A Data-Emergency-Aware Scheduling Scheme for Internet of Things in Smart Cities

Tie Qiu, *Senior Member, IEEE*, Kaiyu Zheng, *Student Member, IEEE*, Min Han, *Senior Member, IEEE*, C. L. Philip Chen, *Fellow, IEEE*, and Meiling Xu

***Abstract*—With the applications of Internet of Things (IoT) for smart cities, the real-time performance for a large number of network packets is facing serious challenge. Thus, how to improve the emergency response has become a critical issue. However, traditional packet scheduling algorithms cannot meet the requirements of the large-scale IoT system for smart cities. To address this shortcoming, this paper proposes EARS, an efficient data-emergency-aware packet scheduling scheme for smart cities. EARS describes the packet emergency information with the packet priority and deadline. Each source node informs the destination node of the packet emergency information before sending the packets. The destination node determines the packet scheduling sequence and processing sequence according to emergency information. Moreover, this paper compares EARS with a first-come, first-served, multilevel queue algorithm and a dynamic multilevel priority packet scheduling algorithm. Simulation results show that EARS outperforms these previous scheduling algorithms in terms of packet loss rate, average packet waiting time, and average packet end-to-end delay.**

***Index Terms*—Data emergency aware, Internet of Things (IoT), packet scheduling, priority, smart cities.**

I. Introduction

THE Internet of Things (IoT) for smart cities [1]–[3] is a large integrated system that consists of sensor networks, wireless mesh networks, mobile communications, social networks, smart transportation, and intelligent transportation. It has become a hot topic in many applications at present [4]. In most of the existing works in IoT for smart cities, the researchers mainly focused on the improvement of the routing algorithm [5], [6] and node energy saving management [7]–[11], to improve the network performance. However, with the expansion of network scale and applications, the data in network become so large or complex that traditional data scheduling schemes are inadequate to deal with them. In addition, there are a lot of data packets that need to be sent to the sink node as soon as possible, called emergency packets, such as the alarm information in medical rescue service, the fire information in forest fire monitoring service, etc. Thus, data packets emergency response has become a serious challenge, especially in the disaster recovery system [12]. Lots of packet scheduling algorithms for the large-scale networks with big data are proposed to reduce the end-to-end packet transmission delays and make the emergency packets reach the sink node before the expiration of deadline [13]–[15].

In recent years, the research works for data packet scheduling are mainly divided into three types: first-come, first-served (FCFS), earliest deadline first (EDF), and emergency task first rate monotonic. Here, FCFS is widely used. This algorithm is generally used in multilevel priority queues (PQs) [16]. Some other studies about packet scheduling in sensor networks [17], [18] mainly concentrate on single node's data packet scheduling, such as the packet with the higher priority can preempt the packet with the lower priority. These algorithms do not consider the impact between different nodes, so there will be some extra time when the emergency data packets are sent to the sink node. Thus, we need an efficient packet scheduling algorithm that can allocate network resources rationally. The packets can be scheduled based on their emergency information, which can ensure timeliness of emergency data packets and the effectiveness of nonemergency data packets. The classical QoS approach, such as differentiated services (diffserv), adopts this idea. It assigns priorities according to the type of the sensor and characteristics of the data in networks, as well as the time constraints of transmitted packets. However, this method is a coarse-grained, class-based mechanism for network traffic management. In the packet scheduling scheme for smart cities, along with the important information, such as the alarm information in fire monitoring service, some more fine-grained data should also be provided.

In this paper, we propose an efficient emergency-aware packet scheduling algorithm for smart cities, which is called EARS. Our major contributions are as follows.

Manuscript received August 22, 2017; accepted October 9, 2017. Date of publication October 16, 2017; date of current version May 2, 2018. This work was supported in part by the National Natural Science Foundation of China under Grant 61374154 and Grant 61672131 and in part by the Fundamental Research Funds for the Central Universities under Grant DUT16QY27 and Grant DUT17ZD216. Paper no. TII-17-1906. *(Corresponding author: Tie Qiu.)*

T. Qiu is with the School of Software, Dalian University of Technology, Dalian 116620, China; he is also with the School of Computer Science and Technology, Tianjin University, Tianjin 300072, China (e-mail: qiutie@ieee.org).

K. Zheng is with the School of Software, Dalian University of Technology, Dalian 116620, China (e-mail: kaiyuzheng@mail.dlut.edu.cn).

M. Han and M. Xu are with the Faculty of Electronic Information and Electrical Engineering, Dalian University of Technology, Dalian 116023, China (e-mail: minhan@dlut.edu.cn; xuml@dlut.edu.cn).

C. L. P. Chen is with the Faculty of Science and Technology, University of Macau, Macau (e-mail: Philip.Chen@ieee.org).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TII.2017.2763971

1) When multiple nodes send data packets to the same destination node at the same time, we use the data-emergency-aware mechanism to deliver and deal with emergency data packet first. Therefore, the timeliness of the emergency data packet can be ensured.
2) In EARS, time slots are strictly controlled, so that all the processing modules collaborate to manage the data packets. The network performance can be improved by supporting the diffserv to the relevant packets based on the packet emergency information.
3) We evaluate EARS in end-to-end delay, packet loss rate, and waiting time of data packet through simulation experiments. Meanwhile, we compare them with previous related works and obtain a better performance.

The rest of this paper is organized as follows. In Section II, we discuss some related works and give our problem statement. Section III describes the implementation of the EARS and provides a real-time performance analysis of the EARS. Section IV evaluates the performance of the EARS algorithm through simulations and compares EARS with an FCFS, multilevel queue algorithm [17] and a dynamic multilevel priority packet scheduling algorithm (DMP) [18]. Finally, we conclude this paper in Section V.

II. Related Work and Problem Statement

A. Related Work

For the existing packet scheduling algorithms, there are three types: deadline-based scheduling, priority-based scheduling, and packet-type-based scheduling.

1) Deadline-Based Scheduling: In this type of packet scheduling, the deadlines of packets are considered. In the RMS algorithm [19], the priorities of packets are static priorities, which are assigned based on the deadlines of packets. The priorities of the data packets for RMS are static. Thus, the RMS scheme is limited and inapplicable. In the EDF algorithm [20], the priority of each packet is determined according to the deadline. Thus, the overhead is relatively large.

2) Priority-Based Scheduling: The priority-based scheduling algorithms include two types: preemptive scheduling and nonpreemptive scheduling [21]. In the nonpreemptive scheduling algorithm, a packet once started to be processed, other packets, even though the priorities are higher, have to wait until the packet processing is completed. Differently, preemptive scheduling algorithm is more flexible. At each node, the packets with higher priorities can preempt packets with lower priorities. Buttazzo *et al.* [22] evaluate preemptive scheduling and nonpreemptive scheduling through a lot of experiments.

3) Packet-Type-Based Scheduling: There are different types of packets in this scheduling scheme, such as real-time packets and non-real-time packets. The priorities of the packets are determined based on the packet types. Chennakesavula *et al.* [23] propose an effective real-time packet scheduling policy (ERTS). In ERTS, each intermediate node determines the forwarding order of data packets by analyzing the packet deadline and the distance from the destination node. Lee *et al.* [17] propose a multilevel queue scheduler scheme. The different number of queues is used based on the location of the sensor nodes. In [18], Nidal *et al.* propose the DMP algorithm. A Time Division Multiple Address (TDMA) method is used in this algorithm. This algorithm can effectively distinguish different types of packets, but it cannot remove the data packets with shorter deadline at intermediate node, which results in a waste of network resources.

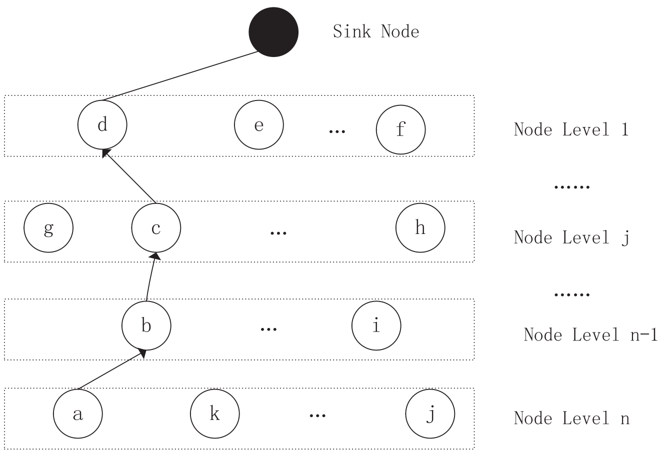


Fig. 1. Example of sending an emergency packet.

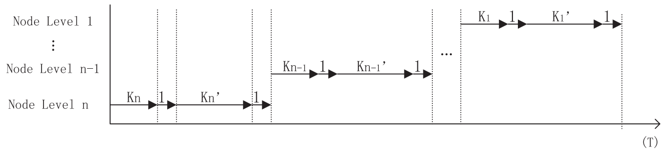


Fig. 2. Timing diagram of sending an emergency packet.

B. Problem Statement

In [16] and [18], the packets from different sending nodes compete against each other for the shared network resources when multiple nodes send data packets to the same destination node. Moreover, it can cause a serious network congestion, even result in breakdown of the network.

The DMP algorithm adopts a hierarchical structure to organize all the nodes in the network [18]. Different nodes can be assigned a reasonable time slot by TDMA. The DMP algorithm is efficient to avoid the packets collision in the Media Access Control (MAC) layer. However, in practice, we find that the real-time performance of the DMP algorithm still needs to be improved. For example, in Fig. 1, the sink node is located at the root node. The nodes are considered to be at Node Level 1, Node Level 2, ... , Node Level n based on the number of hops from the sink node. When a data packet with the highest emergency is generated at node a, it is then forwarded to the sink node through relay node b, node c, and node d. We assume that during the process of packets transmission, the packet needs to spend two time slots to arrive at the next hop. Fig. 2 shows the timing diagram of sending an emergency packet in the DMP algorithm. The process of forwarding data packet is measured in time slots T. D_{e2e} is the end-to-end delay using the DMP algorithm. k_j denotes the number of slots waiting for their own time slots in the TDMA method at Node Level j. k_j' is the number of users to share the same communication channel. In Fig. 2, Node Level 1 corresponding to the horizontal axis is the

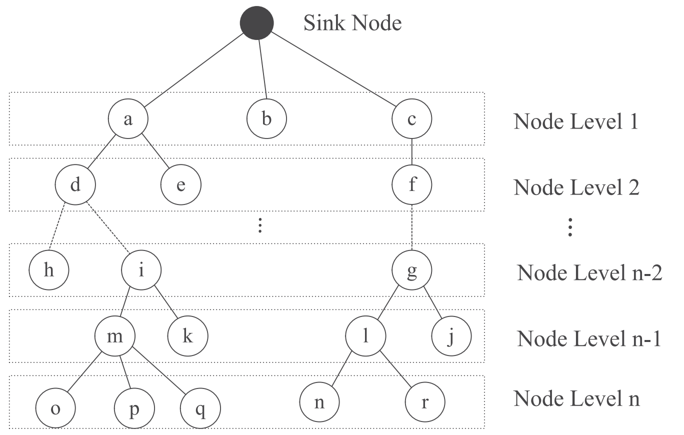
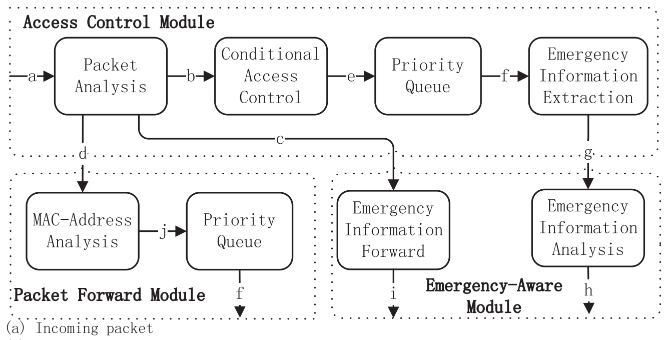


Fig. 3. Network topology.



(a) Incoming packet
(b) Received data packet or local data packet
(c) Emergency information packet from children nodes
(d) MAC-address packet
(e) Admitted packet
(f) The packet with the highest emergency
(g) Local emergency information packet
(h) MAC-address packet
(i) The emergency information packet that uses TDMA to send
(j) The certain result about the MAC-address

Fig. 4. Structure of EARS.

time at which data packets reach the sink node. D_{e2e} is given by

$$D_{e2e} = \sum_{j=1}^{n} (k_j + k'_j + 2)T. \tag{1}$$

Thus, we try to explore a strategy that enables the destination node to know the emergency information about the source nodes' data packets, then the packet with the highest emergency can be sent first. So, the fastest time at which the data packets arrive at the sink nodes can be as close as possible to $2nT$. It is obvious that the difference value compared to the optimal value is quite large, so there is a great room for improvement in sending emergency data packets in the DMP algorithm. Additionally, the TDMA time slot T increases when a packet length is increased. Hence, the emergency data packets will have a sharper decline in timeliness because T is used as a multiplication factor. It shows that the DMP algorithm is not suitable for smart cities with the long packets. Moreover, there are two points that can be as the enhancements. 1). Assigning packet priority based on packet deadline instead of the shortest packet processing time. 2). Removing the packets with expired deadlines to achieve the goal of reducing processing and transmitting overhead as well as improving bandwidth utilization.

In order to solve these above-mentioned problems, we propose an data-emergency-aware packet scheduling algorithm for smart cities.

III. Scheduling Model

The nodes in network are organized in a tree-based hierarchical structure, as shown in Fig. 3. There are not mobile nodes in our network and the network topology maintenance is carried out periodically. The dotted line means the node at the upper level connects the node at the lower level crossing many levels. Every data packet needs to be sent to the sink node. The intermediate nodes can generate and forward the data packets. Each node has its own ID number while recording the ID numbers and MAC addresses of the neighbor nodes. If the node in a network completes all of its tasks, it can switch into sleeping mode to save energy consumption. During the process of data packet transmission, we use a multichannel MAC protocol to send the packets simultaneously if there are different branches' nodes sending data packets at the same time. In Fig. 3, node a, node b, and node c constitute a branch, and, similarly, node o, node p, and node q constitute a different branch.

In our proposed algorithm, the data packets are classified into various types based on their priorities, which should be defined according to application scenario before our scheduling scheme is used. In order to introduce our scheme more clearly and concisely, we select a specific situation that includes three types of data packets as an classical example, such as fire monitoring service including the alarm information, network information, and sensor information, to introduce our algorithm in this paper.

1) Emergency data packets (pr_1). These packets need to be sent to the sink node as fast as possible, so they can preempt the packets with lower priorities that are being processed (e.g., the alarm information in fire monitoring service).
2) General data packets (pr_2). The rate of this kind of packets is the highest in the network. They can be preempted by emergency data packets (e.g., the network information in fire monitoring service).
3) Nonemergency data packets (pr_3). Their deadlines are longer than the deadlines of the other packets, so they have the lowest priority (e.g., the sensor information in fire monitoring service).

The proposed EARS algorithm consists of three modules: access control module (ACM), emergency-aware module (EAM), and packet forward dealt module (PFM). Fig. 4 shows the structure of EARS.

A. Access Control Module

The function of ACM is to distinguish the incoming packets, and then process packets based on their types.

Except for three kinds of data packets with different priorities, the incoming packets also include emergency information packets and MAC-address packets. Packet analysis (PA) is used to distinguish these packets. Then, PA sends data packets to conditional access control (CAC) for further processing. The emergency information packets are sent to EAM and MAC-address packets need to be analyzed in PFM. Meanwhile, CAC will check the data packet deadline. The valid packets will be placed into the PQ.

There are three-level PQs at each node: priority queue A (PQA), priority queue B (PQB), and priority queue C (PQC). The emergency data packets are put into the PQA, which has the highest priority. The general data packets are put into the PQB, which has the second highest priority. Finally, nonemergency data packets are put into the PQC, which has the lowest priority. In the PQ, packets are sorted based on their deadlines, that is, the packet with the shortest deadline is placed at the front of PQ.

We select the packet with the highest emergency at the node to extract its emergency information. Then, the emergency information packet is sent to EAM. Each emergency information packet includes packet header, packet priority, and packet deadline. The packet header length will be different based on the different communication protocols. But it always includes the source MAC address and the destination MAC address. The packet priority and the packet deadline, both of length 4 B, describe the packet emergency information.

B. Emergency-Aware Module

EAM is composed of two phases: One is the emergency information forward (EIF), the other one is emergency information analysis (EIA). EIF sends the emergency information packet received from local node's ACM and the sibling nodes' ACMs to the destination node in the TDMA method. It has to be noted that the request for emergency information packets will be terminated, when there are not packets that need to be forwarded at the sibling nodes. Then, the current node will monopolize the channel to send the packet. Taking the scenario shown in Fig. 3 as an example, when the node o, node p, and node q send packets to node m at the same time, node o is assigned time slot 1, node p is assigned time slot 2, and node q is assigned time slot 3. The second phase EIA can analyze the emergency information packets from the child nodes. Then, it gets a MAC-address packet of a child node, at which the packet with the highest emergency among the all child nodes' packets is waiting for being sent to the father node. Finally, it broadcasts the MAC-address packet to all child nodes. In the EARS algorithm, the way to analyze the emergency information packets is that the higher the priority is, the higher the emergency of the packet will be. When the priorities are same, the packets deadlines need to be compared. The packet with higher emergency will have a shorter deadline.

In EIA, there are two situations about the emergency information packets that are received by the destination node, as shown in Fig. 5. There are i nodes at Node Level L_{k-1}. They will send packets to the destination node at Node Level L_k. pr_a denotes the priority of the packet from node a. dl_a denotes the deadline of the packet from node a.

1) The packets priorities are all same, as shown in Fig. 5(a).

After receiving the emergency information packets, the destination node gets an equation $pr_a = pr_b = \cdots = pr_i$. Then,

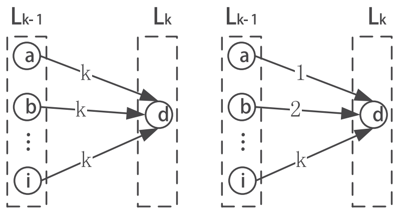


Fig. 5. Emergency-aware situations. (a) The priorities are same. (b) The priorities are different.

TABLE I
VARIABLES DEFINITION

Symbols	Description
N	The total number of packets
n	The ID of packet
PA	MAC-address packet
p_n	The ID of the packet with the highest emergency
PN_i	The number of $packets$ with pr_i
P_i	The set of pr_i packets
CSI	The channel state
$Pkts$	The receiving data packets
$Pkts'$	The data packets that should be sent
PK_e	The two-dimensional matrix of emergency information packet, in which the packet priorities are all same
PK_a	The two-dimensional matrix of emergency information packet, in which the packet priorities are different

$dl_a, dl_b, \ldots, dl_i$ need to be compared. If $dl_x < dl_y \leq \cdots \leq dl_i$, then the packet from Node$_x\,(x \leq i)$ is the packet with the highest emergency; if $dl_x = dl_y = \cdots = dl_m < dl_n \leq \cdots \leq dl_i$, then we randomly select a packet from Node$_x$, Node$_y$, ..., Node$_m\,(x, y, \ldots, m \leq i)$ as the packet with the highest emergency. Finally, we broadcast the MAC address of the node that sends the data packet with the highest emergency. Algorithm 1 analyzes the emergency data packets with the same packet priorities to get the MAC address of the source node that sends the packet with the highest emergency. All variables are presented in Table I. There is a loop in Algorithm 1 to scan array PK_e and the array length is N. The variable N determines the complexity of Algorithm 1. N is influenced by the number of source nodes and it is limited. Thus, the complexity of Algorithm 1 is $\mathcal{O}(n)$.

2) The priorities of the packets are different, as shown in Fig. 5(b).

After receiving the emergency information packets, the destination node gets the inequation $pr_x < pr_y \leq \cdots \leq pr_i$. Then, the packet from Node$_x\,(x \leq i)$ is the packet with the highest emergency. Moreover, the relationship among the all packets priorities can be $pr_x = pr_y = \cdots = pr_m \leq pr_n \leq \cdots \leq pr_i$. Then, the strategy in the first situation is called among Node$_x$, Node$_y, \ldots,$ Node$_m\,(x, y, \ldots, m \leq i)$. The details are shown in Algorithm 2. There is a loop to scan array PK_a in Algorithm 2 and the number of the emergency information packets with different priorities is numerable. When it analyzes the emergency information packets, Algorithm 1 is called and its complexity is $\mathcal{O}(n)$. Thus, the complexity of Algorithm 2 also is $\mathcal{O}(n)$.

Algorithm 1: Preparatory Work for Emergency-Aware.

```
Input: $PK_e$
Output: $PA$
 1: procedure Equal_Priority ($PK_e$)
 2:     $PA \leftarrow PK_e[0][0]$
 3:     $p_n \leftarrow 0$
 4:     for $n \leftarrow 1$ to $N$ do
 5:         if $PK_e[n][2] < PK_e[p_n][2]$ then
 6:             $PA \leftarrow PK_e[n][0]$
 7:             $p_n \leftarrow n$
 8:         else if $PK_e[n][2] == PK_e[p_n][2]$ then
 9:             Randomly select $n$ or $p_n$ as the new $p_n$
10:             $PA \leftarrow PK_e[p_n][0]$
11:         end if
12:     end for
13: end procedure
```

Algorithm 2: Emergency-Aware Process.

```
Input: $PK_a$
Output: $PA$
 1: procedure InEqual_Priority ($PK_a$)
 2:     □Upon receiving packets
 3:     for $n \leftarrow 0$ to $N$ do
 4:         if $PK_a[n][1] == 1$ then
 5:             $PN_1 \leftarrow PN_1 + 1$
 6:             Put the $n$ into the set $P_1$
 7:         else if $PK_a[n][1] == 2$ then
 8:             $PN_2 \leftarrow PN_2 + 1$
 9:             Put the $n$ into the set $P_2$
10:         else if $PK_a[n][1] == 3$ then
11:             $PN_3 \leftarrow PN_3 + 1$
12:             Put the $n$ into the set $P_3$
13:         end if
14:     end for
15:
16:     if $PN_1 == 1$ then
17:         Pick up the only element in $P_1$ as the $p_n$
18:         $PA \leftarrow PK_a[p_n][0]$
19:     else if $PN_1 > 1$ then
20:         Define a two dimensional array $PK_e[PN_1][3]$
        by allocating memory dynamically
21:         Using the elements in $P_1$ as the array $PK_a$'s row
        vector to initialize the $PK_e$
22:         Call Algorithm 1 with the $PK_e$ as the parameter
23:     else if $PN_1 == 0$ then
24:         Repeat the process of Lines 16–22 substituting
        $PN_2$ for $PN_1$
25:     else if $PN_2 == 0$ then
26:         Repeat the process of Lines 16–22 substituting
        $PN_3$ for $PN_1$
27:     end if
28: end procedure
```

Algorithm 3: Packet Scheduling and Forwarding.

```
Input: $Pkts$
Output: $Pkts'$
 1: procedure Packets_Scheduling ($Pkts$)
 2:     □Upon receiving packets
 3:     if Priority($Pkts$) == 1 then
 4:         Take $Pkts$ into $Queue_A$
 5:     else if Priority($Pkts$) == 2 then
 6:         Take $Pkts$ into $Queue_B$
 7:     else if Priority($Pkts$) == 3 then
 8:         Take $Pkts$ into $Queue_C$
 9:     end if
10:
11:     Check the $CSI$
12:     if $CSI == 0$ then
13:         Put the emergency information of the packet
    with the highest emergency into $PK_a$
14:         $PK_a$ are sent in TDMA way
15:         $CSI \leftarrow 1$
16:     else if $CSI == 1$ or $2$ then
17:         Wait until the $CSI$ changes
18:         Check the $CSI$ again
19:     end if
20:
21:     □Upon arriving at the destination
22:     Call Algorithm 2 to get the $PA$
23:     Broadcast the $PA$
24:     $CSI \leftarrow 2$
25:     The node whose MAC address is same as $PA$
    sends the $Pkts'$
26:     Destination node receives the $Pkts'$
27:     $CSI \leftarrow 0$
28: end procedure
```

C. Packet Forward Module

PFM completes the packet forwarding. The MAC-address packet from ACM is sent to MAC-address analysis. If the result of analyzing MAC-address packet is same as the local node's MAC address, the current node can take the channel alone to send the data packet with the highest emergency to the destination node. Otherwise, EIF will resend the emergency information packet.

During the process of the execution process of EARS, each packet needs to check the channel flag before it is sent to the destination node. There are three kinds of channel states: The first is idle state. There is not any node using the channel in this state. The incoming packets can be applied for using the channel in the next time slot. The second state is taking the channel to send the emergency information packet in the TDMA method. The third state is taking the channel to broadcast MAC-address packet and send the data packet with the highest emergency. In our algorithm, the channel state can be switched in order by controlling the time slot. Algorithm 3 realizes the packet scheduling and forwarding.

There is no loop in Algorithm 3. But the Algorithm 3 calls Algorithm 2 and the complexity of Algorithm 2 is $\mathcal{O}(n)$. Therefore, the complexity of Algorithm 3 is the same as the complexity of Algorithm 2. Thus, the complexity of Algorithm 3 is $\mathcal{O}(n)$.

D. Analysis of Timeliness

In order to analyze the timeliness of packets with different priorities, we define the following variables: T_a is the end-to-end delay between Node$_a$ (Node$_a$ is at Node Level L_n) and sink node, $t_{(x,y)}$ denotes the forwarding delay from Node$_x$ to Node$_y$, $t'_{(x,y)}$ is the waiting time for a packet sent from Node$_x$ to Node$_y$, t_S denotes the processing time at the sink node, and t'_S is the waiting time at the sink node. Thus, the packet end-to-end delay between Node$_a$ and sink node can be expressed as

$$T_a = t_{(a,S)} + t'_{(a,S)} + t_S + t'_S. \tag{2}$$

Here, $t_{(a,S)}$ includes

1) the time of receiving packet at each node, denoted as t_r;
2) proc(t) is the time of processing packet at each node;
3) the time of placing a packet into the medium is equal to d_s/s_t , where d_s is the packet size and s_t is the data transmission speed;
4) the propagation time, formulated as d/s_p, where $d = \sum_{i=1}^{L_n} d_i$ is the distance between current node and sink node, and s_p is the propagation speed.

Therefore, $t_{(a,S)}$ can be computed as

$$t_{(a,S)} = n * (t_r + \text{proc}(t) + \frac{d_s}{s_t}) + \frac{d}{s_p} \tag{3}$$

where t_r, s_t, and proc(t) are equal at the same node for different kinds of packets whose sizes are same. Thus, $t_{(a,S)}$ are equal for different nodes at the same node level. In the same way, the values of t_S for different packets are equal. Therefore, the end-to-end delay is mainly influenced by the waiting time at the nodes for the different kinds of packets. In the following, we formulate the waiting time of the EARS algorithm.

The packets with priority pr_1 are sorted according to the packets deadlines. We assume that $N_{i,j}$ represents the number of sending packet with pr_j form the node at Node Level L_i. te_m denotes the time from sending m emergency information packets to receiving the broadcast packets in the following equations. Thus, if the number of the emergency information packets that need to be sent are same, the te_m in the different equations are equal. For a packet with priority pr_1, the total waiting time is given by

$$t'_1 = \sum_{i=1}^{n} \left[k_{i,1} * t_{(a,S)} + \sum_{m=N_{i,1}-k_{i,1}}^{N_{i,1}} te_m \right] \tag{4}$$

where $k_{i,j}$ denotes the number of packets whose priorities are same as j but the deadlines are shorter than the current packet's deadline at Node Level L_i.

Proof: Because there are n node levels from Node$_a$ to sink node, we consider the process hop by hop. When $i = n$, we can get the waiting time as

$$t'_{n,1} = k_{n,1} * ts_n + \sum_{m=N_{n,1}-k_{n,1}}^{N_{n,1}} te_m \tag{5}$$

where $t'_{n,1}$ denotes the waiting time at Node level n, and ts_n is the forwarding time. By analogy, we can get the waiting time at Node Level $n-1$, Node Level $n-2$, ..., Node Level 1. Then, we add them together, we get the following:

$$\sum_{i=1}^{n} t'_{i,1} = \sum_{i=1}^{n} \left[k_{i,1} * ts_i + \sum_{m=N_{i,1}-k_{i,1}}^{N_{i,1}} te_m \right]. \tag{6}$$

According to the definition, we can get

$$\begin{cases} t'_j = \sum_{i=1}^{n} t'_{i,j} \\ t_{(a,S)} = \sum_{i=1}^{n} ts_i. \end{cases} \tag{7}$$

Therefore, we get the total waiting time of the packet with priority pr_1, as shown in (4) based on (6) and (7). ∎

The packets with priority pr_2 need to wait for the transmission of the packets with priority pr_1, and they also need to wait for the packets with priority pr_2 whose deadlines are shorter than current packet's. Thus, the total waiting time for a packet with priority pr_2 is given by

$$t'_2 = \sum_{i=1}^{n} \left[(k_{i,2} + N_{i,1}) * t_{(a,S)} + \sum_{m=N_{i,1}+N_{i,2}-k_{i,2}}^{N_{i,1}+N_{i,2}} te_m \right] \tag{8}$$

Proof: We also consider the process hop by hop. When $i = n$, the time that $N_{n,1}$ packets with priority pr_1 have been sent to the destination node is $N_{n,1} * ts_n + \sum_{m=N_{n,2}+1}^{N_{n,1}+N_{n,2}} te_m$, and the waiting time for the transmission of the packets with priority pr_2 is $k_{n,2} * ts_n + \sum_{m=N_{n,2}-k_{n,2}}^{N_{n,2}} te_m$. Adding up these two values, the waiting time at Node Level L_n is given as

$$t'_{n,2} = (k_2 + N_{n,1}) * ts_n + \sum_{m=N_{n,1}+N_{n,2}-k_{n,2}}^{N_{n,1}+N_{n,2}} te_m \tag{9}$$

By analogy, we can get the waiting time at Node Level $n-1$, Node Level $n-2$, ..., Node Level 1. Then, we add them together, we get the following:

$$\sum_{i=1}^{n} t'_{i,2} = \sum_{i=1}^{n} \left[(k_{i,2} + N_{i,1}) * ts_i + \sum_{m=N_{i,1}+N_{i,2}-k_{i,2}}^{N_{i,1}+N_{i,2}} te_m \right] \tag{10}$$

Therefore, the total waiting time of the packet with priority pr_1 is given by (8) based on (10) and (7). ∎

Similarly, the packets with priority pr_3 need to wait for the transmission of the packets with priority pr_1, packets with priority pr_2, and packets with priority pr_3 whose deadlines are shorter than current packet's. Thus, the waiting time of packet with priority pr_3 is given by

$$t'_3 = \sum_{i=1}^{n} (k_{i,3} + N_{i,1} + N_{i,2}) * t_{(a,S)} + \sum_{i=1}^{n} \sum_{m=N_{i,1}+N_{i,2}+N_{i,3}-k_{i,3}}^{N_{i,1}+N_{i,2}+N_{i,3}} te_m. \quad (11)$$

The proof is similar to the proof of (9), thus we do not repeat it here.

From above, we can see that the waiting time of the packet with priority pr_1 is quite shorter than the waiting time of the packet with priority pr_2 and the waiting time of the packet with priority pr_3. It proves that EARS can deal with emergency data packet first, that is, EARS can guarantee the timeliness of emergency data packets.

IV. Simulation and Analysis

Based on the NS2 tool [24], we conduct simulation on the EARS. Taking the average waiting time, the end-to-end delay, and the packet loss rate as metrics, we compare EARS with other algorithms including FCFS, multilevel algorithm and DMP. We pick up a specific scenario, e.g., fire monitoring service, which requires an emergency prioritization scheme, as the simulation environment. Approximately 50–500 nodes (a sink node included) are deployed in an area of 300 m $*$ 300 m, constructing a tree network based on Spanning Tree Protocol. The network size means a typical "cell-coverage" of one fire-fighting station. The nodes are the monitoring facilities. The communication radius is to simulate the wireless transmission range of monitoring facilities. The transmission speed is the speed to push the packet's bits into the wire. The alarm information (there is fire) is generated once it appears, among other background data. The background data include sensor data and network information data. The sensor data are generated every 15 min and the network information is sent every 120 min. The sink node is located at the root node. The ordinary nodes generate packets of different emergency at random. Every packet has its own priority and deadline once generated, and the priority keeps unchanged while their deadline decreases with time. Every node maintains the neighbor list of last node level and next node level. By changing the packet length, we get MAC layer collision rate and the average waiting time, end-to-end delay, and packet loss rate based on the algorithms of EARS, FCFS, multilevel algorithm, and DMP. Table II presents the simulation settings.

A. Average Waiting Time

In the experiments, we control packets generation rate to make sure the network is under normal load. Then, we change the generation rate of packets with different emergency to simulate various network situations (including normal working condition, balanced condition, and emergency condition). The reason why we have taken these three situations as examples is that they represent the changing situation of the network. Normal working condition means that the network can work well, the

TABLE II
Simulation Settings

Parameter	Description
Network size	300 m $*$ 300 m
Transmission radio range	20 m
Channel	Wireless channel
Propagation	TwoRayGround
Traffic patterns	CBR
Number of nodes	Maximum 500
Transmission speed	250 kb/s
Packet size	100B/300B/500B/700B/900B/1100B
Simulation time	100 s

data packets with different priorities reach the sink node without congestion. Balanced condition means the number of different data packets is same. Emergency condition shows that there is large number of emergency data packets in the network, that is, emergency events occur frequently. Therefore, the EARS, FCFS, DMP, and multilevel algorithms are called successively when the ratio of packets with priority pr_1, packets with priority pr_2, and packets with priority pr_3 are set $\{3{:}5{:}2,\ 1{:}1{:}1,\ 5{:}3{:}2\}$. Other simulation parameters are set according to Table II.

Fig. 6 illustrates how waiting time of packets with priority pr_1, packets with priority pr_2, and packets with priority pr_3 change under different packet lengths. Each value represents an average of waiting time under the three network conditions. For each algorithm, the packet with the lower priority results in longer average waiting time at particular packet length. This indicates that the four algorithms conduct priority-based scheduling effectively. From Fig. 6(a)–(c), we know that DMP and EARS have obvious advantages over FCFS and multilevel algorithm. Furthermore, the average waiting time of EARS is much shorter when dealing with emergency packets (packets with priority pr_1), slightly shorter when dealing with packets with priority pr_2, and slightly longer when dealing with packets with priority pr_3, compared with the DMP algorithm. The reason is that on the same network condition, EARS tends to allocate more resources to emergency packets to ensure their prior transmission. However, DMP, although taking the strategy that the emergency packets preempt the other packets to complete transmission, turns out to have more average waiting time because it transfers packets in the TDMA method all the time. So the emergency packets are treated equally with nonemergency packets when they are transferred at the same time. Fig. 6(c) shows that due to the way EARS allocates resources, the waiting time of packets with priority pr_3 is longer but not much more than that in the DMP algorithm, because the DMP algorithm does not sign packets by deadlines. Thus, invalid packets are transmitted to the sink node and valid packets result in longer waiting time. Therefore, EARS is more efficient to deal with emergency packets for smart cities.

B. Packet Loss Rate

We test the packet loss rate in EARS and make a comparison with FCFS, DMP, and multilevel algorithm.

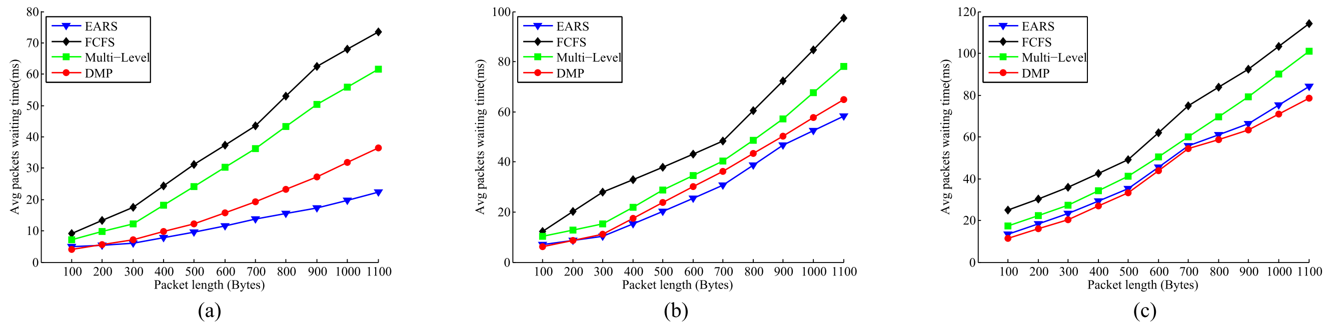


Fig. 6. Average waiting time of packets with different priorities. (a) Packets with priority pr_1. (b) Packets with priority pr_2. (c) Packets with priority pr_3.

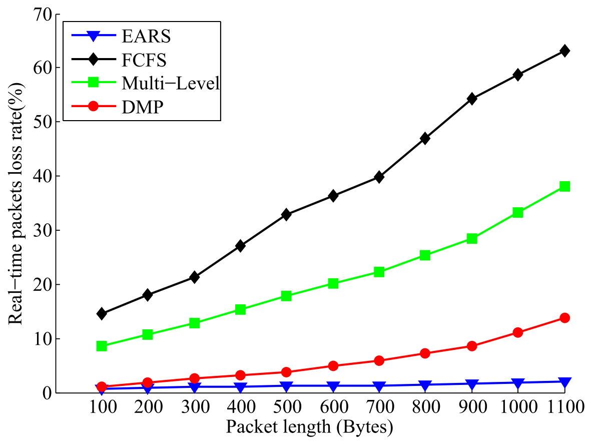


Fig. 7. Loss rate of emergency packets.

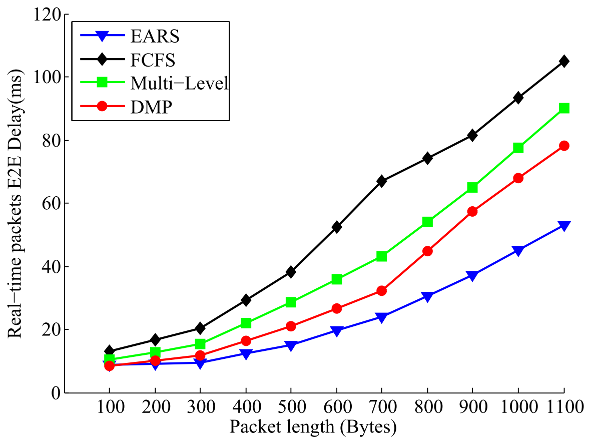


Fig. 9. End-to-end delay of emergency packets.

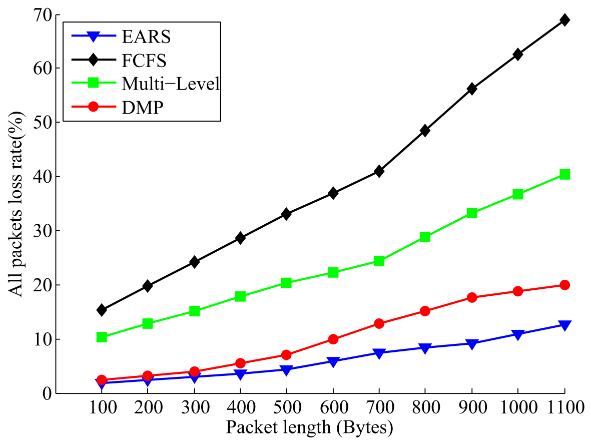


Fig. 8. Loss rate of all packets.

Fig. 7 shows how loss rate of emergency packet changes under different packet lengths. The packet loss rate of EARS is especially low. This value goes higher under longer packet, but the growth is not very obvious. The packet loss rate is 0.7452% at length of 100 B, and reaches maximum at length of 1100 B where its value is merely 2.0661%. The packet loss rate of DMP increases obviously with the increase in packet length, far more than that of EARS. Both FCFS and multilevel suffer from low efficiency.

Fig. 8 shows the loss rate of all packets under different packet lengths. Similar to Fig. 7, EARS gets the best result. DMP is not as good as EARS, but better than Multi-Level algorithm. FCFS turns out to be the worst. Meanwhile, the gap between DMP and EARS narrows in Fig. 8.

We conclude from Figs. 7 and 8 that the loss rate of all packets is higher than emergency packets' for every algorithm. The FCFS and multilevel algorithm both get a higher incidence of packets retransmission due to the MAC layer collision, which further worsens the network condition. It leads to a higher packet loss rate. DMP has no MAC layer collision, but the packet is discarded when its deadline is expired. In the condition where the packet length is short, the time slot is set correspondingly short because sending a packet is quite fast. From above, we know the waiting time of packet is short enough to make sure the emergency packet can be sent to the sink node before its deadline expires. The packet loss rate turns out to be low. For packets with low priorities, those with the shortest deadlines will be forwarded first to ensure the efficiency.

C. End-to-End Delay

Figs. 9 and 10 illustrate the average end-to-end delay of emergency packets and all packets under different packet lengths, respectively. In Fig. 9, the average end-to-end delay of EARS and DMP is almost the same for packets whose lengths are short. EARS algorithm is slightly worse. The reason is that the delay

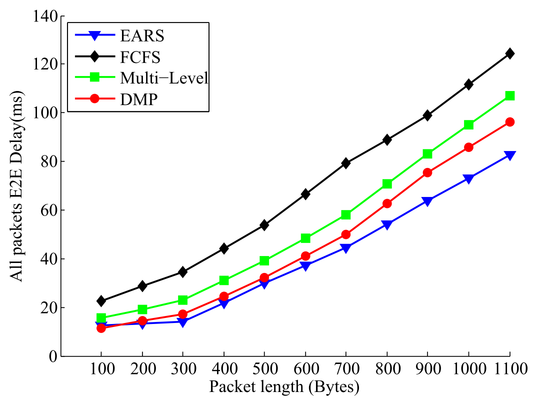


Fig. 10. End-to-end delay of all packets.

of EARS confirming the packet emergency is longer than that of DMP sending an emergency packet in the TDMA method. However, with the increase in packet length, time slot of the TDMA method gets longer and the waiting time becomes longer as well. The time of confirming emergency shares less and less ratio in time of sending an emergency packet. For EARS, the time of confirming emergency is far less than the delay of emergency packet transmission in the TDMA method. The advantage becomes more explicit when the packet length further increases.

The results in Fig. 10 have the similar trend to Fig. 9. EARS algorithm preferentially allocates resources to emergency packet, resulting in longer waiting time of packet with low priority. The end-to-end delay of all packets is larger than that of emergency packets. Furthermore, EARS has a lower loss rate of all packets and emergency packets than DMP. It means DMP can be more time consuming, which affects packets sent to the sink node. Thus, EARS has less end-to-end delay of all packets than DMP.

From Figs. 9 and 10, we see clearly that EARS and DMP are better than FCFS and multilevel algorithm in the end-to-end delay. When packet length is short, the real-time performance of EARS and DMP is approximately similar. With the increase in packet length, the advantage of EARS on real-time performance will be more obvious, especially in average end-to-end delay of emergency packets.

V. Conclusion

After fully considering the actual applications of packet scheduling for smart cities, such as fire monitoring service and medical rescue service, this paper proposes EARS, a data-emergency-aware packet scheduling scheme for smart cities. According to EARS, the destination node is able to get emergency information of every packet. The resource allocation in IoT for smart cities is optimized to ensure the timeliness of emergency packets. At the same time, in order to reduce the packet loss, the nonemergency packets need to be transmitted to the sink node within their deadlines. Finally, we carry out experiments to evaluate EARS. The simulation results show that EARS almost has no MAC layer collision and gets better performance than FCFS, DMP, and multilevel algorithms in term of average waiting time, end-to-end delay, and packet loss rate.

EARS guarantees the emergency packets to get the highest priorities, so that emergent event can be dealt with before the deadline expires for smart cities. However, it only meets the packet scheduling in the tree-based networks. For the star networks and the mesh networks, it is limited in the applications. Therefore, our future work will focus on how to deal with the emergency packets in the star networks and the mesh networks to ensure the real-time performance of emergency packets.

Moreover, as enhancements to the proposed EARS scheme, we will consider the updating dynamically of network when nodes die or new nodes are added, which increases the flexibility of the EARS algorithm and greatly decreases the cost of network maintain. At present, we solve this problem by using topology maintain periodicity. In addition, we have considered adding and integrating with business intelligence [25]. Then, the smart service can act accordingly due to different demands.

References

[1] M. V. Moreno *et al.*, “Applicability of big data techniques to smart cities deployments,” *IEEE Trans. Ind. Informat.*, vol. 13, no. 2, pp. 800–809, Apr. 2017.

[2] T. Qiu, R. Qiao, and D. Wu, “EABS: An event-aware backpressure scheduling scheme for emergency internet of things,” *IEEE Trans. Mobile Comput.*, doi: 10.1109/TMC.2017.2702670.

[3] I. A. T Hashem *et al.*, “The role of big data in smart city,” *Int. J. Inf. Manage.*, vol. 36, no. 5, pp. 748–758, 2016.

[4] M. Nitti, R. Girau, and L. Atzori, “Trustworthiness management in the social internet of things,” *IEEE Trans. Knowl. Data Eng.*, vol. 26, no. 5, pp. 1253–1266, May 2014.

[5] T. Qiu, K. Zheng, H. Song, M. Han, and B. Kantarci, “A local-optimization emergency scheduling scheme with self-recovery for smart grid,” *IEEE Trans. Ind. Inf*, doi: 10.1109/TII.2017.2715844.

[6] G. Lu and B. Krishnamachari, “Minimum latency joint scheduling and routing in wireless sensor networks,” *Ad Hoc Netw.*, vol. 5, no. 6, pp. 832–843, 2007.

[7] P. Guo, T. Jiang, Q. Zhang, and K. Zhang, “Sleep scheduling for critical event monitoring in wireless sensor networks,” *IEEE Trans. Parallel Distrib. Syst.*, vol. 23, no. 2, pp. 345–352, Feb. 2012.

[8] U. Jang, S. Lee, and S. Yoo, “Optimal wake-up scheduling of data gathering trees for wireless sensor networks,” *J. Parallel Distrib. Comput.*, vol. 72, no. 4, pp. 536–546, 2012.

[9] F. Yang and I. Augé-Blum, “Delivery ratio-maximized wakeup scheduling for ultra-low duty-cycled WSN under real-time constraints,” *Comput. Netw.*, vol. 55, no. 3, pp. 497–513, 2011.

[10] R. Gomathi and N. Mahendran, “An efficient data packet scheduling schemes in wireless sensor networks,” in *Proc. Int. Conf. Electron. Commun. Syst.*, Feb. 26–27, 2015, pp. 542–547.

[11] K.-H. Phung, B. Lemmens, M. Goossens, A. Nowe, L. Tran, and K. Steenhaut, “Schedule-based multi-channel communication in wireless sensor networks: A complete design and performance evaluation,” *Ad Hoc Netw.*, vol. 26, pp. 88–102, 2015.

[12] V. Chang, “Towards a big data system disaster recovery in a private cloud,” *Ad Hoc Netw.*, vol. 35, pp. 65–82, 2015.

[13] X. Shen, C. Bo, J. Zhang, S. Tang, X. Mao, and G. Dai, “EFCon: Energy flow control for sustainable wireless sensor networks,” *Ad Hoc Netw.*, vol. 11, no. 4, pp. 1421–1431, 2013.

[14] X. Xu, X. Li, and M. Song, “Distributed scheduling for real-time data collection in wireless sensor networks,” in *Proc. IEEE Global Telecommun. Conf.*, Dec. 9–13, 2013, pp. 426–431.

[15] Y. Xue, B. Ramamurthy, and M. C. Vuran, “SDRCS: A service-differentiated real-time communication scheme for event sensing in wireless sensor networks,” *Comput. Netw.*, vol. 55, no. 15, pp. 3287–3302, 2011.

[16] O. Chipara, C. Lu, and G.-C. Roman, “Real-time query scheduling for wireless sensor networks,” in *Proc. Real Time Syst. Symp.*, Dec. 3–6, 2007, pp. 389–399.

[17] E.-M. Lee, A. Kashif, D.-H. Lee, I.-T. Kim, and M.-S. Park, “Location based multi-queue scheduler in wireless sensor network,” in *Proc Int. Conf. Adv. Commun. Technol.*, Feb. 7–10, 2010, pp. 551–555.

[18] N. Nidal, L. Karim, and T. Taleb, "Dynamic multilevel priority packet scheduling scheme for wireless sensor network," *IEEE Trans. Wireless Commun.*, vol. 12, no. 4, pp. 1448–1459, Apr. 2013.
[19] E. Bini, G. C. Buttazzo, and G. M. Buttazzo, "Rate monotonic analysis: The hyperbolic bound," *IEEE Trans. Comput.*, vol. 52, no. 7, pp. 933–942, Jul. 2003.
[20] G. C. Buttazzo, "Rate monotonic vs. EDF: Judgment day," *Real Time Syst.*, vol. 29, no. 1, pp. 5–26, 2005.
[21] P. Jayachandran and T. Abdelzaher, "Transforming distributed acyclic systems into equivalent uniprocessors under preemptive and non-preemptive scheduling," in *Proc. Euromicro Conf. Real Time Syst.*, Jul. 2–4, 2008, pp. 233–242.
[22] G. C. Buttazzo, M. Bertogna, and G. Yao, "Limited preemptive scheduling for real-time systems. A survey," *IEEE Trans. Ind. Inf.*, vol. 9, no. 1, pp. 3–15, Feb. 2013.
[23] P. Chennakesavula, J. Ebenezer, S. Murty, and T. Jayakumar, "Real-time packet scheduling for real-time wireless," in *Proc. IEEE Int. Adv. Comput. Conf.*, Feb. 22–23, 2013, pp. 273–276.
[24] D. Mahrenholz and S. Svilen, "Real-time network emulation with ns-2," in *Proc. 8th IEEE Int. Symp. Distrib. Simul. Real Time Appl.*, Oct. 21–23, 2004, pp. 29–36.
[25] V. Chang, "The business intelligence as a service in the cloud," *Future Gener. Comput. Syst.*, vol. 37, pp. 512–534, 2014.

Tie Qiu (M'12–SM'16) received the M.Sc. and Ph.D. degrees in computer science from the Dalian University of Technology, Dalian, China, in 2005 and 2012, respectively.

He is currently an Associate Professor with the School of Software, Dalian University of Technology. From January 2014 to January 2015, he was a Visiting Professor with the Department of Electrical and Computer Engineering, Iowa State University, USA. He has authored/coauthored 8 books, more than 100 scientific papers in international journals and conference proceedings, such as ToN, TMC, TII, TIP, IEEE Communications, Computer Networks, etc. He has contributed to the development of 4 copyrighted software systems and invented 15 patents.

Dr. Qiu serves as an Associate Editor of the IEEE ACCESS, *Computers and Electrical Engineering* (Elsevier journal), and *Human-Centric Computing and Information Sciences* (Springer journal), an Editorial Board Member of the *Ad Hoc Networks* (Elsevier journal) and *International Journal on AdHoc Networking Systems*, a Guest Editor of the *Future Generation Computer Systems* (Elsevier journal). He serves as the General Chair, PC Chair, Workshop Chair, Publicity Chair, Publication Chair or TPC Member of a number of conferences. He is a senior member of China Computer Federation and a senior member of ACM.

Kaiyu Zheng (S'16) received the B.E. and Master's degrees in software engineering from the Dalian University of Technology (DUT), Dalian, China, in 2014 and 2017, respectively.

He is an excellent Graduate Student of the DUT and has been awarded several scholarships in academic excellence and technology innovation. He participated in Open Source Hardware and Embedded Computing Contest 2012 and received the first prize. His research interests include embedded system and Internet of Things.

Min Han (M'95–A'03–SM'06) received the B.S. and M.S. degrees from the Department of Electrical Engineering, Dalian University of Technology, Dalian, China, in 1982 and 1993, respectively, and the M.S. and Ph.D. degrees from Kyushu University, Fukuoka, Japan, in 1996 and 1999, respectively, both in control system and engineering.

Since 2003, she has been a Professor in the Faculty of Electronic Information and Electrical Engineering, Dalian University of Technology. She is the author of four books and more than 200 articles. Her current research interests include neural networks, chaos and their applications to control and identification.

C. L. Philip Chen (M'88–SM'94–F'07) received the M.S. degree in electrical engineering from the University of Michigan, Ann Arbor, MI, USA, in 1985, and the Ph.D. degree in electrical engineering from Purdue University, West Lafayette, IN, USA, in 1988.

He is currently a Chair Professor in the Department of Computer and Information Science and the Dean of the Faculty of Science and Technology, University of Macau, Macau.

Dr. Chen is a Fellow of the AAAS. He is the Jr. Past President of the IEEE SYSTEMS, MAN, AND CYBERNETICS SOCIETY, and the Editor-in-Chief of the IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS: SYSTEMS. He is an Accreditation Board of Engineering and Technology Education Program Evaluator for computer engineering, electrical engineering, and software engineering programs.

Meiling Xu received the B.S. and Ph.D. degrees in control system and engineering from the Faculty of Electronic Information and Electrical Engineering, Dalian University of Technology, Dalian, China, in 2011 and 2016, respectively.

Since December, 2016, she has been a Lecturer with the Faculty of Electronic Information and Electrical Engineering, Dalian University of Technology. Up to now, she has authored/coauthored 15 scientific papers in international journals and conference proceedings. Her current research interests include chaotic time series prediction, neural networks, and optimization.