On-line Data Link Layer Scheduling in Wireless Networked Control Systems

Shengyan Hong and Xiaobo Sharon Hu
Department of Computer Science and Engineering
University of Notre Dame
Notre Dame, IN 46556
{shong3,shu}@nd.edu

Tao Gong and Song Han
Department of Computer Science and Engineering
University of Connecticut
Storrs, CT 06269
{tao.gong, song.han}@uconn.edu

***Abstract*—Wireless networked control systems (WNCSs) have received tremendous research interests recently due to their great advantages in easy deployment, enhanced mobility, and reduced maintenance cost. A key design challenge in such systems is to develop efficient data link layer scheduling algorithms to achieve a reliable end-to-end real-time communication. Previous research has a common assumption that the network communication schedule of WNCS, once constructed and distributed, stays unchanged. Using this method, however, cannot handle dynamic variations which are unavoidable in many WNCSs. In this paper, we consider the data link layer scheduling problem in WNCSs, where external disturbances occur sporadically. We employ a rhythmic task model to respond to external disturbances and introduce an effective approach to adjust the static schedule for the WNCS when the disturbances happen. The approach determines the time duration to apply a dynamic schedule and generates the schedule for that time duration to minimize the impact of network dynamics on existing network flows. The effectiveness and efficiency of the proposed algorithm are validated by extensive simulation. Results based on randomly generated task sets indicate that the proposed approach outperforms existing work both in terms of the number of feasible task sets (between 11% and 41% increase on average) and the number of feasible periodic packets (between 122% and 128% increase on average).**

I. Introduction

WNCSs have received significant attention over the past several decades [12], [16], [21], [22], [23] because of their wide applications such as telerobotics [8], aircraft control [30], civil infrastructure monitoring [17], medication service [14] and power management [11]. In a WNCS, sensors, gateways, actuators and other relay nodes are geographically distributed and connected over wireless network media. A task in a WNCS is generally required to deliver measurements from a sensor to the gateway and send control signals from the gateway to an actuator within an end-to-end deadline. The performance of WNCS should degrade gracefully in the presence of various external disturbances, such as failure in critical civil infrastructures and malicious attacks.

Though many WNCSs rely on static data link layer scheduling (typically based on Time Division Multiple Address (TDMA), *e.g.*, [13], [19], [24], [27], [29]) to achieve a deterministic real-time communication, such approaches are unsuitable for handling unexpected disturbances. In response to external events, centralized link layer scheduling approaches [6], [7], [9], [10], [25], [26] have been proposed, which can adapt to network dynamics. However, the protocols in [9], [26] cannot respond to on-line workload changes while [25] assumes that only a predetermined number of link layer schedules are stored in the system. The protocol in [10] ignores the real-time requirement of network traffic. The algorithms in [6] do not support systems containing multiple tasks ending at different actuators. In addition, [6], [7] do not discuss how the centralized scheduler knows the workload changes.

There are a few works on adapting to external events in critical control systems such as the power grid and transportation network. [20] proposed adjustment of sampling rates to improve the performance of event-triggered system. Rate-adaptive and rhythmic task models are introduced in [5] and [18], respectively, which allow tasks to change periods and relative deadlines in some control systems like automotive systems. The task models allow the system to adapt to environmental disturbances, and schedulability analyses under Rate Monotonic (RM) and Earliest Deadline First (EDF) algorithms are also provided in [18] and [5], respectively. The proposed task models, however, cannot be applied to WNCSs straightforwardly because their schedulability analyses do not consider the end-to-end packet delivery in wireless networks.

In this paper, we consider a WNCS adopting a centralized network architecture, which utilizes a static schedule when there is no physical disturbance in the WNCS to conserve system resource. Upon detecting an external disturbance, a temporary dynamic schedule is generated and employed for a relatively short time interval called dynamic scheduling interval. Due to the limited bandwidth in the system, some periodic packets may be dropped according to the dynamic schedule, which can degrade the Quality of Service (QoS). Therefore, it is critical to determine the dynamic scheduling interval and generate a dynamic schedule for this time interval to meet as many periodic packet deadlines as possible.

We introduce a data link layer scheduling problem to determine a dynamic scheduling interval and the corresponding dynamic schedule in order to minimize the number of dropped packets. In formulating the problem, we consider various practical constraints to make our work useful for real-world applications. To solve the problem, we propose an on-line approach to decide a dynamic scheduling interval and construct a dynamic schedule with bounded time and schedule update overheads. Our framework determines candidate dynamic scheduling intervals, calls an algorithm to generate a dynamic schedule for each candidate dynamic scheduling interval, and selects a best dynamic schedule in terms of minimum number of dropped packets. The effectiveness and efficiency of our approach are validated through extensive simulation results.

The remainder of this paper is organized as follows. Section II describes our system model. Section III presents a motivational example and gives the problem formulation. We present our overall framework in Section IV. Then, we describe the heuristic called by our framework in Section V. We summarize our simulation results in Section VI. Section VII concludes the paper and discusses the future work.

II. System Model

We adopt the system architecture of a typical WNCS in our study, which consists of one gateway, multiple sensors, actuators, and relay nodes. We assume that sensors and actuators also have routing capability and they are connected to the gateway wirelessly through one or multiple relay nodes. All devices in the WNCS $G = (\mathcal{V}, \mathcal{E})$ collectively form the node set $\mathcal{V} = \{V_0, V_1, V_2, \cdots, V_g\}$, where V_g represents the gateway. A direct link (V_i, V_j) in the edge set $\mathcal{E}$ represents a reliable link from node V_i to node V_j.

The system runs a set of tasks $\mathcal{T} = \{\tau_0, \tau_1, \tau_2, \cdots \tau_n, \tau_{n+1}\}$. τ_i $(0 \leq i \leq n)$ is a *unicast task* and τ_{n+1} is a *broadcast task*. Task τ_{n+1} runs on the gateway and disseminates data link layer schedule updates to all nodes in the network when necessary, by following a predetermined broadcast graph [13]. A unicast task τ_i $(0 \leq i \leq n)$ follows a designated single routing path with H_i hops. It periodically generates a packet which originates at a sensor node, passes through the gateway and then delivers a control message to an actuator.

In networked control systems, to achieve stability in the presence of unexpected physical disturbances, extra work must be done. Unexpected disturbances are detected by sensors and sent to V_g via the associated tasks. The rhythmic task model introduced by [18] has been shown to be an effective way to handle such disturbances [20]. We adopt this model in this work. Specifically, each unicast task has two states. In the *nominal state*, τ_i has a constant period P_i and a constant relative deadline D_i. When physical disturbance occurs, τ_i enters the *rhythmic state* with reduced periods and relative deadlines for prompt response. In this work, we assume that at most one unicast task can be in the rhythmic state at any time during the system operation. To simplify the notation, we refer to any task currently in the rhythmic state as *rhythmic task* and denote it as τ_0 while task τ_i $(1 \leq i \leq n)$ is a *periodic task* which is not currently in the rhythmic state. The rhythmic task has a hard deadline when it is in the rhythmic state while periodic tasks can tolerate occasional deadline misses.

When the rhythmic task τ_0 is in the nominal state, it has a constant period P_0, and a constant relative deadline D_0. When τ_0 is in the rhythmic state, we assume that it follows the system model used in [20], *i.e.*, its period and relative deadline are reduced abruptly when disturbance is detected and then return to their nominal values by following some monotonically non-decreasing functions. We use vectors $\vec{P}_0 = [P_{0,x}, x = 1, \ldots, \mathcal{R}]^T$ and $\vec{D}_0 = [D_{0,x}, x = 1, \ldots, \mathcal{R}]^T$ to represent the periods and relative deadlines of τ_0 when it is in the rhythmic state. (To be more precise, $P_{0,x}$ is the time duration between two consecutive releases but we still call it period to simplify our notation.) From the moment τ_0 enters the rhythmic state, its period and relative deadline follow the elements of $\vec{P}_0$ and $\vec{D}_0$ in sequence, respectively. When its $(R+1)$-th instance is released, τ_0 returns to the nominal state.

An instance of a task is referred to as a packet, *i.e.*, packet $\chi_{i,k}$ corresponds to the k-th instance of τ_i. According to different task types, we have *rhythmic packets*, *periodic packets*, and *broadcast packets*. A periodic or broadcast packet $\chi_{i,k}$ $(1 \leq i \leq n+1)$ is associated with a release time $r_{i,k}$ and a deadline $d_{i,k}$. A rhythmic packet $\chi_{0,k}$ is associated with its nominal release time $\bar{r}_{0,k}$, nominal deadline $\bar{d}_{0,k}$, actual release time $r_{0,k}$ and actual deadline $d_{0,k}$. Note that $\bar{r}_{0,k}$ and $\bar{d}_{0,k}$ are the release time and deadline of $\chi_{0,k}$, respectively, if τ_0 never enters the rhythmic state. In contrast, $r_{0,k}$ and $d_{0,k}$ are the release time and deadline of $\chi_{0,k}$, respectively, in reality, regardless of whether τ_0 is in the nominal or rhythmic state. To simplify the discussion, we use release time (deadline) to refer to actual release time (deadline) in the rest of the paper unless we need to differentiate actual and nominal release times (deadlines). In this work, we assume that $\bar{d}_{0,k} \leq \bar{r}_{0,k+1}$ and $d_{0,k} \leq r_{0,k+1}$ are satisfied.

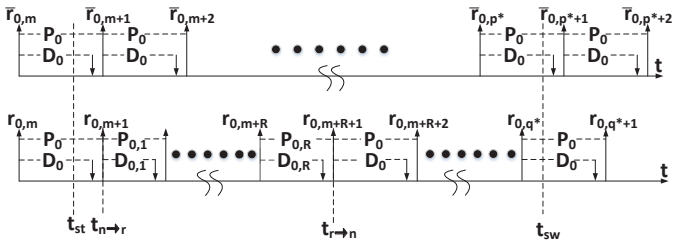

Fig. 1. Nominal (top) and actual (bottom) release times and deadlines of rhythmic packets. An upper arrow represents a nominal and actual release time while a down arrow represents a nominal and actual deadline.

Figure 1 shows the nominal release times, nominal deadlines, actual release times and actual deadlines of rhythmic packets when τ_0 is in the two different states. The top part of the figure shows the nominal release times (deadlines) of rhythmic packets while the bottom part of the figure shows the actual release times (deadlines) of rhythmic packets. We assume that τ_0 is in the nominal state initially. It switches to the rhythmic state at $r_{0,m+1}$ (denoted as $t_{n\to r}$), and returns to the nominal state at $r_{0,m+\mathcal{R}+1}$ (denoted as $t_{r\to n}$). That is, τ_0 stays in the rhythmic state within interval $[t_{n\to r}+1, t_{r\to n}]$, where $t_{n\to r}$ and $t_{r\to n}$ satisfy

$$t_{r\to n} = t_{n\to r} + \sum_{x=1}^{\mathcal{R}} P_{0,x}. \tag{1}$$

The actual release time $r_{0,k+1}$ is determined based on the state of τ_0 when $r_{0,k}$ is released, *i.e.*,

$$r_{0,k+1} = \begin{cases} r_{0,k} + P_0 & \text{if } k \leq m \text{ or } k > m+R \\ r_{0,k} + P_{0,k-m} & \text{if } m < k \leq m+R \end{cases}. \tag{2}$$

Similarly, $d_{0,k+1}$ of $\chi_{0,k+1}$ is equal to

$$d_{0,k+1} = \begin{cases} r_{0,k+1} + D_0 & \text{if } k < m \text{ or } k \geq m+R \\ r_{0,k+1} + D_{0,k+1-m} & \text{if } m \leq k < m+R \end{cases}. \tag{3}$$

Following WirelessHART [27] and ISA100.11a [1], the two prevalent international communication standards designed for wireless sensing and control systems, we apply a TDMA data link layer in our model. A node in the network follows a given schedule to transmit or receive packets. Specifically, at the h-th hop along the routing path of rhythmic or periodic task τ_i, the sender sends packet $\chi_{i,k}$ to the receiver and receives an acknowledgement packet from the receiver. In contrast, at the h-th hop of broadcast task τ_{n+1}, the sender sends packet to one or multiple receivers by following the broadcast graph. We refer to the delivery of a packet at one hop as a transmission and denote it by $\chi_{i,k}(h)$, $h = 1, ..., H_i$. A transmission must

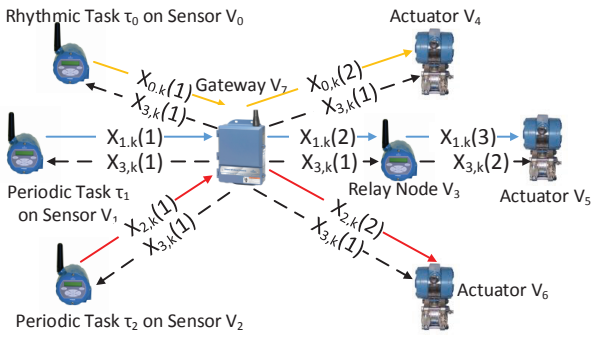

Fig. 2. An example WNCS with 4 tasks running on 8 Nodes.

TABLE I. PARAMETERS OF THE 4 TASKS IN THE EXAMPLE WNCS

Task	Routing Path	Period	Relative Deadline	Period Vector	Deadline Vector
τ_0	$V_0 \rightarrow V_7 \rightarrow V_4$	9	6	$[3,6]^T$	$[2,5]^T$
τ_1	$v_1 \rightarrow v_7 \rightarrow v_3 \rightarrow v_5$	9	7	N/A	N/A
τ_2	$V_2 \rightarrow V_7 \rightarrow V_6$	9	9	N/A	N/A
τ_3	$V_7 \rightarrow *$[1]	N/A	N/A	N/A	N/A

be completed within one time slot. Similar to the definition of packet, a transmission can be either rhythmic, periodic or broadcast. Transmission $\chi_{i,k}(h)$ is associated with release time $r_{i,k}(h)$, deadline $d_{i,k}(h)$ and finish time $f_{i,k}(h)$. Finish time $f_{i,k}(h)$ represents the time slot when $\chi_{i,k}$ completes the h-th hop. For transmission $\chi_{i,k}(h)$ $(h > 1)$, its release time $r_{i,k}(h)$ is the finish time $f_{i,k}(h-1)$ of $\chi_{i,k}(h-1)$. For the reader's convenience, we summarize the set of frequently used symbols and notations in Table II.

Figure 2 illustrates an example WNCS which contains 4 tasks, *i.e.*, τ_0, τ_1, τ_2 and τ_3 running on 8 nodes, *i.e.*, V_i, $i = 0, \ldots, 7$, where V_0, V_1, V_2 are sensors, V_4, V_5, V_6 are actuators, V_3 is a relay node and V_7 is the gateway. Task τ_0 is the rhythmic task, tasks τ_1 and τ_2 are periodic tasks, and task τ_3 is a broadcast task. Each task τ_i has one packet $\chi_{i,k}$ which is composed of a series of transmissions $\chi_{i,k}(h)$'s. Their routing paths, periods and relative deadlines are given in columns 2, 3 and 4 of Table I, respectively. In addition, τ_0 has $\vec{P}_0$ and $\vec{D}_0$ equal to $[3,6]^T$ and $[2,5]^T$, respectively.

Based on the presented system model, we shall elaborate in the following of the paper how to perform effective on-line data link layer schedule adjustment to accommodate rhythmic task instances in the presence of physical disturbances.

III. PROBLEM FORMULATION

In this section, we first give a motivational example to show the deficiency of using a static schedule in our system model. We then describe the formulation of the on-line data link layer scheduling problem in detail.

A. Motivation

Consider the example given in Figure 2. It contains one rhythmic task τ_0, two periodic tasks τ_1 and τ_2 and one broadcast task τ_3. The periodic tasks and rhythmic task are

[1]Task τ_3 is a broadcast task, which has 2 hops. The first hop is from V_7 to V_0, V_1, V_2, V_3, V_4, V_6, and the second hop is from V_3 to V_5.

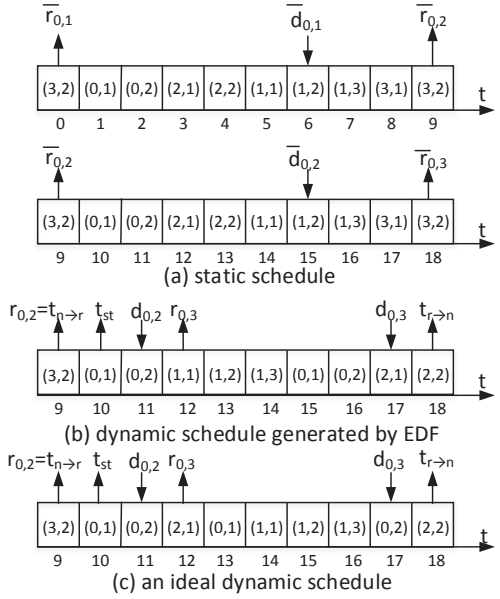

Fig. 3. The time slot assignment in static schedule $\bar{S}$ ((a)) and two alternative dynamic schedules ((b) and (c)) in the motivational example. (i,h) within a slot indicates that this slot is assigned to the h-th hop of task τ_i.

synchronous and packets $\chi_{0,1}$, $\chi_{1,1}$ and $\chi_{2,1}$ are released at time slot 0. The predetermined static schedule for the task set is shown in Figure 3(a). Suppose at time slot $t_{n\rightarrow r}$(=9), τ_0 enters the rhythmic state, and employs $\vec{P}_0 = [3,6]^T$ and $\vec{D}_0 = [2,5]^T$ as the vectors of periods and relative deadlines, respectively. Hence, τ_0 returns to the nominal state at time slot $t_{r\rightarrow n} = t_{n\rightarrow r} + P_{0,1} + P_{0,2} = 18$. If we continue to use the static schedule after $t_{n\rightarrow r}$, packet $\chi_{0,3}$ released at time slot 12 cannot meet its deadline at time slot 17 . Therefore, the network cannot use the static schedule to properly respond to the period and deadline changes of τ_0 in its rhythmic state.

A possible way to make all rhythmic packets schedulable is to calculate the minimal allowed period of τ_0 while scheduling rhythmic and period tasks under EDF. The network then can employ a static schedule whose size is the least common multiple of periods of the rhythmic task and periodic tasks. In this schedule, τ_0 always transmits packets with the minimal allowed period. According to the schedulability analysis under EDF in [3], the minimal allowed period of τ_0 in our example is 5, which is larger than the first period (=3) in $\vec{P}_0$. Hence, we cannot find a feasible static schedule for the given task set.

An alternative is to design a temporary dynamic schedule for the network when τ_0 is in its rhythmic state, and let the system reuse the static schedule when τ_0 returns to its nominal state. One example dynamic schedule is shown in Figure 3 (b). It does not drop any packet, and compared to the static schedule in Figure 3 (a), it introduces 7 slot assignment updates (slot 12-18). In contrast, Figure 3 (c) gives another dynamic schedule which also does not drop any packet but only needs to update 3 slot assignments (slot 13, 17 and 18). This observation motivates our work to determine the optimal dynamic schedule, in which all rhythmic task instances can meet their deadlines and the number of dropped periodic packets is minimized.

B. Problem Statement

At the highest level, the problem that we want to tackle is finding a dynamic schedule that (i) can be used on-line when

disturbance happens, (ii), finishes all the rhythmic packets by their deadlines, (iii) drops minimum number of periodic packets, and (iv) incurs small overhead. Below, we first discuss the execution model, then formulate the constraints and objective for the problem, and finally define the problem formerly.

In our system model, the network starts by following a static schedule $\bar{S}$ assuming that τ_0 has never entered the rhythmic state. $\bar{S}$ is designed off-line and can guarantee that all rhythmic and periodic packets meet their nominal deadlines. A static schedule is defined as $\bar{S} = \{(t, i, h)\}$, where t is the time slot identifier, i is the task identifier and h is the hop index. Given any time slot t, we have $\bar{S}[t] = (i, h)$. If no task uses slot t, it is idle and we have $\bar{S}[t] = (-1, -1)$. The length $\mathcal{L}$ of $\bar{S}$ is the least common multiple of the task periods (assuming τ_0 is in the nominal state) and the schedule is followed in a cyclic fashion, *i.e.*, $\bar{S}[t] = \bar{S}[t + \mathcal{L}]$. ($\bar{S}$ can be constructed using the approaches presented in [13] and [24], and further discussion of this is out of the scope of this paper.)

When τ_0 enters the rhythmic state, the workload of the system is changed and a dynamic schedule S is thus needed before the system resumes using $\bar{S}$. S starts from start point t_{st} and ends at switch point t_{sw}, where t_{st} and t_{sw} are the first and last slots of S, respectively. Suppose that when $\chi_{0,m}$ reaches the gateway V_g at time slot $f_{0,m}(h)$, V_g determines that τ_0 should switch from the nominal to rhythmic state at $t_{n\to r}$ in order to handle a detected disturbance. It then piggybacks the information of S to a broadcast packet $\chi_{n+1,k}$ and propagates S to all nodes in the network by following $\bar{S}$. $\bar{S}$ pre-allocates slots to τ_{n+1} with period P_0 in order to ensure that S reaches all nodes in the network since it is unpredictable when τ_0 enters the rhythmic state and which nodes need to update the schedule. Without loss of generality, the instance index k of such a broadcast packet $\chi_{n+1,k}$ is set to the instance index m of rhythmic packet $\chi_{0,m}$. Furthermore, we set deadline $d_{n+1,m}$ of $\chi_{n+1,m}$ to $t_{n\to r}$ such that the propagation of S can be completed by $t_{n\to r}$. To guarantee that $\chi_{n+1,m}$ is broadcast after the generation of S and reaches all nodes in the network by $d_{n+1,m}$, $\bar{S}$ must satisfy

$$f_{0,m}(h) + 1 < f_{n+1,m}(1) \tag{4}$$

and

$$f_{n+1,m}(H_{n+1}) \le d_{n+1,m}. \tag{5}$$

Start point t_{st} of S is set to the slot just after the finish time of $\chi_{n+1,m}$, *i.e.*, $f_{n+1,m}(H_{n+1})+1$. From slot $t_{sw}+1$, the system resumes the use of $\bar{S}$ and is ready to handle new disturbances experienced by any tasks. Figure 4 summarizes the operations of the system after the gateway receives packet $\chi_{0,m}$.

Based on the execution model discussed above, updated slots must be piggybacked to a broadcast packet. Since the payload size of a broadcast packet $\chi_{n+1,k}$ is bounded, only the information about those slots whose assignments (*i.e.*, task identifier and hop index) have been changed from $\bar{S}$ is piggybacked to $\chi_{n+1,k}$. To help reduce the count of updated slots, in this work, a slot set to idle in S is not treated as an updated slot. (This treatment not only reduces the number of updated slots but also will be exploited to reduce the search space. More details on the latter will be given in Section IV-A.) A slot t is an updated slot if and only if $S[t]$ and $\bar{S}[t]$ are

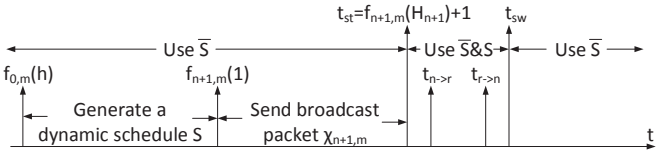

Fig. 4. Operations of the WNCS after the gateway V_g receives $\chi_{0,m}$.

different, and t is not set to idle in S, *i.e.*,

$$S[t] = (i, h) \neq (-1, -1) \text{ and } \bar{S}[t] \neq (i, h). \tag{6}$$

During $[t_{st}, t_{sw}]$, the WNCS follows the three rules below.

1) If t is an updated slot in S, $S[t] = (i, k)$ (*i.e.*, $t = f_{i,k}(h)$), and $\chi_{i,k}(h)$ is released but not finished earlier than slot t, then the WNCS completes $\chi_{i,k}(h)$ at t.
2) If t is not an updated slot in S, $\bar{S}[t] = (i, k)$, and $\chi_{i,k}(h)$ is released but not finished earlier than slot t, then the WNCS completes $\chi_{i,k}(h)$ at t.
3) Otherwise, the WNCS remains idle at t.

By following the three rules, the WNCS actually executes the combination of $\bar{S}$ and S (denoted as $\bar{S}\&S$) during $[t_{st}, t_{sw}]$. We note that executing $\bar{S}\&S$ does not influence the schedulability of both the rhythmic and periodic tasks if they are already schedulable in S, and summarize this in Theorem 1.

Theorem 1. *If a transmission is finished at slot t in S, this transmission can be finished earlier than or equal to t in $\bar{S}\&S$.*

The proofs for all theorems and lemmas in this paper are presented in the appendix.

Based on Theorem 1, we only need to focus on the generation of $S[t_{st}, t_{sw}]$ and let the WNCS follow $\bar{S}\&S$. If the system is overloaded, some periodic packets can be dropped to ensure the schedulability of rhythmic packets. However, the performance of WNCS is thus degraded. Therefore, the primary goal of on-line data link layer scheduling is to make as many periodic packets meet their deadlines as possible, *i.e.*,

$$\min \quad S[t_{st}, t_{sw}].\rho, \tag{7}$$

where $S[t_{st}, t_{sw}].\rho$ is the number of periodic packets dropped by $S[t_{st}, t_{sw}]$. This optimization objective is subject to a number of constraints. First of all, each rhythmic packet $\chi_{0,k}$ must meet its deadline $d_{0,k}$, *i.e.*, $\chi_{0,k}(H_0)$ satisfies

Constraint 1. $f_{0,k}(H_0) \le d_{0,k}$.

To ensure that $\bar{S}$ can be resumed from $t_{sw}+1$, switch point t_{sw} (*i.e.*, the end time of S) is required to be larger than or equal to $t_{r\to n}$. In this work, we assume that physical disturbances occur sporadically [2], [28]. To respond to these events, τ_0 needs to enter the rhythmic state accordingly. We define t^u_{sw} to be the latest release time of a rhythmic packet (called switch point upper bound) when τ_0 re-enters the rhythmic state after $t_{r\to n}$. To guarantee the performance of the WNCS, t_{sw} is required to be smaller than or equal to t^u_{sw}. Thus, we have

Constraint 2. $t_{r\to n} \le t_{sw} \le t^u_{sw}$.

Since the size of the broadcast packet is limited, we set a maximum allowed number of updated slots in S and denote

it by Δ^u. Let $S[t_{st}, t_{sw}].\delta$ be the actual number of updated slots. Then, we have the following constraint

Constraint 3. $S[t_{st}, t_{sw}].\delta \leq \Delta^u$.

In all, our aim is to solve the following problem:

On-Line Scheduling with Bounded Overhead (OLSBO): Given task set $\mathcal{T}$, $\bar{S}$, $t_{n\to r}$, t_{st}, t^u_{sw}, and Δ^u, determine t_{sw} and $S[t_{st}, t_{sw}]$ such that objective function (7) is achieved and Constraint 1, 2, and 3 are satisfied.

Since the gateway starts to compute $S[t_{st}, t_{sw}]$ at $f_{0,m}(h)+1$ and the earliest slot reserved for $\chi_{n+1,m}$ in $\bar{S}$ is $f_{n+1,m}(1)$, the maximum allowed time for solving OLSBO, T_{max}, is

$$T_{max} = f_{n+1,m}(1) - 1 - f_{0,m}(h). \tag{8}$$

IV. Overall Approach

In this section, we present an on-line scheduling framework to solve OLSBO. We first introduce a new constraint to reduce the solution space of OLSBO in Section IV-A. We then discuss how to decide the switch point candidates and their corresponding transmission sets in Section IV-B and Section IV-C. In Section IV-D, we describe our on-line scheduling framework to construct the optimal dynamic schedule.

A. Reducing Solution Space

To reduce the solution space of OLSBO, we require that slot t cannot be set to idle in S if t is assigned to the h-th hop of τ_i in $\bar{S}$ and $\chi_{i,k}(h)$ is released before t but finished after t. Therefore, we add the following constraint to OLSBO.

Constraint 4. *If $\bar{S}[t] = (i,h) \neq (-1,-1)$, $r_{i,k}(h) < t$ and $f_{i,k}(h) > t$, then t cannot be set to idle,* i.e.*, $S[t] \neq (-1,-1)$.*

We use OLSBO$^+$ to refer to the new problem with reduced solution space. The following lemma shows the equivalence of Problem OLSBO and OLSBO$^+$ in finding feasible solutions.

Lemma 1. *If S is a feasible solution to Problem OLSBO, Problem OLSBO$^+$ must also have one feasible solution S' which satisfies $S.\rho = S'.\rho$ and $S.\delta = S'.\delta$.*

Based on Lemma 1, we have the following Theorem.

Theorem 2. *If S is the optimal solution to Problem OLSBO, then Problem OLSBO$^+$ has an optimal solution S' that satisfies $S.\rho = S'.\rho$ and $S.\delta = S'.\delta$.*

According to Theorem 2, solving Problem OLSBO is equivalent to solving Problem OLSBO$^+$. Therefore, in the rest of paper, we focus on finding the optimal solution to Problem OLSBO$^+$, which has a much reduced solution space.

B. Determining Switch Point Candidates

Choosing the right t_{sw} is important as it impacts not only the schedulability of system but also the number of updated slots in S. Theoretically, any slot within $[t_{r\to n}, t^u_{sw}]$ can be a switch point, but checking each slot is time consuming. To tackle this challenge, we first identify the following sufficient condition under which $\bar{S}$ can be employed from $t_{sw}+1$.

Theorem 3. *Given time slot t_{sw}, let χ_{0,p^*+1} and χ_{0,q^*+1} be rhythmic packets having the earliest nominal and actual release times later than or equal to t_{sw}, respectively. (See the example in Figure 1). $\bar{S}$ can be employed from $t_{sw}+1$ without dropping any rhythmic packet if the following three conditions are satisfied by $S[t_{st}, t_{sw}]$.*

Condition 1. *$t_{sw} \geq t_{r\to n}$ and $\bar{r}_{0,p^*+1} = r_{0,q^*+1}$.*

Condition 2. *All transmissions of rhythmic packet $\chi_{0,k}$ released earlier than r_{0,q^*} ($k < q^*$) must be finished by $d_{0,k}$.*

Condition 3. *Let $\bar{S}[t_{h_0}] = (0, h_0)$ and $\bar{S}[t_{H_0}] = (0, H_0)$, where t_{h_0} is the earliest slot serving τ_0 after t_{sw} in $\bar{S}[t]$ and is reserved for the h_0-th hop of τ_0, and t_{H_0} is the earliest time slot reserved for the H_0-th hop of τ_0 after t_{sw} in $\bar{S}[t]$. If $t_{H_0} > d_{0,q^*}$, all transmissions of χ_{0,q^*} are finished by $\min\{t_{sw}, d_{0,q^*}\}$. Otherwise, $\chi_{0,q^*}(h_0-1)$ is finished by t_{sw}.*

Our goal is then to select a t_{sw} such that the conditions in Theorem 3 are satisfied. Although it is easy to select a t_{sw} satisfying $t_{sw} \geq t_{r\to n}$, $\bar{r}_{0,p^*+1} = r_{0,q^*+1}$ in Condition 1 may not always be satisfied. In this case, we explore different methods to realize $\bar{r}_{0,p^*+1} = r_{0,q^*+1}$ assuming t_{sw} is determined. The first method, referred to as *earlier release method*, is to shorten the time interval between r_{0,q^*} and r_{0,q^*+1} by shifting r_{0,q^*+1} backward to the closest nominal release time of τ_0, $\bar{r}_{0,p^*+1}$, which makes Condition 1 in Theorem 3 satisfied. Implementing such a shift only needs all nodes in the WNCS to follow $\bar{S}$ from $t_{sw}+1$. If $d_{0,q^*} > \bar{r}_{0,p^*+1}$, d_{0,q^*} is adjusted to $\bar{r}_{0,p^*+1}$ such that the deadline d_{0,q^*} of χ_{0,q^*} is smaller than or equal to the release time r_{0,q^*+1} of χ_{0,q^*+1}. Therefore, the deadline d_{0,q^*} of χ_{0,q^*} is calculated by

$$d_{0,q^*} = \min\{d_{0,q^*}, \bar{r}_{0,p^*+1}\}. \tag{9}$$

Since Theorem 3 assumes $t_{sw} \geq r_{0,q^*}$ and the earlier release method requires $r_{0,q^*+1} \geq \bar{r}_{0,p^*+1}$, this method requires

$$\bar{r}_{0,p^*} \leq r_{0,q^*} < t_{sw} \leq \bar{r}_{0,p^*+1} \leq r_{0,q^*+1}, \tag{10}$$

as shown in Figure 1. If the other two conditions in Theorem 3 are still satisfied, then $\bar{S}$ can be employed from $t_{sw}+1$.

Another method is to shorten the nominal release time interval between $\bar{r}_{0,p^*}$ and $\bar{r}_{0,p^*+1}$ and shift $\bar{r}_{0,p^*+1}$ to r_{0,q^*+1} for $\bar{r}_{0,p^*+1} > r_{0,q^*+1}$. Such a shift also makes Condition 1 satisfied. However, this method needs to adjust the clocks of all nodes in the WNCS to follow $\bar{S}$ from $t_{sw}+1$, which may not be suitable for on-line use. There are also other possible methods to make $\bar{r}_{0,p^*+1}$ equal to r_{0,q^*+1}. For example, we can either reduce the periods $P_{0,x}$'s or evenly shorten the constant period P_0 to make $r_{0,q+1}$ equal to $\bar{r}_{0,p+1}$. These methods result in high computational overhead within $[t_{st}, t_{sw}]$, which not only increase the number of updated slots in S but also make it hard to find a feasible S. Hence, in this work, we will only focus on the earlier release method.

Since t^u_{sw} is set to the release time of a specific future rhythmic packet in Section III-B and r_{0,q^*+1} is shifted to $\bar{r}_{0,p^*+1}$ by using the earlier release method, t^u_{sw} is also set to the nominal release time of a future rhythmic packet. Thus, we can calculate t^u_{sw} as follows:

$$t^u_{sw} = t_{r\to n} + (\alpha - 1) \cdot P_0, \tag{11}$$

where $\alpha \in N^+$ is called *switch point scaling factor*. Given t^u_{sw}, any time slot within $[t_{r\to n}, t^u_{sw}]$ can be a switch point

candidate. According to the following lemma, we only need to check whether the nominal release times of rhythmic packets can be the switch point.

Lemma 2. *Suppose $S[t_{st}, t^*]$ ($t^* \in [r_{0,q^*} + 1, \bar{r}_{0,p^*+1} - 1]$) satisfies the conditions in Theorem 3 when r_{0,q^*+1} is shifted to $\bar{r}_{0,p^*+1}$ and d_{0,q^*} is set to $\min\{d_{0,q^*}, \bar{r}_{0,p^*+1}\}$. Let $S[t_{st}, \bar{r}_{0,p^*+1}]$ be the schedule generated by concatenating $S[t_{st}, t^*]$ and $\bar{S}[t^* + 1, \bar{r}_{0,p^*+1}]$. Then $S[t_{st}, \bar{r}_{0,p^*+1}]$ satisfies (i) the conditions in Theorem 3, (ii) $S[t_{st}, t^*].\rho = S[t_{st}, \bar{r}_{0,p^*+1}].\rho$ and (iii) $S[t_{st}, t^*].\delta = S[t_{st}, \bar{r}_{0,p^*+1}].\delta$.*

Based on Lemma 2, if no feasible schedule exists when using $\bar{r}_{0,p^*+1}$ as the switch point regardless of which slot in $[t_{r\to n}, t^u_{sw}]$ being used as the switch point, then no feasible schedule exists for whatever slot in $[t_{r\to n}, t^u_{sw}]$ used as the switch point. Therefore, we only consider each nominal release time $\bar{r}_{0,p+1}$ within $[t_{r\to n}, t^u_{sw}]$ as a switch point candidate (denoted as t^c_{sw}). We refer to the set of all these switch point candidates as the switch point candidate set $\Gamma(t^u_{sw})$, where $\Gamma(t^u_{sw}) = \{t^c_{sw} | t^c_{sw} = \bar{r}_{0,p+1}, \forall \bar{r}_{0,p+1} \in [t_{r\to n}, t^u_{sw}]\}$.

C. Constructing Transmission Set

After generating $\Gamma(t^u_{sw})$, the gateway needs to determine which transmissions should be scheduled by S for each switch point candidate t^c_{sw}. In the rest of this paper, we only consider transmissions that have not finished before t_{st} since only such transmissions need to be scheduled by S. Let $\Psi(t^c_{sw})$ be the transmission set containing all transmissions to be finished in $[t_{st}, t^c_{sw}]$ by S. The following lemma determines which rhythmic transmissions belong to $\Psi(t^c_{sw})$.

Lemma 3. *Suppose t_{sw} is set to t^c_{sw} and d_{0,q^*} of χ_{0,q^*} is adjusted to $\min\{d_{0,q^*}, t^c_{sw}\}$, where χ_{0,q^*} is the latest transmission released earlier than t^c_{sw}. A rhythmic transmission $\chi_{0,k}(h)$ belongs to $\Psi(t^c_{sw})$ if and only if $r_{0,k} < t^c_{sw}$. If $S[t_{st}, t^c_{sw}]$ ensures any rhythmic transmission $\chi_{0,k}(h)$ in $\Psi(t^c_{sw})$ finished by $\min\{t^c_{sw}, d_{0,k}\}$, $S[t_{st}, t^c_{sw}]$ satisfies all conditions of Theorem 3.*

In order for the system to resume the use of $\bar{S}$ from $t^c_{sw}+1$, a periodic transmission $\chi_{i,k}(h)$ should be included in $\Psi(t^c_{sw})$ if it is scheduled to be finished within $[t_{st}, t^c_{sw}]$ by S. A transmission $\chi_{i,k}(h)$ ($\tilde{h} \le h \le \hat{h}$) in $\Psi(t^c_{sw})$ is associated with release time $r_{i,k}(h)$, deadline $d_{i,k}(h)$ and finish time $f_{i,k}(h)$. Here, $\tilde{h}$ is the first hop of $\chi_{i,k}(h)$ to reserve a slot in S and $\hat{h}$ is the last hop of $\chi_{i,k}(h)$ to reserve a slot in S. The release time $r_{i,k}(h)$ and deadline $d_{i,k}(h)$ of $\chi_{i,k}(h)$ are calculated by

$$r_{i,k}(h) = \begin{cases} r_{i,k} & \text{if } h = \tilde{h} = 1 \\ t_{st} - 1 & \text{if } h = \tilde{h} > 1 \\ f_{i,k}(h-1) & \text{if } \tilde{h} + 1 \le h \le \hat{h} \end{cases} \quad (12)$$

and

$$d_{i,k}(h) = \min\{t^c_{sw}, d_{i,k}\} - (\hat{h} - h), \quad (13)$$

respectively while the finish time $f_{i,k}(h)$ is to be determined.

D. On-Line Scheduling Framework

Given $\Gamma(t^u_{sw})$ and $\Psi(t^c_{sw})$ for each switch point candidate t^c_{sw}, we are now ready to solve OLSBO$^+$. We design an on-line scheduling (OLS) framework. Every time τ_0 enters the rhythmic state at $t_{n\to r}$, OLS generates a primary dynamic schedule and a backup dynamic schedule for each switch point candidate t^c_{sw}. The primary schedule considers all transmissions in $\Psi(t^c_{sw})$. It aims at finishing all rhythmic packets on time while meeting as many deadlines of periodic packets as possible. In contrast, a backup schedule only assigns time slots to rhythmic transmissions and drops all periodic transmissions. It serves as the baseline dynamic schedule in case no primary schedule is found. If the number of updated slots in a generated (either primary or backup) dynamic schedule is smaller than or equal to Δ^u, this schedule is accepted by OLS and stored in the schedule set $\mathbf{S}$. Finally, OLS selects the best dynamic schedule in terms of fewest number of dropped periodic packets first and fewest number of updated slots second.

OLS determines the dynamic schedule for each switch point candidate in the increasing order of these time points. To speed up the generation of dynamic schedules for the current switch point candidate $\bar{t}^c_{sw}$, OLS reuses partial slot assignments generated for a previous switch point candidate t^c_{sw}. Specifically, every time a backup or primary schedule $S[t_{st}, t^c_{sw}]$ is generated, we identify a time slot t^r_b ($t_{st} \le t^r_b \le t^c_{sw}$, called reusable slot) such that schedule $S^r_b = S[t_{st}, t^r_b]$ (called reusable schedule) can be reused when generating primary or backup dynamic schedules for later switch point candidates. Reusable slot t^r_b is calculated as follows:

$$t^r_b = \min\{r_{i,k} | \forall \chi_{i,k}(h) \in \Psi(t^c_{sw}), d_{i,k} > t^c_{sw}\}, \quad (14)$$

and thus S^r_b only schedules transmissions whose deadlines are earlier than or equal to t^c_{sw}.

Changing switch point from t^c_{sw} to $\bar{t}^c_{sw}$ does not directly help reduce the number of dropped periodic packets or the number of updated slots by $S[t_{st}, t^r_b]$ since $[t_{st}, t^r_b]$ can only be used by transmissions whose deadlines are smaller than or equal to t^c_{sw} according to the definition of t^r_b. Thus, when OLS generates $S[t_{st}, \bar{t}^c_{sw}]$, it only needs to decide $S[t^r_b+1, \bar{t}^c_{sw}]$, and then concatenate S^r_b and $S[t^r_b + 1, \bar{t}^c_{sw}]$ to derive $S[t_{st}, \bar{t}^c_{sw}]$. We use $b = 0$ to indicate that t^r_0 and S^r_0 are used to generate a primary schedule, and use $b = 1$ to show that t^r_1 and S^r_1 are for generating a backup schedule.

The high-level description of OLS framework is given in Algorithm 1. Every time τ_0 enters the rhythmic state at $t_{n\to r}$, OLS initializes a set of variables (Lines 2-3). Here, $\mathbf{S}$ is the schedule set that contains all schedule candidates. S^r_0, S^r_1 and t^r_0, t^r_1 are reusable primary/backup schedules and reusable primary/backup time slots, respectively. Then, OLS constructs switch point candidate set $\Gamma(t^u_{sw})$, where switch point candidates in $\Gamma(t^u_{sw})$ are sorted in the increasing order of their values. For each switch point candidate, OLS first generates a backup schedule (Line 10) and then generates a primary schedule (Line 13). $S[t_{st}, t^c_{sw}]$ is generated by using a slot assignment algorithm such as dynamic programming (DP), EDF and their variations. (More details on DP will be given in Section V). While the generated $S[t_{st}, t^c_{sw}]$ updates more than Δ^u updated slots and still schedules transmissions of at least one periodic packets (Line 15), OLS keeps selecting a periodic packet to drop (Line 16) in order to satisfy Constraint 3. This process is repeated until each $t^c_{sw} \in \Gamma(t^u_{sw})$ has been checked. Finally, V_g selects $S^*[t_{st}, t_{sw}]$ from $\mathbf{S}$ (Line 26) such that

$$S^*[t_{st}, t_{sw}].\rho \le S[t_{st}, t^c_{sw}].\rho, \forall S[t_{st}, t^c_{sw}] \in \mathbf{S}. \quad (15)$$

Algorithm 1 On-Line Scheduling (OLS)

1: Upon τ_0 entering the rhythmic state:
2: $\mathbf{S} = \emptyset$;
3: $t_0^r = t_{st} - 1$, $t_1^r = t_{st} - 1$, $S_0^r = \emptyset$, $S_1^r = \emptyset$;
4: Select $\bar{r}_{0,p+1}$'s as t_{sw}^c's and construct $\Gamma(t_{sw}^u)$;
5: **for** $(\forall t_{sw}^c \in \Gamma(t_{sw}^u))$ **do**
6: Construct $\Psi(t_{sw}^c)$; //Use Lemma 3 and $\bar{S}$ to determine which rhythmic and periodic transmissions belong to $\Psi(t_{sw}^c)$, respectively
7: $b = 1$;
8: **while** $(b \geq 0)$ **do**
9: **if** $(b == 1)$ **then**
10: Generate a backup schedule $S[t_{st}, t_{sw}^c]$ based on $\Psi(t_{sw}^c)$, reusable slot t_0^r and reusable schedule S_0^r;
11: **end if**
12: **if** $(b == 0)$ **then**
13: Generate a primary schedule $S[t_{st}, t_{sw}^c]$ based on $\Psi(t_{sw}^c)$, reusable slot t_1^r and reusable schedule S_1^r;
14: **end if**
15: **while** ($S[t_{st}, t_{sw}^c].\delta > \Delta^u$ and $S[t_{st}, t_{sw}^c]$ schedules transmissions of at least one periodic packets) **do**
16: Drop all transmissions $\chi_{i,k}(h)$'s of periodic packet $\chi_{i,k}$ whose slot reservation in $S[t_{st}, t_{sw}^c]$ causes the maximum number of updated slots;
17: Set associated updated slots to idle in $S[t_{st}, t_{sw}^c]$;
18: **end while**
19: **if** $(S[t_{st}, t_{sw}^c].\delta \leq \Delta^u$) **then**
20: $\mathbf{S} = \mathbf{S} \cup \{S[t_{st}, t_{sw}^c]\}$;
21: Calculate t_b^r and record $S[t_{st}, t_b^r]$ as S_b^r, where $S_b^r \subseteq S[t_{st}, t_{sw}^c]$ is satisfied;
22: **end if**
23: $b = b - 1$;
24: **end while**
25: **end for**
26: **return** $S^*[t_{st}, t_{sw}] \in \mathbf{S}$ where $S^*[t_{st}, t_{sw}].\rho \leq S[t_{st}, t_{sw}].\rho$ among all $S[t_{st}, t_{sw}]$'s in $\mathbf{S}$;

Ties are broken in favour of the minimum number of updated slots first and the earliest switch point candidate second. When T_{max} is reached, OLS is interrupted and the gateway V_g just returns the best dynamic schedule found till now.

V. Dynamic Schedule Generation

In this section, we propose our DP based heuristic to assign time slots to $\Psi(t_{sw}^c)$. The heuristic is called in Lines 10 and 13 of Algorithm 1 to generate a dynamic schedule with the minimum number of dropped periodic packets subject to the bounded update overhead. In Section V-A, we present an exact DP based algorithm given a switch point candidate t_{sw}^c. Since DP needs to check an exponential number of schedules to search for a solution, it is very time consuming and unsuitable for on-line use. Thus, we introduce a heuristic in Section V-B, which is constrained to check a limited number of schedules.

A. Exact Dynamic Programming Algorithm

We first present an exact DP based algorithm (eDP) given a switch point candidate t_{sw}^c. The key idea of eDP is to construct $S[t_{st}, t]$ by determining the assignment of time slot t given $S[t_{st}, t-1]$. We refer to $S[t_{st}, t-1]$ as a parent schedule of $S[t_{st}, t]$ and $S[t_{st}, t]$ a child schedule of $S[t_{st}, t-1]$. The maintenance of schedule $S[t_{st}, t]$ not only influences the efficiency of solution search but also impacts the number of dropped periodic packets within $[t_{st}, t]$. Specifically, there are five issues to be addressed: (i) which schedules should be kept if multiple $S[t_{st}, t]$'s are found for slot t; (ii) which transmission can use slot t; (iii) how to guarantee that any found schedule $S[t_{st}, t]$ satisfies $S[t_{st}, t].\delta \leq \Delta^u$ (Constraint 3); (iv) how to ensure that no rhythmic packet misses its deadline in $S[t_{st}, t]$ (Constraint 1); (v) Constraint 4 is satisfied in any found schedule $S[t_{st}, t]$. We will present the design of eDP to address the aforementioned issues.

Based on a found schedule $S[t_{st}, t]$, we say that $S[t_{st}, t]$ finishes transmission subset $\psi^f(t)$ if and only if each transmission in $\psi^f(t)$ is finished by t according to $S[t_{st}, t]$. According to $\psi^f(t)$ and timing parameters of transmissions in $\Psi(t)$, we can derive which transmissions miss deadlines by t in $S[t_{st}, t]$. We say that $S[t_{st}, t]$ drops transmission subset $\psi^d(t)$ if and only if each transmission in $\psi^d(t)$ misses deadlines by t in $S[t_{st}, t]$. By combining $\psi^f(t)$ and $\psi^d(t)$, we get transmission subset $\psi(t) = \psi^f(t) \cup \psi^d(t)$. We say that $S[t_{st}, t]$ serves $\psi(t)$ if and only if each transmission in $\psi(t)$ is either finished or misses its deadline by t according to $S[t_{st}, t]$. If multiple schedules serve the same transmission subset, such schedules are defined to be equivalent schedules as follows.

Definition 1. *If $S[t_{st}, t]$ and $S'[t_{st}, t]$ serve the same transmission subset $\psi(t)$, $S[t_{st}, t]$ is $S'[t_{st}, t]$'s equivalent schedule (denoted as $S[t_{st}, t] \equiv S'[t_{st}, t]$).*

Since various $\psi^f(t)$'s can be finished within $[t_{st}, t]$, different $\psi(t)$'s can be served by different $S[t_{st}, t]$'s correspondingly.

Our eDP approach follows the general process below to keep schedules for t. (Note that $\Delta^u[t_{st}, t]$ represents the upper bound on the number of updated time slots for $S[t_{st}, t]$.) For each time slot t, each possible value of $\Delta^u[t_{st}, t]$ and each possible transmission subset $\psi(t)$, we find and maintain $S[t_{st}, t]$ in terms of the minimum number of dropped periodic packets among all equivalent schedules serving $\psi(t)$. By using the maintained schedules $S[t_{st}, t]$'s as parent schedules, we generate child schedules $S[t_{st}, t+1]$'s. This process is repeated until we find $S^*[t_{st}, t_{sw}^c]$ (i) resulting in the minimum number of dropped periodic packets among all equivalent schedules serving $\Psi(t_{sw}^c)$ and (ii) satisfying $S^*[t_{st}, t_{sw}^c].\delta \leq \Delta^u$. We provide a more formal description of the eDP approach below.

Let $\mathbf{opt}(\psi^f(t), \Delta^u[t_{st}, t])$, denoting the eDP procedure, return $S[t_{st}, t].\rho$, where (i) $S[t_{st}, t]$ finishes $\psi^f(t)$, (ii) $S[t_{st}, t]$ drops the fewest number of periodic packets among all equivalent schedules serving $\psi(t) = \psi^f(t) \cup \psi^d(t)$, and (iii) $S[t_{st}, t].\delta \leq \Delta^u[t_{st}, t]$. We say that $\mathbf{opt}(\psi^f(t), \Delta^u[t_{st}, t])$ minimizes $S[t_{st}, t].\rho$. We aim to determine

$$\mathbf{opt}(\psi^f(t_{sw}^c), \Delta^u), \tag{16}$$

where $\Psi(t_{sw}^c) = \psi^f(t_{sw}^c) \cup \psi^d(t_{sw}^c)$. The return value of $\mathbf{opt}(\psi^f(t), \Delta^u[t_{st}, t])$ is determined not only by the time slot assignment on t but also by the selection of $S[t_{st}, t-1]$.

Function $\mathbf{opt}(\psi^f(t), \Delta^u[t_{st}, t])$ can be represented by recursive expressions discussed below. We first define function $\phi(S[t] = (i, h), \psi^f(t))$, which returns the number of periodic packets with transmissions missing the deadlines at t due to $S[t] = (i, h)$ while $S[t_{st}, t]$ finishes $\psi^f(t)$. We consider two cases, (i) $t = t_{st}$ and (ii) $t_{st} < t \leq t_{sw}^c$. Case (i) is the initial case of $\mathbf{opt}(\psi^f(t_{st}), \Delta^u[t_{st}, t_{st}])$, and is only determined by

$S[t_{st}]$ since there is no parent schedule at t_{st}. In case (ii), if t is set to idle, we have $(i,h)=(-1,-1)$. Then,

$$\mathbf{opt}(\psi^f(t),\Delta^u[t_{st},t])|_{(-1,-1)} = \mathbf{opt}(\psi^f(t-1),\Delta^u[t_{st},t])+$$
$$\phi(S[t]=(-1,-1),\psi^f(t)) \text{ and } \psi^f(t-1)=\psi^f(t), \quad (17)$$

where function $\mathbf{opt}(\psi^f(t),\Delta^u[t_{st},t])|_{(-1,-1)}$ designates the case of $\mathbf{opt}(\psi^f(t),\Delta^u[t_{st},t])$ when t is set to idle.

We then define *ready transmission of time slot t*, which can potentially use t given $S[t_{st},t-1]$.

Definition 2. *Given $S[t_{st},t-1]$, transmission $\chi^*_{i,k}(h) \in \Psi(t^c_{sw})$ is a ready transmission of slot t if (i) it is not finished by $t-1$ and (ii) $r_{i,k}(h) < t \le d_{i,k}(h)$. Transmission subset $\psi^r(t)$ of $\Psi(t^c_{sw})$ is a ready transmission subset of t if (i) any transmission in $\psi^r(t)$ is a ready transmission of t and (ii) $\psi^r(t)$ contains all ready transmissions of t.*

According to Definition 2, only a ready transmission can be assigned to t given $S[t_{st},t-1]$. If t is assigned to a ready transmission $\chi^*_{i,k}(h)$, we have

$$\mathbf{opt}(\psi^f(t),\Delta^u[t_{st},t])|_{(i,h)} = \mathbf{opt}(\psi^f(t)-\{\chi^*_{i,k}(h)\},\Delta^u[t_{st},t]$$
$$-S[t]|_{(i,h)}.\delta) + \phi(S[t]=(i,h),\psi^f(t)), \quad (18)$$

where function $\mathbf{opt}(\psi^f(t),\Delta^u[t_{st},t])|_{(i,h)}$ designates the case of $\mathbf{opt}(\psi^f(t),\Delta^u[t_{st},t])$ when $\chi^*_{i,k}(h)$ is assigned to t. $S[t]|_{(i,h)}.\delta$ returns 0 if $\bar{S}[t]=(i,h)$ and 1 otherwise. The boundary condition when $t=t_{st}$ for equations (17) and (18) can be set as follows

$$\mathbf{opt}(\psi^f(t_{st}-1),\Delta^u[t_{st},t_{st}-1])=0. \quad (19)$$

By combining equations (17) and (18), we have

$$\mathbf{opt}(\psi^f(t),\Delta^u[t_{st},t]) = \min\{\mathbf{opt}(\psi^f(t),\Delta^u[t_{st},t])|_{(-1,-1)},$$
$$\min_{\chi^*_{i,k}(h)\in\psi^f(t)\cap\psi^r(t)}\{\mathbf{opt}(\psi^f(t),\Delta^u[t_{st},t])|_{(i,h)}\}\}, \ t_{st}\le t\le t^c_{sw}, \quad (20)$$

where $\psi^f(t)\cap\psi^r(t)$ contains all ready transmissions in $\psi^f(t)$. In order to satisfy Constraint 3, we set

$$\mathbf{opt}(\psi^f(t),\Delta^u[t_{st},t]) = +\infty, \text{ if } \Delta^u[t_{st},t]<0. \quad (21)$$

To ensure any generated schedule $S[t_{st},t]$ in eDP satisfying Constraints 1 and 4, we introduce two new definitions: the urgent time slot and favourite time slot.

Definition 3. *A time slot t is an urgent time slot of $\chi^*_{0,k}(h)$ if ready transmission $\chi^*_{0,k}(h)\in\psi^r(t)$ satisfies $d^*_{0,k}(h)=t$.*

An urgent time slot t of $\chi^*_{0,k}(h)$ must be assigned to rhythmic transmission $\chi^*_{0,k}(h)$ since no rhythmic packet can be dropped according to Constraint 1. Hence, we have

$$\phi(S[t]=(i',h'),\psi^f(t))=+\infty, \text{ if } t \text{ is an urgent time slot of } \chi^*_{0,k}(h) \text{ and } (i',k',h')\neq(0,k,h), \quad (22)$$

Definition 4. *A time slot t is a favourite time slot of $\chi^*_{i,k}(h)$ if ready transmission $\chi^*_{i,k}(h)\in\psi^r(t)$ satisfies $\bar{S}[t]=(i,h)$.*

A favourite time slot t of $\chi^*_{i,k}(h)$ cannot be set to idle according to Constraint 4. Therefore, we have

$$\phi(S[t]=(-1,-1),\psi^f(t))=+\infty, \text{ if } t \text{ is a favourite time slot of } \chi^*_{i,k}(h). \quad (23)$$

We claim the optimality of eDP in the following theorem.

Theorem 4. *Assume t^c_{sw} is used as the switch point. The eDP procedure based on (16), (19), (20), (21), (22) and (23) finds a schedule if and only if there exists a solution satisfying all constraints of Problem OLSBO$^+$. In addition, the found solution minimizes the objective function in (7).*

Therefore, OLS-eDP can generate a dynamic schedule if and only if there exists a feasible schedule for Problem OLSBO$^+$, and the found solution minimizes objective function (7) by always setting t^r_b to t_{st} according to Theorem 4. To avoid repeatedly calling recursive function $\mathbf{opt}(t,\psi(t),\Delta^u[t_{st},t])$, we design an iterative version of eDP to solve function (16). The time complexity of eDP is $O(\frac{n^{2t^c_{sw}+2}-1}{n^2-1})$. If there are κ switch point candidates in $\Gamma(t^u_{sw})$, OLS-eDP needs $O(\kappa\cdot\frac{n^{2t^c_{sw}+2}-1}{n^2-1})$ time to find the best dynamic schedule $S^*[t_{st},t_{sw}]$. The pseudo code of the iterative version of eDP is given in the appendix.

B. Modified Dynamic Programming Algorithm

Although DP is optimal in solving Problem OLSBO$^+$ for a given switch point candidate, it is time consuming and unsuitable for on-line use. This is because it constructs an exponential number of child schedules for time slot t and uses all of them as parent schedules for time slot $t+1$. To make the search process more efficient, we modifies DP by reducing the solution space. The heuristic, referred to as mDP, limits the maximum allowed number of maintained child schedules β to a positive integer value at each slot t. Specifically, at time slot t, at most β child schedules that drop the fewest number of packets are maintained and used as parent schedules at $t+1$. The value of β in a benchmark is determined off-line by using the following 2 steps for a specific benchmark.

1) Evaluate OLS-mDP using different values of β.
2) Choose the optimal value of β which results in the minimum number of dropped periodic packets.

The time complexity of mDP is $O(t^c_{sw}\cdot n^2\cdot\beta^2)$. If there are κ switch point candidates in $\Gamma(t^u_{sw})$, OLS-mDP needs $O(\kappa\cdot t^c_{sw}\cdot n^2\cdot\beta^2)$ time to find the best dynamic schedule $S^*[t_{st},t_{sw}]$. For a specific task set, β is set to a value based on the previous numbers of dropped packets on-line.

VI. Performance Evaluation

In this section, we present performance evaluation of our OLS-mDP approach using randomly generated task sets and network topology. We first calibrate the maximum allowed number of maintained schedules (β) and switch point upper bound (t^u_{sw}) in OLS-mDP. We then compare its performance with another efficient approach under different workloads.

A. Simulation Setup

We evaluate the performance of OLS-mDP using 20 groups of task sets with different sizes and utilization levels in a wireless network. The network topology is shown in Figure 9

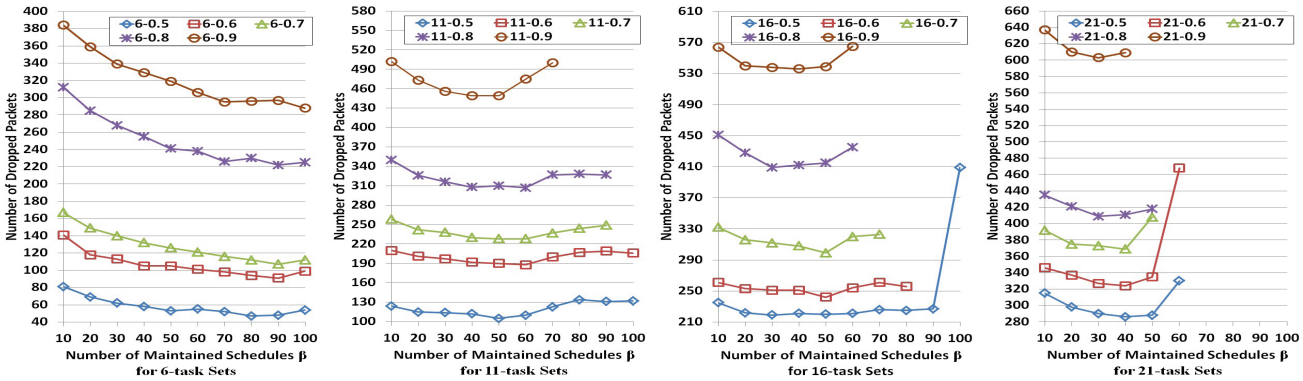

Fig. 5. Number of dropped periodic packets under OLS-mDP with different β values.

in the appendix. Each group, labelled as "(n+2)-u", has 100 task sets. Each task set in group "(n+2)-u" contains n periodic tasks, 1 rhythmic task, 1 broadcast task and has utilization level of u, *i.e.*, $u = \sum_{i=0}^{n+1} \frac{H_i}{P_i}$. We select (n+2) to be 6, 11, 16 and 21 while assigning utilization levels to be 50%, 60%, 70%, 80% and 90%. The routing path of each rhythmic or periodic task is pre-selected randomly and is composed of a chain of 4 to 13 hops from a sensor to an actuator. We use UUnifast algorithm [4] to generate the initial period of each task since UUnifast provides better control on how to assign periods to tasks than a random assignment. Following the WirelessHART protocol, we then adjust the generated task period to the value closest to 2^a ($a \leq 11$) in order to limit the size of $\bar{S}$. Each task set is generated with the guarantee that its utilization level is always equal to the specified utilization level.

We compare OLS-mDP with another time slot assignment approach which also uses the framework of OLS but employs a modified EDF algorithm to generate dynamic schedules (called OLS-mEDF). For each t_{sw}^c in $\Gamma(t_{sw}^u)$, OLS-mEDF first uses On-Line Distributed Algorithm (OLDA) [2] to test the schedulability of $\Psi(t_{sw}^c)$. If $\Psi(t_{sw}^c)$ is found unschedulable, some periodic packets are dropped to ensure the schedulability of $\Psi(t_{sw}^c)$ by following packet drop policy MLET presented in [15]. OLS-mEDF then applies EDF on $\Psi(t_{sw}^c)$ to generate a backup dynamic schedule and a primary dynamic schedule for each t_{sw}^c. A backup schedule can be used when eventually T_{max} is reached and no primary schedule is found by OLS-mDP. OLS-mEDF is very efficient in generating dynamic schedules since EDF is optimal on a uniprocessor [3].

We implement OLS-mDP and OLS-mEDF in C++ and collect simulation data on a Sun Ultra 20 (x86-64) workstation with Red Hat Enterprise Linux 6.5. We compare the performance of both approaches using two metrics. The first metric is the number of feasible task sets and the second metric is the average drop rate for the same feasible task sets identified by both approaches. The average drop rate is the ratio between the number of dropped periodic packets and the number of active periodic packets in $\bar{S}[t_{st}, t_{sw}^c]$. A periodic packet is active during $[t_{st}, t_{sw}^c]$ if and only if at least one of its transmissions is finished within $[t_{st}, t_{sw}^c]$ according to $\bar{S}$. It is possible that both approaches may not employ the same switch point for the same feasible task set. To ensure the fairness of comparison, we use the larger one out of the two switch points when calculating the numbers of dropped and active periodic packets. Both metrics indicate the effectiveness of an approach to generate a dynamic schedule in response to external disturbances.

[2]OLDA [15] is an on-line algorithm which combines local-deadline assignment with schedulability analysis in a distributed real-time system.

B. Parameter Selection for OLS-mDP

Since the performance of OLS-mDP depends on β (the maximum allowed number of maintained child schedules in OLS-mDP) and t_{sw}^u (the upper bound on t_{sw}), we first calibrate these parameters to fully exploit the potential of OLS-mDP in reducing the number of dropped periodic packets. Note that Δ^u (the maximum allowed number of updated slots by $S[t_{st}, t_{sw}]$) is determined by the deployed wireless protocol, which will be discussed in Section VI-C. As a starting point, we test OLS-mDP using different values $(10, 20, 30, \ldots, 100)$ for β while assigning Δ^u and t_{sw}^u both to $+\infty$. The numbers of dropped packets for groups of 6-task, 11-task, 16-task and 21-task sets are shown in Figure 5. Our results show that increasing β can reduce the number of dropped periodic packets initially. The number of dropped packets then fluctuates as β further increases, and eventually rises abruptly when β increases to a certain extent. In our simulation, we observe that 90, 50, 50 and 40 are the best values of β for OLS-mDP to drop the fewest number of packets for groups of 6-task, 11-task, 16-task and 21-task sets, respectively.

Next, we choose the value of t_{sw}^u for OLS-mDP. The goal is to keep it as small as possible without performance degradation. A small t_{sw}^u reduces the computational overhead of running OLS-mDP on the gateway. Recall that we express t_{sw}^u as $t_{sw}^u = t_{r \to n} + (\alpha - 1) \cdot P_0$ in (11). By fixing β to the values selected above, we measured the numbers of dropped packets under OLS-mDP for different α values, $1 \leq \alpha \leq 3$, for all groups of task sets. The results are summarized in Figure 6. We observe that increasing α from 1 to 2 and from 2 to 3 reduce the dropped packet count by 16% and 1% on average (56% and 9% at most), respectively. Without sacrificing the performance of OLS-mDP, we set α to 2.

C. Performance Comparison: OLS-mDP vs. OLS-mEDF

If WirelessHART protocol is employed in response to physical disturbances, no task set can be solved. This is because rhythmic packets will be dropped for each task set when rhythmic task enters the rhythmic state. Thus, we only compared the performance of OLS-mDP with OLS-mEDF in terms of number of feasible task sets and number of dropped packets. We set β and α of OLS-mDP to the values selected above for different groups of task sets. To ensure the fairness of comparison, we also set α to 2 for OLS-mEDF. Following WirelessHART standard, We assume that the payload size of a packet in our experiments is 90 bytes. We use 3 bytes (24 bits) to represent one updated slot, where 13, 7 and 4 bits are used

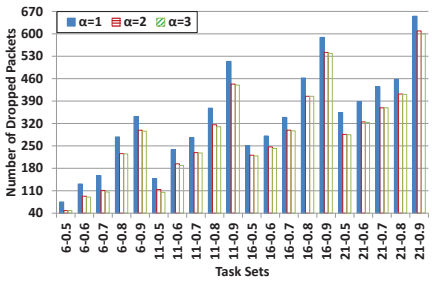

Fig. 6. Number of dropped packets in different task sets under OLS-mDP with different α values.

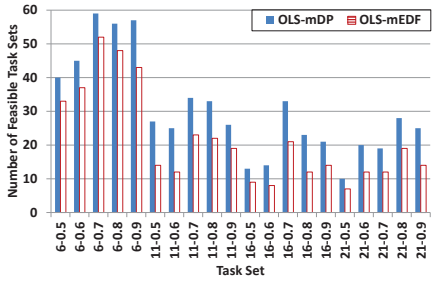

Fig. 7. Number of task sets solved by OLS-mDP and OLS-mEDF with Δ^u equal to 30.

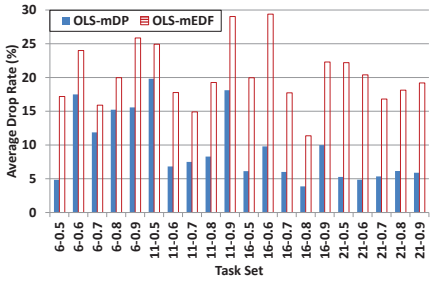

Fig. 8. Comparison of average drop rates of periodic packets in the same feasible task sets.

for slot identifier, task identifier and hop index, respectively. Therefore, one broadcast packet can accommodate information of at most 30 updated slots, *i.e.*, $\Delta^u = 30$.

Figure 7 shows the number of task sets solved by OLS-mDP and OLS-mEDF. One can readily see that OLS-mDP finds 41% (108%) more task sets on average (at most) than OLS-mEDF. Out of 2000 task sets, 431 task sets are solved by both OLS-mDP and OLS-mEDF. The average drop rates of periodic packets for these task sets are shown in Figure 8. One can observe that OLS-mEDF drops 122% (321%) more packets on average (at most) than OLS-mDP. Since mEDF ignores Constraint 3, *i.e.*, $S[t_{st}, t_{sw}].\delta \leq \Delta^u$, OLS-mEDF either drops too many periodic packets to reduce the number of updated slots (Lines 15-18 of Algorithm 1) or even fails to find a dynamic schedule satisfying this constraint.

We also compared OLS-mDP with OLS-mEDF when setting Δ^u to 60, which represents the case where the wireless protocol provides a higher bandwidth and a larger payload size. The results show that OLS-mDP can solve 11% (31%) more task sets on average (at most) than OLS-mEDF. For the 1765 task sets solved by both approaches, OLS-mEDF drops 128% (263%) more packets on average (at most) than OLS-mDP. Therefore, OLS-mDP performs better than OLS-mEDF in terms of both the number of feasible task sets and the number of dropped periodic packets.

VII. Conclusion and Future Work

In this paper, we introduce a data link layer scheduling problem to minimize the impact of physical disturbances on existing network flows in a WNCS. We propose an efficient framework which uses an effective heuristic to perform online schedule design. Our approach constructs a dynamic schedule for this time interval with bounded time and schedule adjustment overheads. Our simulation results validate the effectiveness of our approach. As future work, we will extend our system model to support multiple communication channels and allow multiple rhythmic tasks to enter the rhythmic state simultaneously. We will explore more efficient algorithms to generate the dynamic schedule. Moreover, we plan to implement our approach in a WNCS testbed and evaluate its performance in real-world applications.

References

[1] "ISA100," http://www.isa.org/isa100.

[2] J. Araujo, A. Teixeira, E. Henriksson, and K. H. Johansson, "A down-sampled controller to reduce network usage with guaranteed closed-loop performance," in *CDC*, 2014.

[3] S. K. Baruah, L. E. Rosier, and R. R. Howell, "Algorithms and complexity concerning the preemptive scheduling of periodic, real-time tasks on one processor," *Real-Time Syst*, 1990.

[4] E. Bini and G. C. Buttazzo, "Biasing effects in schedulability measures," in *ECRTS*, 2004.

[5] G. Buttazzo, E. Bini, and D. Buttle, "Rate-adaptive tasks: Model, analysis, and design issues," in *DATE*, 2014.

[6] O. Chipara, C. Lu, and G.-C. Roman, "Real-time query scheduling for wireless sensor networks," in *RTSS*, 2007.

[7] O. Chipara, C. Wu, C. Lu, and W. G. Griswold, "Interference-aware real-time flow scheduling for wireless sensor networks," in *ECRTS*, 2011.

[8] T. Chiueh, R. Krishnan, P. De, and J.-H. Chiang, "A networked robot system for wireless network emulation," in *RoboComm*, 2007.

[9] T. L. Crenshaw, S. Hoke, A. Tirumala, and M. Caccamo, "Robust implicit edf: A wireless mac protocol for collaborative real-time systems," *ACM Trans. Embed. Comput. Syst.*, 2007.

[10] F. Ferrari, M. Zimmerling, L. Mottola, and L. Thiele, "Low-power wireless bus," in *SenSys*, 2012.

[11] K. Gatsis, A. Ribeiro, and G. Pappas, "Optimal power management in wireless control systems," in *ACC*, 2013.

[12] R. Gupta and M.-Y. Chow, "Networked control system: Overview and research trends," *IEEE Trans. Ind. Electron.*, 2010.

[13] S. Han, X. Zhu, D. Chen, A. K. Mok, and M. Nixon, "Reliable and real-time communication in industrial wireless mesh networks," in *RTAS*, 2011.

[14] X. Hei, X. Du, S. Lin, and I. Lee, "Pipac: Patient infusion pattern based access control scheme for wireless insulin pump system," in *INFOCOM*, 2013.

[15] S. Hong, T. Chantem, and X. S. Hu, "Meeting end-to-end deadlines through distributed local deadline assignments," in *RTSS*, 2011.

[16] B. Hu and M. D. Lemmon, "Using channel state feedback to achieve resilience to deep fades in wireless networked control systems," in *HiCoNS*, 2013.

[17] V. M. Karbhari and F. Ansari, "Structural health monitoring of civil infrastructure systems," CRC Press, 2009.

[18] J. Kim, K. Lakshmanan, and R. Rajkumar, "Rhythmic tasks: A new task model with continually varying periods for cyber-physical systems," in *ICCPS*, 2012.

[19] Q. Leng, Y.-H. Wei, S. Han, A. K. Mok, W. Zhang, and M. Tomizuka, "Improving control performance by minimizing jitter in RT-WiFi networks," in *RTSS*, 2014.

[20] L. Li, B. Hu, and M. Lemmon, "Resilient event triggered systems with limited communication," in *CDC*, 2012.

[21] S. K. Mazumder, "Wireless networking based control," Springer, 2011.

[22] P. Naghshtabrizi and J. P. Hespanha, "Implementation considerations for wireless networked control systems," Springer, 2011.

[23] M. Pajic, S. Sundaram, G. J. Pappas, and R. Mangharam, "The wireless control network: A new approach for control over networks," *IEEE Trans. Automat. Contr.*, 2011.

[24] A. Saifulah, C. Lu, Y. Xu, and Y. Chen, "Real-time scheduling for WirelessHART networks," in *RTSS*, 2010.

[25] M. Sha, R. Dor, G. Hackmann, C. Lu, T.-S. Kim, and T. Park, "Self-adapting mac layer for wireless sensor networks," in *RTSS*, 2013.

[26] W. Shen, T. Zhang, M. Gidlund, and F. Dobslaw, "SAS-TDMA: a source aware scheduling algorithm for real-time communication in industrial wireless sensor networks," *WIREL NETW*, 2013.

[27] J. Song, S. Han, A. K. Mok, D. Chen, M. Lucas, M. Nixon, and W. Pratt, "WirelessHART: Applying wireless technology in real-time industrial process control," in *RTAS*, 2008.

[28] J. Verhaegh, T. Gommans, and W. Heemels, "Extension and evaluation of model-based periodic event-triggered control," in *ECC*, 2013.

[29] Y.-H. Wei, Q. Leng, S. Han, A. K. Mok, W. Zhang, and M. Tomizuka, "RT-WiFi: Real-time high-speed communication protocol for wireless cyber-physical control applications," in *RTSS*, 2013.

[30] R. Yedavalli and R. Belapurkar, "Application of wireless sensor networks to aircraft control and health management systems," *IJCTA*, 2011.

APPENDIX

Proof for Theorem 1: Suppose $S[t] = (i,k)$. If t is an updated slot, and $\chi_{i,k}(h)$ is released but not finished earlier than t, then $\chi_{i,k}(h)$ is still finished at t by following Rule 1. If t is an updated slot, and $\chi_{i,k}(h)$ is finished earlier than slot t, then the finish time of $\chi_{i,k}(h)$ is earlier than $f_{i,k}(h)$ in S by following Rule 3. If $\chi_{i,k}(h)$ is not released at slot t, then $S[t]$ should not be set to (i,k), which is a contradiction to the condition $S[t] = (i,k)$.

If t is not an updated slot, $\bar{S}[t] = (i,k)$ is satisfied. Similar to the proof in the above case, we can prove that $\chi_{i,k}(h)$ is still finished earlier than or equal to $f_{i,k}(h)$ by employing rules 2 and 3. Hence, the theorem is proved. □

Proof for Lemma 1: We first prove that if Problem OLSBO has a feasible solution S, we can find at least one feasible solution to Problem OLSBO$^+$. If S does not violate Constraint 4, it is also a feasible solution to Problem OLSBO$^+$. Otherwise, we show below that we can modify S to find a feasible solution to Problem OLSBO$^+$.

Suppose that the assignment of slot t in S violates Constraint 4, *i.e.*, $S[t] = (-1,-1)$, $\bar{S}[t] = (i,h)$, $r_{i,k}(h) < t$ and $f_{i,k}(h) > t$. Thus, we can find a slot t' that is after t and assigned to $\chi_{i,k}(h)$ in S, *i.e.*, $S[t'] = (i,h)$ and $t' = f_{i,k}(h) > t$. We can swap the assignments of t and t' such that $S[t] = (i,h)$ and $S[t'] = (-1,-1)$. Such a swap does not violate the precedence requirement of transmissions because $f_{i,k}(h-1) = r_{i,k}(h) < t$ and $\chi_{i,k}(h-1)$ is finished earlier than t. By repeatedly applying the swapping scheme, all slot assignments violating Constraint 4 can be adjusted in a similar way, and the resulting schedule S' is a feasible solution to Problem OLSBO$^+$.

Next, we prove that $S.\rho = S'.\rho$ and $S.\delta = S'.\delta$ are satisfied. Since the WNCS finishes $\chi_{i,k}(h)$ earlier by using the swap, no more packets are dropped due to this swap. In addition, t and t' are not updated slots because $\bar{S}[t] = (i,h)$ is satisfied and t' becomes idle after the swap. Thus, the lemma is proved. □

Proof for Theorem 2: According to Lemma 1, OLSBO$^+$ has a feasible solution S' satisfying $S.\rho = S'.\rho$ and $S.\delta = S'.\delta$. If we can prove the claim that S' is an optimal solution to OLSBO$^+$, the theorem is proved. Now we prove the claim by contradiction.

Suppose we can find an optimal solution S'' to OLSBO$^+$ that satisfies $S''.\rho < S'.\rho$. Since $S''.\rho < S'.\rho = S.\rho$, S can not be an optimal solution to Problem OLSBO. This contradicts the condition that S is the optimal solution to OLSBO. □

Proof for Theorem 3: If Condition 1 is satisfied, the actual packet release patterns after r_{0,q^*+1} become exactly the same as the nominal packet release patterns. Thus, switching to $\bar{S}$ at r_{0,q^*+1} will not cause any rhythmic packet loss after r_{0,q^*+1}. When Condition 2 is satisfied, all rhythmic packets released before r_{0,q^*} meet their deadlines. By satisfying Condition 3, χ_{0,q^*} is guaranteed to always finish by its deadline d_{0,q^*}. Specifically, if $t_{H_0} > d_{0,q^*}$, all transmissions of χ_{0,q^*} are finished by $\min\{t_{sw}, d_{0,q^*}\}$. Hence, χ_{0,q^*} can meet its deadline d_{0,q^*}. If there is any time slots reserved for transmissions of τ_0 in $\bar{S}[t_{sw}+1, \bar{r}_{0,p^*+1}]$, the WNCS just keeps them idle. In the case $t_{H_0} \le d_{0,q^*}$, $\chi_{0,q^*}(h_0-1)$ is finished by t_{sw} in S. Thus, $\chi_{0,q^*}(h_0), \ldots, \chi_{0,q^*}(H_0)$ can be completed by d_{0,q^*} by following $\bar{S}$. If transmission $\chi_{0,q^*}(h)$ ($h \ge h_0$) has been completed by t_{sw} and $\bar{S}[t_h] = (0,h)$ ($t_{sw} < t_h \le t_{H_0}$) is satisfied, the WNCS just keeps idle at slot t_h. Hence, $\bar{S}$ can be employed from $t_{sw}+1$ without dropping any rhythmic packet. □

Proof for Lemma 2: With $t_{sw} = \bar{r}_{0,p^*+1} > t^* \ge t_{r\to n}$ and the shift of r_{0,q^*+1} to $\bar{r}_{0,p^*+1}$, Condition 1 is satisfied. Since concatenating $S[t_{st}, t^*]$ and $\bar{S}[t^*+1, \bar{r}_{0,p^*+1}]$ does not change the time slot assignment in $[t_{st}, r_{0,q^*}]$, $S[t_{st}, \bar{r}_{0,p^*+1}]$ guarantees that all rhythmic packets released earlier than r_{0,q^*} meet their deadlines, *i.e.*, Condition 2 is satisfied. Since the WNCS reuses $\bar{S}$ from t^*+1, no slot after $\bar{r}_{0,p^*+1}$ can be used by transmissions of χ_{0,q^*}. Thus, $S[t_{st}, \bar{r}_{0,p^*+1}]$ guarantees that all transmissions of χ_{0,q^*} are finished by $\min\{\bar{r}_{0,p*+1}, d_{0,q^*}\}$. In addition, when setting t_{sw} to $\bar{r}_{0,p^*+1}$, we have $t_{H_0} > \min\{\bar{r}_{0,p^*+1}, d_{0,q^*}\}$ according to the definition of t_{H_0} in Theorem 3. Hence, $S[t_{st}, \bar{r}_{0,p^*+1}]$ also satisfies Condition 3. Furthermore, since there is neither packet drop nor updated slot in $\bar{S}[t^*+1, \bar{r}_{0,p^*+1}]$, $S[t_{st}, t^*].\rho = S[t_{st}, \bar{r}_{0,p^*+1}].\rho$ and $S[t_{st}, t^*].\delta = S[t_{st}, \bar{r}_{0,p^*+1}].\delta$ are satisfied. □

Proof for Lemma 3: First, condition 1 is satisfied because of the definition of t^c_{sw}. Next, we consider two possible cases of rhythmic packet $\chi_{0,k}$, (i) $r_{0,k} < r_{0,q^*}$ and (ii) $r_{0,k} = r_{0,q^*}$. In case (i), all transmissions of $\chi_{0,k}$ satisfy $d_{0,k}$, *i.e.*, Condition 2 is satisfied. In case (ii), with the shift of r_{0,q^*+1} backward to $\bar{r}_{0,p^*+1}$, d_{0,q^*} is updated to $\min\{d_{0,q^*}, t^c_{sw}\}$. Since $t_{H_0} > \min\{d_{0,q^*}, t^c_{sw}\}$ is satisfied according to the definition of t_{H_0} in Theorem 3 and all transmissions of χ_{0,q^*} are finished by $\min\{t^c_{sw}, d_{0,q^*}\}$, Condition 3 is satisfied. □

Proof for Theorem 4: We prove the "if" part first by contradiction. We assume that eDP cannot find a solution. Thus, we have

$$\mathbf{opt}(\psi^f(t), \Delta^u[t_{st}, t]) = +\infty, \tag{24}$$

at a specific slot t for any transmission subset $\psi^f(t)$ finished by any $S[t_{st}, t]$ given any $\Delta^u[t_{st}, t]$. Suppose solution $S[t_{st}, t^c_{sw}]$ satisfies all constraints of Problem OLSBO$^+$. Thus, there exists a specific $S[t_{st}, t]$ satisfying

$$\mathbf{opt}(\psi^f(t), \Delta^u[t_{st}, t]) \neq +\infty, \tag{25}$$

TABLE II. SUMMARY OF USED NOTATIONS (UNLESS SPECIFIED, TASK IDENTIFIER i SATISFIES $0 \leq i \leq n+1$ IN THIS TABLE.)

Parameter	Definition	Parameter	Definition
V_i, $0 \leq i \leq g$	Gateway, sensors, actuators and relay nodes	$\chi_{i,k}$	Rhythmic, periodic and broadcast packet $\chi_{i,k}$
τ_i, $0 \leq i \leq n$	Unicast task	$\chi_{i,k}(h)$	The h-th transmission of $\chi_{i,k}$
τ_{n+1}	Broadcast task	$\chi^*_{i,k}(h)$, $0 \leq i \leq n$	Ready transmission
τ_0	Rhythmic task	$r_{i,k}$, $d_{i,k}$	Release time and deadline of $\chi_{i,k}$
τ_i, $1 \leq i \leq n$	Periodic task	$\bar{r}_{0,k}$, $\bar{d}_{0,k}$	Nominal release time and nominal deadline of $\chi_{0,k}$
H_i	Hop number of τ_i	$r_{i,k}(h)$, $d_{i,k}(h)$	Release time and deadline of
P_i, D_i	Period and relative deadline	$0 \leq i \leq n$	rhythmic or periodic transmission $\chi_{i,k}(h)$
$0 \leq i \leq n$	of rhythmic or periodic task τ_i	$f_{i,k}(h)$	Finish time of $\chi_{i,k}(h)$
$\vec{P}_0$, $\vec{D}_0$	Period and relative deadline vectors of τ_0	$\Gamma(t^u_{sw})$	Set of switch point candidates
t_{st}, t_{sw},	Start point, switch point, switch point		Slot when τ_0 leaves the nominal state
t^c_{sw}, t^u_{sw}	candidate and switch point upper bound	$t_{n\to r}$, $t_{r\to n}$	and the rhythmic state, respectively
$\bar{S}$, S	Static schedule and dynamic schedule		Set of transmissions to be scheduled within
$S[t_{st}, t]$	Dynamic schedule within $[t_{st}, t]$	$\Psi(t_{sw})$, $\Psi(t^c_{sw})$	$[t_{st}, t_{sw}]$ and $[t_{st}, t^c_{sw}]$, respectively
$\bar{S}[t]$, $S[t]$	Assignment of slot t in $\bar{S}$ and S, respectively		The maximum allowed numbers of updated
$S[t_{st}, t].\rho$	Number of periodic packets dropped by $S[t_{st}, t]$	Δ^u, $\Delta^u[t_{st}, t]$	slots for $S[t_{st}, t_{sw}]$ and $S[t_{st}, t]$, respectively
$\bar{S}\&S$	Combination of $\bar{S}$ and S	$S[t_{st}, t].\delta$	Number of updated slots due to $S[t_{st}, t]$
γ	Maximum allowed number of updated slots in DP and mDP		Transmission subsets finished, dropped and
	Maximum allowed number of maintained child	$\psi^f(t)$, $\psi^d(t)$, $\psi(t)$	served by $S[t_{st}, t]$, respectively,
β, α	schedules in DP/mDP and switch point scaling factor	$\psi^r(t)$	Ready transmission subset of t

at each slot t within $[t_{st}, t^c_{sw}]$. Hence, (24) and (25) contradict with each other. Algorithm eDP should be able to find a feasible solution.

We next prove the "only if" part. If eDP finds a schedule $S[t_{st}, t^c_{sw}]$ by solving equations (16), (19), (20), (21), (22) and (23), then there is no rhythmic packet missing deadlines, *i.e.*, Constraint 1 is satisfied. Otherwise, the condition of (22) is satisfied and no solution can be found by eDP. The selection of switch point candidates in Line 4 of Algorithm 1 guarantees that Constraint 2 of Problem OLSBO$^+$ is satisfied. Equations (21) and (23) ensure that Constraints 3 and 4 are satisfied, respectively.

Finally, we prove that the found solution by eDP minimizes objective function (7) among all equivalent schedules serving $\Psi(t^c_{sw})$ by induction. We consider $\mathbf{opt}(\psi^f(t_{st}), \Delta^u[t_{st}, t_{st}])$ as the base case. According to equations (17), (18), (19) and (20), we have

$$\mathbf{opt}(\emptyset, \Delta^u[t_{st}, t_{st}]) = \phi(S[t_{st}] = (-1, -1), \emptyset) \text{ if } \psi^f(t_{st}) = \emptyset, \tag{26}$$

and

$$\mathbf{opt}(\{\chi_{i,k}(h)\}, \Delta^u[t_{st}, t_{st}]) = \phi(S[t_{st}] = (i, h), \{\chi_{i,k}(h)\}) \text{ if } \psi^f(t_{st}) = \{\chi_{i,k}(h) | \chi_{i,k}(h) \in \psi^r(t_{st})\}. \tag{27}$$

Hence, $\mathbf{opt}(\psi^f(t_{st}), \Delta^u[t_{st}, t_{st}])$ minimizes $S[t_{st}, t_{st}].\rho$.

Assume that $\mathbf{opt}(\psi^f(t-1), \Delta^u[t_{st}, t-1])$ minimizes $S[t_{st}, t-1].\rho$. We need to prove that $\mathbf{opt}(\psi^f(t), \Delta^u[t_{st}, t])$ minimizes $S[t_{st}, t].\rho$. According to the assumption, $\mathbf{opt}(\psi^f(t-1), \Delta^u[t_{st}, t])$ minimizes $S[t_{st}, t-1].\rho$, where $S[t_{st}, t-1]$ finishes $\psi^f(t-1) = \psi^f(t)$. Similarly, $\mathbf{opt}(\psi^f(t) - \{\chi^*_{i,k}(h)\}, \Delta^u[t_{st}, t] - S[t]|_{(i,h)}.\delta)$ minimizes $S'[t_{st}, t-1].\rho$, where $S'[t_{st}, t-1]$ finishes $\psi^f(t-1) = \psi^f(t) - \{\chi^*_{i,k}(h)\}$. According to equation (20), $\mathbf{opt}(\psi^f(t), \Delta^u[t_{st}, t])$ considers all possible assignments of slot t and therefore minimizes $S[t_{st}, t].\rho$ since $S[t_{st}, t-1].\rho$ and $S'[t_{st}, t-1].\rho$ are minimized. Hence, the found solution by eDP minimizes objective function (7). □

Algorithm 2 summarizes DP based algorithm. The inputs to DP are transmission set $\Psi(t^c_{sw})$, start time t_{st}, reusable time slot t^r_b, switch point candidate t^c_{sw}, static schedule $\bar{S}$, reusable schedule S^r_b, number of periodic tasks, n, the maximum allowed number of maintained schedules for each slot, β, and the maximum allowed number of updated slots γ. In eDP, β and γ are set to $+\infty$ and Δ^u, respectively. However, β is set to a positive integer value in mDP. The algorithm starts with initializing set of parent schedules, $\mathbf{S_p}$, to be $\{S^r_b\}$ (Line 2). The main loop of Algorithm DP (Lines 3-36) generates child schedules $S[t_{st}, t]$'s based on each parent schedule $S[t_{st}, t-1]$ for each slot t starting from $t^r_b + 1$ to t^c_{sw}. When generating a dynamic schedule, DP first determines the state of t, *i.e.*, if t is an urgent and favourite time slot of some ready transmissions according to Definitions 3 and 4, respectively (Lines 5-6). Based on the state of t, DP determines the assignment of t and gets a child schedule $S[t_{st}, t]$ (Lines 7-20). Every time a new child schedule $S[t_{st}, t]$ for slot t is generated, DP determines which child schedules should be reserved in set of child schedules, $\mathbf{S_c}$ (Lines 21-31). After DP has obtained $\mathbf{S_c}$, it only reserves the first β child schedules at most in $\mathbf{S_c}$ and put them to $\mathbf{S_p}$ in Line 35. Such a process is repeated for each time slot t from t^r_b to t^c_{sw}. After the main loop, DP returns $S^*[t_{st}, t^c_{sw}]$ dropping the fewest number of periodic packets among all schedules in $\mathbf{S_p}$ (Line 37).

Figure 9 shows our experiment setup where 36 nodes are deployed in a 6×6 square mesh grid. Both sensors and actuators can serve as relay nodes. The wireless network topology is a connected graph, *i.e.*, the gateway V_{35} can reach all nodes in the network. Each routing path from a sensor to the gateway or from the gateway to an actuator is predetermined.

Algorithm 2 DP($\Psi(t_{sw}^c)$, t_{st}, t_b^r, t_{sw}^c, $\bar{S}$, S_b^r, n, β, γ)

$\mathbf{S_p} = \{S_b^r\}$;
$\mathbf{S_c} = \emptyset$;
for ($t = t_b^r + 1$; $t \leq t_{sw}^c$; $t++$) **do**
  for ($\forall S[t_{st}, t-1] \in \mathbf{S_p}$) **do**
    Determine if t is an urgent time slot using Definition 3;
    Determine if t is a favourite time slot using Definition 4;
    for ($i = -1$; $i \leq n$; $i++$) **do**
      if t is an urgent time slot and $i \neq 0$ **then**
        continue; //t can only be assigned to a rhythmic transmission to satisfy Constraint 1 of Problem OLSBO$^+$
      end if
      if t is a favourite time slot of $\chi_{i,k}^*(h)$ and $i == -1$ **then**
        continue; //t cannot be set to idle to satisfy Constraint 4 of Problem OLSBO$^+$
      end if
      if $t == -1$ **then**
        $S[t] = (-1, -1)$;
      else
        Get a ready transmission of $\chi_{i,k}(h)$ using Definition 2;
        $S[t] = (i, h)$;
      end if
      $S[t_{st}, t] = S[t_{st}, t-1] \cup S[t]$;
      Calculate $S[t_{st}, t].\rho$ and $S[t_{st}, t].\delta$;
      if ($S[t_{st}, t].\rho > \gamma$) **then**
        Skip $S[t_{st}, t]$ to satisfy Constraint 3 of Problem OLSBO$^+$;
      end if
      if ($\exists S'[t_{st}, t] \in \mathbf{S_c}$ such that $S'[t_{st}, t] \equiv S[t_{st}, t]$ and $S[t_{st}, t].\delta \geq S'[t_{st}, t].\delta$ and $S[t_{st}, t].\rho \geq S'[t_{st}, t].\rho$) **then**
        Skip $S[t_{st}, t]$;
      else
        Insert $S[t_{st}, t]$ to $\mathbf{S_c}$ and maintain schedules in $\mathbf{S_c}$ by sorting them in the non-increasing order of num of dropped packets first and num of updated slots second;
      end if
      if ($\exists S'[t_{st}, t] \in \mathbf{S_c}$ such that $S'[t_{st}, t] \equiv S[t_{st}, t]$ and $S[t_{st}, t].\delta \leq S'[t_{st}, t].\delta$ and $S[t_{st}, t].\rho \leq S'[t_{st}, t].\rho$) **then**
        $\mathbf{S_c} = \mathbf{S_c} - \{S'[t_{st}, t]\}$;
      end if
    end for
  end for
  The first β child schedules in $\mathbf{S_c}$ are stored in $\mathbf{S_p}$;
end for
return $S^*[t_{st}, t_{sw}] \in \mathbf{S}$ where $S^*[t_{st}, t_{sw}].\rho \leq S[t_{st}, t_{sw}].\rho$ among all $S[t_{st}, t_{sw}]$'s in $\mathbf{S}$;

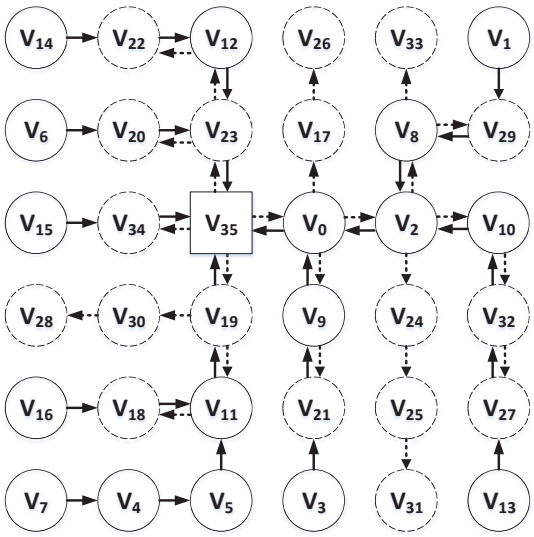

Fig. 9. Wireless network topology used in the simulation. It is composed of one gateway (V_{35}), 17 sensors ($V_0 - V_{16}$) and 18 actuators ($V_{17} - V_{34}$). A chain of solid or dashed direct links represents a routing path from a sensor to the gateway or from the gateway to an actuator, respectively.