View metadata, citation and similar papers at core.ac.uk
brought to you by CORE
provided by University of East Anglia digital repository

Multi-Slot Allocation Protocols for Massive IoT Devices with Small-Size Uploading Data

Tsung-Yen Chan, Yi Ren, Yu-Chee Tseng, and Jyh-Cheng Chen

***Abstract*—The emergence of Internet of Things (IoT) applications introduces new challenges such as massive connectivity and small data transmission. In traditional data transmission protocols, an ID (i.e., IP address or MAC address) is usually included in a packet so that its receiver is able to know who sent the packet. However, this introduces the *big head-small body* problem for light payload. To address this problem, the Hint protocols have been proposed. The main idea is to "encode" information in a tiny broadcast Hint message that allows devices to "decode" their transmission slots. Thus, it can significantly reduce transmission and contention overheads. In this paper, we extend eHint to support multi-slot data transmissions. Several efficient protocols are proposed. Our simulation results validate that the protocols can significantly increase the number of successfully transmitted devices, channel utilization, and payload of transmitted devices compared with eHint.**

***Index Terms*—Internet of Things (IoT), Machine-to-Machine (M2M) Communication, Multi-Slot Allocation, Random Access, Wireless Networks.**

I. Introduction

INTERNET of Things (IoT) traffic characterized by small data transmission introduces new challenges to research community. There are various types of IoT devices ranging from small tags, sensors, to complicated actuators and machines [1]. Statistics show that 50% of IoT packets are less than 100 bytes [2]. Collecting *small* data from such *massively* connected IoT devices introduces new challenges. In most protocol designs, an ID (e.g., IP address or MAC address) is included in a packet so that its receiver is able to identify the sender. When IoT data is small, however, the overhead of its ID becomes relatively large, leading to the *big head-small body* problem.

Connection-oriented network architectures have been studied in [3]–[6]. When a device intends to send data to the Base Station (BS), a connection is established by using the Random Access (RA) procedure. As the number of devices increases, it may cause significant collision in RA [7], leading to a long delay. To solve the problem, several solutions have been proposed, including Access Class Barring (ACB) [3], dynamic resource allocation [4], slotted access [5], deep learning scheduling [6]. The idea of ACB is to control the number of User Equipment (UE) terminals intending to join RA. The BS broadcasts a parameter (e.g., probability) such that UEs can perform RA probabilistically. In dynamic resource allocation, the BS can dynamically allocate resources of RA channel and data channel. It also derives an optimal trade-off problem to maximize the Machine-to-Machine (M2M) throughput. In slotted access, each Machine-Type Communication (MTC) device is assigned a dedicated RA slot to access RA. The dilemma is that short RA cycles may lead to collision while long RA cycles may lead to long delay.

To address the *big head-small body* problem, a series of Hint protocols [8]–[10], which remove IDs from data packets, have been proposed. In these protocols, a tiny Hint message is broadcast to allow IoT devices to decode their assigned transmission resources. Interestingly, the assigned location of these resources also imply the sender's ID, thus eliminating a large part of packet header. Therefore, even the device's ID is not transmitted, its receiver (usually a BS) is still able to know its identity. In [8], we proposed a set of Hint-based frameworks for small data transmission. Later, a Chinese remainder theorem based Hint protocol [9] is applied to LTE-A networks to reduce the signaling cost in random access procedures. However, the both Hint protocols [8], [9] are based on a strong assumption that the small data has the same size and is carried by the same size channel resource, i.e., a time slot. To release the assumption, the Hint protocols are then enhanced by supporting multi-slot data transmission in [10]. However, when the number of transmitting devices is large, the multi-slot Hint protocol requires intensive computation capacity to find a satisfactory seed, which may lead to transmission latency. To address this issue, in this paper we use a novel iterative approach to reduce computation overhead and enable more devices to transmit. Through extensive simulations, we demonstrate that the proposed iterative approach significantly outperforms eHint [10] in terms of the number of successfully transmitted devices, channel utilization, and the total payload of transmitted devices.

II. Multi-Slot Allocation Problem

We consider a set $D = \{d_1, d_2, \cdots, d_m\}$ of m IoT devices covered by a BS. Each IoT device $d_i, i = 1..m$, needs to report its data at a regular pattern to the BS. Our goal is to allocate radio resources to D to transmit data to the BS. We make the following assumptions:

1) The value of m is quite large.
2) Each d_i switches between two modes, *active* and *sleep*. When intending to transmit data, d_i goes to the active mode; otherwise, it switches to the sleep mode.
3) The active pattern of d_i is denoted by a binary periodical function $P_i(t)$, where t is time (by frame count). $P_i(t) = 1$ if d_i intends to transmit data at the t-th frame; otherwise, $P_i(t) = 0$. For example, if d_i is attached to a temperature sensor which needs to report in every 3

T.-Y. Chan, Y.-C. Tseng, and J.-C. Chen are with the Department of Computer Science, College of Computer Science, National Chiao Tung University, Taiwan. E-mail: {tychan, yctseng, jcc}@cs.nctu.edu.tw.

Y. Ren is with the School of Computing Science, University of East Anglia (UEA), Norwich, UK. He was with the Department of Computer Science, National Chiao Tung University, Hsinchu, Taiwan. E-mail: e.ren@uea.ac.uk.

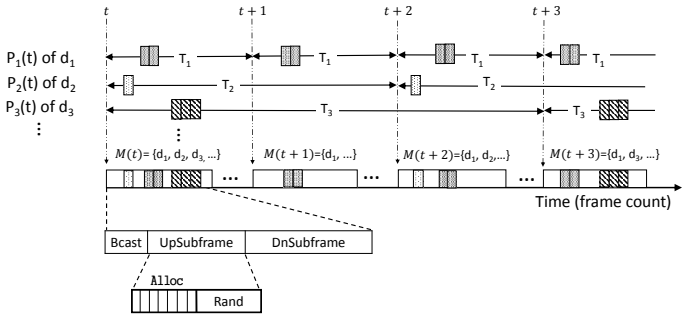

Fig. 1: The proposed frame structure.

minutes, $P_i(t)$ has a period of 3 minutes. If one more humidity sensor which needs to report in every 5 minutes is attached to d_i, $P_i(t)$ is a function of the combination of two periodical functions with periods = 3 and 5 minutes, respectively.

4) Whenever $P_i(t) = 1$, device d_i needs to send n_i slots of data to the BS at t. Note that n_i is also quite small (such as less than 3 or 5 slots).
5) When our Hint protocol starts, the BS is informed of its $P_i(t)$ and n_i, for each $d_i \in D$.

To solve the multi-slot allocation problem, the communication channel is divided into a sequence of fixed-length *frames*. Each frame consists of three parts: (1) Broadcast (`Bcast`): It is for the BS to broadcast and announce resource allocation information (i.e., Hint) to devices. (2) Allocated (`Alloc`): It consists of multiple slots for uplink data transmission (no transmitter's ID required in our case). (3) Random (`Rand`): It is for any unscheduled/unpredicted transmission not arranged in `Alloc`. Our goal is to design an efficient access protocol for such transmission.

Fig. 1 shows the frame structure. In this example, the active periods of d_1, d_2, and d_3 are T_1, T_2, and T_3, respectively. Their requirements are $n_1 = 2$, $n_2 = 1$, and $n_3 = 3$ slots, respectively. At frame t, all devices will transmit. At $t + 1$, only d_1 will transmit. At t+2, d_1 and d_2 will transmit. The BS will schedule their transmissions in `Alloc`, through the announcement in `Bcast`. Details will be discussed later. For exceptions (such as transmission errors or emergency traffics), devices can use `Rand`.

III. Proposed Protocols

The main idea of the Hint protocol is to arrange a common function shared by the BS and all devices. The function takes two inputs: (i) a small piece of information computed by the BS, and (ii) a device's ID. The BS will broadcast the information computed in (i) by using `Bcast`. Each device then can use the broadcast information and its ID to retrieve the transmission slots allocated to it through the function. The broadcast Hint is thus more efficient than typical 1-by-1 notifications. Next, we first review the VF proposed in [10]. We then propose two more efficient protocols called 2VF and IVF.

A. Review: Virtual Frame (VF)

Let $h(s, ID)$ be a hash function which takes a device ID and a seed s as inputs. In frame t, the Hint message = $< s, v >$, where v is a vector. The length of v is $|v| \geq \sum_{d_i \in M(t)} n_i$, where $M(t) = \{d_i | P_i(t) = 1\}$. The i-th element of v (resp., `Alloc`), is denoted by $v[i]$ (resp., `Alloc`$[i]$). The value of $v[i]$ falls in $\{0, 1, 2\}$. v is for devices to decode whether they can transmit or not. If it is safe to transmit in n_i continuous slots in `Alloc`, the corresponding elements in v will be: "$1\underbrace{2\cdots2}_{n_i-1}$", where "1" means "starting slot" and "2" means "continuous slot". If it is not safe to transmit or a slot is not assigned to any device, a "0" is used.

Next, we review the VF briefly:

1) The BS randomly picks a seed s and computes $h(s, d_i)\ mod\ |v|$ for each device $d_i \in M(t)$. Next, the BS selects a subset $M' \subseteq M(t)$ of devices that can correctly decode v for safe transmission.
2) The BS executes Step 1 a few times with different seeds and then selects the s from the iteration leading to the largest number of safe transmission slots.
3) After selecting, the BS encodes v.
4) The BS broadcasts $< s, v >$ in `Bcast`.
5) Each device $d_i \in M(t)$ checks whether it is allowed to transmit or not.
6) If $d_i \in M(t)$ can transmit safely, it uploads its data at the corresponding `Alloc`. Otherwise, d_i can use random access to contend for transmission in `Rand`.

Discussion: In VF, some devices may not receive slots for transmission due to collision in v. Given a fixed v, more devices may lead to higher collisions. This not only reduces transmission opportunities in `Alloc`, but also lacks of flexibility. These shortcomings motivate us to design two new protocols, 2VF and IVF.

B. Two Virtual Frame (2VF)

To reduce computation overhead and enable more devices to transmit in `Alloc`, the 2VF protocol divides `Alloc` into two parts, `Alloc_1` and `Alloc_2`, and uses two seeds s_1 and s_2 to achieve this goal. Two vectors v_1 and v_2 will be used. Also, the length of v_1 and v_2 are set to $|v_1|$ and $|v_2|$, respectively, where $|v_1| \geq \sum_{d_i \in M(t)} n_i$ and $|v_2| \geq \sum_{d_i \in M(t)-M'} n_i$.

It works as follows:

1) The BS repeats the following steps a few times.
   - Randomly pick a seed s_1 and compute $h(s_1, d_i)\ mod\ |v_1|$ for each device $d_i \in M(t)$.
   - Select a subset $M' \subseteq M(t)$ of devices such that only devices in M' can correctly decode v_1 for safe transmission.
   - A seed s_2 is randomly picked. Compute $h(s_2, d_i)\ mod\ |v_2|$ for each device $d_i \in M(t) - M'$.
   - Select a subset $M'' \subseteq M(t) - M'$ of devices such that only devices in M'' can correctly decode v_2 for safe transmission.
2) The BS then selects the s_1 and s_2 from the iteration in Step 1, leading to the largest number of safe transmission slots. (Note: once s_1 and s_2 are selected, the corresponding M' and M'' are also selected.)
3) v_1 and v_2 are encoded as follows:
   - (Collision-free) For each $d_i \in M'$ such that $k = h(s_1, d_i)\ mod\ |v_1|$, set $v_1[k]$ = "1" and $v_1[k + 1$:

$k + n_i - 1] =$ "$\underbrace{2 \cdots 2}_{n_i - 1}$". For each $d_i \in M''$ such that $x = h(s_2, d_i)\ mod\ |v_2|$, set $v_2[x] =$ "1" and $v_2[x+1 : x+n_i-1] =$ "$\underbrace{2 \cdots 2}_{n_i - 1}$".

- (Empty/Collision) The rest of elements of v_1 and v_2 are all set to "0".

4) The BS broadcasts $< s_1, v_1, s_2, v_2 >$ in `Bcast` to all devices.
5) For each device $d_i \in M(t)$, receiving $< s_1, v_1, s_2, v_2 >$ in `Bcast`, d_i computes $k = h(s_1, d_i)\ mod\ |v_1|$ and $x = h(s_2, d_i)\ mod\ |v_2|$. To check whether d_i is allowed to transmit or not, there are two cases:
   - Case of $n_i = 1$: If $v_1[k] =$"1" and $v_1[k+1] =$"0" or "1" , d_i is allowed to transmit; otherwise, it checks v_2. If $v_2[x] =$"1" and $v_2[x+1] =$"0" or "1" , d_i is allowed to transmit; otherwise, d_i cannot transmit.
   - Case of $n_i \geq 2$: If $v_1[k] =$"1" , $v_1[k+1 : k+n_i-1] =$"$\underbrace{2 \cdots 2}_{n_i - 1}$" , and $v_1[k+n_i] =$"0" or "1" , d_i is allowed to transmit; otherwise, it checks v_2. If $v_2[x] =$"1" , $v_2[x+1 : x+n_i-1] =$"$\underbrace{2 \cdots 2}_{n_i - 1}$" , and $v_2[x+n_i] =$"0" or "1" , d_i is allowed to transmit; otherwise, it cannot transmit.
6) For each device $d_i \in M(t)$ that is allowed to transmit safely in v_1, d_i uploads its data at `Alloc`$[j : j+n_i-1]$, where j is the number of 1's and 2's in $v_1[0 : k-1]$ and $k = h(s_1, d_i)\ mod\ |v_1|$. If d_i is not allowed to transmit in v_1, but is allowed to transmit in v_2, d_i uploads its data at `Alloc`$[z : z+n_i-1]$, where z is the number of 1's and 2's in v_1 and in $v_2[0 : x-1]$ and $x = h(s_2, d_i)\ mod\ |v_2|$. If d_i is not allowed to transmit in v_1 and v_2, it may consider to contend for transmission in `Rand`.

Fig. 2(a) shows the message flow of 2VF. As shown in Fig. 2(b), there are 7 devices intending to transmit in frame t. Hence, $|v_1| = n_1 + n_2 + n_3 + n_4 + n_5 + n_6 + n_7 = 13$. Suppose that devices d_1, d_4, and d_7 are selected to transmit in v_1 (i.e., $M' = \{d_1, d_4, d_7\}$). Then $v_1 =$"0122201201200" . Although d_2, d_3, d_5, and d_6 are not allowed to transmit in v_1, they can perform VF again. Hence, $|v_2|$ is set to $n_2 + n_3 + n_5 + n_6 = 5$. After hashing, d_5's slot and d_2's second slot are overlapped. Suppose that d_2, d_3, and d_6 are allowed to transmit in v_2 (i.e., $M'' = \{d_2, d_3, d_6\}$). Then $v_2 =$"10121" . d_6 will transmit in `Alloc`[8] because there are eight numbers of 1's or 2's in v_1. Also, d_2 will transmit in `Alloc`[9 : 10] because there are nine numbers of 1's or 2's in v_1 and $v_2[0 : 1]$. Similarly, d_3 will transmit in `Alloc`[11] because there are eleven numbers of 1's or 2's in v_1 and $v_2[0 : 3]$. Since device d_5 finds $v_2[3] =$"2" , it knows that it cannot transmit.

C. Iterative-Virtual-Frame (IVF)

2VF enables most of the devices to transmit data in `Alloc` by running VF twice. However, as the number of IoT devices becomes larger, the difficulty of finding satisfactory seeds also grows accordingly. Also, the computation power of the BS has

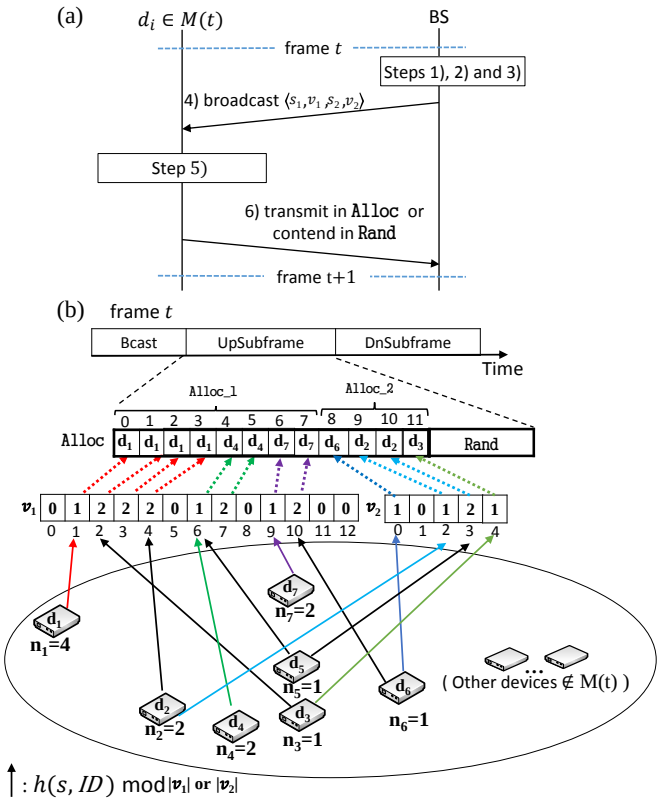

Fig. 2: (a) The message flow of the 2VF. (b) An example of 2VF.

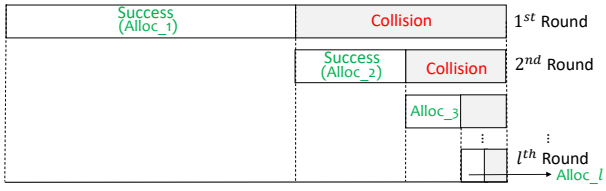

Fig. 3: `Alloc` vector of IVF.

its limitation. There are two main concepts in IVF (refer to Fig. 3):

- Iterative: The VF process is executed l times. The k-th iteration takes the devices unable to transmit in the previous iterations and tries to find a seed s_k, where $k = 1, 2, ..., l$.
- Time-bound: A timer T_k is set for iteration k such that the search for s_k would stop even if a satisfactory s_k cannot be found after this time bound. In this case, the best s_k so far will be used. Note that this only sacrifices the transmission ratio, but our protocol still works correctly.

We summarize how IVF works as follows:

1) The BS computes `Alloc`$\_k$ for $k = 1, 2, \cdots, l$ as follows. In iteration k, we define the length of vector v_k as

$$|v_k| \geq \begin{cases} \sum_{d_i \in M(t)} n_i & \text{if } k = 1 \\ \sum_{d_i \in M(t) - \bigcup_{\hat{k}=1}^{k-1} M_{\hat{k}}} n_i & \text{if } k = 2, 3, ..., l \end{cases}, \quad (1)$$

where $M_{\hat{k}}$ is the set of devices that can transmit in `Alloc`$\_\hat{k}$ in iteration $\hat{k}$. The BS repeatedly chooses a

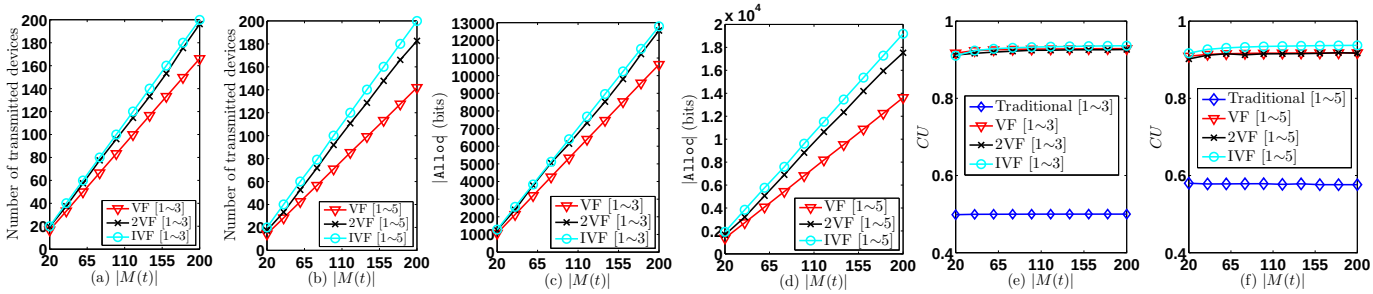

Fig. 4: The different effects of the number of successfully transmitted devices, |Alloc|, and CU with different parameters.

seed s_k and computes v_k until (i) a satisfactory seed s_k is found, or (ii) the time bound T_k spent on the search has reached. In the later case, the best seed s_k so far and the corresponding v_k is selected. After these l iterations, the BS broadcasts $\langle \{s_k, v_k\}_{k=1}^{l} \rangle$ in Bcast.

2) Upon receiving Bcast, a device d_i with $P_i(t) = 1$ can transmit in one of the l spaces (Alloc_1, Alloc_2, $\cdots$, Alloc_l) according to its hash results $h(s_k, d_i)$ and vectors v_k. If it is not allowed to transmit in Alloc, it can contend in Rand.

IV. Performance Evaluation

We evaluate 2VF and IVF through simulations in terms of three metrics, the number of successfully transmitted devices in $M(t)$, the length of Alloc (i.e., |Alloc|), and Channel Utilization (CU). The results are compared with VF and traditional polling protocol which collects data from each device one-by-one. Each slot size is assumed to be 32 bits and the length of s is set to 16 bits. Device address is assumed to be 64 bits. The value of n_i is small and it randomly falls in the range of $[1 \sim 3]$ or $[1 \sim 5]$ slots.

The CU is defined as: $CU = \frac{\text{payload}}{\text{payload+packet header}}$. For 2VF and IVF, $CU = \frac{|\texttt{Alloc}|}{|\texttt{Alloc}| + 2\times16+2\times(|v_1|+|v_2|)}$, and $CU = \frac{|\texttt{Alloc}|}{|\texttt{Alloc}|+l\times16+2\times(|v_1|+|v_2|+\ldots+|v_l|)}$, respectively. For VF, $CU = \frac{|\texttt{Alloc}|}{|\texttt{Alloc}|+16+2\times|v|}$. For traditional polling protocols, $CU = \frac{|\texttt{Alloc}|}{|\texttt{Alloc}|+64\times|M(t)|}$. For IVF, we execute VF until all the devices are allocated in Alloc successfully.

Fig. 4(a) and Fig. 4(b) demonstrate the number of successfully transmitted devices in $M(t)$ with $n_i = [1 \sim 3]$ and $[1 \sim 5]$, respectively. Both the figures show that IVF and 2VF outperform VF significantly. Considering Fig. 4(b) as an example, when $|M(t)| = 200$, IVF and 2VF gain 44% and 30% transmitted devices compared with VF, respectively.

Fig. 4(c) and Fig. 4(d) demonstrate the impacts of $|M(t)|$ on |Alloc| by varying $n_i = [1 \sim 3]$ and $[1 \sim 5]$, respectively. We observe that both 2VF and IVF outperform VF with large margins, and IVF performs the best. Specifically, |Alloc| increases as the number of devices intending to transmit grows. We also observe that the three curves are linear. This means that the proposed protocols are scalable and resilient to the increasing of $|M(t)|$.

Fig. 4(e) and Fig. 4(f) show the impacts of $|M(t)|$ on CU when $n_i = [1 \sim 3]$ and $[1 \sim 5]$, respectively. Overall, the Hint protocols (i.e., VF, 2VF, and IVF) outperform the traditional protocol significantly since we use a Hint message instead of notifying devices by $64 \times |M(t)|$ times. The CU of the traditional scheme is around 0.5 and 0.6 in Fig. 4(e) and Fig. 4(f), respectively, when $|M(t)|$ increases. The traditional protocol with $[1 \sim 3]$ has higher CU than $[1 \sim 5]$ since $[1 \sim 5]$ has more payload to transmit. The values of CU gradually converge for VF, 2VF, and IVF with the increase of $|M(t)|$.

Overall, Figs. 4(a)-(f) demonstrate that the iteration based Hint protocol has much better performance compared with [10] in terms of the number of successfully transmitted devices, the total payload of transmitted devices, and channel utilization.

V. Conclusions

In this paper, we propose two advanced Hint protocols, 2VF and IVF, which enable more devices to upload their data in a collision-free manner. Compared to the eHint [10], the proposed 2VF and IVF provide more flexibility and address the issue when the number of devices intending to transmit is large. We demonstrate, through extensive simulations, that the proposed 2VF and IVF outperform both eHint [10] and traditional polling protocol in terms of CU and slot allocation capability. As to future work, designing a *smart* seed with low computation cost is worth of exploring.

References

[1] H. Li, K. Ota, and M. Dong, "Energy cooperation in battery-free wireless communications with radio frequency energy harvesting," *ACM Trans. on Embedded Comput. Syst.*, vol. 17, no. 2, p. 44, 2018.

[2] W. John and S. Tafvelin, "Analysis of Internet backbone traffic and header anomalies observed," in *Proc. ACM SIGCOMM Conf. on Internet Measurement*, 2007.

[3] Z. Wang and V. WS, "Optimal access class barring for stationary machine type communication devices with timing advance information," *IEEE Trans. on Wirel. Commun.*, vol. 14, no. 10, pp. 5374–5387, 2015.

[4] D. T. Wiriaatmadja and K. W. Choi, "Hybrid random access and data transmission protocol for machine-to-machine communications in cellular networks," *IEEE Trans. Wirel. Commun.*, vol. 14, no. 1, pp. 33–46, 2015.

[5] 3GPP TR 37.868, *Study on RAN improvements for machine-type communications, (Release 11)*, 3GPP Std., Sep. 2011.

[6] H. Li, K. Ota, and M. Dong, "Learning IoT in edge: Deep learning for the Internet of things with edge computing," *IEEE Network*, vol. 32, no. 1, pp. 96–101, 2018.

[7] 3GPP TR 38.912, *Study on new radio (NR) access technology*, 3GPP Std., 2017.

[8] Y. Ren, R.-J. Wu, T.-W. Huang, and Y.-C. Tseng, "Give me a hint: An ID-free small data transmission protocol for dense IoT devices," in *Proc. IEEE Wireless Days*, 2017.

[9] T.-W. Huang, Y. Ren, K. C.-J. Lin, and Y.-C. Tseng, "r-Hint: A message-efficient random access response for mMTC in 5G networks," in *Proc. IEEE PIMRC*, 2017.

[10] T.-Y. Chan, Y. Ren, Y.-C. Tseng, and J.-C. Chen, "eHint: An efficient protocol for uploading small-size IoT data," in *Proc. IEEE WCNC*, 2017.