# Remedy or Resource Drain: Modeling and Analysis of Massive Task Offloading Processes in Fog

Jie Wang, *Member, IEEE*, Wenye Wang, *Fellow, IEEE*, and Cliff Wang

***Abstract*—Task offloading, which refers to processing (computation-intensive) data at facilitating servers, is an exemplary service that greatly benefits from the *fog computing* paradigm, which brings computation resources to the *edge* network for reduced application latency. However, the resource-consuming nature of task execution, as well as the sheer scale of IoT systems, raises an open and challenging question: *whether fog is a remedy or a resource drain*, considering frequent and massive offloading operations? This question is nontrivial, because participants of offloading processes, i.e., *fog nodes*, may have diversified technical specifications, while task generators, i.e., *task nodes*, may employ a variety of criteria to select offloading targets, resulting in an unmanageable space for performance evaluation. To overcome these challenges of heterogeneity, we propose a *gravity model* that characterizes offloading criteria with various gravity functions, in which individual/system resource consumption can be examined by the *device/network effort* metrics, respectively. Simulation results show that the proposed gravity model can flexibly describe different offloading schemes in terms of application and node-level behavior. We find that the expected lifetime and device effort of individual tasks decrease as $O(1/N)$ over the network size $N$, while the network effort decreases much slower, even remain $O(1)$ when load balancing measures are employed, indicating a possible resource drain in the edge network.**

***Index Terms*—Fog computing, performance evaluation, task offloading.**

## I. Introduction

With the proliferation of smart devices [1], [2], including smart devices, wearables, and sensors, Internet of Things (IoT) is driving a digital transformation in all aspects of our modern life [3]. However, IoT brings about new challenges to its provision network, such as stringent requirements on latency, resource-limited end devices, and the prohibitive scale. Consequently, *fog computing* is introduced [2] and envisioned as a promising paradigm in IoT provisioning [3].

### A. Fog Emerges on the Edge: Remedy or Resource Drain?

In fog-based IoT systems, data are processed by fog servers deployed on edge devices, e.g., edge routers, Cloudlet and IOx [4], or even mobile devices [5], all of which are referred to as *fog nodes*. The architecture of IoT then evolved into a "cloud-fog-things" layered scheme [6], [7], [8], in which powerful but remote cloud computing resources are reserved for bulky and computationally heavy data, while fog computing [2] is expected to provide fast and frequent offloading services, with two design objectives [9] tailored for IoT.

*1) Short Delay:* Per requests of real-time applications, such as smart infrastructure, autonomous driving, and virtual assistance [10], [11], fog computing is expected to reduce service latency to the ms level ($<$ 10 ms), comparing to 30–100 ms in the cloud computing paradigm. This demand requires service providers to maintain sufficient fog nodes in the vicinity of any *task node*, which can be any device seeking to offload computation-intensive data, or *tasks*.

*2) Scalability:* Due to the massive scale of geo-dispersed IoT systems [3], a fog network is expected to process data offloaded from a large number of task nodes, e.g., sensors, cameras, and smart vehicles. Considering the frequent changes of locations, connectivity, and offloading criteria, this objective requires the fog paradigm to scale easily, i.e., operating with simple organization and minimum overhead. In fact, openness and scalability have been identified as the major design concerns in the IEEE OpenFog reference architecture [9].

With these two objectives fulfilled, the fog computing paradigm seems a good remedy for large-scale delay-sensitive IoT applications, as it rallies up resource at the network edge to execute offloaded tasks, taking advantage of the huge numbers of candidate fog nodes in a distributed manner. However, as mobile traffic continues to grow [1], can the resource-limited edge network keep such a fog eco-system afloat? In other words, *is fog task offloading a remedy to delay-sensitive applications, or rather, a resource drain to the network edge*?

Answer to this question is essential to fog computing, a large-scale paradigm that is yet to be deployed. To satisfy the stringent delay bound, tasks may be offloaded to nearby fog nodes, which consumes multiple types of resources, including storage space, processing power, energy, spectrum slices, transmission opportunities, and so on. From the perspective of individual devices, fog server/nodes, unlike cloud servers, often times have limited storage space and processing power, and are hence sensitive to the consumption of such device-owned resources. From the perspective of the network, on the other hand, scarcity of communication resources, including spectrum, time slots, code, space, etc. has long been recognized as a major issue in the edge network [12].

Though resource drain may be caused by various reasons, such as spectrum misuse, network overlay, and lack

Manuscript received 22 July 2022; revised 13 January 2023; accepted 3 February 2023. Date of publication 16 February 2023; date of current version 23 June 2023. This work was supported in part by NSFC under Grant 62102288; in part by NSF under Grant 1824518; in part by NSF under Grant 2030134; and in part by the Fundamental Research Fund for Central Universities. *(Corresponding author: Jie Wang.)*

Jie Wang is with the School of Software Engineering, Tongji University, Shanghai 200092, China (e-mail: wangjie_tongji@tongji.edu.cn).

Wenye Wang and Cliff Wang are with the Department of Electrical and Computer Engineering and the Department of Computer Science, North Carolina State University, Raleigh, NC 27606 USA (e-mail: wwang@ncsu.edu; xwang4@ncsu.edu).

This article has supplementary downloadable material available at https://doi.org/10.1109/JIOT.2023.3245100, provided by the authors.

Digital Object Identifier 10.1109/JIOT.2023.3245100

2327-4662 © 2023 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission.
See https://www.ieee.org/publications/rights/index.html for more information.

Authorized licensed use limited to: N.C. State University Libraries - Acquisitions & Discovery S. Downloaded on April 13,2026 at 07:22:58 UTC from IEEE Xplore. Restrictions apply.

TABLE I
SUMMARY OF REPRESENTATIVE TASK OFFLOADING LITERATURE IN FOG COMPUTING SYSTEMS

| Ref. | Type | Approach | Scale | Mobility | Criteria | Communication Model | Task |
|---|---|---|---|---|---|---|---|
| [13] | centralized | Optimization | $< 100$ | static | delay/energy | BW=20 MHz | Normal/Uniform/Pareto on $[2, 10]$Mb |
| [14] | centralized | RL | $< 10$ | static | delay | BW=2 MHz, Power=20 dBm | fixed (500 Mb) |
| [15] | distributed | Game Th. | $< 100$ | static | delay | variable BW, Power=0.4W | Uniform on $[0.1, 3.4]$Mb |
| [16] | distributed | Deep RL | $< 100$ | mobile | incentive | BW=10 MHz, fixed range | Uniform on $[0.02, 0.2]$Mb |
| [17] | centralized | Matching Th. | $< 10$ | static | hybrid | BW=10 MHz, variable power | Uniform on $[300, 600]$Kb |
| [18] | centralized | RL | $< 10$ | static | delay/energy | variable BW | fixed (5 Kb) |
| This model | distributed | Stochastic Th. Analysis | $\leq 10^4$, Any(Th.) | mobile | delay/energy and hybrid | multi-RAT, variable rate, range, and power | Uniform on $\{1, 10, 10^2, 10^3\}$Mb and variable processing intensity |

of interoperability between systems, the explosive growth of connections between IoT devices and popularity of bandwidth-hogging applications certainly add to this concern. While device-owned resources, such as storage and processing power, can be boosted by deploying more fog nodes, communication/network resources are *shared* by the edge system, and hence may subject to further degeneration as the system scales. Moreover, the open nature of fog diversifies communication options, further complicating the problem. Therefore, it is crucial to evaluate the performance of heterogeneous task offloading with respect to delay and resource consumption, before it is marketed and deployed.

### B. Related Work on Task Offloading Evaluation

Task offloading can be viewed as a special type of data offloading that has been extensively studied in various contexts. In the IoT provisioning scenario, we refer to it as the offloading process of computation-intensive data blocks [6] that need to be processed within a delay bound. Existing literature mainly focus on the design of *offloading schemes*, that is, a set of rules and operations determining where a task should be executed, which can be broadly categorized into centralized and distributed types.

The former family relies on a single centralized entity to make all offloading decisions, e.g., [13], which to some extent is against the open nature of fog computing. The major challenge in this setting is the offloading/scheduling decision with partial system information, to address which reinforcement learning(RL)-based schemes [14], [16], [18], [19], [20], [21] are recently developed. These schemes are designed for homogeneous systems of small scales, in the sense that task nodes adopt the same radio access technology (RAT), and cooperate to minimize the average task completion time, or the total energy consumption of the system. Nonetheless, real-world IoT systems are much more large-scale and diverse, in aspects of RATs, task types, and offloading criteria.

The distributed approaches further branches into cooperative and noncooperative types. In cooperative offloading schemes, e.g., [22], [23], and [24], fog nodes exchange information to optimize resource allocation, or collectively schedule the offloaded tasks, usually to minimize the offloading latency [25] and/or energy consumption [26]. In other words, fog nodes cooperate to determine the offloading target for each task. Though cooperation among fog nodes are highly beneficial to QoS provisioning and load balancing, it is less likely in a large-scale fog continuum, due to the prohibitive communication overhead, lack of interoperability among different providers, and security/privacy concerns of end users.

In this regard, Tang and He [27] recognized the competition among fog nodes, modeled the offloading process as a noncooperative game and evaluated their results with small-scale experiments. With a similar game theoretic approach, Jošilo and Dán model task allocation as a noncooperative game, and proposed a decentralized random offloading scheme, which requires the first and second moment of task attributes, but achieves nearly the same task completion time as myopic best response (MBR) solutions that requires real-time node-specific information [15]. Considering wireless links may not be stable during offloading, Wu [28] studied fault recovery in the offloading process, and aimed to maximize revenue of the network.

Table I summarizes the characteristics and parameters of representative schemes. A clear observation is that, despite previous efforts on scheme design for systems of small (less than 100 nodes) scales, there is little effort on the performance evaluation of *frequent* and *massive* task offloading operations, which are inevitable in fog computing based IoT systems.

Seemingly simple, large-scale performance evaluation is, nonetheless, challenging due to the sheer *scale* and *diversity* of the fog-based IoT system. On the application level, IoT applications are so broad and complex, that data are generated in various volume, form, and frequency [29], e.g., periodic, short sensory data from smart meters, as well as spontaneous, long video clips from ground traffic monitoring. On the network level, due to the open nature of fog computing, a task can be directed to a certain fog node by a variety of offloading schemes, e.g., centralized and cooperative [13], latency minimizing [15], [25], energy minimizing [26], latency-energy-tradeoff [17], [30], and so on. These two layers of complexity, i.e., openness and diversity, superpose, creating an unmanageable space for experiment evaluation in massive IoT system [3]. Accordingly, simulation and theoretic analysis seem to be viable options to evaluate the performance of fog task offloading. However, both of these two approaches demand a *descriptive model* of the task offloading procedure, which can easily adapt to different offloading criteria, yet manageable to examine the scalability of the fog-based IoT system, to address the "remedy or resource drain" concern.

### C. Our Approach: The Gravity Model

To tackle the aforementioned crux of performance evaluation, we take a modeling approach to describe the movements

Authorized licensed use limited to: N.C. State University Libraries - Acquisitions & Discovery S. Downloaded on April 13,2026 at 07:22:58 UTC from IEEE Xplore. Restrictions apply.

of data blocks as a result of task offloading, rather than trying to achieve a system-level optimal, which is both unnecessary and infeasible for a large heterogeneous system. To the best of our knowledge, we are the first to address the sacalability issue of fog systems with respect to delay and resource consumption. Our contributions can be summarized as follows.

We define evaluation metrics, namely, *task lifetime*, *device effort*, and *network effort*, to quantify the end-to-end delay, the amount of consumed storage/processing resource owned by individual fog nodes, and communication resource shared by the edge network, in a heterogeneous setting.

We propose a *gravity model* that views task-related data blocks as particles, whose movements are driven by a *gravity force* between the task particle and candidate fog nodes, and influenced by the preference of task nodes. The gravity function can describe multiple, even hybrid offloading criteria for systems of variable scales, so performances of large-scale fog systems with heterogeneous criteria can be examined.

We find that, under the gravity model, which applies to distance-, energy-, and/or delay-oriented offloading, the lifetime and device effort of individual tasks decrease as $O(1/N)$ over the network size $N$, while the network effort decreases much slower, even remain $O(1)$ when load balancing is adopted. Both the analysis and numerical results indicate that, though the expected application delay improves as the system scales, the total consumed network resource may exceeds what the system can provide, leading to a concern of resource drain.

The remainder of this article is organized as follows. We introduce the system model and performance metrics in the following Section II, to formulate the performance evaluation problem in the fog computing context. The proposed gravity offloading model is discussed in Section III. Analysis of the performance is derived in Section IV, which are validated with numerical results in Section V. Finally, we conclude this article in Section VI.

## II. System Model

In essence, a task offloading process is a sequence of intermittent movements of task-related data blocks across the network,[1] directions of which are determined by the offloading criteria. In this section, we introduce the scenario, network models, task model, and performance metrics to formulate the performance evaluation problem of task offloading processes.

### A. Node Model

Consider a fog computing system hosted by cellular networks and wireless LANs, a miniature example of which is illustrated by Fig. 1. In this network,[2] fog service is provided by eNodeBs/gNodeBs, access points (APs), and mobile devices (including smart phones, tablets, laptops, and so on), which we refer to as *fog nodes*. Any device that is authorized to offload task (to fog nodes) is referred to as a *task node*.

[1]In this article, we consider indivisible tasks to focus on the offloading process, as task break-down is a separate line of research that focus more on contextual information of application data.

[2]Considering resource bottlenecks mainly exist in wireless networks, we restrict the scope of the fog paradigm to the network edge in this discussion.

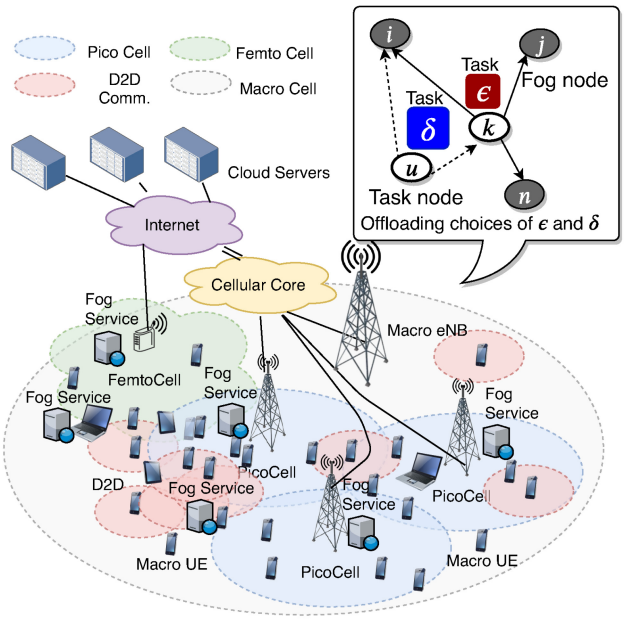



Fig. 1. Example of a fog-based IoT system hosted in the edge network: colors correspond to transmission ranges of different RATs listed in Table II.

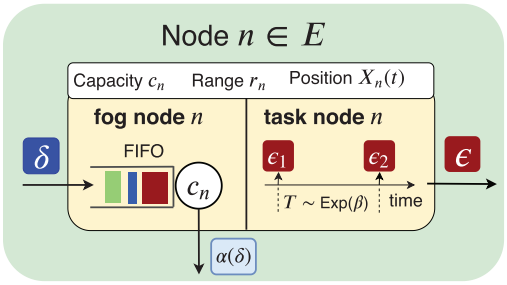



Fig. 2. Node model of fog-based task offloading system.

For example in Fig. 1, smart phone $k$ can act as a fog node to task $\delta$ originated from task node $u$, e.g., uploading temperature measurements from smart home devices, and as the task node to generate task $\epsilon$, e.g., video trans-coding.

Accordingly, each node $n \in E$ can act as two roles: a fog node, who is associated with a CPU of processing capacity $c_n$, and a first-in-first-out (FIFO) queue; and/or a task node, who generates tasks at the rate of $\beta$. As a fog node, node $n$ announces[3] its capacity $c_n$ within its transmission range $r_n$, while as a task node, node $n$ decides whether, and to whom, a generated task will be offloaded, as shown in Fig. 2. Since fog nodes may move, denote the position of node $n$ as $X_n(t) : T \to \mathbb{R}^2$, where $t \in T$ is the current time instant.

### B. Communication Model

For node $e \in E$, located at $X_e(t)$, denote its *neighbors* $N_e(t) = \{i \in E | d(X_i(t), X_e(t)) \leq \min\{r_e, r_i\}\}$, where $r_e$ is the transmission range of node $e$. Suppose node $e$ utilizes a wireless channel of bandwidth $\mathrm{BW}_{e \to n}(t)$ (Hz) to communicate with neighboring node $n \in N_e(t)$, located at

[3]The processing capacity $c_n$ is short (a few bytes) and fixed, so it may piggyback on link discovery messages. Consequently, task nodes can easily gather this information when it enters the communication range of others.

Authorized licensed use limited to: N.C. State University Libraries - Acquisitions & Discovery S. Downloaded on April 13,2026 at 07:22:58 UTC from IEEE Xplore. Restrictions apply.

TABLE II
RADIO ACCESS TECHNOLOGIES (RAT) IN IOT SYSTEMS

| | RAT | Range | Channel Bandwidth | Trans. Power | Frequency Bands | Peak Rate |
|---|---|---|---|---|---|---|
| Broad-band | 5G-NR FR2 (FemtoCell) | <10 m | 50, 100, 200, 400 MHz | 10-20 dBm | mmWave (24-50 GHz) | 20 Gbps |
| | 5G-NR FR2 (PicoCell) | <200 m | 50, 100, 200, 400 MHz | 20-24 dBm | mmWave (24-50 GHz) | 20 Gbps |
| | WiFi 6 (indoor-5GHz) | <50 m | 20, 40, 80, 160 MHz | <20 dBm | ISM bands (5 GHz) | 10 Gbps |
| | WiFi 6 (indoor-2.4GHz) | <50 m | 20, 40 MHz | <20 dBm | ISM bands (2.4 GHz) | 2 Gbps |
| | 5G-NR FR2 (MicroCell) | <1 Km | 50, 100, 200, 400 MHz | <23 dBm | Sub-6G (3.3-6 GHz) | 1 Gbps |
| | 5G-NR FR1 (MicroCell) | <1 Km | 5-100 MHz | <23 dBm | Sub-6G (3.3-6 GHz) | 1 Gbps |
| | LTE-M Cat2 (release 14) | <1 Km | 5 MHz | 20/23 dBm | LTE (0.9-2 GHz) | 7 Mbps |
| Narrow-band | NB-IoT Cat NB2 | <1 Km | 180 KHz | 20/23 dBm | LTE (700-900 MHz) | 200 Kbps |
| | LoRa-WAN | <5 Km | 125, 250 KHz | <14 dBm | ISM bands (2.4, 5, 6 GHz) | 50 Kbps |

TABLE III
ATTRIBUTES OF A TASK $\epsilon$ (DETERMINED AT $t_\epsilon$)

| Attribute | Range | Physical meaning |
|---|---|---|
| $t_\epsilon$ | $\mathbb{R}^+$ | the time instant when $\epsilon$ is generated |
| $L_\epsilon$ | $\mathbb{N}^+$ | volume of task $\epsilon$ (bits) |
| $\tau_\epsilon$ | $\mathbb{R}^+$ | timeout/delay bound of task $\epsilon$ |
| $S_\epsilon$ | $E$ | task node who generated $\epsilon$ |
| $D_\epsilon$ | $E$ | target fog node to offload/execute $\epsilon$ |
| $\phi_\epsilon$ | $[100, 400]$ | processing intensity (CPU cyles/bit) [31], [32] |
| $\alpha_\epsilon$ | $(0, 1]$ | coefficient of volume change after processing |

$X_n(t)$, where the noise power measures at $P_0 = \text{BW}_{e\to n} N_0 \text{T}$, $N_0 = 1.38065 \times 10^{-23}$, and $T = 290$ K. Then if node $e$ transmits with power $P_{e\to n}(t)$, and occupies a channel of width $\text{BW}_{e\to n}(t)$, both of which remain the same throughout the offloading process, the transmission rate from node $e$ to node $n$ can be obtained as a function of three variables, i.e.,

$$R_{e\to n}(t) = \text{BW}_{e\to n}(t) \log_2\left(1 + \frac{P_{e\to n}(t) \cdot d(X_e(t), X_n(t))^{-\kappa}}{\text{BW}_{e\to n}(t) \cdot N_0 \cdot T}\right) \quad (1)$$

where $\kappa$ is the path-loss exponent and $d(X_e(t), X_n(t))$ measures the distance between node $e$ and node $n$ at time $t$. Note that the three variables in (1), $\text{BW}_{e\to n}(t)$, $P_{e\to n}(t)$, and $d(X_e(t), X_n(t))$ are determined by the time instant $t_\epsilon$ when task $\epsilon$ is generated. But their values can not be set freely. Instead, they need to comply with the RAT adopted in the current offloading process, and may change over different tasks if the IoT device is multimode, e.g., equipped with software-defined radio (SDR). Commonly used RAT's in a fog-based IoT system are listed in Table II, where broadband and narrow-band RATs are both presented.

### C. Task Model

Consider a task[4] $\epsilon \in \Omega$, which is captured as a seven-element tuple $(t_\epsilon, L_\epsilon, \tau_\epsilon, S_\epsilon, D_\epsilon, \phi_\epsilon, \alpha_\epsilon)$, whose ranges and physical meanings can be found in Table III. For example, a file $\epsilon$ of size $L_\epsilon$, to be offloaded from the source node $S_\epsilon = k$ to the destination node $D_\epsilon = n$, needs to be completed by time instant $t_\epsilon + \tau_\epsilon$. Task $\epsilon$ also requires a processing intensity $\phi_\epsilon$, and expects the execution result $\alpha(\epsilon)$ of volume $\alpha_\epsilon L_\epsilon$, where coefficient $\alpha_\epsilon \in (0, 1]$ is the ratio between the volume of data block $\alpha(\epsilon)$ and that of data block $\epsilon$. For example, a video trans-coding task $\epsilon$ to compress a 1-min 1080p video clip to a 360p version has volume $L_\epsilon = 165$ Mb, processing intensity $\phi_\epsilon \simeq 290$ cycles/bit [31], [32], and coefficient $\alpha_\epsilon = (55 \text{ Mb}/165 \text{ Mb}) \simeq 0.33$, as the processing result $\alpha(\epsilon)$, i.e., the 1-min 360p video clip, is only 55 Mb in volume.

Among these, volume $L_\epsilon$, processing intensity $\phi_\epsilon$, and coefficient $\alpha_\epsilon$, may differ considerably among tasks, which has been demonstrated by IoT traffic analysis [29], [33] and projection of future IoT applications [34]. For instance, a text-translation task has a small $L_\epsilon$ (several kb), and coefficient $\alpha_\epsilon \simeq 1$, because the size of the translated file $\alpha(\epsilon)$ is comparable with that of the original file $\epsilon$; VR/AR tasks have large $L_\epsilon$ (Gb level [11], [34]), and coefficient $\alpha_\epsilon \simeq 1$, because both $\epsilon$ and $\alpha(\epsilon)$ are video clips with relatively large volume; while object recognition and cloud offloading tasks have medium $L_\epsilon$, and small coefficient $\alpha_\epsilon \simeq 0$, because the expected $\alpha(\epsilon)$ is much shorter than data block $\epsilon$.

All attributes of task $\epsilon$ are populated at their generation. Particularly, the offloading target $D_\epsilon$ decides the direction of $\epsilon$'s movements, and hence is most relevant to the delay and resource consumption performances. This key parameter is determined by the *offloading criterion*, accounting for the current status of the network, e.g., reachability, processing capacity, and cost of the fog service. For instance, under a delay-oriented criterion, task $\epsilon$ will most likely be offloaded to fog node $i$ in Fig. 1, who can complete the task with the minimum delay, according to task node $k$'s observation at $t_\epsilon$. If task $\epsilon$ is viewed as a particle, it moves toward node $i$, as if drawn by a probabilistic *gravity force* between the two, and the resource consumed during the offloading process can be viewed as the *effort* along its moving path. This observation motivates us to develop effort-based performance metrics and the gravity-based offloading model, for heterogeneous communication options and offloading criteria, respectively.

### D. Performance Metrics

To address the performance evaluation problem, we consider two aspects of the task offloading process, i.e., the *completion time* of individual tasks, to see if is a remedy to delay-sensitive IoT applications, and the *resource consumption* during an offloading process, to answer whether it is a resource drain to the edge network. The former is quantified by the *task lifetime* metric defined as follows.

[4]Technically, a task is a piece of computation work to be completed before a certain time, which usually contains a block of data to be processed. So we do not differentiate the two, and use "task" and "data block" interchangeably.

Authorized licensed use limited to: N.C. State University Libraries - Acquisitions & Discovery S. Downloaded on April 13,2026 at 07:22:58 UTC from IEEE Xplore. Restrictions apply.

*Definition 1:* Consider task $\epsilon$ generated by task node $S_\epsilon \in E$ at time $t_\epsilon$. The *lifetime* of $\epsilon$ is define as the time between its generation, and the time that the last bit of its expected result $\alpha(\epsilon)$ reaches the task node $S_\epsilon$, that is

$$\theta_\epsilon \coloneqq \min\left\{\tau_\epsilon,\ \inf\left\{t \geq 0 | S_\epsilon \in \mathcal{H}_t\big(\alpha(\epsilon)\big)\right\} - t_\epsilon\right\} \tag{2}$$

where $\tau_\epsilon$ is the timeout bound of task $\epsilon$, and $\mathcal{H}_t(\alpha(\epsilon)) \subset E$ denotes the set of nodes that has $\alpha(\epsilon)$ locally stored at $t$.

To IoT applications, task lifetime $\theta_\epsilon$, which equals to the task completion time in value, quantifies the quality of the offloading service provided by the fog system, and is upper bounded by the timeout $\tau_\epsilon$ of the task, while to the fog system, it measures the period of time that data block $\epsilon$ is physically present in the system.

During its lifetime, the offloading process of task $\epsilon$ can be decomposed into four stages:[5] 1) offloading (from $S_\epsilon$ to $D_\epsilon$); 2) queuing (at $D_\epsilon$); 3) processing (at $D_\epsilon$); and 4) responding (from $D_\epsilon$ back to $S_\epsilon$). In this process, three types of resources are consumed to complete the task offloading service, that is, communication resource (during the offloading and responding stage) shared by the edge network, storage resource (across the whole process), and processing resource (in the processing stage) located at individual devices. Inspired by the "effort" analogy, we quantify the amount of resource consumption as the integral of the unit consumption for a task ("power" of the force) over time. In this sense, no matter a task $\epsilon$ is offloaded or executed locally, the total amount of processing resource needed (unit: CPU cycles) equals to the product of data volume $L_\epsilon$ and processing intensity $\phi_\epsilon$, which is a constant value. Consumption of the storage and communication resource, on the other hand, are more difficult to analyze, as the unit consumption varies over time, for which the *device effort* and *network effort* are defined, respectively.

*Definition 2:* The *device effort* spent on task $\epsilon$ is defined as the (cumulative) amount of storage space in the system occupied by offloading/executing $\epsilon$ over time

$$\mathrm{DE}_\epsilon(t) \coloneqq \int_{t_\epsilon}^{t} L_\epsilon |\mathcal{H}_\epsilon(s)| + \alpha_\epsilon L_\epsilon |\mathcal{H}_{\alpha(\epsilon)}(s)|\ ds \tag{3}$$

where $|\mathcal{H}_\epsilon(s)|$ is the number of copies of task $\epsilon$ stored on any device/node in $E$. Denote $\mathrm{DE}_\epsilon \coloneqq \mathrm{DE}_\epsilon(t_\epsilon + \theta_\epsilon)$ as the total device effort spent during the offloading process of task $\epsilon$.

The device effort metric quantifies the total amount of storage resource consumed up to time $t$. For instance, in the offloading process of task $\epsilon$ from node $k$ to $n$, the storage space occupied by task $\epsilon$ is $2L_\epsilon$ during the queuing and processing stage, $L_\epsilon$ during the offloading stage, and $\alpha L_\epsilon$ during the responding stage. This is because both the task node $k$ and the fog node $n$ need to keep a copy of data block $\epsilon$ to avoid transmission failures or processing failures. For the communication resource, we define the *network effort* metric with respect to the traversing path of data block $\epsilon$.

*Definition 3:* For task $\epsilon$ with lifetime $\theta_\epsilon$, the *network effort* of $\epsilon$ at time $t \in [t_\epsilon, t_\epsilon + \theta_\epsilon]$ is defined as the summation of transmission distance-time products, that is

$$\mathrm{NE}_\epsilon(t) \coloneqq \sum_{s \in \mathcal{T}^\epsilon} \sum_{i \in \Delta\mathcal{H}_\epsilon(t)} T^\epsilon_{p(i)\to i} d\big(X_i(s), X_{p(i)}(s)\big) + \sum_{s \in \mathcal{T}^{\alpha(\epsilon)}} \sum_{j \in \Delta\mathcal{H}_{\alpha(\epsilon)}(t)} T^{\alpha(\epsilon)}_{p(j)\to j} d\big(X_j(s), X_{p(j)}(s)\big) \tag{4}$$

where $\Delta\mathcal{H}_\epsilon(t) = \mathcal{H}_\epsilon(t) \backslash \mathcal{H}_\epsilon(t^-)$ is the set of new recipients of data block $\epsilon$ at time $t$, set $\mathcal{T}^\epsilon = \{s \in (t_\epsilon, t] | \Delta\mathcal{H}_\epsilon(t) \neq \phi\}$ contains the time instants when $\epsilon$ is transmitted, and $p(i)$ denotes the source node of a transmission to node $i$. Similarly, we refer to $\mathrm{NE}_\epsilon \coloneqq \mathrm{NE}_\epsilon(t_\epsilon + \theta_\epsilon)$ as the total network effort spent on the offloading process of task $\epsilon$.

The network effort metric quantifies the amount of shared communication resource it takes the fog system to perpetuate the movements of data blocks over the air interface during the offloading process, such as spectrum slices, time slots, transmission opportunities, and so on. We consider distance-time product rather than bandwidth-time product because spectrum resource (measured by channel bandwidth) may not be compared across RAT systems, but the occupation of space and time resource are universal. Note that the two types of efforts, i.e., device effort and network effort, can not be added together directly, because they quantify resource consumption in different types (and are hence in different units). Nonetheless, they jointly illustrate the amount of resources spent on the offloading process, and can be compared (separately) among offloading processes of different tasks.

It is worth noticing that the performance metrics are defined with respective to movements of data blocks, and hence easily apply to scenarios other than task offloading. In this article, we restrict ourselves to one-hop offloading of indivisible tasks, which are the most prevalent in a fog-based IoT system. To be more specific, we aim to find how the expected task lifetime $\mathbb{E}_\Omega(\theta)$, total device effort $\mathbb{E}(\sum_\epsilon \mathrm{DE}_\epsilon)$, and network effort $\mathbb{E}(\sum_\epsilon \mathrm{NE}_\epsilon)$ scales with the system size $N = |E|$, to address the "remedy or resource drain" concern.

## III. Gravity Rule: One Model for Multiple Offloading Criteria

Seemingly straight forward, the three performance metrics are, as a matter of fact, challenging to evaluate for a task offloading process due to the following reasons.

1) *Crux of Diversity:* A task node may employ different RATs, as well as different offloading criteria with or without internodal coordination, resulting in an unmanageable space for performance evaluation.
2) *Curse of Openness:* The lifetime, device effort, and network effort of an individual task are inevitably affected by other tasks' offloading processes, which are unclear as the size of the system $N$ grows.

The first challenge on diversity calls for a generic offloading model, while the second on openness requires meticulous analysis on the competition among tasks. In this section, we introduce a *gravity*-based offloading model to tackle the crux of diversity, and leave analysis to the next section. Under the gravity setting, different offloading criteria can be written as gravity functions quantifying the "suitableness" of candidate

[5]Task node $S_\epsilon$ may choose to execute task $\epsilon$ locally, then $D_\epsilon = S_\epsilon$ and the offloading process only contains the queuing stage and the processing stage.

Authorized licensed use limited to: N.C. State University Libraries - Acquisitions & Discovery S. Downloaded on April 13,2026 at 07:22:58 UTC from IEEE Xplore. Restrictions apply.

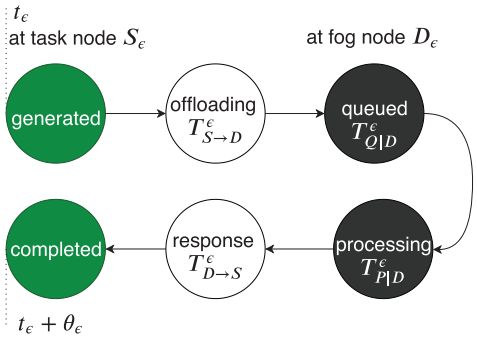



Fig. 3. State transitions of a task/data block $\epsilon$ during an offloading process.

fog nodes. Then, each task node selects a qualified candidate with a *probability* that is proportionate to the gravity force.

### A. Gravity-Based Probabilistic Offloading

In this model, the four-stage offloading process of task $\epsilon$ can be described by the following state transitions of data block $\epsilon$ and the corresponding $\alpha(\epsilon)$, as depicted in Fig. 3.

1) At time $t_\epsilon$, node $S_\epsilon = k \in E$ generates task $\epsilon$, so the data block is only stored at nodes in set $\mathcal{H}_\epsilon(t_\epsilon) = \{k\}$.
2) Task node $k$ chooses the offloading target $D_\epsilon = n$ out of the candidate set $N_k(t_\epsilon) \cup \{k\}$, with an *offloading probability* $\mathbb{P}(D_\epsilon = n)$.
3) Task $\epsilon$ is then transmitted from node $k$ to node $n$ at data rate $R_{k\to n}(t_\epsilon)$ and $\mathcal{H}_\epsilon(t_\epsilon + T^\epsilon_{k\to n}) = \{n, k\}$. This step is omitted if $k$ decides to execute $\epsilon$ locally, and $T^\epsilon_{k\to n} = 0$.
4) Upon receiving task $\epsilon$, node $n$ will put $\epsilon$ in a queue if its processor is busy, and start executing $\epsilon$ when all of the tasks queued before are completed.
5) When task $\epsilon$ is completed (after queuing delay $T^\epsilon_{Q|n}$ and processing delay $T^\epsilon_{P|n}$), the corresponding result $\alpha(\epsilon)$ of size $\alpha_\epsilon L_\epsilon$ will be transmitted back to its corresponding task node $k$ at the data rate of $R_{n\to k}(t)$.
6) At time instant $t_\epsilon + T^\epsilon_{k\to n} + T^\epsilon_{Q|n} + T^\epsilon_{P|n} + T^\epsilon_{n\to k}$, the task response $\alpha(\epsilon)$ is returned to node $k$ at the same data rate $R_{k\to n}(t_\epsilon)$.

The probabilistic nature of the gravity model is captured by the *offloading probability* $\mathbb{P}(D_\epsilon = n)$ in step 2), formally defined as follows:

$$\mathbb{P}(D_\epsilon = n) := \begin{cases} 0, & \text{if } \sum_{i\in N_k(t_\epsilon)} F(\epsilon, i) = 0 \\ \frac{\rho(F(\epsilon,n))}{\sum_{i\in N_k(t_\epsilon)} \rho(F(\epsilon,i))}, & \text{otherwise} \end{cases} \tag{5}$$

where gravity $F : \Omega \times E \to [0, \infty)$ is a nonnegative function, and the selectiveness $\rho : [0, \infty) \to [0, \infty)$ is an increasing, convex function, both of which are chosen by task node $S_\epsilon$. The former reflects node $S_\epsilon$'s unique objective in task offloading, e.g., energy saving, latency reducing, and so on. The selectiveness function $\rho(\cdot)$ allows the task node to fine-tune its preferences, that is, how reluctant it is against adjustments due to load balancing reasons. It may take any form as long as it is an increasing function. For example, we may set $\rho(x) = x^\gamma$, where $\gamma$ is the selectiveness coefficient. The larger $\gamma$, the more likely task node $S_\epsilon$ will favor fog nodes imposing the highest gravity, i.e., it is the more reluctant to settle for 'suboptimal' offloading choices. Together, the two functions $\{F, \rho\}$ compose an offloading profile of task node $S_\epsilon$, in which different offloading criteria can be described in a unified form to address the crux of diversity.

From the perspective of probabilistic/randomized offloading setting, our model bears some resemblance with [15], who proved the existence of an equilibrium, and solved the optimization problem of delay minimizing with a distributed algorithm. Nonetheless, our model serves a different purpose, that is, describing heterogeneous offloading criteria, and is hence more general as it is not restricted to delay-oriented offloading as in [15]. Moreover, the "optimal" and "suboptimal" are determined solely by the task node of interest, without any collaboration that may induce overhead, nor information of other tasks. This setting of local decision is especially practical for IoT devices that lack the capacity for cooperation, e.g., due to power constraints. Next, we discuss the form of gravity functions for typical and generic offloading schemes.

### B. Typical and Generic Gravity Rules

The gravity force $F(\epsilon, n)$ is meant to quantify the "attractiveness" of fog node $n$, in regard to processing task $\epsilon$. Considering most offloading criteria have an objective to optimize, such as delay, energy consumption, etc., we define the gravity function as the performance gain achieved by offloading, compared to local processing, or the remaining allowance against a performance budget. We list a few example rules for typical application scenarios.

*1) Distance-Oriented Gravity Rule:* The simplest offloading scheme is to always offload to fog nodes nearby, as long as there is one within the communication range of the task node. The corresponding gravity function can be written as follows:

$$F(\epsilon, n) = \frac{1}{d(X_k(t_\epsilon), X_n(t_\epsilon))} \quad \forall n \in N_e(t_\epsilon). \tag{6}$$

This rule applies to IoT devices that lack the capacity to execute complex algorithms or participate in cooperation, as the only parameter is transmission distance, which can be easily deduced from the received signal power. To always select the nearest fog node, the selectiveness function $\rho(x)$ can be set to $\rho(x) = x^\gamma$ with a large $\gamma$, e.g., $\gamma = 10$.

*2) Delay-Oriented Gravity Rule:* As shown in the state transition diagram (Fig. 3), for a task $\epsilon$ generated by $S_\epsilon = k$, three out of the four delay components can be determined, once the offloading decision $D_\epsilon = n$ is made as at time $t_\epsilon$: the uplink transmission delay $T^\epsilon_{S\to D} = (L_\epsilon/[R_{k\to n}(t_\epsilon)])$, processing delay $T^\epsilon_{P|D} = (L_\epsilon \phi_\epsilon / c_n)$, and downlink transmission delay $T^\epsilon_{D\to S} = (\alpha_\epsilon L_\epsilon/[R_{k\to n}(t_\epsilon)])$.

The queuing delay $T^\epsilon_{Q|n}$, on the other hand, also depends on the offloading processes of *other* tasks, so it is bounded by

$$0 \le T^\epsilon_{Q|n} \le \frac{Q_n\left(t_\epsilon + T^\epsilon_{k\to n}\right) + Y_n}{c_n} < \frac{Q_n(t_\epsilon) + Y_n}{c_n} \tag{7}$$

where $Q_n(t)$ denotes the processing queue length of fog node $n$ at time $t$, and random variable $Y_n \in [0, \infty)$ is the workload offloaded to node $n$, during the transmission of task $\epsilon$, i.e.,

Authorized licensed use limited to: N.C. State University Libraries - Acquisitions & Discovery S. Downloaded on April 13,2026 at 07:22:58 UTC from IEEE Xplore. Restrictions apply.

interval $[t_\epsilon, t_\epsilon + T^\epsilon_{k\to n}]$. Note that task node $k$, as a fog node, is also accepting tasks offloaded from other task nodes, hence local execution may induce queuing delay as well.

Then, the corresponding gravity function can be defined as follows:

$$F(\epsilon, n) = \left[\tau_\epsilon - \left(T^\epsilon_{k\to n} + T^\epsilon_{P|n} + T^\epsilon_{n\to k}\right)\right]^+ = \left[\tau_\epsilon - \left(\frac{(1+\alpha_\epsilon)L_\epsilon}{R_{k\to n}(t_\epsilon)} + \frac{L_\epsilon \phi_\epsilon}{c_n}\right)\right]^+ \quad (8)$$

where $c_n$ (cycles/s) is the capacity of candidate fog node $n$, and $[x]^+ = \max\{x, 0\}$.

This rule applies to many IoT applications, as reducing application latency is the primary motivation [2] of fog computing. In this case, devices tend to transmit with the highest data rate that an RAT can provides, so the number of variables in (1) can be further reduced, that is

$$R_{k\to n}(t) = g_d(d(X_n(t), X_k(t))). \quad (9)$$

The reason behind this reduction is that the possible value of $P_{k\to n}(t_\epsilon)$ and $\mathrm{BW}_{e\to n}(t_\epsilon)$ are actually determined by the range of an RAT, as indicated by Table II. For instance, RAT 5G-NR femto-cell and pico-cell, although reaching a staggering 5–40 Gb/s in lab environments [35], [36] with the mmWave technology, only covers tens of meters of communication, due to the acute attenuation in the 26–60 GHz bands. For the range of hundreds of meters, WiFi and LTE/5G micro-cell are available, offering a transmission rate at ten to a hundred Mb/s [37]. Finally, for the range of over 1 km, LTE-M (4–7 Mb/s) and narrow-band RATs can be adopted, such as NB-IoT (200 kb/s) and LoRA WAN (50 kb/s). In this sense, $g(\cdot)$ is a piece-wise function over the distance, and we can further simplify (9) as $g_d(d(X_n(t), X_k(t))) \triangleq R_{\max} - \psi d(X_n(t), X_k(t))$ in the analysis, where $R_{\max}$ is the maximum transmission rate that can be achieved in the fog system, and $\psi$ (kb/s/m) is the rate attenuation factor.

*3) Energy-Oriented Gravity Rule:* For pocketable, battery-operated task nodes, e.g., IoT sensors, energy consumption is a major concern. So the ultimate goal of offloading is to conserve as much energy as possible, i.e., task $\epsilon$ may be offloaded to node $n$ only if the energy spent on transmission $P_{k\to n}T^\epsilon_{k\to n}$ is less than the energy consumed by local execution $E^\epsilon_k = \xi\phi_\epsilon L_\epsilon$, where constant $\xi$ (Joule/cyle) is the amount of energy consumed by each CPU cycle. Then the gravity function can be defined as the amount of energy saved if the corresponding task is offloaded, that is

$$F(\epsilon, n) = \left[\xi\phi_\epsilon L_\epsilon - \frac{P_{k\to n}(t_\epsilon)L_\epsilon}{R_{k\to n}(t_\epsilon)}\right]^+ = \left[\xi\phi_\epsilon L_\epsilon - \frac{L_\epsilon}{g_e\left(\frac{P_{k\to n}(t_\epsilon)}{\mathrm{BW}_{k\to n}(t_\epsilon)}; d(X_k(t_\epsilon), X_n(t_\epsilon))\right)}\right]^+ \quad (10)$$

where the duel-variate function $g_e(\cdot;\cdot)$ is defined as follows:

$$g_e(x; y) = \frac{1}{x}\log_2\left(1 + \frac{xy^{-\kappa}}{N_0 \mathrm{T}}\right). \quad (11)$$

Note that $g_e([P_{k\to n}(t_\epsilon)/\mathrm{BW}_{k\to n}(t_\epsilon)]; d(X_k(t_\epsilon), X_n(t_\epsilon))) = (P_{k\to n}(t_\epsilon)/R_{k\to n}(t_\epsilon))$ in value, which is different from the delay-oriented rule, where the single-variate function $g_d(d(X_k(t_\epsilon), X_n(t_\epsilon))) = R_{k\to n}(t_\epsilon)$. But they are both functions of the distance.

*4) Generic Form of the Gravity Function:* For the ease of analysis, we generalize the aforementioned gravity functions into the following form:

$$F(\epsilon, n) = f\left(A_\epsilon - \frac{B_\epsilon}{c_n} + \frac{G_\epsilon}{g(d(X_k(t_\epsilon), X_n(t_\epsilon)))}\right) \quad (12)$$

quantities in which have the following properties.

1) Function $f : \mathbb{R} \to \mathbb{R}$ is nondecreasing; $f(a) > 0$ for any $a > 0$; and $f(a) = 0$ otherwise.
2) Function $g : \mathbb{R}^+ \to \mathbb{R}$ is a nondecreasing function of the transmission distance $d(X_k(t_\epsilon), X_n(t_\epsilon))$.
3) Quantities $A_\epsilon$, $B_\epsilon$, and $G_\epsilon$ are nonnegative constants, which are determined solely by the attributes of task $\epsilon$, and satisfy constraint $B_\epsilon + G_\epsilon > 0$.

Together with the selectiveness function $\rho(\cdot)$, the generic rule in (12) can describe all the aforementioned gravity rules. For the distance-oriented rule in (6), $A_\epsilon = B_\epsilon = 0$, $G_\epsilon = 1$, and function $g(x) = x$, while for the delay-oriented rule in (8), $A_\epsilon = \tau_\epsilon$, $B_\epsilon = \phi_\epsilon L_\epsilon$, $G_\epsilon = (1+\alpha_\epsilon)L_\epsilon$, and $g(x) = -g_d(x)$, whose absolute value $|g(x)|$ measures the data transmission rate at distance $x$. Any joint scheme that can be described as the linear combination of (6) and (8) still has the same form as (12), which indicates the versatility of this generic model. For energy-oriented offloading, function $g(\cdot) = -g_e(\cdot;\cdot)$ is multivariate, in which $(P_{k\to n}(t_\epsilon)/\mathrm{BW}_{k\to n}(t_\epsilon))$ is determined by the power control mechanism.

### C. Offloading Probabilities Under the Generic Gravity Rule

Under the generic gravity rule, consider task $\epsilon$, whose associated data block is of size $L_\epsilon$, processing intensity $\phi_\epsilon$, and coefficient $\alpha_\epsilon$. For any fog node $n \in E$, let random variable $V_n = d(X_k(t_\epsilon), X_n(t_\epsilon))$ denote the distance between $n$ and $k$ at time $t_\epsilon$, with PDF $f_V(v)$ $(v \in [0, +\infty))$, and let random variable $C_n$ denote its processing capacity, with PDF $f_C(c)$ $(c \in [c_{\min}, c_{\max}])$. First, we present the probability that task $\epsilon$ will be executed locally.

*Lemma 1:* Under the generic gravity function defined in (12), the probability that task node $S_\epsilon = k$ decides to process task $\epsilon$ locally is given by

$$\mathbb{P}(D_\epsilon = k) = 1 - \left(1 - \mathbb{P}^\epsilon_{pos}\right)^{N-1}. \quad (13)$$

Particularly, when $A_\epsilon \neq 0$, quantity $\mathbb{P}^\epsilon_{pos}$ can be calculated as follows:

$$\mathbb{P}^\epsilon_{\mathrm{pos}} = \int_{\frac{B_\epsilon}{A_\epsilon}}^{c_{\max}} \int_0^{g^{-1}\left(\frac{cG_\epsilon}{cA_\epsilon - B_\epsilon}\right)} f_V(v)dv f_C(c)dc \quad (14)$$

where $c_{\max}$ is the maximum capacity among fog nodes in $E$.

Proof of Lemma 1 can be found in Appendix A in the supplementary material. Lemma 1 provides a criterion of selecting offloading candidates from $N_k(t_\epsilon)$ for task $\epsilon$, as well as the probability that $\epsilon$ will be executed locally at task node $k$. Particularly when node statistics (captured by the distributions

Authorized licensed use limited to: N.C. State University Libraries - Acquisitions & Discovery S. Downloaded on April 13,2026 at 07:22:58 UTC from IEEE Xplore. Restrictions apply.

of distance $V_n$ and processing capacity $C_n$) are known, offloading probabilities can be derived numerically. Consequently, performance of the offloading system can be estimated without costly simulation or experiments. Compared with task statistics [15], which are highly diverse and prone to privacy leak, node statistics are much easier to obtain. With node statistics, we discuss the offloading probability $\mathbb{P}(D_\epsilon = n)$ defined in (5).

*Theorem 1 (Offloading Probabilities):* Let $\mu_C = \int_{c_{\min}}^{c_{\max}} cf_C(c)dc$ denote the mean capacity of fog nodes in $E$, and $F_V(v) = \int_0^v f_V(s)ds$ denote the CDF of distance $V$ between task node $k$ and a randomly chosen fog node at time $t_\epsilon$. The probability $\mathbb{P}(D_\epsilon = n)$ that task $\epsilon$ is offloaded to node $n$ with capacity $c_n$ and distance $v_n$ is upper bounded by

$$\frac{\rho(F(\epsilon,n))}{\mathrm{LB}(v_{\max})\rho\left(A_\epsilon - \frac{B_\epsilon}{\mu_C} + \frac{G_\epsilon}{F_V(v_{\max})}\int_0^{v_{\max}} \frac{f_V(v)}{g(v)}dv\right)} \tag{15}$$

where $v_{\max} = g^{-1}([c_{\max}G_\epsilon/(B_\epsilon - c_{\max}A_\epsilon)])$ and $\mathrm{LB}(v_{\max}) = NF_V(v_{\max})f_C(c_{\max})([-B_\epsilon G_\epsilon g(v_{\max})]/[(A_\epsilon g(v_{\max}) + G_\epsilon)^2])$ is a lower bound of the total number of offloading candidates.

Proof of Theorem 1 can be found in Appendix B in the supplementary material. There are two implications from Theorem 1.

*1) Local Action and Global Status:* This theorem connects the local offloading action to the global status of the system. With the set of offloading probabilities $\{\mathbb{P}(D_\epsilon = n)\}_{n\in E}$, interactions of multiple tasks can then be examined to obtain the performances of the fog system.

*2) Invariant Property:* Regardless of offloading criteria, probability $\mathbb{P}(D_\epsilon = n) = \Theta(1/N)$ holds when selectiveness function $\rho(\cdot)$ takes a polynomial form.

## IV. Performance Evaluation of Fog System

With Theorem 1 at hand, this section addresses the other major challenge in the performance evaluation problem, that is, competition and interactions among tasks and fog nodes. Specifically, we leverage Theorem 1 to examine the expected lifetime and efforts for a single task $\epsilon$ to understand the system-level performances, that is

$$\begin{cases} \theta_\epsilon^n = T_{k\to n}^\epsilon + T_{Q|n}^\epsilon + T_{P|n}^\epsilon + T_{n\to k}^\epsilon, \\ \mathrm{DE}_\epsilon^n(\theta_\epsilon) = L_\epsilon\left(T_{k\to n}^\epsilon + 2T_{Q|n}^\epsilon + 2T_{P|n}^\epsilon\right) + \alpha_\epsilon L_\epsilon T_{n\to k}^\epsilon \\ \mathrm{NE}_\epsilon^n(\theta_\epsilon) = T_{k\to n}^\epsilon d(X_k(t_\epsilon), X_n(t_\epsilon)) \\ \qquad\qquad + T_{n\to k}^\epsilon d(X_k(\theta_\epsilon + t_\epsilon), X_n(\theta_\epsilon + t_\epsilon)) \end{cases} \tag{16}$$

when the offloading target $D_\epsilon$ is chosen to be fog node $n \in E$, all quantities in (16) can be obtained by definitions, except the queuing delay $T_{Q|n}^\epsilon$, which is the result of simultaneous task offloading to the same fog node. To overcome this challenge, we first analyze the queuing delay to derive the task lifetime, which is one of the most important performance metric in real-time IoT applications, such as AR/VR.

### A. Queuing Delay $T_{Q|n}^\epsilon$ Under the Generic Gravity Rule

Consider task $\delta$ that is also offloaded to fog node $n$, which happens with probability $\mathbb{P}(D_\delta = n)$. Its execution may add to the queuing delay of $\epsilon$, if and only if the last bit of $\delta$ arrived at node $n$ before the last bit of $\epsilon$, such that $\delta$ will be processed before $\epsilon$, i.e., $t_\epsilon < t_\delta + T_{S_\delta\to n}^\delta < t_\epsilon + T_{S_\epsilon\to n}^\epsilon$. Here, we only discuss the case of $N\mu_C > N\beta\mu_L\mu_\phi$, that is, the generated execution load does not exceed the processing capacity of the entire fog system, because otherwise the system will eventually be overloaded, resulting in an increasing queuing delay that goes to infinity. Then, we have the following results on the expected queuing delay and expected lifetime.

*Theorem 2 (Queuing Delay and Task Lifetime):* For offloading procedures that can be described by the gravity function in (12), the expected queuing delay $\mathbb{E}(T_{Q|n}^\epsilon) = O(1/N)$, when $\beta\mu_L\mu_\phi < \mu_C$. The expected lifetime $\mathbb{E}(\theta_\epsilon)$ decreases as $O(1/N)$ first, and converges to a constant value $\theta$.

*Proof:* With the divide-and-conquer method depicted in Fig. 8, Appendix B in the supplementary material, we can upper bound the expected amount of data $\mathbb{E}(Y_n)$ offloaded to $n$ during time interval $[t_\epsilon, t_\epsilon + T_{S_\epsilon\to n}^\epsilon]$ as follows:

$$\mathbb{E}(Y_n) \le \beta N T_{k\to n}^\epsilon \int_0^{g^{-1}(0)} \frac{f_V(v)}{|N_+(t_\epsilon)|} \cdot \int_0^{T_{k\to n}^\epsilon} \mu_\phi s H_L\big(g(v)\big(s - T_{k\to n}^\epsilon\big)\big)dsdv \tag{17}$$

where $\mu_\phi$ is the expected processing intensity of tasks in $\Omega$, and function $H_L(l) = \int_0^l xf_L(x)dx$. Then, the expected queuing time can be upper bounded by

$$\mathbb{E}\left(T_{Q|n}^\epsilon\right) \le \frac{\mathbb{E}(Y_n)}{c_n} + \left[\frac{Q_n(t_\epsilon)}{c_n} - T_{k\to n}^\epsilon\right]^+ \tag{18}$$

where $Q_n(t_\epsilon)$ is the remaining load of fog node $n$ at time $t_\epsilon$.

From a realistic point of view, the peak data rates in existing cellular and WiFi systems measure at 10 Mb/s to 10 Gb/s, while the processing capacity of most mobile devices measure at GHz level (e.g., 1.4-GHz Raspberry Pi 3, and 2.6–3.0 GHz smart phone processors), so the second term of (18) is less significant compared to the first term. As $|N_+(t_\epsilon)| = \Theta(N)$, we have $\mathbb{E}(T_{Q|n}^\epsilon) = O(1/N)$ from (17) and (18), especially when $\beta$ is sufficiently large $\beta > 0.01$.

Then the expected task lifetime $\mathbb{E}(\theta_\epsilon)$ follows from the exact form of the queuing delay, as the other three delay terms in (16) do not affect the scaling behavior over the network size $N$, nor are they affected by the task generation rate $\beta$. Then when the network size $N$ is sufficiently large such that $\mathbb{E}(T_{Q|n}^\epsilon)$ is small enough to be neglected, the expected lifetime $\mathbb{E}(\theta_\epsilon)$ will remain constant over the network size $N$. ■

It should be noted that the scaling behavior of both the queuing time and the lifetime has the same coefficient $\beta$, which is introduced by (17). Consequently, the $O(1/N)$ decreasing is more evident when the system is heavily loaded, i.e., $\beta$ is relatively large. Contrarily, when $\beta < 0.01$, the probability that transmissions of other offloaded tasks (to node $n$) complete in time interval $[t_\epsilon, t_\epsilon + T_{S_\epsilon\to n}^\epsilon]$ will be small, such that $\mathbb{E}(T_{Q|n}^\epsilon) \simeq [(Y_n/c_n) - T_{k\to n}^\epsilon]^+$, and the task lifetime will appear to be almost constant over the system size $N$.

### B. Device Effort and Network Effort

Analysis of the queuing delay outlines a way to obtain the exact upper bound of the device effort of individual tasks, i.e., $\mathbb{E}(\mathrm{DE}_\epsilon) = \sum_{i\in N_+(t_\epsilon)\cup\{k\}} \mathrm{DE}_\epsilon^n P(D_\epsilon = i)$, which is also true for the network effort. Moreover, performance of the fog system can be evaluated through the gravity model, when the

Authorized licensed use limited to: N.C. State University Libraries - Acquisitions & Discovery S. Downloaded on April 13,2026 at 07:22:58 UTC from IEEE Xplore. Restrictions apply.

node mobility model, or the distribution of internodal distances $\{V_n\}_{n\in E}$ is known. Particularly, when nodes in $E$ move according to a speed-constraint random walk model in the square region $[-(a/2),(a/2)]^2 \subset \mathbb{R}^2$ of interest, such that the maximum speed $speed_{\max} << a$ and there is no boarder effect, PDF $f_V(v)$ has the following form.

*Lemma 2 (Special Case of [38]):* Consider a square region $[-(a/2),(a/2)]^2 \subset \mathbb{R}^2$. For two randomly chosen points $X_1, X_2 \in [-(a/2),(a/2)]^2$ in this region, the distance $V = d(X_1, X_2)$ is a random variable with PDF as follows:

$$f_V(v) = \begin{cases} \frac{\pi}{a^2} + \frac{v^2}{a^4}, & 0 \le v \le a \\ -\frac{2+\pi}{a^2} + \frac{2}{a^2}\arcsin\left(\frac{a}{v}\right) + \frac{4\sqrt{s^2-a^2}}{a^3}, & a \le v \le \sqrt{2}a \\ 0, & \text{otherwise.} \end{cases} \tag{19}$$

Proof of Lemma 2 can be found as a special case of [38], and is hence omitted here. With this lemma, we discuss data offloading processes under the distance-oriented and delay-oriented offloading rules. Without loss of generality, we make the following assumptions on tasks and nodes in the analysis.

1) The data volume $L$ and processing intensity $\phi$ of tasks satisfy uniform distributions, with mean $\mu_L$ and $\mu_\phi$, respectively.
2) The data transmission rate $R_{n\to k}$ takes the form stated in (9).
3) The selectiveness function takes the form of $\rho(x) = x$.

For these two offloading criteria unifies under the generic gravity rule, we have Theorem 3.

*Theorem 3 (Network and Device Effort):* For distance-oriented and delay-oriented offloading, with selectiveness function $\rho(x) = x$, the expected network effort spent on offloading task $\epsilon \in \Omega$ does not change over the system size $N$, i.e., $\mathbb{E}_{\epsilon\in\Omega}(\mathrm{NE}_\epsilon) = O(1)$. The expected device effort $\mathbb{E}_{\epsilon\in\Omega}(\mathrm{DE}_\epsilon)$ first decreases as $O(1/N)$ to a constant value.

Leveraging the gravity model, Theorem 3 outlines the scaling behavior of resource consumption during an offloading process. It indicates that, as the size $N$ of the fog system increases, the total device effort of the system $\mathbb{E}(\sum_{\epsilon\in\Omega}\mathrm{DE}_\epsilon)$ will increase slowly (sublinearly) at first, but rather quickly (linearly) when the size of the system exceeds a certain value. The network effort $\mathbb{E}(\sum_{\epsilon\in\Omega}\mathrm{NE}_\epsilon)$, on the other hand, increases linearly with the size of the system. In other words, there exists an optimal size of an offloading system, beyond which, the application-level delay $\mathbb{E}_\Omega(\theta_\epsilon)$ will no longer benefit from the growth of a fog system. Instead, the linearly increasing communication and coordination demand will lead to a drain of networking resource in the edge network.

Next, we prove Theorem 3 for the distance-oriented and delay-oriented offloading scenarios separately.

*1) Distance-Oriented Offloading:* In this case, a task $\epsilon$ will be offloaded to fog node $n$ with probability $\mathbb{P}(D_\epsilon = n)$, which is reversely proportionate to distance $V_n$, as indicated in (6). For task $\epsilon$ under this rule, the only criterion is the transmission range, i.e., $v_{\max} = (R_{\max}/\psi)$, because $F(\epsilon, n) > 0$ holds for any fog node $n \in E$. Note that, by Lemma 2, the probability that two entities are located at the exact same location, i.e., $\mathbb{P}(F_V(v = 0))$, is strictly zero, and that it is very unlikely (with a probability less than $(\pi/a^2) + (1/3a^4)$) that distance $V_n$ is less than 1 unit length, especially when $a$ is large. Therefore, gravity $F(\epsilon, i)$ is well-defined ($< \infty$) for almost all offloading candidates. To prove Theorem 3 for this scenario, we first derive the network and device effort of an individual task.

*Proposition 1:* Consider a distance-oriented offloading scheme, with selectiveness function $\rho(x) = x$. For task $\epsilon$ offloaded to fog node $n$, we have

$$\mathbb{E}\left(\mathrm{NE}_\epsilon^n\right) = \frac{(1+\alpha)L_\epsilon V_n}{R_{\max} - \psi V_n} \tag{20}$$

$$\mathbb{E}\left(\mathrm{DE}_\epsilon^n\right) = L_\epsilon\left[\frac{(1+\alpha)L_\epsilon}{R_{\max} - \psi V_n} + \frac{2\phi_\epsilon L_\epsilon}{c_n} + 2\mathbb{E}\left(T_{Q|n}\right)\right] \tag{21}$$

where expected queuing delay $\mathbb{E}(T_{Q|n}) \le ((\beta\mu_\phi R_{\max} L_\epsilon^3)/(12Nc_n(R_{\max} - \psi V_n)^3))$.

Proof of Proposition 1 can be found in Appendix c in the supplementary material. Proposition 1 gives the exact expected effort for task $\epsilon$, if it is offloaded to fog node $n$. From the perspective of the system, we discuss the mean expected effort $\mathbb{E}(\mathrm{NE}_\epsilon)$ and $\mathbb{E}(\mathrm{DE}_\epsilon)$ with respect to the network size $N$.

*Corollary 1:* In distance-oriented offloading, the expected network effort $\mathbb{E}(\mathrm{NE}_\epsilon)$ to offload any task $\epsilon \in \Omega$ is $O(1)$ with respect to network size $N$. The device effort $\mathbb{E}(\mathrm{DE}_\epsilon)$ decreases as $O(1/N)$ and converges to a constant value.

Proof of Corollary 1 can be found in Appendix D in the supplementary material.

*2) Delay-Oriented Offloading:* Next, we consider the more complicated, but most widely adopted offloading criterion, i.e., delay-oriented offloading schemes depicted by (8).

We adopt the same assumption as in the distance-based offloading: there are a total number of $N = |E|$ fog nodes in the region $[-(a/2),(a/2)]^2$, in which $a$ is large enough, such that $a >> \max_{e\in E}\{r_e\}$. Suppose a task node $k$ located at the center of this region, i.e., location $(0, 0)$, and generates task $\epsilon$ to offload at time $t_\epsilon$. Denote the distance from fog node $n$ to this task node $k$ as $V_n^\epsilon := d(X_k(t_\epsilon), X_n(t_\epsilon))$, which obeys the distribution in Lemma 2. In delay-oriented offloading, whose gravity function follows the generic form, Theorems 1 and 2 still hold, so we have the following corollary, similarly as Proposition 1 and Corollary 1.

*Corollary 2:* In delay-oriented offloading, the expected network effort to offload an individual task $\mathbb{E}(\mathrm{NE}_\epsilon)$ is constant with respect to the size of the fog network $N$, and the expected device effort $\mathbb{E}(\mathrm{DE}_\epsilon) = O(1/N)$, which decreases first, and eventually converges to a constant value as $N$ increases.

Proof of Corollary 2 can be found in Appendix E in the supplementary material.

Combining Corollaries 1 and 2, we proved Theorem 3 for the gravity offloading model. We highlight two insights from this analysis. From the perspective of application-level performances, the delay of the offloading service decreases quickly at the beginning, but maintains a certain level as the fog system scales (size $N$ increases). From the perspective of resource consumption, the demand for storage and processing resources (measured by the device effort) increases sublinearly, which can be satisfied by the simultaneous capacity boost as the fog system scales, while the demand for communication (measured by the network effort) increases linearly, leading to a possible network resource drain when the size of the system is large.

Authorized licensed use limited to: N.C. State University Libraries - Acquisitions & Discovery S. Downloaded on April 13,2026 at 07:22:58 UTC from IEEE Xplore. Restrictions apply.

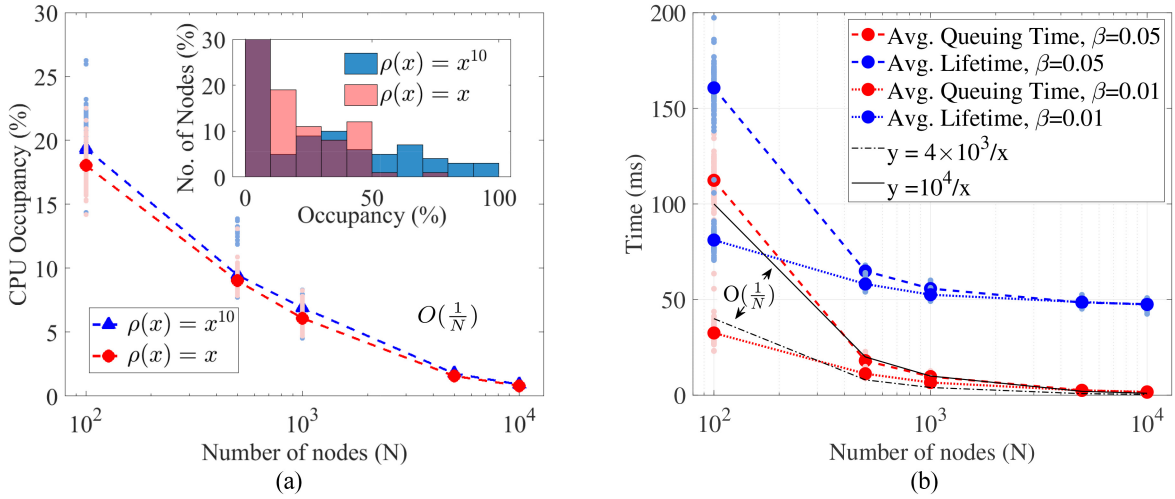



Fig. 4. Describing a delay-oriented offloading process with the gravity model: (a) CPU occupancy decreases with the system size $N$, and relaxing the selectiveness (from $\rho(x) = x^{10}$ to $\rho(x) = x$) can achieve load balancing among fog nodes and (b) both the queuing delay $\mathbb{E}(T^{\epsilon}_{Q|n})$ and task lifetime $\mathbb{E}(\theta_{\epsilon})$ of an individual task are proportionate to the task generating rate $\beta$, and exhibit $O(1/N)$ decreasing with respect to the network size.

TABLE IV
SIMULATION CONFIGURATION

| Parameter | Description | Value/Range |
|---|---|---|
| $N = \|E\|$ | network size | 100, 500, 1000, 5000, 10000 |
| $a$ | region edge | 1000 m |
| N/A | simulation run times | 100 |
| N/A | # of tasks | 1000 |
| $\beta$ | task generation rate | 0.01, 0.05, 0.5 |
| $speed_{max}$ | max moving speed | 10 m/s |
| $\psi$ | data rate attenuation | 20 Mbps/m |
| $R_{max}$ | max data rate | 10 Gbps |
| $L$ | data volume | 1Mb, 10 Mb, 100 Mb, 1 Gb |
| $\phi$ | processing intensity | 100, 200, 300, 400 cycles/bit |
| $\alpha$ | volume change coef. | 0.1, 0.5, 1 |
| $C$ | CPU frequency | 1 GHz (w.p. 0.4), 4GHz (w.p. 0.6) |
| $P_e$ | transmission power | [0.01, 0.25] W |
| $\xi$ | processing power | [1.3, 3.5]×$10^7$ J/cycle |

## V. NUMERICAL EVALUATION

The significance of the proposed gravity model is two-folded. As a versatile model, it can be flexibly configured to mimic the behavior of different offloading schemes, such that their performances can be evaluated through analysis or simulation to answer the "remedy or resource drain" question for massive offloading in large-scale systems. On the other hand, the gravity model itself can also be executed as a distributed offloading scheme on task nodes, whose performance can be compared with others to select the most appropriate scheme for IoT applications. To validate these two aspects, we simulate task offloading in fog systems of different sizes, with configurations listed in Table IV.

We consider a 1000 × 1000 $m^2$ region, where task/fog nodes move with a speed-constrained random walk of maximum speed 10 m/s. The maximum data rate $R_{\max}$ is set to be 10 Gb/s, according to the RAT configuration listed in Table II. We consider tasks with different data volumes, ranging from 1 Mb to 1 Gb, accounting for data collection, translation, video trans-coding, VR/AR applications. This uniform distribution with uneven dispersion is motivated by the existing IoT application traffic [29], future VR/AR application traffic [11], [34], also following conventions of related works [13], [15], [16], [17]. The number of task/fog nodes ranges from 100 to 10000, while the task generation rate $\beta$ varies from 0.01 to 0.5, to examine the performances as the system scales. Transmission power of task/fog nodes ranges from 0.01 to 0.25 W, taking different RATs in Table II into consideration. The processing power of fog nodes ranges from 1.3 to 3.5×$10^{-7}$ J/cycle, referencing [17]. For each configuration, the simulation is run for 100 times, each of which has randomly changing initial configurations (light-colored markers), to obtain the ensemble mean (bright-colored markers and lines).

### A. Describing Offloading Scheme With the Gravity Model

We first examine whether the gravity model can describe offloading schemes with different offloading criteria. We consider the delay-oriented offloading scenario, because delay is the major concern for many IoT applications.

First, two application-level performances of the delay-oriented gravity model, namely, CPU occupancy of fog nodes and the task lifetime, are presented in Fig. 4. From the perspective of fog nodes, the CPU occupancy in Fig. 4(a) quantifies the percentage of times slots in which the CPU of a fog node is busy processing tasks. The $O(1/N)$ decreasing occupancy over the number of nodes $N$ echoes with the analysis of offloading probability in Theorem 1 and the offloaded processing load $\mathbb{E}(Y_n)$ in Theorem 2. Meanwhile, relaxing the selectiveness function from $\rho(x) = x^{10}$ (blue dashed line with triangle markers) to $\rho(x) = x$ (red dashed line with round markers) successfully balance the processing load among fog nodes, as indicated by the lower average CPU occupancy as well as the number of high-occupancy nodes in the inner plot. When $\rho(x) = x$, task nodes are less reluctant to switch offloading targets, as a result of which, the processing load is divided more evenly among fog nodes (red bars), instead of overloading a small number of powerful fog nodes (blue bars that concentrating at the high occupancy end). From the perspective of the offloading service, the task queuing delay (red lines

Authorized licensed use limited to: N.C. State University Libraries - Acquisitions & Discovery S. Downloaded on April 13,2026 at 07:22:58 UTC from IEEE Xplore. Restrictions apply.

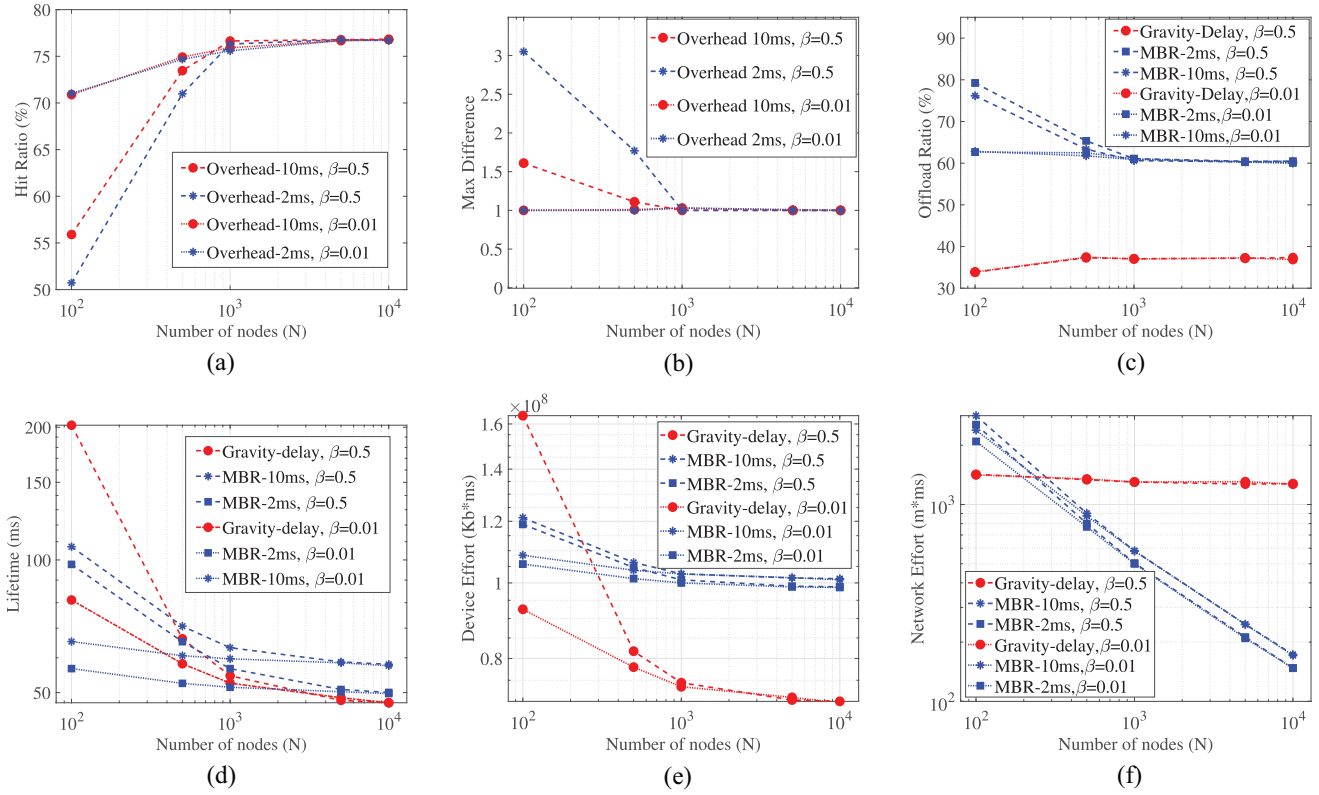



Fig. 5. Comparison between the delay-oriented gravity model (red, $\rho(x) = x$) and the MBR offloading scheme (blue): The chosen offloading targets under the two schemes are very similar, i.e., (a) in over 50% of the offloading procedures, the MBR target fog node also imposes the largest gravity, and when they differ, (b) maximum difference in ranking is at most 3. The difference in ranking the offloading candidates is mainly caused by (c) more frequent preference of local processing in the gravity model. However, the delay-oriented gravity model achieves (d) lower application delay and (e) lower storage and processing resource per task, but (f) more communication resource.

and markers), and task lifetime (blue lines and markers) in Fig. 4(b) are both $O(1/N)$, as stated by Theorem 2. These indicate the designed gravity model can describe a family of offloading schemes with different settings.

### B. Comparison With Existing Offloading Schemes

Next, to examine the detailed behavior of the gravity model, which is a fully distributed scheme that does require any coordination, we compare it with the cooperative MBR offloading [15] and matching theory-based efficient task offloading (METO) [17], to test offloading scenarios with single criterion (delay oriented) and multiple criteria (delay-energy-hybrid), respectively.

*1) Single Criterion:* The MBR scheme [15] is designed to minimize the task lifetime, so we compare it with the gravity model with delay-oriented gravity function in (8). Numerical results of this comparison are presented in Fig. 5. MBR is expected to achieve the global optimal (at the current time slot) with the necessary information of all fog nodes, such as the queue length on each candidate fog node, so an extra communication time (2 and 10 ms) is added in the simulation as overhead. We highlight three observations from the comparison.

*Overlapping Offloading Targets:* The target in MBR is in most cases the same candidate fog node that imposes the largest gravity force on the task, with a hit ratio higher than 70%, as shown in Fig. 5(a). When it is not the case, the MBR target still resides in the top 3 candidates obtained by the gravity model, as shown in Fig. 5(b). Particularly, as the system scales (larger $N$), the maximum difference between the two schemes becomes smaller.

*Similar but Better Delay Performance*: In both schemes, the decreasing trends of the task lifetime in Fig. 5(d), and hence the device effort in Fig. 5(e), are all $O(1/N)$ as the system scales, indicating that the gravity model accurately captures the performance change in the fog-based IoT system. Despite the same $O(1/N)$ decreasing trend, the gravity model, which is also a distributed offloading scheme, achieves a lower task lifetime than the MBR scheme in most scenarios. The reason behind this delay reduction is the lower queuing delay due to the automatic load balancing as a result of the probabilistic offloading, which is especially true when the system is large. In these cases, the gravity model allows a strictly positive probability to offloading a task to the "suboptimal" candidate to avoid burdening some fog nodes. Consequently, tasks may be offloaded to fog nodes that are farther away, which explains the higher average network effort in Fig. 5(f).

*Diverging Offloading Ratios and Network Efforts:* The main difference between the two offloading schemes lies in the offloading ratio, as shown in Fig. 5(c), which leads to the diverging network effort in Fig. 5(f). The gravity model tends to execute the task locally, due to its strict criterion for offloading, resulting a lower than 50% offloading ratio. However, this

Authorized licensed use limited to: N.C. State University Libraries - Acquisitions & Discovery S. Downloaded on April 13,2026 at 07:22:58 UTC from IEEE Xplore. Restrictions apply.

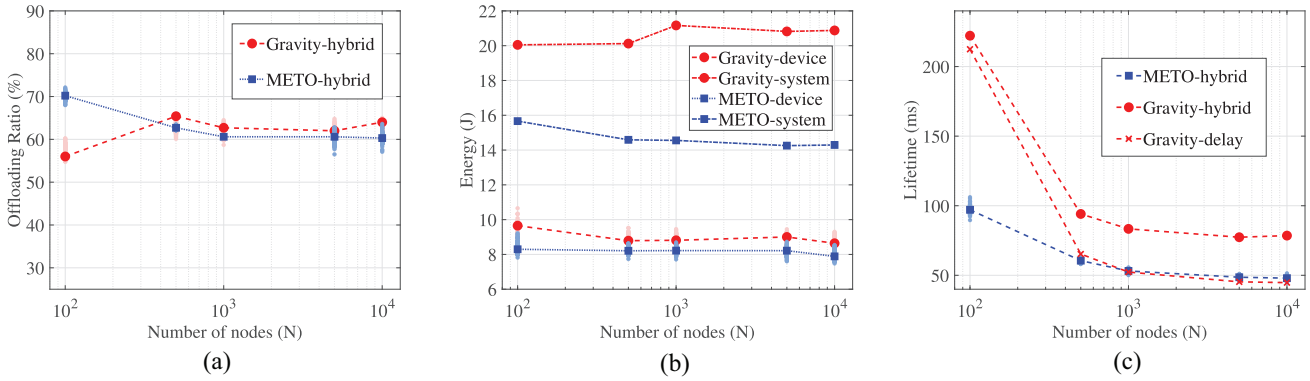



Fig. 6. Comparison between the gravity model (red, $\rho(x) = x^{10}$) and METO scheme[17] (blue) in a hybrid offloading scenario: (a) offloading ratio of METO is well captured by the gravity model, especially when the system ($N$) is large; (b) gravity model achieves similar local energy consumption, i.e., energy consumed by the task node, as METO, but consumes more energy in total; (c) scaling of the lifetime is captured, but METO achieves lower lifetime due to the MBR + TOPSIS enhancement.

ratio can actually be tuned by adjusting the threshold (zero in this case) of triggering an offloading, as will be discussed in the next section.

*2) Multiple Criteria:* The METO scheme [17] allows task node to employ multiple offloading criteria, but requires global information, including processing time and energy, to rank the candidate and perform matching. We compared it with the gravity model in a delay-energy-hybrid offloading scenario, where half of the tasks are subject to delay-oriented offloading and the other half to energy-oriented offloading. To better compare the two, we combine MBR and the TOPSIS-ranking algorithm in METO to enhance its performance, such that the queuing time is also taken into consideration for delay-oriented offloading criterion. Numerical results are shown in Fig. 6, by which we observe that the offloading ratio and energy consumption of task nodes are well captured by the gravity model. We also observe that the MBR-enhanced METO achieves a task lifetime as low as the delay-oriented offloading with the gravity model. This gap in total energy consumption and task lifetime can be attributed to the full information of candidate fog nodes (which incurs communication overheads of at least $O(N)$ rounds of control message exchanges that are not accounted for), and the TOPSIS-based ranking calculation, whose complexity is $O(N^2)$. Nonetheless, the scaling behavior of energy consumption and task lifetime over the system size $N$ are still captured by the gravity model.

Results in Fig. 4–6 indicate that the proposed gravity model can effectively describe an offloading process, while the designed forms of gravity and selectiveness allow flexible adjustments in offloading preferences. Moreover, it also enables us to theoretically analyze the performance of offloading service in the fog-based IoT, as the analytical results in Section III and Section IV apply to systems of any scale ($N$).

### C. Gravity Model as Offloading Scheme

Considering its advantages of being fully distributed, and easy for scalability analysis, the gravity model itself can also be implemented as an offloading scheme. To further examine its performances in different scenarios, more simulations are conducted under different task generation rates ($\beta$) and selectiveness functions ($\rho(x)$), as illustrated in Fig. 7.

We highlight the following observations from these results.

*1) Theorems 2 and 3 Hold for All Cases:* The scaling laws over the network size $N$ derived in Theorems 2 and 3, i.e., $O(1/N)$ task lifetime, $O(1/N)$ device effort, and $O(1)$ network effort, are captured by the red lines in Fig. 7.

*2) Rate $\beta$ Does Not Change the Scaling Behavior:* Theorems 2 and 3 hold for all task generation rates ($\beta$). The higher the processing load (larger $\beta$), the longer the application-level delay, and the larger device effort. Due to its stochastic nature, however, the amount of communication resource spent on a single task does not change over the workload, as indicated by the overlapping red/blue lines in Fig. 7(c). This behavior differs from the MBR scheme, but is especially useful in network planning due to its insensitivity to processing load.

*3) Selectiveness $\rho(x)$ Greatly Affects the Network Effort:* Theorem 2 holds for all $\rho(x)$ in polynomial forms, which is well captured by both the red and blue lines in Fig. 7(a). Consequently, device effort also follows the $O(1/N)$ decreasing, as shown in Fig. 7(b). The network effort, nonetheless, is greatly affected by $\rho(x)$, changed from $O(1)$ to $O(1/N)$ as indicated by the blue lines in Fig. 7(c), alleviating the possibility of resource drain when the system scales. Moreover, comparing it with Fig. 5, we see that tightening selectiveness in the gravity model to $\rho(x) = x^{10}$ can capture the decreasing trend of network effort of the MBR scheme and the METO scheme, implying the flexibility and descriptiveness of the gravity model.

From the perspective of performance, these simulation results indicate that, task offloading in the fog system is a remedy to delay-sensitive applications for IoT, as task lifetime decreases quickly with the size of the network. Nonetheless, it is also a possible resource drain to the edge network, in the sense that the linearly growing consumption of total communication resource, i.e., the sum of network effort during a certain time period, may not be sustained in this course, considering the limited network capacity of the edge network. While resource drain at the edge may be attributed to many possible reasons, including spectrum misuse, network overlay, and so on, uncontrolled, massive, and frequent task offloadings introduced by an open fog system clearly adds

Authorized licensed use limited to: N.C. State University Libraries - Acquisitions & Discovery S. Downloaded on April 13,2026 at 07:22:58 UTC from IEEE Xplore. Restrictions apply.

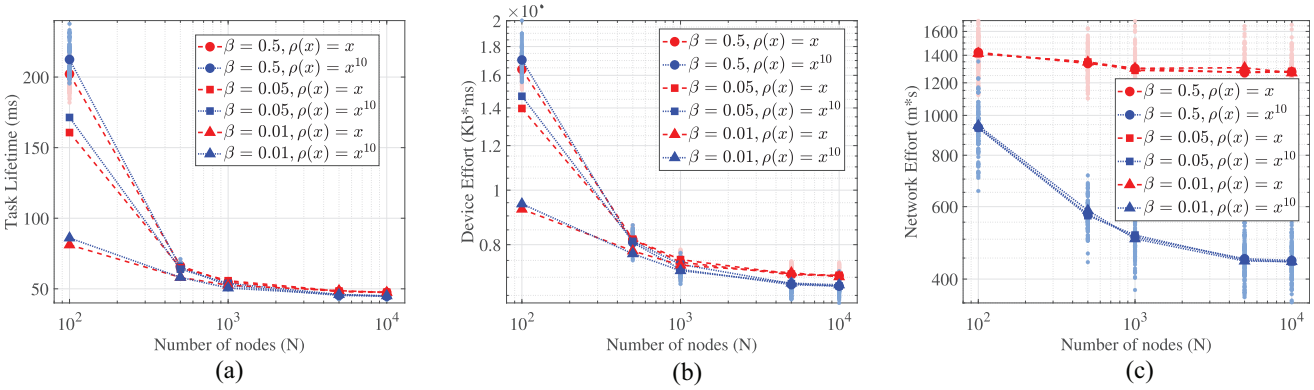



Fig. 7. Under the gravity offloading model, the task lifetime and device effort scales as $O(1/N)$, while the network effort scales as $O(1)$. The form of the selectiveness function $\rho(x)$ has a great impact on all three performance metrics, especially when the load is high ($\beta = 0.5$ dashed line with round markers). (a) Lifetime. (b) Device Effort. (c) Network Effort.

to the risk, especially considering the explosive growth of IoT devices.

## VI. Conclusion

In this article, we study the performance evaluation problem of task offloading processes in fog-based IoT systems. To address challenges incurred by the diversity of offloading schemes, we define new metrics, and propose a gravity-based offloading model, which can describe various offloading criteria. Our analysis and simulation results show that, while the QoS of offloading service improves as the system grows larger, the amount of share network resource consumed by the offloading process scales linearly, indicating a possible resource drain at the edge. Our findings extend the knowledge on fog computing, particularly on its scalability, which are important steps before prototype design, experiments, or real-world system implementation. More importantly, the proposed metrics, gravity model, and analysis framework shed lights on modeling of heterogeneous wireless systems through the lens of data, contributing to the design, planning, and deployment of such large-scale paradigms.

## References

[1] "Cisco visual networking index: Forecast and trends, 2017–2022," Cisco, San Jose, CA, USA, White Paper, Feb. 2019.

[2] L. M. Vaquero and L. Rodero-Merino, "Finding your way in the fog: Towards a comprehensive definition of fog computing," *SIGCOMM Comput. Commun. Rev.*, vol. 44, no. 5, pp. 27–32, Oct. 2014. [Online]. Available: http://doi.acm.org/10.1145/2677046.2677052

[3] Y. Ai, M. Peng, and K. Zhang, "Edge computing technologies for Internet of Things: A primer," *Digit. Commun. Netw.*, vol. 4, no. 2, pp. 77–86, 2018. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S2352864817301335

[4] S. Yi, C. Li, and Q. Li, "A survey of fog computing: Concepts, applications and issues," in *Proc. Workshop Mobile Big Data*, 2015, pp. 37–42. [Online]. Available: http://doi.acm.org/10.1145/2757384.2757397

[5] J. He, J. Wei, K. Chen, Z. Tang, Y. Zhou, and Y. Zhang, "Multitier fog computing with large-scale IoT data analytics for smart cities," *IEEE Internet Things J.*, vol. 5, no. 2, pp. 677–686, Apr. 2018.

[6] A. Yousefpour et al., "All one needs to know about fog computing and related edge computing paradigms: A complete survey," *J. Syst. Archit.*, vol. 98, pp. 289–330, Sep. 2019. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1383762118306349

[7] F. Chiti, R. Fantacci, and B. Picano, "A matching theory framework for tasks offloading in fog computing for IoT systems," *IEEE Internet Things J.*, vol. 5, no. 6, pp. 5089–5096, Dec. 2018.

[8] L. Tong, Y. Li, and W. Gao, "A hierarchical edge cloud architecture for mobile computing," in *Proc. 35th Annu. IEEE Int. Conf. Comput. Commun.*, Apr. 2016, pp. 1–9.

[9] *IEEE Standard for Adoption of OpenFog Reference Architecture for Fog Computing*, IEEE Standard 1934-2018, Aug. 2018.

[10] C. Perera, C. H. Liu, and S. Jayawardena, "The emerging Internet of Things marketplace from an industrial perspective: A survey," *IEEE Trans. Emerg. Topics Comput.*, vol. 3, no. 4, pp. 585–598, Dec. 2015.

[11] "VR and AR pushing connectivity limits," Qualcomm, San Diego, CA, USA, Rep., Oct. 2018. [Online]. Available: https://www.qualcomm.com/content/dam/qcomm-martech/dm-assets/documents/presentation_-_vr_and_ar_are_pushing_connectivity_limits_-web_0.pdf

[12] Y. Mao, C. You, J. Zhang, K. Huang, and K. B. Letaief, "A survey on mobile edge computing: The communication perspective," *IEEE Commun. Surveys Tuts.*, vol. 19, no. 4, pp. 2322–2358, 4th Quart., 2017.

[13] M. Chen and Y. Hao, "Task offloading for mobile edge computing in software defined ultra-dense network," *IEEE J. Sel. Areas Commun.*, vol. 36, no. 3, pp. 587–597, Mar. 2018.

[14] J.-Y. Baek, G. Kaddoum, S. Garg, K. Kaur, and V. Gravel, "Managing fog networks using reinforcement learning based load balancing algorithm," in *Proc. IEEE Wireless Commun. Netw. Conf. (WCNC)*, Apr. 2019, pp. 1–7.

[15] S. Josilo and G. Dán, "Decentralized algorithm for randomized task allocation in fog computing systems," *IEEE/ACM Trans. Netw.*, vol. 27, no. 1, pp. 85–97, Feb. 2019.

[16] J. Shi, J. Du, J. Wang, J. Wang, and J. Yuan, "Priority-aware task offloading in vehicular fog computing based on deep reinforcement learning," *IEEE Trans. Veh. Technol.*, vol. 69, no. 12, pp. 16067–16081, Dec. 2020.

[17] C. Swain et al., "METO: Matching-theory-based efficient task offloading in IoT-fog interconnection networks," *IEEE Internet Things J.*, vol. 8, no. 16, pp. 12705–12715, Aug. 2021.

[18] J. Baek and G. Kaddoum, "Online partial offloading and task scheduling in SDN-fog networks with deep recurrent reinforcement learning," *IEEE Internet Things J.*, vol. 9, no. 13, pp. 11578–11589, Jul. 2022.

[19] J. Baek and G. Kaddoum, "Heterogeneous task offloading and resource allocations via deep recurrent reinforcement learning in partial observable multifog networks," *IEEE Internet Things J.*, vol. 8, no. 2, pp. 1041–1056, Jan. 2021.

[20] J. Yao and N. Ansari, "Task allocation in fog-aided mobile IoT by Lyapunov online reinforcement learning," *IEEE Trans. Green Commun. Netw.*, vol. 4, no. 2, pp. 556–565, Jun. 2020.

[21] Q. Fan, J. Bai, H. Zhang, Y. Yi, and L. Liu, "Delay-aware resource allocation in fog-assisted IoT networks through reinforcement learning," *IEEE Internet Things J.*, vol. 9, no. 7, pp. 5189–5199, Apr. 2022.

[22] H. Zhang, Y. Xiao, S. Bu, D. Niyato, F. R. Yu, and Z. Han, "Computing resource allocation in three-tier IoT fog networks: A joint optimization approach combining Stackelberg game and matching," *IEEE Internet Things J.*, vol. 4, no. 5, pp. 1204–1215, Oct. 2017.

[23] J. Yao and N. Ansari, "Reliability-aware fog resource provisioning for deadline-driven IoT services," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, Dec. 2018, pp. 1–6.

[24] L. Pu, X. Chen, J. Xu, and X. Fu, "D2D fogging: An energy-efficient and incentive-aware task offloading framework via network-assisted D2D collaboration," *IEEE J. Sel. Areas Commun.*, vol. 34, no. 12, pp. 3887–3901, Dec. 2016.

Authorized licensed use limited to: N.C. State University Libraries - Acquisitions & Discovery S. Downloaded on April 13,2026 at 07:22:58 UTC from IEEE Xplore. Restrictions apply.

[25] Y. Yang, X. Chang, Z. Han, and L. Li, "Delay-aware secure computation offloading mechanism in a fog-cloud framework," in *Proc. IEEE Int. Conf. Parallel Distrib. Process. Appl., Ubiquitous Comput. Commun., Big Data Cloud Comput., Social Comput. Netw., Sustain. Comput. Commun. (ISPA/IUCC/BDCloud/SocialCom/SustainCom)*, Dec. 2018, pp. 346–353.
[26] Y. Xiao and M. Krunz, "QoE and power efficiency tradeoff for fog computing networks with fog node cooperation," in *Proc. IEEE Conf. Comput. Commun.*, May 2017, pp. 1–9.
[27] L. Tang and S. He, "Multi-user computation offloading in mobile edge computing: A behavioral perspective," *IEEE Netw.*, vol. 32, no. 1, pp. 48–53, Jan./Feb. 2018.
[28] H. Wu, "Performance modeling of delayed offloading in mobile wireless environments with failures," *IEEE Commun. Lett.*, vol. 22, no. 11, pp. 2334–2337, Nov. 2018.
[29] Y. Wan, K. Xu, F. Wang, and G. Xue, "Characterizing and mining traffic patterns of IoT devices in edge networks," *IEEE Trans. Netw. Sci. Eng.*, vol. 8, no. 1, pp. 89–101, Jan.–Mar. 2021.
[30] Z. Liu, X. Yang, Y. Yang, K. Wang, and G. Mao, "DATS: Dispersive stable task scheduling in heterogeneous fog networks," *IEEE Internet Things J.*, vol. 6, no. 2, pp. 3423–3436, Apr. 2019.
[31] R. Aparicio-Pardo, K. Pires, A. Blanc, and G. Simon, "Transcoding live adaptive video streams at a massive scale in the cloud," in *Proc. 6th ACM Multimedia Syst. Conf.*, 2015, pp. 49–60. [Online]. Available: http://doi.acm.org/10.1145/2713168.2713177
[32] A. Lottarini et al., "Vbench: Benchmarking video transcoding in the cloud," *ACM SIGPLAN Notices*, vol. 53, no. 2, pp. 797–809, 2018.
[33] Y. Nabil, H. ElSawy, S. Al-Dharrab, H. Mostafa, and H. Attia, "Data aggregation in regular large-scale IoT networks: Granularity, reliability, and delay tradeoffs," *IEEE Internet Things J.*, vol. 9, no. 18, pp. 17767–17784, Sep. 2022.
[34] Y. Al Mtawa, A. Haque, and B. Bitar, "The mammoth Internet: Are we ready?" *IEEE Access*, vol. 7, pp. 132894–132908, 2019.
[35] Y. Huo, X. Dong, and W. Xu, "5G cellular user equipment: From theory to practical hardware design," *IEEE Access*, pp. 13992–14010, 2017.
[36] R. M. Borges, L. A. M. Pereira, A. C. Ferreira, L. L. Mendes, D. H. Spadoti, and A. C. Sodre, "Fifth-generation new radio fiber-wireless system for long-reach and enhanced mobile broadband scenarios," *Microw. Opt. Technol. Lett.*, vol. 63, no. 2, pp. 662–669, 2021.
[37] I. Fogg, "The state of Wifi vs mobile network experience as 5G arrives," OpenSignal, Boston, MA, USA, Rep., Nov. 2018. [Online]. Available: https://cdn.opensignal.com/public/data/reports/global/data-2018-11/state_of_wifi_vs_mobile_opensignal_201811.pdf
[38] J. Philip, *The Probability Distribution of the Distance Between Two Random Points in a Box*, KTH Mathematics, KTH Royal Inst. Technol., Stockholm, Sweden, 2007.



**Jie Wang** (Member, IEEE) received the Ph.D. degree in computer engineering from the Department of Electrical and Computer Engineering, North Carolina State University, Raleigh, NC, USA, in 2019.

She is currently an Assistant Professor with the School of Software Engineering, Tongji University, Shanghai, China. Her research interests include modeling and performance analysis of networked systems (including wireless networks, IoT, and social networks), edge computing, and mobile bid data.



**Wenye Wang** (Fellow, IEEE) received the M.S.E.E. and Ph.D. degrees in computer engineering from Georgia Institute of Technology, Atlanta, GA, USA, in 1999 and 2002, respectively.

She is a Professor with the Department of Electrical and Computer Engineering, North Carolina State University, Raleigh, NC, USA. Her research interests include mobile and secure computing, modeling and analysis of wireless networks, network topology, and architecture design.

Dr. Wang was the recipient of the NSF CAREER Award 2006. She was a co-recipient of the 2006 IEEE GLOBECOM Best Student Paper Award—Communication Networks and the 2004 IEEE Conference on Computer Communications and Networks Best Student Paper Award. She has been a member of the Association for Computing Machinery since 1998 and a member of the Eta Kappa Nu and Gamma Beta Phi honorary societies since 2001.



**Cliff Wang** received the Ph.D. degree in computer engineering from North Carolina State University, Raleigh, NC, USA, in 1996.

He currently serves as the Computing Sciences Division Chief with the Army Research Office, Durham, NC, USA. He is also appointed as an Adjunct Faculty Member of Computer Science with the College of Engineering, North Carolina State University. He has been carrying out research in the area of computer vision, medical imaging, high-speed networks, and most recently information security.

Authorized licensed use limited to: N.C. State University Libraries - Acquisitions & Discovery S. Downloaded on April 13,2026 at 07:22:58 UTC from IEEE Xplore. Restrictions apply.