, 2005, 2(1/2): 71-84.

[2] GITHENS G. Product lifecycle management: driving the next generation of lean thinking by Michael Grieves[J]. Journal of Product Innovation Management, 2007, 24(3): 278-280.

[3] TAO F, LIU W R, ZHANG M, et al. Five-dimension digital twin model and its ten applications[J]. Computer Integrated Manufacturing Systems, 2019, 25(1): 1-18.

[4] XIANG F, ZHANG Z, ZUO Y, et al. Digital twin driven green material optimal-selection towards sustainable manufacturing [J]. Procedia Cirp, 2019, 81: 1290-1294.

[5] JING Q F, SHEN H L, YIN L. A ship digital twin framework based on virtual reality system[J]. Journal of Beijing Jiaotong University, 2020, 44(5): 117-124.
[6] WU Y, ZHANG K, ZHANG Y. Digital twin networks: A survey [J]. IEEE Internet of Things Journal, 2021, 8(18): 13789-13804.
[7] WANG Y, XU R, ZHOU C, et al. Digital twin and cloud-side-end collaboration for intelligent battery management system[J]. Journal of Manufacturing Systems, 2022, 62: 124-134.
[8] LU Y, HUANG X, ZHANG K, et al. Low-latency federated learning and blockchain for edge association in digital twin empowered 6G networks[J]. IEEE Transactions on Industrial Informatics, 2020, 17(7): 5098-5107.
[9] SUN W, ZHANG H, WANG R, et al. Reducing offloading latency for digital twin edge networks in 6G[J]. IEEE Transactions on Vehicular Technology, 2020, 69(10): 12240-12251.
[10] TAO Y, XU P, JIN H. Secure data sharing and search for cloud-edge-collaborative storage [J]. IEEE Access, 2019, 8: 15963-15972.
[11] LOU P, LIU S, HU J, et al. Intelligent machine tool based on edge-cloud collaboration [J]. IEEE Access, 2020, 8: 139953-139965.
[12] SHARMA S K, WANG X. Live data analytics with collaborative edge and cloud processing in wireless IoT networks[J]. IEEE Access, 2017, 5: 4621-4635.
[13] HUANG M, LIU W, WANG T, et al. A cloud-MEC collaborative task offloading scheme with service orchestration[J]. IEEE Internet of Things Journal, 2019, 7(7): 5792-5805.
[14] CHEN X, TANG S, LU Z, et al. iDiSC: A new approach to IoT-data-intensive service components deployment in edge-cloud-hybrid system[J]. IEEE Access, 2019, 7: 59172-59184.
[15] ZHANG W Z, YU J H. Task offloading stragegy in mobile egde computing Based on Cloud-Edge-End cooperation[J]. Journal of Computer Research and Development, 2022, 23(1): 1-14.
[16] FAN Q, LIN J, FENG G, et al. Joint service caching and computation offloading to maximize system profits in mobile edge-cloud computing[C]// 2020 16th International Conference on Mobility, Sensing and Networking (MSN). IEEE, 2020: 244-251.
[17] LIU F, HUANG Z, WANG L. Energy-efficient collaborative task computation offloading in cloud-assisted edge computing for IoT sensors[J]. Sensors, 2019, 19(5): 1105.
[18] REN J, YU G, HE Y, et al. Collaborative cloud and edge computing for latency minimization[J]. IEEE Transactions on Vehicular Technology, 2019, 68(5): 5031-5044.
[19] CUI G, LONG Y, XU L, et al. Joint offloading and resource allocation for satellite assisted vehicle-to-vehicle communication[J]. IEEE Systems Journal, 2020, 15(3): 3958-3969.
[20] WANG B, FENG T, HUANG D Y. A joint computation offloading and resource allocation strategy for LEO satellite edge computing system[C]// IEEE 20th International Conference on Communication Technology. 2020: 649-655.
[21] CUI G, LI X, XU L, et al. Latency and energy optimization for MEC enhanced SAT-IoT networks[J]. IEEE Access, 2020, 8: 55915-55926.
[22] TANG Q, FEI Z, LI B, et al. Computation offloading in leo satellite networks with hybrid cloud and edge computing[J]. IEEE Internet of Things Journal, 2021, 8(11): 9164-9176.
[23] NING Z, DONG P, KONG X, et al. A cooperative partial computation offloading scheme for mobile edge computing enabled Internet of Things[J]. IEEE Internet of Things Journal, 2018, 6(3): 4804-4814.
[24] XUE J, SHEN B. A novel swarm intelligence optimization approach: sparrow search algorithm[J]. Systems Science & Control Engineering, 2020, 8(1): 22-34.
[25] TANG C, SUN W, WU W, et al. A hybrid improved whale optimization algorithm[C]// 2019 IEEE 15th International Conference on Control and Automation (ICCA). IEEE, 2019: 362-367.
[26] EBERHART R, KENNEDY J. A new optimizer using particle swarm theory[C]// Proceedings of the Sixth International Symposium on Micro Machine and Human Science (MHS'95). IEEE, 1995: 39-43.

SHANG Yuye, born in 1999, postgraduate. Her main research interests include cloud-edge-end collaborative computing and autonomous technology for deep space probes.

YUAN Jiabin, born in 1968, Ph.D, professor, Ph.D supervisor, is a member of China Computer Federation. His main research interests include high-performance computing, deep space autonomous technology, intelligent information processing and autonomous navigations.

(责任编辑:何杨)