# User Behavior Driven MAC Scheduling for Body Sensor Networks

Mao V. Ngo*†, Quang Duy La*, Derek Leong†, and Tony Q.S. Quek*
*Singapore University of Technology and Design, Singapore
†Institute for Infocomm Research (I$^2$R), A*STAR, Singapore
vanmao_ngo@mymail.sutd.edu.sg, quang_la@sutd.edu.sg, dleong@i2r.a-star.edu.sg, tonyquek@sutd.edu.sg

***Abstract*—We propose a new framework combining dynamic sampling rates for healthcare sensors driven by user behavior, and an adaptive MAC scheduling scheme applied to the time-slotted channel hopping (TSCH) protocol in IEEE 802.15.4, which provides high throughput and reliable communications. First, we introduce a system software architecture for machine-learning-assisted healthcare monitoring that detects the user's behavior using edge computing and adjusts the sampling rates of the healthcare sensors accordingly. Second, we propose an adaptive MAC scheduling scheme for TSCH based on a state machine model that reacts to the dynamic traffic generated by the healthcare sensors. In case of an urgent state, the MAC scheduler automatically allocates extra timeslots to the appropriate sensors so as to enable the reliable transfer of high-resolution sensor data for further analysis. Experimental results from our testbed, implemented in Contiki-OS on the OpenMote-CC2538 platform, show that the proposed adaptive scheduling scheme can respond quickly to changes in user behavior and ensure the reliable transfer of sensor data in emergency situations.**

## I. Introduction

Body sensor networks (BSNs) consist of multiple sensor nodes worn on or around the human body, sending and receiving data through wireless communications. Among the wireless technologies suitable for BSNs are Bluetooth, Bluetooth low energy (BLE) and IEEE 802.15.4 (ZigBee). IEEE 802.15.4 (ZigBee) is a low-power wireless protocol commonly used for BSNs, which has major applications in healthcare domain [1]. In most of these applications, the default MAC scheduling scheme for IEEE 802.15.4 is carrier-sense multiple access with collision avoidance (CSMA/CA). However, the performance of CSMA/CA suffers in environments with other coexisting technologies that also use the 2.4 GHz band for transmission [2], [3], or in dense environments with multiple co-located BSNs [4]. To address the limitations of CSMA/CA, we use time-slotted channel hopping (TSCH), which is an emerging MAC protocol defined in IEEE 802.15.4 [5]. TSCH combines time-slotted access and multi-channel hopping capabilities to provide deterministic latency for applications, and mitigate the effects of interference and multi-path fading, improving the reliability of wireless communications in the presence of heterogeneous technologies. As such, TSCH is well-suited for healthcare applications where sensors transmit data regularly, in close proximity to coexisting wireless systems.

MAC scheduling based on TSCH was investigated in [6], [7]. Duquennoy *et al.* [6] proposed and implemented in Contiki-OS a scheduling scheme called Orchestra that uses local knowledge of sensor neighborhoods. Jin *et al.* [7] developed a centralized scheduling algorithm for TSCH to allocate as many timeslots in a slotframe as possible, which guarantees low latency for time-critical applications in multi-hop networks. However, it allocates too many timeslots in a slotframe even when not required by the nodes, leading to the problem of resource hogging. Our paper extends the sender-based scheme from Orchestra and builds an adaptive MAC scheduling scheme to increase link utilization, allocating extra timeslots in a slotframe only when the application needs higher bandwidth. In addition, we build an end-to-end system for autonomous healthcare monitoring that adaptively adjusts the sensor sampling rates and dynamically allocates the appropriate number of timeslots for emergency sensors, based on the requirements from the application layer, and releases them promptly after the emergency period.

An adaptive sampling rate mechanism provides several benefits for a healthcare monitoring system. First, it can save energy for healthcare sensors powered by small batteries. The adaptive sampling rate mechanism can switch the sensors from a low sampling rate with infrequent transmissions in a normal situation, to a high sampling rate with frequent transmissions in an emergency situation. Using this mechanism, the system can conserve energy when high-resolution sensor data is unnecessary. Consider, for example, a healthcare monitoring system for a heart disease patient undergoing cardiac rehabilitation. The battery-powered ECG sensor on the patient's body can take measurements at a low sampling rate when the patient is sedentary with a low risk of a heart attack or cardiac event (e.g., while sitting or standing). However, when the patient begins a more vigorous activity with a higher risk of a cardiac event (e.g., running or even walking), the ECG sensor can switch to a higher sampling rate, so that high-resolution data can be collected for more complex analysis, to provide more accurate observations or predictions. Subsequently, this high-resolution data can also be reviewed offline by the cardiac specialist overseeing the patient's rehabilitation.

To support the adaptive sampling rate mechanism, the

The work of D. Leong was supported in part by A*STAR, Singapore, under SERC Grant 1524100032.

system needs an adaptive MAC scheduling scheme for the network to handle the additional data traffic. In emergency situations, some high-priority sensors could generate more data than can be supported by the current MAC schedule. A simple solution to this problem is to allocate as much bandwidth as possible for both emergency and normal situations. However, this could cause the sensors to wake up for listening too frequently in the normal situation, increasing the energy consumption unnecessarily. Furthermore, the bandwidth may not be allocated to the right sensors that need it most, resulting in suboptimal resource allocation and an unscalable BSN. We therefore need a MAC scheduling scheme that can promptly allocate enough bandwidth for the higher sampling rates in an emergency situation, and release the allocated bandwidth when the emergency situation is over, ensuring a high quality of service (QoS) at all times.

Apart from enabling adaptive sampling rates and adaptive MAC scheduling, we also need the healthcare monitoring system to quickly detect whether the user is in a normal or emergency situation. To achieve this, we need to locate the decision-making module close to the sensor nodes, such as in a gateway, so that the decision can be made at the edge without sending data to the cloud or back-end server which could incur long delays. Such a decision-making module would essentially be performing human activity recognition (HAR) [8], [9]. While most prevailing HAR approaches use fairly complex models with large feature sets for better accuracy, it may be desirable for BSNs to use simpler lightweight models with lower accuracy but better response time and energy efficiency, so as to minimize delays and preserve battery life.

To the best of our knowledge, this is the first work introducing a jointly dynamic sensor sampling scheme and adaptive MAC scheduling scheme based on TSCH for BSNs, for a healthcare monitoring system. Our main contributions in this paper are as follows:

- We describe a system software architecture for machine learning assisted healthcare monitoring that detects the user's behavior to dynamically adjust sampling rates. The decision making is performed in an edge node to reduce latency.
- We present an adaptive MAC scheduling scheme for BSNs based on the IEEE 802.15.4 TSCH protocol, implemented as a state machine model, to support the proposed dynamic sampling rate mechanism.
- The proposed healthcare monitoring system is implemented in a gateway and the border-router, using Contiki-OS [10] on the OpenMote-CC2538 platform. Experimental results obtained from our testbed show that our system can quickly respond to dynamic traffic demand and ensure reliable communications in emergency situations.

## II. System Overview

We consider a BSN comprising one or more wearable sensors that are periodically sampling physiological signals and sending the data to a gateway wirelessly using IEEE 802.15.4; the gateway, which possesses greater storage and processing power than a sensor node, communicates with a back-end server over an Internet connection. Such a BSN could correspond to an individual patient whose conditions are being monitored in real-time.

In the gateway, we have a health-monitoring module that automatically detects the occurrence of urgent events in human activities, determines the appropriate sampling rates for the sensors, and provides feedback to the scheduling module. To perform the required human activity recognition, we can train a machine learning model offline using collected sensor data at the back-end server. Subsequently, using the trained model, the health-monitoring module at the gateway can classify the human activity in real-time, to quickly determine whether the individual is resting, running, overexerting, in distress, etc.

Fig. 1a shows the software architecture of the system, which consists of three main parts: the BSN with star topology, the gateway, and the back-end server.

The gateway has two interfaces with the BSN and the server. The gateway receives collected data from all sensors at the *border-router*, and then forwards them to the health-monitoring module via the *tun-slip-ipv6* module. Subsequently, the data can be analyzed at the decision making module, stored in the database (e.g., SQLite3), or forwarded to the back-end server. When an emergency event occurs, a decision is passed from the decision-making module to the border-router via SLIP (Serial Line Internet protocol) interface. Consequently, the border-router runs the adaptive MAC scheduling algorithm based on the current network conditions and the requirements of the emergency sensor (e.g., increased sending rate). The details of the adaptive MAC scheduling algorithm are described in Section IV. Note that in such a system, we are employing *edge computing* by migrating computations and storage of data to edge nodes right next to the sensors.

## III. Machine Learning-Assisted Healthcare Monitoring

The objective of the health-monitoring module is to track the physiological conditions of the patients, predict or draw attention to the occurrence of abnormal conditions. Because such occurrences are very rare, sampling at high frequency all the time could generate a lot of redundant data while wasting energy. Therefore, it is desirable for sensors to use a low sampling rate or sleep most of the time (e.g., most of heart rate monitoring duration), and only sample at high frequency in "urgent" cases (e.g., in intensive activities). In order to obtain a fairly accurate detection of such urgent situations, we implement the health-monitoring module which utilizes a *machine learning* model in order to classify different human activities, as a proof of concept. For scalability and fast real-time implementation, we consider a lightweight approach which only uses accelerometer sensors and a few extracted features to detect *four human activities*: sitting, standing, walking, and running.

**Data Collection**: In this system, the triaxial accelerometer sensor is worn on the left hand of the user via a strapped band,

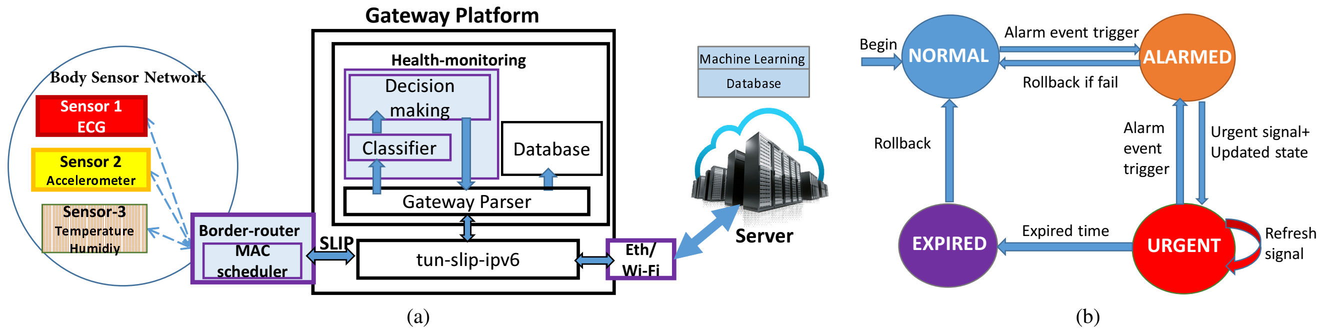



Fig. 1. (a) System software architecture. (b) State-machine of the scheduling scheme.

and samples with frequency 30 Hz. Data was collected for each of the four labels on two male adults of average build. Each activity was performed for 5 minutes with precisely recorded timestamps, which will be used for labeling later. The dataset contains 72000 three-dimensional accelerometer samples.

**Feature Extraction**: Several consecutive samples of the same signals are grouped into one window. The window's size for feature selection is empirically selected at 2 seconds so that it is not too long but also not too short to capture enough feature data for human activities. Two consecutive windows overlap by 50%. Subsequently, we carry out the feature extraction process with the following set of features: *minimum, maximum, mean, variance, skewness* and *kurtosis*, which are computed across all samples in each window.

**Training Machine Learning Models**: The following algorithms are utilized: *Support vector machine (SVM)*, *Decision tree*, *Gaussian Naive Bayes (GNB)*. We follow the standard practice of dividing the entire data set into a *training set* (70%) and a *test set* (30%) via random sampling. We use the off-the-shelf library *scikit-learn* [11] in Python to implement these machine learning models. Table I summarizes the final accuracy performance of the selected algorithms. Base on the results, in subsequent experiments we use *Decision tree* model for classification.

TABLE I
ACCURACY OF MACHINE LEARNING MODELS

| Model | Training accuracy (%) | Test accuracy (%) |
|---|---|---|
| SVM | 98.73 | 97.65 |
| Decision tree | 100 | 99.60 |
| Gaussian Naive Bayes | 98.40 | 98.11 |

## IV. An Adaptive MAC Scheduling Scheme

### A. Scheduling with Time Slotted Channel Hopping

In TSCH networks [5], all nodes synchronize on a periodic slotframe which contains a number of timeslots. A timeslot can be assigned to a pair of transmitting and receiving nodes. Each timeslot's duration, typically 10ms, is long enough for a node to transmit a maximum-size data packet and receive the acknowledgment (ACK). A TSCH link between two nodes is described by two parameters: timeslot in the slotframe and a channel in the frequency hopping channel list. Fig. 2 shows an example of allocating timeslots to the border-router and three sensors. There are two type of links – downlink and uplink. In the downlink timeslot, the border-router sends to a destination sensor, which may change from slotframe to slotframe. In the uplink timeslot, the sensor node sends to the border-router. It can be a dedicated uplink, which is allocated to a single sender-receiver pair at a timeslot, i.e., contention-free; or a shared uplink, which is allocated to more senders transmitting to a receiver, i.e., contention-based with back-off mechanism.

The adaptive MAC scheduler solves the problem of assigning available TSCH timeslots to the desired sensors while maintaining the QoS of the other sensors. When the emergency sensor increases its sending rate, the MAC scheduler assigns the appropriate number of dedicated uplinks to it, which do not affect the performance of the shared uplinks of the other sensors. This makes sure the QoS of the other sensors are still guaranteed while supporting the newly generated traffic from the emergency sensors. Whenever the emergency period is expired, these extra dedicated timeslots are released back to the available timeslots pool of the whole network. Fig. 2 presents an example of extra dedicated uplinks from sensor 3 to the border-router, which are denoted as TX with an arrow below, instead of TX(S) for a shared uplink initially allocated when joining the network. The extra uplinks are allocated and added in the slotframe with equally spaced timeslots fashion, which aims to create regular gaps between two packets.

### B. The Adaptive MAC Scheduling Scheme as a State Machine

The whole scheduling module operates according to the state-machine depicted in Fig. 1b. There are four distinct states: NORMAL, ALARMED, URGENT and EXPIRED states, each of which corresponds to different sets of operations for the sensors and the gateway. The scheme initiates at NORMAL state, then changes to ALARMED state when the health-monitoring module triggers an alarm event. The border-router and the emergency sensors cooperate during ALARMED state and move to URGENT state when they successfully allocate extra links for newly-generated data packets. Note that each sensor maintains its own state-machine, and the border-router maintains a set of state-machines for each associated sensor.

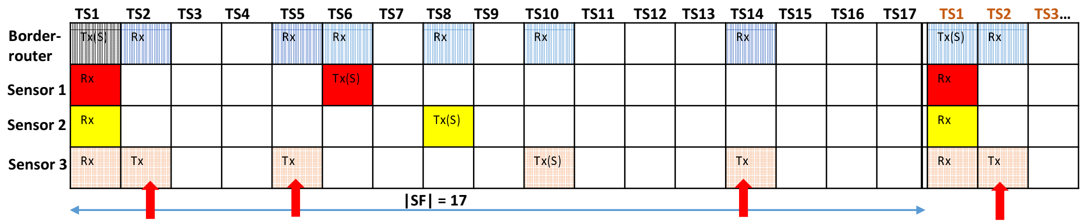



Fig. 2. Allocation of timeslots in a TSCH slotframe to sensors in ALARMED state with equal space extra timeslot allocation scheme. The timeslot with an arrow below denotes the extra timeslot in URGENT state.

*1) NORMAL State:* In this initial state, sensor nodes join to the BSN after receiving advertising messages from the border-router (i.e., Routing Protocol for Low-Power and Lossy Networks (RPL) root), and have been assigned the uplinks to upload the collected sensor data to the border-router. The network is initialized with two parameters: slotframe size ($|SF|$), and timeslot duration ($|TS|$). For example, Fig. 2 shows our setup with $|SF| = 17$ timeslots, and timeslot duration $|TS| = 10ms$. With these parameters, the number of slotframe per second ($N_{SF}$) is given by $N_{SF} = \left\lfloor \frac{1000}{|SF| \cdot |TS|} \right\rfloor = \left\lfloor \frac{1000}{17 \cdot 10} \right\rfloor = 5$. The sending rate $r$ (packet/s) is the number of sent packets per second. Each sensor node is assigned a number of $k$ timeslots within a slotframe, which is typically $k = 1$ in the normal situation [6].

*2) ALARMED State:* The ALARMED state is a transient state which transforms the system from the NORMAL state to the URGENT state. When the decision-making module detects an emergency health state, it will generate an emergency signal to the border-router. The emergency sensor node requires a new sending rate, which is based on the specific application and calculated as $r' = \alpha \times r$, where $r$ and $r'$ are the old and new sending rates respectively; and $\alpha$ is the scale factor of sampling rate. The adaptive MAC scheduler allocates a number of extra timeslots within a slotframe $N_{ex}$ for the new sending rate, which is calculated as $N_{ex} = \left\lceil \frac{r'}{N_{SF}} \right\rceil - k$. During the ALARMED state, the border-router allocates the extra timeslots from the available pool of timeslots in equally spaced timeslots in the slotframe; and then sends a control message to the emergency sensor for configuring new sampling frequency and adding extra TX-timeslots. The control message sent by the border-router contains the following parameters: **[signal type, $\alpha$, expired duration, $N_{ex}$, set of extra timeslots]**.

Before the border-router sends the control message, it adds the appropriate RX-timeslots to itself. Once the emergency sensor node receives the control message, it changes to the ALARMED state and adds the extra TX-timeslots based on the set of extra timeslots in the received control message. Since the control message can be lost, or the assignment of extra timeslots at the emergency sensor may fail, we design a mechanism that tries to re-send several control messages (e.g., at most three times), and allows a delay period for processing at the emergency sensor. After the tolerant delay period (e.g., 3 seconds), if the border-router still does not receive any updated state from the emergency sensor, it will roll back to the NORMAL state by removing the advanced allocated extra RX-timeslots, and informs the health-monitoring module for further decision.

*3) URGENT State:* Once the emergency node successfully adds the extra TX-timeslots, it increases the sampling rate and moves to the URGENT state immediately. The emergency sensor node in the URGENT state sends the data with the updated state as piggybacked data to the gateway via the normal TX-timeslot or the extra TX-timeslots. The border-router receives the updated state from the emergency sensor node as an acknowledgment signal. Subsequently, both the border-router and the emergency sensor node have entered the URGENT state. The normal sensors are still sampling at the normal sampling rate, and sending the sensor data to the border-router via their normal uplinks. As such, the adaptive MAC scheduling scheme does not affect the performance of the non-emergency sensors. During the URGENT state, the decision-making module can either extend the emergency period, or escalate the level of emergency by further increasing the sampling rate of a normal sensor or the sensor already in the URGENT state. In the latter situation, the decision-making module sends a different trigger signal type to the border-router; then the border-router moves back to the ALARMED state and allocates extra timeslots from the current pool of available timeslots. When the border-router receives the updated state from the escalated emergency sensor, it changes to the URGENT state. If the emergency sensor does not receive any extending signal from the border-router to refresh the URGENT state before the emergency period expires, it will move to the EXPIRED state. Meanwhile, the border-router is still in the URGENT state.

*4) EXPIRED State:* In the EXPIRED state, the emergency sensor node reduces its sampling rate to the normal value, removes the all extra TX-timeslots, and then rollbacks to the NORMAL state. After that, it sends the updated state (NORMAL) to the border-router in the next data packet in piggybacked fashion. Once the border-router receives the updated state, it changes the state from URGENT to EXPIRED state, and removes the extra RX-timeslots. Subsequently, the border-router changes to the NORMAL state. Note that the order of deleting extra timeslots is important: first the extra TX-timeslots are deleted in the emergency sensor node, then the extra RX-timeslots are deleted in the border router. This ensures there is no missing data packet sent from the emergency sensor node in the extra timeslots before they are removed.

TABLE II
STATE PROFILES WITH SAMPLING RATE $f$ (HZ) AND SENDING RATE $r$ (PACKETS/S)

| State profile | ECG | | Accelerometer | | Temperature | |
|---|---|---|---|---|---|---|
| | $f$ | $r$ | $f$ | $r$ | $f$ | $r$ |
| Normal | 64 | 2 | 30 | 3 | 2 | 1 |
| Urgent-medium | 128 | 4 | 60 | 6 | 2 | 1 |
| Urgent-high | 256 | 8 | 120 | 12 | 2 | 1 |

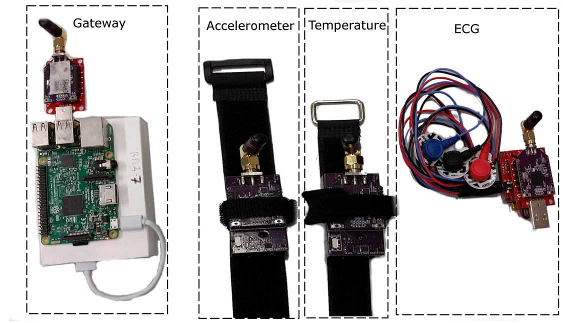



Fig. 3. Experimental testbed.

## V. EXPERIMENTAL EVALUATION

### A. Experimental Setup and Network Performance Metrics

To evaluate network performance, the proposed system is compared to a baseline that uses the sender-based dedicated Orchestra scheduling scheme [6], which is the most widely used MAC scheduler for TSCH. Orchestra allocates one timeslot per sensor per slotframe for the whole lifetime of the network. Note that most other adaptive MAC scheduling schemes have been designed for CSMA and are not suitable for our comparison. Both systems change sampling rates according to the user behavior described in Table II. The accelerometer and ECG sensors increase their sampling frequencies when the user changes behavior, while temperature sensor keeps the same sampling frequency due to the human temperature cannot change sharply in a short time (half of a second). The experiment is conducted with three human activities *sitting/standing*, *walking*, and *running*, which map to the *normal*, *urgent-medium*, and *urgent-high* states of the BSN, respectively.

In order to verify the robustness of our state-machine model, we also conduct the experiment in two scenarios. In the **first scenario**, the *urgent-medium* and *urgent-high* states are raised in two disjoint periods and directly from the *normal* state. In particular, the user is initially in sitting/standing posture (*normal* state); then changes to walking activity (*urgent-medium* state). After that, the user rests for a few minutes in standing posture and then changes to running (*urgent-high* state). In the **second scenario**, the *urgent-medium* and *urgent-high* states are raised contiguously which aims to verify the robustness of our system. The user starts in the standing/sitting posture, then changes to walking activity. After two minutes of walking, the user sharply changes to running activity, such that the system has to increase immediately the sampling frequency and add more timeslots from the current URGENT state.

Note that although higher frequency sensor data is produced in the urgent states, we still use the lower frequency sensor data for the purpose of classifying the human activity because the machine learning models have been trained with the normal sampling rate. In our work, we simply down-sample the higher frequency data and feed it to the health-monitoring module. The higher frequency data can be used by other applications, or can be stored in the database for further offline analysis; this usage is beyond the scope of the current paper.

We use the following QoS metrics to evaluate the performance of the BSNs. These metrics are calculated using a sliding window:

1) *Packet delivery ratio* (PDR) is an indicator for the network *reliability*. PDR is defined as the percentage of data packets successfully delivered from the sensor nodes to the gateway. Let $N_P$ and $N_P^*$ be the total number of transmitted data packets and the number of successfully delivered data packets, respectively. The PDR is given by PDR$=\frac{N_P^*}{N_P}$.
2) *Goodput* (G) is the application-level throughput, which denotes the number of information bits, excluding protocol overheads and data retransmission, forwarded successfully from the source to the targeted destination per unit of time. The goodput is given by G $= \frac{N_P^* \times Size}{t}$.

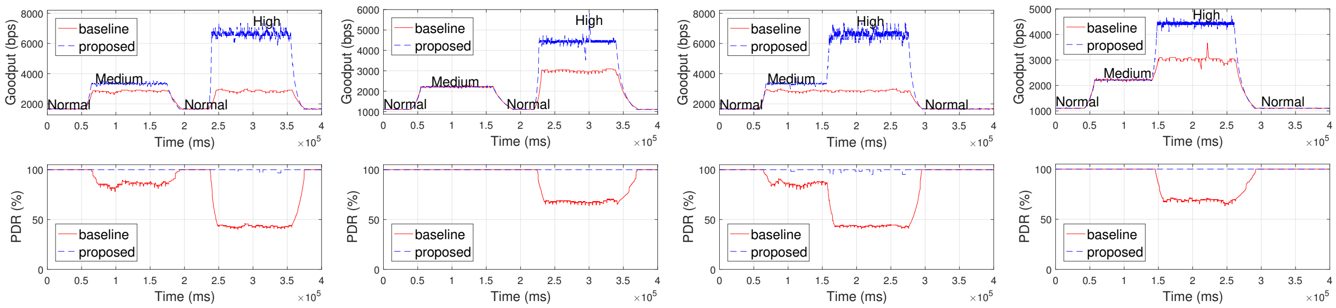
### B. Testbed

Fig. 3 shows the components of our experimental testbed. Details of each component are given as follows.

1) *Sensors*: We consider three different types of sensors, i.e., accelerometer (ADXL346), temperature and humidity (SHT21), and ECG sensor plugged into a SparkFun AD8232 board. These sensors use the OpenMote-CC2538 platform as the radio communication module.
2) *Border-router*: We use another OpenMote-CC2538 platform plugged into the gateway via serial connection as the border-router. The operating system Contiki-OS [10] is used for programming the sensors and the border-router, and use TSCH for the MAC layer and (RPL + 6LowPAN) for the network layer.
3) *Gateway*: Raspberry-Pi 3 is used as the gateway which handles several tasks such as collecting, processing, storing data locally, and forwarding the data to the back-end server.
4) *Server*: A DELL workstation running Ubuntu with 64 GB RAM is used for back-end analysis, e.g., in building machine-learning models for monitoring and detection, and analyzing the overall network performance.

### C. Experimental Results

The experimental results of the first and second scenarios with three different states are plotted in Figs. 4a, 4b, and in Figs. 4c, 4d.




(a) (b) (c) (d)

Fig. 4. Goodput and PDR of (a) accelerometer in the *first scenario*, (b) ECG in the *first scenario*, (c) accelerometer in the *second scenario*, and (d) ECG in the *second scenario*. The red-solid line represents the *baseline system*; and the blue-dashed line represents our *proposed system*.

In the **first scenario**, Figs. 4a, 4b show the goodput and PDR of accelerometer and ECG, respectively. In the *normal* state, both systems provide the same goodput and reliability (PDR≈100%). When the system triggers a walking activity, the health monitoring module changes the sampling rates to *urgent-high* according to Table II. Since the sampling rates are increased, the accelerometer and ECG sensors generate more data packets. While the baseline system provides PDR=100% and goodput for ECG sensor as high as our proposed system provides, it cannot fully support the accelerometer sensor newly generated data. Specifically, accelerometer's PDR of the baseline system gradually drops to 85% approximately, and the accelerometer's goodput in the baseline system is 16% less than our proposed system. It is because the accelerometer's sending rate of 6 packet/s exceeds the maximum number of timeslots per second ($N_{TS} = N_{SF} \cdot k = 5 \cdot 1 = 5$ packet/s) under the baseline TSCH scheme. After the medium traffic period, the user reverts to the sitting/standing activity (normal traffic) for a duration of about 1 minute. Subsequently, the user changes to the running activity (*urgent-high* state). In this state, our system performs significantly better than the baseline system in terms of both goodput and PDR, for the accelerometer (100% and 50% higher, resp.) and the ECG (46% and 68% higher, resp.).

In the **second scenario**, the *urgent-high* state is triggered when the system is in the *urgent-medium* state. As seen in Figs. 4c, 4d, we have the same observations for the *normal* and *urgent-medium* states as in the first scenario. When the user immediately changes to *urgent-high* state, we observe similar results as in the first scenario. Except that the accelerometer's PDR of our system sometimes decreases to 99.3%, which is worse than that of the first scenario. Meanwhile, the baseline system keeps the same goodput and PDR as low as that in the first scenario for both emergency sensors.

Regarding temperature sensor, the PDR and goodput are maintained high in both systems during the testing time. This illustrates that our system not only provides high QoS for sensors with high data rate, but also maintains the QoS of the other low data rate sensor. We do not include the temperature sensor's results in this paper due to space limitation.

## VI. Conclusion

This paper proposes a framework with adaptive sampling and dynamic timeslot allocation within a slotframe for TSCH. The state-machine module is used to handle the allocation between the border-router and the emergency sensors. The health-monitoring module detects human activity and adjusts the sampling scheme accordingly. In the experiments, the adaptive MAC scheduling scheme is shown to maintain a high QoS of communication channel (almost 100% PDR all the time) and provide high goodput compared to the baseline system.

## References

[1] M. Chen, S. Gonzalez, A. Vasilakos, H. Cao, and V. C. M. Leung, "Body area networks: A survey," *Mobile Netw. App.*, vol. 16, no. 2, pp. 171–193, 2011.

[2] R. G. Garroppo, L. Gazzarrini, S. Giordano, and L. Tavanti, "Experimental assessment of the coexistence of Wi-Fi, ZigBee, and Bluetooth devices," in *Proc. IEEE WoWMoM*, Jun. 2011, pp. 1–9.

[3] M. U. Memon, L. X. Zhang, and B. Shaikh, "Packet loss ratio evaluation of the impact of interference on zigbee network caused by Wi-Fi (IEEE 802.11b/g) in e-health environment," in *Proc. IEEE Healthcom*, Oct. 2012, pp. 462–465.

[4] S. Liang, Y. Ge, S. Jiang, and H. P. Tan, "A lightweight and robust interference mitigation scheme for wireless body sensor networks in realistic environments," in *Proc. IEEE WCNC*, Apr. 2014, pp. 1697–1702.

[5] "IEEE Standard for Low-Rate Wireless Networks," *IEEE Std 802.15.4-2015*, vol. 1, pp. 57–150, 2015.

[6] S. Duquennoy, B. Al Nahas, O. Landsiedel, and T. Watteyne, "Orchestra: Robust mesh networks through autonomously scheduled TSCH," in *Proc. ACM SenSys*, 2015, pp. 337–350.

[7] Y. Jin, P. Kulkarni, J. Wilcox, and M. Sooriyabandara, "A centralized scheduling algorithm for IEEE 802.15.4e TSCH based industrial low power wireless networks," in *Proc. WCNC*, April 2016, pp. 1–6.

[8] O. D. Lara and M. A. Labrador, "A survey on human activity recognition using wearable sensors," *IEEE Commun. Surveys Tuts.*, vol. 15, no. 3, pp. 1192–1209, Third quarter 2013.

[9] M. Cornacchia, K. Ozcan, Y. Zheng, and S. Velipasalar, "A survey on activity detection and classification using wearable sensors," *IEEE Sensors J.*, vol. 17, no. 2, pp. 386–403, Jan. 2017.

[10] A. Dunkels, B. Gronvall, and T. Voigt, "Contiki - a lightweight and flexible operating system for tiny networked sensors," in *Proc. IEEE Local Comput. Netw.*, Nov. 2004, pp. 455–462.

[11] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *J. Mach. Learn. Res.*, vol. 12, pp. 2825–2830, 2011.