Real-Time Communication in Low-Power Mobile Wireless Networks

Behnam Dezfouli, Marjan Radi, Octav Chipara
Department of Computer Science, University of Iowa, IA, USA
{behnam-dezfouli, marjan-radi, octav-chipara}@uiowa.edu

Abstract—Real-time wireless communication infrastructure is increasingly deployed to support industrial and cyber-physical applications. A limitation of existing real-time protocols is that they do not support mobility. This paper presents the development of a real-time network composed of a multi-hop infrastructure, and mobile nodes that associate with infrastructure nodes as they move. Once a mobile node joined the network, its real-time communication is guaranteed irrespective to the number and mobility pattern of mobile nodes. To develop this network, we propose Mobility-Aware Scheduling Algorithm (MASA), which benefits from new transmission scheduling approaches that cleverly combine potential packet transmissions to increase real-time capacity. We have developed a realistic trace-based simulator to evaluate the performance of MASA against two baseline algorithms. Experimental results indicate that MASA increases the number of admitted mobile nodes by 7x and 1.6x, and extends the network lifetime by 110% and 30%, compared to the baselines.

I. INTRODUCTION

Real-time wireless communication is increasingly adopted by process control industries to reduce the cost and ease the deployment of monitoring and control infrastructure. This transition is enabled by the adoption of WirelessHART [1] and ISA100 [2] industrial standards, which employ centralized network management and provide real-time communication in multi-hop low-power 802.15.4 networks. The research community has complemented these efforts by developing scheduling algorithms and real-time schedulability analysis [3]–[9] in the framework of these standards or by exploring alternative network architectures and protocols [10]–[12]. However, either the underlying scheduling algorithm is designed for immobile nodes [4]–[10], or real-time communication is supported for mobile nodes only one hop away from the destination [11], [12]. The lack of mobility significantly limits the applicability of these protocols to applications involving the mobility of patients, workers, or robots [13]–[15].

Real-time communication is typically supported through the construction of TDMA schedules [1]–[12]. However, supporting mobility in TDMA protocols is particularly challenging as schedules must be adapted with movements [16]. The problem is even harder when a centralized scheduler is used [1]–[12]: as a node moves through the network, it must establish new paths to the central station to route its data that, in turn, triggers network-wide rescheduling. The frequent reconstruction and distribution of schedules may consume significant bandwidth and energy. A related problem is that rescheduling may fail when the workload introduced by the nodes exceeds the network capacity. As a consequence, connections to one or more nodes may be dropped. This is unacceptable for a real-time mobile network. Therefore, real-time communication with mobile nodes requires bandwidth reservation over potential communication paths. However, when existing scheduling algorithms (e.g., [3], [4], [6]–[9], [17], [18]) are modified to satisfy this requirement, the constructed schedule achieves very low bandwidth utilization, and a reasonable number of mobile nodes cannot join the network. On the other hand, in contrast with the bandwidth reservation strategy of real-time standards, if mobility is supported through employing some sort of randomized channel access [19]–[23], both timeliness and reliability depend on the number and mobility pattern of mobile nodes, which is again unacceptable for a real-time mobile network.

This paper presents the development of a real-time mobile wireless network, and in particular, a new scheduling algorithm that supports real-time communication even in the presence of mobility. Our design is based on two key insights: (i) As our previous experiments indicate that mobility may lead to unstable routes [24], we organize the network into fixed infrastructure nodes and mobile nodes; mobile nodes dynamically associate with infrastructure nodes as they move. The benefit of imposing this structure is that it insulates the routing of packets over infrastructure nodes from mobility, effectively simplifying the scheduling process. (ii) To support real-time communication, mobile nodes must be dynamically associated with infrastructure nodes without requiring schedule re-computation or high signaling overhead. This requires bandwidth reservation through all the potential communication paths when a mobile node wants to join the network. We present a set of techniques to satisfy this requirement efficiently. The first technique minimizes the number of transmissions that should be scheduled for a mobile node’s data flow, through optimizing the release times of the transmissions on the paths from a mobile node towards the destination. The other three techniques allow a scheduling algorithm to combine the transmissions belonging to a data flow of a mobile node in a cell of the scheduling matrix. The proposed channel search algorithm (CSA) formulates some of these techniques and can be used for the development of real-time scheduling algorithms for mobile networks. We also present the mobility-aware scheduling algorithm (MASA) which benefits from the proposed techniques and provides a heuristic approach for fast and efficient scheduling of mobile nodes’ data flows. To the best of our knowledge, this is the first work that addresses real-time communication of mobile nodes over multiple hops. Additionally, our work is done within the context of WirelessHART, significantly broadening the standard’s applicability.

In order to ensure repeatability of the results under the same mobility pattern, we have developed a trace-driven simulator based on packet reception characteristics of 23 infrastructure nodes measured on various mobility paths within a floor of a building. The simulation setup is based on our insights about patient monitoring from our prior work [24]. We compare the performance of MASA against two baseline algorithms: (1) a laxity-based scheduling algorithm designed for stationary real-time networks, and (2) a mobility-enhanced version of (1).
The experimental results show that MASA results in 7x and 1.6x more number of admitted mobile nodes, 110% and 30% increase in lifetime, shorter algorithm execution duration and lower beaconing overhead, compared to the two baselines.

II. RELATED WORK

Protocols such as [19]–[23] have been proposed to realize the existence of low-power mobile wireless networking. Unfortunately, none of these solutions provides real-time communication due to the probabilistic (i.e., CSMA) nature of channel access. More importantly, they do not impose any admission mechanism for joining mobile nodes; therefore, both reliability and delay depend on the mobility pattern and number of mobile nodes. In contrast, this paper employs real-time scheduling and admission control for joining mobile nodes so that both reliability and timeliness are independent of the mobility pattern and number of mobile nodes.

Several channel access scheduling mechanisms have been recently proposed for enabling low-power real-time communication. In [6], [8] and [18], transmission schedules are determined based on the number of nodes and network topology, before network deployment. [3] and [9] mainly address the problem of scheduling when multiple paths exist between stationary source and destinations. The development of a real-time wireless network for a refinery with stationary nodes has been presented in [7]. After collecting and filtering packet reception traces, the employed scheduling algorithm computes all the feasible tree topologies and their schedules, assuming every node needs to send only one packet. A laxity-based heuristic scheduling algorithm has been proposed by [4] for reducing schedule computation delay in static WirelessHART networks. None of these scheduling algorithms is suitable for real-time mobile networks because they present very poor bandwidth utilization when mobility is introduced. We will discuss about these shortcomings in Section V.

MBStar [11] reduces transmission collisions in body sensor networks through offset-free scheduling. In contrast to our work, both MBStar and RT-WiFi [12] consider one-hop communication, which eliminates the need for dynamic association.

III. NETWORK COMPONENTS AND BASIC FLOWS

The network is composed of a collection of mobile nodes that rely on a set of infrastructure nodes organized in a multi-hop fashion to forward data towards a central node called Gateway (GW). The set of the infrastructure and mobile nodes are denoted as \( V = \{v_1, v_2, \ldots\} \) and \( M = \{m_1, m_2, \ldots\} \), respectively. Gateway is responsible for route and schedule determination.

Each flow \( f_i \) can be represented as \( f_i : (x_i, p_i, \phi_i, d_i) \), where \( x_i \) is the generator of flow \( f_i \), \( p_i \) is the flow generation period, \( \phi_i \in [0, p_i - 1] \) is the phase, and \( d_i \in [0, p_i] \) is the deadline. When a packet of a flow is released, the nodes on the path towards the destination should forward the packets of that flow based on the schedule computed by the GW. The forwarding of flow \( f_i \) from node \( x \) to next-hop node \( y \) is referred to as a transmission, denoted by \( (x, y, f_i) \). The scheduler determines the time slot and channel for each transmission. The release time of a transmission is the earliest time slot in which that transmission can be considered for scheduling. For example, for scheduling flow \( f_1 : (m_1, p_1, \phi_1, d_1) \) over path \( m_1 \rightarrow v_2 \rightarrow v_1 \), the release times of transmission \( (m_1, v_2, f_1) \) are \( \{\phi_1, p_1 + \phi_1, 2p_1 + \phi_1, \ldots\} \).

The basic flows required to implement a real-time wireless network are as follows:

1) Data Flows: Represents the data generated by mobile nodes. At a given time, a mobile node needs to be associated with an infrastructure node to be able to communicate with the GW. A routing graph, called upstream graph, determines paths from the infrastructure nodes towards the GW. The period, phase and deadline of a data flow are denoted by \( p_{data}, \phi_{data} \) and \( d_{data} \), respectively.

2) Control Flow: This is the control data sent by the GW to other nodes for various purposes. For example, control flow is used for distributing a newly computed schedule. Control data are routed using the downstream graph. The control flow is denoted by \( f_{ctr} : (GW, \phi_{ctr}, \phi_{ctr}, d_{ctr}) \).

3) Reporting Flows: Represents the reporting data generated by the infrastructure nodes. In addition to periodical health report, infrastructure nodes use reporting packets to convey mobile nodes’ join request and leave notice to the GW. We assume the same \( p_{rpt}, \phi_{rpt} \) and \( d_{rpt} \) for all the reporting flows.

IV. MOBILITY SUPPORT

A. Joining the Network

A mobile node joins the network in three phases:

1) Beaconing: Each infrastructure node periodically broadcasts a beacon packet to allow network discovery by the mobile nodes. For scheduling purposes, we model beacon transmission as a periodic flow with \( d_{beac} = p_{beac} \) and \( \phi_{beac} = 0 \).

2) Join Request: In order to join the network, a mobile node needs to send a join request packet, including information about the data flows the mobile node intends to transmit. This requires a slot in which the infrastructure nodes listen for join request packets. The scheduling of this slot is so that all the infrastructure nodes use the same channel and time slot within a period \( p_{req} \). Also, \( d_{req} = p_{req} \) because it is a one-hop flow (mobile node to infrastructure node).

3) Schedule Reception: When an infrastructure node receives a join request, it forwards that request through the next reporting data sent to the GW, and the GW reserves bandwidth for the data flows of the mobile node through rescheduling. Afterwards, the new schedule is distributed to the infrastructure nodes through the control flow. Additionally, to reduce packet overhead, each node only forwards the schedule related to the lower-level nodes in the routing graph. When an infrastructure node receives a schedule in response to a join request, it should include that schedule in its next beacon to be received by the mobile node.

B. Mobility and Association

After a mobile node \( m_1 \) joined the network, it may use different infrastructure nodes for association while moving. We denote this set by \( \mathbb{M}_1 \), and it is either explicitly declared by the mobile node, or it is determined by the GW. For example, in a multi-level hospital/factory, if a patient/robot is restricted to move within a particular level, \( \mathbb{M}_1 \) may vary based on the level in which the node is being used.

A very important feature of our proposed mobile network is that, when a mobile node \( m_1 \) is admitted to the network, the employed scheduling algorithm efficiently reserves bandwidth from every node in \( \mathbb{M}_1 \) towards the GW; therefore, the need for rescheduling upon each association is eliminated. Furthermore, when a mobile node finds a better infrastructure node for association (e.g., when it receives a beacon from an infrastructure node with higher link quality), it only changes the destination...
address of its subsequent packets; hence, association does not impose any signaling overhead.

C. Leaving the Network

A mobile node should send a leave notice before stopping communication with the GW. This allows the GW to release the bandwidth assigned to the mobile node. If the GW does not receive any packet from a mobile node for a specific number of its data flow’s period, the GW can produce an alert and then release the resources assigned to that node.

V. IMPLICATIONS OF MOBILITY ON SCHEDULING

To avoid intra-network interference and achieve high communication reliability, no more than one transmission should happen in a given time slot and a channel [1]. The lack of spatial reuse alongside with node mobility result in a considerable number of time slots and channels required to meet network bandwidth requirements. In this section, we show the effects of mobility on scheduling, and we present techniques for improving the scheduling efficiency of mobile nodes’ data flows. We assume that the upstream graph is a spanning tree, and the downstream graph is constructed through reversing the edges of the upstream graph.

A. Mobility Support with Existing Scheduling Algorithms

Through Figure 1 we show the shortcomings of the scheduling algorithms designed for stationary real-time wireless networks.

At time $t_1$, $m_3$ joins the network and associates with node $v_5$. Data transmission through $v_5$ requires the GW to successfully reserve bandwidth for the data flows generated by $m_1$ through path $m_1 \rightarrow v_5 \rightarrow v_1$. The computed schedule should be distributed to the infrastructure nodes as well as the mobile node. During $t_1$ to $t_6$, $m_1$ moves and needs to associate with different nodes. However, there is no guarantee that bandwidth reservation for $m_1$ through the new paths would be successful. For example, after bandwidth reservation over link $v_5 \rightarrow v_1$ for $m_2$, the GW may be unable to reserve bandwidth over this link for $m_1$. Even if we assume the feasibility of scheduling on association times, on-demand scheduling presents high delay and signaling overhead. As discussed in Section IV, request for rescheduling and schedule distribution require communication with the GW, resulting in high delay, packet transmission overhead and increased energy consumption.

These observations suggest that, in order to implement a real-time mobile wireless network, the GW should reserve bandwidth through all the potential infrastructure nodes that the mobile node may be associated with. For example, assuming $m_1 = \{v_1, v_2, v_3, v_4, v_5\}$, bandwidth should be reserved through: $m_1 \rightarrow v_1$, $m_1 \rightarrow v_2 \rightarrow v_1$, $m_1 \rightarrow v_3 \rightarrow v_2 \rightarrow v_1$, $m_1 \rightarrow v_4 \rightarrow v_2 \rightarrow v_1$, and $m_1 \rightarrow v_5 \rightarrow v_1$. Using existing algorithms (e.g., [3], [4], [6]–[9], [17], [18]), these paths are scheduled separately, which results in a low channel utilization and a very low number of mobile nodes admitted. The main inefficiency comes from the fact that a transmission $(x, y, f_1)$ cannot be scheduled in a slot if either $x$ or $y$ is involved in another schedule, as a sender or a receiver.

Consider data flow $f_1 : \langle m_1, p_1 = 16, o_1 = 0, d_1 = 12 \rangle$ generated by $m_1$ in Figure 1. Table I shows the schedule produced by a scheduling algorithm that reserves bandwidth over all the mentioned paths. $c_1$ and $c_2$ in Table I refer to two channel numbers. We assume that at each time slot, the priority of scheduling a transmission is determined based on the flow’s deadline and the distance of the packet from its destination. This scheduling strategy is referred to as laxity based or least-laxity first, and will be discussed in Section VI. We refer to this algorithm as the basic scheduling algorithm (BSA).

B. Efficient Scheduling of Mobile Nodes’ Data Flows

In Figure 1, BSA schedules $(v_2, v_1, f_1)$ three times, one for each path. For example, on path $m_1 \rightarrow v_2 \rightarrow v_1$, transmission $(v_2, v_1, f_1)$ is released and can be scheduled as soon as $(m_1, v_2, f_1)$ is scheduled. However, we should note that link $(v_2, v_1)$ needs to forward flow $f_1$ only once before its deadline (i.e., slot time 12), irrespective to the path through which $m_1$ communicates with the GW. Therefore, $(v_2, v_1, f_1)$ should be released after transmissions $(\ast, v_2, f_1)$ have been scheduled, where $\ast$ could be any node. Assume $\Upsilon(v_i) = \{v_1, \ldots, v_k\}$ represents the set of the infrastructure nodes that are children of node $v_i$.

Table I. Sample scheduling using BSA, ESA, and MASA.

<table>
<thead>
<tr>
<th>Slot</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>BSA</td>
<td>$(v_1, v_2, f_1)$</td>
<td>$(v_1, v_2, f_1)$</td>
<td>$(v_1, v_2, f_1)$</td>
<td>$(v_1, v_2, f_1)$</td>
<td>$(v_1, v_2, f_1)$</td>
<td>$(v_1, v_2, f_1)$</td>
</tr>
<tr>
<td>ESA</td>
<td>$(v_1, v_2, f_1)$</td>
<td>$(v_1, v_2, f_1)$</td>
<td>$(v_1, v_2, f_1)$</td>
<td>$(v_1, v_2, f_1)$</td>
<td>$(v_1, v_2, f_1)$</td>
<td>$(v_1, v_2, f_1)$</td>
</tr>
<tr>
<td>MASA</td>
<td>$(v_1, v_2, f_1)$</td>
<td>$(v_1, v_2, f_1)$</td>
<td>$(v_1, v_2, f_1)$</td>
<td>$(v_1, v_2, f_1)$</td>
<td>$(v_1, v_2, f_1)$</td>
<td>$(v_1, v_2, f_1)$</td>
</tr>
</tbody>
</table>

Technique 1. A transmission $(v_i, v_j, f_q)$ should be released after transmissions $(\{m_r, v_i, f_q\} \cup \{v_i, v_j, f_q\} | v_i \in \Upsilon(v_i))$ have been scheduled.

Using this technique in a scheduling algorithm, redundant transmissions are removed and transmissions are released appropriately. We refer to the enhanced version of BSA with Technique 1 as the enhanced scheduling algorithm (ESA).

Table I shows the scheduling improvement achieved with ESA.

Both BSA and ESA generate two-dimensional scheduling matrices, where each entry can be indexed by channel number $c_i$ and slot number $s_j$ as $M[c_i][s_j]$. However, in the rest of this section we show that multiple schedules can be combined within a cell of $M[]$. In terms of implementation, a cell $M[c_i][s_j]$ is an array/vector/linkedin-list, which can include more than one schedule. When Technique 1 and the following three techniques are used for scheduling, we refer to the algorithm as the mobility-aware scheduling algorithm (MASA).

The real implementation of MASA is given in Section VI.

Technique 2. Any subset of $(\{m_r, v_i, f_q\} \cup \{v_i, v_j, f_q\} | v_i \in \Upsilon(v_i))$ can be combined.

1Laxity-based scheduling has been widely used for task scheduling, where processing time is equivalent with number of hops [25].

2It should be noted that transmission combination is different from slot sharing. The former guarantees that only one transmission is activated in a time slot, the latter allows several nodes compete for channel access.
Proof: Two transmissions \((*, v_i, f_q)\) cannot be combined if they transmit concurrently. We prove that this never happens. During an interval \([k \times p_0 + \phi_q, (k+1) \times p_0 + \phi_q - 1]\), node \(m_r\) has only one packet belonging to flow \(f_q\). This packet is either sent to node \(v_i\) (directly), or is sent to one of the nodes in set \(T(v_i)\), directly or through multiple hops. Therefore, exactly one link \((*, v_i)\) should forward \(f_q\) during the given interval.

In Table 1 slot 1, the three transmissions \((m_1, v_2, f_1), (v_3, v_2, f_1)\) and \((v_4, v_2, f_1)\) are combined by MASA because they all forward the same flow. While Technique 2 proves the possibility of combining transmissions \((*, v_i, f_q)\), the following technique proves that transmissions \((m_r, *, f_q)\) can also be combined.

Technique 3. For a set \(\{(m_r, v_i, f_q) | v_i \in m_r\}\), which is the set of transmissions for flow \(f_q\) from a mobile node \(m_r\) to the potentially associatable infrastructure nodes, any subset of this set can be combined.

Proof: During an interval \([k \times p_0 + \phi_q, (k+1) \times p_0 + \phi_q - 1]\), where \(k \in \{0, 1, 2, ..., \}\), mobile node \(m_r\) has only one packet of flow \(f_q\) to forward. Therefore, it can activate only one link to an infrastructure node within this interval.

Referring to Table 1, this technique allowed MASA to combine \((m_1, v_3, f_1)\) and \((m_1, v_4, f_1)\) in slot 0, and \((m_1, v_2, f_1)\) and \((m_1, v_5, f_1)\) in slot 1.

The following technique proves a more general possibility of transmission combination.

Technique 4. When Technique 1 is applied and the upstream graph is a spanning tree, a released transmission \((w, z, f_q)\) can be combined with any scheduled transmission \((x, y, f_q)\).

Proof: Conflict happens when the two transmissions share common ends. We assume schedules are assigned in an incremental order of time slots \(s\). Technique 2 proves the possibility of combination if \(z = y, w = x\) cannot happen because we assume that the upstream graph is a spanning tree, \(y = w\) never happens because if \((x, y, f_q)\) is scheduled in this time slot, \((w, z, f_q)\) should be released in the next time slot. \(z = x\) never happens for a similar reason.

Having a released transmission \((w, z, f_q)\) and slot number \(s\), the Channel Search Algorithm (CSA) (Algorithm 1) employs Technique 2, 3 and 4 to find the best cell, if any, in column \(s\) of matrix \(M\) for scheduling this transmission. The procedure returns a channel number \(c_j\) \(\in [0, C-1]\) to index the chosen matrix cell. The algorithm first checks if either \(w\) or \(z\) are involved in a transmission of another flow in this slot; if so, the schedule cannot be made. Otherwise, the algorithm applies Technique 2, 3 and 4 in the following order: (i) Technique 2 (line 5), (ii) Technique 3 (line 6), and (iii) Technique 4 (line 7). If none of the techniques was applicable, the algorithm looks for an empty cell (line 8).

Note that CSA checks Technique 2 before Technique 3. The reason is that, it is more desirable to combine transmissions \((m_r, v_i, f_q)\) with transmissions \((*, v_i, f_q)\) because it reduces the number of slots in which \(v_i\) is involved in the transmission of flow \(f_q\). For example in Table 1, although \((m_1, v_2, f_1)\) could be scheduled in slot 0, combination with \((*, v_2, f_1)\) in slot 1 allows node \(v_2\) to be involved in other transmissions in slot 0.

It is worth noting that Technique 1 is not implemented in CSA because it should be implemented within an actual scheduling algorithm. The implementation will be presented in the next section.

VI. MOBILITY-AWARE SCHEDULING ALGORITHM

This section introduces the mobility-aware scheduling algorithm (MASA) (Algorithm 2). MASA uses Technique 1 for optimizing transmission release times, employs CSA (Algorithm 1) for schedule combination, and prioritizes released transmissions based on their laxities.

The algorithm maintains three sets of transmissions: (i) \(\Theta\): the transmissions that should be scheduled but have not been released yet; (ii) \(\Theta_{rel}\): the set of the released transmissions (i.e., ready to be scheduled); (iii) \(\Theta_{new, rel}\): the transmissions scheduled in the current iteration of the algorithm.

Starting from time slot 0, whenever a new time slot \(s\) is considered, function \(\text{updRelTrans()}\) is called to update the set of released transmissions (line 5 and 14). Given a time slot \(s\) and the set of flows, function \(\text{updRelTrans()}\) evaluates if a new flow should be released in this slot. If a flow \(f_q\) is generated by a node \(m_r\) in this time slot (line 21), new transmissions on the links from the mobile node to the infrastructure nodes (in \(m_r\)) are added to \(\Theta_{rel}\) (line 24). Furthermore, transmissions on the path from the infrastructure nodes to the root node are added to \(\Theta\) (line 25). For example, considering node \(m_1\) in Figure 1, after the first call to \(\text{updRelTrans()}\): \(\Theta_{rel} = \{ (m_1, v_1, f_1), (m_1, v_2, f_1), (m_1, v_3, f_1), (m_1, v_4, f_1), (m_1, v_5, f_1) \} \) and \(\Theta = \{ (v_5, v_1, f_1), (v_2, v_1, f_1), (v_3, v_2, f_1), (v_4, v_2, f_1) \}\). Function \(\text{updRelTrans()}\) also implements Technique 1 and evaluates the possibility of adding new transmissions to \(\Theta_{rel}\) through considering the transmissions scheduled in the current time slot (line 28-33).

At each time slot, the algorithm checks the schedulability of every transmission in set \(\Theta_{rel}\). Additionally, amongst the released transmissions, we give higher priority to the transmission that is most urgent to be scheduled with respect to its deadline and remaining number of hops to the destination. To this aim, we employ transmission laxity. The laxity of a transmission in a given time slot is defined as the remaining number of time slots until flow deadline minus the number of hops to the destination. In fact, laxity reflects the maximum number of slots a transmission can be postponed. The computation of laxity is performed by function \(\text{laxity()}\) in Algorithm 2. \(h_x\) in \(\text{laxity()}\) is the number of hops from node \(x\) to the
of the flow corresponding to that transmission. The scheduling algorithm will not be able to meet the deadline computing the laxity of the released transmissions that have not been scheduled. Therefore, the safe place for switching to a new schedule is at the end of the hyper-period. The shortcoming of this solution is long join delay when the data flow of the mobile node being joined is significantly shorter than the hyper-period. To solve this problem, the GW should compute the time at which the new mobile node could generate a packet after all the nodes have received the new schedule. If the interval between this time and the start of the next hyper-period is long, the GW computes a temporary schedule before switching to the new schedule. With respect to Algorithm 2, this is achieved through including the flows of the new mobile node starting from time slot $s_k + 1$

### Algorithm 2: Mobility-Aware Scheduling Algorithm (MASA)

**Input:** $F$: set of the flows that should be scheduled  
**Output:** generates scheduling matrix $\mathcal{M}[C][T]$ if the scheduling was successful, otherwise returns "unsuccesful"

1. **begin**
2. $T \leftarrow$ least common multiplier of flows’ periods;
3. $\Theta \leftarrow \emptyset$; $\Theta_{rel} \leftarrow \emptyset$; $\Theta_{new-sch} \leftarrow \emptyset$;
4. $s \leftarrow 0$;
5. updRelTrans$(s, F, \Theta, \Theta_{rel}, \Theta_{new-sch})$;
6. Sort $\Theta_{rel}$ in ascending order of laxities;
7. **while** $\Theta_{rel} \neq \emptyset$ **do**
8. **for** index $i \leftarrow 1$ to $\Theta_{rel}$ **do**
9. $(w, z, f_i) \leftarrow$ the first transmission in set $\Theta_{rel}$;
10. $c_j = \text{CSA}(w, z, f_i, s, \mathcal{M}[C][T])$;
11. **if** $c_j \neq -1$ **then**
12. $s \leftarrow (s + 1) \mod T$;
13. updRelTrans$(s, F, \Theta, \Theta_{rel}, \Theta_{new-sch})$;
14. **end if**
15. **for every transmission** $(x, y, f_i)$ in $\Theta_{rel}$ **do**
16. **if** $\exists x f y (x, y, f_i) < 0$ **then** return unsuccessful ;
17. **end if**
18. **end for**
19. **end while**
20. **Procedure** updRelTrans$(s, F, \Theta, \Theta_{rel}, \Theta_{new-sch})$
21. **for every flow** $f_j$ in $F$ **do**
22. **if** $s \mod p_j = \phi_0$ **then**
23. $m_w \leftarrow$ the mobile node generating flow $f_j$;
24. **for every** $v_j$ in $m_w$ **do**
25. $\Theta_{rel} \leftarrow \{(m_v, v_f, f_j)\}$;
26. **end for**
27. **for every transmission** $(v_f, v_f, f_j)$ in $\Theta$ **do**
28. **end for**
29. **end if**
30. **for every link** $(w, z, f_i)$ in $\Theta_{new-sch}$ **do**
31. **if** $(\text{transmission}) (z, y, f_i) \exists$ **in** $\Theta$ **and**
32. $(\text{no transmission}) (z, y, f_i) \exists$ **in** $\Theta \cup \Theta_{rel}$ **then**
33. $\Theta_{rel} \leftarrow \{(z, y, f_i)\}$;
34. **remove** $(z, y, f_i)$ from $\Theta$;
35. **endif**
36. **return**
37. **Procedure** addSchedule$(w, z, f_i, s, c_j, \Theta_{rel})$
38. $\mathcal{M}[c_j][s] \leftarrow (w, z, f_i)$;
39. **remove** $(w, z, f_i)$ from $\Theta_{rel}$;
40. **return**
41. **Procedure** laxity$(x, y, f_i)$
42. $s' = x \mod p_i$;
43. **if** $s' \geq \phi_0$ **then** return $d_i + \phi_i - s' - h_x$ ;
44. **else** return $d_i + \phi_i - (s' + p_i) - h_x$ ;
45. **end if**

Before evaluating the schedulability of the released transmissions in a time slot, $\Theta_{rel}$ is sorted based on transmission laxities (line 6,15), then the transmissions are evaluated from the beginning of this list (line 8). Using CSA (Algorithm 1), MASA finds the most suitable matrix cell, if any, for scheduling a transmission in the current time slot (line 10). After considering all the transmissions in $\Theta_{rel}$, the scheduling algorithm evaluates the feasibility of scheduling before proceeding to the next time slot. This is evaluated through computing the laxity of the released transmissions that have not been scheduled (line 16). If a transmission’s laxity is negative, the scheduling algorithm will not be able to meet the deadline of the flow corresponding to that transmission.

### A. Results and Discussions

1) **Scalability:** The number of admitted mobile nodes reflects the efficiency of bandwidth reservation. Figure 3(a)
shows that MASA results in up to 7x and 1.6x more number of admitted nodes compared to BSA and ESA, respectively. Although ESA provides higher bandwidth reservation efficiency compared to BSA (2.6x) through minimizing the number of transmissions that should be scheduled, MASA can provide even higher bandwidth utilization through schedule combination. This can also be observed from Figure 3(b), where the GW’s packet exchange rate corresponding to the maximum number of admitted mobile nodes is demonstrated. Figure 4 shows bandwidth reservation efficiency from two other perspectives. For example, while the maximum number of admitted mobile nodes with ESA is about 40% less than MASA, Figure 4(a) shows that ESA presents almost the same maximum delay as MASA, which indicates using the same number of time slots for scheduling a lower number of data flows. We have also measured the beaconing overhead as the number of scheduling matrix entries used for beaconing over the total number of entries. Figure 4(b) shows that MASA reduces this overhead by more than 90% and 60% compared to BSA and ESA, respectively.

2) Lifetime: Figure 5 presents the node lifetime achieved with various scheduling algorithms versus the number of supported mobile nodes. Staring with 5 mobile nodes, we increase the number of mobile nodes in steps of 5 and measure the steady-state energy consumption. For a given $p_{data}$ and a number of mobile nodes, MASA provides about 110% and 30% improvement in lifetime compared to BSA and ESA, respectively. Although with a given $p_{data}$ and a number of mobile nodes the time spent in transmit mode is the same for the three algorithms, the difference in lifetimes is due to the different durations the radio spends in receive mode. From the MAC point of view, the techniques introduced in Section V reduce the number of time slots in which a node expects to receive a packet, therefore shortening the energy consumed in idle listening mode. It should be noted that Figure 5(a) does not include BSA because the maximum number of mobile nodes admitted with this algorithm for $p_{data} = 128$ is less than 5.

3) Algorithm Execution Duration: Figure 6 shows the execution duration of MASA, BSA and ESA using i7-4980HQ processor. For a given number of mobile nodes scheduled, MASA provides the lowest execution duration. Referring to Algorithm 2, in each time slot the algorithm needs to check the schedulability of every transmission in $T_{rel}$, compute trans-

### Table II. General Performance Evaluation Parameters

<table>
<thead>
<tr>
<th>$p_{data}$</th>
<th>$d_{data} = d_{req} = p_{rx} = p_{tx} = d_{av} = p_{req} = 512$ slots</th>
</tr>
</thead>
<tbody>
<tr>
<td>$p_{data} \in [64, 128, 256, 512]$</td>
<td>$d_{data} \in [0, P_{data} - 1]$</td>
</tr>
</tbody>
</table>

**Notes:**
- Packets: 802.15.4 compatible - maximum 127 bytes
- Time Slot Duration = 10ms (as defined by WirelessHART standard)
- Battery: 2500mAh 3V | Radio Transmission Power = 0dBm

**Fig. 3.** (a): Maximum number of admitted mobile nodes, (b): GW’s throughput, achieved with various scheduling algorithms.

**Fig. 4.** (a): Packet delivery delay, (b): beaconing overhead, achieved with various scheduling algorithms.

**Fig. 5.** Average node lifetime achieved with various scheduling algorithms.
VIII. CONCLUSION

In this paper we presented the development of a real-time and low-power mobile wireless network. The developed network is composed of a multi-hop infrastructure and mobile nodes, where mobile nodes associate with various infrastructure nodes as they move to exchange data with the Gateway. We showed that fast and energy-efficient association requires bandwidth reservation over potential communication paths whenever a new node joins the network; however, bandwidth reservation over multiple paths cannot be efficiently achieved with existing scheduling algorithms. Therefore, we introduced techniques for improving the efficiency of scheduling mobile node’s data flows. This paper also proposed a practical scheduling algorithm, called MASA, which benefits from the proposed techniques, and results in a higher number of mobile nodes admitted to the network, longer network lifetime and lower beaconing overhead, compared to the baseline algorithms.

IX. ACKNOWLEDGMENT

This work was supported by the National Science Foundation (Grant No. 1144664) and by the Roy J. Carver Charitable Trust (Grant No. 14-4355).

REFERENCES