Reliability and Temporality Optimization for Multiple Coexisting WirelessHART Networks in Industrial Environments

Xi Jin, Fanxin Kong, *Student Member, IEEE*, Linghe Kong, *Member, IEEE*, Wei Liu, and Peng Zeng

***Abstract*—WirelessHART is a networking technology that is widely used in industrial wireless sensor networks. Its reliability and real-time performance are essential to industrial production. Many works have studied these two aspects, primarily focusing on a single WirelessHART network. However, multiple WirelessHART networks usually coexist in a real industrial environment. Applying existing approaches to such coexisting networks would cause performance degradation due to communication interference among these networks. In this paper, we propose a holistic framework that optimizes both reliability and temporality for multiple coexisting networks. The framework consists of two levels. The upper level targets communication channel management, and the lower level addresses data flow scheduling. For the upper level, we propose a network isolation algorithm that improves the data transmission reliability through dynamically adjusting channel assignments to different WirelessHART networks. For the lower level, we propose data flow scheduling algorithms that guarantee the temporality of data flows within each isolated network. These algorithms minimize the number of channels reserved by each isolated network and further enhance the transmission reliability through alleviating channel resource contention. We conduct trace-driven simulations of the channel management algorithm, and the results demonstrate that our algorithm exhibits stable performance and reduces packet loss by 36%. For the scheduling algorithms, the simulations demonstrate that in contrast with existing algorithms, the greater the number of coexisting networks, the fewer resources our algorithms use. When eight networks coexist, our algorithms outperform existing ones by consuming up to 63% fewer channel resources.**

***Index Terms*—Channel management, industrial environment, scheduling algorithms, wireless sensor networks.**

Manuscript received May 25, 2016; revised December 13, 2016 and February 9, 2017; accepted February 21, 2017. Date of publication March 14, 2017; date of current version July 10, 2017. This work was supported in part by the National Natural Science Foundation of China under Grant 61502474, Grant 61233007, and Grant 61501447, and in part by the Youth Innovation Promotion Association of the Chinese Academy of Sciences. *(Corresponding author: Peng Zeng.)*

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

W. Liu is with the Department of Informatics, Technical University of Dortmund, 44227 Dortmund, Germany (e-mail: liuwei-neu@hotmail.com).

Digital Object Identifier 10.1109/TIE.2017.2682005

I. Introduction

WIRELESSHART [1]–[3] is a wireless networking technology that has been widely used in industrial applications, e.g., industrial process control [4], collaborative location and tracking systems [5], and monitoring systems [6], [7]. These applications usually have stringent requirements regarding the reliability and real-time performance of WirelessHART networks. Data packet loss or delayed packets can cause catastrophic consequences. For example, in the cement manufacturing industry, the temperature data of rotary kilns must be sent to the control room within a deadline. If a data packet with a high-temperature message is lost or delayed, the kilns can explode. To enhance the reliability and temporality of WirelessHART networks, various methods have been proposed, such as real-time scheduling algorithms [8], [9], end-to-end delay analysis [10], and a reliable routing algorithm [11]. However, all of these methods focus on a single WirelessHART network.

In real industrial environments, there are often multiple coexisting WirelessHART networks. The scale of a WirelessHART network is limited to 100 sensor/actuator devices because a network manager has limited computing capacity [12], [13]. A large-scale industrial application usually requires more than 100 such devices and thus involves multiple WirelessHART networks. A canonical scenario involving coexisting networks is as follows. Data flows are transmitted between two devices that are separated by obstructions such as walls, ceilings, or floors. A low penetration ability would result in significant packet loss and transmission delay, which would thus compromise the reliability and temporality of the network. To avoid these negative consequences, an effective approach is to construct WirelessHART networks for the devices on each side of an obstruction. Each WirelessHART network is individually managed by a centralized network manager, and these network managers are connected with each other by a robust communications medium, such as a wired network. The same structure is proposed by the EMERSON company [12].

Fig. 1 illustrates the above two scenarios with three coexisting WirelessHART networks in a factory. Their coverage areas overlap, and this overlapping causes communication

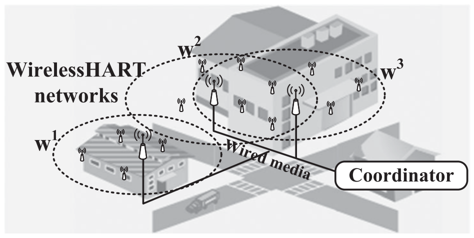

Fig. 1. Multiple WirelessHART networks coexist in a factory.

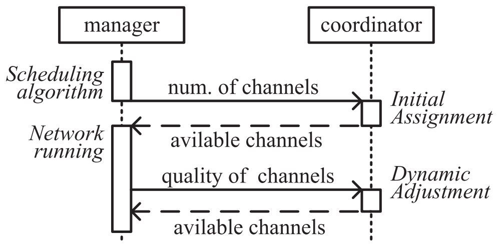

Fig. 2. Operation mechanism.

interference among these networks. Channel hopping and blacklisting can reduce partial interference, but the benefits are limited. In this paper, we focus on the issue of how to optimize the reliability and temporality of multiple coexisting networks. Commonly, two types of methods are used to solve this issue: centralized and distributed. For centralized methods, managing multiple coexisting networks as an entire entity will overload the centralized network manager. A WirelessHART gateway can support at most 100 nodes [12], [13]. However, there can be thousands of wireless devices in a factory. A gateway cannot manage the entire network. Even when the centralized management can be implemented in a high-speed computing unit, the runtime of management programs for a large-scale network is unacceptable. For distributed methods, we might apply single-network approaches, such as those previously presented [8], [10], [14], [15] to each coexisting network; then, the network considers the interference from neighboring networks external interference and manages its own communications to avoid the interference. However, this management is based on limited local information, and it is difficult to optimize the results. To address these issues, we propose a tradeoff between these two methods—a novel, two-level framework that optimizes both reliability and temporality. The lower level addresses data flow scheduling in a single WirelessHART network. The upper level does not manage data flows but instead targets centralized channel resource management. Specifically, we make the following contributions.

First, we propose two flow-scheduling algorithms for the lower level. The objective of the first algorithm is to minimize the number of channels required by scheduling flows in a single network. The second algorithm not only optimizes the number of channels but also enhances the temporality of data flows. For these algorithms, we provide performance guarantees through rigorous theoretical analysis. Based on the results of these algorithms, the upper level can eliminate channel contention and improve data transmission reliability. The simulation results indicate that the greater the number of coexisting networks, the fewer resources our algorithms use compared with the existing algorithms, and our algorithms outperform existing ones by using up to 63% fewer channel resources.

Second, we propose a network isolation algorithm for the upper level. To avoid external interference and enhance the data transmission reliability, we propose an algorithm to dynamically adjust channel assignments. The algorithm significantly reduces packet loss through eliminating the communication interference among different networks. Using data traces collected from a real test-bed, we conduct trace-driven simulations for the channel management algorithm. The results indicate that the proposed algorithm exhibits stable performance and reduces packet loss by 36%.

II. System Overview

The system consists of two levels. The lower level corresponds to a single WirelessHART network, and the upper level is the entire network. In our system model, superscripts of symbols are used to distinguish different WirelessHART networks in the upper level, and subscripts indicate different elements in a WirelessHART network in the lower level. The entire network W contains multiple WirelessHART networks $W = \{w^1, w^2, \ldots\}$. If networks w^i and w^j can interfere with each other when they transmit packets simultaneously on the same channel, then $c^{ij} = 1$; otherwise, $c^{ij} = 0$. Each WirelessHART network w^i is equipped with a manager. These managers are connected to the coordinator by wired media. In the lower level, the WirelessHART protocol is based on a centralized management and a multichannel time division multiple access scheme and supports 15 nonoverlapping channels. The centralized network manager is responsible for management functions such as scheduling data flows and configuring the network. To enhance reliability, the WirelessHART protocol adopts graph routing [1], [16]. In this paper, all routing graphs were specified before generating schedules. Some previously published routing methods [11], [17] can be used to obtain routing graphs. The manager assigns a time slot and a channel offset to each hop in these routing graphs; the channel offset is used to calculate which channel can be used. Controlled by the manager, each node sends the packet in the assigned time slot and on the assigned channel. A detailed description of WirelessHART networks is presented in Section III. We present the operation mechanism of our system in Fig. 2 and state the problems considered.

In the initial stage, each network manager generates the schedule for all flows under real-time constraint in its own network. If two WirelessHART networks transmit packets on the same channel and their coverage areas overlap, they will interfere with each other, resulting in packet loss. Therefore, we isolate networks by employing different channels. Channel resources are scarce because there are only 15 available channels. Therefore, in the lower level, *our problem is how to schedule real-time data flows using the minimum number of channels in a single network.*

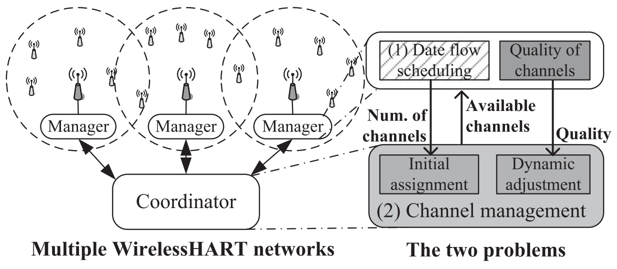

Fig. 3. System overview.

After schedule generation, the network manager determines and submits to the coordinator the number of required channels. Based on these submissions, the coordinator assigns available channels and sends channel assignments to network managers. In industrial environments, there exist wireless communication devices that use the same 2.4 GHz industrial, scientific and medical band with the WirelessHART protocol, e.g., Wi-Fi, Bluetooth, and cordless phones. The external interference is uncontrolled and dynamic. A previously published work demonstrated that infrequent channel offset adjustment can improve reliability [18]. Thus, when the system is running, each manager also sends information about the quality of channels to the coordinator. When the quality is below a given threshold, the coordinator changes the poor quality channel to an unused one and sends the new assignment to the manager. In the upper level, *our problem is how to design the channel management scheme not only to assign channels to isolate multiple networks at the initial stage but also to support dynamic adjustment to alleviate external interference.*

Fig. 3 shows the relationship between the two problems. The pattern block denotes the lower level problem. The algorithm for this problem is only in the network managers. The gray blocks in managers and the coordinator are about the upper level problem. In Section III, we initially propose two scheduling algorithms for the lower level. In Section IV, we then solve the network isolation problem in the upper level.

III. Flow Scheduling for the Lower Level

In this section, we focus on the lower level and describe how to solve the scheduling problem using a minimum number of channels under a real-time constraint in a single WirelessHART Network.

A. Single WirelessHART Network Model

For simplicity, we omit superscripts in this section. We consider a WirelessHART network characterized by $w = \langle N, L, M\rangle$. A WirelessHART network consists of sensor/actuator devices and a gateway that is connected to the centralized network manager. We use a node set $N = \{n_1, n_2, \ldots\}$ to denote them. Each device is equipped with a transceiver, therefore a node does not serve multiple packets simultaneously. If nodes n_i and n_j can directly communicate with each other, the link l_{ij} in set L is equal to 1; otherwise, $l_{ij} = 0$. Transmitting a packet through a link is a transmission. Each transmission is

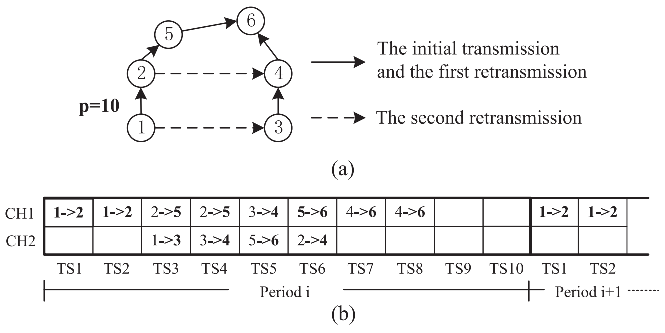

Fig. 4. WirelessHART network model. (a) Graph routing for a flow. (b) Superframe for the subgraph.

followed immediately by its acknowledgment. For simplicity, we call a transmission and its acknowledgment a *transmission* in the following. Not all 15 channels can always be accessed. The channel sets $\bar{M}$ and M denote the blacklist and whitelist, respectively.

The flow set is denoted by $F = \{f_1, f_2, \ldots\}$. These flows are transmitted either from nodes to the gateway or the reverse. Each flow f_i is characterized by $f_i = \langle p_i, \phi_i\rangle$. The first packet of every flow is released at time 0. Then, flow f_i periodically generates a packet at period p_i to send. The relative deadline is equal to the period. The packet released at time t must be delivered to its destination before its deadline $t + p_i$. The network manager assigns a time slot and a channel offset to each transmission. If no packets miss deadlines, the set is schedulable and the assignment is called a feasible schedule.

A graph routing supplies multiple paths from the source node to the destination. Employing multiple paths can avoid packet loss introduced by link failures. Fig. 4(a) shows an example of a routing graph. The directed links constitute the routing graph ϕ_i of flow f_i. The WirelessHART protocol specifies that each node has at most two links from itself to other nodes and attempts to send a packet at most three times (including the initial transmission and two retransmissions). The initial transmission and the first retransmission are on one link, whose type is defined as $L1$, and the second retransmission is on another link, whose type is $L2$. A node can receive the same packet from different nodes, e.g., in Fig. 4(a) node n_4 receives the same packet from nodes n_2 and n_3. In this case, the time slots assigned to node n_4 for sending the packet are after the time slots in which the node n_4 has received all copies of the packet.

Based on the information regarding the network and flows, the network manager schedules transmissions in the time slot and channel dimensions to eliminate interference and satisfy the real-time requirement. These schedules are organized within *superframes* that repeat themselves periodically, as shown in Fig. 4(b). In one superframe, the packet generated by flow f_i should be transmitted from its source node to the destination, and the period of the superframe is equal to the period of flow f_i. The period supported by the WirelessHART protocol is $b \times 2^a$, where a is an integer value and b is the unit period. Flows with the same period are scheduled in the same superframe. Multiple superframes with different periods

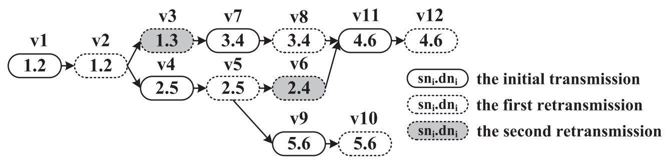
v1
1.2
v2
1.2
v3
1.3
v4
2.5
v5
2.5
v6
2.4
v7
3.4
v8
3.4
v9
5.6
v10
5.6
v11
4.6
v12
4.6
sn_i.dn_i the initial transmission
sn_i.dn_i the first retransmission
sn_i.dn_i the second retransmission

Fig. 5. Subgraph of the flow in Fig. 4.

are used to perform all flow schedules, and they start from the absolution slot number (ASN) 0. These schedules are delivered to devices via the gateway. We consider all superframes in a network hyperframe whose period is the hyperperiod $H = \text{LCM}(p_1, p_2, \ldots) = \max_{\forall f_i \in F}\{p_i\}$. We schedule flows in the first hyperperiod because subsequently all schedules are cyclically repeated.

To enhance reliability, the WirelessHART protocol applies a channel-hopping scheme in which the actual channel used by each transmission must be calculated based on the assigned channel offset as follows:

Actual Channel = (ASN + Channel Offset) **mod** $|M|$.

Because in the same time slot (for the same ASN), a channel offset corresponds to the only actual channel. Therefore, the scheduling algorithm only solves how to assign channel offsets to transmissions; it need not consider the effect of the channel hopping on schedulability. A similar conclusion has been presented in a previously published study [19].

B. Establishing the Releasing Sequence Graph

The graph routing is more complex than single-path routing, and the number of transmissions on the $L1$ and $L2$ links is different. Therefore, we use a graph to denote the releasing sequence of transmissions explicitly. We establish the *releasing sequence graph* $G = \langle V, E\rangle$ for network w and flow set F. Graph G contains $|F|$ connected subgraphs. These subgraphs are disjoint, and each subgraph corresponds to a flow. Fig. 5 shows the subgraph of the flow in Fig. 4. An element v_i in the set V denotes a transmission that needs a time slot to be scheduled, and $v_i = \langle p'_i, sn_i, dn_i\rangle$. p'_i is the period of the flow to which the vertex belongs and is the implicit deadline. sn_i and dn_i are the transmission's source and destination nodes. Element e_{ij} in set E specifies the release order. Vertex v_j is released after all vertices in set $\{v_i | \exists e_{ij} \in E\}$ are scheduled.

The method for establishing the releasing sequence graph is presented in Algorithm 1. We traverse the routing graph of each flow from the root. The transmissions are added to graph G one by one (lines 5–17). For each link in ϕ_k, if the type of the link is $L1$, there are two transmissions (the initial transmission and the first retransmissions) on that link, and two corresponding vertices $\{v_a, v_{a+1}\}$ are added into vertex set V (lines 7–8). If the type of the link is $L2$, there is one transmission (the second retransmission) passing through the link, and only one vertex $\{v_a\}$ is added (lines 14–15). The symbol Ω_i denotes the set $\{v_a | v_a \in G, dn_a = n_i, \forall e_{ab} \in G, dn_b \neq n_i\}$ and $\Omega_i \subseteq V$. The symbol $I_{k,i}$ denotes the in-degree of node n_i in routing graph ϕ_k. When $|\Omega_i|$ is equal to $I_{k,i}$, the packet has been transmitted to node n_i from all paths, and the transmissions from node n_i to

Algorithm 1: Establish the releasing sequence graph G.

```
Require: $w$, $F$
Ensure: $G = \langle V, E\rangle$
 1: $a \leftarrow 1$;
 2: for each $f_k \in F$ do
 3:     set all $\Omega_i$ as $\emptyset$;
 4:     $Q.enqueue$(the root of $\phi_k$); // $Q$ is a FIFO queue
 5:     while !$Q.isEmpty()$ do
 6:         $n_i \leftarrow Q.dequeue()$;
 7:         if $\exists l_{ij} \in \phi_k$ and the type of $l_{ij}$ is $L1$ then
 8:             $V \leftarrow V + \{v_a, v_{a+1}\}$; $E \leftarrow E + \{e_{a,a+1}\}$;
 9:             for each $v_b \in \Omega_i$ do
10:                 $E \leftarrow E + \{e_{ba}\}$;
11:             $\Omega_j \leftarrow \Omega_j + \{v_{a+1}\}$; $a \leftarrow a + 2$;
12:             if $|\Omega_j| = I_{k,j}$ then
13:                 $Q.enqueue(n_j)$;
14:         else if $\exists l_{ij} \in \phi_k$ and the type of $l_{ij}$ is $L2$ then
15:             $V \leftarrow V + \{v_a\}$; $E \leftarrow E + \{e_{a-1,a}\}$;
                $\Omega_j \leftarrow \Omega_j + \{v_a\}$; $a++$;
16:             if $|\Omega_j| = I_{k,j}$ then
17:                 $Q.enqueue(n_j)$;
18: return $\langle V, E\rangle$;
```

other nodes can be released (lines 12–13 and 16–17). Inputs w and F will influence the performance of all algorithms proposed in this paper. However, to model more general applications, we do not restrict the input.

The numbers of iterations of the **for** loop in line 2, **while** loop in line 5 and **for** loop in line 9 are $O(|F|)$, $O(|N|)$, and $O(|N|)$, respectively. Thus, the time complexity of Algorithm 1 is $O(|F||N|^2)$.

C. Problem Formulation

The network manager assigns the r_ith channel offset and the s_ith time slot to vertex v_i. Note that r_i denotes the r_ith channel offset in the local available channel offsets M, i.e., $r_i \leq m$, where $m = |M|$. If the vertex is assigned time slot s_i, it will be scheduled in time slots $s_i + j \cdot p'_i$ $(j \geq 0)$ because the flow is periodic. Therefore, the scheduling problem of a single WirelessHART network is described as follows. Given flow set F and releasing sequence graph G, our objective is to find r_i and s_i for each vertex such that

$$\min m.$$

The minimizing problem should respect the following constraints.

1) *Channel offset constraint:* $\forall v_i \in V, 0 \leq r_i < m$. For each vertex, its channel offset r_i cannot exceed the number of used channels in the local network.
2) *Real-time constraint:* $\forall v_i \in V,\ 0 \leq s_i < p'_i$. To satisfy the real-time performance constraints, no transmission can miss the deadline.
3) *Sequence constraint:* $\forall e_{ij} \in E,\ s_i < s_j$. As shown in the releasing sequence graph, if there exists element e_{ij}, the scheduling of vertex v_j is after vertex v_i.

4) *Interference constraint:* There are two types of interference, *node interference* (two vertices use the same node) and *scheduling interference* (two vertices are scheduled in the same time slot and on the same channel). If two vertices use the same node, they cannot be scheduled in the same time slot [see (1)]. Otherwise, they cannot be scheduled in the same time slot and on the same channel [see (2)]. If the schedules do not respect the interference constraint, the transmissions fail. We use the following logical expressions to formulate these constraints.

$\forall v_i, v_j \in V, v_i \neq v_j$, for simplicity, we use v_a to denote the vertex that has the longer period between v_i and v_j, i.e., $a = \arg\max_{\sigma=i,j}\{p'_\sigma\}$, and v_b has the shorter period. Note that during a period of vertex v_a, vertex v_b is scheduled $\frac{p'_a}{p'_b}$ times. Therefore, $s_a = s_b + g \cdot p'_b$ $[\forall g \in [0, \frac{p'_a}{p'_b})]$ is used to denote whether interference between the two vertices exists.

If $\{sn_a, dn_a\} \cap \{sn_b, dn_b\} \neq \emptyset$, then

$$\bigwedge_{\forall g \in [0, \frac{p'_a}{p'_b})} (s_a \neq (s_b + g \cdot p'_b)) = 1 \tag{1}$$

else

$$\bigwedge_{\forall g \in [0, \frac{p'_a}{p'_b})} (s_a \neq (s_b + g \cdot p'_b)) \vee (r_a \neq r_b) = 1. \tag{2}$$

D. Our Heuristic Algorithm: Extended Rate-Monotonic (E-RM)

The WirelessHART protocol suggests using the classical rate-monotonic (RM) policy as a base scheduling strategy [1]. In RM, the vertex with the shorter period is assigned the higher priority. Note that if a vertex has node interference in this time slot, it cannot be scheduled no matter how many channels are idle. Therefore, node interference reduces the channel utilization. The gateway is a hotspot and experiences more node interference than other nodes do. The vertices using the gateway are called *gateway vertices*, and those vertices not on the gateway are *nongateway vertices*. We trace the schedule of RM, learning that at the end of each period, only gateway vertices remain. Thus, only one channel is used by gateway vertices due to node interference on the gateway; the other channels are wasted. Therefore, we propose an E-RM policy to optimize the schedule of gateway vertices. In E-RM, gateway vertices are scheduled as preferentially as possible. Thus, they and other nongateway vertices can be scheduled in the same time slot on different channels. This scheme improves the time efficiency and channel utilization.

We use Algorithm 2 to check the number of channels on which the flow set is schedulable. If for the given number of channels m, the schedules of all flows in the hyperperiod H can be generated, the flow set is schedulable (SCH); otherwise, it is unschedulable (UNSCH). The minimum number of required channels is obtained by $\min\{m | \forall m \in [1, 15], E-RM(m) \text{returns SCH}\}$.

In Algorithm 2, a vertex has two priorities: the prime priority $\rho 1_i$ and the subpriority $\rho 2_i$. The smaller value indicates the higher priority. The subpriority is the traditional RM priority

Algorithm 2: *E-RM(m).*

```
Require: $G$, $F$, $m$
Ensure: $\{\langle r_1, s_1\rangle, \langle r_2, s_2\rangle, ...\}$, SCH or UNSCH
1:  for each $v_i \in \mathbb{V}$ do
2:      $\rho 1_i \leftarrow (sn_i$ or $dn_i$ is the gateway)?1 : 2;
3:      $\rho 2_i \leftarrow p'_i$;   // the priority in the RM policy
4:  for each $t \in [0, H)$ do
5:      update the released set $\mathbb{V}$;
6:      for each $v_i \in \mathbb{V}$ do
7:          if $p'_i - t =$ the number of remaining hops then
8:              $\rho 1_i \leftarrow 0$;   //the highest priority
9:      $ch \leftarrow 0$;
10:     while $\mathbb{V} \neq \emptyset$ do
11:         find vertex $v_i$ such that
            $\bigwedge_{\forall v_j \in \mathbb{V} \setminus \{v_i\}} ((\rho 1_i < \rho 1_j) \vee ((\rho 1_i = \rho 1_j) \wedge (\rho 2_i < \rho 2_j)))$
12:         if $t > p'_i$ then return UNSCH;
13:         if $\{sn_i, dn_i\} \cap \{sn_j, dn_j | \forall v_j \in \Lambda_{t+g \cdot p'_i}$,
            $g \in [0, \frac{H}{p'_i})\} \neq \emptyset$ then
14:             delete $v_i$ from $\mathbb{V}$;
15:         else if $ch < m$ then
16:             $r_i \leftarrow ch$; $s_i \leftarrow t$; $ch++$;
17:             move $v_i$ from $\mathbb{V}$ to $\Lambda_{t+g \cdot p'_i}$, where $g \in [0, \frac{H}{p'_i})$;
18: return SCH, $\{\langle r_1, s_1\rangle, \langle r_2, s_2\rangle, ...\}$;
```

(line 3). The prime priority of gateway vertices is greater than that of nongateway vertices (line 2). In addition, when the number of remaining time slots before the deadline is equal to the number of remaining hops, the vertex must be scheduled immediately. Otherwise, the deadline will be missed. Therefore, its prime priority is set to the highest value (lines 6–8). $\mathbb{V}$ is the set of vertices that have been released but not scheduled. The vertices in set $\mathbb{V}$ are scheduled in order from the highest priority to the lowest priority (line 11). The vertex with the highest prime priority is chosen first. If more than one vertex has the same prime priority, the vertex with the higher subpriority is chosen. For each chosen vertex, if the current time slot exceeds its deadline, the flow set is unschedulable (line 12). The set Λ_t contains the vertices that have been scheduled in time slot t. If the chosen vertex interferes with the scheduled vertices, it cannot be scheduled in this time slot (lines 13–14), which corresponds to constraint (4.a). Otherwise, it is scheduled when it satisfies constraint (4.b) (lines 15–17). Note that a vertex is scheduled periodically. When it is scheduled in time slot t, it must be scheduled in time slots $t + g \cdot p'_i$ $[\forall g \in [0, \frac{H}{p'_i})]$ and each $\Lambda_{t+g \cdot p'_i}$ $[\forall g \in [0, \frac{H}{p'_i})]$ must be updated.

In Algorithm 2, the number of iterations of the **for** loop in line 4 is $O(H)$. The number of iterations of the **for** loop in line 6 and **while** loop in line 10 are both at most $O(|V|)$. The time complexity of determining the highest priority vertex is $O(|V|^2)$ (line 11). The complexity of the verification in line 13 is $O(\frac{H}{p'_i})$, and the complexity of line 17 is $O(\frac{H}{p'_i})$. Therefore, the complexity of Algorithm 2 is $O(H|V|(|V|^2 + (\frac{H}{p_{min}})^2))$, where $\frac{H}{p_{min}}$ is the quotient of the maximum period and the minimum period. In a real WirelessHART network, the periods of all flows are usually similar, i.e., $\frac{H}{p_{min}}$ is usually small.

The only difference between E-RM and RM is how to identify the highest priority vertex (line 11). The complexity of that part in RM is $O(|V|)$. Therefore, the time complexity of RM is $O(H|V|(|V| + (\frac{H}{p_{min}})^2))$. Our algorithm only introduces $O(|V|)$ cost. In Section V, we will present a comparison of their running time.

E. Schedule Performance Analysis

In this section, we analyze the schedule performance of E-RM and RM. Because flows have different periods, each unit period contains different vertices. Therefore, the packets generated by a flow in different periods will meet different higher priority vertices and suffer different interference. We distinguish vertices in different periods as follows. The symbol $v_{k,g}$ $[g \in [0, \frac{H}{p'_k})]$ denotes vertex v_k in the gth period. Then, we can determine that when a packet is released at time $p_i \cdot g$, vertex $v_{k,g}$ belonging to the packet will be scheduled in time interval $\gamma_{k,g} = [p_i \cdot g, p_i \cdot (g+1) - 1]$. We call the time interval the lifetime of vertex $v_{k,g}$.

In RM, the node interference suffered by vertex $v_{k,g}$, which belongs to flow f_i, is denoted as $\alpha_{k,g} = \{v_{a,\vartheta} | \rho_a < \rho_k,\ \{sn_a, dn_a\} \cap \{sn_k, dn_k\} \neq \emptyset,\ \gamma_{k,g} \cap \gamma_{a,\vartheta} \neq \emptyset \quad \forall \vartheta \in [0, \frac{H}{p'_a})\}$. Then, the node interference that flow f_i suffers is defined as $A_{i,g} = \bigcup_{\forall v_k \in \phi_i} \alpha_{k,g} - \phi_i$. We assume that vertex $v_{j,\vartheta}$ has higher RM priority than does vertex $v_{k,g}$, and that their lifetimes overlap. If vertex $v_{j,\vartheta}$ is not in set $A_{i,g}$, it is in the scheduling interference set $\beta_{k,g}$. The scheduling interference is denoted as $\beta_{k,g} = \{v_{b,\vartheta} | \rho_b < \rho_k,\ \gamma_{k,g} \cap \gamma_{b,\vartheta} \neq \emptyset \quad \forall \vartheta \in [0, \frac{H}{p'_b}), v_{b,\vartheta} \notin A_{i,g}\}$ and $B_{i,g} = \bigcup_{\forall v_k \in \phi_i} \beta_{k,g} - \phi_i$. In E-RM, some vertices' priorities are changed. Then, the interferences suffered by vertex v_k in the gth period are $\alpha^E_{k,g} = \{v_{a,\vartheta} | ((\rho 1_a < \rho 1_k) \vee (\rho 1_a = \rho 1_k \wedge \rho 2_a < \rho 2_k)),\ \{sn_a,\ dn_a\} \cap \{sn_k,\ dn_k\} \neq \emptyset, \gamma_{k,g} \cap \gamma_{a,\vartheta} \neq \emptyset, \forall \vartheta \in [0, \frac{H}{p'_a})\}$ and $\beta^E_{k,g} = \{v_{b,\vartheta} | (\rho 1_b < \rho 1_k) \vee (\rho 1_b = \rho 1_k \wedge \rho 2_b < \rho 2_k), \gamma_{k,g} \cap \gamma_{b,\vartheta} \neq \emptyset, \forall \vartheta \in [0, \frac{H}{p'_b}), v_{b,\vartheta} \notin A^E_{i,g}\}$. Then $A^E_{i,g} = \bigcup_{\forall v_k \in \phi_i} \alpha^E_{k,g} - \phi_i$, $B^E_{i,g} = \bigcup_{\forall v_k \in \phi_i} \beta^E_{k,g} - \phi_i$.

In a *1-speed network* means, all nodes of the network run at the normal speed. In the *z-speed* network, all nodes run on speed z; the bandwidth is z times as fast, and the number of channels is the same as the original network. Therefore, if the period (and deadline) of a flow is p in a 1-speed network, then its period is $z \cdot p$ in the z-speed network. We use *speedup factor* z to evaluate the schedule performance of a scheduling algorithm. To derive the speedup factors of RM and E-RM, we assume there exists an optimal scheduling algorithm that can find the feasible schedule without any node interference, i.e., the optimal algorithm can find a feasible schedule, if

$$\forall f_i \in F, \sum_{\forall f_b \in hp(f_i)} \left(|V_b| \cdot \frac{p_i}{p_b}\right) + |V_i| \leq m \cdot p_i \tag{3}$$

where V_i is the vertex set of flow f_i and $hp(f_i)$ denotes the set of flows whose priorities are greater than f_i. In other words, for flow f_i, if the resources required by it and the flows in the set $hp(f_i)$ are supplied on m channels before its deadline, then the assumed optimal algorithm can find a feasible schedule. There are no other algorithms better than the assumed optimal algorithm because if (3) does not hold, at least two transmissions are scheduled in the same time slot and on the same channel. This situation represents scheduling interference and is not supported by our system. Therefore, (3) is the utilization upper bound, and our speedup factors analysis is safe. Based on (3), we derive the speedup factor of RM (see Theorem 1).

Lemma 1: If $\forall f_i \in F$, $\forall g \in [0, \frac{H}{p_i})$, $\left\lfloor \frac{|B_{i,g}|}{m} \right\rfloor + |A_{i,g}| + |\phi_i| \leq p_i$, then the flow set can be real-time scheduled by RM.

Proof: Packet P is generated by flow f_i in the gth period. P can be delayed by sets $A_{i,g}$ and $B_{i,g}$. In the worst case scenario, the packet suffers interference from all vertices in set $A_{i,g}$, and this interference is in $|A_{i,g}|$ time slots. Therefore, the delay introduced by the node interference is at most $|A_{i,g}|$. For set $B_{i,g}$, if the vertices in $B_{i,g}$ do not occupy all m channels, then an idle channel can be used to schedule packet P. In this case, the vertices do not delay the packet P. Therefore, in the worst case scenario, the vertices in $B_{i,g}$ occupy all m channels in one time slot. The delay introduced by set $B_{i,g}$ is $\left\lfloor \frac{|B_{i,g}|}{m} \right\rfloor$. In addition, the packet passes path ϕ_i using at most $|\phi_i|$ time slots. Therefore, the worst case delay is $\left\lfloor \frac{|B_{i,g}|}{m} \right\rfloor + |A_{i,g}| + |\phi_i|$. When the delay is not greater than the deadline (period), the packet is schedulable. Therefore, Lemma 1 holds. ■

Lemma 2: If $\forall f_i \in F, \forall g \in [0, \frac{H}{p_i}]$ and $\left\lfloor \frac{|B^E_{i,g}|}{m} \right\rfloor + |A^E_{i,g}| + |\phi_i| \leq p_i$, then the flow set can be real-time scheduled by E-RM.

The proof of Lemma 2 is the same as that of Lemma 1.

Theorem 1: Any flow set that can be real-time scheduled on the 1-speed network is real-time scheduled by RM on the z-speed network with $z \geq m$.

Proof: From Lemma 1, we know that if a flow set can be scheduled on the z-speed network, then $\forall f_i \in F, \forall g \in [0, \frac{H}{p_i})$

$$\left\lfloor \frac{|B_{i,g}|}{m} \right\rfloor + |A_{i,g}| + |\phi_i| \leq z \cdot p_i. \tag{4}$$

In RM, the vertices in $A_{i,g}$ and $B_{i,g}$ have higher priorities than do those in ϕ_i. Therefore, they belong to flows $f_b \in hp(f_i)$. Thus, from (3), $|B_{i,g}| + |A_{i,g}| + |\phi_i| \leq \sum_{\forall f_b \in hp(f_i)} (|V_b| \cdot \frac{p_i}{p_b}) + |V_i| \leq m \cdot p_i$. Then,

$$|A_{i,g}| + |\phi_i| \leq m \cdot p_i - |B_{i,g}|. \tag{5}$$

We can obtain that $\frac{\left\lfloor \frac{|B_{i,g}|}{m} \right\rfloor + |A_{i,g}| + |\phi_i|}{p_i} \leq \frac{\left\lfloor \frac{|B_{i,g}|}{m} \right\rfloor + m \cdot p_i - |B_{i,g}|}{p_i} \leq m + |B_{i,g}| \cdot (\frac{1}{m \cdot p_i} - \frac{1}{p_i})$. Thus, when

$$z \geq m + |B_{i,g}| \cdot \left(\frac{1}{m \cdot p_i} - \frac{1}{p_i}\right) \tag{6}$$

(4) holds. Because $|B_{i,g}|$ is always greater than or equal to zero, in the worst case $|B_{i,g}| = 0$ and thus $z \geq m$. ■

The proposed E-RM adds two strategies to the traditional RM policy: (a) the gateway vertex has higher priority and (b) the vertex whose number of remaining hops is equal to the remaining time slots has the highest priority. Then, we prove that strategy (b) does not degrade the schedule performance and illustrates the speedup factor of E-RM.

Lemma 3: If flow f_i is real-time scheduled by algorithm E-RM without strategy (b), it can be real-time scheduled by E-RM.

Proof: We assume that vertex v_k can trigger strategy (b). Before vertex v_k is released, the schedules of algorithm E-RM with or without strategy (b) are the same. After vertex v_k is released, if algorithm E-RM without strategy (b) can real-time schedule the flow, the vertices between vertex v_k and the destination of the flow are given the highest priority and are scheduled in every time slot. Otherwise, the flow will miss its deadline. The situation that these vertices are scheduled in every time slot is the same as that in strategy (b). Therefore, the flow can be scheduled by algorithm E-RM with strategy (b). Lemma 3 holds. ■

Theorem 2: Any flow set that can be real-time scheduled on the 1-speed network can be real-time scheduled by E-RM on the z-speed network with $z \geq \frac{m \cdot H}{p_{min}}$.

Proof: According to Lemma 3, the speedup factor z of E-RM without strategy (b) is not less than that of E-RM with strategy (b). Then we derive the speedup factor of E-RM without strategy (b). From Lemma 2, we have

$$\left\lfloor \frac{|B^E_{i,g}|}{m} \right\rfloor + |A^E_{i,g}| + |\phi_i| \leq z \cdot p_i. \tag{7}$$

Similarly, in the worst case $|B^E_{i,g}| = 0$ and the upper bound of $|A^E_{i,g}| + |\phi_i|$ is $m \cdot H$. Therefore, $z \geq \frac{m \cdot H}{p_i}$. Thus, for all packets, we can determine that $z \geq \frac{m \cdot H}{p_{min}}$, which is also the upper bound of the speedup factor for algorithm E-RM with strategy (b). ■

F. Our Heuristic Algorithm: Z-RM

From Theorems 1 and 2, we obtain that compared with RM, E-RM can relieve the congestion of the gateway; however, in the worst case, it might have low schedule performance. Therefore, we propose a tradeoff algorithm between these two algorithms, Z-RM.

Compared with RM, E-RM changes the priorities of gateway vertices. These vertices have node interference with each other on the gateway. Therefore, they are not in set $B^E_{i,g}$, i.e., $B^E_{i,g} = B_{i,g}$. However, $A^E_{i,g} \neq A_{i,g}$ because in E-RM, if v_k is a gateway vertex, it suffers less node interference than exists in RM $\alpha^E_{k,g} = \alpha_{k,g} - \{v_{b,\vartheta} | \rho 2_b < \rho 2_k, \gamma_{b,\vartheta} \cap \gamma_{k,g} \neq \emptyset, \vartheta \in [0, \frac{H}{p'_b}), sn_b$ and dn_b are not the gateway $\}$; otherwise, it will suffer more node interference $\alpha^E_{k,g} = \alpha_{k,g} + \{v_{b,\vartheta} | \rho 2_b > \rho 2_k, \gamma_{b,\vartheta} \cap \gamma_{k,g} \neq \emptyset, \vartheta \in [0, \frac{H}{p'_b}), sn_b$ or dn_b is the gateway$\}$. We set $|A^E_{i,g}| = |A_{i,g}| + a$, where a is an integer that represents the difference value between them. Then, from (7), we determine that

$$z \geq \frac{\left\lfloor \frac{|B_{i,g}|}{m} \right\rfloor + |A_{i,g}| + |\phi_i| + a}{p_i}. \tag{8}$$

Algorithm 3: Priority assignment in algorithm *Z-RM*.

```
Require: the given threshold $\bar{z}$
Ensure: $\forall \rho 1_i, \forall \rho 2_i, \forall \rho 3_i$
 1: sort flows according to the decreasing order of their
    priorities, where $f_1$ has the highest priority;
 2: for $i = 1$ to $|F|$ do
 3:     $\tilde{\rho}_i \leftarrow \min\{k | \bar{z} < \frac{|A^k_{i,g}| - |A_{i,g}|}{p_i}, k \in [i, |F|],$
        $g \in [0, \frac{H}{p_i})\}$;
 4: for $i = |F| - 1$ to 1 do
 5:     if $\tilde{\rho}_{i+1} < \tilde{\rho}_i$ then
 6:         $\tilde{\rho}_i \leftarrow \tilde{\rho}_{i+1}$;
 7: for each $v_b$ do
 8:     $\rho 2_b \leftarrow$ the ID of the flow that $v_b$ belongs to; //
        the priority in the RM policy
 9:     if $sn_b$ and $dn_b$ are not the gateway then
10:         $\rho 1_b \leftarrow 2$;
11:         $\rho 3_b \leftarrow \tilde{\rho}_i$, where vertex $v_b$ belongs to flow $f_i$;
12:     else
13: $\rho 1_b \leftarrow 1$;
```

According to (5), the right side of (8) is $\frac{\left\lfloor \frac{|B_{i,g}|}{m} \right\rfloor + |A_{i,g}| + |\phi_i| + a}{p_i} \leq \frac{\left\lfloor \frac{|B_{i,g}|}{m} \right\rfloor + m \cdot p_i - |B_{i,g}| + a}{p_i}$. When $z \geq \frac{\left\lfloor \frac{|B_{i,g}|}{m} \right\rfloor + m \cdot p_i - |B_{i,g}| + a}{p_i}$, (7) holds. Similarly, $|B_{i,g}| = 0$. Then,

$$z \geq m + \frac{a}{p_i}. \tag{9}$$

Regarding (6) and (9), we find that the difference between the two algorithms is the expression $\frac{a}{p_i}$, which can be positive or negative. In Z-RM, to restrict the schedule performance loss, we set the parameter $\bar{z} = \frac{a}{p_i}$ as a given threshold, and then assign priorities for vertices according to this threshold. When the parameter $\bar{z}$ is set to zero, Z-RM is the same as RM. We will demonstrate the effect of parameter $\bar{z}$ on the schedule performance of Z-RM in Section V.

In Z-RM, the priority assignment method is presented in Algorithm 3. First, all flows are sorted in decreasing order of their RM priorities (line 1), i.e., flow f_i has higher priority than flow f_{i+1}. Similarly, the gateway vertex has prime priority and subpriority (RM priority) (lines 8 and 13). Nongateway vertices have a descent priority $\rho 3$ in addition to prime priority and subpriority (lines 8–11). The descent priority is assigned according to the given threshold $\bar{z}$ and denotes which gateway vertices are allowed to be scheduled before nongateway vertices with higher subpriorities. For example, gateway vertex v_b is allowed to be scheduled before nongateway vertex v_c if $\rho 3_c > \rho 2_b$.

Because threshold $\bar{z}$ is for an entire routing graph, we set that all nongateway vertices of a flow have the same descent priority. In every period of flow f_i, the minimum value of parameter k that does not guarantee $\bar{z} \geq \frac{a}{p_i}$ is its descent priority (line 3). Set $A^k_{i,g}$ denotes the node interference suffered by flow f_i in the gth period when its descent priority is equal to k. The definition of set $A^k_{i,g}$ is similar to that set $A^E_{i,g}$. The difference is that the gateway vertex whose flow ID is greater than k is not contained in set $A^k_{i,g}$. However, using this method to assign priorities, the following problem can occur. Given two nongateway vertices v_b and v_c and $\rho3_b > \rho3_c$, $\rho2_b < \rho2_c$, there exists a gateway vertex v_e such that $\rho3_b > \rho2_e > \rho3_c$. When we compare the priorities of v_b and v_c, vertex v_b should be scheduled first. When the comparison is between v_c and v_e, vertex v_c should be scheduled first. When the comparison is between v_b and v_e, vertex v_e has higher priority than vertex v_b; their priorities are deadlocked with each other. Therefore, the descent priorities of flows are in nondecreasing order (lines 4–6). The time complexity of line 3 is $O(|F|(\frac{H}{p_{min}})|V|)$. The number of iterations of the **for** loop in line 2 is $O(|F|)$. Therefore, the time complexity of Algorithm 3 is $O(|F|^2(\frac{H}{p_{min}})|V|)$.

Based on these priority assignments, the method in Z-RM that selects the highest priority vertex in set $\mathbb{V}$ is as follows. If a vertex has the highest prime priority, it is scheduled first. Otherwise, selected vertex v_b guarantees the following terms $\forall v_c \in \mathbb{V}$ and $v_c \neq v_b$:

$$\begin{cases} \rho2_b < \rho2_c, & \text{if } \rho1_b == \rho1_c \\ \rho2_b < \rho3_c, & \text{if } \rho1_b \neq \rho1_c,\ v_b \text{ is a gateway vertex} \\ \rho3_b < \rho2_c, & \text{if } \rho1_b \neq \rho1_c,\ v_b \text{ is a non-gateway vertex.} \end{cases}$$

Except for the priority assignment and the vertex selection, the other pseudocode segments are the same as those in E-RM, and their time complexities are the same.

IV. Channel Management Scheme

The proposed E-RM and Z-RM schedule data flow in the lower level and obtain the number of required channels. In this section, we focus on the upper level and propose a channel management scheme to isolate multiple networks according to the results of E-RM and Z-RM.

Recall that we use superscripts to denote to which network the symbol belongs, e.g., the symbol m^i denotes the number of actual channels used by network w^i, and if networks w^i and w^j overlap, then $c^{ij} = 1$. Our proposed channel management scheme contains the initial assignment and the dynamic adjustment. They are as follows.

A. Initial Assignment

There are only 15 channels. Owing to external interference, some channels cannot be accessed and some channel will be used to adjust the initial assignment. Hence, the channel resource is scarce. To isolate networks and reserve channels for adjusting assignments, our objective for the initial assignment algorithm is to minimize the number of channels required to isolate networks. This problem is similar to the *minimum vertex coloring* problem [20], which uses the minimum number of colors to solve the graph-coloring problem. The graph construction of our system is as follows. Each network corresponds to m^i vertices, and any two of these vertices directly connect with each other. If $c^{ij} = 1$, then the vertices of the network w^i directly connect to all vertices of network w^j. The colored graph of the example in Fig. 3 is shown in Fig. 6. Then, we use the classical DSATUR [20] to solve this problem. The classical algorithm DSATUR assigns the least colors to the vertices in order from maximal to minimal connection degree. In our system, except for the channels that are on blacklists, other channels are usable colors. Thus, for each vertex in order from maximal to minimal connection degree, the possible least channel is assigned to it. Fig. 6 also shows the resulting assignments of channels $\{1,2,3\}$ to w^1; $\{4,5,6\}$ are assigned to w^2 and w^3. The interference relationship, which is $c^{12} = c^{23} = 1$ in Fig. 6, will influence the effect of the channel assignment. Previously described topology control techniques [21], [22] can be used to reduce interference among coexisting networks. However, these topics are not the focus of this paper; we will study them in future work.

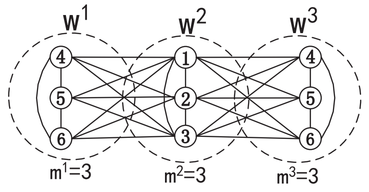

Fig. 6. Colored graph of the example in Fig. 3.

Algorithm 4: Dynamic Adjustment Algorithm.

```
Require: $M'^g$, $\mathbb{M}^g$ and $\forall \mathbb{M}^i$
Ensure: $M^g$
1: $\mathcal{M} \leftarrow \{j | j \in [11,25]$ and $j \notin (\mathbb{M}^g \cup \bar{M}^g \cup \bigcup_{\forall a, c^{ga}=1} \mathbb{M}^a)\}$;
2: for each $i \in M'^g$ do
3:     $\forall j \in \mathcal{M}$, if $\bigwedge_{\forall a \in M'^g} (|j-a| > 2)$ then $\xi \leftarrow j$; break;
4:     $\xi \leftarrow$ a random channel offset in $\mathcal{M}$;
5:     $M^g \leftarrow M^g - \{i\} + \{\xi\}$; $\mathcal{M} \leftarrow \mathcal{M} - \{\xi\}$;
6: return $M^g$;
```

B. Dynamic Adjustment

The second subproblem is how to adjust assignments dynamically to alleviate interference from external peripherals. A previously published work [18] indicates that a short time window is sufficient for estimating channel quality and infrequent channel adjusting can improve reliability. Therefore, we adjust channel assignments according to the transmission loss rate in a time window. In each time window, every node records the number of transmissions that are not received on the assigned channel, and before the end of the time window, the last three packets generated by the node piggyback records to the network manager. Using three packets can improve reliability. The network manager calculates transmission loss rates and sends them to the coordinator. Then in the coordinator, for each channel, if its transmission loss rate is greater than the given threshold, the quality of the channel is set to poor and the adjustment algorithm (see Algorithm 4) is invoked to reassign a new channel. At the beginning of the next time window, the information about the reassigned channel is sent to all nodes of the network. The new channel is then used.

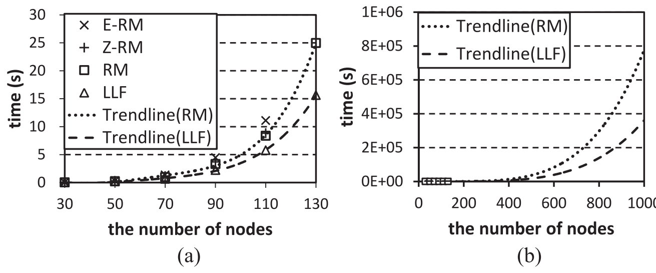

Fig. 7. Run times. (a) 30 $\leq n \leq 130$. (b) $n \leq 1000$.

In Algorithm 4, set M'^g contains the channels that need to be reassigned. The channel IDs used in the WirelessHART network are from 11 to 25, and $\mathbb{M}^g$ is the set of channels that are used by network w^g in the last two time windows. If a channel is used by network w^a that overlaps with network w^g, it cannot be reassigned to network w^g. The selectable channels are in set $\mathcal{M}$ (line 1). A previous study [18] indicates that the reliability of adjacent channels is strongly correlated and that the new channel is at least three offsets away from the original channel. Therefore, if there exists a channel that is at least three offsets away from all channels in M'^g, the channel is selected as a new one (line 3). Otherwise, a random channel in set $\mathcal{M}$ is the new one (line 4). Then, the original channel is changed and the selectable channel set $\mathcal{M}$ is updated (line 5). Its time complexity is $O(|M'^g|^2|\mathcal{M}|)$, where $|M'^g|$ and $|\mathcal{M}|$ are not greater than 15. Therefore, the complexity is $O(1)$.

V. Evaluation

In this section, our simulations are presented to demonstrate the effectiveness of our methods. The evaluation contains three parts. They focus on different aspects and compare different performance metrics in parallel. First, a run time comparison is presented to explain why we use the two-level framework rather than centralized management. Second, we demonstrate that our scheduling algorithms use less channel resources than do existing ones. Finally, based on trace-driven simulations, we present that our channel management algorithm can reduce the packet loss rate and improve network reliability. We compare our algorithms with the classical real-time scheduling policies RM and least laxity first (LLF), which first schedules the transmission whose packet has the least laxity time). For our system model, the classical EDF policy is the same as RM. Therefore, we do not consider it. To avoid a biased comparison and perform a thorough evaluation, we impose no restriction on how to choose test cases. In our simulations, thousands of test cases were generated randomly. The network deployment followed (1) that appeared in a previous study [23], and several random shortest paths combine in a routing graph. This approach is more general and can yield unbiased results.

A. Our Framework Versus Centralized Managements

The classical policies LLF and RM are considered the centralized methods. Fig. 7 shows a run time comparison among centralized methods and our two-level framework. Each point in Fig. 7(a) denotes the average value of 100 test cases. When the number of nodes in a network is 110, the running time of these methods is 11.2, 8.8, 8.4, and 6.1 s. In reality, this time scale is acceptable because in industrial wireless sensor networks, initialization, including, for example, node joining, topology generation, and path generation, will last for a few minutes. The extra several seconds introduced by our algorithms can be neglected. For a large-scale system, our methods focus on the single WirelessHART network and need only schedule at most 100 nodes. However, the centralized methods must schedule the entire network. Because the running time is too long for the large-scale system, we use trend lines to help predict it. The polynomial trend lines are drawn through the data points in Fig. 7(a). Then, the same trend lines are extended to 1000 nodes as shown in Fig. 7(b). From the figure, we find that although the centralized methods are the simple LLF and RM, the run time remains unacceptable. When the number of nodes is 1000, the run times of these two methods are approximately 97 and 211 h.

B. Scheduling Algorithm

We initially evaluate our scheduling algorithms in a single network, and then in coexisting networks. The period parameter is randomly selected in the set $\{2^i|\forall i \in [p^l, p^u]\}$. The CPLEX solver is exploited to obtain the optimal solution from the formulation (see Section III-C). To allow the problem to be solved by using CPLEX, we set the number of nodes $n = 6$ and $p^l = p^u = 1$. One hundred test cases that can be solved by CPLEX in an acceptable time are used to evaluate the effectiveness of other algorithms. In these test cases, the algorithms E-RM, Z-RM, RM, and LLF have the same solutions. The solutions of 87 test cases, which are solved by these algorithms, are the same as CPLEX. In these simple networks, almost all of the number of routing hops are less than 3. Over 70% of transmissions pass through the gateway. In this case, even without our proposed prime priority, the gateway is almost fully used in every time slot. Therefore, RM and LLF are the same as ours.

Because CPLEX cannot determine a feasible solution in an acceptable time, the following simulations do not employ it. We compare these algorithms under different parameter configurations, and the number of nodes ranges from 20 to 90. The comparisons are shown in Fig. 8. For each figure, 5000 test cases were randomly generated. We use extensive simulations to validate the universality of our algorithms. A column in the figure represents the number of test cases that can be scheduled when the minimum number of used channels is equal to the abscissa value. For example, in Fig. 8(a), the value of the first column is 4715, thus indicating that 4715 test cases can be scheduled using at least two channels. From the figures, we can see that when the number of channels is small, the columns of our algorithms E-RM and Z-RM are greater than are those of the classical algorithms. Thus, then with the increase in the number of channels, the columns of the classical algorithms are greater than ours are. Our algorithms, therefore, make flows schedulable on fewer channels. Among these algorithms, E-RM is more effective than the others, and LLF has the worst results. From Algorithm 3, we know that when $\bar{z} = 0$, Z-RM is the

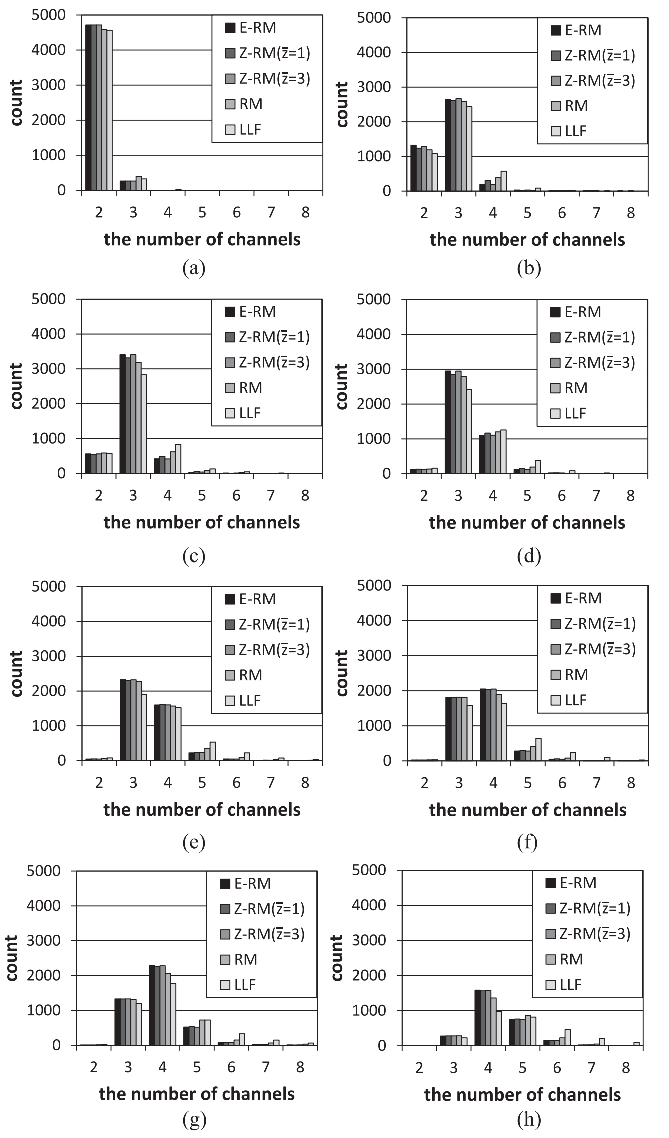

Fig. 8. Minimum number of required channels in a single network. (a) $n = 20$, $p^l = 2$, and $p^u = 3$. (b) $n = 30$, $p^l = 2$, and $p^u = 4$. (c) $n = 40$, $p^l = 3$, and $p^u = 4$. (d) $n = 50$, $p^l = 3$, and $p^u = 5$. (e) $n = 60$, $p^l = 3$, and $p^u = 6$. (f) $n = 70$, $p^l = 3$, and $p^u = 7$. (g) $n = 80$, $p^l = 3$, and $p^u = 8$. (h) $n = 90$, $p^l = 3$, and $p^u = 8$.

same as RM. Therefore, the simulation adopts $\bar{z} = 1$ and $\bar{z} = 3$ to evaluate Z-RM. From extensive simulations, we know that when $\bar{z} = 3$, the results of Z-RM are similar to E-RM. Then, we compare these algorithms on schedulability. The performance metric is a schedulable ratio, which is defined as the percentage of test cases for which an algorithm is able to find a feasible schedule. Fig. 9 shows the normalized schedulable ratios of the simulations in Fig. 8 with RM used as the baseline. Each set of columns corresponds to a figure of Fig. 8. We find that our algorithms have similar schedule performance to RM (97% on average).

The objective of our scheduling algorithms is to reserve more channel resources for dynamic channel adjustment. Fig. 10 shows the comparison of remaining channel resources under multiple coexisting networks. The number of nodes in a

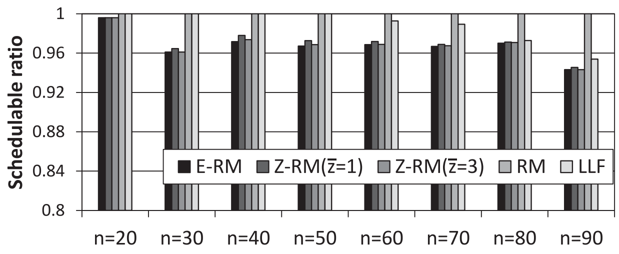

Fig. 9. Schedulable ratio.

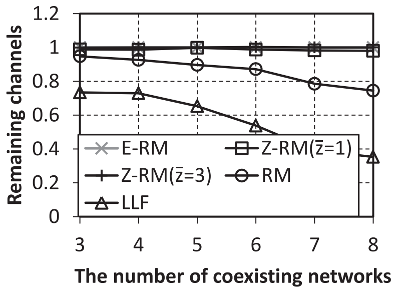

Fig. 10. Remaining channels.

single network is randomly selected in $\{30, 50, 70, 90\}$. For each configuration, we randomly generate 100 test cases, and then calculate the sum of the remaining channel resources. Fig. 10 shows the normalized number with E-RM as the baseline. From the figure, we find that as the number of coexisting networks increases, the classical policies waste more channels. With the increase in the number of coexisting networks, the difference is increasingly distinct. Therefore, when eight networks coexist, compared with the classical policies RM and LLF, our algorithms conserve approximately 23% and 63% of the channel resources, respectively.

C. Channel Management

We consider the following comparison methods.

1) Original: There is no channel management, and each WirelessHART is scheduled based on RM.

2) Our+noBL: Our channel management scheme without blacklists.

3) Our+BL: Our scheme with blacklists.

The performance metric we used is the packet loss rate. Packet loss is severely affected by the external interference, because the external interference is uncontrolled and dynamic. Even when we evaluate the three methods on the same area but not for the same duration, the external interference suffered by the methods is different. Therefore, to guarantee fairness for all methods, we conduct trace-driven simulations. We trace the states of 15 channels from 8 A.M. to 12 A.M. over two days, and simulate these methods based on the same trace file. Our wireless nodes that are used to collect channel states are implemented on an MSP430 and a CC2420. The transmission power is 0 dBm. Due to limited space, we omit the detailed introduction of the collecting system. The entire network in Fig. 3 is simulated based on trace files. Each network contains 50 nodes and other

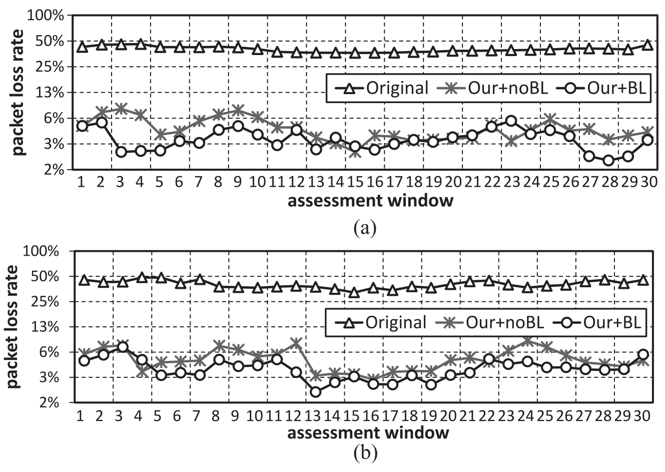

Fig. 11. Comparison of packet loss rates. (a) Day 1. (a) Day 2.

parameters are the same as those in Fig. 8. Each assessment window is 8 min. Fig. 11 shows the packet loss rates in each assessment window. In Fig. 11(a), the average packet-loss rates of Original, Our+noBL, and Our+BL are 40%, 4.5%, and 3.6%, respectively. In Fig. 11(b), they are 40%, 5.3%, and 4.0%. We also conduct more trace-driven simulations. The results are similar to the two simulations. Therefore, our channel management can reduce packet loss by 36%.

VI. Related Work

The coexistence problem of multiple networks has been widely studied in previous works. Some methods to address the interference introduced by other different networks have been proposed [24], [25]. For homogeneous coexistence, the distributed and collaborative schemes for wireless body area networks have been deeply investigated in two studies [26], [27]. In another study, low-duty signaling design was used to enhance homogeneous coexistence among multiple networks [28]. In a different study, the authors adopted a contention-based access mechanism, which is not only beneficial to homogeneous coexistence but can also avoid interference from heterogeneous networks [29]. However, these previous works are based on distributed scheme, whereas our system supports coordination that is not considered in previous works. Although the architecture in a similar study is the same as ours, this paper did not design algorithms for channel assignment [30].

Some previous works propose time- and channel-optimal scheduling methods for networks with linear topology or tree topology [31]–[34], whereas we focused on a network with mesh topology. A previous study proposed scheduling algorithms for mesh networks; however, their networks do not support graph routing, and a transmission is not periodically scheduled in different periods [8]. Therefore, existing methods cannot be used in this paper.

VII. Conclusion

To address the entire application area, it is necessary to use multiple coexisting networks. However, previous works only focus on the reliability and real-time performance of a single WirelessHART network. In this paper, we focused on the coexistent management of multiple WirelessHART networks. We proposed two algorithms to schedule flows in a single network. The two algorithms save more channel resources than do the classical policies. Our proposed scheme adopt different channels to avoid interference between multiple coexisting networks and alleviate external interference. Based on real-life state traces, the simulation results indicate that our methods can significantly improve reliability.

References

[1] "Industrial communication networks—wireless communication network and communication profiles—wirelessHART," IEC 62591, 2009.

[2] T. Sauter, "The three generations of field-level networks-evolution and compatibility issues," *IEEE Trans. Ind. Electron.*, vol. 57, no. 11, pp. 3585–3595, Nov. 2010.

[3] K. Al Agha *et al.*, "Which wireless technology for industrial wireless sensor networks the development of OCARI technology," *IEEE Trans. Ind. Electron.*, vol. 56, no. 10, pp. 4266–4278, Oct. 2009.

[4] J. Song *et al.*, "WirelessHART: Applying wireless technology in real-time industrial process control," in *Proc. Real-Time Embedded Technol. Appl. Symp.*, 2008, pp. 377–386.

[5] X. Zhu *et al.*, "Colloc: A collaborative location and tracking system on wirelessHART," *ACM Trans. Embedded Comput. Syst.*, vol. 13, 2014, Art. no. 125.

[6] Y. Liu, "Wireless sensor network applications in smart grid: Recent trends and challenges," *Int. J. Distrib. Sens. Netw.*, vol. 2012, pp. 1–8, 2012.

[7] B. Martinez, X. Vilajosana, F. Chraim, I. Vilajosana, and K. S. J. Pister ,"When scavengers meet industrial wireless," *IEEE Trans. Ind. Electron.*, vol. 62, no. 5, pp. 2994–3003, May 2015.

[8] A. Saifullah, Y. Xu, C. Lu, and Y. Chen, "Real-time scheduling for wirelessHART networks," in *Proc. Real Time Syst. Symp.*, 2010, pp. 150–159.

[9] D. Yang *et al.*, "Assignment of segmented slots enabling reliable real-time transmission in industrial wireless sensor networks," *IEEE Trans. Ind. Electron.*, vol. 62, no. 6, pp. 3966–3977, Jun. 2015.

[10] A. Saifullah, Y. Xu, C. Lu, and Y. Chen, "End-to-end communication delay analysis in industrial wireless networks," *IEEE Trans. Comput.*, vol. 64, no. 5, pp. 1361–1374, May 2015.

[11] S. Han *et al.*, "Reliable and real-time communication in industrial wireless mesh networks," in *Proc. 7th IEEE Real-Time Embedded Technol. Appl. Symp.*, 2011, pp. 3–12.

[12] "*The engineer's guide to industrial wireless measurement*," Emerson, Ferguson, MO, USA, 2014.

[13] "Fieldkey wireless adapter unlock the process and maintenance information stranded in your field instruments," ABB, Zürich, Switzerland, 2012.

[14] X. Jin *et al.*, "Collision–free multichannel superframe scheduling for ieee 802.15. 4 cluster–tree networks," *Int. J. Sensor Netw.*, vol. 15, pp. 246–258, 2014.

[15] S. Yu *et al.*, "Concurrent transmission performance modeling of wireless multimedia sensor network and its experimental evaluation," *Inf. Control*, vol. 45, pp. 328–334, 2016.

[16] K. Dang *et al.*, "A graph route-based superframe scheduling scheme in wirelessHART mesh networks for high robustness," *Wireless Pers. Commun.*, vol. 71, pp. 2431–2444, 2013.

[17] C. Wu *et al.*, "Maximizing network lifetime of wirelessHART networks under graph routing," in *Proc. 1st IEEE Int. Conf. Internet-of-Things Des. Implementation*, 2016, pp. 176–186.

[18] M. Sha, G. Hackmann, and C. Lu, "Real-world empirical studies on multi-channel reliability and spectrum usage for home-area sensor networks," *IEEE Trans. Netw. Service Manage.*, vol. 10, no. 1, pp. 56–69, Mar. 2013.

[19] A. Saifullah *et al.*, "Scheduling analysis under graph routing in wirelessHART networks," in *Proc. Real Time Syst. Symp.*, 2015, pp. 165–174.

[20] D. Brélaz, "New methods to color the vertices of a graph," *Commun. ACM*, vol. 22, pp. 251–256, 1979.

[21] M. Burkhart *et al.*, "Does topology control reduce interference?" in *Proc. 5th ACM Int. Symp. Mobile Ad Hoc Netw. Comput.*, 2004, pp. 9–19.

[22] T. M. Chiwewe and G. P. Hancke, "A distributed topology control technique for low interference and energy efficiency in wireless sensor networks," *IEEE Trans. Ind. Informat.*, vol. 8, no. 1, pp. 11–19, Feb. 2012.

[23] T. Camilo *et al.*, “Gensen: A topology generator for real wireless sensor networks deployment,” in *Proc. Softw. Technol. Embedded Ubiquitous Syst.*, 2007, pp. 436–445.
[24] J. M. Winter and C. E. Pereira, “Coexistence aware for wirelessHART networks,” in *Proc. IEEE Int. Conf. Ind. Informat.*, 2014, pp. 803–806.
[25] L. H. Correia, T.-D. Tran, V. N. S. S. Pereira, J. C. Giacomin, J. M. Sá Silva, “Dynmac: A resistant MAC protocol to coexistence in wireless sensor networks,” *Comput. Netw.*, vol. 76, pp. 1–16, 2015.
[26] M. Deylami and E. Jovanov, “A novel method for mitigating the effects of dynamic coexistence on the operation of IEEE 802.15. 4-based mobile WSNs,” *Wireless Commun. Mobile Comput.*, vol. 16, pp. 362–372, 2014.
[27] M. N. Deylami and E. Jovanov, “A distributed scheme to manage the dynamic coexistence of IEEE 802.15. 4-based health-monitoring WBANs,” *IEEE J. Biomed. Health Informat.*, vol. 18, no. 1, pp. 327–334, Jan. 2014.
[28] C.-S. Sum, M. A. Rahman, Z. Lan, F. Kojima, R. Funada, and H. Harada, “Performance analysis of low duty FSK system for smart utility network,” in *Proc. Wireless Commun. Netw. Conf.*, 2011, pp. 1568–1573.
[29] S. Nethi, J. Nieminen, and R. Jäntti, “Exploitation of multi-channel communications in industrial wireless sensor applications: Avoiding interference and enabling coexistence,” in *Proc. Wireless Commun. Netw. Conf.*, 2011, pp. 345–350.
[30] P. Ferrari, Y. Huang, and H. Sun, “Synchronized wireless sensor networks for coexistence,” in *Proc. Int. Conf. Automat. Control Artif. Intell.*, 2008, pp. 656–663.
[31] H. Zhang, P. Soldati, and M. Johansson, “Optimal link scheduling and channel assignment for convergecast in linear wirelessHART networks,” in *Proc. 7th Int. Symp. Model. Optim. Mobile, Ad Hoc, Wireless Netw.*, 2009, pp. 1–8.
[32] H. B. Zhang, F. Österlind, P. Soldati, T. Voigt, and M. Johansson, “Time-optimal convergecast with separated packet copying: Scheduling policies and performance,” *IEEE Trans. Veh. Technol.*, vol. 64, no. 2, pp. 793–803, Feb. 2015.
[33] O. Chipara, C. Lu, and G.-C. Roman, “Real-time query scheduling for wireless sensor networks,” *IEEE Trans. Comput.*, vol. 62, no. 9, pp. 1850–1865, Sep. 2013.
[34] Y. Guo, F. Kong, D. Zhu, A. S. Tosun, and Q. Deng, “Sensor placement for lifetime maximization in monitoring oil pipelines,” in *Proc. 1st ACM/IEEE Int. Conf. Cyber-Physical Syst*, 2010, pp. 61–68.

Xi Jin received the M.Sc. and Ph.D. degrees in computer science from Northeastern University, Shenyang, China, in 2008 and 2013, respectively.

From 2013 to 2015, she was an Assistant Researcher, involved in research on wireless sensor networks. She is currently an Associate Professor with 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 (S’09) received the Ph.D. degree in computer science from McGill University, Montreal, QC, Canada, in 2016.

He is currently a Postdoctoral Researcher in 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’s degree in telecommunication from TELECOM SudParis, Evry, 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. He had been 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.

Wei Liu received the M.Sc. degree in computer science from Northeastern University, Shenyang, China, in 2010. He is currently working toward the Ph.D. degree in the Chair for Design Automation of Embedded Systems, Technical University of Dortmund, Dortmund, Germany.

His current research interests include real-time embedded systems and parallel computing.

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 Shenyang Institute of Automation, Chinese Academy of Sciences, Shenyang, China, in 2005.

From 2005 to 2007, he was an Associate Professor with Shenyang Institute of Automation, Chinese Academy of Sciences, where he was involved in research on wireless sensor networks. He is currently a Professor at Shenyang Institute of Automation, Chinese Academy of Sciences. His current research interests include wireless sensor networks for industrial automation, smart grids, and demand response.