Receiver-Initiated Data Collection in Wake-up Radio Enabled mIoT Networks: Achieving Collision-Free Transmissions by Hashing and Partitioning

Chia-An Hsu, Chung-Hsiang Tsai, Frank Y. Li, Chiuyuan Chen, and Yu-Chee Tseng

***Abstract*—To achieve ultra-low energy consumption and decade-long battery lifetime for Internet of Things (IoT) networks, wake-up radio (WuR) appears as an eminent solution. While keeping devices in deep sleep for most of the time, a WuR enabled IoT device can be woken up for data transmission at any time by a wake-up call (WuC). However, collisions happen among WuCs for transmitter-initiated *data reporting* and among data packets for receiver-initiated *data collection*. In this paper, we propose three novel hashing-based schemes in order to achieve collision-free data transmissions for receiver-initiated data collection. We consider first a simple scenario where all devices in a region of interest are reachable by a WuC message and propose a scheme which facilitates a scheduled time instant for data uploading of each device through a hash function. In the second scenario where IoT devices are distributed across a large region that cannot be covered by a single WuC, we propose two partitioning algorithms to enable data collection across multiple partitions. Furthermore, we extend the scenario by considering device mobility and propose another scheme which improves the partitioning algorithm to deal with mobility. Both analysis and simulations are performed to demonstrate the effectiveness of the proposed schemes.**

***Index Terms*—Massive IoT, wake-up radio, receiver-initiated data collection, collision-free, hash and partition.**

I. Introduction

To connect a huge number of Internet of Things (IoT) devices, known as massive IoT (mIoT) [2] [3], two categories of connections are most appealing, i.e., cellular network based connections such as long term evolution for machine-type communication (LTE-M) [4] and narrowband IoT (NB-IoT) [5]; and non-cellular based connections such as long range wide area network (LoRaWAN) [6] and Sigfox [7] [8]. No matter which solution is adopted, energy consumption always plays a pivotal role for IoT network design since these devices are typically battery powered and are often deployed at places that are not easily accessible.

Traditionally, data collection in IoT networks adopts duty-cycling (DC) mechanisms which can significantly save energy consumption, however, at a cost of long delay. Recently, a paradigm shift from DC medium access control (MAC) to wake-up radio (WuR) enabled medium access is undergoing for both cellular and non-cellular networks, thanks to the overwhelming superiority for energy consumption provided by WuR [9]–[14]. Different from the DC principle, the novelty of WuR operation is to associate the main radio (MR) of an IoT device with a wake-up receiver (WuRx) which consumes power at the magnitude of 1000 times less than that of the MR. While the MR is in *deep sleep* for most of the time and wakes up for data transmission in an on-demand manner, the WuRx is always on listening to any activities over the channel.

To initiate data transmission in WuR enabled IoT networks, there are two operating modes, i.e., transmitter-initiated (TI) *data reporting* and receiver-initiated (RI) *data collection*. In both cases, the activity of the MR of an IoT device will be triggered by a wake-up call (WuC) received by its associated WuRx. A WuC is a specifically designed message dedicated to wake a target device up for receiving or transmitting a data frame when an event occurs. When multiple devices initiate their WuCs at the same time (in the TI mode) or a broadcast/multicast WuC is received by multiple devices (in the RI mode), collision(s) may happen.

A. Related Work

Intensive studies on data transmission and protocol design in WuR enabled non-cellular IoT networks have been performed in recent years [15] [16]. Another very recent paper provides a comprehensive survey and taxonomy on MAC, routing, and cross layer protocols for WuR enabled wireless networks [17]. In general, the most recent progress in the state-of-the-art research on WuR is the trend of driving WuR development from academic and lab efforts towards real-life systems. This trends is represented by the recent standardization activities to enable WuR in the two most eminent wireless systems, i.e., 5^{th} generation (5G) cellular networks and wireless fidelity (WiFi) [18] [13]. For example, an effective access mechanism enabled through WuR can achieve significant power saving for 5G mobile devices [19]. WuR can also significantly reduce idle listening energy cost in both LTE and WiFi systems [20]. Other applications like unmanned aerial vehicle (UAV) based systems for data collection are also appealing, including the work presented in this paper.

In [25] and [22], four schemes for WuC collision avoidance have been proposed for TI- or RI-based networks respectively. Through clear channel assessment (CCA) and backoff, much lower collision probabilities have been achieved. However,

C.-A. Hsu, C.-H. Tsai, and C.Y. Chen are with the Department of Applied Mathematics, National Chiao Tung University, Hsinchu, Taiwan, R.O.C. (email: ab89414@gmail.com; alan85107@gmail.com; cychen@mail.nctu.edu.tw).

F. Y. Li is with the Department of Information and Communication Technology, University of Agder (UiA), N-4898 Grimstad, Norway (email: frank.li@uia.no).

Y.-C. Tseng is with the Department of Computer Science, National Chiao Tung University, Hsinchu, Taiwan, R.O.C. (email: yuchee@cs.nctu.edu.tw).

TABLE I: State-of-the-art WuR MAC protocols for *receiver-initiated* transmissions: A qualitative comparison.

Protocol versus Features	AWR-PS-MAC [21]	RI-CPT-WuR [22]	RS-CPR [23]	RI-LD-WuR [24]	Scheme 1 [ours]	Scheme 2 [ours]	Scheme 3 [ours]
Network scenario	Small-scale	Small-scale	Multi-hop	Small-scale but multiple groups	Small-scale	Large-scale	Large-scale
Network partitioning	No	No	No	Yes (grouping)	No	Yes (clustering)	Yes (clustering)
Device mobility	No	No	No	No	No	No	Yes
Collision avoidance mechanism	Polling via beacon messages	Backoff + medium reservation	Relay-node selection + medium reservation	Random access + grouping	Hashing + random access	Hashing + random access + partitioning	Hashing + random access + partitioning
Tools for mathematical analysis	Probability and algebra	DTMC	Probability and algebra	DTMC	Probability and graph theory	Probability and graph theory	Probability and graph theory
Simulator	OMNeT++ and MiXiM	SMPL	Unknown	ns-2.35	MATLAB and Python	MATLAB and Python	MATLAB and Python
Real-life testbed	No	No	No	No	No	No	No

‡Abbreviations used in this table: Discrete-time Markov chain (DTMC); objective modular network testbed in C++ (OMNeT++); mixed simulator (MiXiM); simulation and modeling programming language (SMPL); network simulator (ns). For protocol acronym, refer to each reference paper for details.

contention among IoT devices still exists during the transmission procedure, especially when the number of devices is large. To avoid collision for data collection (or uploading) in RI-based networks, which is the focus of this study, one may consider to use a unicast WuC to poll each IoT device consecutively. However, such a procedure leads to long delay since WuCs are transmitted at a much lower data rate compared with the date rate for packet transmissions [9] [11]. On the other hand, a contention-based uploading mechanism may also cause long delay due to backoff and comparatively high energy consumption due to CCA. In Table I above, we provide a qualitative comparison of our schemes versus four WuR protocols which are dedicated to data collections in IoT or wireless sensor networks operated *based on the RI mode*.

B. Contributions

In this paper, we propose three energy-efficient RI-based data collection schemes targeting at collision-free transmissions for WuR enabled non-cellular IoT networks in which a data collector could be a fixed station or a UAV. For each round of data collection, a dedicated WuC is multicast by the UAV to a group of IoT devices that are reachable by one single WuC message. The specially designed WuC contains the information for multiple devices through which the time slot assigned to each involved device can be deduced. To realize this idea, we leverage a hash function through which each device can derive its dedicated time slot for transmission. Furthermore, we extend our scenario from a small-scale to large-scale network in which the coverage of the UAV cannot reach all devices through one WuC and devices inside the region may be either static or mobile. In these cases, the data collector needs to partition the devices into multiple clusters and perform multiple rounds of data collection, each for one cluster. Then we propose two spanning forest based heuristic algorithms which are able to obtain an optimal partition for grouping devices into an appropriate number of clusters. For the scenario with mobility, the partitioning algorithm is further extended for dealing with device mobility with two a priori known patterns. Via both analysis and simulations, we demonstrate that our schemes can effectively avoid collisions, shorten transmission times, and reduce device energy consumption.

Briefly, the novelty and main contributions of this work are summarized as follows.

- Three transmission schemes for RI-based data collection in WuR enabled IoT networks are proposed, targeting at small- and large-scale networks with and without device mobility respectively. The schemes facilitate collision-free data transmissions based on a hash function which generates distinct transmission times for devices in a group triggered by a common WuC.
- For large-scale networks where devices are spread across a region that is much larger than the coverage of a WuC, two spanning forest based algorithms with four different flavors of cluster partitioning are proposed.
- For networks with device mobility, two mobility patterns are considered and the scheme for large-scale networks is extended to address clustering when devices are mobile.
- The proposed schemes and algorithms are implemented and their performance is assessed through extensive simulations under various network configurations.

To the best of our knowledge, this is the first work which intends to provide collision-free data transmissions through hashing and network partitioning for receiver-initiated data collection considering both small- and large-scale WuR enabled IoT networks.

The rest of this paper is organized as follows. After explaining the network scenarios in Sec. II, we present the proposed schemes in Sec. III. The performance of the schemes is analyzed in Sec. IV, followed by simulation results presented in Sec. V. Finally, the paper is concluded in Sec. VI.

II. Network Scenarios and Assumptions

Three network scenarios for RI-based data collection are envisaged, as illustrated in Figs. 1∼3 respectively. Throughout the context, a *region* indicates an area inside which a number of WuR enabled IoT devices are deployed. Among these deployed devices, a set of them that can be reached by one

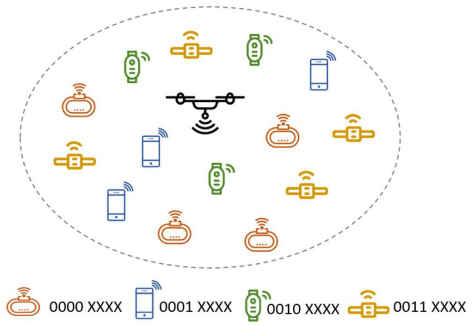


Fig. 1: Scenario 1: All devices are covered by a UAV but multiple groups exist. 4 groups, each with its pre-configured netmask, are illustrated as Red($\mathcal{S}_1$), Blue($\mathcal{S}_2$), Green($\mathcal{S}_3$), and Yellow($\mathcal{S}_4$) respectively.

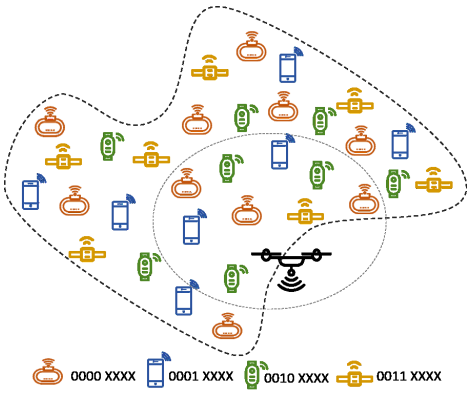
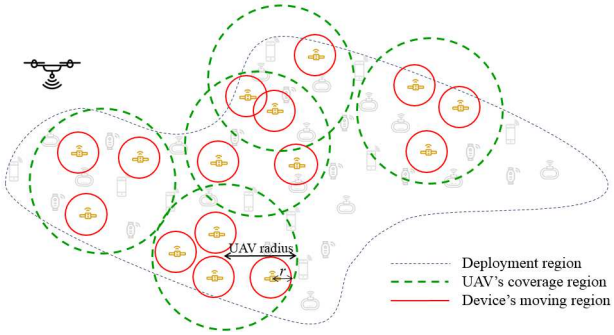
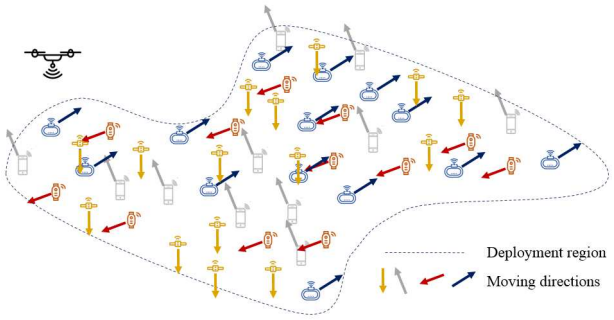


Fig. 2: Scenario 2: Devices are distributed across a much larger region and network partition into clusters is expected for data collection.



(a) Scenario 3A (micro mobility): Each device can only move within the vicinity of its initial deployment.



(b) Scenario 3B (group mobility): Devices in the same group move in the same direction at the same speed.

Fig. 3: Scenario 3: Large-scale networks with two types of device mobility patterns.

WuC form a *cluster* and how a region is divided into multiple clusters will be decided by a partitioning algorithm to be presented in Sec. III. Inside each cluster, there may be multiple groups of devices and each *group* has a common netmask. Although all devices covered by a cluster are reachable by a single WuC, only the group which has its netmask matching the received WuC group address will upload their packet to the data collector. Moreover, each device has only one packet to upload for each *round* of data collection where one round of data collection is meant for a group of devices in a cluster. A data collection *mission* consists of multiple rounds to collect data from devices in all groups and clusters in the region of interest. Furthermore, we assume that both uplink and downlink channels are error-free and propagation delay is negligible. Nor is the altitude of the flying UAV on a data collection mission considered in this study.

Scenario 1 (small-scale networks): Consider a cluster of devices, denoted by $\mathcal{S}$, that can be covered by one multicast WuC message, e.g., when a UAV hovers for a moment to collect data from a garden. Although located within the same cluster, the type of data to be collected is heterogeneous. Accordingly, *the devices in the same cluster are categorized into multiple groups* by their functions (e.g., weather monitoring and asset tracking devices belong to two different groups) such that $\mathcal{S} = \bigcup_{i=1}^{n} \mathcal{S}_i$, where each $\mathcal{S}_i$ is a group and $\mathcal{S}_i \cap \mathcal{S}_j = \emptyset$ if $i \neq j$. Furthermore, the addresses of the devices in the same group are configured through one common subnet mask.

When a WuC is sent from the UAV, all devices will receive this WuC. Accordingly, an energy-efficient data collection procedure needs to be designed to wake the targeted group of devices up and for uploading their data such that the total energy consumption will be minimized. The total energy consumption is defined as the sum of the energy consumption by all devices and the procedure will be presented later. When multiple groups exist in one cluster, several rounds of data collection need to be performed each for one group.

Scenario 2 (large-scale networks with static devices): Consider a more realistic scenario where the deployment region is much larger than the coverage of a WuC. In this case, the UAV is not able to collect data from the same type of devices by sending just a WuC to one cluster since the same type of devices may be distributed across the region and thus will be covered by more than one cluster.

In other words, the UAV has to initiate multiple WuCs for multiple rounds of data collections in one mission. To do so, a partitioning algorithm needs to be designed in order to find an optimal partition of the region that minimizes total energy consumption. For data collection in each cluster, the same procedure for Scenario 1 applies.

Scenario 3 (large-scale networks with device mobility): Consider another more complicated situation where device mobility is foreseen. Such scenarios could occur for instance when multiple devices jointly perform scanning tasks in a lawn, search for specific materials/items in an open field, or monitor environmental parameters in a lake or along a bayou.

For the sake of cluster partitioning and device traceability, we assume that devices may move within the region following

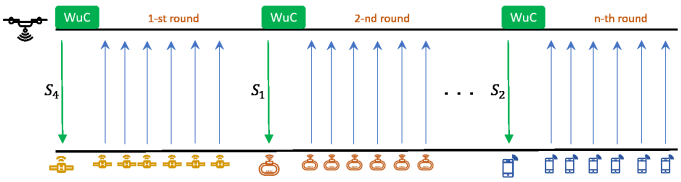


Fig. 4: A mission with n rounds of data collection.

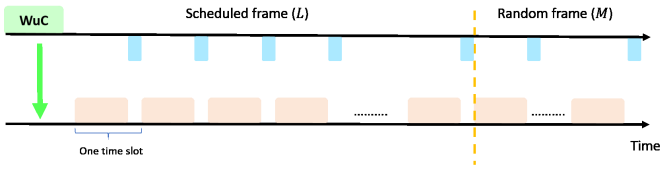


Fig. 5: Two frames exist in one round of data collection.

certain rules known to the UAV beforehand. In particular, two patterns of moving behaviors, referred to as *micro mobility* and *group mobility* respectively, are considered.

- **Scenario 3A (micro mobility):** Each device can only move in the vicinity of its initial deployment within a disc with radius r, where $r \ll$ UAV coverage radius, as shown in Fig. 3a).
- **Scenario 3B (group mobility):** Devices in the same group move towards the same direction at the same speed. However, different groups of devices may travel towards distinct directions, as shown in Fig. 3b).

More detailed discussions on these mobility patterns will be presented in Subsec. III-C. Note anyhow that a scenario which allows all devices to arbitrarily move in the whole deployment region is beyond the scope of this paper.

III. Proposed RI-based Data Collection Schemes

The main idea of our schemes is to schedule the transmissions of IoT devices in a data collection round at different time instants through a common WuC initiated by the data collector. To do so, each device obtains its time slot by computing a simple hash function based on a common seed multicast by the data collector via a WuC. Correspondingly, the collector avoids collisions among transmissions of end devices at a cost of slightly higher overhead when computing the seed. A list of notations used in this paper can be found in Appendix A.

A. Scheme 1: Small-Scale Networks

Considering Scenario 1 where only one cluster exits, Fig. 4 illustrates the operation of a mission consisting of n rounds of data collections initiated by a UAV. One round, or a cycle, represents the data collection procedure for a group of devices initiated through one WuC.

Within one cycle we divide the whole duration into multiple time slots. The duration of the cycle as well as the length of each time slot are configurable. As illustrated in Fig. 5, each cycle is composed of two frames: a *scheduled frame* (SF) and a *random frame* (RF). The numbers of time slots needed in each SF and RF, denoted by L and M respectively, depend on device population as well as performance requirements and these numbers are pre-configured. The length of each time slot is the sum of time needed for transmitting a data packet and an acknowledgment (ACK) or a negative ACK (NACK) message.

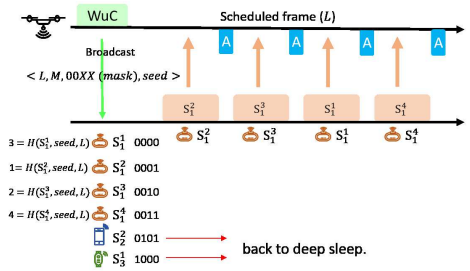


Fig. 6: Use hashing to assign time slot to each device.

1) Protocol design: In Scenario 1, the devices in a cluster are classified into mutually exclusive groups with different subnet masks. For data transmission in a specific group, each time slot in the SF is uniquely assigned to one device by a hash function so that it can upload its data without contention.

To facilitate collision-free transmissions, we embed a common hash function among the UAV and devices. The hash function takes an ID, a common seed, and a frame length of L as the input and computes an integer value between $[1, L]$. To perform the hash function, each device uses its own ID as an input. The seed is decided and sent by the collector via a WuC. An example of such an operation is shown in Fig. 6.

However, finding an ideal seed for the hash function, i.e., a seed that does not produce any collision, is not an easy process [26], especially when the device population is large. To address this potential problem, we introduce another frame, known as an RF, and append it immediately after the SF. The purpose of introducing an RF is to resolve a collision that occurred during the SF through random access.

To collect data from a specific group, $\mathcal{S}_i$, the proposed protocol works as presented below.

- Step 1: Prior to a data collection mission, the collector selects a seed for the hash function and decides the frame lengths of L and M, as well as the mask of $\mathcal{S}_i$. It then initiates a cycle by broadcasting a WuC to all devices.
- Step 2: Upon receiving the broadcast WuC, each device in $\mathcal{S}$ wakes partially up from the *deep sleep* mode to the *light sleep* mode and checks whether it belongs to this group by comparing its address with the netmask. Since each group is covered by a distinct netmask, only the devices in $\mathcal{S}_i$ will go to *Step 3*. Other devices will go back to the *deep sleep* mode. Furthermore, since the length of a netmask is much shorter than the length of a full address, those devices that are not in $\mathcal{S}_i$ will terminate decoding and go back to the *deep sleep* mode in a much early stage [27].
- Step 3: Each device in $\mathcal{S}_i$ then computes the hash function $\mathcal{H}(ID_k, seed, L)$ based on its own ID, L, and the seed. The retrieved value from $\mathcal{H}(ID_k, seed, L)$ is the index of a time slot that is assigned to this device in the SF which contains the first L time slots in the frame structure.
- Step 4: After computing the time slot index, the device stays in the *light sleep* mode until it transmits at time slot $\mathcal{H}(ID_k, seed, L)$. Then it waits for an ACK.
- Step 5: If an ACK is received, the device goes back to the *deep sleep* mode. Otherwise, it stays in the *light sleep*

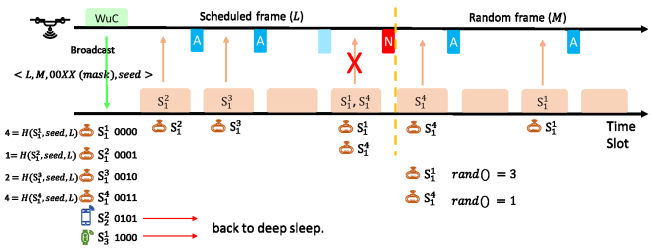


Fig. 7: An example illustrating how Scheme 1 works for a small-scale network with 6 devices among which 4 belong to the same group.

mode. Note that a NACK might be caused by a hashing collision or erroneous decoding.

- Step 6: Any device which failed in the SF will randomly select a time slot in the subsequent RF to upload its data again. Then, it goes back to *deep sleep*.

In Fig. 7, we give a simple example to illustrate the operation by considering a group of devices, $\mathcal{S}_1$, that has a netmask as 00. Among the six devices shown in the figure, two of them which have a netmask as 01 or 10 respectively will go to deep sleep right after decoding the netmak. The other four devices will continue to retrieve the time slot allocated to them. A collision happens since two devices retrieve a common time slot in the scheduled frame, but the collision is resolved by two separate transmissions in the subsequent RF.

2) Parameter configuration: For any given number of N devices, the lengths of the SF and RF, i.e., L and M, need to be configured properly. With respect to L, a higher success probability in the SF will be achieved when L increases. However, a longer frame leads to higher power consumption and longer transmission time. Herein, we define a configurable parameter λ such that $L = \lambda N$. For a specific group, an appropriate value for λ needs to be configured.

To decide a proper value for M, it is preferable to configure it to a comparatively short length since the purpose of an RF is to provide a second chance to those devices which failed in the SF. To do so, we identify an optimal value for M, denoted as M^*, which optimizes the time slot utilization in the RF. The definition of time slot utilization $U(M, N, \alpha)$ and the procedure of finding M^* are explained as follows (since we are determining M for a given N and α, the only variable in function $U(M, N, \alpha)$, denoted hereafter as $U(M)$, is M):

$$\begin{aligned} U(M) &\triangleq \frac{E[\text{number of success in an RF}]}{\text{number of allocated time slots in the RF}} \\ &= \frac{E[\text{no. of devices in the RF}]\ P(\text{success in the RF})}{M} \\ &= \frac{N\alpha\ (1-1/M)^{N\alpha-1}}{M}, \end{aligned} \tag{1}$$

where α is the hash function collision probability and $0 < \alpha < 1$. To configure M optimally, we need to find

$$M^* \triangleq \arg\max_{M>0} U(M). \tag{2}$$

To find a global optimal value for M, we take the first order derivative of $U(M)$ w.r.t. M and obtain

Algorithm 1: The Partitioning Algorithm in [1]

Input: $\mathcal{S}_i$ and the location of each device in $\mathcal{S}_i$
Output: $\boldsymbol{D} = (P, \mathcal{L})$

construct a complete weighted graph $G = (V, E)$ by $\mathcal{S}_i$; /* where $V = \mathcal{S}_i$ and each $e \in E$ represents the distance between two devices in $\mathcal{S}_i$ */
$V_R \leftarrow V$; $E_R \leftarrow E$; $k \leftarrow 0$;
while $V_R \neq \emptyset$ **do**
  find $u \in V_R$ that is closest to $\mathcal{S}_{i_k}$; /* choose any $u \in V_R$ if $k = 0$ */
  $k \leftarrow k + 1$;
  $\mathcal{S}_{i_k} \leftarrow \{u\}$;
  $E_{cut} \leftarrow \{e = xy \in E_R \,|\, x \in \mathcal{S}_{i_k},\ y \in V_R \setminus \mathcal{S}_{i_k}\}$;
  while $E_{cut} \neq \emptyset$ **do**
    find min $e_m = xy \in E_{cut}$;
    if $\mathcal{S}_{i_k} \cup \{y\}$ has diameter $\leq \sqrt{3}$ (UAV radius) **then**
      $\mathcal{S}_{i_k} \leftarrow \mathcal{S}_{i_k} \cup \{y\}$;
      $E_{cut} \leftarrow E_{cut} \setminus \{e = ya \,|\, a \in \mathcal{S}_{i_k}\}$;
      $E_{cut} \leftarrow E_{cut} \cup \{e = yb \,|\, b \in V_R \setminus \mathcal{S}_{i_k}\}$;
    else
      $E_{cut} \leftarrow E_{cut} \setminus \{e_m\}$;
    end
  end
  $(x_k, y_k) =$ the middle point of $\mathcal{S}_{i_k}$;
  $V_R \leftarrow V_R \setminus \mathcal{S}_{i_k}$;
  $E_R \leftarrow E_R \setminus \{e \,|\, e$ has endpoint in $\mathcal{S}_{i_k}\}$;
end
$P \leftarrow (\mathcal{S}_{i_1}, \ldots, \mathcal{S}_{i_k})$; $\mathcal{L} \leftarrow ((x_1, y_1), \ldots, (x_k, y_k))$;
return $\boldsymbol{D} = (P, \mathcal{L})$

$$\frac{\mathrm{d}U(M)}{\mathrm{d}M} = \frac{N\alpha\,(1-1/M)^{N\alpha-2}}{M^3}\,(N\alpha - M)\,. \tag{3}$$

From (3), it is clear that $M = N\alpha$ is a critical point since $\mathrm{d}U(M)/\mathrm{d}M$ has a positive or negative value when $M < N\alpha$ or $M > N\alpha$ respectively. Thus, we can conclude that $M^* = N\alpha$ is a global optimal value. Note that $N\alpha$ may not necessarily be an integer. When this happens, the integer closest to $N\alpha$, expressed as $\lfloor N\alpha + 1/2 \rfloor$, will substitute for $N\alpha$.

B. Scheme 2: Large-Scale Networks with Static Devices

In order to perform RI-based data collection in Scenario 2, the data collector needs to partition the region into multiple clusters. For each cluster, the data collection procedure is the same as presented in Scheme 1 since multiple groups of devices may exist in each cluster.

Accordingly, multiple rounds are required in order to collect data across the whole region. Therefore, how to partition the region into multiple clusters is essential for Scheme 2 since different partitions lead to a variable number of clusters resulting in various levels of energy consumption. In what follows, we propose two partitioning algorithms with four different flavors (as listed in Table II and explained successively below) where the first algorithm has been proposed in [1] and it is presented in Algorithm 1 above.

1) Elaboration of the algorithms and ingredients: Let us investigate this partitioning problem from the perspective of graph theory. Without loss of generality, we use device and

Algorithm 2: The Partitioning Algorithm in this paper

Input: $\mathcal{S}_i$, the location of each device in $\mathcal{S}_i$, and the location of the center c of the deployment region
Output: $\boldsymbol{D} = (P, \mathcal{L})$

1 construct a complete weighted graph $G = (V, E)$ by $\mathcal{S}_i$;
/* where $V = \mathcal{S}_i$ and each $e \in E$ represents the distance between two devices in $\mathcal{S}_i$ */
2 $V_R \leftarrow V$; $E_R \leftarrow E$; $k \leftarrow 0$;
3 **while** $V_R \neq \emptyset$ **do**
4 find $u \in V_R$ that is farthest to c; // ingredient I1
5 $k \leftarrow k + 1$;
6 $\mathcal{S}_{i_k} \leftarrow \{u\}$;
7 $E_{cut} \leftarrow \{e = xy \in E_R \,|\, x \in \mathcal{S}_{i_k},\ y \in V_R \setminus \mathcal{S}_{i_k}\}$;
8 **while** $E_{cut} \neq \emptyset$ **do**
9 find min $e_m = xy \in E_{cut}$;
10 **if** $\mathcal{S}_{i_k} \cup \{y\}$ has diameter $\leq \sqrt{3}$ (UAV radius) **then**
11 $\mathcal{S}_{i_k} \leftarrow \mathcal{S}_{i_k} \cup \{y\}$;
12 $E_{cut} \leftarrow E_{cut} \setminus \{e = ya \,|\, a \in \mathcal{S}_{i_k}\}$;
13 $E_{cut} \leftarrow E_{cut} \cup \{e = yb \,|\, b \in V_R \setminus \mathcal{S}_{i_k}\}$;
14 **else**
15 $E_{cut} \leftarrow E_{cut} \setminus \{e_m\}$;
16 **end**
17 **end**
18 $(x_k, y_k) =$ find_center$(\mathcal{S}_{i_k})$;
19 **for** $\mu \in V_R \setminus \mathcal{S}_{i_k}$ **do** // ingredient I2
20 **if** distance from μ to $(x_k, y_k) \leq$ UAV radius **then**
21 $\mathcal{S}_{i_k} \leftarrow S_{i_k} \cup \{\mu\}$;
22 **end**
23 **end**
24 $V_R \leftarrow V_R \setminus \mathcal{S}_{i_k}$;
25 $E_R \leftarrow E_R \setminus \{e \,|\, e$ has endpoint in $\mathcal{S}_{i_k}\}$;
26 **end**
27 $P \leftarrow (\mathcal{S}_{i_1}, \ldots, \mathcal{S}_{i_k})$; $\mathcal{L} \leftarrow ((x_1, y_1), \ldots, (x_k, y_k))$;
28 **return** $\boldsymbol{D} = (P, \mathcal{L})$

vertex interchangeably when presenting our algorithms below. Denote by ALG1 and ALG2 respectively the partitioning algorithm in [1] and Algorithm 2 presented in this paper.

A common feature of these two algorithms is that both ALG1 and ALG2 are based on Prim's algorithm, which begins with a tree that contains a single vertex (this may be any of the vertices) and repeatedly adds a closest vertex to the current tree until all vertices have been added. Consider an example where there are four collinear devices x, y, z, w with relative distances shown below, where $t\,d$ and d indicate the distances between two neighboring vertices, $t > 1$, and $t\,d \leq \sqrt{3}\,(\text{UAV radius}) < t\,d + d$.

$$x \overset{t\,d}{---------} y \overset{d}{-} z \overset{t\,d}{---------} w$$

In ALG1, any device among x, y, z, w can be chosen as the initial vertex. Suppose that y is chosen. Then only z can be added into the cluster containing y and the resultant partition by ALG1 consists of three clusters: $\{y, z\}$, $\{x\}$, and $\{w\}$. However, by observing the topology more carefully, one may notice that two clusters $\{x, y\}$ and $\{z, w\}$ would be sufficient.

ALG2 improves ALG1 with the intention of finding an optimal partition for a given network. To do so, two new ingredients are added to ALG1. The first new ingredient (I1) intends to solve the aforementioned x, y, z, w problem. The second new ingredient (I2) is triggered by the observation from our simulation results. More details on I1 and I2 will be elaborated later in this subsection. The benefits of I1 and I2 are shown in Table II, where the performance of the algorithm options is represented by the number of clusters needed in a region. The fewer, the better. It is evident that both I1 and I2 improve ALG1 whereas ALG2 performs best.

TABLE II: Number of clusters for ten data sets each with 700 devices: "+ I1" means "adding I1" and "+ I2" means "adding I2".

Algorithm	ALG1	ALG1 + I1	ALG1 + I2	ALG2
Data set 1	34	29	32	27
Data set 2	26	24	24	22
Data set 3	30	27	27	25
Data set 4	27	21	24	19
Data set 5	28	25	25	22
Data set 6	28	27	27	24
Data set 7	33	29	28	26
Data set 8	33	29	29	26
Data set 9	27	24	24	21
Data set 10	29	28	27	24
Average	29.5	26.3	26.7	23.6

2) Algorithm description: The thrust of ALG2 is to divide $\mathcal{S}_i$ by device locations through an enhanced Prim's algorithm [28] [29].

More specifically, the original Prim's algorithm forms a minimum spanning tree for a *connected* weighted graph by adding a closest vertex to the current subtree at each step. We enhance the original Prim's algorithm by checking whether the diameter of the current subtree at each step exceeds the coverage of the data collector after adding any vertex. If yes, i.e., the current subtree cannot be further extended, the established subtree is stored, and the algorithm starts from a new vertex that has not been added into any subtree. The enhanced algorithm (i.e., ALG2) retrieves a spanning forest in which each subtree represents one cluster of the original device group. Different from the original Prim's algorithm which starts from an arbitrary vertex, ALG2 starts from a vertex that is farthest to the center of the deployment region (hence it will belong to the current convex hull (to be defined later)) to form a new subtree. This property can guarantee that the new subtree starts with an optimal choice. Furthermore, it is worth mentioning that the "if" condition in Line 10 of the enhanced algorithm (i.e., ALG2) is checked by using the following theorem [30].

Theorem 1. Each set $\mathcal{X} \subset \mathbb{R}^2$ of diameter at most d (i.e., any two points have a distance at most d) is contained in some disc of radius $d/\sqrt{3}$.

Consider now a moving data collector, e.g., a UAV, and a set of devices $\mathcal{S} = \bigcup_{i=1}^{n} \mathcal{S}_i$. For a given device group $\mathcal{S}_i$, our aim is to find an optimum partition for this group and the corresponding location for WuC multicasting for each partition set $\mathcal{S}_{i_j}, j = 1, \ldots, k$, such that the total energy consumption of these devices is minimized. The task can be formulated as an optimization problem as follows:

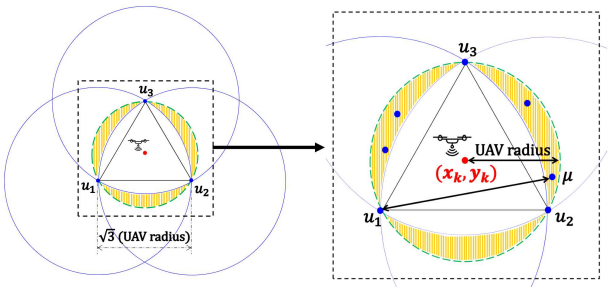


Fig. 8: ALG1 does not add devices in the shaded regions to S_{i_k}.

$$\min_{\boldsymbol{D}} \quad Energy(\mathcal{S}_i) = \sum_{s \in \mathcal{S}_i} Energy(s, \boldsymbol{D})$$
$$\text{s.t. } \forall j \in \{1, \ldots, k\}, \text{ all devices} \in \mathcal{S}_{i_j} \text{ are covered by a WuC sent at } (x_j, y_j) \in \mathbb{R}^2. \quad (4)$$

Here $\boldsymbol{D} = (P, \mathcal{L})$ is a vector that consists of the partition result $P = (\mathcal{S}_{i_1}, \ldots, \mathcal{S}_{i_k})$ and the corresponding WuC locations $\mathcal{L} = ((x_1, y_1), \ldots, (x_k, y_k))$. Note that $\mathcal{S}_{i_h} \cap \mathcal{S}_{i_l} = \emptyset$ for $h \neq l$ and $\bigcup_{j=1}^{k} \mathcal{S}_{i_j} = \mathcal{S}_i$.

After solving the optimization problem, we obtain a solution $\boldsymbol{D}^* = (P^*, \mathcal{L}^*)$ which represents the optimal partition. Hereafter, Scenario 2 can be represented by a collection of multiple clusters each resembling Scenario 1. In other words, the rest of Scheme 2 works the same as or similarly to Scheme 1. However, how to design a travel trajectory for flying UAV across the region of interest is beyond the scope of this paper.

3) Ingredient I1: In the aforementioned x, y, z, w problem, two clusters (i.e., $\{x, y\}, \{z, w\}$) are sufficient, and this partition $\{x, y\}, \{z, w\}$ can be achieved if x (or w) is selected as the initial vertex. Note that x and w are two vertices of the "convex hull" of x, y, z, w (i.e., the line segment $\overline{xw}$) since a line segment can be regarded as a degenerate case of a convex polygon, where the *convex hull* of a set of points is the smallest convex polygon containing all points in the set. Ingredient I1 is based on Fact 1, which intends to find a vertex of the convex hull without explicitly constructing the hull.

Fact 1. If c is the center of the deployment region and V_R is the set of devices that have not been assigned to any cluster, then $u \in V_R$ that is farthest to c (ties can be broken arbitrarily) is a vertex of the convex hull of V_R.

4) Ingredient I2: Now we are ready to explain the second ingredient. According to Line 10 of ALG1, a device, say y, is added to a cluster $\mathcal{S}_{i_k}$ by ALG1 only when $\mathcal{S}_{i_k} \cup \{y\}$ has its diameter $\leq \sqrt{3}$ (UAV radius). In the following, we show that this restriction can be relaxed.

Take Fig. 8 as an example for illustration, in which three devices u_1, u_2, u_3 form an equilateral triangle with the common length of the sides being $\sqrt{3}$ (UAV radius). Suppose that u_1, u_2, u_3 have been added to S_{i_k} and (x_k, y_k) is the center of S_{i_k}. Since the distance from μ to u_1 is larger than $\sqrt{3}$ (UAV radius), ALG1 will not add μ to S_{i_k}. However, the distance from μ to (x_k, y_k) is $\leq$ (UAV radius). Thus μ is within the UAV's coverage disc, meaning that μ can be added to S_{i_k}.

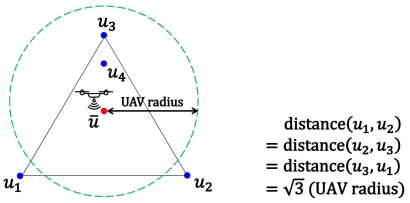


Fig. 9: Illustration of a WuC location: $\overline{u}$ is not a feasible WuC location for $\{u_1, u_2, u_3, u_4\}$ since u_1 and u_2 are not covered by the WuC.

Note that ALG1 will not add any device(s) in the shaded areas in Fig. 8 to S_{i_k}. Let $R =$ (UAV radius) for expression clarity. Then the ratio of the shaded areas to UAV's coverage disc is: $\frac{\pi R^2 - 3(\pi(\sqrt{3}R)^2 \frac{60}{360}) + 2(\frac{\sqrt{3}}{4}(\sqrt{3}R)^2)}{\pi R^2} = \frac{\pi - \frac{3}{2}\pi + \frac{3\sqrt{3}}{2}}{\pi} \approx 32.7\%$. This ratio represents a quite high percentage of uncovered area by ALG1. Accordingly, ingredient I2 intends to add devices that "can be added" but "are not added by ALG1" to S_{i_k} (e.g., devices in the shaded regions in Fig. 8) and it is realized by the for-end loop listed in Lines 19∼23 of ALG2.

5) Clarifying the WuC location (x_k, y_k): According to Line 18 of ALG1, "the middle point of $\mathcal{S}_{i_k}$" is selected as the WuC location for $\mathcal{S}_{i_k}$. Suppose $\mathcal{S}_{i_k} = \{u_1, \ldots, u_m\}$ and let $(u_{j,1}, u_{j,2})$ denote the coordinates of u_j. The middle point of $\{u_1, u_2\}$ is clearly $(\frac{u_{1,1}+u_{2,1}}{2}, \frac{u_{1,2}+u_{2,2}}{2})$. However, when $m \geq 3$, the middle point of $\{u_1, \ldots, u_m\}$ needs some clarification. Fig. 9 shows that $\overline{u} = (\frac{u_{1,1}+\cdots+u_{m,1}}{m}, \frac{u_{1,2}+\cdots+u_{m,2}}{m})$ is not a feasible WuC location.

The find_center function in Line 18 of ALG2 identifies the center for a set of points and it is formulated as follows.

$$\min_{(x_k, y_k) \in \mathbb{R}^2} \quad f(x_k, y_k) = 1$$
$$\text{s.t.} \quad \forall u_j \in \mathcal{S}_{i_k}, \text{distance from } (x_k, y_k) \text{ to } u_j \leq \text{UAV radius}.$$

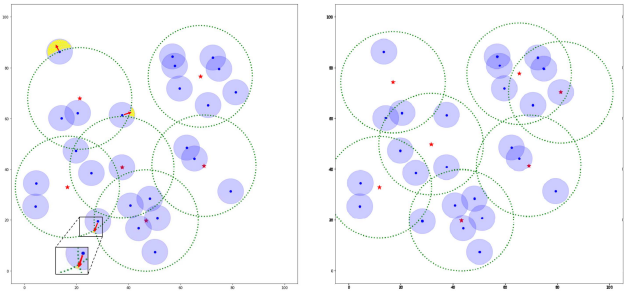
To further clarify this function, a Python implementation of the find_center function is illustrated in Appendix B.

Summary of schemes and algorithms. Scheme 2 integrates Scheme 1 with a specific flavor of the partitioning algorithm and is dedicated to Scenario 2. While ALG1 alone, ALG1+I1, or ALG1+I2 works independently as a partitioning algorithm, ALG2 integrates both I1, I2 and find_center into a single and standalone algorithm.

C. Scheme 3: Large-Scale Networks with Device Mobility

Corresponding to the two mobility patterns outlined in Scenario 3, Scheme 3 deals with these two cases separately. In both cases, ALG2 which performs best in Scenario 2 serves as the basis for Scheme 3 operation. In other words, the main task for Scheme 3 design is to enhance ALG2 to facilitate full coverage clustering under mobility.

Scheme 3A: With *micro mobility*, devices in Scenario 3A can only move within a small region inside a disc with radius r, where $r \ll$ UAV radius. However, a device may move outside the coverage if its initial location is close to the edge. Fig. 10a) shows an example in which ALG2 is not directly applicable to Scenario 3A since three devices have moved to three areas (all marked in yellow with a zoomed-in detail of the smallest



(a) ALG2 using UAV radius. (b) ALG2 using R'.

Fig. 10: Illustration of Scheme 3A, where each shaded circle indicates the micro area within which a device can move. The red stars indicate the cluster centers selected by ALG2 with micro mobility support.

one) outside UAV's coverage disc. Accordingly, an updated ALG2 integrated Scheme 3A works as follows.

- First, replace UAV radius in Lines 10 and 20 of ALG2 by $R' = (\text{UAV radius}) - r$.
- Then, run ALG2 to obtain a feasible partition of the devices based on R'.
- Last, UAV uses its original radius to collect data.

Scheme 3B: Consider a square deployment region with a side length Len within which devices move according to the *group mobility* pattern described in Scenario 3B. For the sake of algorithm simplicity, we make the following assumptions:

(i) A data collection base (DCB) is located at the center of the deployment region and a UAV starts a data collection mission from the DCB.

(ii) The UAV hovers when collecting data from one cluster, and it resumes flying when it finishes data collection of a designated cluster.

(iii) A device stops moving when it receives a WuC, and it resumes moving when both SF and RF end (i.e., after $L + M$ time slots).

Based on these assumptions, ALG2 is updated as follows. The output $\boldsymbol{D} = (P, \mathcal{L})$ of ALG2 consists of the partition result $P = (\mathcal{S}_{i_1}, \ldots, \mathcal{S}_{i_k})$ and the corresponding WuC coordinates $\mathcal{L} = ((x_1, y_1), \ldots, (x_k, y_k))$. Set $\ell_j = (x_j, y_j)$ and $\ell_j(t) = (x_j(t), y_j(t))$, $1 \le j \le k$ for expression clarity, where t is the time elapsed and $\ell_j(0) = \ell_j$.

When devices are static as in Scenario 2, ALG2 guarantees that each device receives at least one WuC. When devices are mobile, one may imagine that devices in each cluster $\mathcal{S}_{i_j}$ "form a team" with their corresponding WuC location ℓ_j. We say that a device in $\mathcal{S}_{i_j}$ "leaves its team" if it lies outside the coverage disc of WuC sent from $\ell_j(t)$. Thus unless a device "leaves its team", it will receive a WuC sent from $\ell_j(t)$. Hence we only need to elaborate when a device will "leave its team".

Note that a device may receive more than one WuC. If a device u receives a WuC sent from $\ell_{j'}(t)$ *before* it receives a WuC sent from its designated $\ell_j(t)$ for any $j' \neq j$, then by assumption (iii), u stops when it receives the WuC sent from $\ell_{j'}(t)$ while its designated $\ell_j(t)$ continues moving and therefore u might "leave its team".

Let v denote the speed and θ denote the moving direction of the target group of devices, of which θ is the angle w.r.t. the positive direction of the x-axis. When a moving device *hits* the boundary of the deployment region, it *rebounds*. Since the devices in $\mathcal{S}_{i_j}$ and their WuC location $\ell_j(t)$ "form a team", they move together and therefore $\ell_j(t)$ rebounds when it hits the boundary of the deployment region.

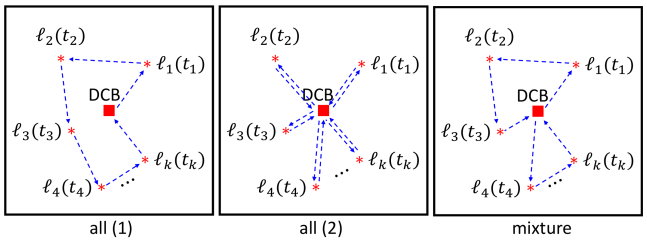


Fig. 11: A UAV flies to collect data with its flying trajectory based on (1) only, (2) only, or a mixture of (1) and (2).

Consider the x-direction and the y-direction of ℓ_j separately. Let p and q be positive integers. For the x-direction, after ℓ_j moves $p(2Len)$ distance along the x-direction, ℓ_j returns back to the same x position with the same x-component of its velocity. Similarly, for the y-direction, after ℓ_j moves $q(2Len)$ distance along the y-direction, ℓ_j returns back to the same y position with the same y-component of its velocity. Thus $(x_j(t), y_j(t))$ (i.e., $\ell_j(t)$) can be calculated as follows. Set

$$\widehat{x}_j = (x_j + vt\cos\theta) \bmod (2Len)$$

and

$$\widehat{y}_j = (y_j + vt\sin\theta) \bmod (2Len)$$

for expression clarity. Then,

$$x_j(t) = \begin{cases} \widehat{x}_j & \text{if } \widehat{x}_j \le Len, \\ 2Len - \widehat{x}_j & \text{if } \widehat{x}_j > Len, \end{cases}$$

and

$$y_j(t) = \begin{cases} \widehat{y}_j & \text{if } \widehat{y}_j \le Len, \\ 2Len - \widehat{y}_j & \text{if } \widehat{y}_j > Len. \end{cases}$$

Denote by V_{UAV} the flying speed of the UAV that is performing a data collection mission and by t_j the time that the UAV is scheduled to collect data from cluster $\mathcal{S}_{i_j}$ (i.e., the time that the UAV is scheduled to send a WuC to $\mathcal{S}_{i_j}$). Let N_j denote the number of devices that receive the WuC sent from $(x_j(t_j), y_j(t_j))$. Scheme 3B works as follows.

- First, run ALG2 and obtain $\boldsymbol{D} = (P, \mathcal{L})$ which consists of the partition result $P = (\mathcal{S}_{i_1}, \ldots, \mathcal{S}_{i_k})$ and the corresponding WuC coordinates $\mathcal{L} = ((x_1, y_1), \ldots, (x_k, y_k))$.
- Then, for $j = 1$ to k do:

1. Evaluate t_j by

$$t_j = \begin{cases} \frac{3}{4}\frac{Len}{V_{\text{UAV}}} & \text{if } j = 1, \\ t_{j-1} + T_s(\underbrace{\lambda N_{j-1}}_{\text{for SF}} + \underbrace{\alpha N_{j-1}}_{\text{for RF}}) & \\ \quad + \frac{3}{2}\frac{Len}{V_{\text{UAV}}} & \text{if } j > 1, \end{cases}$$

where T_s is the length of a time slot.

2. Compute $(x_j(t_j), y_j(t_j))$.
3. The UAV flies to $(x_j(t_j), y_j(t_j))$, hovering there until time t_j; it multicasts a WuC at time t_j and begins to collect data.

Let us now explain the idea behind $\frac{3}{4}\frac{Len}{V_{\text{UAV}}}$ and $\frac{3}{2}\frac{Len}{V_{\text{UAV}}}$. Recall that the deployment region is a square of side length Len and the UAV's flight starts from the DCB. Thus it takes at most $\frac{\sqrt{2}\frac{1}{2}Len}{V_{\text{UAV}}} \approx \frac{3}{4}\frac{Len}{V_{\text{UAV}}}$ time for the UAV to fly from the DCB to the center of $\mathcal{S}_{i_1}$. In general, $j > 1$ and there are two cases:

(1) The UAV flies from the center of $\mathcal{S}_{i_{j-1}}$ to the center of $\mathcal{S}_{i_j}$ directly.

(2) The UAV flies from the center of $\mathcal{S}_{i_{j-1}}$ to the center of $\mathcal{S}_{i_j}$ via the DCB.

The former case takes at most $\frac{\sqrt{2}Len}{V_{\text{UAV}}} \approx \frac{3}{2}\frac{Len}{V_{\text{UAV}}}$ time and the latter case takes at most $\frac{\sqrt{2}\frac{1}{2}Len}{V_{\text{UAV}}} + \frac{\sqrt{2}\frac{1}{2}Len}{V_{\text{UAV}}} \approx \frac{3}{2}\frac{Len}{V_{\text{UAV}}}$ time. In either case, $\frac{3}{2}\frac{Len}{V_{\text{UAV}}}$ applies. Fig. 11 illustrates how a UAV performs a data collection mission with its flying trajectory following case (1) or (2), or a hybrid route.

IV. Performance Analysis

Recall that L is the length of the SF and M is the length of the RF. We assume that each time slot in the SF has the same probability (i.e., $\frac{1}{L}$) of being selected and each time slot in the RF also has the same probability (i.e., $\frac{1}{M}$) of being selected. In this section, we analyze the performance of our schemes with respect to three parameters, i.e., access success probability, access delay, and energy consumption. Closed-form expressions are deduced based on our definitions. However, calculating the flying time of the UAV for a data collection mission is beyond the scope of this paper.

A. Access Success Probability

Consider a randomly selected device as a reference device (RD). We calculate the access success probability for this device in one round of data collection, denoted by P_{rd}, as follows. For a success access of the RD, its successful transmission could happen during either the SF or the RF. Therefore, P_{rd} is equal to the sum of success probability in the SF and the success probability in the RF. Let P_s^{sf} and P_s^{rf} stand for the successful transmission probability during the SF and the RF, respectively. Then, $P_{rd} = P_s^{sf} + P_s^{rf}$.

Furthermore denote by P_i^{sf} and P_j^{rf} the successful transmission probability during the i-th slot of the SF and the j-th slot of the RF respectively. Clearly, $P_s^{sf} = \sum_{i=1}^{L} P_i^{sf}$ and $P_s^{rf} = \sum_{j=1}^{M} P_j^{rf}$. Thus a critical step to obtain P_{rd} is to elaborate P_i^{sf} and P_j^{rf}. Recalling that α is the hash function collision probability, P_i^{sf} and P_j^{rf} can be expressed as follows.

First, given that the probability of the i-th slot of the SF being selected is $\frac{1}{L}$ and such a selection contributes to a success access of the RD only when there is no hash function collision, we obtain $P_i^{sf} = \frac{1-\alpha}{L}$.

Second, consider P_j^{rf}. Keep in mind that the RD needs to retransmit in the RF only when there is a hash function collision and the corresponding probability is α. Suppose that the RD retransmits during the j-th slot of the RF. As mentioned above, the probability of the j-th slot of the RF being selected is $\frac{1}{M}$. According to (3), M is set to $N\alpha$, meaning that the expected number of devices that need to retransmit in the RF is $N\alpha$. Since the RD succeeds in the j-th slot of the RF only when this slot is not selected by any of the other $N\alpha - 1$ devices that retransmit in the RF, we obtain consequently $P_j^{rf} = \alpha\frac{1}{M}(1-\frac{1}{M})^{N\alpha-1}$.

Therefore, P_{rd} can be expressed as follows.

$$\begin{aligned} P_{rd} &= P_s^{sf} + P_s^{rf} = \sum_{i=1}^{L} P_i^{sf} + \sum_{j=1}^{M} P_j^{rf} \\ &= (1-\alpha) + \alpha(1-\frac{1}{M})^{N\alpha-1}. \end{aligned} \tag{5}$$

B. Access Delay

The access delay of the RD, T_{rd}, is defined as the expected duration from the instant when a WuC is multicast to the moment until the successful transmission of the RD. Note that both data frame and ACK/NACK transmissions happen within the same time slot. As presented in the previous subsection, $P_i^{sf} = \frac{1-\alpha}{L}$ and $P_j^{rf} = \alpha\frac{1}{M}(1-\frac{1}{M})^{N\alpha-1}$. Denote by P_k^F the failed transmission probability during the k-th slot of the RF. We now elaborate P_k^F.

Suppose that the RD needs to retransmit during the RF after a hash function collision and it has selected the k-th slot for its retransmission. As assumed earlier, this probability is $\frac{1}{M}$. The RD fails in the k-th slot of the RF only when this slot is also selected by at least one of the other $N\alpha - 1$ devices that retransmit in the RF and the corresponding probability is $1-(1-\frac{1}{M})^{N\alpha-1}$. Consequently, $P_k^F = \alpha\frac{1}{M}(1-(1-\frac{1}{M})^{N\alpha-1})$.

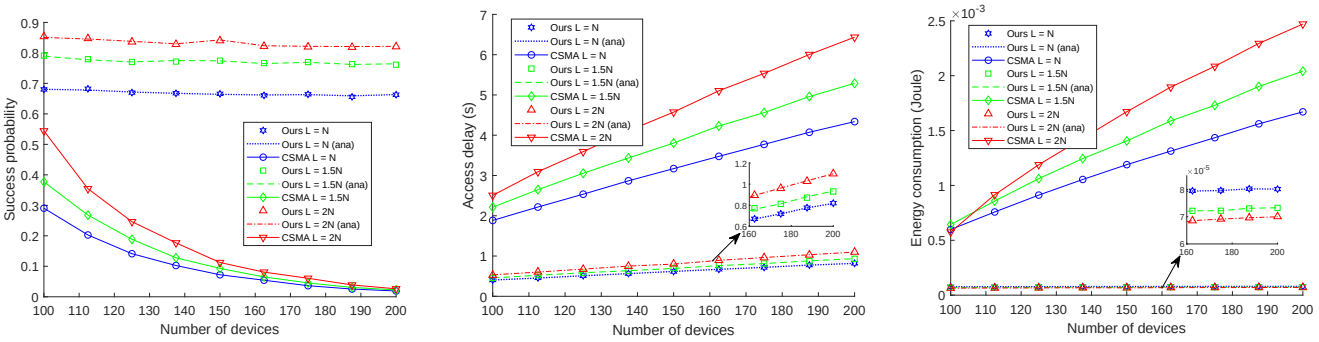
Based on the above deduction, T_{rd} can be obtained as:

$$\begin{aligned} T_{rd} &= T_{wuc} + \sum_{i=1}^{L} iT_sP_i^{sf} + \sum_{j=1}^{M}(L+j)T_sP_j^{rf} \\ &\quad + \sum_{k=1}^{M}(L+M)T_sP_k^F \\ &= T_{wuc} + T_s\frac{(1-\alpha)(L+1)}{2} \\ &\quad + T_s\frac{\alpha(1-\frac{1}{M})^{N\alpha-1}(1+2L+M)}{2} \\ &\quad + T_s(\alpha - \alpha(1-\frac{1}{M})^{N\alpha-1})(L+M) \end{aligned} \tag{6}$$

where T_{wuc} denotes the WuC transmission and decoding duration.

C. Energy Consumption

Consider that the RD belongs to the target group for a cycle of data transmission. After the reception of a WuC which consumes E_{wuc} amount of energy, the RD needs to stay in the light sleep mode until its scheduled time slot arrives. Assuming that it transmits during the i-th slot of the SF, it will then consume energy in the light sleep mode for $i-1$ slots and in the active mode for one slot. If there is a hash function collision, the transmission will continue in the RF. This means that the RD will stay in the active mode for 2 time slots in total. Based on the expressions obtained above for P_i^{sf}, P_j^{rf}, and P_k^F, the expected energy consumption of the RD during one cycle, E_{rd}, is obtained as follows.



(a) Scheme1: Access success probability. (b) Scheme 1: Access delay. (c) Scheme 1: Energy consumption.

Fig. 12: Performance of Scheme 1 in term of access success probability, access delay, and energy consumption as N varies.

$$
\begin{aligned}
E_{rd} = \; & E_{wuc} \\
& + \sum_{i=1}^{L}((i-1)P_{ls} + P_a + (L-i+M)P_{ds})T_s P_i^{sf} \\
& + \sum_{j=1}^{M}((L+j-2)P_{ls} + 2P_a + (M-j)P_{ds})T_s P_j^{rf} \\
& + \sum_{k=1}^{M}((L+k-2)P_{ls} + 2P_a + (M-k)P_{ds})T_s P_k^{F} \\
= \; & E_{wuc} \\
& + T_s \frac{(1-\alpha)((L-1)P_{ls} + 2P_a + (L+2M-1)P_{ds})}{2} \\
& + T_s \frac{\alpha((2L+M-3)P_{ls} + 4P_a + (M-1)P_{ds})}{2}
\end{aligned} \tag{7}
$$

where P_{ls}, P_{ds}, and P_a denote the power consumption levels for light sleep, deep sleep, and active respectively [11].

V. Simulations and Numerical Results

To evaluate the performance of the proposed schemes and algorithms, we develop on a custom-built simulator in Python and MATLAB which mimics the behavior of the studied networks and conduct extensive discrete-event simulations. The performance of our schemes is compared with that of the carrier sense multiple access with collision avoidance (CSMA/CA)-based access (for Scheme 1) and another reference scheme which is referred to as *Square* in [1] (for Scheme 2) with respect to the three parameters defined above. For Scheme 3, we assess the completeness of clustering and device coverage in additions to these three parameters.

A. Simulation Setup

Consider a region of interest with a given number of devices N and a known UAV radius. Our simulations are performed based on the following three steps. 1) Randomly generate the locations of N devices in the given region; 2) Apply separately each partitioning algorithm on these N devices (for Scenarios 2 and 3 only); and 3) Perform the WuR protocol based on the employed scheme and partitioning algorithm. The results reported below are the average values calculated over 1800 simulation runs (for Schemes 1 and 2) and 1000 simulation runs (for Schemes 3A and 3B) respectively.

For network configuration in our simulations, we vary the number of devices in the deployment region, N, from 100 to 200 (for Scheme 1) or from 200 to 700 (for Schemes 2 and 3), respectively. For each frame transmission, a packet size of 1000 bits transmitted at the data rate of 250 kbps is considered. Furthermore, three values of L are configured. Based on the value of N, the optimal value of M is decided by $M^* = N\alpha$. To evaluate the effect of frame length on our schemes, we configure λ, which decides frame length L based on the relationship of $L = \lambda N$, as $\lambda = 1$, 1.5, and 2 respectively. Correspondingly, we set $L = N$, $L = 1.5N$, and $L = 2N$ in our simulations. From the curves shown in Fig. 12, it is evident that the simulation results match the analytical ones precisely. For the clarity of illustration, we do not include the results from both analysis and simulations in other figures.

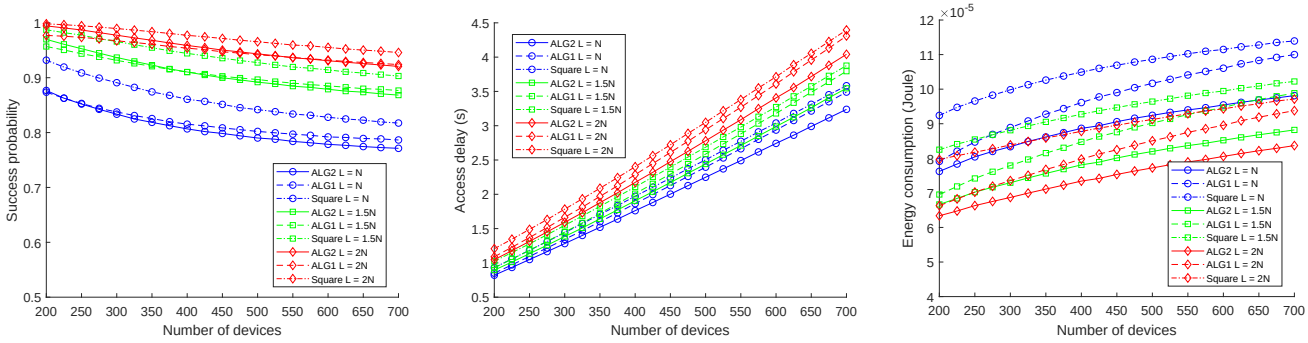
B. Performance of Scheme 1 as Device Population Varies

1) Access success probability: To make a fair comparison, we configure the number of time slots for CSMA/CA the same as what is used in our scheme. The obtained results for access success probability as the number of devices varies are shown in Fig. 12a).

As can be observed, the access success probabilities obtained by Scheme 1 remain stable as the number of devices increases whereas the CSMA/CA-based probabilities decrease monotonically. This is because the employment of a hash function assigns dedicated time slots for data uploading and can effectively avoid collisions. In contrast, the CSMA/CA-based access scheme relies solely on random access causing lots of collisions and the collision problem becomes worse as the device population grows.

2) Access delay: According to the design of our scheme, a longer frame length is needed with a larger number of devices. Correspondingly, a longer delay is expected as the number of devices N increases. However, the slope of the delay increase in our scheme is not as sharp as in CSMA/CA and the obtained delay from our scheme is shorter for the whole range of device population, as illustrated in Fig. 12b).

The reason for this behavior is as follows. To resolve collisions in CSMA/CA, multiple retransmissions of the same



(a) Scheme 2: Access success probability. (b) Scheme 2: Access delay. (c) Scheme 2: Energy consumption.

Fig. 13: Performance of Scheme 2 in term of access success probability, access delay, and energy consumption as N varies.

frame are allowed. The larger the device population, the higher the number of retransmissions. Consequently, much longer access delay is experienced due to retransmissions. With our scheme, a longer frame length is allocated based on the value of α. Since our scheme requires typically only one transmission, the achieved delay is still much shorter than the one obtained in CSMA/CA.

Furthermore, it is worth mentioning that the achieved delay in our scheme is also shorter than what is obtained in traditional unicast based data collection. This is because WuCs are transmitted at a much lower data rate [11]. Instead of sending a WuC individually to each device, our schemes require only one WuC for each group of devices in one cluster.

3) Energy consumption: The energy consumption performance of Scheme 1 is compared with that of CSMA/CA in Fig. 12c). As expected, our scheme outperforms CSMA/CA overwhelmingly for the whole range of device population. The reason is as follows.

For each frame transmission in CSMA/CA, a device has to sense the channel and perform backoff before each data frame transmission. The channel sensing and backoff procedures also consume a certain amount of energy at the mW power level. On the other hand, our scheme requires WuC decoding and devices wait for scheduled transmission only at the μW power level. Consequently, the same superiority for energy consumption in WuR based solutions is observed.

C. Performance of Scheme 2 as Device Population Varies

The performance of Scheme 2 is compared with a reference scheme, *Square*, which partitions the region of interest into multiple small square areas each covered by one WuC. A similar algorithm referred to as *disk cover* for charging pad distribution in wireless rechargeable sensor networks can be found in [31]. The principle of *Square* is as follows.

Assume that the region of interest as a rectangle with sides $length$ and $width$ respectively and recall that R =(UAV radius). *Square* partitions this region into $\left\lceil \frac{length}{\sqrt{2}R} \right\rceil \left\lceil \frac{width}{\sqrt{2}R} \right\rceil$ number of small square areas, each of these small square areas with side length of $\sqrt{2}R$. Thus, *Square* guarantees that each small square area can be covered by one WuC. Since devices are randomly deployed across the whole region, some small square areas might not contain any device. *Square* then removes these redundant small square areas.

Square does not only completely cover the deployed region of interest but also guarantees at least one WuC for each non-empty small square area. To collect data, the UAV hovers only in the areas that have non-empty devices. The obtained results are illustrated in Fig. 13. Therein and hereafter Scheme 2 in [1] and Scheme 2 in this paper are denoted by ALG1 and ALG2 respectively.

1) Access success probability: As illustrated in Fig. 13a), with a much larger population of $200 \sim 700$ devices, all three schemes achieve very high probabilities when $L \geq 1.5N$. The curves of ALG1 and ALG2 twist together when $L \geq 1.5N$. Moreover, *Square* achieves about $2 \sim 5\%$ higher success probabilities than both ALG1 and ALG2. This is because *Square* divides the same region into more clusters leading to better coverage. However, the better coverage is obtained at a cost of longer access delay and higher energy consumption as presented below. In general, a joint performance assessment including all the three parameters is needed for proper parameter configuration.

2) Access delay: For access delay in Scheme 2, it is worth mentioning that we consider only access delay as defined above, not the flying or/and hovering time for the UAV. It is evident from Fig. 13b) that ALG2 achieves shortest delay under all three frame length configurations.

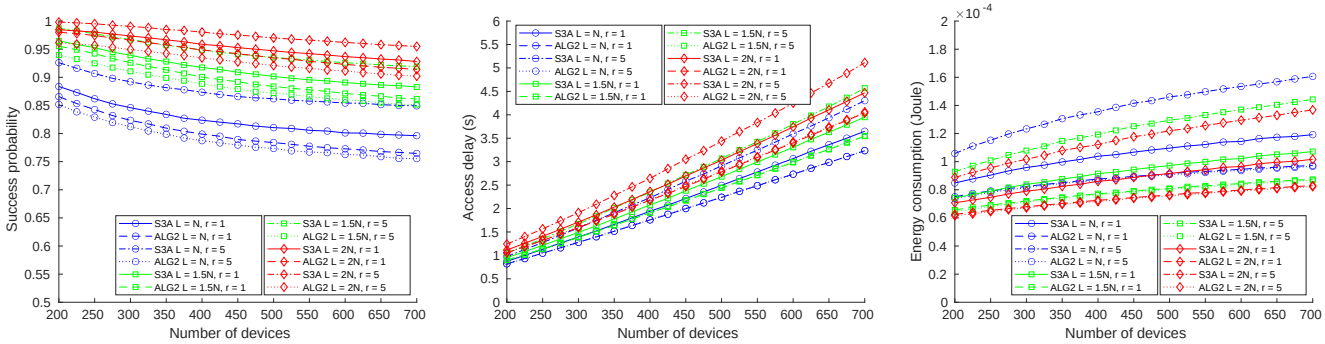
When a longer frame length is adopted, it will lead to a longer delay. If we care about the delay performance only, it is preferable to configure a frame with the shortest length.

3) Energy consumption: For energy consumption, it is evident from Fig. 13c) that lowest energy consumption has been achieved when ALG2 is employed regardless of the configured frame length.

For energy consumption with different frame lengths, we observe that a longer length leads to lower per-node energy consumption. By jointly considering the performance of all these three parameters, we would recommend configuring the frame length as $L = 1.5N$ for Scheme 2.

D. Performance of Scheme 3A as Device Population and Device Moving Region Vary

Recall that each device in Scenario 3A can only perform micro mobility inside a disc with radius r. In our simulations, r is configured with different values ranging from 0 to 5 meters. Due to the page limit, only the results with $r = 1$ m and



(a) Scheme 3A: Access success probability. (b) Scheme 3A: Access delay. (c) Scheme 3A: Energy consumption.

Fig. 14: Performance of Scheme 3A in term of access success probability, access delay, and energy consumption as N and r vary.

$r = 5$ m (marked simply as $r = 1$ and $r = 5$ respectively) are included in Fig. 14 where Scheme 3A is denoted by S3A and ALG2 is regarded as a reference scheme.

1) Clustering and coverage under micro mobility: By running the updated partitioning algorithm presented in Scheme 3A in Subsec. III-C, the UAV can adjust device clustering as well as its locations to send WuCs. As illustrated in Fig. 10b), the micro mobility area of each device (which is a disc of radius r) lies *completely* within the coverage disc with at least one WuC. As such, it is evident that Scheme 3A can properly deal with micro mobility of devices foreseen in Scenario 3A.

2) Access success probability: As illustrated in Fig. 14a), both S3A and ALG2 achieve very high probabilities when $L \geq 1.5N$. Moreover, S3A achieves about $4 \sim 10\%$ and $1 \sim 3\%$ higher success probabilities than ALG2 does when $r = 5$ and $r = 1$, respectively. This result seems somehow counter-intuitive. However, this benefit is achieved thanks to S3A's adaptability by replacing the UAV radius in Lines 10 and 20 in ALG2 with (UAV radius) $- r$. Correspondingly, S3A divides the same region into more clusters, leading to better coverage. On the other hand, this benefit is achieved at a cost of deteriorated performance for access delay and energy consumption, as presented below.

3) Access delay: Keep in mind again that we consider only access delay as defined above, not the flying or/and hovering time for the UAV. From Fig. 14b), the following performance can be observed for all three frame length configurations.

(i) S3A results in longer delay than ALG2 does; (ii) S3A with $r = 5$ takes longer delay than S3A with $r = 1$; (iii) the curves of ALG2 with $r = 1$ and ALG2 with $r = 5$ nearly overlap but the latter one is slightly higher. This is compatible with the fact that the success probability of $r = 5$ is lower than that of $r = 1$; and (iv) when a longer frame length is adopted, both S3A and ALG2 will lead to longer delay no matter $r = 1$ or $r = 5$.

4) Energy consumption: From Fig. 14c), it is evident that S3A consumes higher energy especially when r becomes larger. If higher energy consumption is tolerable for certain applications, then the gain is higher success probability as shown in Fig. 14a). Moreover, when comparing the energy consumption based on different frame lengths, we observe that a longer length leads to lower per-node energy consumption. When taking into account all these three parameters, the recommended configuration for frame length as $L = 1.5N$ applies also to S3A.

E. Performance of Scheme 3B as Device Population and Device Velocity Vary

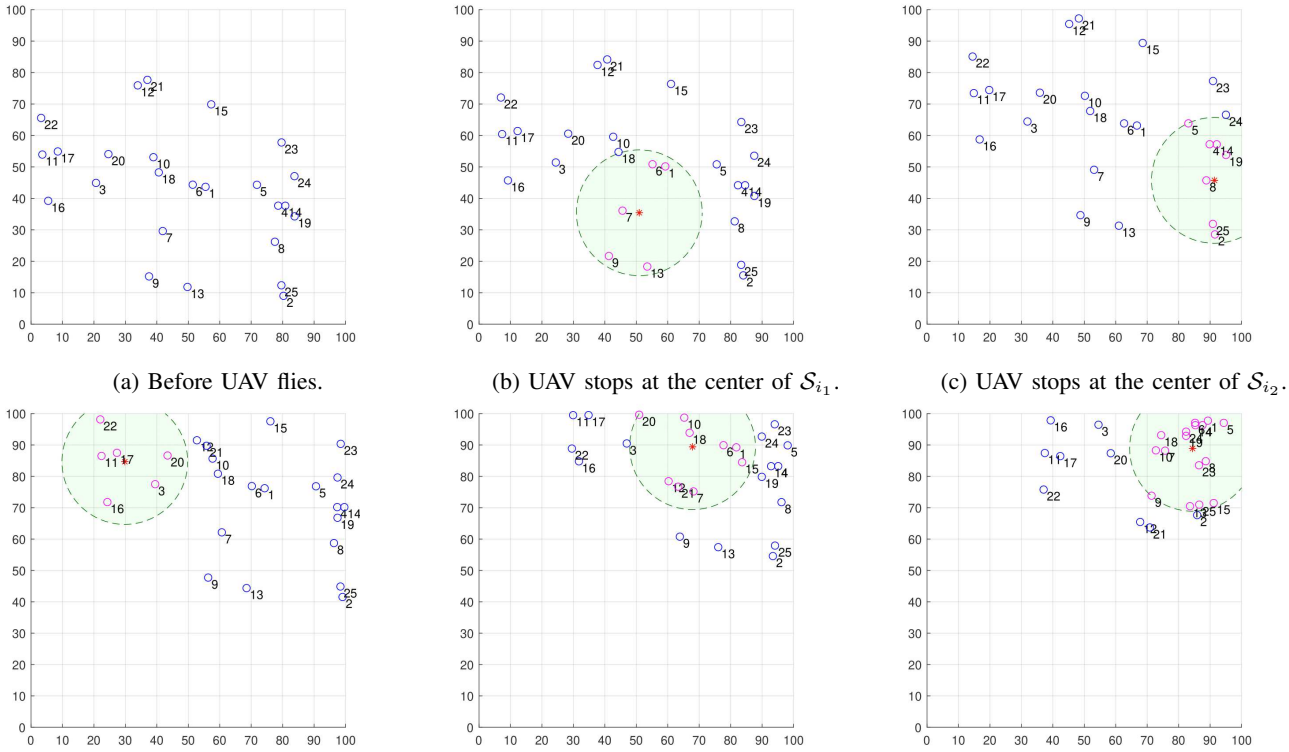
To evaluate the performance of Scheme 3B (denoted as S3B hereafter), we configure group mobility at two speeds, i.e., low at $v = 1$ m/s and high at $v = 3$ m/s, as presented in Scheme 3B description in Subsec. III-C. The results are illustrated in Figs. 15, 16, and 17 respectively.

1) Clustering and coverage under group mobility: To illustrate device locations and the clustering results obtained by S3B explicitly, we configure the region of interest with merely 25 mobile devices instead of hundreds of devices which are used for obtaining the other performance results for Schemes 2 and 3. Figs. 15 and 16 depict a few snapshots of a data collection mission. The devices are moving either at a slow speed with $\theta = \frac{1}{3}\pi$ as shown in Fig. 15 or at a high speed with $\theta = 0$ as shown in Fig. 16, according to the group mobility pattern presented in Scenario 3B. For traceability, we label each device with a reference number. In both figures, the UAV speed is $V_{\text{UAV}} = 10$ m/s.

As can be observed from these two figures, each device receives at least one WuC despite at a different location for each snapshot. In more details, devices 2 and 25 in Figs. 15d) and 15e), as well as devices 4 and 10 in Figs. 16b) and 16c), *rebound*, respectively. Moreover, device 7 in Fig. 15 and device 1 in Fig. 16 receive three and two WuCs respectively, although both devices might "leave their team". As such, the completeness of clustering under group mobility and device coverage provided by Scheme 3B is validated.

2) Access success probability: As shown in Fig. 17a), S3B achieves very high probabilities when $L \geq 1.5N$. Furthermore, as shown in the figure, a longer frame length yields a higher access success probability. The access success probabilities remain stable as the number of devices increases.

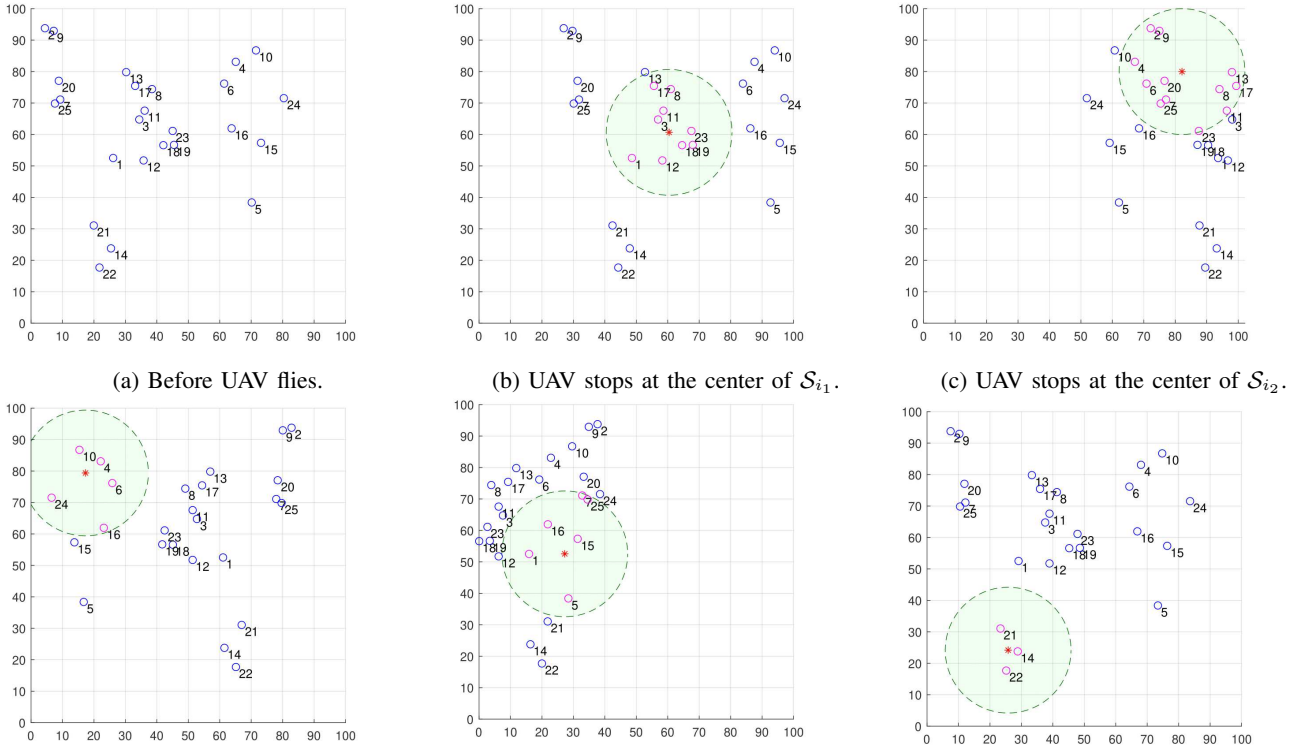
For each frame length, it can be observed that the curves obtained with these two speeds almost overlap. The reason is that the average time for each round of data transmission by one cluster is only $0.2 \sim 0.3$ sec; and during such a short time interval, the displacement of a device is insignificant no matter it is moving at a low or high speed as defined above.



(a) Before UAV flies. (b) UAV stops at the center of $\mathcal{S}_{i_1}$. (c) UAV stops at the center of $\mathcal{S}_{i_2}$.

(d) UAV stops at the center of $\mathcal{S}_{i_3}$. (e) UAV stops at the center of $\mathcal{S}_{i_4}$. (f) UAV stops at the center of $\mathcal{S}_{i_5}$.

Fig. 15: Illustration of Scheme 3B w.r.t. $v = 1$ m/s (low speed), $\theta = \frac{1}{3}\pi$, and $V_{\text{UAV}} = 10$ m/s.

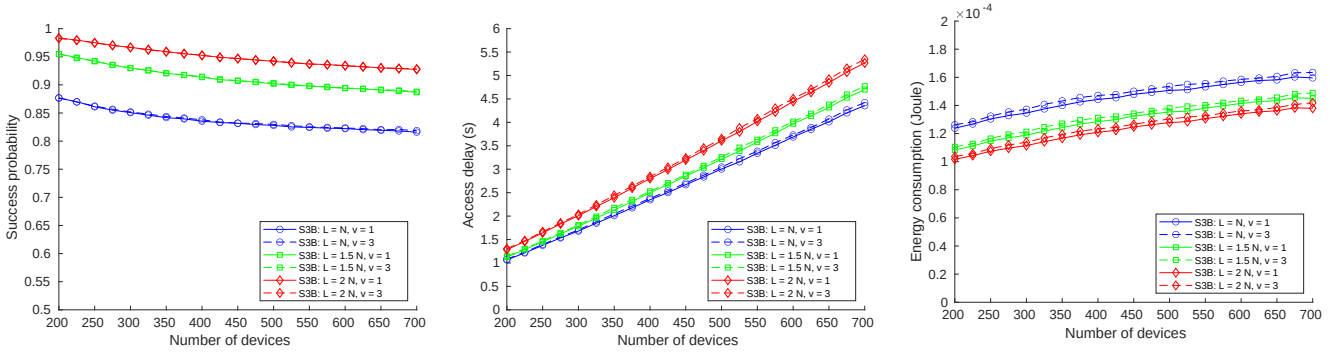


(a) Before UAV flies. (b) UAV stops at the center of $\mathcal{S}_{i_1}$. (c) UAV stops at the center of $\mathcal{S}_{i_2}$.

(d) UAV stops at the center of $\mathcal{S}_{i_3}$. (e) UAV stops at the center of $\mathcal{S}_{i_4}$. (f) UAV stops at the center of $\mathcal{S}_{i_5}$.

Fig. 16: Illustration of Scheme 3B w.r.t. $v = 3$ m/s (high speed), $\theta = 0$, and $V_{\text{UAV}} = 10$ m/s.

3) Access delay: From Fig. 17b), the performance of S3B is observed as follows. (i) S3B with a higher device speed takes longer delay than S3B with a lower speed but the difference is small; and (ii) when a longer frame length is adopted, S3B will lead to longer delay regardless of the device speed.



(a) Scheme 3B: Access success probability. (b) Scheme 3B: Access delay. (c) Scheme 3B: Energy consumption.

Fig. 17: Performance of Scheme 3B in term of access success probability, access delay, and energy consumption as N and v vary.

4) Energy consumption: As evident in Fig. 17c), a longer frame length leads to lower per-node energy consumption. For all three frame length configurations, S3B at a higher device speed leads to higher energy consumption in comparison with S3B at a lower speed but the difference is small. The reason is the same as explained above for access delay.

Finally, when comparing the results obtained based on the three frame length configurations, we reach the same conclusion as what was observed for ALG2 and S3A, i.e., $L = 1.5N$ achieves the best performance as a tradeoff among the three parameters.

VI. Conclusions and Future Work

In this paper, we have proposed three receiver-initiated data collection schemes integrated with two cluster partitioning algorithms, targeting at providing collision-free data transmissions for WuR enabled small- and large-scale IoT networks, with and without device mobility respectively. Instead of waking up each device individually through a unicast WuC, we initiate a data collection procedure via a multicast WuC to all devices in a cluster and schedule their data transmissions via a hash function. For large-scale networks, a region of interest is partitioned into multiple clusters and a data collection mission is composed of multiple data collection rounds each responsible for collecting data from a group of devices in one cluster. Through analysis and simulations based on both static and mobile network topologies, we demonstrate that the proposed schemes significantly improve the performance of traditional data collection schemes. To achieve ideal performance, dedicated scheme and algorithm selection as well as suitable parameter configuration are needed. As our future work, we plan to study the effects of channel impairments on our schemes. Another definite direction for our future work is to implement the schemes in a prototype testbed and perform real-life experiments.

References

[1] C.-A. Hsu, F. Y. Li, C.Y. Chen, and Y.-C. Tseng, "Achieving ultra energy-efficient and collision-free data collection in wake-up radio enabled mIoT," in *Proc. IEEE ICC*, Jun. 2020, pp. 1-6.

[2] Ericsson, "Key technology choices for optimal massive IoT devices," *Ericsson Technology Review*, Jan. 2019.

[3] A. Ghasempour, "Internet of Things in smart grid: Architecture, applications, services, key technologies, and challenges," *Inventions Journal*, vol. 4, no. 1, pp. 1-12, 2019.

[4] Sierra Wireless, Ericsson, Altair, Sony, Virtuosys, AT&T, Verizon, Sequans, Orange, KDDI, Nokia, NTT Docomo, KT, SoftBank, SK Telcom, Telenor, and KPN, "Evaluation of LTE-M towards 5G IoT requirements," *White Paper*, v1.1, Mar. 2018.

[5] 3GPP TR 45.820, "Cellular system support for ultra-low complexity and low throughput Internet of Things (CIoT)," v13.1.0, Dec. 2015.

[6] F. Adelantado, X. Vilajosana, P. Tuset-Peiro, B. Martinez, J. Melià-Seguí, and T. Watteyne, "Understanding the limits of LoRaWAN," *IEEE Commun. Mag.*, vol. 55, no. 9, pp. 34–40, Sep. 2017.

[7] A. Lavric, A. I. Petrariu, and V. Popa, "Long range SigFox communication protocol scalability analysis under large-scale, high-density conditions," *IEEE Access*, vol. 7, pp. 35816–35825, Apr. 2019.

[8] C. Gomez, A. Minaburo, L. Toutain, D. Barthel, and J. C. Zuniga, "IPv6 over LPWANs: Connecting low power wide area networks to the Internet (of Things)," *IEEE Wireless Commun.*, vol. 27, no. 1, pp. 206–213, Feb. 2020.

[9] J. Oller, I. Demirkol, J. Casademont, J. Paradells, G. U. Gamm, and L. Reindl, "Has time come to switch from duty-cycled MAC protocols to wake-up radio for wireless sensor networks?" *IEEE/ACM Trans. Netw.*, vol. 24, no. 2, pp. 674-687, Apr. 2016.

[10] M. Magno, V. Jelicic, B. Srbinovski, V. Bilas, E. Popovici, and L. Benini, "Design, implementation, and performance evaluation of a flexible low-latency nanowatt wake-up radio receiver," *IEEE Trans. Ind. Informat.*, vol. 12, no. 2, pp. 633-644, Apr. 2016.

[11] A. Frøytlog, T. Foss, O. Bakker, G. Jevne, M. A. Haglund, F. Y. Li, J. Oller, and G. Y. Li, "Ultra-low power wake-up radio for 5G IoT," *IEEE Commun. Mag.*, vol. 57, no. 3, pp. 111–117, Mar. 2019.

[12] S. Rostami, P. Kela, K. Leppanen, and M. Valkama, "Wake-up radio-based 5G mobile access: Methods, benefits, and challenges," *IEEE Commun. Mag.*, vol. 58, no. 7, pp. 14-20, Jul. 2020.

[13] D.-J. Deng, M. Gan, Y.-C. Guo, J. Yu, Y.-P. Lin, S.-Y. Lien, and K.-C. Chen, "IEEE 802.11ba: Low-power wake-up radio for green IoT," *IEEE Commun. Mag.*, vol. 57, no. 7, pp. 106-112, Jul. 2019.

[14] S. Rostami, K. Heiska, O. Puchko, K. Leppanen, and M. Valkama, "Pre-grant signaling for energy-efficient 5G and beyond mobile devices: Method and analysis," *IEEE Trans. Green Commun. Netw.*, vol. 3, no. 2, pp. 418-432, Jun. 2019.

[15] F. Z. Djiroun and D. Djenouri, "MAC protocols with wake-up radio for wireless sensor networks: A review," *IEEE Commun. Surveys & Tuts.*, vol. 19, no. 1, pp. 587-618, 1st Quart., 2017.

[16] R. Piyare, A. L. Murphy, C. Kiraly, P. Tosato, and D. Brunelli, "Ultra low power wake-up radios: A hardware and networking survey," *IEEE Commun. Surveys & Tuts.*, vol. 19, no. 4, pp. 2117-2157, 4th Quart., 2017.

[17] M. Ghribi and A. Meddeb, "Survey and taxonomy of MAC, routing and cross layer protocols using wake-up radio," *J. Netw. & Comput. Appl., Elsevier*, vol. 149, article 102465, pp. 1–24, Jan. 2020.

[18] 3GPP TS 36.213, "LTE; Evolved universal terrestrial radio access (E-UTRA); Physical layer procedures," R16, v16.3.0, Nov. 2020.

[19] S. Rostami, S. Lagen, M. Costa, M. Valkama, and P. Dini, "Wake-up radio based access in 5G under delay constraints: Modeling and optimization," *IEEE Trans. Commun.*, vol. 68, no. 2, pp. 1044–1057, Feb. 2020.

[20] N. Mazloum and O. Edfors, "Interference-free OFDM embedding of wake-up signals for low-power wake-up receivers," *IEEE Trans. Green Commun. Netw.*, vol. 4, no. 3, pp. 669–677, Sep. 2020.

[21] M. S. Adam, L. Y. Por, M. R. Hussain, N. Khan, T. F. Ang, M. H. Anisi, Z. Huang, and I. Ali, "An adaptive wake-up-interval to enhance receiver-based PSMAC protocol for wireless sensor networks," *Sensors*, vol. 17, no. 17, article 3732, Sep. 2019.
[22] L. Guntupalli, D. Ghose, F. Y. Li, and M. Gidlund, "Energy efficient consecutive packet transmissions in receiver-initiated wake-up radio enabled WSNs," *IEEE Sensors J.*, vol. 18, no. 11, pp. 4733–4745, Jun. 2018.
[23] M. Peng, W. Liu, T. Wang, and Z. Zeng, "Relay selection joint consecutive packet routing scheme to improve performance for wake-up radio-enabled WSNs," *Wireless Commun. and Mobile Comput.*, vol. 2020, article 7230565, Jan. 2020.
[24] R. Singh and B. Sikdar, "A receiver initiated low delay MAC protocol for wake-up radio enabled wireless sensor networks," *IEEE Sensors J.*, vol. 20, no. 22, pp. 13796–13807, Nov. 2020.
[25] D. Ghose, F. Y. Li, and V. Pla, "MAC protocols for wake-up radio: Principles, modeling and performance analysis," *IEEE Trans. Ind. Informat.*, vol. 14, no. 5, pp. 2294-2306, May 2018.
[26] I. B. Damgård, "A design principle for hash functions," *Lecture Notes in Computer Science*, vol. 435, pp. 416-427, Jul. 2001.
[27] D. Ghose, A. Frøytlog, and F. Y. Li, "Enabling early sleeping and early data transmission in wake-up radio-enabled IoT networks," *Comput. Netw.*, vol. 153, pp. 132-144, Apr. 2019.
[28] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms (3rd Ed.)*. Cambridge, MA, USA: The MIT Press, 2009.
[29] D. B. West, *Introduction to Graph Theory (2nd Ed.)*. London, UK: Pearson, 2015.
[30] J. Matoušek, *Lectures on Discrete Geometry*, New York, NY, USA: Springer-Verlag, 2002.
[31] J. Chen, C. W. Yu, and W. Ouyang, "Efficient wireless charging pad deployment in wireless rechargeable sensor networks," *IEEE Access*, vol. 8, pp. 39056–39077, Mar. 2020.

APPENDIX A
LIST OF NOTATIONS

Notation	Description
N	Number of devices
L	Length of a scheduled frame (SF)
$\mathcal{S}_i$	A set of devices belonging to group i
$\mathcal{S}$	A cluster of devices where $\mathcal{S} = \bigcup_{i=1}^{n} \mathcal{S}_i$ and $\mathcal{S}_i \cap \mathcal{S}_j = \emptyset$ if $i \neq j$
$\mathcal{S}_{i_j}$	The j-th cluster of $\mathcal{S}_i$, where $j = 1, \ldots, k$, $\mathcal{S}_{i_h} \cap \mathcal{S}_{i_l} = \emptyset$ for $h \neq l$, and $\bigcup_{j=1}^{k} \mathcal{S}_{i_j} = \mathcal{S}_i$
M	Length of a random frame (RF)
M^*	An optimal value for M
α	Collision probability of a hash function
λ	A configurable parameter to decide L based on N such that $L = \lambda N$
r	The radius of a disc with which devices with micro mobility can move
$U(M, N, \alpha)$	Time slot utilization
$U(M)$	Time slot utilization for given N and α
$\mathrm{d}U(M)/\mathrm{d}M$	First order derivation of $U(M)$ w.r.t. M
$\boldsymbol{D} = (P, \mathcal{L})$	A vector that consists of the partition result $P = (\mathcal{S}_{i_1}, \ldots, \mathcal{S}_{i_k})$ and the corresponding WuC locations $\mathcal{L} = ((x_1, y_1), \ldots, (x_k, y_k))$
$\boldsymbol{D^*} = (P^*, \mathcal{L}^*)$	The optimal partition among k partitions
c	Center of a deployment region
V_R	Set of devices that have not been assigned to any group
R	Abbreviation of UAV radius used to elaborate the benefit of I2
R'	Adjusted UAV radius considering micro mobility within disc r, as $R' = R - r$
$\ell_j = (x_j, y_j)$	WuC coordinates for cluster $\mathcal{S}_{i_j}$ where $1 \leq j \leq k$
$\ell_j(t) = (x_j(t), y_j(t))$	WuC coordinates for cluster $\mathcal{S}_{i_j}$ where $1 \leq j \leq k$, t is the time elapsed, and $\ell_j(0) = \ell_j$
u	A device
$\ell'_j(t)$	WuC coordinates where device u can receive another WuC before the WuC from its designated $\ell_j(t)$ is sent
v	Device moving speed for group mobility
θ	Angle of group mobility w.r.t. the positive direction of the x-axis
p and q	Positive integers
Len	Side length of a deployment region which is square
V_{UAV}	Flying speed of the UAV that is performing a data collection mission
P_{rd}	Access success probability of the reference device (RD)
P_s^{sf}	Access success probability of the RD during an SF
P_s^{rf}	Access success probability of the RD during an RF
T_{rd}	Access delay of the RD
T_{wuc}	WuC transmission and decoding duration
T_s	Length of a time slot
P_i^{sf}	Access success probability of the RD during the i-th slot of an SF
P_j^{rf}	Access success probability of the RD during the j-th slot of an RF
P_k^F	Failed transmission probability of the RD during the k-th slot of an RF
E_{rd}	Energy consumption of the RD
E_{wuc}	Energy consumption of the RD for receiving a WuC
P_{ls}	Main radio power consumption level for light sleep
P_{ds}	Main radio power consumption level for deep sleep
P_a	Main radio power consumption level for active

APPENDIX B
PYTHON IMPLEMENTATION OF FIND_CENTER

The arguments of the function find_center are a list (or an ndarray) of the coordinates of the devices and the UAV radius. It returns the coordinates of the center of these devices. As

an example, consider a network with five devices located at coordinates (295,255); (270,260); (225,180); (205,110); and (260,90) respectively and a UAV with coverage radius of 100 meters hovering above it. The sample input and the function call are shown below. Executing find_center on Google Colab results in an outcome of [251.0, 179.0], i.e., the obtained coordinates of the center are (251.0, 179.0).

```
import numpy as np
from scipy.optimize import minimize

def find_center(V, R):
  def obj(x):
    return 1
  def constraint_gen(v):
    def constraint_of_v(x): # constraint for v
      return R**2 - (x[0]-v[0])**2 - (x[1]-v[1])**2
    return constraint_of_v
  x0 = np.mean(V, axis=0) # initial center
  con = []
  for v in V:
    con.append({'type':'ineq', 'fun':constraint_gen(v)})
  sol = minimize(obj, x0, method='SLSQP', constraints=con)
  x = sol.x
  return [float(x[0]), float(x[1])]

# sample input, function call, and result
V = [[295,255],[270,260],[225,180],[205,110],[260,90]]
R = 100
center = find_center(V, R)
print(center)
```