Krzysztof TURCHAN[1], Krzysztof PIOTROWSKI[1], Emil MICHTA[2]

[1]IHP- Innovations for High Performance Microelectronics, Frankfurt (Oder), Germany
[2]Uniwersytet Zielonogórski, Instytut Metrologii, Elektroniki i Informatyki, Zielona Góra, Polska

EFFICIENT AND FLEXIBLE INTELLIGENCE ON THE EDGE

The paper presents a way to move away from the classical approach to data transfer in WSN systems. It aims to reduce the amount of data transmitted in the WSN by preprocessing data at edge nodes, which directly affects the occupancy of free radio frequencies such as 868 MHz.

WYDAJNE I ELASTYCZNE PRZETWARZANIE NA BRZEGOWYCH WĘZŁACH BEZPRZEWODOWEJ SIECI SENSORÓW

W artykule przedstawiono sposób na odejście od klasycznego podejścia do przesyłania danych w bezprzewodowych sieciach sensorów. Ma on na celu zmniejszenie ilości danych przesyłanych w sieci poprzez wstępne przetwarzanie danych na węzłach brzegowych, co bezpośrednio wpływa na zajętość darmowych częstotliwości radiowych, takich jak 868 MHz.

1. MOTIVATION

In wireless sensor networks (WSNs), the conventional strategy is to transmit measurement data from edge nodes (sensor data collection nodes) to the base node (the node for processing measurement data) [1]. However, this technique has a number of disadvantages, including the possibility of data loss due to low network bandwidth or limited resources of the base node.

WSNs make extensive use of open civilian frequencies such as 863-870 MHz. However, using these frequencies requires compliance with rules, which are primarily concerned with the amount of data transmitted per hour [2]. In the case of the 863-870 MHz frequencies, on which operates well-known system such as LoRaWAN, the number of constantly operating devices is large, and thus the frequency occupancy rate is also higher. With the help comes Intelligence on the Edge. It is a solution in which the entire measurement data management process starts at the edge nodes, reducing the amount of transmitted data as much as possible. This document describes how to effectively delegate tasks to all WSN nodes in order to dispose of the network bandwidth problem. It also describes proposed approach for pre-processing data collected from the environment into a most relevant information.

2. DUTY CYCLE / TIME ON AIR (TOA)

Before describing approaches, it is necessary to present the terms related to frequency capacity, because algorithms will be evaluated on this basis. *Time on Air* is a time between sending data and receiving it by the receiver. The duty cycle of a device, or system is the percentage of time that it is used. The duty cycle can be stated as a percentage or a ratio. In Europe the duty cycles per hour in the case of LoRaWAN are 0.1%, 1.0% and 10%, depending on the channel. It means that the device can send data only 3.6, 36 or 360 seconds in 1 hour [3]. A device that exceeds its duty cycle will be blocked for 1 hour, so it is important to respect this regulation.

Equation 1 shows an example calculation of time between sending next packets with a duty cycle of 1%. If the baud rate is 0.3 kb/s and the packet size is 42 bytes then:

$$ToA = \frac{bytesToSend}{transmissionSpeed} = \frac{42b}{300b/s} = 0{,}14s = 140ms \quad (1)$$

And the delay time is:

$$DelayTime = (100 - dutyCycle) * ToA = (100 - 1) * \ 140ms = 13{,}86\ s \tag{2}$$

This means that a packet of this size should not be sent more than once every 13,86 seconds in order to stay under the fixed 1% limit. The delay time can be manipulated by reducing number bytes to send or increasing the transmission speed, but it is always worth to consider the worst case when designing a system.

3. INTELLIGENCE ON THE EDGE

Decentralized nodes of a WSNs can use an intelligent edge to handle different types of data that would normally be handled at a central point in the WSNs. For WSNs in particular, the classical model of routing all data streams from nodes to a central data store (base node) has several issues. The first issue has to do with retention of the duty cycle, as mentioned previously. The second issue is that the data collected from the sensors is not encrypted thus the system may be more vulnerable to attacks.

Edge nodes in a smart Edge configuration can intelligently process data, possibly bundling, refining or encrypting it for transmission to a base node. This allows systems that process data to be more flexible, while also increasing their cybersecurity[4]. Chapter 4 describes an approach to increase the efficiency and flexibility of a WSNs using intelligence on the edge.

4. PROPOSED APPROACH

Wireless sensor networks are used in many areas of life. Examples include military, medicine, air traffic control, health and property protection, intelligent buildings. Each of these areas generates large number of data that must be transferred to the base node. Proposed approach of preprocessing this data is called Filtering. Filtering is a method designed for edge nodes and aims to respond to changes in the state of an observed variable. The operation of this method is illustrated using a floodbank monitoring system in a riverside town. The entire system is a wireless sensor network designed to measure weather conditions and to detect a dangerously rapid rise in the water table's height. Each measuring point is equipped with a weather station, soil moisture sensor and piezometer. Subsections 4.1 - 4.3. present the configuration process and the way the described method works.

4.1. Configuration Process

The node configuration process is divided into two parts. The first part includes selecting which of the sensors connected to the edge node will be filtered. It also involves entering two values for each edge node which are *dt* (Delta time) and *dv* (Delta value). The second part deals mainly with the configuration of measurement and sending times. The operation of the node will depend on this configuration. Configuration takes place by filling in the configuration structures, shown in Figures 1 and 2.

```
struct node_preConfig{
    uint16_t measurements_period;
    uint16_t verification_packet_delay;
};
```

Fig. 1. node_PreConfig configuration structure
Rys. 1. Struktura konfiguracyjna do konfiguracji węzła node_preConfig

```
struct sensor_properties{
    uint8_t status;     //1 - turn on, 0 - turn off
    uint16_t dt;        //Delta time (seconds)
    uint16_t dv;        //Delta value (percent)
};

//List of available sensors
struct sensors_preConfig{
    struct sensor_properties sensor_BME280_temp;
    struct sensor_properties sensor_BME280_Hum;
    struct sensor_properties sensor_BME280_Press;
    struct sensor_properties sensor_SPS30_PM_2_5;
    struct sensor_properties sensor_SPS30_PM_10;
    struct sensor_properties sensor_HCHO;
    struct sensor_properties sensor_NO2;
    struct sensor_properties sensor_SO2;
    struct sensor_properties sensor_RAIN;
    struct sensor_properties sensor_WindSpeed;
    struct sensor_properties sensor_WindWane;
    struct sensor_properties sensor_3Ch_meter;
    struct sensor_properties sensor_RTC_DS3241;
    struct sensor_properties sensor_GroPoint;
};
```

Fig. 2. Sensors_preConfig and sensor_properties configuration structures
Rys. 2. Struktury konfiguracyjne Sensors_preConfig i sensor_properties do konfiguracji sensorów.

4.2. The Idea Behind the Algorithm

After filling in these structures, the algorithm knows how to react to a change in the state of the enabled sensors. First, the program calculates the number of samples to store and compare by dividing *dt* by *measurements_period,* in this case it is 60s/15s. These samples are highlighted in light green in the figure 3 The example is realized only on measurements from the BME_280_temp sensor. Next, a new value is taken from the sensor. This is followed by a comparison of the previous 4 values with +/-15% of the new value. If any of the stored variables exceeds the set thresholds then new value is moved to be sent. This process is repeated for all activated sensors. If at least one of all running sensors exceeds the thresholds set in the configuration process, the packet will be sent. This approach ensures that data will be sent every measurement in a worst case scenario. However, only data from sensors that have exceeded this threshold are sent, and as mentioned in Chapter 2 - the less data the shorter the Time on Air.

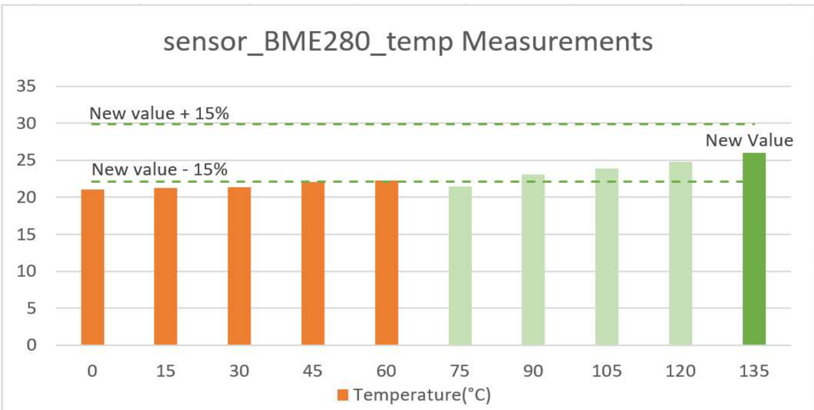

Fig. 3. Graphical visualization of algorithm operation based on BME_280 sensor data
Rys. 3. Graficzna wizualizacja działania algorytmu na podstawie danych z sensora BME_280

There could be a situation where the change will occur too slowly and the data could not be sent at all. For this purpose the variable *verification_packet_delay* has been implemented. This variable determines when a packet containing all measurement data is sent. This is the best situation because the measurement data are sent at the time specified in this variable. It can be for example 10 minutes which significantly reduces ToA. The next subsection presents the calculations for the presented method.

4.3. ToA and Delay Times calculations

The measurement data from all sensors connected to the station is 157 bytes. Assume that the transmission rate is 1.1 kb/s. In the case where all data is sent every 15 seconds, ToA is about 0.143 s. This gives the capability to send data with delay time 14.13 seconds. Table 1 shows different ToA and DelayTimes values depending on the number of bytes reserved by each sensor.

Table 1

Sensor	Bytes	ToA(s)	DelayTime(s)
sensor_BME280_temp	4	0,0036	0,36
sensor_BME280_Hum	4	0,0036	0,36
sensor_BME280_Press	4	0,0036	0,36
sensor_SPS30_PM_2_5	4	0,0036	0,36
sensor_SPS30_PM_10	4	0,0036	0,36
sensor_HCHO	4	0,0036	0,36
sensor_NO2	4	0,0036	0,36
sensor_SO2	4	0,0036	0,36
sensor_RAIN	4	0,0036	0,36
sensor_WindSpeed	4	0,0036	0,36
sensor_WindWane	1	0,0009	0,09
sensor_3Ch_meter	24	0,0218	2,16
sensor_RTC_DS3241	4	0,0036	0,36
sensor_GroPoint	88	0,0800	7,92

How much data will be transmitted the base node depends on the variability of the measurement data. This shows that ToA will be 0.143 s in the worst case and 0.0009 s in the best case when sending 1 byte. Of course, it is also necessary to send all data at the intervals defined in the verification packet delay variable.

ACKNOWLEDGMENTS

This work was supported by the European Regional Development Fund within the BB-PL INTERREG V A 2014-2020 Programme, “reducing barriers - using the common strengths”, project SmartRiver, grant number 85029892 and by the European Union ebalance-plus project under the H2020 grant number 864283. The funding institutions had no role in the design of the study, the collection, analyses, or interpretation of data, in the writing of the manuscript, or in the decision to publish the results.

REFERENCES

1. Aquino-Santos R., Villasenor-Gonzalez L., Sanchez J., Rosario Gallardo J.: Routing Strategies for Wireless Sensor Networks, University of Colima, Mexico, 2007.

2. The Things Network, Duty Cycle, Available at: https://www.thethingsnetwork.org/docs/lorawan/duty-cycle/, (last accessed on: 20.04.2022).

3. LoRa, LoRa Documentation, Available at https://lora.readthedocs.io/en/latest/#duty-cycle-time-on-air-toa, (last accessed on: 20.04.2022).

4. Wang X., Han Y., Leung V., Niyato D., Yan X., Chen X., Convergence of Edge Computing and Deep Learning: A Comprehensive Survey, IEEE, 2019.