Proximity-Driven, Load-Balancing Task Offloading Algorithm for Enhanced Performance in Satellite-Enabled Mist Computing

Messaoud Babaghayou[1], Noureddine Chaib[1], Leandros Maglaras[2(✉)], Yagmur Yigit[2], Mohamed Amine Ferrag[3], and Carol Marsh[2]

[1] Laboratoire d'Informatique et de Mathématiques, Université Amar Telidji de Laghouat, Laghouat, Algeria
{messaoud.babaghayou,n.chaib}@lagh-univ.dz
[2] School of Computing, Engineering and the Built Environment, Edinburgh Napier University, Edinburgh, UK
{l.maglaras,yagmur.yigit,C.Marsh}@napier.ac.uk
[3] Technology Innovation Institute, Masdar City 9639, Abu Dhabi, United Arab Emirates
mohamed.ferrag@tii.ae

Abstract. In an era of rapidly evolving mobile computing, integrating satellite technologies with the Internet of Things (IoT) creates new communication and data management horizons. Our research focuses on the emerging challenge of efficiently managing heavy computing tasks in satellite-based mist computing environments. These tasks, crucial in fields ranging from satellite communication optimization to blockchain-based IoT processes, demand significant computational resources and timely execution. Addressing these challenges, we propose a novel orchestration algorithm, K-Closest Load-balanced Selection (KLS), explicitly designed for satellite-based mist computing. This innovative approach prioritizes the selection of mist satellites based on proximity and load balance, optimizing task deployment and performance. Our experimentation involved varying the percentages of mist layer devices and implementing a round-robin principle for equitable task distribution. The results showed promising outcomes in terms of energy consumption, end-to-end delay, and network usage times, highlighting the algorithm's effectiveness in specific scenarios. However, it also highlighted areas for future improvements, such as CPU utilization and bandwidth consumption, indicating the need for further refinement. Our findings contribute significant insights into optimizing task orchestration in satellite-based mist computing environments, paving the way for more efficient, reliable, and sustainable satellite communication systems.

Keywords: Satellite Edge Computing · Task Orchestrationn · K-Closest Load-balanced Selection · Energy-efficient Offloading · End-to-End Delay Reduction

L. A. Maglaras and C. Douligeris (Eds.): WiCON 2023, LNICST 527, pp. 29–44, 2024.
https://doi.org/10.1007/978-3-031-58053-6_3

1 Introduction

Mobile computing has undergone a significant transformation in recent years, driven by the convergence of two key trends: the burgeoning Internet of Things (IoT) and the advent of 5G technology. This shift is steering the field away from traditional Mobile Cloud Computing towards Mobile Edge Computing (MEC), a more decentralized approach [1,8]. At the same time, satellite communication, a field with over four decades of history, is witnessing a resurgence [13]. The foundational work in this area, which includes pioneering concepts in satellite repeaters, orbital dynamics, and path-loss calculations, remains relevant even today through the increasing integration of IoT in satellite-to-satellite communications. The demand for satellite-to-satellite IoT services is rapidly growing, fueled by global enterprises and governments seeking to monitor and manage assets scattered across the globe. This has led to a significant expansion in the satellite market, with operators now offering comprehensive IoT solutions that leverage satellite technology [5]. As a result, edge computing within the satellite domain garners interest from various stakeholders, encompassing different layers from mist devices to data centres and cloud services, as seen in Fig. 1.

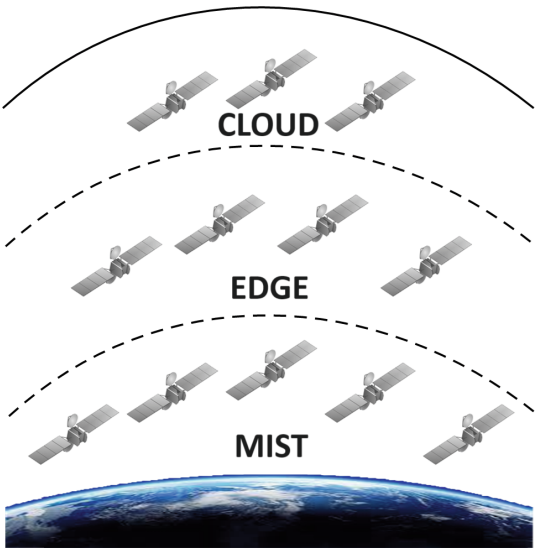


Fig. 1. The general satellite edge computing architecture.

In evolving mobile and satellite communications, the imperative for "heavy computing tasks" is critical. These tasks span a range of compute-intensive operations, from intricate calculations in satellite communication optimizations to the processing demands of blockchain-based IoT frameworks. In the field of Unmanned Aerial Vehicles (UAVs), such tasks involve video preprocessing,

pattern recognition, and feature extraction, necessitating substantial computational capabilities [9]. Conversely, Yan et al. show that heavy computing arises during the mining process in integrating blockchain within IoT, requiring significant processing power [14]. The authors' research delves into efficiently handling these computationally intensive tasks within the satellite-based mist computing environment, aiming to optimize energy consumption and ensure timely task execution. This novel approach represents a pivotal step towards accommodating heavy computing tasks and fostering a more efficient satellite-based mist computing paradigm.

Undoubtedly, the meticulous preservation of privacy and security holds paramount importance in IT systems at large [2], and this significance is magnified in the domain of satellite-based mist computing [3]. These critical aspects form the bedrock upon which the success and reliability of the entire system rest. Neglecting robust privacy and security measures would expose the system to vulnerabilities, potentially allowing breaches by adversaries and malicious actors [6]. Such breaches could lead to severe consequences, undermining the efficacy and trustworthiness of the entire infrastructure. Therefore, a vigilant and proactive approach to safeguarding privacy and security is indispensable, fortifying the foundations of satellite-based mist computing and ensuring its sustained success.

Moreover, the meticulous preservation of Quality of Service (QoS), covering essential factors like latency and various types of delays, is indispensable. This necessity extends beyond general IT systems [4] and holds particular significance within the realm of satellite systems [5]. These systems' seamless operation and performance rely on consistently delivering QoS standards, underscoring the importance of maintaining and optimizing these aspects for sustained effectiveness. However, in the rapidly evolving landscape of satellite mist computing, the burgeoning potential of this paradigm is met with the inherent challenge of identifying orchestration algorithms tailored to specific application and scenario requirements. The intricate interplay between diverse factors such as energy efficiency, latency minimization, and resource optimization necessitates the continuous exploration and refinement of orchestration strategies. As the demand for satellite-based edge computing intensifies across various domains, from IoT deployments to remote sensing applications, the research community finds itself propelled to develop novel orchestration solutions that effectively address the unique demands posed by these diverse use cases. The motivation to bridge this gap and provide orchestration algorithms attuned to specific application intricacies drives ongoing research endeavours in the satellite mist computing domain. This paper makes noteworthy contributions to the domain of satellite-based mist computing through the following key elements:

1. **Innovative Orchestration Algorithm:** Our work introduces a novel orchestration algorithm, "K-Closest Load-balanced Selection" (KLS), explicitly designed for satellite-based mist computing environments. This algorithm prioritizes the selection of mist satellites based on proximity and load balancing, contributing to optimized task deployment.

2. **Fine-Tuned Parameters:** Our study investigates the impact of varying the percentage value k in the KLS algorithm. We provide insights into how parameter adjustments influence the algorithm's performance under different scenarios by testing k values of 30%, 50%, and 70%.
3. **Investigation at the Mist Level:** Our paper makes noteworthy contributions by focusing on the mist layer within the satellite-based computing environment. This investigation delves into the intricacies of task orchestration, specifically at the mist level, shedding light on the unique challenges and opportunities that arise in this crucial layer of computing architecture.

The remainder of this paper is organized as follows:

Section 2 provides an overview of related work in satellite-based mist computing. In Sect. 3, we present the system model, detailing the key entities and components within our simulated environment. Section 4 outlines the methodology, including developing and describing our novel task orchestration algorithm. Section 5 presents the simulation setup, environment, and the corresponding results. Finally, in Sect. 6, we conclude from our findings and discuss potential avenues for future research.

2 Related Work

A recent investigation conducted by Wang.F et al. [10] delves into the capabilities of Low Earth Orbit (LEO) satellites within the context of satellite-based edge computing. LEO satellites, characterized by expansive coverage and low latency, are valuable assets for delivering computing services to user access terminals. The study addresses resource allocation challenges in Edge Computing Satellites (ECS) arising from diverse resource requirements and access planes. To manage inter-satellite links (ISLs) and ECS resource scheduling, the study proposes a three-layer network architecture incorporating a software-defined networking (SDN) model. This model utilizes advanced algorithms for efficient resource allocation and ISL construction, including the Advanced K-means Algorithm (AKG) and a breadth-first-search-based spanning tree algorithm (BFST). Simulation results validate the feasibility and effectiveness of this dynamic resource scheduling approach, providing solutions to key challenges in satellite-based edge computing systems.

Zhang et al. investigated to examine the crucial integration of satellite and terrestrial networks within the realm of 6G wireless architectures [15]. These integrated networks ensure robust and secure connectivity across extensive geographic areas. The authors introduce the concept of double-edge intelligent integrated satellite and terrestrial networks (DILIGENT), emphasizing the imperative synergy between communication, storage, and computation capabilities within satellite and cellular networks. By harnessing multi-access edge computing technology and artificial intelligence (AI), the DILIGENT framework is meticulously crafted for systematic learning and adaptive network management. The article offers a comprehensive overview of academic research, standardization endeavours, and a detailed exploration of the DILIGENT architecture,

accentuating its inherent advantages. Various strategies, including task offloading, content caching, and distribution, are scrutinized, and numerical results underscore the superior performance of this network architecture compared to existing integrated networks.

Wang.Y et al. explore the escalating significance of IoT within the information industry [11]. Edge computing is a promising paradigm that addresses challenges associated with network distance and the deployment of remote IoT devices. In scenarios where IoT devices are located in remote or challenging-to-access areas, the reliance on satellite communication becomes imperative. However, conventional satellites often lack universal computing capabilities and are specialized. The proposed solution involves the transformation of traditional satellites into space edge computing nodes, enabling dynamic software loading, resource sharing, and coordinated services with the cloud. The article delineates such satellites' hardware structure and software architecture and presents modelling and simulation results. The findings indicate that the space edge computing system exhibits superior performance, manifesting in reduced time and energy consumption compared to traditional satellite constellations. The service quality is influenced by factors such as satellite quantity, performance, and task offloading strategies.

Gao et al. [7] concentrate on the role of satellite networks as complements to terrestrial networks, addressing the computing needs of IoT users in remote regions and acknowledging the intrinsic limitations of satellites, including constraints in computing power, storage, and energy, an innovative strategy involves decomposing IoT user computation tasks into segments. This approach leverages multiple satellites for collaborative processing to enhance the efficiency of satellite networks. Integrating Network Function Virtualization (NFV) technology with satellite edge computing represents a burgeoning area of interest. The paper introduces a potential game-based solution for Virtual Network Function (VNF) placement within satellite edge computing. The primary goal is to minimize deployment costs for individual user requests while maximizing the provision of computing services across the satellite network. This optimization problem is formulated as a potential game, employing a game-theoretical approach to maximize overall network benefits. The proposed decentralized resource allocation algorithm, known as the possible game-based resource allocation (PGRA), seeks a Nash equilibrium to effectively address the VNF placement challenge. Experimental simulations validate the efficacy of the PGRA algorithm in solving the VNF placement problem within satellite edge computing.

In their investigation, Babaghayou et al. present a scheme called OVR (for Overseers) [1], underscoring its importance in tackling issues related to location privacy and road congestion within the Internet of Vehicles (IoV). OVR ingeniously utilizes silent periods to improve location privacy and effectively handle real-time congestion. Comparative assessments against established privacy schemes underscore OVR's exceptional privacy and QoS performance.

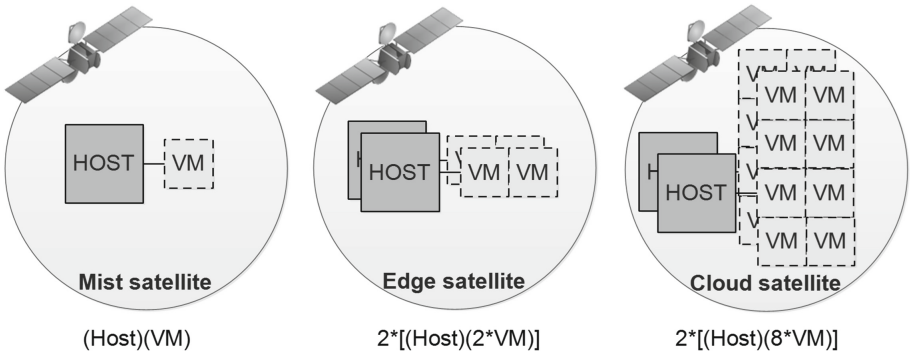


Fig. 2. The assumed types of satellites and their resources.

3 System Model

Satellite mist computing is assumed to comprise various entities, each playing a crucial role in the overall system. They are illustrated in Fig. 2 and they include:

- **Mist Satellites:** These satellites represent the mist layer and are equipped with a single host per satellite. Within each host resides a dedicated Virtual Machine (VM), where computational tasks are offloaded for processing.
- **Edge Satellites (Edge Datacenter Satellites):** Characterized as edge datacenters, these satellites are designed with heightened processing capabilities. Each edge satellite features two hosts, and each host accommodates two VMs. This architecture enhances the computational capacity for more demanding tasks.
- **Cloud Satellites:** Positioned at the highest layer, cloud satellites boast substantial computational resources. Each cloud satellite incorporates two hosts; eight VMs operate concurrently within each host. This configuration enables extensive parallel processing and scalability.
- **Tasks:** Computational workloads that necessitate offloading for efficient processing. These tasks are initiated at various points within the system and are orchestrated to suitable VMs based on the specific requirements and constraints of the satellite mist computing environment.

4 Methodology

To evaluate the performance of our proposed orchestration algorithm, which leverages the KLS approach in the mist layer, we implemented the algorithm within the satellite mist computing framework SatEdgeSim [12]. The algorithm aims to enhance task orchestration by selecting VMs strategically based on proximity, achieving load balance and efficient resource utilization. This implementation involved integrating the algorithm into the existing simulation environment, allowing for a comprehensive assessment of its impact on various performance metrics. The orchestration algorithm was tested and compared against

other existing task orchestration approaches to provide insights into its effectiveness and suitability for different scenarios within the satellite mist computing paradigm.

4.1 Task Orchestration Algorithm Development

We developed a novel algorithm; that is KLS algorithm, to optimise task orchestration in the satellite mist computing framework. The pseudo-algorithm 1 is outlined below:

Algorithm 1. K-Closest Load-balanced Selection (KLS)

function KLS($task, k$)
  Initialize empty lists *satelliteDistances* and *mistSatelliteIndices*
  for i in 0 to (*orchestrationHistory* - 1) **do**
    if satDevice.getType() == TYPES.mist_Device **then**
      //Calculate distance between mist satellite i and task's satellite device
      distance ← getDistance(satDevice, task.getSatDevice)/propagation_Speed
      Add *distance* to *satelliteDistances*
      Add i to *mistSatelliteIndices*
    end if
  end for
  Sort *mistSatelliteIndices* based on corresponding distances in *satelliteDistances*
  $vm \leftarrow -1$
  $k \leftarrow k \times \frac{\text{mistSatelliteIndices.size()}}{100}$
  for i in 0 to mistSatelliteIndices.size() **do**
    satelliteIndex ← mistSatelliteIndices.get(i)
    if offloadingIsPossible($task$, vmList.get(*satelliteIndex*)) **then**
      if $vm == -1$ or orchestrationHistory.get(*satelliteIndex*).*size*() < orchestrationHistory.get(vm).*size*() **then**
        $vm \leftarrow$ *satelliteIndex*
      end if
    end if
  end for
  return vm
end function

The **(KLS)** algorithm aims to efficiently orchestrate task offloading in a satellite-based mist computing environment. The following entities are integral to understanding the algorithm:

- **orchestrationHistory:** This is a data structure that maintains a historical record of task assignments to different VMs of the existing satellites.
- **satDevice.getType():** This method retrieves the type of the satellite device.

- **getDistance(satDevice, task.getSatDevice):** This function calculates the distance between a given mist satellite (*satDevice*) and the satellite device associated with the task to be offloaded (*task.getSatDevice*).
- **offloadingIsPossible(task, vmList.get(satelliteIndex)):** This function checks whether offloading the given task to a specific mist satellite (retrieved from *vmList* using *satelliteIndex*) is feasible.
- **satelliteDistances:** This list stores the calculated distances between the mist satellites and the satellite device associated with the task.
- **mistSatelliteIndices:** This list contains the indices of mist satellites eligible for task offloading.
- **vm:** This variable represents the selected mist satellite VM for task offloading.
- **k:** This variable determines the number of closest mist satellites to consider for task offloading. Its value is adjusted based on a percentage (k) of the total mist satellites.

4.2 Used Metrics

In this work, we have carefully chosen several metrics to assess the performance of our recently designed orchestration algorithm, KLS, in satellite mist computing. The metrics hold significant importance in comprehending the system's behaviour and evaluating the effectiveness of our solution within this distinct computing environment.

VM CPU Utilization Assessment: We examined the mean CPU consumption of VMs to ensure they ran within ideal bounds. This was essential to preventing resource overloads and optimizing the system's overall effectiveness.

Analysis of Average End-to-End Delay: We calculated the average time needed to complete tasks from beginning to end. This end-to-end delay is important to know if our orchestration method effectively meets the required deadlines and how well it handles time-sensitive, compute-intensive tasks.

Satellite Energy Consumption Measurement: Analyzing the energy consumption of the satellites while they processed tasks was a crucial component of our assessment. This allowed us to quantify our orchestration algorithm's sustainability and environmental impact.

Tasks Success Rate Evaluation: We examined the overall task success rate carried out using our orchestration technique. A high success rate indicates our approach's dependability and efficacy since a low success rate could negatively impact the system's functionality.

Network Usage Analysis: We measured the time spent on the network infrastructure to determine how much strain our system puts on the network infrastructure. Monitor the efficiency of the network, which involves tracking the volumes of data transmitted across various satellite mist computing system layers.

Bandwidth Consumption Monitoring: During task execution, we also calculated the bandwidth consumption. This metric is essential when determining the necessary network capacity and spotting possible bottlenecks in the system.

Total Executed Tasks per Layer: We counted the total number of tasks successfully executed at each satellite mist computing system layer. This metric provided a comprehensive view of how the workload was distributed and handled across different system layers.

4.3 Task Orchestration Algorithms

This section delves into the diverse task orchestration algorithms explored in our research. Effective task orchestration is pivotal for enhancing resource efficiency within satellite mist computing. We compare different algorithms, including our proposed KLS algorithm, to evaluate their impact on different metrics, precisely: CPU utilization, end-to-end delay, energy consumption, and overall system reliability. This analysis sheds light on the strengths and weaknesses of each algorithm, guiding their applicability in diverse scenarios within the satellite mist computing paradigm.

Round_Robin: The algorithm selects the satellite node with the least assigned tasks, fostering an evenly distributed computing workload. Its focus on underutilized resources aims to mitigate potential bottlenecks and ease the burden on heavily loaded nodes. This strategy contributes to optimal system performance and task completion rates. Additionally, it minimizes the risk of overwhelming any individual satellite node, ensuring efficient and effective resource utilization. In situations with diverse task requirements, the scheme provides a fair and practical approach to optimize task deployment in satellite-based mist computing environments.

Trade_Off: It adopts a dynamic strategy that intelligently chooses the most appropriate satellite node by striking a nuanced balance among factors such as latency, energy consumption, and resource availability. This algorithm assesses each potential satellite node, considering its type (cloud or edge device), the number of pending tasks, available CPU processing power, and task-specific attributes. Through computing a combined weighted score for each candidate, the scheme guarantees task allocation to the satellite with the most favourable trade-off among these factors. This approach enhances the efficiency and effectiveness of task orchestration, optimizing satellite resource utilization in satellite-based mist computing environments.

WEIGHT_GREEDY (WG): It represents a sophisticated strategy specifically crafted for satellite-based mist computing environments. The distinguishing feature of the "WEIGHT_GREEDY" scheme lies in its comprehensive assessment of four crucial performance indicators: transmission distance, CPU processing time, number of parallel tasks, and equipment energy consumption. Notably, the scheme's effectiveness is attributed to its meticulous weighting of these indicators, with the authors fine-tuning the ratios to be 6:6:5:3. This implies a deliberate emphasis on factors associated with task timeliness and minimal energy consumption, given their substantial weight in the proportion. In satellite edge computing, where timely task execution and energy efficiency take precedence, this weighting ensures an optimal balance between low latency and minimal power consumption demands.

K-Closest Load-Balanced Selection: The KLS scheme intelligently chooses satellite nodes based on a nuanced balance of latency, energy consumption, and resource availability. Designed explicitly to work on the mist layer devices, it prioritizes efficient orchestration in satellite-based mist computing. The scheme optimally balances these aspects by evaluating candidate nodes using factors like node type, distance, waiting tasks, and load on VMs. This dynamic approach, which considers the k percent closest mist-type satellites and loads on VMs, enhances orchestration efficiency, optimising satellite resource utilization. The emphasis on proximity-based selection and load balancing prevents resource bottlenecks, making it ideal for scenarios prioritizing energy efficiency and low-latency task execution.

5 Results and Discussion

5.1 Simulation Environment

The simulations were conducted using the SatEdgeSim framework [12], utilizing a mobility dataset generated by the Satellite Tool Kit (STK). The key parameters for the simulation environment are summarized in Table 1.

5.2 Simulation Results

The simulation results were evaluated based on several key metrics. Each metric provides insights into the performance of different orchestration algorithms. The following sub-subsections discuss the outcomes of the simulation runs:

Table 1. Simulation Environment Parameters

Parameter	Value
Number of Mist Satellites	1000 (maximum)
Number of Edge Datacenter Satellites	24
Number of Cloud Satellites	18
Duration	600 s
Task Generation Frequency	6 tasks per minute
K Values for Scheme	30%, 50%, 70%
Network Update Interval	1 s
Minimum Height	400.000 m
Earth Radius	6.378.137 m
Edge Devices Range	32.000.000 m
Edge Datacenters Coverage	36.000.000 m
Cloud Coverage	40.000.000 m

VM CPU Usage. The analysis of VM CPU usage, as shown in Fig. 3, revealed that the "Weighted Greedy" algorithm consumed a significant amount of CPU resources. Our scheme, specifically the variant with $k = 30\%$, also exhibited relatively high CPU usage. In contrast, the other orchestration schemes demonstrated lower CPU consumption.

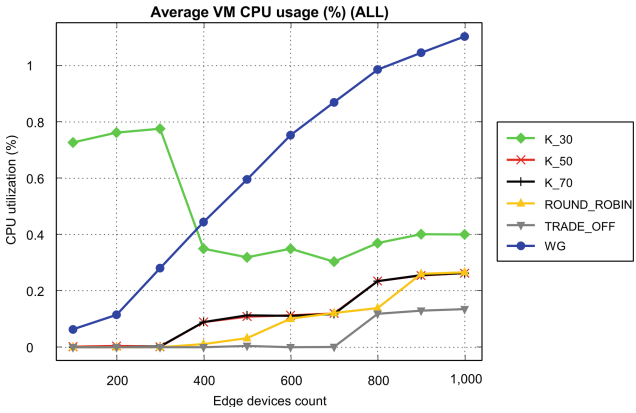


Fig. 3. The average CPU usage.

Average End-to-End Delay. In terms of average end-to-end delay shown in Fig. 4, our scheme (variant with $k = 30\%$) demonstrated superior performance. This is attributed to the scheme's emphasis on selecting close satellites to minimise communication latency. The "Weighted Greedy" algorithm followed, with other methods exhibiting varying levels of delay.

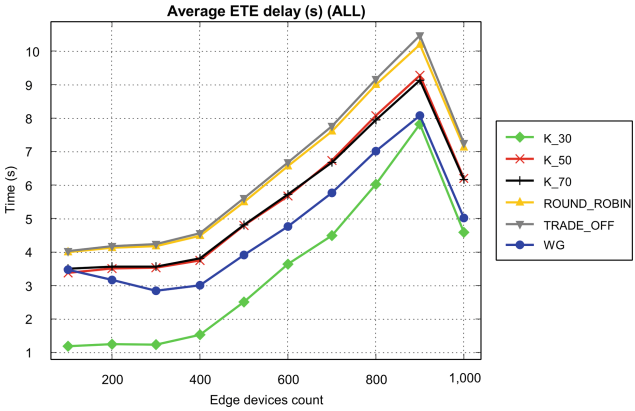


Fig. 4. The average end-to-end delay.

Satellites Energy Consumption. Our scheme (variant with $k = 30\%$) excelled in minimizing energy consumption among all orchestration algorithms, as shown in Fig. 5. The variants with $k = 50\%$ and $k = 70\%$ also performed well, along with the "Weighted Greedy" algorithm. However, the "Round Robin" and "Trade-Off" algorithms did not fare well regarding energy efficiency.

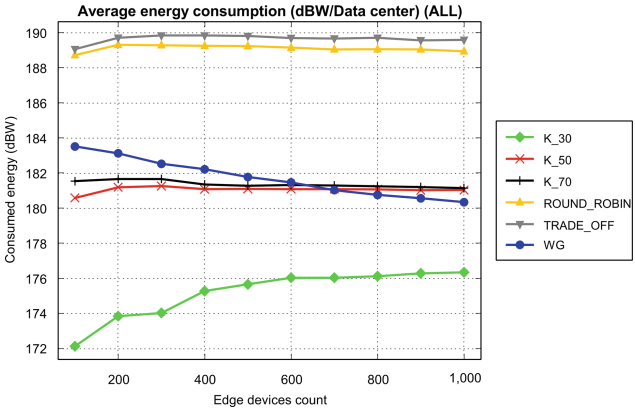


Fig. 5. The average energy consumption.

Bandwidth and Network Usage. For bandwidth usage shown in Fig. 6, the "Trade-Off" algorithm showcased the best performance, achieving a low average bandwidth per task. "Round Robin" and "Weighted Greedy" algorithms followed, while our scheme (variants with $k = 50\%$ and $k = 70\%$) demonstrated

moderate performance. Surprisingly, our method (variant with $k = 30\%$) showed the highest bandwidth consumption among the tested variants.

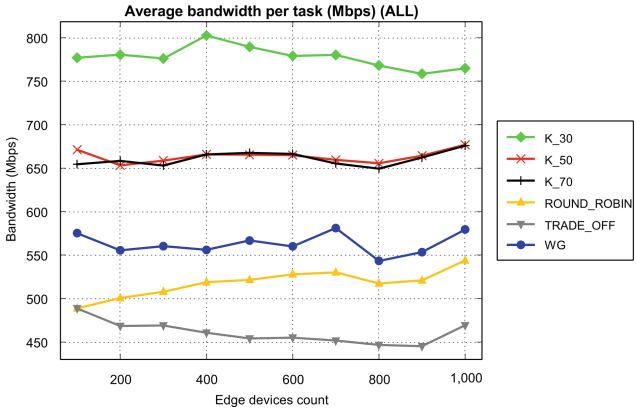


Fig. 6. The average bandwidth per task.

In the network usage metric shown in Fig. 7, our scheme (variant with $k = 30\%$) performed exceptionally well, exhibiting the lowest network usage time. The "Weighted Greedy" algorithm followed closely, with our scheme (variants with $k = 50\%$ and $k = 70\%$) showing moderate performance. "Round Robin" and "Trade-Off" algorithms demonstrated higher network usage times.

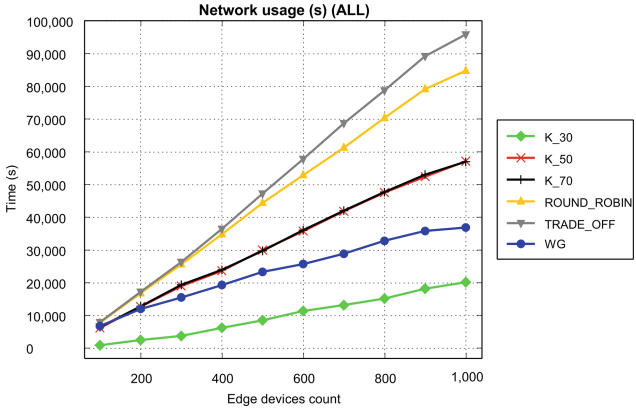
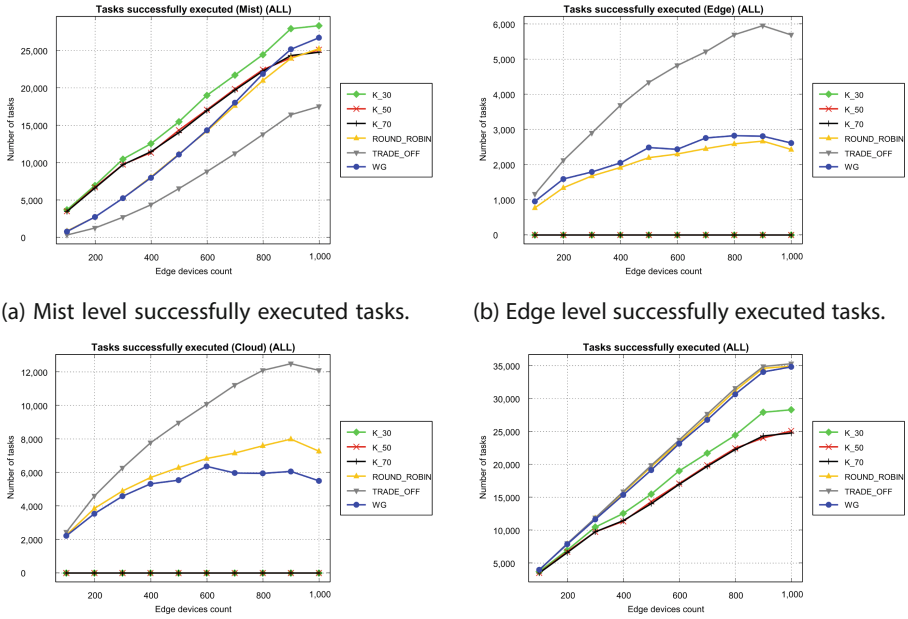


Fig. 7. The network usage.

Successfully Executed Tasks. The evaluation of successfully executed tasks revealed a unique characteristic of our KLS scheme. As it operates exclusively in the mist layer, it achieved no successful tasks in the edge data centre or cloud layers. However, our scheme (variant with $k = 30\%$) exhibited impressive results within the mist layer. Overall, our scheme's variants (30%, 50%, 70%) did not outperform other algorithms in the total number of successfully executed tasks across all layers. The obtained results are shown in Fig. 8a, Fig. 8b, Fig. 8c, and Fig. 8d.



(a) Mist level successfully executed tasks.

(b) Edge level successfully executed tasks.

(c) Cloud level successfully executed tasks.

(d) Successfully executed tasks for all layers.

Fig. 8. Overall Task Execution

6 Conclusion and Future Work

In this study, we presented and rigorously evaluated the KLS algorithm, a novel task orchestration solution designed explicitly for satellite-based mist computing environments. Focused on optimizing various performance metrics, our algorithm, particularly the variant with $k = 30\%$, demonstrated promising results in reducing average end-to-end delay, minimizing satellites' energy consumption, and optimizing network usage time. However, the study also revealed particular challenges, including higher CPU usage, increased bandwidth consumption, and

a lower total number of successfully executed tasks, highlighting areas for future improvement. Our comparative analysis with existing algorithms like "Weighted Greedy", "Round Robin", and "Trade-Off" uncovered varied performance characteristics across different metrics. This diversity emphasizes carefully considering specific application requirements when selecting an orchestration strategy.

Potential future directions can include optimizing CPU usage for improved efficiency and resource utilization, fine-tuning parameters like k for more balanced performance, extending the algorithm's capability across different layers, not just the mist layer, and integrating dynamic adaptation mechanisms to adjust to varying workloads and network conditions. Additionally, addressing security and privacy concerns in mist computing environments is essential to ensure that orchestration algorithms adhere to robust privacy and security standards. Lastly, real-world deployment scenarios and validations are crucial to assessing the algorithm's performance in satellite-based mist computing conditions. Future work can significantly contribute to advancing efficient and reliable orchestration strategies in dynamic satellite-based mist computing by addressing these identified limitations and exploring new paths.

References

1. Babaghayou, M., Chaib, N., Lagraa, N., Ferrag, M.A., Maglaras, L.: A safety-aware location privacy-preserving iov scheme with road congestion-estimation in mobile edge computing. Sensors **23**(1), 531 (2023)
2. Babaghayou, M., Labraoui, N., Ari, A.A.A., Ferrag, M.A., Maglaras, L.: The impact of the adversary's eavesdropping stations on the location privacy level in internet of vehicles. In: 2020 5th South-East Europe Design Automation, Computer Engineering, Computer Networks and Social Media Conference (SEEDA-CECNSM), pp. 1–6. IEEE (2020)
3. Baselt, G., Strohmeier, M., Pavur, J., Lenders, V., Martinovic, I.: Security and privacy issues of satellite communication in the avlatlon domain. In: 2022 14th International Conference on Cyber Conflict: Keep Moving!(CyCon), vol. 700, pp. 285–307. IEEE (2022)
4. Dagli, M., Keskin, S., Yigit, Y., Kose, A.: Resiliency analysis of onos and opendaylight sdn controllers against switch and link failures. In: 2020 Fifth International Conference on Research in Computational Intelligence and Communication Networks (ICRCICN), pp. 149–153. IEEE (2020)
5. De Sanctis, M., Cianca, E., Araniti, G., Bisio, I., Prasad, R.: Satellite communications supporting internet of remote things. IEEE Internet Things J. **3**(1), 113–123 (2015)
6. Ferrag, M.A., Maglaras, L., Janicke, H., Smith, R.: Deep learning techniques for cyber security intrusion detection: A detailed analysis. In: 6th International Symposium for ICS & SCADA Cyber Security Research 2019, vol. 6, pp. 126–136 (2019)
7. Gao, X., Liu, R., Kaushik, A.: Virtual network function placement in satellite edge computing with a potential game approach. IEEE Trans. Netw. Serv. Manage. **19**(2), 1243–1259 (2022)

8. Mao, Y., You, C., Zhang, J., Huang, K., Letaief, K.B.: A survey on mobile edge computing: the communication perspective. IEEE Commun. Surv. Tutorials **19**(4), 2322–2358 (2017)
9. Messous, M.A., Senouci, S.M., Sedjelmaci, H., Cherkaoui, S.: A game theory based efficient computation offloading in an UAV network. IEEE Trans. Veh. Technol. **68**(5), 4964–4974 (2019)
10. Wang, F., Jiang, D., Qi, S., Qiao, C., Shi, L.: A dynamic resource scheduling scheme in edge computing satellite networks. Mob. Netw. Appl. **26**, 597–608 (2021)
11. Wang, Y., Yang, J., Guo, X., Qu, Z.: Satellite edge computing for the internet of things in aerospace. Sensors **19**(20), 4375 (2019)
12. Wei, J., Cao, S., Pan, S., Han, J., Yan, L., Zhang, L.: Satedgesim: a toolkit for modeling and simulation of performance evaluation in satellite edge computing environments. In: 2020 12th International Conference on Communication Software and Networks (ICCSN), pp. 307–313. IEEE (2020)
13. Wu, W.W.: Satellite communications. Proc. IEEE **85**(6), 998–1010 (1997)
14. Yan, Y., Dai, Y., Zhou, Z., Jiang, W., Guo, S.: Edge computing-based tasks offloading and block caching for mobile blockchain. Comput. Mater. Contin **62**(2), 905–915 (2020)
15. Zhang, J., Zhang, X., Wang, P., Liu, L., Wang, Y.: Double-edge intelligent integrated satellite terrestrial networks. China Commun. **17**(9), 128–146 (2020)