Iraqi Journal of Science, 2024, Vol. 65, No. 12, pp: 7237-7250
DOI: 10.24996/ijs.2024.65.12.36

ISSN: 0067-2904

Performance Improvement for Weather Wireless Sensor Network by Mixed-Byte Compression Scheme

Saif Al-Alak
Department of Computer Science, College of Science for Women, University of Babylon, Babylon, Iraq

Received: 16/6/2023 Accepted: 7/11/2023 Published: 30/12/2024

Abstract

A Weather Wireless Sensor Network (WWSN) has many limitations related to node throughput (NT), node power (P), and node energy (E). To address the limitations of WWSN, a mixed-byte LZW scheme is proposed. Mixed-Byte reduces the size of the sensed data by merging every two readings of data from the sensor into a single byte. The merged bytes would be highly frequent, which made them appropriate for compression by the Lempel Ziv Welsh (LZW) scheme. The result shows that the proposed Mixed-Byte-LZW gives the best compression ratio (CR) (0.01) for humidity and the best saving compressing rate (SR) (99%) for humidity and pressure. The simulated result shows that the Mixed-Byte-LZW scheme gives the best NT (580 Kbps) for humidity, the highest power saving (PS) (15.98 Kw) for pressure, and the lowest E (0.39 Kj) for transferring humidity data.

Keywords: Data Compression, Node Energy, Node Throughput, Power Consumption, Weather Wireless Sensor Network.

تحسين اداء شبكات المتحسسات اللاسلكية للطقس باستعمال طريقة البايت المدمج للضغط

سيف محمود العلاق
قسم علوم الحاسوب، كلية العلوم للبنات، جامعة بابل، بابل، العراق

الخلاصة

تحتوي شبكة استشعار الطقس اللاسلكية على العديد من القيود المتعلقة بإنتاجية وقوة وطاقة العقدة. لمعالجة متطلبات شبكة استشعار الطقس اللاسلكية تم اقتراح طريقة البايت المدمج مع طريقة LZW. الطريقة المقترحة تعتمد على دمج كل قراءتين متتاليتين من بيانات المتحسس في بايت واحد وذلك من خلال حساب الفرق بينهما. ان تكرار الفروقات بين القراءات المتتالية بشكل كبير جعل امكانية ضغطها بطريقة LZW. اظهرت النتائج المحسوبة في هذا العمل ان الطريقة المقترحة تعطي افضل نسبة ضغط بيانات لمقياس الرطوبة (0,01) وافضل نسبة حفظ للبيانات المضغوطة هي (99%) لبيانات الضغط الجوي والرطوبة. افضل انتاجية كانت (580كبت\ثا) لأرسال بيانات الرطوبة و اعلى حفظ للقوة (15,98كيلو واط) في ارسال بيانات الضغط الجوي المقروءة واقل طاقة (0,39 كيلو جول) لأرسال بيانات الرطوبة.

[*]Email: saif.mahmood@uobabylon.edu.iq

1. Introduction

This paper proposes a scheme that reduces the size of the sensed weather data by merging any two sequenced readings of data from the sensor into a single byte. The merged bytes would be highly frequent, which made them available for compression by the Lempel Ziv Welsh (LZW) [1] scheme. The computed result pointed to the ability of using the proposed scheme (Mixed-Byte-LZW) to improve the performance and energy of WWSN better than previous work (DVM-LZW). The main contribution of the proposed work is to increase the lifespan of WWSN.

2. Related Work

Many researchers proposed variant ways to decrease the size of transmitted data, which would improve the performance and energy consumption of the network. Moreover, for WWSN, a Data Value Minimizing (DVM) scheme is proposed by Zahra and Saif [1] that reduces the sensed weather data by keeping the amount of incremental reading data instead of keeping the whole real data. The DVM scheme aims to improve WWSN's throughput and energy by minimizing the transmitted data by LZW. A scheme by Sawalha and Al-Naymat [2] counts similar successive data of WWSN, then sends the data with its own number of occurrences to minimize the amount of transmitted data, which leads to improved WWSN performance and energy. In [3], Ibrahim et al. proposed a scheme to optimize the data size of WWSN. The scheme uses either data aggregation, compression, or prediction techniques to improve the node's performance and energy based on the state of the sensor. In [4], Gao et al. proposed a scheme that uses the same idea of Huffman as a base to compress WSN data for energy improvement. In [5], Liang and Li proposed a scheme called S-LEC to work on both smooth WSN data and dynamic WSN data. It focuses on exploiting temporal information that remains in the residue series. Kolo et al. [6] proposed a scheme that breaks data into blocks and then deals with each block separately. It depends on multiple code options for compressing WSN data. Xue et al. [7] introduce an improved Huffman for environment data compression that utilizes a multi-scale wavelet for compressing data at the wireless channel level. Maulunida and Solichin [8] proposed a modification to the dictionary of the LZW scheme to improve the compression ratio. The proposed modification is using variable-length code in a dictionary instead of fixed-length code. In [9], Alalak et al. modified the indexes of codes in a dictionary using the LZW scheme to improve the data compression ratio. The index is represented by its exact size. The size of an index in a dictionary is recognized by its class. Indexes are classified according to their size.

3. Theoretical Background

3.1. WSN Applications

The WSN is deployed in many life applications because it consists of small sensors that are connected by a wireless network. The size of sensors and the nature of connectivity among them enable people to use this type of network in many fields like environment, military, structural monitoring, health, and transposition. WSN is appropriate for narrow, deep, sharp, polluted, and non-accessible places. The WWSN helps researchers and weather monitors collect accurate data on weather for many different environments, like forests, deserts, oceans, mountains, etc. [10] [11].

3.2. Sensor Node's Design Issues

The WSN faced many challenges related to its design during its deployment. They are fault tolerance, scalability, production cost, sensor network topology, and power consumption. Moreover, the use of nodes in a wild environment has many reasons that would lead to the nodes of the network failing, such as hardware and physical damage and battery halts. The biggest challenge is transmitting data over a wireless channel that consumes and exhausts the sensor node's energy. However, many techniques are implemented to increase the life of the sensor node by reducing the energy consumed for its activities. Furthermore, data compression, aggregation, and prediction are reducing the amount of transmitted data, which means keeping batteries longer to increase node life [12].

3.3. Structure of Sensor Node

The standard structure for a sensor node includes four main parts (units), which are the sensor, microprocessor, transmitter, and power supply. The other parts depend on the function of the node sensor, for example, position acquisition and mobilization. Moreover, the sensor unit is responsible for acquiring data from outside and then converting the analog signal of the sensor into a digital signal to be understood inside the sensor node. The processor unit is dealing with a small amount of memory to implement the major tasks of the sensor node through sensing and passing data to other nodes. A transmitter unit is responsible for sending and receiving messages from and to other nodes in the network via a communication channel. A power unit includes a battery to provide power to the sensor node. The important issue with wireless sensor nodes is to keep them alive as long as possible by consuming power efficiently. Many algorithms are implemented in WSN to minimize the consumption of power for implementing sensor nodes' tasks [13].

3.4. Structure of WSN

The WSN is structured into three main topologies, which are star, mesh, and hybrid star-mesh. Furthermore, the star topology connects a single base station to multiple sensor nodes via wireless communication. This type of topology allows the base station to transmit messages with sensor nodes; however, the sensor nodes are prohibited from transmitting messages with each other. This type of topology is simple, and it provides low power consumption and low latency for message transmission between the sensor node and base station. The sensor nodes should be in communication with the base station [14].

The second structure is mesh topology, which communicates base station and sensor nodes to each other in the network, enabling each node to transmit messages to other nodes. When the sensor node lies out of range of communication with the base station, the node sends a message to its neighbor node to pass it to the base station, which is called multi-hop. This type of topology supports network scalability and data redundancy. Moreover, the topology allows adding many other nodes to the network that are lying out of base station range. The base station would receive messages through more than one route to overcome the fault route. However, mesh topology consumes more power than star topology, especially in the multi-hop nodes, which decreases its life.

The third type of topology is a hybrid star-mesh network, which mixes star and mesh topologies. It includes the advantages of both star and mesh topologies. Furthermore, some nodes would transmit messages with low power, and other nodes would forward messages. The forwarding of messages from low-power nodes to other nodes would consume node power, so this type of node needs to be supported by a high-power battery [15].

3.5. WSN Data Compression

Data compression is one of the popular methods in WSN to optimize the size of transmitted messages. Data compression is an operation that runs a process (compressor) that converts the source data into its target data, where the source data is larger than the target data. Furthermore, the compressor is coding source data in a manner that enables other processes (the decompressor) to retrieve the source data from the target by decoding target data into source data. The way or procedure in which the coding operation is done is called the compression algorithm.

There are two main types of compressing algorithms: lossy and lossless. The lossy compressing algorithms are eliminating unimportant data. However, lossless compressing algorithms retrieve the original source data without losing a bit after decompressing the target. Many algorithms are considered by researchers to reduce the size of transmitted data over a network. The reduction of data size leads to improvements in many WSN metrics, for example, WSN performance and WSN life. [12] [16] Moreover, the most popular lossless data compression algorithm is LZW, which falls under the dictionary-based data compression family. It appeared in 1984 as an upgraded version of the LZ78, which was found in 1978. The idea of the algorithm is simple. Furthermore, each 8-bit character is encoded into a single token in the dictionary for all possible 256 characters. For each new input, one token is added to the dictionary after creating a new string by concatenating the new input with the old string. If the string from concatenation exists in the dictionary, then it is kept as an old string and another new input is read as explained in the encoder algorithm [17].
Encoder Algorithm:

```
1.  Begin
2.  For i← 0 to 255
3.  Dictionary[i] ← Dictionary_code (i)
4.  End For
5.  Prev ← empty
6.  While true
7.  Begin
8.  Input ← read (1-byte)
9.  If (Input + Prev) is not in Dictionary
10. Output (Dictionary_code (Prev))
11. Dictionary[i++] ← Dictionary_code (Input + Prev)
12. Prev ← Input
13. Else Prev ← Input + Prev
14. End If
15.  End While
16. End
```

On the decoder side, the work is reversed. Furthermore, after initializing the dictionary with 256 codes, the input is decoded from the dictionary to get its original string. A new string is added to the dictionary after each input. The new string is computed by concatenating the new input with the previous string, as explained in the decoder algorithm [17].
Decoder Algorithm:

```
1. Begin
2. For i←0 to 255
3. Dictionary[i] ← Dictionary_code (i)
4. End For
5. Prev  ← read (data)
6. Output(Dictionary_decode(Prev))
7. C ← empty
8.   While true
9.   Begin
10.  Input ← read (data)
```

```
11. If (Input) is not in Dictionary
12.  Then S ← Dictionary_decode(Prev) + C
13. Else  S ← Dictionary_decode(Input)
14. Output(S)
15. C ← first_character (S)
16. Dictionary[i++] ← Prev + C
17. Prev ← Input
18. End While
19. End
```

4. Proposed Work (Mixed-Byte-LZW)

The proposed work is called Mixed-Byte-LZW. Its idea is based on the nature of the weather data gathered by the sensor node. A sensor node senses data (d_min_1, d_min_2, …) of weather every minute, then sends the collected data for one hour to the base station. After one minute, the sensed weather data (d_min_i) is changed by decreasing or increasing its value. Furthermore, the amount of difference between any two sequenced sensed weather data points (d_min_i, d_min_i+1) is between -7 and +7 for each minute because the weather is changed slowly. For example, if the sensed temperature of the weather at two consecutive minutes is represented as a pair (temperature, minute) = (33, 1) and (34, 2), it means the weather temperature is 33° at the first minute and 34° after one minute. Then the difference between two sensed temperatures is +1. Moreover, this work suggests a scheme to store the sensed data in a novel way, where the size of the stored data would be less than the size of the original data. The scheme supposes that the sensed data (d_min_i, min_i) will be computed relative to the previous one (d_min_i-1, min_i-1) as shown in Eq. (1). In the beginning, the first sensed data (d_min_1) is stored without change, then Diff1 and Diff2 are computed, which are the differences between the first and second sensed data as shown in Eq. (2) and the difference between the second and third sensed data as shown in Eq. (3). The new value Wrd (with one byte size) is used to replace d_min_2 and d_min_3. Wrd is computed by mixing Diff_1 and Diff_2, as shown in Eq. (4). The magnitudes of Diff1 and Diff2 must be between -7 and +7 because the change in the weather is very slow at one minute. A flow chart in Figure 1 shows how the proposed scheme (Mixed_Byte) works for one hour (36 readings).

$$d_i = d_{i-1} \pm diff \qquad (1)$$

Where $d_{\_i}$: reading data in minute i, $d_{\_i\text{-}1}$: reading data in minute i-1.

$$diff_1 = d_1 - d_2 \qquad (2)$$
$$diff_2 = d_2 - d_3 \qquad (3)$$
$$Wrd = diff_1 \cup diff_2 \qquad (4)$$

In more detail, it is possible to code the values of Diff_1 and Diff_2 into one byte, as illustrated in Figure 2. As explained above, the value of (Diff_1 and Diff_2) is from -7 to +7, so that the value of Diff_1 and Diff_2 would be concatenated into one byte. The value of Diff_1 is considered the left part (4 bits) and Diff_2 the right part (4 bits) of the code byte. The code byte consists of two parts (left and right), which are computed by concatenating the code of Diff_1 (4 bits left part) and Diff_2 (4 bits right part) as shown in Figure 2-a. The computing of a left code (L) from Diff_1 or a right code (R) from Diff_2, where there is a unique code for each value of Diff_1 or Diff_2, Concatenating the left and right parts results in a hexadecimal code byte (Wrd), which is illustrated in Figure 2-b. Some examples of coding Diff_1 and Diff_2 are shown in Figure 2-c. The above demonstration is applicable to other readings from the sensor.

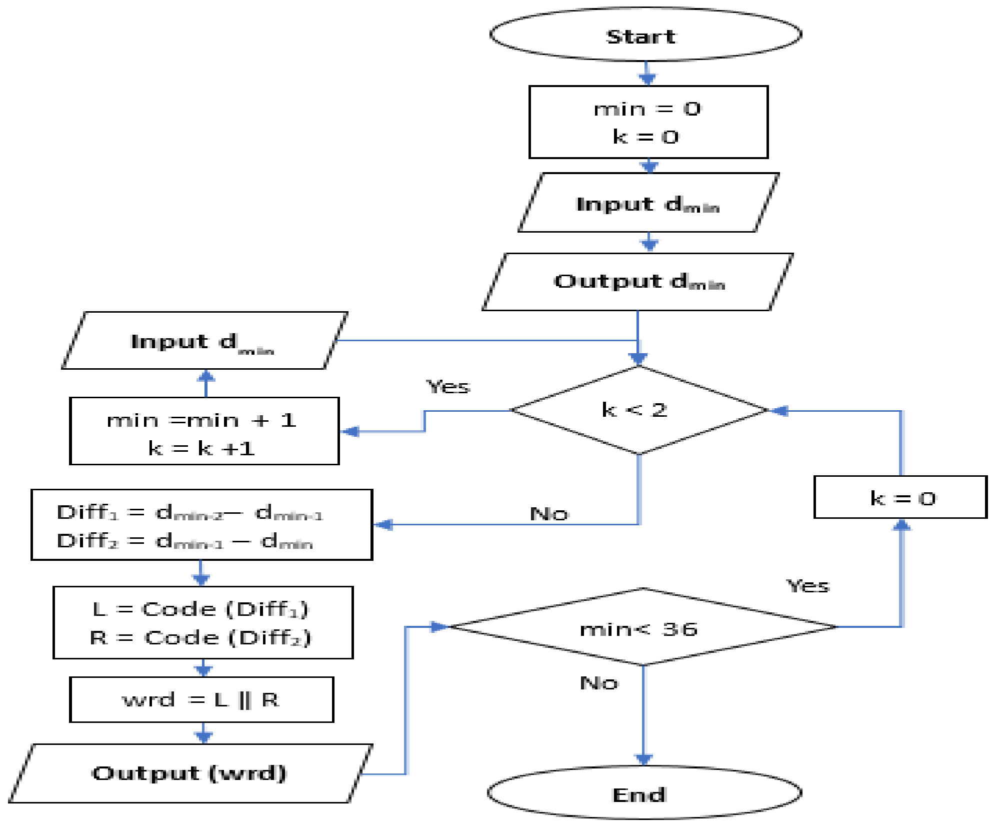

Figure 1: Flow Chart of Mixed-Byte

Input: bytes from Sensor
Output: mixed bytes

```
Begin
set i←0
data_i ←read from sensor
save (data_i)
repeat (18 times)
repeat (2 times)
    i←i+1
    data_i←read from sensor
End repeat (2 times)
Diff_1← data_i-2– data_i-1
Diff_2← data_i-1 – data_i
  Left ← Code (Diff_1), Right ← Code (Diff_2)
Wrd←Union (Left, Right)
  save (Wrd)
End repeat (18 times)
End
```

(a) Proposed Algorithm

$Diff_{\_1,2}$	Hex
0	0x0
1	0x1
2	0x2
3	0x3
4	0x4
5	0x5
6	0x6
7	0x7
-1	0x9
-2	0xa
-3	0xb
-4	0xc
-5	0xd
-6	0xe
-7	0xf

(b): Coding of Diff_$_1$ anddiff_$_2$

Diff_1	Diff_2	Code in Hex
5	-3	0x5b
0	1	0x01
-7	-1	0xf9
4	2	0x42

(c): Examples

Figure 2: Mixed-Byte Scheme

5. Methodology of the Research

To measure the impact of the Mixed-Byte-LZW on the WSN, two directions are examined (data reduction and network performance). The data set for examining the Mixed-Byte-LZW is weather data, which consists of readings from eight sensors. The readings of the sensors are: apparent temperature, dew point, humidity, pressure, temperature, visibility, wind bearing, and wind speed. The data set readings are a time span of one minute for one year of house appliances in the weather conditions of that special area. The data set is taken from a web site [18].

5.1. Data Size Reduction

In this part, two experiments were accomplished. The first experiment includes three tests. The first test compares the size of the original data to the data from DVM and mixed-byte. The second test compares the size of LZW compressed data to data compressed by DVM-LZW and Mixed-Byte-LZW. The third test computes a compression ratio (CR) for compressed data for LZW, DVM-LZW, and Mixed-Byte-LZW. The fourth test computes a saving rate (SR) for LZW, DVM-LZW, and Mixed-Byte-LZW. The data in the first experiment includes readings from eight sensors for one, two, three, four, and five hours. In the second experiment, the data set includes the readings from eight sensors for 1 year (503911 minutes). The second experiment examines the same three tests from the first experiment with different data sizes. The metrics for data reduction parts are CR and SP. They are computed by Eqs. (5) and (6), respectively.

$$CR = \frac{DA}{DB} \tag{5}$$

Where CR: compression ratio, DA: data size after compression, DB: data size before compression

$$SR = 1 - CR \times 100\% \tag{6}$$

Where SR: Saving rate

When the CR is 1, it means that the compressing scheme performs zero-bit data compression. On the other side, when the value of CR is decreased, then the compressing scheme is going better. The increasing SP rate means the compressing scheme is going better, and vice versa.

5.2. Network Performance

The measured metrics for wireless sensor nodes include node throughput (NT), node power saving (PS), and energy (E) for data transmission, which are computed by Equations 7, 8, 9, and 10, respectively. Two experiments were implemented, each with a different size of data set. The data set of the first experiment contains readings of weather for 5 hours, and the second experiment contains readings of weather for one year. Each experiment has three tests. The network is simulated by considering XbeeS2C [19] with the configuration from Table 1.

$$NT = \frac{D}{T} \tag{7}$$

Where NT: Node throughput, D: Transferred Data, T: Time

$$SP = P_{OD} - P_{CD} \tag{8}$$

Where SP: Node saving power, P_{OD}: Original data transferred power, P_{CD}: Compressed data transferred power

$$P = I \times V \tag{9}$$

Where P: Data transferred power, I: Current, V: Voltage

$$E = I^2 \times R_{sh} \times T \tag{10}$$

Where E: Data transferred Energy, I: Current, Rsh: shunt resistor, T: Data transferred time

Table 1: Sensor Node Configuration

Feature	Value
Band	2.4 GHz
Channels	16
Tx Current	33 mA
Rx Current	28 mA
Distance	30 m
Environment	outdoor
Baud rate	115200
RTL	0.079 s
VDC	3.3
Rsh	9 **Ω**

6. Result and Discussion

The result of the experiments includes two parts: the first is related to data size reduction, and the second is related to network performance.

6.1. Result of Data Size Reduction

In the data size reduction part, the result of the first test from the first experiment is illustrated in Figure 3. The size of sensed data (weather for one hour) is reduced by using a mixed-byte scheme (the proposed scheme) better than the DVM scheme (the previous work). For the data reading during two, three, four, and five hours, the size of sensed data is more reduced by a mixed-byte model than a DVM.

In the second test of the first experiment, the impact of mixed-byte-LZW on the size of sensed data is measured in Figure 4. When the Mixed-Byte-LZW scheme is used, the size of the original sensed data is reduced more (less size) than in DVM-LZW (previous work) and LZW. Normally, when the LZW scheme is used to compress data, the size of the compressed data is based on the nature of the original data because LZW depends on the number of frequent strings. The Mixed-Byte-LZW is going to reduce the size of the original data and increase the frequency of strings that reduce the compressed data. The third test of the first experiment computes the compression ratio (CR) of LZW, DVM-LZW, and Mixed-Byte-LZW. Figure 5 illustrates that the CR for the sensed data in 1 hour is 0.29 with LZW, 0.2 with DVM-LZW, and 0.12 with Mixed-Byte-LZW. The CR of Mixed-Byte-LZW is better (less) than DVM-LZW and LZW for compressed sensed data (2...4) hours. In 5 hours of sensed data, the CR decreased from 0.22 (LZW) to 0.13 (DVM-LZW) and then to 0.08 (Mixed-ByteLZW).

In the second test of the first experiment, the impact of mixed-byte-LZW on the size of

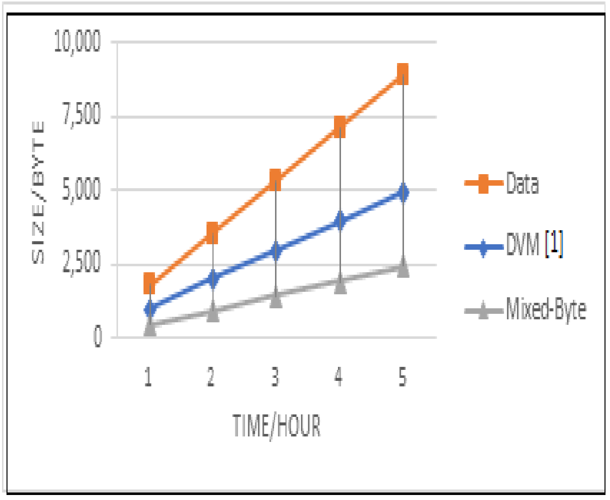

Figure 3: One to Five Hours Sensed Data

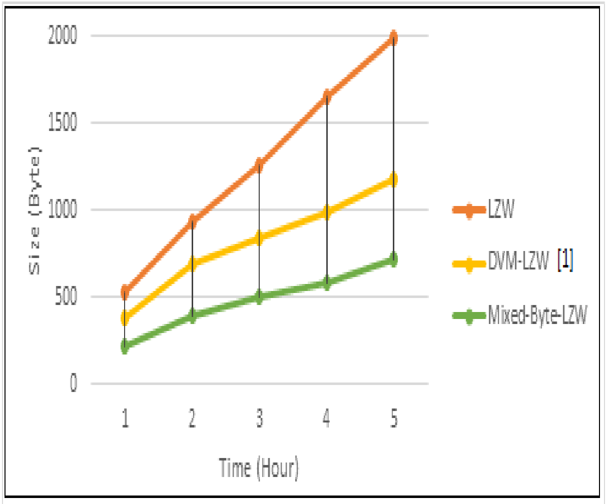

Figure 4: Size of One to Five Hours

sensed data is measured in Figure 4. When the Mixed-Byte-LZW scheme is used, the size of the original sensed data is reduced more (less size) than in DVM-LZW (previous work) and LZW. Normally, when the LZW scheme is used to compress data, the size of the compressed data is based on the nature of the original data because LZW depends on the number of frequent strings. The Mixed-Byte-LZW is going to reduce the size of the original data and increase the frequency of strings that reduce the compressed data. The third test of the first experiment computes the compression ratio (CR) of LZW, DVM-LZW, and Mixed-Byte-LZW. Figure 5 illustrates that the CR for the sensed data in 1 hour is 0.29 with LZW, 0.2 with

DVM-LZW, and 0.12 with Mixed-Byte-LZW. The CR of Mixed-Byte-LZW is better (less) than DVM-LZW and LZW for compressed sensed data (2...4) hours. In 5 hours of sensed data, the CR decreased from 0.22 (LZW) to 0.13 (DVM-LZW) and then to 0.08 (Mixed-ByteLZW).

In the second test of the first experiment, the impact of mixed-byte-LZW on the size of sensed data is measured in Figure 4. When the Mixed-Byte-LZW scheme is used, the size of the original sensed data is reduced more (less size) than in DVM-LZW (previous work) and LZW. Normally, when the LZW scheme is used to compress data, the size of the compressed data is based on the nature of the original data because LZW depends on the

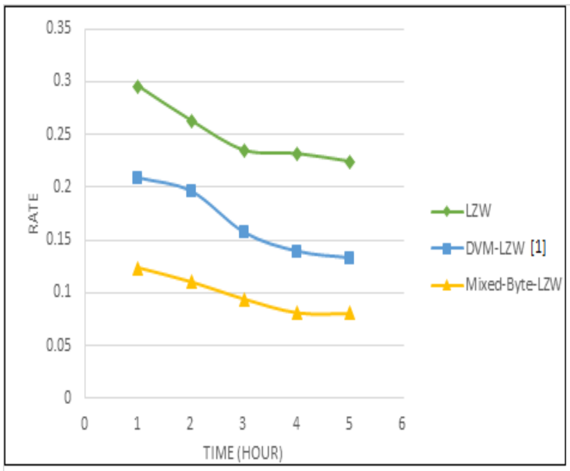

Figure 5: CR of One to Five Hours Sensed Data

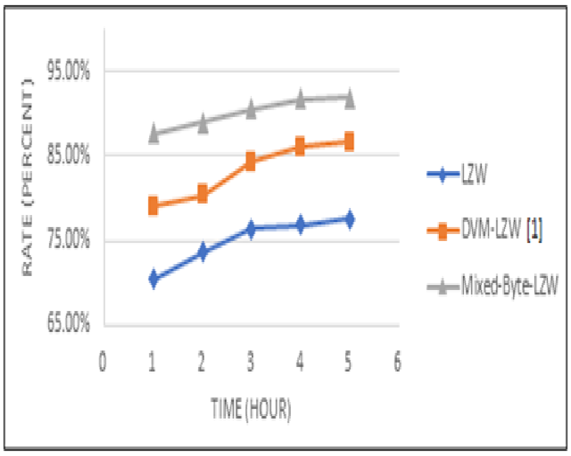

Figure 6: SR of One to Five Hours Compressed Sensed Data

number of frequent strings. The Mixed-Byte-LZW is going to reduce the size of the original data and increase the frequency of strings that reduce the compressed data. The third test of the first experiment computes the compression ratio (CR) of LZW, DVM-LZW, and Mixed-Byte-LZW. Figure 5 illustrates that the CR for the sensed data in 1 hour is 0.29 with LZW, 0.2 with DVM-LZW, and 0.12 with Mixed-Byte-LZW. The CR of Mixed-Byte-LZW is better (less) than DVM-LZW and LZW for compressed sensed data (2...4) hours. In 5 hours of sensed data, the CR decreased from 0.22 (LZW) to 0.13 (DVM-LZW) and then to 0.08 (Mixed-ByteLZW).

The fourth test computes SR when LZW, DVM-LZW, and Mixed-Byte-LZW are used to compress sensed data. The result is illustrated in Figure 6, which shows that the SR is 70.41%, 79.09%, and 87.65% when LZW, DVM-LZW, and Mixed-Byte-LZW are used to compress one-hour sensed data, respectively. In comparison to LZW and DVM-LZW, the In the second experiment, the tests were applied to the data set for about 1 year, and each item was tested separately. The result of the first test is illustrated in Figure 7-a, which measures the size of the data (8-item weather) after processing it by Mixed-Byte and by DVM. Figure 7-a shows that the size of eight different data sets is decreased after processing them by Mixed-Byte less than by DVM. In the second test, the size of the data is measured after compressing it with LZW, DVM-LZW, and Mixed-Byte-LZW. Figure 7-b shows that the size of compressed data by Mixed-Byte-LZW is less than LZW and DVM-LZW. In Figure 7-b, the implementation of DVM-LZW to the 7th (windBearing) item leads to a change in the nature of data that makes LZW better than DVM-LZW (previous work); however, Mixed-Byte-LZW gives a better result than DVM-LZW and LZW.

The third test of the second experiment measures CR. Figure 8 illustrates that the value for CR of compressing eight weather items is decreased when the data is compressed by Mixed-Byte-LZW more than by DVM-LZW, which means that Mixed-Byte-LZW is better than DVM-LZW. The fourth test measures SR when data is compressed by Mixed-Byte-LZW and by DVM-LZW. The use of Mixed-Byte-LZW increases the SR rate compared to DVM-LZW when compressing the eight weather items, as illustrated in Figure 9.

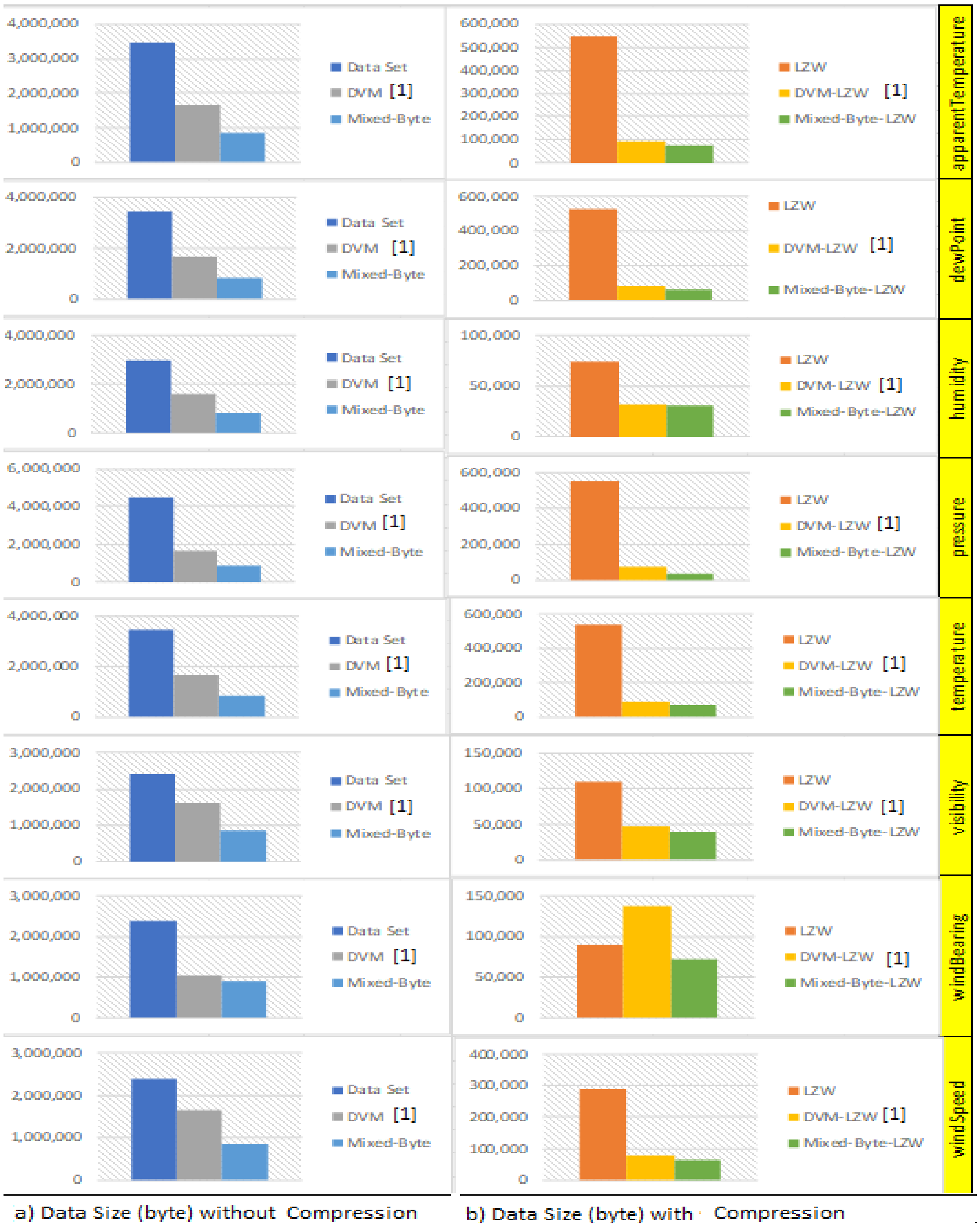

Figure 7: Data Size of 8-Item Weather Before and After Compression

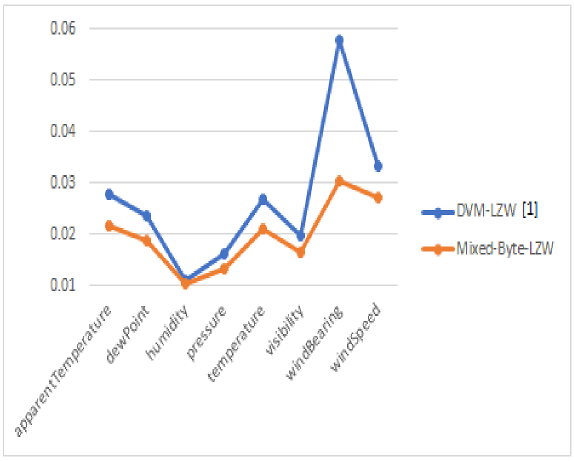

Figure 8: CR for Compressed 8 Weather Items

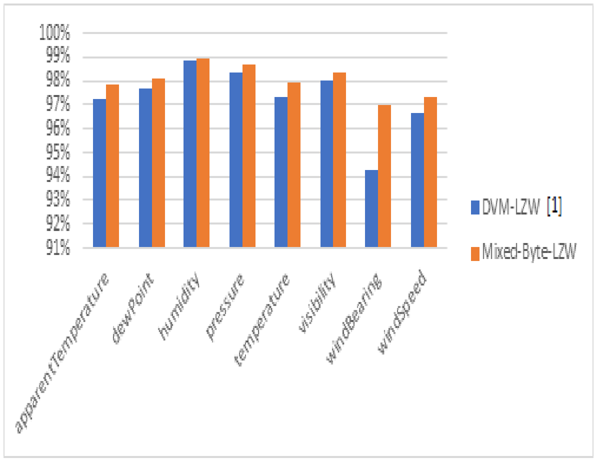

Figure 9: SR of LZW for Eight Weather Items Compression

6.2. Result of Network Performance Part

In this part, network performance metrics are measured. In the first experiment, the measurement is applied to sensed data during 1, 2,... 5 hours. The measured result proves that the proposed scheme, Mixed-Byte-LZW, increases NT and SP and decreases E for data transfer. Furthermore, it is useful to decrease the size of sensed data to increase NT by decreasing the transfer time for sensed data. Figure 10 shows that the use of Mixed-Byte-LZW increases NT higher than using DVM-LZW and LZW for compressing various size-sensed data. Figure 11 illustrates that Mixed-Byte-LZW increases the saving power for transferring sensed data compared to DVM-LZW and LZW. Figure 12 shows that E for data transmission with Mixed-Byte-LZW is less than with DVM-LZW and LZW. In the second experiment, the measurement is applied to a separate data set of eight weather items that is sensed over a period of one year to evaluate the impact of the proposed scheme on a huge amount of data. The result proves that Mixed-Byte-LZW provides a better result than DVM-LZW and LZW in terms of NT, SP, and E. Furthermore, when the node uses Mix-Byte-LZW to reduce its data size, its throughput is higher than when using DVM-LZW and LZW. The Mix-Byte-LZW gives the highest NT improvement for humidity data because its values are more frequent than others. However, Mix-Byte-LZW gives the lowest NT improvement for wind bearings, as illustrated in Figure 13. The PS of the node for data transfer using Mix-Byte-LZW is highest for the pressure data set and lowest for wind bearing, as illustrated in Figure 14. The consumed E for transferring data of the node that uses Mix-Byte-LZW is lower than that of the node that uses DVM-LZW and LZW. When the node uses Mix-Byte-LZW, the lowest E is measured for humidity data and the highest for pressure, as illustrated in Figure 15.

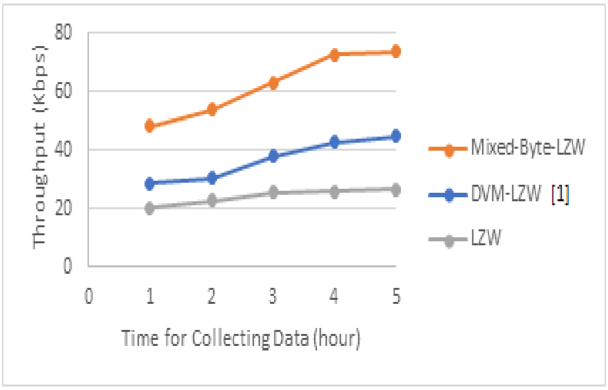

Figure 10: NT for various Sensing Time

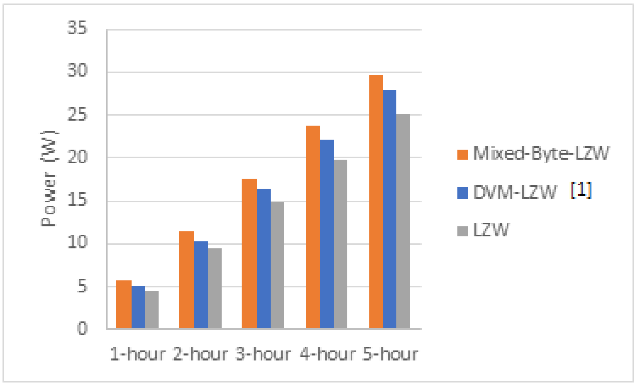

Figure 11: Node SP for Variants Sensing Time

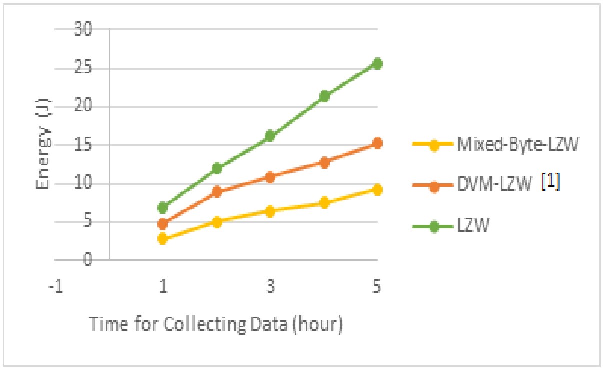

Figure 12: Node Energy Consumption for Data Transferring

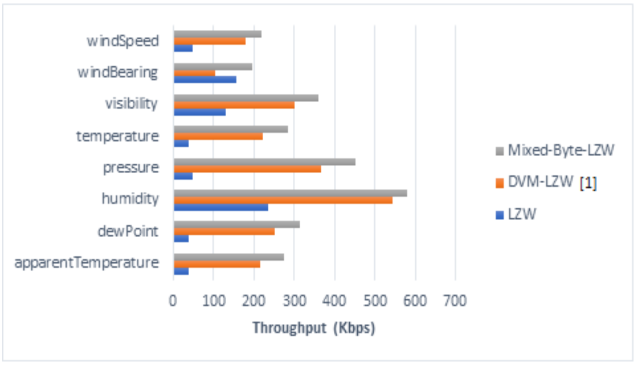

Figure 13: NT for Transferring Data of 8 Weather Items Separately

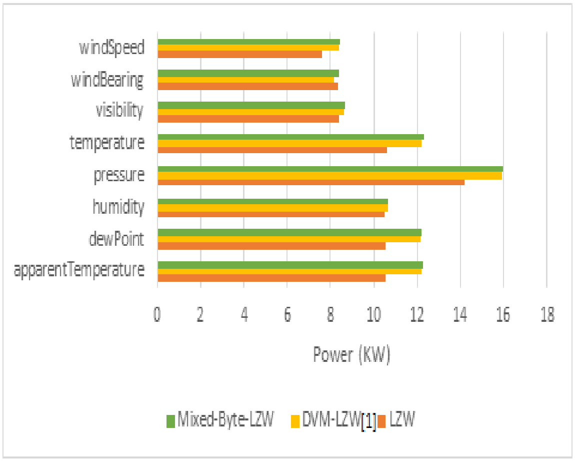

Figure 14: PS for Transferring Data of 8 Weather Items Separately

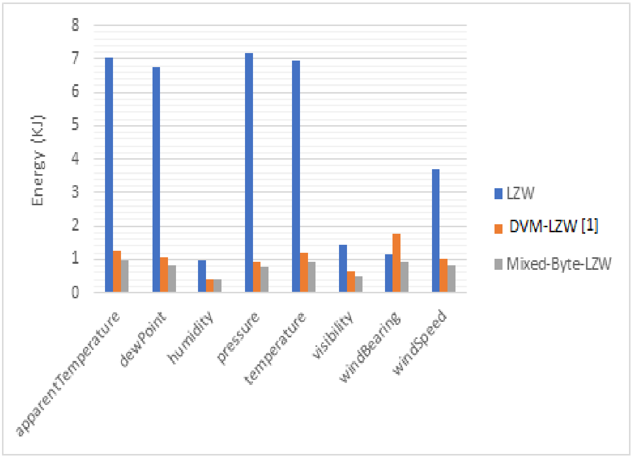

Figure 15: Energy Consumption for Transferring Data of 8 Weather Items Separately

7. Conclusions

Mixed-Byte-LZW is a good scheme to optimize the size of the transmitted messages over WWSN. Compared to other schemes, it benefits by making most of the contents of the message frequent, which enables it to implement a high compression ratio with LZW. The mixed-byte LZW depends on the nature of the data in its work. It exploits the slowness of changes in weather to save two weather readings into a single byte. The Mixed-Byte-LZW scheme improves the performance of WWSN in terms of network throughput, sensor power savings, and energy consumption; it also improves compression ratio and saves compressing.

References

[1] M. Zahra and A. Saif, "A Developed Compression Scheme to Optimize Data Transmission in Wireless Sensor Networks," *Iraqi Journal of Science*, vol. 64, no. 3, pp. 1463–1476, Mar. 2023. Doi: https://doi.org/10.24996/ijs.2023.64.3.35

[2] S. Sawalha and G. Al-Naymat, "Internet of things data compression based on successive data grouping," *Turkish J. Electr. Eng. Comput. Sci.*, vol. 29, no. 1, pp. 32-45, Jan. 2021. Doi: https://doi.org/10.3906/elk-2003-114

[3] M. Ibrahim, H. Harb, A. Mansour, A. Nasser, and C. Osswald, "All-in-one: Toward hybrid data collection and energy saving mechanism in sensing-based IoT applications," *Peer-to-Peer Networking and Applications*, vol. 14, no. 5, pp. 1154 - 1173, May 2021. Doi: https://doi.org/10.1007/s12083-021-01095-5

[4] L. Gao, B. Zhang, T. Jiang, and Z. He, "Application of a Lossless Compression Algorithm Based on Dynamic Huffman in Smart Ammunition Wireless Sensor Network," *2022 2nd International Conference on Consumer Electronics and Computer Engineering (ICCECE)*, Guangzhou, China, pp. 643-647, Feb. 2022. Doi: 10.1109/ICCECE54139.2022.9712693

[5] Y. Liang and Y. Li, "An Efficient and Robust Data Compression Algorithm in Wireless Sensor Networks," *IEEE Communications Letters*, vol. 18, no. 3, pp. 439-442, Mar. 2014. Doi: 10.1109/LCOMM.2014.011214.132319.

[6] J. G. Kolo, S. A. Shanmugam, D. W. G. Lim, L.-M. Ang, and K. P. Seng, "An Adaptive Lossless Data Compression Scheme for Wireless Sensor Networks," *J. Sensors*, vol. 2012, no., pp. 1-20, Sep. 2012. Doi: https://doi.org/10.1155/2012/539638

[7] X. Xue, C. Wang, H. W. Ma, Q. H. Mao, X.-g. Cao, X. Zhang, *et al.*, "Self-Derived Wavelet Compression and Self Matching Reconstruction Algorithm for Environmental Data in Complex Space of Coal Mine Roadway," *Energies*, vol. 15, no. 20, pp. 7505-7510, Oct. 2022. Doi: https://doi.org/10.3390/en15207505.

[8] R. Maulunida and A. Solichin, "Optimization of LZW Compression Algorithm With Modification of Dictionary Formation," *Indonesian Journal of Computing and Cybernetics Systems*, vol. 12, no. 1, pp. 73-82, Jan. 2018. Doi: https://doi.org/10.22146/ijccs.28707

[9] S. M. K. Al-alak, I. Alwan, and A. A. Hussein, "Adapted LZW Protocol for ECG Data Compression," *Journal Of University Of Babylon for Pure and Applied Sciences*, vol. 25, no. 5, pp. 1618-1626, Nov. 2017.

[10] S. I. Jassim and S. W. Nourildean, "IEEE 802.15. 4 ZigBee-based wireless sensor network in medical application," *Iraqi Journal of Science,* vol. 53, no. 4, pp. 1055-1066, December 2012.

[11] K. G. Salim, S. M. K. Al-alak, and M. J. Jawad, "Improved image security in internet of thing (IoT) using multiple key AES," *Baghdad Sci J*, vol. 18, no. 2, pp. 04-17, Jan. 2021. Doi: https://doi.org/10.21123/bsj.2021.18.2.0417

[12] M. A. A. Salih and A. Mohammed, "Design and performance analysis of building monitoring system with wireless sensor networks," *Iraqi Journal of Science*, vol. 53, no. 4, pp. 1097-1102, Dec. 2012.

[13] M. N. Abbas, A. A. Bara'a, and N. J. Kadhim, "Evolutionary based set covers algorithm with local refinement for power aware wireless sensor networks design," *Iraqi Journal of Science*, vol. 59, no. 4A, pp. 1959–1966, Oct. 2018.

[14] S. Hasan and A. Amer, "A Vehicle ID identification Architecture: A Parallel-Joining WSN Algorithm," *Iraqi Journal of Science*, Special Issue, pp. 267-270, Jan. 2021. Doi**:** https://doi.org/10.24996/ijs.2021.SI.1.37

[15] A. A. Bara'a and S. M. Hameed, "A genetic algorithm for minimum set covering problem in reliable and efficient wireless sensor networks," *Iraqi Journal of Science,* vol. 55, no. 1, pp. 224-240, Jan 2014.

[16] J. M. Kadhim, "Security of wireless sensor nodes," *Iraqi Journal of Science,* vol. 61, no. 7, pp. 1773–1780, Jul. 2020. Doi: https://doi.org/10.24996/ijs.2020.61.7.26

[17] D. Salomon, *Data compression: the complete reference*: Springer-Verlag London , 4th ed., Mar. 2007. Doi: https://doi.org/10.1007/978-1-84628-603-2

[18] Starter: Smart Home Dataset with a8895cb2-c," S. CHOUDHURY, Ed., 2019. Available: https://www.kaggle.com/saikatchoudhury/starter-smart-home-dataset-with-a8895cb2-c

[19] K. F. Haque, A. Abdelgawad, and K. Yelamarthi, "Comprehensive Performance Analysis of Zigbee Communication: An Experimental Approach with XBee S2C Module," *Sensors*, vol. 22, no. 9, pp. 3245-3265, Apr. 2022. Doi: https://doi.org/10.3390/s22093245