Research Article

A Lossless Compression Approach Based on Delta Encoding and T-RLE in WSNs

Abdeldjalil Saidani, Xiang Jianwen , and Deloula Mansouri

School of Computer Science and Technology, Wuhan University of Technology, Wuhan 430070, China

Correspondence should be addressed to Xiang Jianwen; jwxjiang@whut.edu.cn

Received 13 March 2020; Revised 25 June 2020; Accepted 28 August 2020; Published 21 September 2020

Academic Editor: Tony T. Luo

Copyright © 2020 Abdeldjalil Saidani et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

The sending/receiving of data (data communication) is the most power consuming in wireless sensor networks (WSN) since the sensor nodes are depending on batteries not generally rechargeable characterized by limited capacity. Data compression is among the techniques that can help to reduce the amount of the exchanged data between wireless sensor nodes resulting in power saving. Nevertheless, there is a lack of effective methods to improve the efficiency of data compression algorithms and to increase nodes' energy efficiency. In this paper, we proposed a novel lossless compression approach based on delta encoding and two occurrences character solving (T-RLE) algorithms. T-RLE is an optimization of the RLE algorithm, which aims to improve the compression ratio. This method will lead to less storage cost and less bandwidth to transmit the data, which positively affects the sensor nodes' lifetime and the network lifetime in general. We used real deployment data (temperature and humidity) from the sensor scope project to evaluate the performance of our approach. The results showed a significant improvement compared with some traditional algorithms.

1. Introduction

Wireless technologies offer new perspectives in the field of telecommunications and computer networks. As a result of the progress made, a new type of ad hoc network has emerged, which is called wireless sensor networks [1]. These are networks without fixed infrastructure; they can be deployed quickly in sensitive areas and/or difficult to access. Their mission is most often to monitor a space, to take regular measurements, and to raise alarms to specific nodes of the network, known as collecting nodes (sink), capable of relaying information on a large scale to the remote control center [2]. The intrinsic characteristics of this new generation of microsensors (tiny, processing capacity, wireless communication, low cost, diversity of sensors (optical, thermal, multimedia, etc.)) have opened up new and varied application perspectives for sensor networks in many fields (military, home automation, environmental, etc.) [3]. However, they raise in the same proportions, many research problems as much by the potential applications that they suggest by the

various constraints that they impose. It is widely accepted that the techniques and approaches developed in both wired and wireless networks are not directly “transferable” in the field of sensor networks [4]. Ultimately, WSNs remain an open domain for the scientific community, and it is almost at all levels: autoconfiguration, localization, coverage, deployment, communication, dynamic topology, collection, and dissemination of data, queries, and processing. Most of the works on sensor networks are aimed at reducing or at least rationalizing energy consumption [5]. Indeed, the networks of sensors are intended, in most cases to record information in a hostile environment or challenging to access without any human intervention. Therefore, it is difficult to find another source of energy than batteries [6]. It is for this reason they are considered as autonomous devices. Their lifetime is equal to the life of their battery, which indicates that energy consumption is a critical constraint in sensor networks since energy is considered a valuable resource [7]. The creation of several methods in energy conservation was an outgrowth of controlling the energy

consumption of sensor node components, which include data compression [8].

The sensor nodes in the WSNs consume energy during the acquisition, processing, and transmission phases. However, the power spent by a node in the communication module for transmitting and receiving data is higher than that required for processing. An important approach to conserving energy and maximizing network lifetime is the use of efficient data compression systems. Data compression methods reduce the size of the data before transmitting it over the wireless channel, which result in a reduction in total power consumption. These energy savings due to compression translate directly by extending the lifetime of nodes and the network in general. Saving a byte of data through data compression has been shown to provide between four thousand (using Chipcon CC2420 transceiver) and two million (using MaxStream XTend transceiver) cycles of computation [9].

In general, the WSNs are used to monitor natural phenomena, which constitutes the correlation between each consecutive observation of the sensor nodes, especially the weather measurements (temperature and humidity). In this work, we will present an optimized lossless compression approach based on delta encoding and modified RLE algorithms (T-RLE). The main idea about this method is to increase the similarity of the data by applying delta encoding and compress it by using the T-RLE algorithm. We organized the rest of the paper as follows: we present some related works in Section 2. For Section 3, it discusses the proposed method; concerning Section 4, we analyze the result. Lastly, we conclude the paper in Section 5.

2. Related Works

This section aims at presenting a state art of the works dealing with the conservation of energy using compression algorithms in this particular type of network. Lossless compression is divided into three techniques, firstly, dictionary-based like Lempel–Ziv–Welch (LZ77) and Lempel–Ziv 77 (LZW). The LZ77 algorithm uses a horizon to find the matches that will help with compression. However, this method has been improved with the arrival of the LZ78 algorithm, which uses an auxiliary data structure to store the repetitive sequence; this structure calls a dictionary. This time to represent a repeated sequence, we use the index of this sequence in the dictionary not the positions. Then, we will have tuples in the form $\langle \text{index}, \& \rangle$ where $\&$ represents the character that follows the repeated sequence [10]. LZW is another improvement of the LZ78 algorithm; it has eliminated the use of tuples; it only uses the indexes of sequences in the dictionary $\langle \text{index} \rangle$ [11]. On the other hand, LZW has some problems that affect the performance efficiency. It should introduce additional elements in the dictionary that can be used. In this situation, the dictionary can be in the full state, and the research within the dictionary consumes time, secondly, entropy-based such as Huffman encoding [12]. Huffman's method briefly is to find the number of occurrences of each sequence, then assign a shorter code to the most frequent sequences and a longer code for the less frequent sequences. Thirdly, the probabilistic-based, which is

the combination of the two earlier techniques as GZIP and Lempel–Ziv–Markov chain Algorithm (LZMA) [13]. Due to limited processing and storage resources of sensor nodes, only a few research papers have discussed the possibility of embedding lossless compression algorithms into WSNs [14]. The direct implementation of these algorithms was not applicable, and they must be adjusted to fit the requirements of WSNs [15]. For example, the adaptation of LZW for sensor networks, called S-LZW (LZW for sensor nodes), allowed the evaluation of a data compression algorithm based on a dictionary for WSNs. However, the construction of a data dictionary is a memory-intensive process, which is an obstacle to the use of such techniques on sensors with limited memory. Therefore, to adapt LZW to the sensors, a new morphology of this algorithm is articulated around three main correlated points which are the size limitation of the dictionary, the limitation of the size of the data to be compressed, and the procedures to follow when the dictionary is filled [16]. Marcelloni and Vecchio proposed an efficient lossless entropy compression (LEC) algorithm [17] in which exploits the high temporal correlation that exists between the consecutive readings gathered by the sensors. LEC is one of the earliest algorithms that deal with multiple data types. The application of the algorithm on real datasets showed good results, which reached over 70% in terms of the compression ratio. Still, it is matched by the imperfection of the energy saving. On the other hand, the algorithm has a shortcoming that can not adapt to the changing in the correlation of the sensor reading. It needs prior knowledge of the statistical characteristics of the data like entropy. These shortcomings were the motivation for the emergence of an adaptive lossless data compression (ALDC), which presented by Kolo et al. [18]. It gives several code options to achieve the optimal performance. But regarding the compression efficiency, it has a low performance comparing to LEC. They also introduced a lightweight algorithm called FELACS (fast and efficient lossless adaptive compression scheme) [9]. It bases on RICE coding to compress the block of data, and the adaptivity is added by giving the two-block size option and variable compression ratio. FELACS has high robustness against loss of data in addition to low memory usage. It also does a single sensing attribute which results in reduce collision and retransmissions. A lossless neighborhood indexing sequence (NIS) algorithm has been proposed for the data compression in WSN in [19]. NIS uses a traversing method based on 0's and 1's. This method gives two codewords for each character, then chooses the shortest one to present each character. The authors claim that it is highly robust to packet losses since every individual packet is decompressed independently.

The main common point between these related works that they used is the same real-world datasets to test their algorithms against, as we used it from our side too. Before applying their compression algorithms, they converted the physical measures of humidity and temperature into `raw_h` and `raw_t` by using the inverted versions of the conversion functions in the sensor scope project web site [20]. The only difference is that we have not converted the data to the raw data. We used it directly as it published the sensor scope project web site, as we did in our previous work [21]. This later

was built based on several datasets taken from real-world datasets from the sensor scope project. All the datasets are a float-point dataset. The main idea of this work is to compress the float part and the integer part separately, using RLE and Bzip2 for the integer part and Bzip2 only for the float part. The result performance shows a significant improvement compared with other algorithms.

Compared to the conventional compression approaches, we find the unique characters of temperature and humidity that the repetition of integer parts appears more frequently in the used datasets. The new approach aims at dealing with this kind of real datasets based on a modified RLE (T-RLE). T-RLE has the ability to deal with one and two character occurrences, which have no reduction in the size of the data when applying RLE. In this situation, the data compression becomes an overloaded problem which consumes more energy of the network nodes without benefits. The results show the benefits of the new approach which represented by the compression ratio compared with other algorithms, which means less space storage, less transmission time, less bandwidth, and less energy consumption.

3. Proposed Method

The proposed method is an optimization of our recent work. In this work, we kept the same main idea of the previous work which is splitting the integer and the float parts and compressed them with different methods. The improvement we have made in this work is summarized in the following points:

First, we used the delta encoding algorithm in the preprocessing step to make the data more convergent, resulting in increasing the similarity rate of the data that will be proved later.

Second, we will apply a proposed modified RLE (T-RLE) in the integer part that overcomes the variables that have two occurrences, which we have noticed in abundance in the previous work. Figure 1 is a flowchart of the proposed method, followed by the description of the innovative phase.

3.1. Delta Encoding. Delta encoding, also known as relative encoding, is one of the lossless compression algorithms, which has a simple concept that works on counting the difference between the successive samples [22], as shown in

$$\sigma = x_i - x_{i-1}. \quad (1)$$

We used this algorithm to process the new data (the difference between the successive samples) instead of the original data. This step showed a significant impact on the similarity ratio in the datasets because the range of the new datasets became smaller than the original dataset.

3.2. Modified RLE (T-RLE). The RLE algorithm has a simple strategy that replaces sequences of repeated characters with the number of occurrences [23]. At first glance, it seems to save a lot of space, especially for characters that have more than three occurrences, which gives a positive reduction. On the other hand, a negative reduction will be given for a

character that has one occurrence (adding more characters to represent the original character). Instead of having one character, applying RLE will give (character: 1, Repetition: 1) which is two characters to represent the original character. This case has been mentioned in [24].

Among the applications of RLE algorithm is to separate the runs from the variables for further compression and in case of dealing with numbers like temperature and humidity. Let consider a set of 26 characters (1 1 1 3 5 5 7 9 9 11 13 13) after applying RLE:

Variables (1 3 5 7 9 11 13).

Repetition (3 1 2 1 2 1 2).

Total: 28 characters.

RLE 1 has been applied for text data (contains letters). As we see in the example below, it is not suitable for numerical data. Most of the time the readings will be wrong like the first variable can be interpreted as eleven one time or two times one's which requires access to the repetition. We proposed a solution to solve this problem by adding a (,) after the variables that have one occurrence for the modified RLE 1:

Applying RLE 1 will give the following:

Variables (11 3 55 7 99 11 1313)

Repetition (3 2 2 2)

Total: 27

Modified RLE 1:

Variables (1 3,5 7,9 11,13)

Repetition (3 2 2 2)

Total: 22

A no reduction will be given for the characters that have two occurrences (the number of the represent characters and the original characters are equal); the application of RLE will give (character: 1, Repetition: 2). We are going to treat this case in our proposed T-RLE. Algorithm 1 presents the pseudocode of the T-RLE algorithm.

T-RLE is applied for the same example:

Variables (1 3,5&7,9&11,13&)

Repetition (3)

Total: 17

Regarding the characters that have more than 2 occurrences, the T-RLE functions similarly to the RLE algorithm. In the variable output, we put a space () after the character, which indicates that this character has more than 2 occurrences. The character repetition will be saved in the repetition output. Run-length encoding can be expressed in multiple ways to accommodate data properties like weather measurement data (temperature and humidity). We proposed a T-RLE algorithm that fits with this kind of data, especially in regards to numbers that have 1 or 2 occurrences.

3.3. Bzip2. Bzip2 compression is an alternative to GZIP that works almost the same way. Bzip2 uses the Burrows-Wheeler transform with Huffman coding. The advantage of Bzip2 is the greater compression capability compared to LZ77/LZ78 [25].

4. Results and Discussion

4.1. Dataset Description. To evaluate our proposed approach, we used some real deployment datasets from the sensor scope

FIGURE 1: A flowchart for the proposed method.

T-RLE pseudocode.

```

While is not EOF do
  repetition = 0
  Repeat
    get input Character;
    repetition = repetition + 1
  Until Character unequal to next Character
  If repetition < 2 do
    output1 << character,
  If repetition = 2 do
    output1 << character&
  If repetition > 2 do
    output1 << character␣
    output2 << repetition
  End while
  
```

ALGORITHM 1:

project described in Table 1. Sensor scope is an environmental monitoring system of several projects; these are made publicly available for research purposes. These datasets are weather measurement, which have many parameters (wind speed, wind direction, etc.). In our case, we only process the temperature and humidity measurements. We compared the performance of our approach to the previous method and with some traditional algorithms like GZIP, Bzip2, LZMA, LZ77, LZW, and RLE-Bzip2.

4.2. Dispersion Measures. To show the interest of applying delta encoding, we plot below the frequency distribution of

the original data and after applying delta encoding in Figures 2 and 3, respectively. Besides, we calculate some dispersion measures, which are numerical measures that used to measure the degree of homogeneity (convergence) or dispersion (divergence) of samples. They are used to describe the dataset, as well as to compare different datasets [26]. However, we have calculated the mean (2) and the variance (3), and the standard deviation (4). To have a quick idea of how dispersed the data are and the convergence of the samples in the original datasets and after applying delta encoding as Table 2 shows.

$$\bar{x} = \frac{x_1 + x_2 + x_3 + \dots + x_n}{N} = \frac{1}{N} \sum_{i=1}^N x_i, \quad (2)$$

$$\text{Var}(x) = \frac{1}{N} \sum_{i=1}^N (x_i - \bar{x})^2, \quad (3)$$

$$S = \sqrt{\frac{\sum_{i=1}^N (x_i - \bar{x})^2}{N - 1}}. \quad (4)$$

The analysis of the results shows an evident magnitude of the effect of applying delta encoding through the dispersion measurement of all the datasets. The mean indicates the center of the datasets which equal to 0 in all the datasets approximately, together with the variance and the standard deviation which is smaller in all the datasets that we applied delta encoding. Note that the smaller the variance and the standard deviation, the smaller the dispersion rate which

TABLE 1: Data description.

Deployment name	Node ID	Symbolic name	No. of samples	Time interval	
				From day	To day
LUCE	84	LU_84	64913	23/11/2006	17/12/2006
HES-SO FishNet	101	FN_101	12652	09/08/2007	31/08/2007
Le G��n��pi	20	LG_20	21523	04/09/2007	03/10/2007
Grand-St-Bernard	10	GSB_10	23813	15/09/2007	18/10/2007

FIGURE 2: Data distribution of the original data.

FIGURE 3: Data distribution after applying delta encoding.

expresses the high convergence the samples are. Besides, we have plotted the frequency distribution to observe the data distribution before and after applying delta encoding. All the plots show a high affinity for all the samples from 0. The high similarity of the data will be detected, which is the main aim of applying the delta encoding.

5. Compression Performance

5.1. Packet Compression Ratio. The compression ratio is the most crucial metric to begin with evaluating the parameters of any compression algorithms. To prevent the repeating of

the same results in the compression ratio and the packet compression ratio (PCR), we only calculate the CR which defined in

$$CR = \left(1 - \frac{CD}{UCD}\right) \times 100, \quad (5)$$

where CD and UCD represent the size of the compressed and the original data, respectively, and each packet can contain at most 29 bytes of payload [13]. From here, we can count the number of packets required to transmit the compressed and

TABLE 2: Dispersion measurement.

Dataset	Original data			Delta encoding		
	Mean	Variance	Standard deviation	Mean	Variance	Standard deviation
LU_84 Temp	7.21445	10.00894	3.16369	0.00009	0.00381	0.06174
FN_101 Temp	14.92182	15.02011	3.87558	0.00108	0.07598	0.27564
LG_20 Temp	4.09063	16.37425	4.04651	0.00024	0.11180	0.33437
GSB_10 Temp	4.34265	15.62750	3.95316	-0.00019	0.04547	0.21323
LU_84 Hum	87.0403	64.6389	8.0398	0.00115	0.36820	0.60680
FN_101 Hum	83.94107	95.86922	9.79128	0.00592	2.28435	1.51141
LG_20 Hum	69.71255	377.71902	19.43499	0.00419	8.08112	2.84273
GSB_10 Hum	68.16569	330.42367	18.17756	0.00331	3.12383	1.76744

TABLE 3: The compression ratio of the proposed method on several datasets.

Dataset	Original size (bytes)	Compressed size (bytes)	Number-UnComp-packet	Number-Comp-packet	CR (%)
LU_84 Temp	403,410	32,580	13,911	1123	91.92
FN_101 Temp	87,078	8,615	3,003	297	90.11
LG_20 Temp	134,178	20,075	4,627	692	85.04
GSB_10 Temp	148,118	19,632	5,108	677	86.75
LU_84 Hum	454,391	90,381	15,669	3,117	80.11
FN_101 Hum	88,557	17,575	3,054	606	80.15
LG_20 Hum	150,661	36,660	5,195	1264	75.67
GSB_10 Hum	166,107	38,135	5,728	1315	77.04

TABLE 4: Comparative results of the proposed method with existing algorithms in terms of the compression ratio.

Dataset	Data size (bytes)					
	GZIP	Bzip2	LZMA	LZ77	LZW	RLE-Bzip2
LU_84 Temp	81.17	89.93	88.09	73.48	71.84	90.58
FN_101 Temp	76.42	86.03	85.61	67.91	65.48	88.54
LG_20 Temp	73.06	83.37	80.84	65.18	64.81	84.24
GSB_10 Temp	74.36	84.54	82.63	66.28	65.27	85.36
LU_84 Hum	65.42	73.74	75.86	54.90	59.18	79.16
FN_101 Hum	64.65	72.43	74.85	53.74	53.48	78.36
LG_20 Hum	60.95	69.74	70.56	49.72	53.37	75.38
GSB_10 Hum	61.76	69.83	71.98	50.32	54.13	76.27

uncompressed data. Table 3 presents the results obtained of the proposed method against several datasets in terms of the compression ratio. Table 4 and Figure 4 present the results obtained and the comparison with other algorithms. The results show a significant improvement in the compression ratio compared with our previous work (RLE-Bzip2) and other compression algorithms, resulting in the reduction of compressed packets that require less bandwidth hence less transmission time.

5.2. Compression Rate. In addition to CR, another parameter to evaluate any compression algorithm is the compression rate (CRate), as shown in (6). It is known as the number of output bytes per character which defined as [27]

$$\text{CRate} = \frac{\text{CD}}{\text{UCD}}. \quad (6)$$

The ratio represents the number of the bytes per character in which the compressed data will be represented. Table 5 describes the compression rate of the proposed method compared to some compression algorithms. However, the proposed method gave good results under the same datasets in terms of compression rate; this latter will be better whenever it is small. Unlike the compression ratio, the lower the compression rate, the better the compression performance of the algorithm [9].

5.3. Percentage Improvement of Compression Rate (PICR). The percentage improvement of compression rate [28] is the amelioration percentage of the proposed method compared concerning other algorithms which defined in

$$\text{PICR} = \left(1 - \frac{\text{CRate}_P}{\text{CRate}_C}\right) \times 100, \quad (7)$$

FIGURE 4: Compression ratio of the proposed method compared with other algorithms.

TABLE 5: Comparative results of the proposed method with existing algorithms in terms of the compression rate.

Dataset	Data size (bytes)						Proposed
	GZIP	Bzip2	LZMA	LZ77	LZW	RLE-Bzip2	
LU_84 Temp	0.1883	0.1007	0.1191	0.2652	0.2816	0.0942	0.0808
FN_101 Temp	0.2358	0.1397	0.1439	0.3209	0.3452	0.1146	0.0989
LG_20 Temp	0.2694	0.1663	0.1916	0.3482	0.3519	0.1576	0.1496
GSB_10 Temp	0.2564	0.1546	0.1737	0.3372	0.3473	0.1464	0.1325
LU_84 Hum	0.3458	0.2626	0.2414	0.4510	0.4082	0.2084	0.1989
FN_101 Hum	0.3535	0.2757	0.2515	0.4626	0.4652	0.2164	0.1985
LG_20 Hum	0.3905	0.3026	0.2944	0.5028	0.4663	0.2462	0.2433
GSB_10 Hum	0.3824	0.3017	0.2802	0.4968	0.4587	0.2373	0.2296

where $CRate_P$ and $CRate_C$ are the compression rate of the proposed algorithm and the compression rate of the compared algorithm, respectively. The proposed method is an optimization of our previous work (RLE-Bzip2), where the gain percentage has the utmost importance. The gain varied between different datasets, where it reaches the highest value of 14.23% under the LU_84 Temp datasets. It also achieved the lowest value of 1.18% under the LG_20 Hum dataset. Thus, it corresponds precisely to the results we obtained in terms of the compression ratio. Those results are not significant, but they are considered significant results compared to the other traditional algorithms, as Table 6 shows.

6. Conclusions

Energy consumption was and still the utmost important constraint in wireless sensor networks. It considers as a hot topic for the researchers to find new methods to reduce or at least rationalize energy consumption. Data compression showed efficacy in reducing the data size to be transmitted, therefore consuming less energy to transfer the same data by the communication unit. In this paper, we proposed an optimized lossless compression approach of our previous work, which

TABLE 6: PICR of the proposed method over other compression algorithms.

Dataset	PICR of the proposed method concerning to					
	GZIP	Bzip2	LZMA	LZ77	LZW	RLE-Bzip2
LU_84 Temp	57.09	19.76	32.16	69.53	71.31	14.23
FN_101 Temp	58.06	29.21	31.27	69.18	71.35	13.70
LG_20 Temp	44.47	10.04	21.92	57.04	57.49	5.08
GSB_10 Temp	48.32	14.29	23.72	60.71	61.85	9.49
LU_84 Hum	42.48	24.26	17.61	55.90	51.27	4.56
FN_101 Hum	43.85	28.00	21.07	57.09	57.33	8.27
LG_20 Hum	37.70	19.60	17.36	51.61	47.82	1.18
GSB_10 Hum	39.96	23.90	18.06	53.78	49.95	3.24

based on delta encoding and a proposed T-RLE algorithm. Our approach showed a slight improvement compared to our previous work. On the other side, it performed good results comparing with the other traditional algorithms. In the future, we are going to deploy this method in a real WSNs, by considering the output data from the Analog to Digital Converter (ADC).

Data Availability

The datasets have been used by several works, and it is publicly available on sensor scope project: <https://lcav.epfl.ch/page-145180-en.html>. It was last accessed on 9/18/2018.

Conflicts of Interest

The authors declare no conflict of interest.

Acknowledgments

This work was partially supported by the Defense Industrial Technology Development Program (Grant No. JCKY2018110C165) and the Hubei Provincial Natural Science Foundation of China (Grant No. 2017CFA012).

References

- [1] L. Catarinucci, D. de Donno, L. Mainetti et al., "An IoT-Aware architecture for smart healthcare systems," *Ieee Internet of Things Journal*, vol. 2, no. 6, pp. 515–526, 2015.
- [2] V. C. Gungor and G. P. Hancke, "Industrial wireless sensor networks: challenges, design principles, and technical approaches," *Ieee Transactions on Industrial Electronics*, vol. 56, no. 10, pp. 4258–4265, 2009.
- [3] J. Yick, B. Mukherjee, and D. Ghosal, "Wireless sensor network survey," *Computer Networks*, vol. 52, no. 12, pp. 2292–2330, 2008.
- [4] Y. Sankarasubramaniam, I. F. Akyildiz, and S. W. McLaughlin, *Energy Efficiency Based Packet Size Optimization in Wireless Sensor Networks*, Ieee, New York, 2003.
- [5] M. A. Razzaque and S. Dobson, "Energy-efficient sensing in wireless sensor networks using compressed sensing," *Sensors*, vol. 14, no. 2, pp. 2822–2859, 2014.
- [6] W. K. Seah, Z. A. Eu, and H.-P. Tan, "Wireless sensor networks powered by ambient energy harvesting (WSN-HEAP)-Survey and challenges," in *2009 1st International Conference on Wireless Communication, Vehicular Technology, Information Theory and Aerospace & Electronic Systems Technology*, pp. 1–5, Aalborg, Denmark, May 2009.
- [7] A. S. K. Mammu, A. Sharma, U. Hernandez-Jayo, and N. Sainz, "A novel cluster-based energy efficient routing in wireless sensor networks," in *2013 IEEE 27th International Conference on Advanced Information Networking and Applications (AINA)*, pp. 41–47, Barcelona, Spain, March 2013.
- [8] F. Engmann, F. A. Katsriku, J.-D. Abdulai, K. S. Adu-Manu, and F. K. Banaseka, "Prolonging the lifetime of wireless sensor networks: a review of current techniques," *Wireless Communications and Mobile Computing*, vol. 2018, Article ID 8035065, 23 pages, 2018.
- [9] J. G. Kolo, S. A. Shanmugam, D. W. G. Lim, and L.-M. Ang, "Fast and efficient lossless adaptive compression scheme for wireless sensor networks," *Computers & Electrical Engineering*, vol. 41, pp. 275–287, 2015.
- [10] Y. Park and J.-S. Kim, "zFTL: power-efficient data compression support for NAND flash-based consumer electronics devices," *IEEE Transactions on Consumer Electronics*, vol. 57, no. 3, pp. 1148–1156, 2011.
- [11] W. Cui, "New LZW data compression algorithm and its FPGA implementation," in *Proc. 26th Picture Coding Symposium (PCS 2007)*, Lisboa, Portugal, 2007.
- [12] D. L. Duttweiler and C. Chamzas, "Probability estimation in arithmetic and adaptive-Huffman entropy coders," *IEEE Transactions on Image Processing*, vol. 4, no. 3, pp. 237–246, 1995.
- [13] A. Y. Tuama, M. A. Mohamed, A. Muhammed, and Z. M. Hanapi, "A new compression algorithm for small data communication in wireless sensor network," *International Journal of Sensor Networks*, vol. 25, no. 3, pp. 163–175, 2017.
- [14] R. Xuejun and R. Zhongyuan, "A sensor node lossless compression algorithm based on linear fitting residuals coding," in *Proceedings of the 10th International Conference on Computer Modeling and Simulation*, pp. 62–66, Sydney, Australia, January 2018.
- [15] U. Tiwari and S. Mehruz, "Retracted: an adaptive entropy based lossless data compression algorithm for wireless sensor network," *Journal of Fundamental and Applied Sciences*, vol. 10, pp. 706–715, 2018.
- [16] C. M. Sadler and M. Martonosi, *Data compression algorithms for energy-constrained devices in delay tolerant networks*, SenSys 2006, Boulder, Colorado, USA, 2006.
- [17] F. Marcelloni and M. Vecchio, "An efficient lossless compression algorithm for tiny nodes of monitoring wireless sensor networks," *The Computer Journal*, vol. 52, no. 8, pp. 969–987, 2009.
- [18] 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," *Journal of Sensors*, vol. 2012, Article ID 539638, 20 pages, 2012.
- [19] U. J. V. T. and D. P., "A new lossless neighborhood indexing sequence (NIS) algorithm for data compression in wireless sensor networks," *Ad Hoc Networks*, vol. 83, pp. 149–157, 2019.
- [20] "Sensor Scope Project," https://figshare.com/articles/Sensorscope_Data/8081813.
- [21] A. Saidani, J. Xiang, and D. Mansouri, "A new lossless compression scheme for WSNs using RLE Aalgorithm," in *2019 20th Asia-Pacific Network Operations and Management Symposium (APNOMS)*, pp. 1–6, Matsue, Japan, Japan, Sept. 2019.
- [22] W. Xia, H. Jiang, D. Feng, L. Tian, M. Fu, and Y. K. Zhou, "Ddelta: a deduplication-inspired fast delta compression approach," *Performance Evaluation*, vol. 79, pp. 258–272, 2014.
- [23] D. Salomon, *Data Compression: The Complete Reference*, Springer Science & Business Media, 2004.
- [24] M. A. Aziz, M. S. Sahana, and M. P. Thapliyal, "Implementation of H. 264/MPEG-4 AVC for compound image compression using histogram based block classification scheme," *International Journal of Computer Science and Information Technologies*, vol. 6, no. 4, pp. 3479–3488, 2015.
- [25] J. Gilchrist, "Parallel data compression with bzip2," in *Proceedings of the 16th IASTED international conference on parallel and distributed computing and systems*, pp. 559–564, Ottawa, Canada, 2004.
- [26] T. Srisooksai, K. Keamarungsi, P. Lamsrichan, and K. Araki, "Practical data compression in wireless sensor networks: a survey," *Journal of Network and Computer Applications*, vol. 35, no. 1, pp. 37–59, 2012.

- [27] J. S. Vitter and P. Krishnan, "Optimal prefetching via data compression," *Journal of the ACM (JACM)*, vol. 43, no. 5, pp. 771–793, 1996.
- [28] T. Bose, M. H. Mohammed, A. Dutta, and S. S. Mande, "BIND—an algorithm for loss-less compression of nucleotide sequence data," *Journal of Biosciences*, vol. 37, no. 4, pp. 785–789, 2012.