Performance Bottleneck Analysis of Drone Computation Offloading to a Shared Fog Node

Qingyang Zhang
Department of Computer Science
University of Tsukuba
Tsukuba, Japan
zhang.qingyang@sd.cs.tsukuba.ac.jp

Fumio Machida
Department of Computer Science
University of Tsukuba
Tsukuba, Japan
machida@cs.tsukuba.ac.jp

Ermeson Andrade
Department of Computing
Federal Rural University of Pernambuco
Recife, Brazil
ermeson.andrade@ufrpe.br

***Abstract*— Computing in drones has recently become popular for various real-world applications. To assure the performance and reliability of drone computing, systems can also adopt computation offloading to a nearby fog or edge server through a wireless network. As the offloading performance is significantly affected by the amount of workload, the network stability, and the competing use of a shared resource, performance estimation is essential for such systems. In this paper, we analyze the performance bottleneck of a drone system consisting of multiple drones that offload the tasks to a shared fog node. We investigate how resource conflict due to computation offloading causes the performance bottleneck of the drone computation system. To model the behavior of the system and analyze the performance and availability, we use Stochastic Reward Nets (SRNs). Through the numerical experiments, we confirm that the benefit of computation offloading deteriorates as the number of competing drones increases. To overcome the performance bottleneck, we also discuss potential solutions to mitigate the issue of a shared fog node.**

Keywords—Drone, Fog computing, Offload, Performance bottleneck, Stochastic reward nets

I. INTRODUCTION

The rapid growth of the Internet of Things (IoT) applications and their interference with our daily lives have led to many IoT devices and an enormous amount of data. Traditional cloud computing resources are used in part to deal with IoT resource constraints. However, using cloud-centric resources can lead to other issues, such as latency for time-critical applications [1]. Fog computing is a distributed computing paradigm that uses any computing nodes between IoT devices and the cloud. Many latency-sensitive applications, such as self-driving cars, virtual reality, augmented reality, smart roads, and smart cities, are adopting fog or edge computing architectures to meet real-time processing requirements [2-4]. Fog and edge computing are similar as both paradigms aim to bring computation closer to the source of data, while fog node does not necessarily run at the edge of the network (i.e., any intermediate nodes between edge devices and the cloud can be considered a fog node). Moreover, fog computing can also provide a solution to certain applications that need to avoid storing data in the cloud because of privacy issues and concerns [5-6].

As fog computing gradually extends the cloud to the edge of the network, the application of drones with fog computing has received increasing attention. There are many practical applications of drones, such as emergency rescue, cargo transport, remote sensing and telemetry, and precision agriculture [7-10]. Drones can communicate with the ground station via wireless networks, recognize surroundings, and realize autonomous flight. Since drones as mobile edge devices have limited computing resources and battery lifetime, reducing the computation workload during the flight is a fundamental requirement.

Computation offloading is one of the key technologies used in mobile edge computing, allowing mobile edge devices to offload all or part of their computing tasks to edge nodes under the constraints of energy, delay, and computing power. Mobile devices can offload their computation tasks to other computing nodes through the network to reduce the workloads and save battery [12]. Drones can also benefit from computation offloading but may need appropriate resource allocation strategies which account for computation and communication at the same time [11]. As computation offloading by a drone impacts the system quality, such as performance and availability, performability models have been developed to decide on efficient offloading [13]. While the performability model is useful for analyzing the impacts of offloading, the previous study only considered a single fog node that is dedicated to the drone. However, considering practical application scenarios, the fog node can be shared by multiple drones. In such a scenario, the effectiveness of computation offloading may not be obvious.

In this paper, we model a system consisting of multiple drones competing for offloading tasks to a shared fog node and analyze the potential performance bottleneck through numerical analysis. The tasks on the drone to be offloaded are assumed to be part of an application program like image processing tasks such as object recognition. In the evaluation of system performance and availability, we consider uncertainty factors such as workload intensity and link reliability. The workload intensity is parametrized by the arrival rate of image processing requests, while the link reliability is considered as the communication disconnection rate. We use Stochastic Reward Nets (SRNs) [26] to model the interactions between multiple drones and a shared fog node under the uncertainty factors. SRN allows us to define the reward functions over the Petri nets for quantifying several performance measures of interest. In order to quantitatively analyze the bottleneck, we conduct sensitivity analysis by varying the uncertainty factors with different numbers of competing drones. Through the numerical analysis, we confirm both service availability and throughput decrease as the number of drones increases. When the amount of workload on the drones increases, both availability and throughput decrease due to the bottleneck in the fog node that does not have sufficient resources to process all the tasks offloaded from multiple drones. When the communication quality of drones becomes unstable, the service throughput declines since many jobs cannot be successfully offloaded. While low

link reliability to a drone harms the throughput, low link reliabilities to other competing drones benefit the throughput, as it reduces the competition at the fog node. We also discuss potential approaches to overcome the performance bottleneck in the fog node resource.

The rest of the paper is organized as follows. Section II describes related work. Section III and IV introduce the target system and the model we developed to evaluate the system performance and availability, respectively. Section V presents the results of bottleneck analysis through the numerical experiments on the proposed model. Section VI discusses the potential approaches to mitigate the performance bottleneck in the fog node. Finally, Section VII gives our conclusion.

II. Related work

System and network design for drone applications have been emerging challenges for research and industry. Drones need to communicate with ground stations, computing devices, and other drones via various types of wireless communication channels and protocols. For example, an open-source simulator – FlyNetSim [14] allows us to simulate and evaluate swarms of drones operating in a connected multi-layer technology ecosystem, such as the urban Internet of Things (IoT). IoD-Sim is another open-source simulator that offers a ready-to-use simulation development platform for drone application systems [15]. In the present work, we consider the performance impact of offloading among multiple drones to a shared fog node rather than providing a general-purpose simulator. Using SRNs to model the system behavior, we can analytically evaluate both the performance and availability impacts of computation offloading.

Mobile edge computing (MEC) and fog computing have been actively studied in recent years [16][17]. Mobile and edge devices can reduce computational burden and save energy by offloading the tasks to fog and edge servers or cloud nodes. Offloading techniques can be broadly classified into full or partial offloading [18]. Computing tasks can be offloaded to different layers of the underlying computing system. For example, in a three-tier structure, an edge device can decide to completely offload tasks to edge nodes or cloud nodes or to complete the task itself [19]. In our study, we consider partial offloading with a fog layer, by which only important heavy computational tasks, such as image processing, are offloaded to a fog node. We do not consider the cloud layer in this paper.

Sensitivity analysis is one of the essential methods to analyze the system performance bottleneck. There have been many studies using Stochastic Petri nets (SPNs) to model the system and to analyze the sensitivity of system parameters to the system performance [20-24]. For example, Silva *et al.* proposed an SPN to model the cooperative usage of cloud and fog [20]. The presented model allowed the configuration of 12 parameters, such as the number of available resources, workload, and average request arrival time. The SPN is also used to represent the abstract distributed systems consisting of IoT, edge, and cloud layers. A Deterministic and Stochastic Petri Net (DSPN) is developed for evaluating fog and cloud IoT environments consisting of hundreds of physical things. The proposed approach allows evaluating the trade-offs of many performance metrics (e.g., utilization, response time, throughput, and availability) and thus can help system designers to choose the most appropriate fog and cloud IoT environment [25]. In our study, as an extension of the previously developed model [13], we develop a comprehensive SRN to determine the computation offloading for drone tasks under situations where multiple drones share the fog node for offloading.

III. Drone computation system

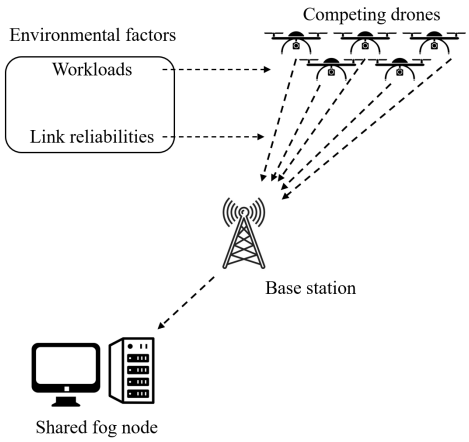
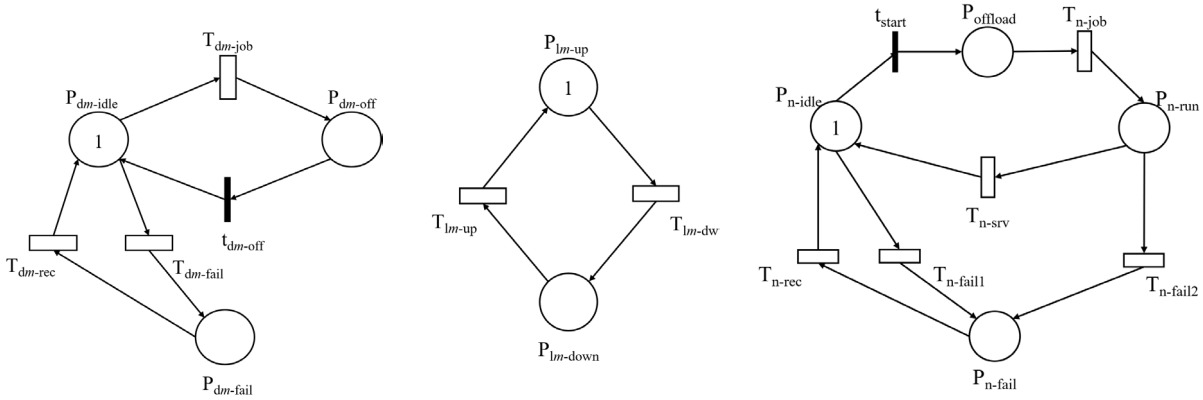


Figure 1. Drone computation system

We introduce the drone computation system considered in this paper. As shown in Figure 1, the system consists of a swarm of flying and working drones, a base station, and a shared fog node server. Drones are equipped with cameras and processors that are used to analyze the images taken by the cameras. Drones can communicate with the fog node via the base station through a mobile link. By using wireless networks, drones can offload computing tasks to the shared fog node. We assume that the shared fog node is a local server located close to the base station. Computation offloading is only possible when the drone is connected to the wireless link and the fog node is idle in which no other tasks are being processed. We assume that tasks are processed on a first-come first-served basis, and the drones need to wait for the offloading until the fog node is idle. Since a fog node can reduce the computation workloads, offloading can also help reduce process failures and unavailability of the drone under a high workload. On the other hand, it may incur an additional delay in processing due to the communication between the drone and the fog node.

There are two key environmental factors that can impact the system performance and availability. One is the computing demand for individual drones. If the workload is small, the drone can handle the tasks without using offloading. However, when the workload becomes heavy, offloading the tasks to the fog node is a desirable option as it can increase the availability of the system. The other key factor is the quality of wireless communication. The drone can send the tasks to the fog node for offloading if the wireless communication is robust. However, when the quality of the networks degrades, computation offloading may fail; hence, computation offloading is not a desirable option.

Besides the environmental factors, the performance and availability of a drone using the fog node also depend on the behavior of other competing drones. As the number of drones requesting tasks to the shared fog node increases, the fog node might become a performance bottleneck of the system. In the following sections, we investigate such a bottleneck through modeling and analysis of the system.



(a) Drone and wireless link models for drone m (b) Fog node model

Figure 2. SRN for a drone system consisting of m drones and a shared model

IV. PROPOSED MODEL

A. Offloading model

We compose SRNs for a drone image processing system using computation offloading to a fog node. The model is based on the fog offloading model presented in the previous study [13], but it is extended to consider the resource conflicts on the fog nodes by multiple drones. Figure 2 shows the SRN that consists of inter-dependent subnets corresponding to drones, a fog node, and the wireless link between them. Note that the subnets for drones and the wireless links are created for $m \geq 1$ competing drones, where m represents the serial number of the drone. The parameters in the model as well as their values used in numerical experiments are summarized in TABLE III.

TABLE I. GUARD FUNCTIONS FOR THE FOG OFFLOADING MODEL

Name	Transition	Function
gstart	t_{start}	if(#$P_{d1\text{-off}}$== 1 || #$P_{d2\text{-off}}$ == 1||···||#$P_{dm\text{-off}}$ ==1) then 1 else 0
gstarted	$t_{dm\text{-off}}$	if(#$P_{\text{n-run}}$== 1) then 1 else 0
goffloadm	$T_{dm\text{-job}}$	if(#$P_{\text{n-idle}}$== 1 && #$P_{lm\text{-up}}$== 1) then 1 else 0

In the drone models, when a token is in the place $P_{dm\text{-idle}}$, it represents the drone is not processing jobs. After firing the transition $T_{dm\text{-job}}$, the token deposited in $P_{dm\text{-off}}$, the transition t_{start} is enabled by the guard function *gstart* as shown in TABLE I. We assume that the arrivals of job processing requests to the drones follow the Poisson processes with rate γ_m. When t_{start} fires, a token is removed from the $P_{\text{n-idle}}$, and a new token is deposited in P_{offload}. Then the token is removed when $T_{\text{n-job}}$ fires and a new token is deposited to $P_{\text{n-run}}$, which means the task is offloading and running. Simultaneously, the immediate transition $t_{dm\text{-off}}$ fires by the guard function *gstarted*. We assume that the average time to start a job processing in the fog node (communication delay) is $1/\omega$, where ω denotes the communication rate. Since offloading only works when the node is idle and the wireless communication link is available, the condition is specified by the guard function *goffloadm* assigned to $T_{dm\text{-job}}$. The firing of $T_{\text{n-srv}}$ represents the completion of a job processing. We assign service rate ν_{n} for $T_{\text{n-srv}}$. Meanwhile, the process may fail when the fog node is idle or running, which is represented by the transitions $T_{\text{n-fail1}}$ or $T_{\text{n-fail2}}$, respectively. We assume that process failures occur due to software bugs or transient hardware faults, and assign the process failure rate λ_{n1} and λ_{n2} for $T_{\text{n-fail1}}$ and $T_{\text{n-fail2}}$. Since process failures can occur more frequently in the processing state than in the idle state, we assume $\lambda_{\text{n1}} < \lambda_{\text{n2}}$. When a token deposited in $P_{\text{n-fail}}$ is removed after $T_{\text{n-rec}}$ fires, a new token is deposited in $P_{\text{n-idle}}$, which represents a node recovery. We assign the process recovery rate μ_{n} for $T_{\text{n-rec}}$. On the other hand, if $T_{dm\text{-fail}}$ fires first, which represents a drone process failure event in the idle state, a new token is deposited in $P_{dm\text{-fail}}$. We assign the process failure rate λ_{dm} for $T_{dm\text{-fail}}$. When a token is deposited in $P_{dm\text{-fail}}$, the transition $T_{dm\text{-rec}}$ is enabled, which represents the recovery of the process. We assign the process recovery rate $\mu_{d\text{m}}$ for $T_{dm\text{-rec}}$.

The wireless link model captures the states of the communication link between drones and the fog node. Even if the drones and the fog node are functioning properly, a failure of the network connection can cause the system to become unavailable. When a token is removed from $P_{lm\text{-up}}$ by firing $T_{lm\text{-dw}}$, it indicates that the communication links with temporarily disconnected. While a token is deposited in $P_{lm\text{-down}}$, the transition $T_{lm\text{-up}}$ is enabled. After $T_{lm\text{-up}}$ is fired, a token is deposited to $P_{lm\text{-up}}$, representing the reconnection. The rates of link disconnection ($T_{lm\text{-dw}}$) and reconnection ($T_{lm\text{-up}}$) are assumed to be $\lambda_{l\text{m}}$ and $\mu_{l\text{m}}$, respectively.

TABLE II. REWARD FUNCTIONS FOR THE FOG OFFLOADING MODEL

Name	Measure	Function
svavail	Service availability	if((#$P_{d1\text{-idle}}$ == 1 || #$P_{d1\text{-off}}$ == 1) && #$P_{l1\text{-up}}$ == 1 && (#$P_{\text{n-idle}}$ == 1 || #$P_{\text{n-run}}$== 1)) then 1 else 0
svthru	Service throughput	prob(#$P_{\text{n-run}}$ == 1)* ν_{n} * (prob(#$P_{d1\text{-idle}}$) * γ_1) / ((prob(#$P_{d1\text{-idle}}$) * γ_1)+ (prob(#$P_{d2\text{-idle}}$) * γ_2)+···+(prob(#$P_{dm\text{-idle}}$) * γ_{m}))

We define two reward functions for computing system availability and performance measures. TABLE II shows the definition of the reward functions. We evaluate the availability and the throughput of the primary drone ($m = 1$). For computing service availability, *svavail* assigns one reward for the marking (#$P_{d1\text{-idle}}$ == 1 or #$P_{d1\text{-off}}$ == 1) and #$P_{\text{l-up1}}$ == 1 and (#$P_{\text{n-idle}}$ == 1 or #$P_{\text{n-run}}$== 1), where #P_x represents the number of tokens in P_x. The specified condition represents that the service is available only when the drone, the fog node and the wireless link are available simultaneously. *svthru* is computed by multiplying the throughput at $T_{\text{n-srv}}$ and the ratio of the tasks from the primary drone over the total offloaded tasks. The throughput at $T_{\text{n-srv}}$ can be computed by the probability of marking #$P_{\text{n-run}}$==1 and ν_n (service rate of the fog node).

V. NUMERICAL EXPERIMENTS

To investigate the performance bottleneck of the drone offloading system using a shared fog node, we conduct a sensitivity analysis on the proposed model.

A. Experimental configuration

For the sake of comparative study with the previous experiments, we follow the parameter values used in the previous study [13]. The job arrival rate γ_m and the link failure rate λ_{lm} are the parameters of uncertainty factors and they are used as the variables in the sensitivity analysis. Note that m in the subscripts of symbols represents the serial number of the drone. For instance, λ_{l1} is the link disconnection rate for the primary drone.

TABLE III. PARAMETERS VALUES FOR NUMERICAL EXPERIMENT

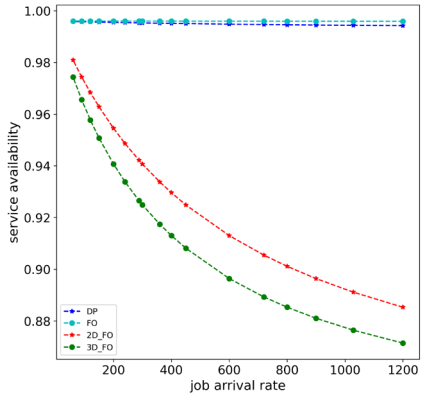
Notation	Transition	Description	Value[1/hour]
γ_m	$T_{dm\text{-job}}$	Job arrival rate	720
ν_n	$T_{n\text{-srv}}$	Service rate of a process on the fog node	1440
λ_{dm}	$T_{dm\text{-fail}}$	Failure rate of a process on the drone m in the idle state	0.002976190
μ_{dm}	$T_{dm\text{-rec}}$	Fecovery rate of a process on the drone m	3
λ_n	$T_{n\text{-fail}}$	Failure rate of a process on the fog node in the idle state	0.000462963
λ_{n2}	$T_{n\text{-dail2}}$	Failure rate of a process on the fog node in the processing state	0.00138889
μ_{nm}	$T_{dm\text{-rec}}$	Recovery rate of a process on the fog node	2
ω	$T_{n\text{-job}}$	Communication rate between the drone and the fog node	7200
λ_{lm}	$T_{lm\text{-dw}}$	Communication link failure rate	0.5
μ_{lm}	$T_{lm\text{-up}}$	Communication link recovery rate	360

B. Sensitivity analysis on workload variation

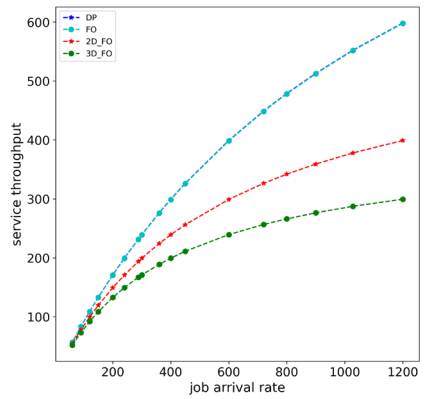
First, we evaluate the impacts of the workload on the service availability and throughput of the fog offloading system with 1, 2, and 3 drones (FO, 2D_FO, and 3D_FO). To compare with the system without using computation offloading, we also evaluate the service availability and throughput by the drone processing model (DP) [13]. We fix the communication link failure rate $\lambda_{l1} = \lambda_{l2} = \lambda_{l3}$ to 1 (1/hour) and vary the job arrival rate $\gamma_1 = \gamma_2 = \gamma_3$ in a range of [60, 1200] (1/hour). The expected service availabilities and service throughputs are plotted in Figures 3 (a) and (b), respectively.

As can be seen, the service availabilities significantly decrease in 2D_FO and 3D_FO compared to the availability of DP and FO. When the workload intensity becomes high, the service availability of 2D_FO and 3D_FO decreases to around 0.88, which may be unacceptable considering practical usages. The availability decrease is caused by the increasing waiting time for offloading at the drone. When the fog node processes the tasks from the other drones, the primary drone must need to wait in the idle state in which the process may encounter a failure.

The service throughput can increase by the increased workloads in all the configurations. Nevertheless, the improved rates of 2D_FO and 3D_FO are significantly smaller than DP and FO. When the amount of workloads is not large, the difference in the service throughputs is marginal. However, as the amount of workloads increases, the difference between DP and FO enlarges. The results are caused by the resource contention in the shared fog node that receives offloading tasks from all the drones. We observe that the throughput reduction is not simply proportional to the

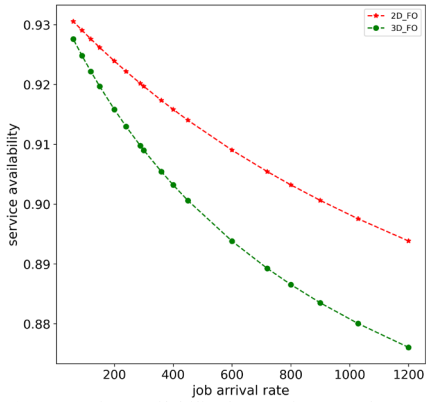


(a) Service availability under varying workload

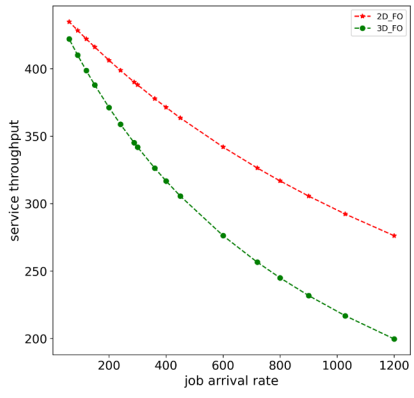


(b) Service throughput under varying workload

Figure 3. Sensitivity analysis results under varying workload

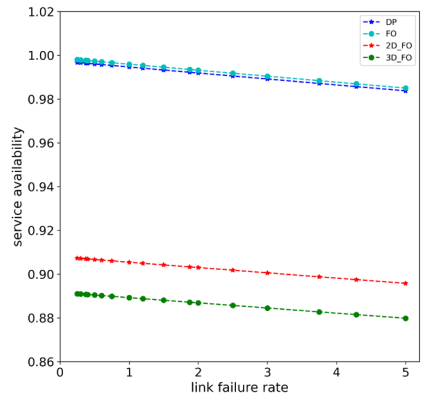


(a) Service availability by varying γ_2 and γ_3

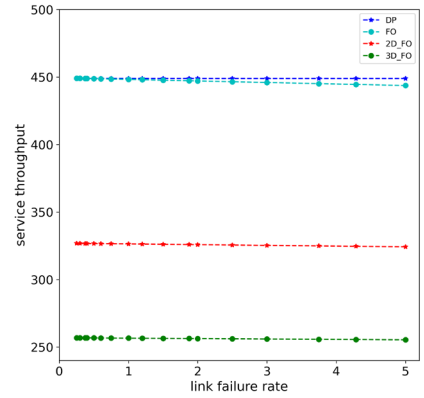


(b) Service throughput by varying γ_2 and γ_3

Figure 4. Sensitivity analysis results by varying γ_2 and γ_3

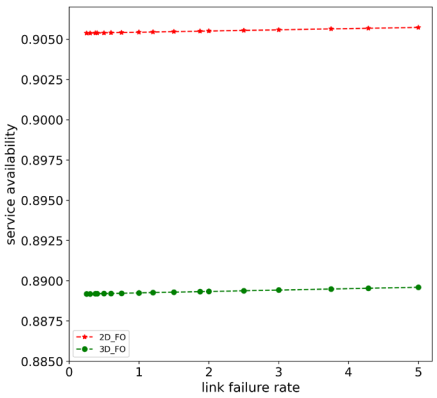


(a) Service availability under varying link reliability

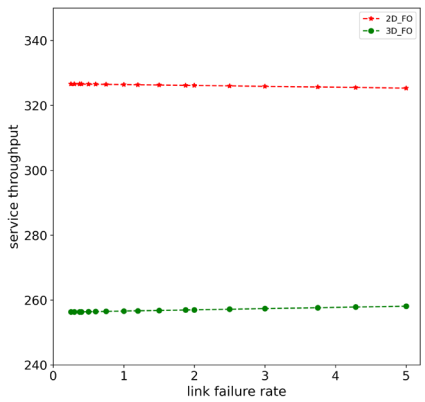


(b) Service throughput under varying link reliability

Figure 5. Sensitivity analysis results under varying link reliability



(a) Service availability by varying λ_{l2} and λ_{l3}



(b) Service throughput by varying λ_{l2} and λ_{l3}

Figure 6. Sensitivity analysis results by varying λ_{l2} and λ_{l3}

number of competing drones. The results indicate that the utilization of the fog node resource is actually improved by increasing the number of drones, but the service throughput does not increase simply because the fog node's computing power is not enough for multiple offloaded tasks.

Next, we fix the job arrival rate of the primary drone ($m = 1$) to be 720 (1/hour) and vary the job arrival rates γ_2 and γ_3 of the other drones between [60,1200] (1/hour) to analyze the impacts of the workload intensities of other drones. The expected service availability and service throughput are plotted in Figures 4 (a) and (b), respectively. As the job arrival rate increases, both service availability and throughput decrease. Compared to the previous case, the job arrival rate of the primary drone is unchanged, and hence the throughput can significantly decrease when the amount of offloading tasks from other drones increases. As the fog node processes tasks on a first-come first-served basis, the imbalance of job arrival rates among the drones matter.

C. Sensitivity analysis on link reliability

In this subsection, we evaluate the impacts of the communication link reliability on service availability and performance. In this experiment, we fix the job arrival rate $\gamma_1 = \gamma_2 = \gamma_3$ to 720 (1/hour) and vary the communication link failure rate $\lambda_{l1} = \lambda_{l2} = \lambda_{l3}$ in a range of [0.25, 5] (1/hour). The expected service availability and service throughputs are plotted in Figure 5. (a) and (b), respectively.

For all the configurations, we observe the small monotonic decreasing trends in the service availability when increasing the communication link failure rates. The service availability in 2D_FO and 3D_FO are significantly smaller than in DP and FO, which are mainly due to the high workloads. On the other hand, the service throughput slightly decreases by increasing the communication link failure rates. The service throughput in DP is unchanged because the link reliability does not impact the processing on the drone in DP mode.

Next, we fix the link failure rate of the primary drone to 1 (1/hour) and vary the link failure rates λ_{l2} and λ_{l3} of the other drones between [0.25,5] (1/hour). The expected service availability and service throughput are shown in Figures 6 (a) and (b), respectively. We can observe that service availability and throughput improve as the other drone's link failure rates increase. The results show that lower link reliabilities of other competing drones limit the access to the shared fog node, resulting in a higher chance to use the resource to process the offloaded tasks from the primary drone.

VI. Discussion

Our numerical results show how the bottleneck in the shared fog node resource impact service availability and throughput of drone computation under different workload and communication quality environments. When the number of competing drones increases, the service availability and throughput decrease due to resource contention. The significance of resource contention depends on the amount of workload and communication link reliabilities of other drones. There are a number of ways to overcome the bottleneck and improve the performance of computation offloading. One approach is to use more powerful fog nodes so that they can handle more tasks. This approach simply scales up the computing resource at the fog node. While the fog node can accept more tasks offloaded, the network bandwidth may become another bottleneck if the tasks entail a high volume of

data. Moreover, the fog node may become a single point of failure that reduces the total system reliability. Alternatively, we may increase the number of fog nodes, which can also increase the total processing powers in the fog infrastructure. This approach requires a dynamic resource allocation mechanism like cloud computing to map the drone requesting the computation offloading to the fog that is available for task processing. The resource allocation is not trivial as it needs to consider the node's distance, energy consumption, communication link quality, and other interrelated factors. Another promising solution that can be considered is hierarchical offloading [27][28], which uses cloud computing to further offload the tasks when fog nodes are not sufficient to process the tasks. This approach also requires a decision mechanism to send the tasks to the cloud. It must consider the importance of tasks because different tasks have different priorities, and the computing capabilities of cloud nodes and edge nodes are different. Special attention may be required for the tasks offloaded to the cloud as they incur an additional delay. The comparisons of the mitigations are considered in future work.

VII. Conclusion

In this paper, we analyze the performance bottleneck of a drone offloading system where competing drones share the fog computing resource for computation offloading. We use SRNs to model the interactions between drones and a shared fog node. By changing the number of competing drones in the numerical experiments, we observe that both service availability and throughput of the primary drone decrease because of resource contention at the fog node. The sensitivity analysis results show that the increased job arrival rate of other drones negatively impacts the primary drone's performance. In contrast, the decreased reliability of communication links for other drones has a positive impact. We also discuss potential mitigations to overcome the performance bottleneck in the shared fog node for drone computation offloading systems.

Acknowledgment

This work was supported in part by the grant of University of Tsukuba Basic Research Support Program Type S.

References

[1] S. Hamdan and M. Ayyash, S. Almajali, "Edge-computing architectures for internet of things applications: A survey," Sensors, vol. 20, no. 22, pp. 6441, 2020.

[2] M. Ran and X. Bai, "Vehicle cooperative network model based on hypergraph in vehicular fog computing," Sensors, vol. 20, no. 8, pp. 2269, 2020.

[3] H. Tanaka and M. Yoshida and K. Mori and N. Takahashi, "Multi-access edge computing: A survey," Journal of Information Processing, vol. 26, pp. 87-97, 2018.

[4] L. U. Khan and I. Yaqoob and N. H. Tran and S. M. A. Kazmi and T. N. Dang and C. S. Hong, "Edge-computing-enabled smart cities: A comprehensive survey," IEEE Internet of Things Journal, vol. 7, no. 10, pp. 10200-10232, 2020.

[5] L. Babun and Z. B. Celik and P. McDaniel and A. S. Uluagac, "Real-time analysis of privacy-(un) aware IoT applications," arXiv preprint arXiv:1911.10461, 2019.

[6] S. Tabassam, "Security and privacy issues in cloud computing environment," Journal of Information Technology & Software Engineering, vol. 7, no. 5, 2017.

[7] V. Baiocchi and D. Dominici and M. Mormile, "UAV application in post-seismic environment," International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences, vol. 1, pp. W2, 2013.

[8] W. Zhang and J. Wu,"To explore the UAV application in disaster prevention and reduction," in Applied Mechanics and Materials, 2014, pp. 609-612.

[9] J. Ghommam and M. Saad and S. Wright and Q. M. Zhu, "Relay manoeuvre based fixed-time synchronized tracking control for UAV transport system," Aerospace Science and Technology, vol. 103, pp. 105887, 2020.

[10] N. Delavarpour and C. Koparan and J. Nowatzki and S. Bajwa and X. Sun, "A technical study on UAV characteristics for precision agriculture applications and associated practical challenges," Remote Sensing, vol. 13, no. 6, pp. 1204, 2021.

[11] T. Tan and M. Zhao and Z. Zeng, "Joint Offloading and Resource Allocation Based on UAV-Assisted Mobile Edge Computing," ACM Transactions on Sensor Networks (TOSN), vol. 18, no. 3, pp. 1-21, 2022.

[12] H. Qian and D. Andresen, "Extending mobile device's battery life by offloading computation to cloud," in ACM International Conference on Mobile Software Engineering and Systems, 2015, pp. 150-151.

[13] F. Machida and E. Andrade,"PA-Offload: performability-aware adaptive fog offloading for drone image processing," in the 5th International Conference on Fog and Edge Computing (ICFEC), 2021, pp. 66-73.

[14] S. Baidya and Z. Shaikh and M. Levorato,"FlyNetSim: An open source synchronized UAV network simulator based on ns-3 and ardupilot," in the 21st ACM International Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems, 2018, pp. 37-45.

[15] G. Grieco and G. Iacovelli and P. Boccadoro and L. A. Grieco, "Internet of Drones Simulator: Design, Implementation, and Performance Evaluation," arXiv preprint arXiv:2203.13710, 2022.

[16] P. Mach and Z. Becvar, "Mobile edge computing: A survey on architecture and computation offloading," IEEE communications surveys & tutorials, vol. 19, no. 3, pp. 1628-1656, 2017.

[17] Y. Mao and C. You and J. Zhang and K. Huang and K. B. Letaief, "A survey on mobile edge computing: The communication perspective," IEEE communications surveys & tutorials, vol. 19, no. 4, pp. 2322-2358, 2017.

[18] P. Mach and Z. Becvar, "Mobile edge computing: A survey on architecture and computation offloading," IEEE communications surveys & tutorials, vol. 19, no. 3, pp. 1628-1656, 2017.

[19] B. Guo and C. Chen, et al., "Mobile crowd sensing and computing: when participatory sensing meets participatory social media," IEEE Communications Magazine, vol. 54, no. 2, pp. 131-137, 2016.

[20] F. A. Silva and I. Fe, et al., "Stochastic models for performance and cost analysis of a hybrid cloud and fog architecture," The Journal of Supercomputing, vol. 77, no. 2, pp. 1537-1561, 2021.

[21] D. Wang, W. Xie, and K. S. Trivedi, "Performability analysis of clustered systems with rejuvenation under varying workload," Performance Evaluation, vol. 64, no. 3, pp. 247-265, 2007.

[22] G. Ciardo, J. K. Muppala, and K. S. Trivedi, "SPNP: Stochastic Petri Net Package.," in PNPM, 1989, pp. 142-151.

[23] J. Muppala, G. Ciardo, and K. S. Trivedi., "Stochastic reward nets for reliability prediction," Communications in reliability, maintainability and serviceability, vol. 1, no. 2, pp. 9-20, 1994.

[24] L. Santos and B. Cunha and I. Fe and M. Vieira and F. A. Silva, "Data processing on edge and cloud: a performability evaluation and sensitivity analysis," Journal of Network and Systems Management, vol. 29, no. 3, pp. 1-24, 2021.

[25] E. Andrade and B. Nogueira and I. Farias Junior and D. Araujo, "Performance and availability trade-offs in fog-cloud iot environments," Journal of Network and Systems Management, vol. 29, no. 1, pp. 1-27, 2021.

[26] G. Ciardo and A. Blakemore and P. F. Chimento and J. K. Muppala and K. S. Trivedi,"Automated generation and analysis of Markov reward models using stochastic reward nets," in Linear algebra, Markov chains, and queueing models, 1993, pp. 145-191.

[27] Z. Zhao and R. Zhao and J. Xia and X. Lei and D. Li and C. Yuen and L. Fan, "A novel framework of three-hierarchical offloading optimization for MEC in industrial IoT networks," IEEE Transactions on Industrial Informatics, vol. 16, no. 8, pp. 5424-5434, 2019.

[28] W. Seifeddine and C. Adjih and N. Achir,"Dynamic Hierarchical Neural Network Offloading in IoT Edge Networks," in 2021 10th IFIP International Conference on Performance Evaluation and Modeling in Wireless and Wired Networks (PEMWN), 2021, pp. 1-6.