A Hierarchical Data Transmission Framework for Industrial Wireless Sensor and Actuator Networks

Xi Jin, Fanxin Kong, Linghe Kong, *Member, IEEE*, Huihui Wang, *Member, IEEE*, Changqing Xia, Peng Zeng, and Qingxu Deng

***Abstract*—A smart factory generates vast amounts of data that require transmission via large-scale wireless networks. Thus, the reliability and real-time performance of large-scale wireless networks are essential for industrial production. A distributed data transmission scheme is suitable for large-scale networks, but is incapable of optimizing performance. By contrast, a centralized scheme relies on knowledge of global information and is hindered by scalability issues. To overcome these limitations, a hybrid scheme is needed. We propose a hierarchical data transmission framework that integrates the advantages of these schemes and makes a tradeoff among real-time performance, reliability, and scalability. The top level performs coarse-grained management to improve scalability and reliability by coordinating communication resources among subnetworks. The bottom level performs fine-grained management in each subnetwork, for which we propose an intrasubnetwork centralized scheduling algorithm to schedule periodic and aperiodic flows. We conduct both extensive simulations and realistic testbed experiments. The results indicate that our method has better schedulability and reduces packet loss by up to 22% relative to existing methods.**

***Index Terms*—Data flow scheduling, hierarchical data transmission framework, resource coordination.**

Manuscript received October 23, 2016; revised February 5, 2017; accepted March 19, 2017. Date of publication March 22, 2017; date of current version August 1, 2017. This work was supported in part by the National Natural Science Foundation of China under Grant 61502474 and Grant 61233007 and in part by the Youth Innovation Promotion Association of the Chinese Academy of Sciences. Paper no. TII-16-1201. (*Corresponding author: Peng Zeng.*)

X. Jin and C. Xia are with the Laboratory of Networked Control Systems, Shenyang Institute of Automation, Chinese Academy of Sciences, Shenyang 110016, China (e-mail: jinxi@sia.cn; xiachangqing@sia.cn).

F. Kong is with the Department of Computer and Information Science, University of Pennsylvania, Philadelphia, PA 19104 USA (e-mail: fanxink@seas.upenn.edu).

L. Kong is with the Department of Computer Science and Engineering, Shanghai Jiao Tong University, Shanghai 200240, China (e-mail: linghe.kong@sjtu.edu.cn).

H. Wang is with the Department of Engineering, Jacksonville University, Jacksonville, FL 32211 USA (e-mail: hwang1@ju.edu).

P. Zeng is with the Laboratory of Networked Control Systems, Shenyang Institute of Automation, Chinese Academy of Sciences, Shenyang 110016, China, and also with the Shenyang Institute of Automation, Guangzhou, Chinese Academy of Sciences, Guangzhou 511458, China (e-mail: zp@sia.cn).

Q. Deng is with the Institute of Cyber-Physical Systems, Northeastern University, Shenyang 110819, China (e-mail: dengqx@mail.neu.edu.cn).

Digital Object Identifier 10.1109/TII.2017.2685689

I. Introduction

SMART factories generate vast amounts of data, including sensing information and control messages, during the working process [1], [2]. These data are transmitted between machines and the control room via wireless transmission [3], [4]. Thus, real-time performance and reliability are essential for industrial systems. Industrial wireless sensor and actuator networks (IWSANs), which serve as the communication media, must be capable of supporting real-time and reliable communication. An IWSAN used in a smart factory contains thousands of sensors and actuators, and data flows are variable because of changes in manufacturing requirements. Distributed data transmission schemes [5], which are scalable and flexible, are suitable for handling large-sized and unpredictable communications. However, they are incapable of optimizing real-time performance and reliability based on local information. By contrast, centralized data transmission schemes can rely on global information to improve the two performances. However, their network scalability is limited by the capacity of the centralized managers. Although the emergence of software defined wireless sensor network (SDWSN) controllers [6] has enabled the use of centralized schemes, no studies have addressed the centralized data management of large-scale wireless sensor networks. Regardless of which of these two schemes is adopted, excessively long paths are unavoidable [see Fig. 1(a)]. As the number of hops increases, the number of packet losses also increases, leading to decreased reliability and increased response times.

Therefore, we adopt a hierarchical data transmission framework [see Fig. 1(b)] to make a tradeoff among system performance, scalability, and flexibility. The bottom level of the hierarchy corresponds to the data flow management in each subnetwork, whereas the top level coordinates the subnetworks. Based on the hierarchical framework, this paper focuses on the proper use of communication resources to improve the real-time performance and reliability of data flows. Industrial wireless standards, such as WirelessHART [7], WIA-PA [8], and ISA100.11a [9], are based on the IEEE 802.15.4 standard and support the multichannel time-division multiple access (TDMA) scheduling strategy, which is more controllable and has higher throughput than the carrier sense multiple access (CSMA) strategy and a single channel. We investigate these strategies; the communication resources refer to time slots and channels. Three

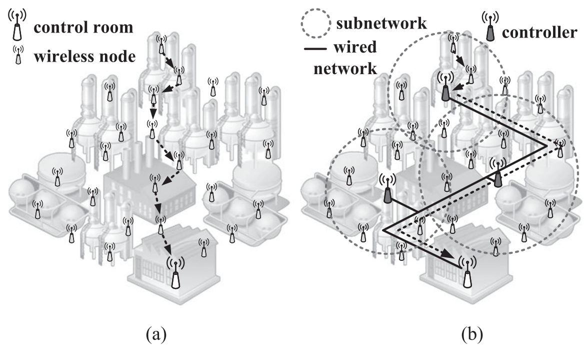


Fig. 1. Large-scale IWSAN. (a) Excessively long path. (b) Hierarchical framework.

major challenges exist. First, a novel framework must be proposed to support resource management with complex dependence and competition among vast amounts of data. Second, for the bottom level, existing centralized scheduling algorithms can improve the real-time performance and reliability of industrial systems but are unable to address the dynamics of data flows. Third, for the top level, the coverage areas of subnetworks overlap, and radio interference affects the reliability. To address these challenges, we present the following contributions.

First, we propose a hierarchical framework to facilitate managing the communication resources. Existing hierarchical frameworks focus on scalability and do not address how to improve the real-time performance and reliability. Our hierarchical framework manages different-grained resources on two levels. The fine-grained management at the bottom level helps to guarantee real-time requirement and improve reliability, whereas the coarse-grained management at the top level not only supports scalability but also enhances the reliability.

Second, we propose a real-time scheduling algorithm for the bottom level. Few studies have considered unpredictable data flows in IWSANs, and reliability requirements have been neglected. In our scheduling algorithm, shared resources can be used to transmit unpredictable data flows, and the objective of selecting shared resources is to reduce packet losses and improve system reliability.

Third, we propose a channel assignment algorithm for the top level. The existing industrial hierarchical frameworks do not consider how to eliminate the interference among subnetworks. Our algorithm assigns as many resources as possible to each subnetwork and guarantees that the overlapped subnetworks do not use the same channels. Thus, the subnetworks are isolated, and more resources can be used to address the dynamics of data flows.

The remainder of this paper is organized as follows: Section II introduces related works. Section III presents our hierarchical data transmission framework. Sections IV and V propose our scheduling algorithms and coordinating algorithms, respectively. Section VI discusses the experimental and simulated results, and Section VII concludes this paper.

II. Related Works

Big data are everywhere, including smart urbanism [10], environmental monitoring [11], and industrial applications [12]. Some works have considered the real-time performance and reliability of big data systems. For example, [13] proposes a real-time software stack and a framework for real-time big data stream processing, whereas [14] designs an architecture for time-critical big data systems based on a cluster of machines. To create a homogeneous programming platform for industrial systems, in [15], distributed cyber physical Java nodes are used to construct a real-time Java-centric architecture. Additionally, the work in [16] focuses on large unconnected networks and proposes a reliable data-muling protocol to collect data. However, these works do not address how to design network frameworks.

In this paper, we only focus on networks in big data systems. Networks are divided into three categories based on the type of management scheme utilized: distributed networks, centralized networks. and hierarchical networks. For large-scale wireless sensor networks, distributed management schemes (e.g., [5], [17], and [18]) render networks scalable and flexible. However, they are not suitable for industrial networks because each network node transmits data according to local and neighbor information and does not have information about remote nodes. Because of the limited information available, predicting and controlling the end-to-end delay of a packet using this method is challenging; thus, real-time performance cannot be optimized. Currently, no strict centralized scheme is used in large-scale wireless sensor networks, even if SDWSNs [6] can support this type of scheme. Researchers studying SDWSNs pay more attention on system frameworks and resource management (e.g., [19]–[22]).

Hierarchical data transmission frameworks (e.g., [23]–[27]), combine the advantages of these two schemes. In [23], a mesh network connects multiple sensor networks, and the low-power listening protocol and collection tree protocol are adopted to gather sensing data. In [24], the backbone network bridges some 802.15.4 subnetworks with an 802.3 Ethernet, and the data-transmitting method is an improved flooding strategy. In [25] and [26], network nodes are partitioned into several sections, which subsequently submit data to the base stations. The EMERSON Company advises the use of a hierarchical network to improve scalability [27]. However, no related works have been performed to eliminate the interference among subnetworks, which reduces the network reliability. For a subnetwork, the scheduling problem is similar to that in small-scale IWSANs. Some works of the real-time scheduling problem of IWSANs (e.g., [28]–[31]) do not address aperiodic and unpredictable data flows. Additionally, although the works [32] and [33] consider how to schedule aperiodic and unpredictable data flows, too much interference is introduced into other flows.

III. Our Hierarchical Data Transmission Framework

A. Framework

A hierarchical network [see Fig. 1(b)] contains controllers, sensor nodes, and actuator nodes. A subnetwork controller and some surrounding nodes constitute a subnetwork, and a controller only corresponds to a subnetwork. The subnetwork controller is also the gateway between the subnetwork and a wired network. These subnetwork controllers are connected to a coordinator by wired networks. In this paper, we do not address

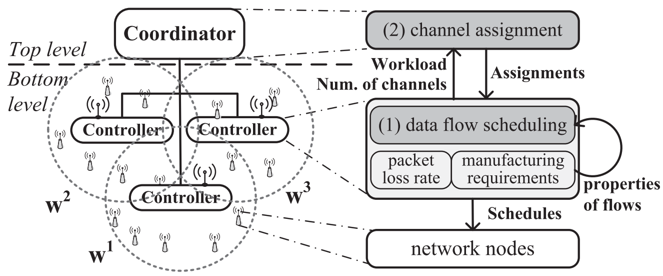


Fig. 2. Framework overview.

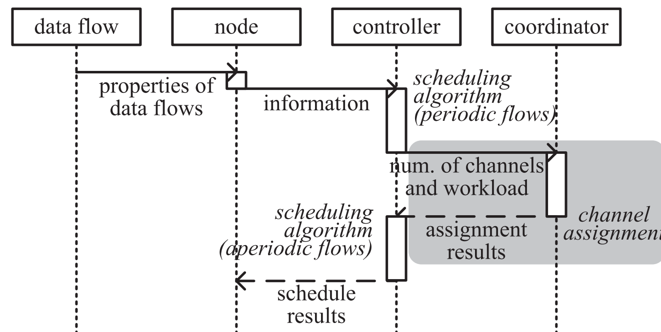


Fig. 3. Working process.

deployment and networking techniques, but we assume that the network topology has been given. Fig. 2 presents our framework.

At the bottom level, each controller manages the data flows in its subnetwork. Two types of data flows exist. The first type of data flow comprises periodic data flows, which include sensing data and normal control messages. The second type of data flow consists of aperiodic and unpredictable data flows, such as alarms and artificial triggering events. We assume that any two aperiodic data flows do not interfere with each other because this type of data flow is infrequent in real systems [32]. These data flows must be delivered to their destinations before their deadlines. The scheduling algorithm in the controller assigns the fixed time slot and channel to each node-to-node transmission, and these assignments constitute real-time schedules. According to the schedules generated by the scheduling algorithm, we can determine the number of channels required by each subnetwork. The coordinator at the top level assigns channels based on the requirements. However, if the dedicated resources are assigned to infrequent aperiodic flows, they are extremely wasteful. Additionally, if only one channel is reserved to transmit aperiodic flows, isolation failure may occur (the evaluation is presented in Section VI). Therefore, only the channel requirement of periodic flows is considered by the channel assignment algorithm. Aperiodic flows use the remaining communication resources if they are sufficient; otherwise, they are allowed to steal resources from periodic flows.

At the top level, the coordinator assigns channels to isolate subnetworks based on the number of required channels. Although the channel resource is limited, the channel requirement must be satisfied, and each subnetwork should acquire as many channels as possible because it must cope with the dynamics of data flows. The properties of data flows are set according to the manufacturing requirements. If the manufacturing requirements change, the properties (e.g., the periods and paths) should be adjusted. A poor wireless environment produces additional data losses. The controller calculates the packet loss rate (PLR) for each data flow in an assessment window. When a wireless environment deteriorates and a flow's PLR exceeds given threshold, the flow should be simultaneously sent via multiple paths to improve reliability; that is the number of routing paths is changed, and the schedules must be regenerated. If the assigned channels are not sufficient to schedule the new flow set, the coordinator must be reinvoked, and the channels of other subnetworks may need to be adjusted. Therefore, the assignment of additional channels to a subnetwork can reduce the number of adjustments of the entire network. In addition to the number of channels, the workload of each subnetwork is piggybacked to the coordinator. When all channel requirements have been satisfied and the idle channels can be used to handle the dynamics, the subnetwork with higher utilization should be assigned additional resources.

B. Working Process

The management process of the hierarchical framework is shown in Fig. 3. Two situations invoke the management process. The first situation is the initial stage of the hierarchical network. The controller receives all information about data flows, and according to this information, the algorithm in the controller schedules periodic data flows and obtains the number of required channels. The number and workload are submitted to the coordinator. The algorithm in the coordinator assigns channels for isolating subnetworks. The channel assignments are published to controllers via a wired network. The controllers schedule aperiodic data flows on the assigned channels and send the scheduling information to their nodes. The second situation occurs when the properties of data flows are changed and is similar to the first situation. The difference is that if the controller can cope with the change using the assigned resources, it does not need to invoke a coordinator (the part in the gray block); otherwise, the process is the same as the initial stage.

In the hierarchical framework, when subnetworks cannot cope with the changes of the data flows, a coordinator is required. Thus, compared to the strict centralized scheme, in which all changes must be submitted to the manager of the entire network, and then, addressed, the hierarchical framework reduces communication costs and decreases the response time. Compared to distributed schemes, our scheduling algorithm is an intrasubnetwork centralized TDMA strategy that is more controllable and reliable.

C. Our Addressed Problems

In the working process, two problems must be solved: the scheduling problem at the bottom level and the channel assignment problem at the top level. The scheduling algorithm schedules the two types of data flows on the shared resources (for more details, refer to Section IV). The objective of the channel assignment algorithm is to isolate subnetworks by different channels and assign as many channels as possible to a subnetwork such that it is more adaptable to flow changes (for more details, refer to Section V).

IV. Scheduling Algorithm at the Bottom Level

In this section, we describe how to schedule two types of data flows with a real-time constraint in a single subnetwork.

A. Subnetwork Model

A subnetwork is characterized by $w = \langle N, L \rangle$. The node set $N = \{n_1, n_2, ...\}$ is used to denote the controller, sensors, and actuators. If l_{ij} exists in the link set L, then the nodes n_i and n_j can directly communicate with each other. Otherwise they cannot.

Two types of data flows exist: periodic flows $F = \{f_1, f_2, ...\}$ and aperiodic flows $F' = \{f'_1, f'_2, ...\}$. Flow f_i periodically releases a packet at period p_i, and the data packet is routed to its destination along path $\pi_i = \{l_{ab}, l_{bc}, ...\}$. All routing paths are fixed before schedule generation, and the routing techniques in wireless sensor networks (e.g., [34] and [35]) can be used to obtain paths. When the data packet is released at time t, it must be delivered to the destination before time $(t + p_i + 1)$; i.e., p_i is also its relative deadline. Similarly, aperiodic flow f'_i is characterized by $\langle p'_i, \pi'_i \rangle$. However p'_i only denotes its deadline, and the releasing time of its data packet is unpredictable. The transmission of a packet via the jth link of path π_i (π'_i) is referred to as *transmission* τ_{ij} (τ'_{ij}). After transmission τ_{ij} is executed, the next transmission (τ_{ij+1}) is released.

A maximum of 16 channels are supported in the IEEE 802.15.4 standard and are shared by all subnetworks. For a subnetwork, the number of available channels is m $(1 \leq m \leq 16)$. The scheduling algorithm assigns a time slot and a channel for scheduling a transmission. The assignments for all transmissions of a packet constitute an end-to-end schedule. In existing industrial wireless protocols (e.g., [7] and [8]), the TDMA scheduling strategy only applies to periodic data flows. A *superframe* organizes all end-to-end schedules with the same period and repeatedly executes at the same period. The periodic flows release packets at the beginning of the superframe, and the packets are delivered to their destinations before the end of the superframe. A subnetwork has multiple superframes with different periods that all start from the same time slot. All superframes execute using the same communication resources. Thus, two types of inter- and intrasuperframe interference must be avoided. 1) Any two transmissions cannot be scheduled at the same time slot and on the same channel; otherwise, the overlapped transmissions cannot be separated. (2) Any node cannot serve more than one transmissions at one time slot because each node is only equipped with one transceiver. To easily manage the transmissions of different superframes, these superframes are considered to constitute a *hyperframe* whit a period that is equal to the lowest common multiple of all superframes. The flow's period follows a harmonic chain that conforms to the expression $p_{\min} \times 2^a$ [7], [8], where $p_{\min}$ is the unit period that contains a certain number of time slots and a is a positive value. Thus, the period of the hyperframe is $T = \max_{\forall f_i \in F} \{p_i\}$. The scheduling algorithm only considers how to schedule data flows

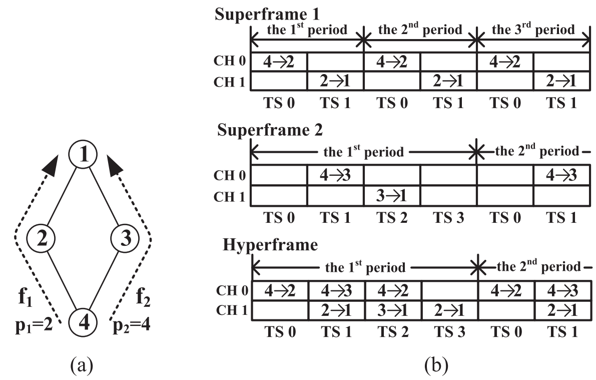


Fig. 4. Example. (a) Subnetwork. (b) Superframes and the hyperframe.

in the first T because all schedules are periodically repeated. Fig. 4 presents a simple subnetwork, its superframes and hyperframe, where CH and TS denote the channels and time slots, respectively.

To guarantee controllability and maintain consistency with periodic flows, aperiodic flows should also be scheduled in the same multichannel TDMA framework. However, they can be released at any time. Dedicated resources should be assigned to them in all situations. Hence, in our system, aperiodic data flows are allowed to steal communication resources from periodic data flows. When a node intends to send a packet of an aperiodic flow at the assigned time slot, it does so at the beginning of the time slot. When the node intends to send a packet of a periodic flow, it listens to the assigned channel for a short time. If the channel is not used, the packet is sent. Otherwise, it is discarded.

The problem that we address at the bottom level can be stated as follows: Given the subnetwork w and the flow sets F and F', our objective is to schedule data flows in real time, such that no interferences exist. The scheduling method should be as simple as possible because the schedules should be rapidly generated as the subnetwork frequently changes.

Note that the subnetwork can support flow period changes and that a flow has multiple paths. We do not specify these changes and paths in the model because when a flow's period changes, it is regarded as a new flow and because flows with multiple paths are regarded as multiple single-path flows. The periods of these single-path flows are equivalent to the period of the original flow. Thus, the transmission of multiple single-path packets is the same as the transmission of a multipath flow packet. The total throughput is unchanged, and other flows are not affected.

B. Our Scheduling Algorithm

The controller first schedules periodic data flows and then schedules aperiodic data flows.

1) Periodic Data Flows: We use the classical rate-monotonic (RM) policy to schedule periodic data flows. In the RM policy, a data flow with a short period is assigned a high priority. The transmissions have the same priority as the flow to which they belong. The scheduling algorithm traverses all time slots in a hyperframe and m available channels. For each time slot, if idle channels exist, the scheduling algorithm selects the released transmission that has the highest priority among the

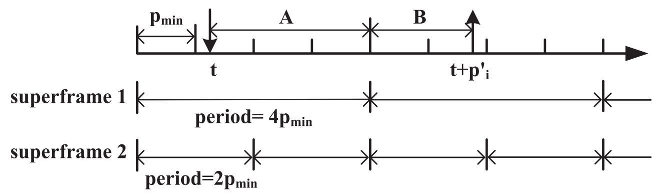


Fig. 5. Illustration for Theorem 1.

flows that do not interfere with the scheduled flows at this time slot to be scheduled on an available channel. If the transmissions of all periodic flows are scheduled before their deadlines, the periodic flow set is schedulable. Otherwise, the subnetwork is unschedulable.

The number of channels required by a subnetwork and the subnetwork workload must be submitted to the coordinator for channel assignment. We check the channel number m in order from 1 to 16. For each value, the RM scheduling algorithm is executed. When a value makes all periodic data flows schedulable, that value is the number of required channels. We define the subnetwork workload as the average number of transmissions per time slot. For flow f_j, $|\pi_j|$ transmissions are scheduled in time interval p_j. Thus, the workload introduced by flow f_j is $\frac{|\pi_j|}{p_j}$, and the total workload in a subnetwork is $\Gamma = \sum_{\forall f_j \in F} \frac{|\pi_j|}{p_j}$.

2) Aperiodic Data Flows: Periodic data flows are released at the beginning of superframes, whereas aperiodic data flows can be released at any time slot. For an aperiodic flow with a deadline p'_k, Theorem 1 is proposed to explain that whenever the aperiodic flow releases a data packet, if all transmissions in the packet can be scheduled in a superframe with a period $\hat{p}_k$, then, the packet does not miss the deadline.

Theorem 1: When aperiodic flow f'_k releases a data packet at time slot t, between time slot t and its deadline $(t + p'_k)$, an entire superframe with period $\hat{p}_k = p_{\min} \times 2^{\left\lfloor \log_2(\frac{p'_k}{p_{\min}}) \right\rfloor - 1}$ must exist.

Proof: Recall that the period of periodic flows and superframes conforms to the expression $p_{\min} \times 2^a$. For aperiodic flow f'_k, in time interval $[t, t + p'_k]$, there must exist $p_{\min} \times 2^v$ time slots, where $v = \left\lfloor \log_2(\frac{p'_k}{p_{\min}}) \right\rfloor$. In the worst case, these time slots do not belong to one superframe. Fig. 5 shows an example, in which $p'_k = p_{\min} \times 4.5$. Although the available time is larger than 2^2 unit periods, it does not contain an entire superframe 1 with period $(p_{\min} \times 2^2)$ because the start time of the superframe is fixed, as shown in Fig. 5. However, the available time contains an entire superframe with the period $p_{\min} \times 2^{v-1}$. We separate the available time into two parts (A and B) using the boundary of the superframe with period $p_{\min} \times 2^v$. We know that $A + B \geq p_{\min} \times 2^v$. Thus, at least one of the two parts is larger than or equal to $p_{\min} \times 2^{v-1}$ because all superframes in a subnetwork begin with the same time lost. Thus, the boundary of the superframe with period $p_{\min} \times 2^v$ is also the boundary of the superframe with period $p_{\min} \times 2^{v-1}$. Therefore, the larger period between A and B contains an entire superframe whose period is $p_{\min} \times 2^{v-1}$. ■

Algorithm 1: Scheduling for aperiodic flow f'_k.

Require: f'_k, S, F
Ensure: s'_{kj}, r'_{kj} $(\forall j \in [1, |\pi'_k|])$
1: Sort periodic data flows in the decreasing order of their utilization $\frac{|\pi_i|}{p_i}$, where f_1 has the highest utilization;
2: **for** $i = 0$ to $|F|$ **do**
3: $Q = S-$ schedules of the flows f_0, ..., f_i;
4: **for** $t = 1$ to $\hat{p}_k$ **do**
5: **if** In Q, an idle channel exists at time slots $t + \hat{p}_k \times g$ $(g \in [0, \frac{T}{\hat{p}_k}))$ **then**
6: **if** In Q, no Interference (2) exists between τ'_{kj} and the scheduled transmissions at time slots $t + \hat{p}_k \times g$ $(g \in [0, \frac{T}{\hat{p}_k}))$ **then**
7: $s'_{kj} = t$; $r'_{kj} =$ an idle channel;
8: **if** flow f'_k has arrived at the destination **then**
9: **return** Schedulable;
10: **return** Unschedulable;

Although period $\hat{p}_k$ may be less than the unit period $p_{\min}$, it can be simply solved by reducing the unit period. When a data packet is released from an aperiodic flow, it waits to be scheduled until a superframe with period $\hat{p}_k$ starts. Then, the packet is transmitted in this superframe. Algorithm 1 shows the scheduling algorithm for aperiodic flow f'_k. We use S to denote the scheduling results of the periodic flows. For τ'_{kj} $(\forall j \in [1, |\pi'_i|])$, the algorithm must assign the s'_{kj}th time slot and the r'_{kj}th channel. The stealing strategy will cause the loss of periodic flows' packets. To reduce the PLR, the aperiodic flow should steal the high-utilization flow, and thus, obtain additional resources from less flows.

Algorithm 1 sorts periodic data flows in the decreasing order of their utilization (line 1), and checks whether the stolen and idle resources are adequate for f'_k. Note that we add the null flow f_0, which does not use any resources because the aperiodic flow may be scheduled on idle resources, and thus, avoid stealing resources from periodic flows. Set Q contains the schedules that cannot be stolen by the aperiodic flow (line 3). Based on set Q, in the superframe with period $\hat{p}_k$ (line 4), if all transmissions of flow f'_k can be assigned available resources without Interferences (1) and (2) (lines 5 and 6), then flow f'_k is schedulable (lines 7 through 9). A superframe is periodically repeated, and a transmission is scheduled at time slots $t + \hat{p}_k \times g$, $(g \in [0, \frac{T}{\hat{p}_k}))$. Thus, these time slots must be checked to determine the existence of interference.

The number of iteration of **for** loop in line 2 is $O(|F|)$. The time complexity of lines 4 through 9 is $O(T)$. Thus, the time complexity of Algorithm 1 is $O(|F|T)$. For each aperiodic flow, Algorithm 1 must be invoked. Thus, the time complexity of processing all aperiodic flows is $O(|F'||F|T)$.

V. Channel Assignment at the Top Level

In this section, we introduce our algorithm, which isolates overlapping subnetworks by assigning different channels to different subnetworks.

A. Entire Network Model and Problem Statement

The entire network contains multiple subnetworks $\{w^1, w^2, \ldots\}$. Subnetwork w^i requires m^i channels. If subnetworks w^i and w^j overlap, e^{ij} is equal to 1; otherwise $e^{ij} = 0$. Thus, when $e^{ij} = 1$, subnetworks w^i and w^j cannot use the same channel. The main objective of the channel assignment is to isolate subnetworks. If unassigned channels exist, they should be assigned such that the subnetwork has additional communication resources to cope with the changes of data flows and invoke the coordinator at the top level as little as possible.

We define the binary variable y^{ij} to represent the assignment. If $y^{ij} = 1$, the jth channel is assigned to subnetwork w^i; otherwise, it is not. The problem that we address at the top level is stated as follows: Given multiple subnetworks $W = \{w^1, w^2, \ldots\}$, the number of required channels $M = \{m^1, m^2, \ldots\}$, and their overlapping relationship $E = \{e^{ab} | \forall w^a, w^b \in W\}$, our objective is to assign channels to subnetworks, such that every subnetwork can use as many channels as possible in addition to those required; that is

$$\max z \tag{1}$$

where $\forall w^i \in W, z \leq \sum_{\forall g \in [1,16]} y^{ig} - m^i$. The maximizing problem should respect the constraint that overlapping subnetworks cannot use the same channel. Two subnetworks cannot overlap with each other ($e^{ij} = 1$), and they all use the gth channel ($y^{ig} = y^{jg} = 1$). Thus

$$\forall w^i, w^j \in W \forall g \in [1, 16], e^{ij} + y^{ig} + y^{jg} < 3.$$

B. Channel Assignment Algorithm

Our proposed channel assignment algorithm CA is shown in Algorithm 2, in which the function *SI(M, Y)* is invoked to assign the given number of channels to subnetworks. The parameter M is the set of channels, and the parameter Y contains the assignment results. In Algorithm 2, the assigned number of channels is first set to the submitted requirements from the bottom level (line 1). If the function *SI()* cannot find a feasible solution to satisfy the initial requirements, the entire network is unschedulable (line 2). In this case, the entire network should be redeployed. This issue exceeds the scope of our paper and is not considered here. If the initial requirements can be satisfied, the assigned number of channels of all subnetworks is increased by one, and *SI()* assigns channels according to the new numbers of channels. Repeat the process of increasing channels until *SI()* cannot find a feasible solution (lines between 3 and 5). Thus, the feasible solution is the previous M; that is all m^i subtract one (line 6). This simple search method is optimal for (1) (as shown in Theorem 2). If some idle channels exist (line 9), they are assigned to subnetworks according to the decreasing order of the subnetwork utilization u^i (lines 6 through 8). Because a subnetwork with higher utilization provides less idle resources that can be used to address changes, it should be assigned the idle channels in a preferential manner (lines between 7 and 10).

The subnetwork isolation function *SI()* is similar to traditional minimum vertex coloring in which channels correspond to colors. The number of required channels is equal to the number of

Algorithm 2: Channel Assignment *CA*.

Require: W, M, E
Ensure: $Y = \{y^{ig} | \forall w^i \in W, \forall g \in [1, 16]\}$
1: $M = \{m^i | \forall w^i \in W\}$, all m^i are the submitted requirements from the bottom level;
2: **if** $!SI(M, Y)$ **then return** unschedulable;
3: **repeat**
4: $\quad \forall m^i \in M, m^i + +$;
5: **until** $!SI(M, Y)$;
6: $\forall m^i \in M, m^i - -; u^i = \frac{\Gamma^i}{m^i}$;
7: sort subnetworks according to the decreasing order of their utilization, where w^1 has the highest utilization u^1;
8: **for** $i = 1$ to $|W|$ **and** $g = 1$ to 16 **do**
9: $\quad$ **if** $y^{ig} + \sum_{\forall w^j \in W} (e^{ij} \times y^{jg}) == 0$ **then**
10: $\quad\quad y^{ig} = 1; m^i + +$;
11: $\quad$ **return** Y;

assigned colors for each subnetwork. If two subnetworks overlap, they are not assigned the same colors. We use the classical DSATUR strategy [36] to solve the problem. First, the function *SI()* calculate the metric $\delta^i = m^i + \sum_{\forall w^j \in W} (e^{ij} \times m^j)$ for each subnetwork. Then, the subnetworks are sorted in decreasing order of δ^i, where w^1 has the largest metric. For each subnetwork w^i (where i ranges from 1 to $|W|$), the possible least channels are assigned. This simple method is very effective in our system, as illustrated in the evaluation discussed in Section VI. The upper bound of the number of channels used to isolate subnetworks is

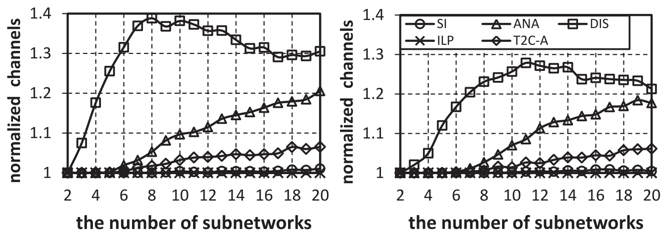
$$\max_{\forall i \in [1, |W|]} \left\{ \min \left\{ \delta^i, \sum_{\forall j \in [1,i]} m^j \right\} \right\}. \tag{2}$$

Because the number of channels needed to isolate subnetwork w^i from other networks is less than or equal to δ^i and because the total number of channels used to isolate i subnetworks cannot exceed $\sum_{\forall j \in [1,i]} m^j$.

The time complexity of *SI()* is $O(|W|^2)$. In Algorithm 2, the number of iterations of the repeat loop in line 3 is less than 16, and the time complexity of lines 8 through 11 is also $O(|W|^2)$. Thus, the time complexity of Algorithm 2 is $O(|W|^2)$.

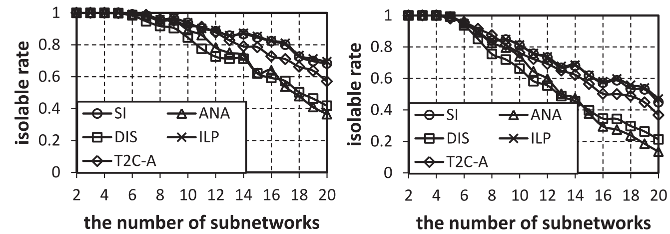
Theorem 2: Excluding the impact caused by the subnetwork isolation function, the search method of Algorithm 2 (lines 2 through 5) is optimal for (1).

Proof: We assume that the original channel requirement from the bottom level is $\{m^1, m^2, \ldots\}$. In the ath iteration of the repeat loop in line 3, the requirement $\{m^1 + a, m^2 + a, \ldots\}$ is checked by *SI()*. If the repeat loop is jumped out when the requirement is $\{m^1 + a + 1, m^2 + a + 1, \ldots\}$, the value a is the optimized result, which corresponds to the variable z in (1). We aim to prove that the only optimized result is the value a, regardless of the search method. We assume the optimized result b ($b > a + 1$) and that the requirement $\{m^1 + b + \Delta^1, m^2 + b + \Delta^2, \ldots\}$ can be satisfied by *SI()*, where $\Delta^* \geq 0$. In *SI()*, for the



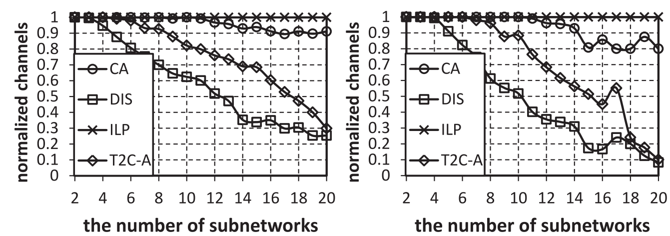
(a) (b)

Fig. 6. Number of channels used to isolate subnetworks. (a) $\rho = 1$. (b) $\rho = 2$.



(a) (b)

Fig. 7. Isolable rates of Fig. 6. (a) $\rho = 1$. (b) $\rho = 2$.



(a) (b)

Fig. 8. Objective [see (1)] comparison. (a) $\rho = 1$. (b) $\rho = 2$.

requirement $\{m^1 + a + 1, m^2 + a + 1, ...\}$, the first subnetwork is assignable because $m^1 + b + \Delta^1$ channels can be satisfied. We assume that the $(i-1)$th subnetwork is also assignable. Relative to the requirement $\{m^1 + b + \Delta^1, m^2 + b + \Delta^2, ...\}$, the previous $(i-1)$ subnetworks use fewer channels. When the ith subnetworks is processed, more channels are idle. Thus, because $m^i + b + \Delta^i$ channels are assigned to the ith subnetwork, $m^i + a + 1$ channels can also be assigned, and the requirement $\{m^1 + a + 1, m^2 + a + 1, ...\}$ can be satisfied. This finding contradicts the previous statement. Therefore, a better optimized result b does not exist. ■

VI. Evaluation

In this section, we describe the simulations and experiments conducted to evaluate the performance of our proposed methods.

A. Simulation

First, we evaluate our proposed channel assignment algorithm and data flow scheduling algorithm. Then, we compare our hierarchical framework to distributed and centralized frameworks.

1) Channel Assignment: Our test cases are randomly generated according to the subnetwork-density ρ. The number of required channels in each subnetwork is randomly selected in the range $[2, 5]$, and the total number of channels is 16. We compare our channel assignment method *CA*, subnetwork isolation method *SI*, and upper bound *ANA* [see (2)] against the following three methods: 1) *DIS*, in which each subnetwork chooses the idle channels to use when it starts to run; (2) *ILP*, in which the integer linear programming (ILP) solver CPLEX is used to find the optimal solution; and (3) *T2C-A*, which denotes the channel assignment method in the T^{2}C network protocol [37]. Some other recent works of channel assignment (e.g., [38] and [39]) were aimed at reducing communication interference but did not isolate nodes (or subnetworks). Thus, we chose [37] for the comparison.

Fig. 6 shows the number of channels used to isolate multiple subnetworks. Each point is the average of 500 test cases. The test cases in Figs. 7 and 8 are the same as those in Fig. 6. All results are normalized by the ILP. The SI method is very similar to the optimal solution ILP, and the upper bound ANA introduces 20% pessimism. The T2C-A method is a centralized heuristic method that produces better results than DIS but poorer results than our SI method. Fig. 7 shows the isolable rate, which is defined as the percentage of test cases for which a method can find an isolable assignment. The SI method is better than T2C-A and highly similar to the optimal ILP. As the density ρ increases, the overlaps among subnetworks become more complex. As a result, 1) the isolable rate decreases in Fig. 7; and 2) the baseline ILP requires more channels to isolate these subnetworks, and other normalized results decrease, as shown in Fig. 6. As the number of subnetworks increases, other methods tend to deviate from the optimal solutions because of the complexity. In Fig. 6 when the number of subnetworks exceeds 12, the DIS method's performance decreases because the maximum number of channels is 16. The results cannot exceed the fixed upper bound, and the baseline increases. Thus, the normalized DIS decreases.

Fig. 8 shows the objective [see (1)] comparison of these methods. The distributed DIS method produces the worst results, and the centralized T2C-A method's performance falls between those of DIS and our proposed CA. In the worst case, our CA algorithm's result is 20% less than the optimal solution ILP. In most cases, however, our CA algorithm is close to the optimal solution.

2) Scheduling Algorithm: Each subnetwork is generated based on the equation $A = \frac{|N| d^2 \sqrt{27}}{2\pi}$ [40], where A is the square area, d denotes the transmitting range of 40 m, and $|N|$ is the number of nodes. Each node is randomly deployed in the square area and has a data flow between itself and its controller. The unit period $p_{\min}$ is 25 [7], [8]. We specify the total workload Γ and use the UUniFast method [41] to assign the upper workload bound for each data flow. We calculate the minimal period of each data flow that satisfies the expression $p_{\min} \times 2^a$ and causes the flow's workload to be less than the assigned upper bound of the workload. Aperiodic flows account for 10% of all flows, and the trigger probability per time slot is set to 1%.

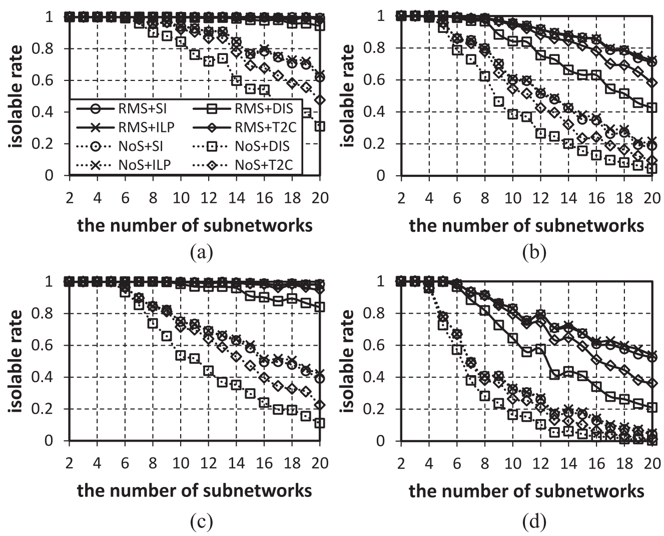


Fig. 9. Comparison of isolable rates among scheduling algorithms. (a) $\Gamma = 3, \rho = 1$. (b) $\Gamma = 5, \rho = 1$. (c) $\Gamma = 3, \rho = 2$. (d) $\Gamma = 5, \rho = 2$.

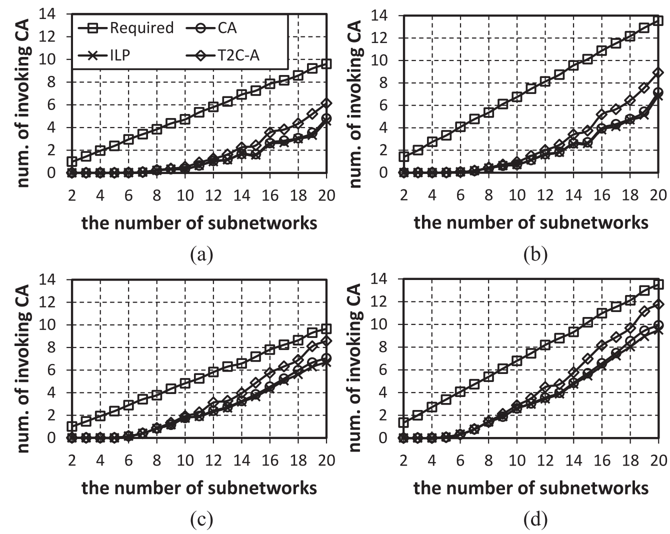


Fig. 11. Number of invoking CA. (a) $\Gamma : 3 \rightarrow 4, \rho = 1$. (b) $\Gamma : 3 \rightarrow 5, \rho = 1$. (c) $\Gamma : 3 \rightarrow 4, \rho = 2$. (d) $\Gamma : 3 \rightarrow 5, \rho = 2$.

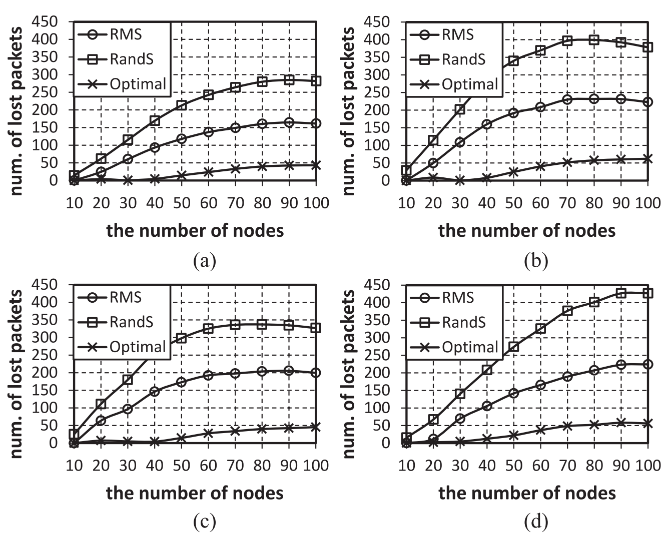


Fig. 10. Comparison of the number of lost packets. (a) $\Gamma = 3, \sigma = 10\%, \vartheta = 1‰$. (b) $\Gamma = 3, \sigma = 10\%, \vartheta = 2‰$. (c) $\Gamma = 3, \sigma = 20\%, \vartheta = 1‰$. (d) $\Gamma = 5, \sigma = 10\%, \vartheta = 1‰$.

The isolate rates are compared in Fig. 9. We compare our proposed scheduling algorithm *RMS* against the *NoS* method in which any two aperiodic flows are not simultaneously transmitted, and aperiodic flows cannot steal the communication resources from periodic flows. The number of nodes contained in a subnetwork is randomly chosen in the range $[10, 100]$. Because the NoS method requires more channels to schedule flows, its isolable rate is far less than that of our RMS method, regardless of the parameter and channel assignment method used. To isolate multiple subnetworks in a large-scale network, the stealing strategy is a better method.

Our RMS algorithm causes data packet losses. We run each test case for 25 600 time slots, and each point in Fig. 10 is the average number of lost packets in 500 test cases. The comparison methods include 1) *RandS*, in which an aperiodic packet is immediately sent and randomly steals channel resources once it is released; and 2) *Optimal*, in which the solution space is completely searched, and the optimal solution can be identified. The parameter σ denotes the proportion of aperiodic flows, and the parameter ϑ denotes the possibility of releasing an aperiodic packet at each time slot. Our RMS algorithm is half of RandS; thus, its reliability is higher than that of RandS. As the subnetwork size increases, the increasing trend decreases because the node interference becomes increasingly complex. This interference produces additional idle resources, which can be used to schedule aperiodic flows; i.e., the aperiodic packet does not need to steal from periodic flows.

3) Hierarchical framework: Our proposed CA algorithm assigns as many channels as possible to each subnetwork. This strategy makes subnetworks more amenable to changes. Fig. 11 shows the number of networks that invoke the channel assignment when the assigned channels cannot cope with data flow changes. Based on the workload increase ($3 \rightarrow 4$ and $3 \rightarrow 5$), we randomly choose data flows and change their periods. Other parameters and the generation of test cases are the same as those described in the previous subsection. Each point is also the average of 500 test cases. In the comparison method *Required*, each subnetwork is only assigned the required channels. The results of our CA algorithm are similar to the optimized results obtained with ILP. When the subnetwork density ρ is 1, our proposed algorithm can reduce the networks invoking CA by more than half. When the density ρ is 2, few extra channels are assigned to each subnetwork. Thus, our CA algorithm's results draw close to Required. The T2C-A method deviates farther from the optimal result as the number of subnetworks increases because as this number increases, the number of channels assigned by T2C-A decreases (as shown in Fig. 8).

Our hierarchical framework can effectively reduce the delay by eliminating long-distance end-to-end transmission. Fig. 12 compares the delays between our hierarchical framework (*Hier.*) and a centralized framework (*Cent.*). AP denotes the number of

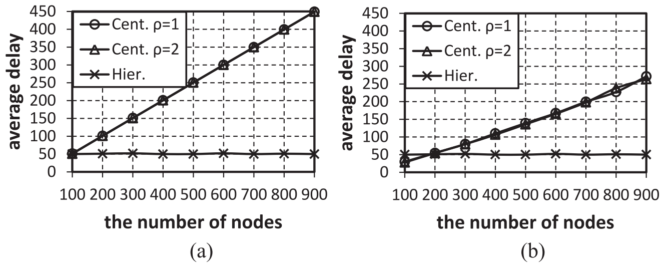


Fig. 12. Average delay. (a) $AP = 1$. (b) $AP = 2$.

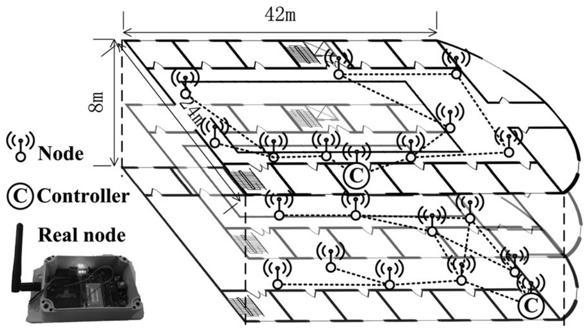


Fig. 13. Our real testbed.

access points in a centralized network. For each point in the figure, 500 test cases are executed. The density has no impact on the delay because access points are the bottleneck. When AP decreases from 2 to 1, the delay reduces significantly. Our subnetworks contain 100 nodes. Thus, our average delay is approximately 50 time slots. Regardless of the number of nodes in a large-scale network, the delay in our proposed architecture is the same as that in a small-scale network.

B. Experiment

We deploy two subnetworks in our building, as shown in Fig. 13. Our network node is implemented on MSP430 and CC2420 chips. We compare the packet arrival rates achieved with and without an isolation strategy; the transmission powers are set to 0 and -15 dBm, respectively. For each test, the testbed collects a continuous arrival rate over a 4-h period. The arrival rates are shown in Fig. 14. The average arrival rates of the four tests are 86.7% (with isol. 0 dBm), 64.3% (without isol. 0 dBm), 71.0% (with isol. -15 dBm) and 60.8% (without isol. -15 dBm). Comparing the two isolation tests reveals that the average arrival rate decreases by 15.7% because of the decrease in the transmission power. If the two subnetworks are not isolated, the arrival rate is approximately 60%. The use of an isolation strategy can improve reliability by increasing the arrival rate by 22%. Note that transmissions of small subnetworks is distributed across all 16 channels and that the utilization of each subnetwork is approximate 11%. Overlapping interference between the two subnetworks seldom occurs. Even in this situation, the overlapping interference produces 22% packet loss. As the subnetwork size increases, the packet losses will become significant.

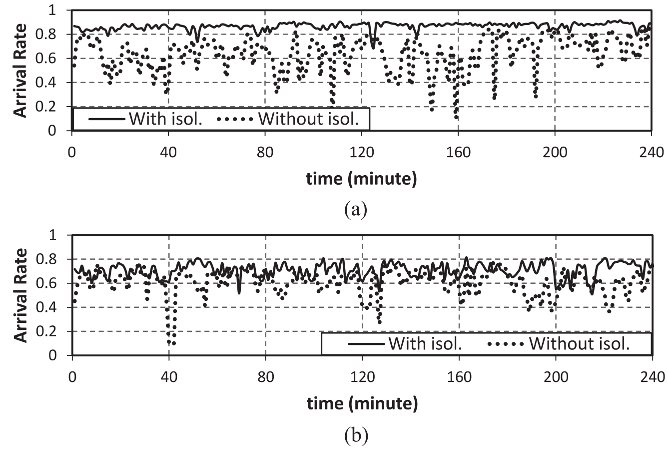


Fig. 14. Packet arrival rate. (a) 0 dBm. (b) -15 dBm.

VII. Conclusion

Smart factories require advanced data transmission frameworks to support real-time, reliable, and large-scale communications. This paper proposes a hierarchical framework to optimize these three aspects. The top level of the hierarchy coordinates communication resources among subnetworks and aims to improve scalability and reliability. The bottom level schedules data flows in each subnetwork to improve the real-time performance and reliability. We performed extensive simulations and realistic testbed experiments. The results confirm the efficiency of our methods. In the future, we will extend our research to address additional scenarios, such as when aperiodic flows are allowed to interfere with each other and when graph routing schemes are supported.

References

[1] H. Song, D. B. Rawat, S. Jeschke, and C. Brecher, *Cyber-Physical Systems: Foundations, Principles and Applications*. Boston, MA, USA: Academic, 2016.

[2] S. Jeschke, C. Brecher, H. Song, and D. B. Rawat, *Industrial Internet of Things: Cybermanufacturing Systems*. Cham, Switzerland: Springer, 2016.

[3] Y. Wang, W. Wei, Q. Deng, W. Liu, and H. Song, "An energy-efficient skyline query for massively multidimensional sensing data," *Sensors*, vol. 16, no. 1, pp. 1–19, 2016.

[4] X. Li, D. Li, J. Wan, A. V. Vasilakos, C.-F. Lai, and S. Wang, "A review of industrial wireless networks in the context of industry 4.0," *Wireless Netw.*, vol. 23, pp. 1–19, 2015.

[5] W. Dong, Y. Liu, Y. He, T. Zhu, and C. Chen, "Measurement and analysis on the packet delivery performance in a large-scale sensor network," *IEEE/ACM Trans. Netw.*, vol. 22, no. 6, pp. 1952–1963, Dec. 2014.

[6] T. Luo, H.-P. Tan, and T. Q. Quek, "Sensor OpenFlow: Enabling software-defined wireless sensor networks," *IEEE Commun. Lett.*, vol. 16, no. 11, pp. 1896–1899, Nov. 2012.

[7] International Electrotechnical Commission, *Industrial Communication Networks—Wireless Communication Network and Communication Profiles–Wirelesshart*, IEC 62591, 2009.

[8] International Electrotechnical Commission, *Industrial Networks—Wireless Communication Network and Communication Profiles—WIA-PA*, IEC 62601, 2015.

[9] International Electrotechnical Commission, *Industrial Networks—Wireless Communication Network and Communication Profiles—ISA 100.11a*, IEC 62734, 2014.

[10] R. Kitchin, "The real-time city? Big data and smart urbanism," *GeoJournal*, vol. 79, no. 1, pp. 1–14, 2014.

[11] Z. Lv, H. Song, P. Basanta-Val, A. Steed, and M. Jo, "Next-generation big data analytics: State of the art, challenges, and future research topics," *IEEE Trans. Ind. Informat.*, doi: 10.1109/TII.2017.2650204, pp. 1–9, 2017.
[12] A. Zaslavsky, C. Perera, and D. Georgakopoulos, "Sensing as a service and big data," in *Proc. Int. Conf. Advances Cloud Comput.*, 2011, pp. 1–8.
[13] P. Basanta-Val, N. Fernández-García, A. J. Wellings, and N. C. Audsley, "Improving the predictability of distributed stream processors," *Future Gener. Comput. Syst.*, vol. 52, pp. 22–36, 2015.
[14] P. Basanta-Val, N. C. Audsley, A. J. Wellings, I. Gray, and N. Fernández-García, "Architecting time-critical big-data systems," *IEEE Trans. Big Data*, vol. 2, no. 4, pp. 310–324, Dec. 1, 2016.
[15] P. Basanta-Val and M. García-Valls, "A distributed real-time Java-centric architecture for industrial systems," *IEEE Trans. Ind. Informat.*, vol. 10, no. 1, pp. 27–34, Feb. 2014.
[16] P. Basanta-Val, M. García-Valls, and M. B. Cuñado, "A simple data-muling protocol," *IEEE Trans. Ind. Informat.*, vol. 10, no. 2, pp. 895–902, May 2014.
[17] F. Ingelrest, G. Barrenetxea, G. Schaefer, M. Vetterli, O. Couach, and M. Parlange, "Sensorscope: Application-specific sensor network for environmental monitoring," *ACM Trans. Sensor Netw.*, vol. 6, no. 2, pp. 1–32, 2010.
[18] X. Ding, Y. Tian, and Y. Yu, "A real-time big data gathering algorithm based on indoor wireless sensor networks for risk analysis of industrial operations," *IEEE Trans. Ind. Informat.*, vol. 12, no. 3, pp. 1232–1242, Jun. 2016.
[19] A.-C. Anadiotis, G. Morabito, and S. Palazzo, "An SDN-assisted framework for optimal deployment of mapreduce functions in WSNs," *IEEE Trans. Mobile Comput.*, vol. 15, no. 9, pp. 2165–2178, Sep. 2016.
[20] L. Galluccio, S. Milardo, G. Morabito, and S. Palazzo, "SDN-WISE: Design, prototyping and experimentation of a stateful SDN solution for wireless sensor networks," in *Proc. IEEE Conf. Comput. Commun.*, 2015, pp. 513–521.
[21] D. Zeng, P. Li, S. Guo, T. Miyazaki, J. Hu, and Y. Xiang, "Energy minimization in multi-task software-defined sensor networks," *IEEE Trans. Comput.*, vol. 64, no. 11, pp. 3128–3139, Nov. 2015.
[22] J. Zhou, H. Jiang, J. Wu, L. Wu, C. Zhu, and W. Li, "SDN-based application framework for wireless sensor and actor networks," *IEEE Access*, vol. 4, no. 3, pp. 1583–1594, Mar. 2016.
[23] J. Wang, W. Dong, Z. Cao, and Y. Liu, "On the delay performance in a large-scale wireless sensor network: Measurement, analysis, and implications," *IEEE/ACM Trans. Netw.*, vol. 23, no. 1, pp. 186–197, Feb. 2015.
[24] P. Dutta *et al.*, "Trio: enabling sustainable and scalable outdoor wireless sensor network deployments," in *5th Int. Conf. Inf. Process. Sensor Netw.*, 2006, pp. 407–415.
[25] A. Arora *et al.*, "Exscal: Elements of an extreme scale wireless sensor network," in *Proc. 11th IEEE Int. Conf. Embedded Real-Time Comput. Syst. Appl.*, 2005, pp. 102–108.
[26] P. Vicaire *et al.*, "Achieving long-term surveillance in VigilNet," *ACM Trans. Sensor Netw.*, vol. 5, no. 1, 2009, Art. no. 9.
[27] H. Tian, "The engineer's guide to industrial wireless measurement," Emerson Electric Company, Tech. Rep., 2014.
[28] A. Saifullah, Y. Xu, C. Lu, and Y. Chen, "Real-time scheduling for wirelesshart networks," in *Proc. IEEE 31st Real-Time Syst. Symp.*, 2010, pp. 150–159.
[29] X. Jin, Q. Zhang, P. Zeng, F. Kong, and Y. Xiao, "Collision–free multichannel superframe scheduling for IEEE 802.15.4 cluster–tree networks," *Int. J. Sensor Netw.*, vol. 15, no. 4, pp. 246–258, 2014.
[30] X. Jin, F. Kong, P. Zeng, Q. Deng, and H. Xu, "Joint management of energy harvesting, storage, and usage for green wireless sensor networks," *Int. J. Distrib. Sensor Netw.*, vol. 2014, pp. 1–13, 2014.
[31] S. Yu, X. Zhang, and W. Liang, "Concurrent transmission performance modeling of wireless multimedia sensor network and its experimental evaluation," *Inf. Control*, vol. 45, no. 3, pp. 328–334, 2016.
[32] W. Shen, T. Zhang, F. Barac, and M. Gidlund, "PriorityMAC: A priority-enhanced MAC protocol for critical traffic in industrial wireless sensor and actuator networks," *IEEE Trans. Ind. Informat.*, vol. 10, no. 1, pp. 824–835, Feb. 2014.
[33] B. Li, L. Nie, C. Wu, H. Gonzalez, and C. Lu, "Incorporating emergency alarms in reliable wireless process control," in *Proc. Int. Conf. Cyber-Phys. Syst.*, 2015, pp. 218–227.
[34] J. N. Al-Karaki and A. E. Kamal, "Routing techniques in wireless sensor networks: A survey," *IEEE Wireless Commun.*, vol. 11, no. 6, pp. 6–28, Dec. 2004.
[35] N. A. Pantazis, S. A. Nikolidakis, and D. D. Vergados, "Energy-efficient routing protocols in wireless sensor networks: A survey," *IEEE Commun. Surveys Tut.*, vol. 15, no. 2, pp. 551–591, Second Quarter 2013.
[36] D. Brélaz, "New methods to color the vertices of a graph," *Commun. ACM*, vol. 22, no. 4, pp. 251–256, 1979.
[37] X. Zhu, X. Tao, T. Gu, and J. Lu, "Target-aware, transmission power-adaptive, and collision-free data dissemination in wireless sensor networks," *IEEE Trans. Wireless Commun.*, vol. 14, no. 12, pp. 6911–6925, Dec. 2015.
[38] J. Chen, Q. Yu, B. Chai, Y. Sun, Y. Fan, and X. S. Shen, "Dynamic channel assignment for wireless sensor networks: A regret matching based approach," *IEEE Trans. Parallel Distrib. Syst.*, vol. 26, no. 1, pp. 95–106, Jan. 2015.
[39] Y. Wu, K. S. Liu, J. A. Stankovic, T. He, and S. Lin, "Efficient multichannel communications in wireless sensor networks," *ACM Trans. Sensor Netw.*, vol. 12, no. 1, pp. 1–23, 2016.
[40] T. Camilo, J. S. Silva, A. Rodrigues, and F. Boavida, "GENSEN: A topology generator for real wireless sensor networks deployment," in *Proc. IFIP Int. Workshop Softw. Technol. Embedded Ubiquitous Syst.*, 2007, pp. 436–445.
[41] E. Bini and G. C. Buttazzo, "Measuring the performance of schedulability tests," *Real-Time Syst.*, vol. 30, no. 1-2, pp. 129–154, 2005.

Xi Jin received the Ph.D. degree in computer science from Northeastern University, Shenyang, China, in 2013.

She is currently an Associate Professor with the Shenyang Institute of Automation, Chinese Academy of Sciences, Shenyang. Her research interests include industrial networks, wireless sensor networks, real-time systems, and cyber-physical systems.

Fanxin Kong received the Ph.D. degree in computer science from McGill University, Montreal, Canada, in 2016.

He is currently a Postdoctoral Researcher with the Department of Computer and Information Science, University of Pennsylvania, Philadelphia, PA, USA. His research interests include the areas of cyber-physical systems and Internet of things, cloud computing and edge computing, and smart grid and energy systems. He develops and applies techniques from game theory, distributed computing, machine learning, data analysis, deterministic and stochastic optimization, and approximate and online algorithm design for these areas.

Linghe Kong (S'09–M'13) received the B. E. degree in automation from Xidian University, Xi'an, China, in 2005, the Master degree in telecommunication from TELECOM SudParis, Évry, France, in 2007, and the Ph.D. degree in computer science from Shanghai Jiao Tong University, Shanghai, China, in 2012.

He is currently a Research Professor with the Department of Computer Science and Engineering, Shanghai Jiao Tong University. Before that, he was a Postdoctoral Researcher at Columbia University, McGill University, and Singapore University of Technology and Design. His research interests include wireless communication, sensor networks, mobile computing, Internet of Things, big data, and smart energy systems.

Huihui Wang (M'13) received the B.S. degree from the Environment Science Department, Northwestern Polytechnical University, Xi'an, China, in 2004, the M.S. degree from the Mechanical Engineering Department, Xi'an Jiaotong University, Xi'an, China, in 2007, and the Ph.D. degree from the Electrical and Computer Engineering Department, University of Virginia, Charlottesville, VA, USA, in 2013.

She joined the faculty of Jacksonville University, Jacksonville, FL, USA, in August 2013, where she is currently an Advisor of the National Society of Professional Engineering.

Dr. Wang is the member of the American Society for Engineering Education, American Society of Mechanical Engineers, and Florida Engineering Society. She has served as a Technical Program Committee Member and a Reviewer for international conferences and journals. Her research interests include cyber-physical systems, Internet of Things, and robotics

Changqing Xia received the Ph.D. degree in computer science from Northeastern University, Shenyang, China, in 2015.

He is currently an Assistant Professor with the Shenyang Institute of Automation, Chinese Academy of Sciences, Shenyang. His research interests include wireless sensor networks and real-time systems, especially real-time scheduling algorithms and smart energy systems.

Peng Zeng received the B.S. degree in computer science from Shandong University, Shandong, China, in 1998 and the Ph.D. degree in mechatronic engineering from the Shenyang Institute of Automation, Chinese Academy of Sciences, Shenyang, China, in 2005.

He is currently a Professor with the Shenyang Institute of Automation, Chinese Academy of Sciences. His current research interests include wireless sensor networks for industrial automation, smart grids, and demand response.

Qingxu Deng received the Ph.D. degree in computer science from Northeastern University, Shenyang, China, in 1997.

He is a Professor with the College of Information Science and Engineering, Northeastern University, where he serves as the Director of the Institute of Cyber-Physical Systems Engineering. His research interests include cyber-physical systems, embedded systems, and real-time systems.