MobiBone: An energy-efficient and adaptive network protocol to support short rendezvous between static and mobile wireless sensor nodes

Kui Zhang, Eyuel D. Ayele, Nirvana Meratnia, Paul J.M Havinga, Peng Guo, Youxin Wu
Pervasive Systems Group, University of Twente, Enschede, The Netherlands
Department of Electronic and Engineering, Huazhong University of Science and Technology
Department of Electronic and Engineering, Nanchang University
Email: {K.Zhang, e.d.ayele, n.meratnia, p.j.m.havinga}@utwente.nl,
guopeng@hust.edu.cn, wuyouxin@ncu.edu.cu

***Abstract*—To ensure long network life-time, the duty-cycle of wireless sensor networks is often set to be low. This brings with itself the risk of either missing a sent packet or delaying the message delivery and dissemination depending on the duration of the duty-cycle and number of hops. This risk is increased in wireless sensor applications with hybrid architecture, in which a static ground wireless sensor network interacts with a network of mobile sensor nodes. Dynamicity and mobility of mobile nodes may lead to only a short rendezvous between them and the backbone network to exchange data. Additionally, such dynamicity generates complex and often random data traffic patterns. To support successful data delivery in case of short rendezvous between static and mobile wireless sensor nodes, we propose MobiBone, an energy-efficient and adaptive network protocol that utilizes data packet traffic to characterize the sleep schedule. Our simulation results show that compared with network protocols with fixed duty-cycles, MobiBone offers a good trade-off between energy consumption, latency, and detection rate of mobile nodes (which indicates awakens of the backbone network at crucial times of mobile node presence).**

***Index Terms*—Hybrid wireless sensor networks, Pipeline scheduling, data traffic learning, adaptive duty-cycle**

I. Introduction

Some wireless sensor network applications have a hybrid architecture, in which a static ground wireless sensor network, which forms the backbone network, interacts with a network of mobile sensor nodes [1, 2]. Wireless animal monitoring applications are examples of this type of networks. The mobile senor nodes (i.e., tagged animals) may or may not form a network among themselves but need to communicate with the backbone network to deliver their data fast. The reason that this data delivery needs to be fast is that the dynamicity and mobility of mobile nodes may lead to only a short rendezvous with the backbone network, during which their data should be transferred to the backbone network to be reported back to the sink node. The sink node is a backbone node often equipped with an Internet connection for example using GPRS. This scenario is shown in Fig. 1.

In addition to the data transfer, the backbone network is often responsible for detection and tracking of mobile nodes (i.e., animal movement). The data traffic of the backbone network is therefore random, as it depends on the presence and movement of the mobile nodes, which by itself is quite random.

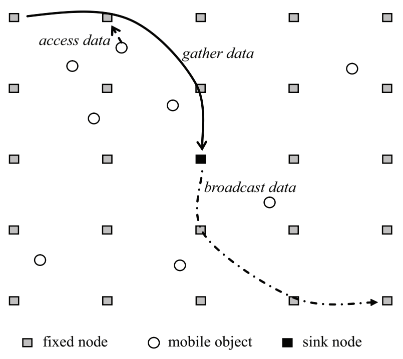


Fig. 1: An example scenario showing the interaction between mobile nodes and a static ground backbone network

To ensure long life-time of the backbone network, the backbone network has often a low duty-cycle during the sender node has to wait for a certain period of time until its destination node becomes active. Depending on the duration of the duty-cycle and number of hops, different packet delivery latency may be experienced.

Generally speaking, duty-cycle based data gathering in wireless sensor networks is straight forward. Each node only needs to set its active time in the duty-cycle according to the active time of its children nodes. Use of pipeline sleep scheduling methods such as staggered active-time scheduling [3], ladder wake-up [4], and streamline wake-up [5] scheduling to set this active time can result in minimizing packet delivery delay. However, most of these techniques only consider static ground nodes and assume predetermined data forwarding paths. This makes them only applicable when all communication links are fixed and reliable. To enhance the pipeline scheduling in unreliable communication environments, authors of [4] proposed the multi-pipeline scheduling algorithm, which employs multiple parent nodes in each layer on the data delivery path to establish multiple pipeline schedules. Utilisation of pipeline scheduling algorithms for gathering random data is also not straight forward. This is because the static ground nodes not only need to relay data of their children nodes to their own parent during their active time but also to specially set their active time to access random data coming from the nearby mobile nodes. Moreover, static ground nodes may need to set active time to forward some command messages from the sink to both static ground nodes as well as mobile nodes to configure their sleep schedule. Hence, in one duty cycle, each static ground node may have three types of active times, i.e., (i) up-link time to relay data to the sink, (ii) access time to access data from the nearby mobile nodes, and (iii) down-link time to forward data received from the sink. The data traffic during these three active times are either up-link (to relay data to the sink and access data from the nearby mobile nodes) or down-link (to forward data received from the sink).

To minimize the risk of missing packets or delaying their delivery in low duty-cycle networks, in which mobile nodes have short rendezvous with static ground nodes, we propose an energy-efficient and adaptive network protocol called MobiBone, which works based on pipeline scheduling concept. To the best of our knowledge, MobiBone is the first adaptive pipeline scheduling protocol which utilizes data packet traffic to characterize the sleep schedule. We propose a greedy algorithm to assign collision-free time slots to all static ground nodes of the backbone network. A frame-based aloha algorithm is used for mobile nodes to randomly access the static ground nodes' channel.

The rest of the paper is organized as follows. In Section III, we present the MobiBone protocol design. Performance evaluation results are presented in Section IV, while Section V concludes the paper.

II. Related Work

In WSN several network protocols have been proposed to achieve energy efficiency in hybrid networks. These algorithms are mainly designed based on network requirements, to minimize the packet delivery delay particularly in data gathering applications. In this section we present the state of the art works related to adaptive scheduling schemes.

As classical slot reservation algorithms tend to be complex and not very flexible, researchers have investigated simpler schemes which, at the same time, aim at achieving a good energy efficiency. For example, a low complexity slot selection mechanism is adopted in [6], where a lightweight medium access protocol (LMAC) is proposed. The main goal of LMAC is to reduce the radio state transitions and the protocol overhead. To this end, data are not acknowledged and the actual slot assignment is based on a binary mask of occupied slot and a random selection among free ones. The main drawback of LMAC is the fixed length of the frame, which has to be specified prior to deployment, and may be problematic. To this end, in [7] an Adaptive Information-centric LMAC (AILMAC) is proposed, so that the slot assignment can be more tailored to the actual traffic needs.

The mobile cluster MAC (MCMAC) [8] is a schedule-based MAC protocol which extends LMAC [6] to support cluster mobility. Unlike most of the proposed mobility-aware MAC protocol, MCMAC is optimized for those nodes which travel in group. MCMAC categorizes the sensor nodes into a static network and a mobile cluster. The protocol defines a Reference Point Group Mobility (RPGM) model and a Random Waypoint Mobility (RWM) model to mimic the movement characteristics of mobile clusters and the individual node movement within cluster. A frame in MCMAC is divided into an active and a sleep period. Since the slot assignment method is different for static and mobile nodes, the active period is further divided into static active slots (SAS) and mobile cluster slots (MCS). Static nodes communicate with each other in the SAS part by dynamically occupying a unique transmission slot in its two hop neighborhood. A static node can only transmit data in the specific slot it chooses and receives data in the remaining part of SAS. A guard time is inserted at the start and the end of every transmission slot to compute a nodes phase difference with its direct neighbors for synchronization

The member and topology changes of a virtual cluster caused by the inter-cluster mobility leads to the disconnection of the mobile node from the network. To expedite the connection set up process, MS-MAC enables each node to discover the presence as well as the level of mobility within its neighborhood, based on the RSSI values obtained from the SYNC messages transmitted by its neighbors. In [8], an enhanc ed MS-MAC protocol named EMS-MAC was introduced to predict the nodes movement more accurately by using both received signal strength indication (RSSI) and link quality indication (LQI), since MS-MAC causes energy wastage because of the inappropriate mobility prediction produced by using only RSSI. However, EMS-MAC has the same periodic procedure for connection setup as MS-MAC. In existing variations of contention-based MAC to support mobility, the fixed period of neighbor discovery and schedule updating does not allow the mobile node with its varying speed to connect to new neighbors in a sufficiently short time. In the contention-based MAC protocols mentioned above, a mobile node synchronizes with its neighbors after hearing a SYNC frame.

In addition pipeline scheduling scheme is one of the techniques proposed to increase adaptability in WSN network protocols. For instance DMAC [4], MERLIN [9], PRI-MAC [10], are designed with duty cycling feature together with a routing layer protocol to collect data to the network sink. But these scheduling algorithms lack the synchronization scheme needed to meet network requirements. For instance, DMAC developed a tree data gathering mechanism but didn't provide any details about the adaptive staggering wake-up time schedules between sensor nodes. The same is true for MERLIN and PRI-MAC on their lack synchronization for hybrid networks. Cao et al. [11] presents a Robust Multi-pipeline Scheduling (RMS) scheme to gather data in dense duty-cycled WSN. In RMS more than one pipeline is established at a time and they collaborate to help data packets choose the more reliable pipeline link when a failure happens. This could minimize the packet loss and decrease the data latency as well. However, RMS algorithm didn't not include a way to synchronize the sensor nodes, this creates a phase shift and collision, when sensor nodes try to access the channel and send data packets.

Our network protocol, however, is able to adapt to changes

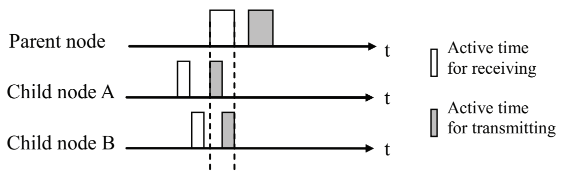


Fig. 2: Active time of nodes

in the hybrid network topology, to verify this we evaluated its performance with mobile and fixed nodes movement scenarios as will be discussed in Section IV.

III. MobiBone Protocol Design

A. Design considerations

To make the pipeline-based scheduling suitable for the gathering of random access data, the following design issues need to be considered:

- Link unreliability while setting the optimal gap between adjacent nodes on the data delivery path to achieve the minimum expected delay.
- Wireless interference to appropriately set for each node the time when its active time starts in each duty cycle so that the collision problem can be avoided.
- Parents' active time to receive multiple children' data during active time of the static ground nodes. This means that the parent active time should be large enough to receive multiple children data, as shown in Figure 2.
- Double pipeline based scheduling for the static ground nodes to relay data to the sink node and to forward data from the sink node in a minimum end-to-end delivery delay.
- Dynamicity of mobile nodes to ensure successful data transmission between mobile and static ground nodes during active time of the backbone network.

We expect that the static ground nodes' up-link time and down-link time follow a double pipeline scheduling scheme to achieve a minimum end-to-end packet delivering delay. Each static ground node's up-link time and down-link time will be set according to the node's hop count to the sink. To avoid any possible collision when one frequency channel is used, static ground nodes within the same layer will be assigned different time slots. Additionally, to be able to receive data from the mobile nodes, active time of static ground nodes should be the same as the active time of the mobile nodes. Setting this active time should be done in a fast and energy-efficient manner, since mobile nodes have limited power source and may move fast.

B. Protocol concept

We assume that all static ground nodes are using duty cycle scheduling and the length of the duty cycle is denoted by L, which is the same for all the static ground nodes. In addition, a shortest path route is established in the network. Both up-link traffic and down-link traffic go along the shortest path to or from the sink. Each static ground node is synchronized with its parent node on the shortest path. Furthermore, each static ground node has a time slot ID assigned according to the node's hop-distance to the sink. According to this ID, the static ground node determines its down-link time and up-link time. All static ground nodes set their access time to $mL + \Delta$, where $m = 1, 2, ...$ and $0 < \Delta < L$. The length of the down-link time is short and constant, while the length of the up-link and access times are dynamic and will be set according to the size of up-link traffic and local access traffic, respectively. This allows the size of the access time and the up-link time to be adaptive. Figure 3 illustrates more details on the three types of active times, by taking an example of several static ground nodes on the same data delivery path. Both the down-link time and the up-link time of the static ground nodes are lined up along the data delivery direction, while their access time is the same.

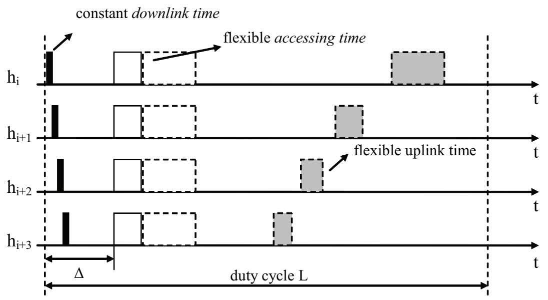


Fig. 3: Sleep scheduling scheme for gathering random data

We set the start of the access time to $mL + \Delta$ and the start of the down-link time to $mL + ID * S_d$, where S_d is size of the down-link time. For simplicity, we set S_d as the value of the optimal pipeline gap of the worst down-link in the network. We also set the start of the up-link time to $(m + 1)L - ID * S_u(m)$, where $S_u(m)$ is the size of up-link time in the mth duty cycle and can be adjusted by the command sent from the sink node.
Each mobile node wakes up during the down-link time of all static ground nodes. Thus, the mobile node can receive the configuration command and the nearby static ground nodes' ID that are broad-casted in the down-link time by the sink. In this way, mobile nodes can get the current size of access time of their nearby static ground nodes. When the access time of a nearby static ground node starts, the mobile node becomes active and will randomly access the channel during the access time. The MobiBone protocol consists of three main steps: (1) collision-free time slot ID assignment, (2) flexible access time setting, and (3) flexible up-link time setting. In what follows, each of the steps are described.
1) Collision-free time slot ID assignment: To avoid collision during the down-link and up-link time, we propose a greedy algorithm, i.e., Algorithm 1, to assign collision-free time slots to all static ground nodes.

According to Algorithm 1, we first assign slot ID 0 to the sink node n_0. After that, the algorithm checks each node n_j in each layer (i.e., same hop counts to the sink node) to judge whether it has unassigned neighbors in common with any other node n_t in current layer that has the same slot ID. If there is no such a node, then node n_j can be assigned with the same

Algorithm 1 Collision-free time slot assignment algorithm

```
1: Input: G=(V,E) and $n_0$
2: Construct BFS tree, and assign slot ID $k \leftarrow 0$ to $n_0$
3: k++
4: C ← 0, where C is the set of nodes having been assigned time slot.
5: Divide $V - n_0$ into a group of sets $L_1, L_2, ..., L_H$, where $L_i$ is the set of nodes with $i$ hop distance to $n_0$
6: for $i \leftarrow 1$ to $H$ do
7:     $C_i \leftarrow 0$
8:     for Each node $n_j \in L_i$ do
9:         $C_i \leftarrow C_i \cup n_j$
10:        if $\exists n_t \in C_i$ and $N(n_t) \cap N(n_j) \cap (V - C) = \Phi$, where $N(n_t)$ is the set of node $n_t^{\prime}s$ neighbors then
11:            Assign slot ID $k$ to $n_j$
12:        else
13:            Assign slot ID $k++$ to $n_j$
14:        end if
15:    end for
16:    $i++$
17:    $C \leftarrow C \cup C_i$
18: end for
```

slot ID of n_t. Otherwise, a new slot ID will be assigned to node n_j. Nodes in layer L_H do not need to be assigned IDs, as they just take the up-link time of their parents as their time to send the data and down-link time of their parents as their time to receive data.

2) Flexible access time setting: Due to the fact that density of mobile nodes near static ground nodes may be very dynamic, the access time may highly vary. To adjust the access time of the static ground nodes based on density of mobile nodes, we propose a frame-based aloha algorithm, i.e., Algorithm 2, for the mobile nopdes to randomly access the static ground nodes' channel.

According to Algorithm 2, the static ground nodes will extend their access time when they do not complete accessing the mobile nodes' data within their accessing time. However, if the accumulated accessing time exceeds the upper bound $L - ID_{max} * S_u(m) - \Delta$, the access time will not be extended anymore as the static ground node with the maximum slot ID starts its up-link time now. In the next duty cycle, the updated access time will be shortened with β parameter to be adaptive to be dynamic accessing data.

3) Flexible up-link time setting: As the up-link traffic on the shortest paths may vary due to the movement of mobile nodes, it is needed to adjust the size of the up-link time according to the up-link traffic.

To this end, as it can be seen from Figure 4, we propose a flexible pipeline scheduling scheme that consists of three components, i.e., (i) traffic learning, (ii) flexible time slot setting, and (iii) flexible duty cycle setting. Since mobile nodes move around, the traffic on a data delivery path to the sink may highly vary. High traffic requires more up-link time. However, the rate of the traffic variation could be high. This introduces a challenge for accurate adaptation as shown in Figure 5. A constant adaptation of up-link time cannot accurately follow

Algorithm 2 Flexible accessing time setting algorithm

```
1: Static ground node broadcasts $S_a = S_{a0}$ in its down-link time, where $S_a$ is the initial size of accessing time in current duty cycle.
2: $counter_A = 0$.
3: Static ground node checks the channel activities within its accessing time $S_a$, and records the size of time when the channel is busy as $S_{aB}$, records the size of time when succeeding to receive accessing data is $S_{aC}$, till accessing time ends.
4: if $S_{aB} - S_{aC} = 0$ then
5:     Go to step 14.
6: end if
7: if $\Delta + counter_A + S_a - S_{aC} + \alpha(S_{aB} - S_{aC}) > L - ID_{max} * S_u(m), (\alpha > 1)$ then
8:     Broadcasts indication that the access time ends.
9:     Go to step 15.
10: else
11:    Broadcast the extension of accessing time for $S_a = S_a - S_{aC} + \alpha(S_{aB} - S_{aC})$
12:    $counter_A = counter_A + S_a$
13:    Go to step 3.
14: end if
15: $S_a = S_{a0} + \beta * counter_A, (0 < \beta < 1)$
16: Break in current duty cycle
```

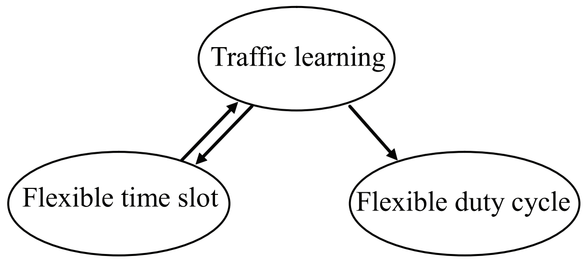


Fig. 4: Flexible pipeline scheduling scheme

changing data traffic. To deal with this situation, we employ traffic learning on the static ground nodes. Although being random in terms of when they occur, mobile nodes (in this case animals) have repetitive movement patterns when they move. Therefore, it is possible to predict the data traffic on the data delivery path at different times. For example, the animals being monitored may regularly walk across some routes to look for water/food or go back home, which may lead to high but predictable change of traffic on some paths to the sink. This predictable change helps set appropriate adaptations on the size of the up-link time.

For example, when there is going to be high change of up-link traffic, the sink will select larger step of adaptation for the size of the up-link time for the static ground nodes. In our scheme, the static ground nodes count the up-link traffic continuously and calculate the statistics of up-link traffic at different times. After some days, the system can have reliable network traffic statistics.

This traffic learning mechanism reduces the gap between the optimal up-link time required by real traffic and the actual up-

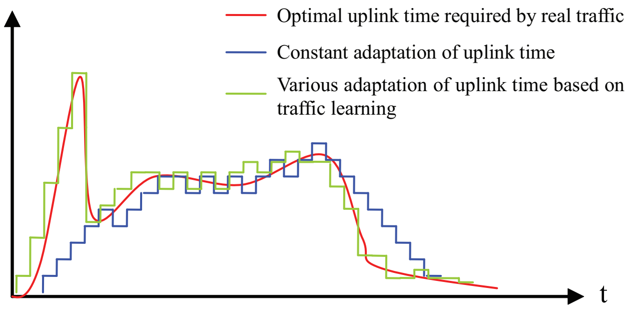


Fig. 5: Highly dynamic up-link traffic and adaptation of up-link time

link time set by the static ground nodes. However, as the gap cannot be reduced to zero, it is still needed to further adjust the up-link time according to the real traffic. If a parent static ground node finds that the ratio of channel busy time within its up-link time is larger than a given threshold Γ_u, it will request the sink to extend its up-link time to $S_u(m+1) = S_u(m) + \partial$. If the parent static ground node finds that the ratio of channel busy time within its up-link time is smaller than threshold Γ_l, it will request the sink to shorten its up-link time to $S_u(m+1) = S_u(m) - \partial$. Figure 6 illustrates this adjustment. The static ground parent node adjusts its up-link time only after the sink broadcasts the command in the down-link time in the next duty cycle.

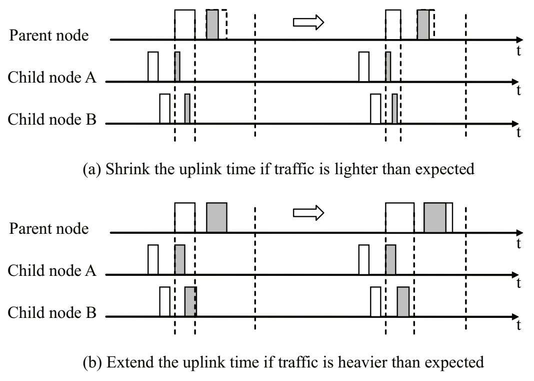

Parent node
Child node A
Child node B
t
(a) Shrink the uplink time if traffic is lighter than expected
Parent node
Child node A
Child node B
t
(b) Extend the uplink time if traffic is heavier than expected

Fig. 6: Adjustment of the uplink time

When information about $S_u(m,i), 1 \leq i \leq H$ is broadcast in the down-link time, each static ground node can find the start of its up-link time and the up-link time of its parent using $f[ID, S_u(m,i)]$. These start times can be computed as follows:

$$f[ID, S_u(m,i)] = \sum_{j=1}^{i} K_j * S_u(m,j) \tag{1}$$

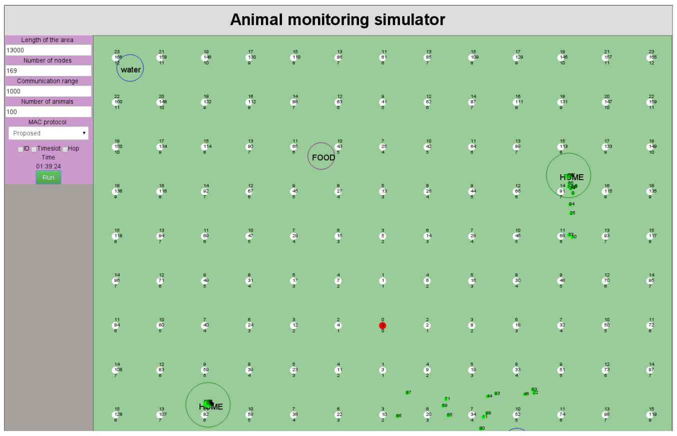
where K_j is the number of IDs with which the static ground nodes are in layer j.

It should be noted that when a parent node receives data from all its children early in its up-link time, it can go to sleep in advance. This results in saving energy, which is one of the advantages offered by our adaptive scheduling. In addition, if based on traffic learning algorithm, a static ground node finds out that there is no traffic in the coming duty cycle, it does not need to wake up. This also results in further energy saving. However, wrong prediction of the traffic learning algorithm will suspend the up-link traffic and bring unexpected traffic load in the next duty cycle of the static ground nodes. This makes the up-link time in the next duty cycle insufficient. Whenever the up-link time is insufficient for the up-link traffic, part of the packets will have to be discarded, which influences the data quality.

IV. Performance evaluation

A. Simulation setup

For the simulation, we deploy 169 static ground nodes in a grid over an area of $12000m * 12000m$, as shown in Figure 7. This means that there are 13 layers (similar hop counts from all static ground nodes to the sink) in the backbone network. The size of each grid cell is $1000m * 1000m$. Transmission range of static ground nodes is $1000m$ and their radio works at $433MHz$ frequency. 200 mobile nodes were also simulated. Communication range between the mobile and static ground nodes is set to $100m$. The minimum access time of each static ground node is set to $L \times 1\%$. If the number of mobile nodes which communicate with a static ground node is less than C_{min} (in the simulation we set to 15), then the periodic length of the static ground node's duty cycle is doubled. If the number of mobile nodes which communicate with a static ground node is more than C_{max} (in the simulation we set to 25), then the periodic length of the static ground node's duty cycle is halved. Simulation is repeated for 6000 runs.


Animal monitoring simulator

Length of the area: 13000
Number of nodes: 169
Communication range: 1000
Number of animals: 100
MAC protocol: Proposed
ID Timeslot Hop
Time 01:39:24

Fig. 7: Simulation setup: The mobile nodes mimic animal herding/clustering behaviour. They move to predefined locations such as home, forage area and water. Simulation parameters such as area, number of nodes, and communication range can be adjusted.

Following the three steps described in Section III, we construct the shortest path tree from each static ground node to the sink. Quality of each link on this tree is set to $p = 0.9$, which indicates the probability that a packet is successfully received on a link. Each packet can contain data from no more than 10 mobile nodes. This data includes location and motion data of the mobile node.

Mobile nodes move around randomly with constant speed v but randomly change the direction every one second. Mobile nodes are set to move at speed $v = 2m/s$. Static ground and mobile nodes employ MobiBone sleep scheduling, and the duty cycle is set to L seconds. As mentioned before, to guarantee sufficient space for the down-link time and the access time, the earliest start of the up-link time should not be earlier than $mL + (M+1)\Delta$. We set $(M+1)\Delta$ to $200ms$ and ∂_i to $5ms$. We assume that one packet transmission takes $5ms$. As there exist 13 layers, the earliest start of the up-link time can be found by:

$$\begin{aligned}(m+1)L - f[ID, S_u(m,13)] &= (m+1)L - \sum_{j=1}^{13} K_j * S_u(m,j) \\ &= (m+1)L - 2\sum_{j=1}^{12} S_u(m,j) + S_u(m,13)\end{aligned} \quad (2)$$

Therefore, when executing the flexible up-link time setting algorithm, $(m+1)L - 2\sum_{j=1}^{12} S_u(m,j) + S_u(m,13)$ should be larger than $mL + 200ms$. In the initialization, we set $m = 0$ and $S_u(0,j)$ to $70, 65, 60, 55, 50, 45, 40, 35, 30, 25, 20, 15, 10ms$.

We use number of detected mobile nodes and energy consumption of the static ground nodes as performance metric. For the energy consumption, we evaluate average energy consumption of all static ground nodes as well as the minimum and the maximum of the whole network.

B. Simulation results

Figure 8 shows the number of mobile nodes detected by the static ground nodes using our adaptive scheduling. As it can be seen, number of mobile nodes detected by the static ground sleep scheduling method is low and remains almost the same. Although the number of mobile nodes being detected by the adaptive scheduling is relatively low at the beginning, it increases over time. This is due to the adaptability feature of the protocol. It can also be observed that in fixed duty-cycle protocols, the longer the duty-cycle the lower the undetected mobile nodes. This higher detection rate, however, comes at the cost of energy consumption, as shown in Figure 9.

The trade off between number of detected mobile nodes and the energy consumption is presented in Table I. These results clearly show that an adaptive design approach to develop a network protocol is beneficial for achieving a good trade-off between energy consumption, packet delivery, and latency.

V. Conclusion

In this paper, we present MobiBone, an energy-efficient and adaptive network protocol to support successful data delivery between fast mobile nodes when they have a short rendezvous with static ground nodes. Mobibone is based on the pipeline sleep scheduling concept with an extra functionality to adapt the duty-cycle based on the data traffic. The supported traffics are uplink, downlink, and random access. The protocol utilizes three components, i.e., (i) traffic learning, (ii) flexible time slot setting, and (iii) and flexible duty cycle setting. Our performance evaluation in terms of number of detected mobile nodes (available data to be transferred) and energy consumption clearly show that adaptability of MobiBone offers a good trade-off between energy consumption, packet delivery, and latency. Although number of mobile nodes being detected by the MobiBone is relatively low at the beginning, it increases over time. This is due to the adaptability feature of the protocol.

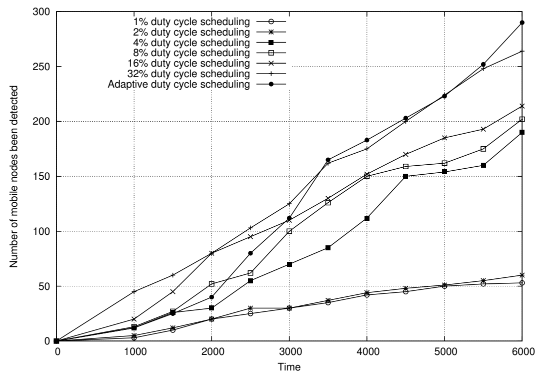


Fig. 8: Number of mobile nodes detected by static ground nodes

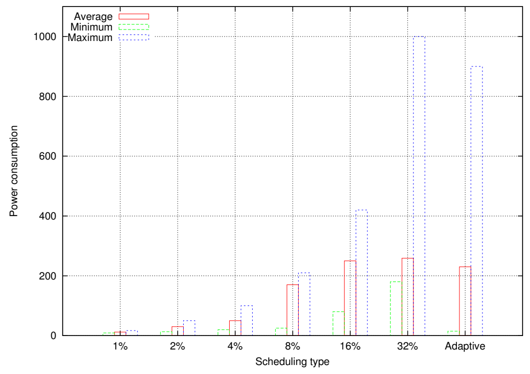


Fig. 9: Average, minimum, and maximum power consumption of the static ground nodes

TABLE I: The time required by the backbone nodes to detect mobile nodes and their corresponding and energy consumption

Scheduling Type	Duration (s)	Energy (mJ)
1%	0.23	15.94
2%	0.5	34.65
4%	0.27	18.71
8%	0.84	58.21
16%	1.17	81.08
32%	0.98	67.91
Adaptive	0.8	55.44

References

[1] Wenfeng Li, Junrong Bao, and Weiming Shen. Collaborative wireless sensor networks: A survey. In *Systems, Man, and Cybernetics (SMC), 2011 IEEE International Conference on*, pages 2614–2619. IEEE, 2011.

[2] Maxim A Batalin, Mohammad Rahimi, Yan Yu, Duo Liu, Aman Kansal, Gaurav S Sukhatme, William J Kaiser, Mark Hansen, Gregory J Pottie, Mani Srivastava, et al. Call and response: experiments in sampling the environment. In *Proceedings of the 2nd international conference on Embedded networked sensor systems*, pages 25–38. ACM, 2004.

[3] Nader Mohamed, Jameela Al-Jaroodi, Imad Jawhar, and Sanja Lazarova-Molnar. Failure impact on coverage in linear wireless sensor networks. In *Performance Evaluation of Computer and Telecommunication Systems (SPECTS), 2013 International Symposium on*, pages 188–195. IEEE, 2013.

[4] Gang Lu, Bhaskar Krishnamachari, and Cauligi S Raghavendra. An adaptive energy-efficient and low-latency mac for data gathering in wireless sensor networks. In *Parallel and Distributed Processing Symposium, 2004. Proceedings. 18th International*, page 224. IEEE, 2004.

[5] Qing Cao, Tarek Abdelzaher, Tian He, and John Stankovic. Towards optimal sleep scheduling in sensor networks for rare-event detection. In *Proceedings of the 4th international symposium on Information processing in sensor networks*, page 4. IEEE Press, 2005.

[6] Lodewijk Van Hoesel and Paul Havinga. Collision-free time slot reuse in multi-hop wireless sensor networks. In *Intelligent Sensors, Sensor Networks and Information Processing Conference, 2005. Proceedings of the 2005 International Conference on*, pages 101–107. IEEE, 2005.

[7] S Chatterjea, LFW Van Hoesel, and PJM Havinga. Ai-lmac: An adaptive, information-centric and lightweight mac protocol for wireless sensor networks. In *Intelligent Sensors, Sensor Networks and Information Processing Conference, 2004. Proceedings of the 2004*, pages 381–388. IEEE, 2004.

[8] Majid Nabi, Milos Blagojevic, Marc Geilen, Twan Basten, and Teun Hendriks. Mcmac: An optimized medium access control protocol for mobile clusters in wireless sensor networks. In *Sensor Mesh and Ad Hoc Communications and Networks (SECON), 2010 7th Annual IEEE Communications Society Conference on*, pages 1–9. IEEE, 2010.

[9] Antonio G Ruzzelli, Gregory MP OHare, and Raja Jurdak. Merlin: Cross-layer integration of mac and routing for low duty-cycle sensor networks. *Ad Hoc Networks*, 6(8):1238–1257, 2008.

[10] Fei Tong, Minming Ni, Lei Shu, and Jianping Pan. A pipelined-forwarding, routing-integrated and effectively-identifying mac for large-scale wsn. In *2013 IEEE Global Communications Conference (GLOBECOM)*, pages 225–230. IEEE, 2013.

[11] Yongle Cao, Shuo Guo, and Tian He. Robust multi-pipeline scheduling in low-duty-cycle wireless sensor networks. In *INFOCOM, 2012 Proceedings IEEE*, pages 361–369. IEEE, 2012.