ISSN 2047-3338

An Approach to Data Compression and Aggregation in Wireless Sensor Networks

Elie T. FUTE[1], Alain B. BOMGNI[2] and Hugues M. KAMDJOU[3]
[1,2,3]University of Dschang, Department of Mathematics and Computer Science, Dschang, Cameroon
[1]eliefute@yahoo.fr, [2]alain.bomgni@yahoo.fr

***Abstract*—Wireless Sensor Networks (WSNs) are used in many application areas. Their deployment should take into account their limited resources (energy, memory, computational capacity and storage). One of the objectives targeted by the deployment is data collection. However, the data collected contain a large amount of redundant information which yields to a large consumption of resources during the storage, processing and transmission. To remedy these constraints, we propose in this paper an approach to data compression and aggregation in WSNs called "BMRARI Distributed Ordering Compress" which is based on the "BMRARI Compress" coding (inspired from the Arithmetic, RLE, BWT and MTF coding).**

***Index Terms*— Compression, Aggregation, Networks and Wireless Sensor**

I. INTRODUCTION

SENSORS communicate via radio links and are generally intended to be deployed in large numbers to cover large surfaces [2]. Their potential applications are diversified in areas such as detection and monitoring of disasters, environmental monitoring, mapping of biodiversity, intelligent building, precision farming, preventive maintenance of machinery, medicine, health, logistics and transport. Thus, the fact that the network has to operate autonomously, without human intervention, the sensors have three main functions, among others: 1) collecting data related to their physical environment (temperature, pressure, vibration, light, movement, sound, image and video) in their structures; 2) processing of data collected; 3) the transmission of such data to a processing center. [3] In a WSN, the amount of collected data consumes a lot of energy during transmission and also requires storage. To this end, several methods of compression and aggregation data have been developed. These methods aim on the one hand to reduce the transmission energy by transforming data into a more compact form at the cost of a compression and decompression effort, on the other hand, to provide a safety measure based on the unintelligible of information.

The matter is to define a good technique of data compression and aggregation that optimizes at the same time the storage memory; the various processing time and transfer time; the compatibility; the security and the lifetime of sensors and hence that of the network.

The main objective of this paper is to propose an effective approach to compression and aggregation of data, which ensures optimal management of sensors resources while ensuring some functional constraints such as the integrity of information. This is specifically minimizing the transfer energy by reducing the size of the collected data so that restoration at the treatment center is without information loss.

The rest of this paper is organized as follows: in Section II, we present a state of the art on data compression and aggregation problems in WSNs; in Section III, we present an approach to compression and aggregation of data in WSNs; in section IV we present an implementation and interpretation of the results of "BMRARI Compress"; in Section V, an implementation of "BMRARI Distributed Compress" in WSNs and we end with a conclusion and perspectives in Section VI.

II. STATE OF THE ART

A. Data compression Approaches

Data compression follows two steps always present in the process namely: compression (reducing the size of the original data) and decompression (restore the original data using the compressed file) [4]. Compression is said to be with loss if the data after decompression are different from the original data, while a compression is said to be without loss if the data after decompression are identical to the original data. However, there are five reasons that can lead to data compression: saving space, the gain in processing time, compatibility, reducing collisions and security. We then present the standard techniques of data compression.

1) Run-length encoding (RLE)

Its principle is to replace multiple occurrences of the same symbol by a copy of the symbol and the number of times it appears consecutively in the data [5].

2) Burrows-Wheeler Transform (BWT)

It's principle is to reorganize a string in a series of similar characters. The output contains exactly the same characters as the input; the only difference is the order in which they appear [6].

3) *Move-to-front (MTF) encoding*

MTF consist in replacing each character with an index given by a table progressing dynamically [7,8]; this in order to show zero (0) when the input word contains identical character sequences.

4) *Compression PPM, LZMA, LZ77, LZ78, LZO, 7-Zip, WinRar, WinZip,Bzip2 et Gzip*

7-Zip is the compression algorithm that incorporates the following [9]: LZMA (improved and optimized version of LZ77), LZMA2 (improved version of LZMA), PPMD, BCJ, BCJ2, Bzip2 (compression using the BWT transform, LZMA and Huffman coding) and Gzip or Deflate (similar to LZO is a standard algorithm based on LZ77 and Huffman). Unlike LZ77, the LZ78 does no longer uses the sliding window but a dictionary containing sequences of byte already encountered. WinRar uses LZ77 and Huffman while WinZip uses LZ77, LZMA, BWT and Huffman.

5) *The Discrete Cosinus Transform (DCT)*

DCT is a mathematical function used for destructive data compression, including sounds, images and videos according to JPEG and MPEG standards. During the compression operation of data in the different formats, the algorithm converts the image pixels or samples of the audio sequence into frequencies [10], while removing frequencies which do not correspond to the relevant data for the human eye or ear.

6) *JPEG (Joint Photographic Experts Group) Compression*

In some areas such as medical imaging, the faithful reproduction of the original image is very important. In order to exploit this property, JPEG compression converts the original image from its original color model RGB (Red, Green, Blue) to the model type chrominance/luminance YIQ (luminance, interpolation and quadrature) [11], [12].

7) *The fractal compression*

The goal of fractal is that every image is a finite set of geometric transformations (rotations, translations, additions, reductions) applied to subsets of identical patterns and of different sizes within it. This method consists in detecting on the one hand the recurrence of patterns at different scales, and on the other hand to eliminate the redundancy of information in the image [13, 14].

8) *Discrete Wavelet Transform (DWT)*

DWT is a compression method for images and videos based on the mathematical theory of signal analysis. Wavelets are sets of elementary signals from which we reconstruct a complex signal [15]. The goal of this transform is to determine the correlation of several wavelets (compressed or expanded the mother wavelet) with the signal, this by highlighting the details and the overall look [16].

9) *Huffman Coding*

This is compression technique for texts, images, sounds and videos aimed at finding the number of occurrences of each character, then to assign a short code to most frequent characters, and a longer code at the least frequent characters. For this purpose Huffman uses a tree based on the basic concepts of information theory (Shannon-Fano), namely the concept of quantity of information and entropy [17], [18].

10) *Arithmétic coding*

This encoding replaces the set of symbols read by a single real number is between [0, 1). Indeed, any sequence of any length can be represented by such a number which will distinguish it from other possible sequences [19].

11) *Distributed source coding*

This coding does not allow communication between coders. Thus, the encoding is always done separately and decoding always together. The general idea is to encode the side information Y at a rate equal to its entropy H (Y), then find a variable length code to encode X closest to rate its conditional entropy H (X | Y) [20], [21].

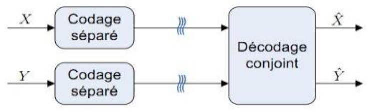

Fig. 1: Distributed source coding system

In view of the above techniques, we notice that some are general while others expect data of a certain nature, such as: images, videos, sounds, texts. Some perform a lossless encoding and others with loss. The firsts mandate is to restore the original data without any difference after having them undergo stages of compression and decompression. The lasts can alter the data after compression and decompression provided the reconstructed data are relatively similar to the original data.

The compression is based on the repetition of proper data from a node. This is the reason why when the neighboring nodes have identical data, the compression process is called aggregation.

B. *Data aggregation approches*

In WSNs, data produced by the neighboring sensor nodes are highly correlated spatially and temporally; it can lead the receipt by the base station of redundant information (the Shannon theory called redundancy whatever appears as surplus in the message) [22]. The aggregation of data therefore reduces this redundancy [23]. Therefore, the intermediate nodes aggregate information received from multiple sources.

1) *Pipelined In-Network Compression*

In this technique [24], data collected by the sensor are stored in the buffer of the aggregation node for some time [25]. During this time, packets of data are combined into one package while suppressing the redundancy (see Fig. 2).

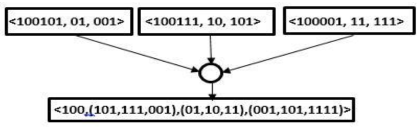

Fig. 2 : In-network Compression

2) *Coding by Ordering*

In this technique the intermediate nodes compress the information received from sources rejecting some data packets while adding the information about removing the packet associated with the node [26].

In view of aggregation techniques, we notice that the aggregation Pipelined In-Network allows to factorize common parts of several packages assuming similarities between the transmitted packets and a large buffer to store the packets arriving before attempting to factorize. On the contrary the Coding by Ordering consist of discarding a number of packets regardless of any similarity with the ability to fully recover the information of discarded packets.

From the descriptions of paragraphs A and B, we realize that the data compression process incorporates aggregation.

III. BMRARI COMPRESS: A RESOLUTION APPROACH TO THE PROBLEM

The BMRARI Compress approach to data compression and aggregation in WSNs has an open architecture with four components (see Figure 3). For now, the four components which are integrated are based on the ARITHMETIC, RLE, MTF and BWT codings. The reason why these codings were chosen is that they are of particular value in applications constrained by energy because their settlements involve simple instructions of additions and integer values shifts.

The components 3 and 4 are based on processing techniques which consist in identifying redundancy in raw data in order to facilitate the aggregation by the component 2. Once the data is without redundancy, the component 1 compresses it to yield an unintelligible file whose size is smaller.

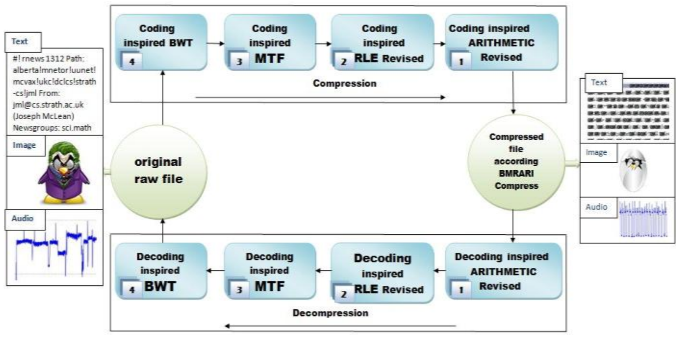
Text
#! rnews 1312 Path: alberta!mnetor!uunet! mcvax!ukc!dclcs!strath -cs!jml From: jml@cs.strath.ac.uk (Joseph McLean) Newsgroups: sci.math
Image
Audio
Coding inspired BWT 4
Coding inspired MTF 3
Coding inspired RLE Revised 2
Coding inspired ARITHMETIC Revised 1
Compression
original raw file
Compressed file according BMRARI Compress
Text
Image
Audio
Decoding inspired BWT 4
Decoding inspired MTF 3
Decoding inspired RLE Revised 2
Decoding inspired ARITHMETIC Revised 1
Decompression

Fig. 3: Architecture of BMRARI Compress (4 components numbered from 1 to 4)

A. *Component 1: based on the Arithmetic coding*

The coding of component 1 aims at replacing the set of symbols read by a single real number that lies in the interval [0,1). Indeed, any sequence of any length can be represented by such a number that will distinguish the other possible sequences. But if we really consider all possible sequences, this number may include an infinite number of digits in its binary representation.

Using this compression method has a defect due to the use of a fixed table of statistics reflecting the same problems as for Huffman coding, that is to say that the file compression with atypical statistics (symbols to be underrepresented find themselves with a high probability of occurrence) will be more voluminous after compression than before.

To overcome this, the most effective solution is to use an adaptive array where all frequencies have the same probability distribution; then at each meeting of a symbol, the statistics are betting updates and suitable intervals (see algorithms 1 and 2). To increase the chances of having a good compression rate, we precede the coding transformations of components 2, 3 and 4.

Algorithm 1: Component 1 coding

```
Input: source file;
Output: compressed file;
BEGIN
  $low \leftarrow 0$; $up \leftarrow 1$; $range \leftarrow 1$;
  WHILE (Not end of file) DO
    $symbol \leftarrow$ get symbol ;
    $low \leftarrow low + range * lower\ of\ current\ symbol$;
    $up \leftarrow low + range * upper\ of\ current\ symbol$;
    $range \leftarrow up - low$;
    $Ouput\ a\ code\ so\ that\ lower \leq code < up$;
  ENDwhile
END
```

Algorithm 2: Component 1 decoding

```
Input: compresed file;
Output: decompressed file;
BEGIN
  $low \leftarrow 0$; $up \leftarrow 1$; $range \leftarrow 1$;
  WHILE (Not end of file) DO
    $value \leftarrow$ get code of symbol ;
    $symbol \leftarrow$ symbol such that upper symbol
      $\leq ((value - low)/(up - low)) < low\ symbol$ ;
    $range \leftarrow up - low$;
    $up \leftarrow low + range * low\ of\ current\ symbol$ ;
    $low \leftarrow low + range * up\ of\ symbol$ ;
    Return symbol ;
  ENDwhile
END
```

B. *Component 2: based on the Run-Length Encoding*

The coding of component 2 is to replace all the consequences of such characters or bits by a number representing the number of repetitions of the character or bits followed by this character (see algorithms 3 and 4) [27], [5].

The compression component that is effective to compress data when consecutive elements of repetitions are numerous (black and white), which is not always the case; that is why we precede this coding transformations of components 3 and 4; this in order to increase the chances of having more repeated characters or bits.

C. *Component 3: based on Move-To-Front (MTF) coding*

The coding of component 3 is intended to show zero (0) when the input word contains sequences of identical characters, producing highly compressible data (see algorithms 5 and 6).

We perform this coding to the transformation of component 4 in order to increase the chances of having more consecutive repeated characters or bits.

```
Algorithm 3: Component 2 coding
Input: source file;
Output: compressed file;
BEGIN
    $meter \leftarrow 0$; $boolean \leftarrow false$ ;
    precedent symbol ← get current symbol;
    WHILE ((current symbol ← next symbol) $\geq 0$) DO
        Write current symbol in le buffer;
        IF (current symbol=precedent symbol) THEN
            compteur ← 0 ;
            booleen ← $faux$ ;
            WHILE (($meter < 255$)et((current symbol ← next symbole) $\geq$ 0)and($boolean = false$)) DO
                IF (current symbol=precedent symbol) THEN
                    $meter \leftarrow meter + 1$ ;
                ELSE
                    $boolean \leftarrow true$;
                ENDif
            ENDwhile
            write meter in buffer;
            IF (($meter \neq 255$)and (current symbol $\geq 0$)) THEN
                Write symbol in buffer;
            ENDif
        ENDif
        precedent symbol ← current symbol;
    ENDwhile
END
```

```
Algorithm 4: Component 2 decoding
Input: compresed file;
Output: decompressed file;
BEGIN
    $meter \leftarrow 0$; precedent symbol ← current symbol;
    WHILE ((current symbol ← next symbol) $\geq 0$) DO
        Write symbol in buffer;
        IF (current symbol=precedent symbol) THEN
            meter ← next symbole ;
            WHILE (($meter \leftarrow meter - 1$) $> 0$) DO
                Write current symbol in buffer;
            ENDwhile
        ENDif
        precedent symbol ← current symbol courant;
    ENDwhile
END
```

```
Algorithm 5: Component 3 coding
Input: source file;
Output: compressed file;
BEGIN
    Initialize the symbol table;
    WHILE (Not end of file) DO
        $S \leftarrow$ get symbol;
        Write position $S$ symbol of the symbol table in the buffer ;
        Move the symbol $S$ to the head of the table symbols ;
    ENDwhile
END
```

```
Algorithm 6: Component 3 decoding
Input: compresed file;
Output: decompressed file;
BEGIN
    Initialize the symbol table;
    WHILE (Not end of file) DO
        $P \leftarrow$ get symbol position;
        Write symbol in position $P$ of the symbol table in the buffer;
        Move the symbol $S$ the head of the table symbols;
    ENDwhile
END
```

D. *Component 4: based on the coding of the Burrows-Wheeler Transform (BWT)*

The coding of component 4 is to reorganize a string into series of similar characters. The result contains the exactly the same character has the input. The only difference is the order in which they appear. This is useful for compression, since it tends to be easy to compress a string that has a series of repeated characters. This component improves the efficiency of text compression algorithms. The principle of the algorithm is the following:

–The first step is to read in a block of *N* symbols (So...Sn-1)

–The next step, we consider a block as a ring buffer of *N* strings (rotations) So. ..Sn-1 which can be constructed such that: So = So,...,Sn-1 ;… ; Sn-1= Sn-1 So,…, Sn-2

–The third step is to sort lexicographically S: So,...,Sn-1

–The final stage of the transformation is the output of a string *L* composed of the last character of each rotation in their sort order with *I* the number of the sorted row containing So.

IV. IMPLEMENTATION AND INTERPRETATION OF RESULTS

To verify the effectiveness of our approach BMRARI Compress, we compressed the eighteen (18) files of Benchmark Calgary Corpus [28]. Though we were talking exclusively of compressor in the description of our experiences, we have also developed a decompressor.

The results of our experiments obtained on a laptop computer (Pentium 4, HDD: 250 GB, CPU: 2x2.5 GHz, RAM: 2 GB OS: Ubuntu 12.04 32-bit) are presented in Table I. The first column contains the names of files that have been compressed, the second column contains the size of the uncompressed files (in bytes), the other seven columns that follow contain the sizes of the files compressed using high-performance standard compression algorithms (gzip, 7-zip, WinZip, WinRAR, Huffman and Arithmetic).

Table I: Results Of The Experiments (Size In Bytes)

File	Original size	Winrar	Winzip	Gzip	7-zip	Huffman	Arithmé-tique	BMRARI Compress
bib	111 261	33 063	35 139	35 063	30 644	73 173	72 789	29 411
book1	768 771	277 126	313 273	313 376	261 125	438 792	436 883	274 869
book2	610 856	181 214	206 638	206 687	169 820	73 845	72 400	61 972
géo	102 400	62 469	68 560	68 493	53 290	73 845	72 400	61 972
news	377 109	126 012	144 844	144 840	118 915	246 891	244 471	133 497
obj1	21 504	9 823	10 406	10 323	9 471	17 338	16 038	10 830
obj2	246 814	71 277	81 657	81 087	61 525	195 384	187 294	81 554
paper1	53 161	18 167	18 655	18 577	17 325	33 819	33 120	17 641
paper2	82 199	28 797	29 809	29 753	27 280	48 077	47 535	26 869
paper3	46 526	17 834	18 170	18 097	17 111	27 702	27 393	16 940
paper4	13 286	5 584	5 621	5 534	5 455	8 267	7 998	5 504
paper5	11 954	5 035	5 080	4 995	4 941	7 893	7 559	5 100
paper6	38 105	13 150	13 310	13 232	12 550	24 495	23 883	13 060
pic	513 216	49 824	56 535	56 442	41 976	107 354	74 804	54 347
progc	39 611	13 188	13 351	13 275	12 603	26 381	25 920	13 255
progl	71 646	15 799	16 356	16 273	15 028	43 425	42 619	16 875
progp	49 379	10 799	11 311	11 246	10 405	30 666	30 209	11 544
trans	93 695	18 010	19 060	18 985	16 783	65 720	64 326	19 179

The plot of Fig. 4 better illustrates Table I. The abscissa are made of the 18 files of Benchmark, and the ordinate their respective compression rate (τ = (1- ([Final volume] / [Initial Volume])) * 100) in percentage. At 0%, we have the graph of 18 uncompressed files which coincides with those representing compression rates of the BWT and MTF techniques. It appears from these results that the curves of BMRARI Compress and 7-zip exhibit best the compression rate with slight differences on the various file of the Benchmark (see Table II). Huffman, RLE, and Arithmetic, which have the lowest compression rate especially on book1, paper4 and paper5 files have values close to those of BMRARI Compress and 7-zip on the book2 file.

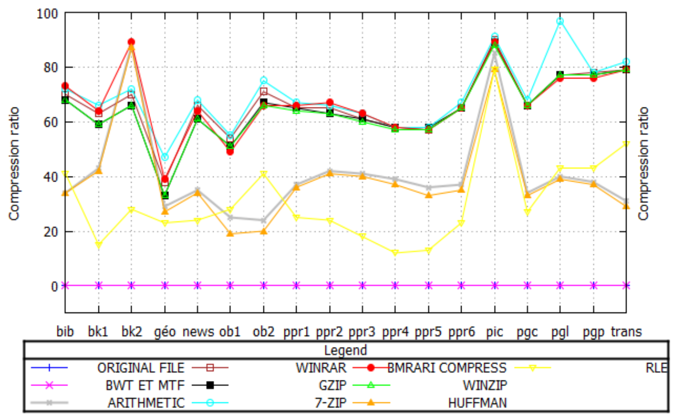

Fig. 4: Comparative curves of different techniques

A. *Results and Discussion*

It appears from these results that BMRARI Compress has good compression rate compared to commonly used compressors namely 7-Zip, WinZip, WinRAR, gzip. In view of Table 2 shows the difference in rate cuts of different techniques compared to BMRARI Compress (the rate with a "+" means that BMRARI is better and the sign "-" means that it is less better) we find that the compression of credit spreads are very low. However, these techniques have a complex default high compared to BMRARI Compress. This could be justified by the fact that it integrates several techniques BMRARI Compress who inspired that ARITHMETIQUE coding, RLE, MTF and BWT (see Fig. 3) [9], [29].

The results show the potential of the technique. Thus, BMRARI Compress decomposition facility module allows us to consider his experiment in a distributed manner in WSNs.

V. IMPLÉMENTATION OF BMRARI DISTRIBUTED COMPRESS DANS UN WSNS

The density of the network and location of sensors, often near to each other that the nodes can receive the same data. In order to avoid duplication of data collected and transmitted; and optimize energy consumption and minimize transmission time and processing of data collected by the sensors, we put on the feet BMRARI Distributed Compress approach of a hand to distribute compression and aggregation data at each node using the coding BMRARI Compress and secondly to distribute the compression and aggregation of data at a single node (Cluster Head) by. This transformation BMRARI Distributed Compress and can replace individual perceptions through a comprehensive and collective reading, avoiding redundancies on calculations and data transmissions performed individually on each sensor node.

Table II: compression ratio Differences BMRARI Compress

File	T.C.BM-Winzip	T.C.BM-Winrar	T.C.BM-Gzip	T.C.BM-7Zip
bib	+5,15	+3,26	+5,08	+1,11
book1	+5,00	+0,28	+5,01	-1,79
book2	+23,68	+19,50	+23,69	+17,66
geo	+6,43	+0,75	+6,37	-8,48
news	+3,01	-1,98	+3,01	-3,87
obj1	1,97	-4,69	-2,36	-6,32
obj2	+0,04	-4,12	-0,19	-8,12
paper1	+1,91	+0,96	+1,76	-0,59
paper2	+3,58	+2,27	+3,51	+0,50
paper3	+2,64	+1,90	+2,49	+0,37
paper4	+0,88	+0,52	+0,23	-0,37
paper5	-0,17	-0,63	-0,88	-1,33
paper6	+0,66	+0,19	+0,45	-1,34
pic	+0,43	-0,89	+0,41	-2,41
progc	+0,24	-0,16	+0,05	-1,65
progl	-0,72	-1,52	-0,84	-21,42
progp	-0,47	-1,46	-0,60	-2,31
trans	-0,13	-0,21	-0,21	-2,56

A. *Distribution BMRARI Compress with more nodes by aggregating area of perception*

In a WSN most part of energy is consumed when transmitting data packets [1] (over the package is large, the greater the amount of energy consumed during transmission and processing). Thus, because of the fact that a sensor with limited resources cannot activate the four components at once, we have divided our WSN into several areas of perception (clusters) and in each area, we decided to turn on the data for each node a component among four processing BMRARI Compress before transmission. Since concentrating all the processing BMRARI Compress at a source node, it will exhaust its energy very quickly and therefore decrease the network's lifetime (see Fig. 5).

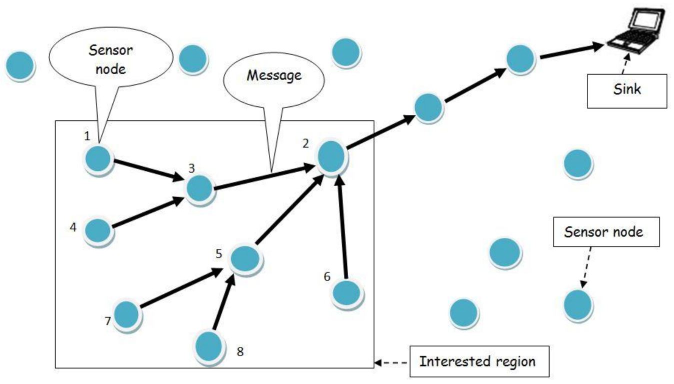

Fig. 5: BMRARI distributed Compress with multiple clusters Head in the collection area

On the example of Fig. 5, seven messages sent by the seven nodes in the region to the output node (Node 2), one message summarizing all the information received is passed to the next destination. This reduces the amount of information flowing through the network and hence saves energy dissipated in the transmission. But several scientific contributions have shown that the transmission of a bit to the treatment center in a WSN is equivalent to the execution of instructions by the processor thousand who demand a substantial amount of energy [30]. In the algorithmic description (Algorithm 7) BMRARI distributed at each sensor, it is assumed that all nodes in the region collect data and each sensor can activate the data collected any of the four component that integrates based on source data.

```
Algorithm 7: BMRARI distributed at each node
Input: m packets;
Output: one packet;
BEGIN
    The source node to enable the component 4 of the collected data and forwards the
    packet to the aggregation node;
    WHILE (deactivated component 1 in the package) DO
        Active on the received packet a component based on already activated components;
        Sends the message to the next node;
    ENDwhile
    Transmits the compressed packet to the treatment center;
END
```

B. BMRARI Compress distribution with a single aggregation node area of perception

In this approach we have divided our WSN into several areas of perception (clusters) and in each zone, we have one aggregation node and other nodes relay their perceptions and packets received aggregation node; When it collects all the data in the region, there activates a component of the four processing BMRARI Compress before transmitting to the next node. Since concentrating all the processing BMRARI Compress at a source node, it will exhaust its energy very quickly and therefore decrease the network's lifetime (see Fig. 6).

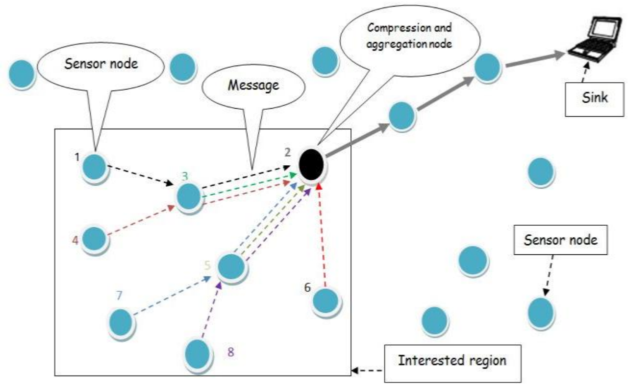

Fig. 6: BMRARI distributed Compress with single cluster Head in the collection area

On the example of Fig. 6, for the eleven messages from seven knots of the region to the Cluster Head node (aggregation node), one message summarizing all the information received is passed to the next destination. This reduces the amount of information flowing through the network and hence saves energy dissipated in the transmission. But several scientists contributions have shown that the transmission of a bit to the treatment center in a WSN is equivalent to the execution of instructions by the processor thousand who demand a substantial amount of energy [30]. In the algorithmic description (algorithm 8) BMRARI Compress distributed at the node of aggregation assumes that each Head cluster can activate any component of the four that integrates depending on the source data, there is only one node aggregation by collecting area and each sensor sends its perception aggregation node after a top time via multi-hop communication (see Fig. 6).

```
Algorithm 8: BMRARI distributed at Cluster Head
Input: m packets;
Output: one packet;
BEGIN
    The source nodes transmit the collected data to the next node;
    WHILE (deactivated component 1 in the package) DO
        IF node = aggregation node THEN
            Active on the super package formed a component based on data souces;
        ENDif
        Transmits the received packet to the next node;
    ENDwhile
    Transmits the compressed packet to the treatment center;
END
```

VI. CONCLUSION AND PERSPECTIVES

At the end of this work, we have proposed an approach to data compression and aggregation based on the four components of BMRARI Compress. Then we have adapted the classical algorithms of compression and aggregation of data to the context of WSNs to give birth to an approach called "BMRARI Distributed Compress" which is based on the "BMRARI Compress" coding (approach inspired from the Arithmetic, RLE, MTF and BWT codings).

Some prospects that can be considered for future works are the following:

- To determine automatically according to the type of file, the best combination of components for an optimal compression.
- To determine an efficient routing protocol suitable for BMRARI Distributed Compress approach.

REFERENCES

[1] W. Heinzelman, A. Chandrakasan, and H. Balakrishnan. Energy efficient communication protocols for wireless microsensor networks. In Proceedings of the 33rd Hawaiian International Conference on Systems Science, pages 3005–3014, January 2000.

[2] Le Hung-Cuong. Optimisation d’accès au médium et stockage de données distribuées dans les réseaux de capteurs. PhD thesis, Université de Franche-Comté, 2009.

[3] Steven Pigeon. Contributions à la compression de données. PhD thesis, Université de Montréal, Decembre 2001.

[4] Mozelle Gerard and Preteux Françoise. Compression de données multiples et dynamiques. PhD thesis, Université de Paris 05, 1998.

[5] Chin-Chen Changa, Chih-Yang Linb, and Yu-Zheng Wangb. New image steganographic methods using run-length approach. Department of Information Engineering and Computer Science, Feng Chia University, pages 3393–3408, 2006.

[6] M. Burrows and D. J. Wheeler. A block-sorting lossless data compression algorithm. Technical Report 124, 1994.

[7] Beaudoin Vincent. Développement de nouvelles techniques de compression de données sans perte. Mémoire, Université Laval Quebec, 2008.

[8] Z. Arnavut. Move-to-front and inversion coding. IEEE Data Compression Conference., pages 193–202, 2000.

[9] Ludovic Loth. Zoom sur 7-zip : est-il le meilleur ? sur clubic.com, octobre 2014.

[10] Min Li, Biswas Mainak, Kumar Sanjeev, and Nguyen Truong. Dct-based phase correlation motion estimation. IEEE, Piscataway NJ, Etats-Unis, Octobre 2004.

[11] G.K Wallace. The jpeg still picture compression standard. IEEE Transactions, Consumer Electronics, aout 2002.

[12] M. Garg. Performance analysis of chrominance red and chrominance blue in jpeg. World Academy of Science, Engineering and Technology, 2008.

[13] Omaima N. Ahmad Al-Allaf. Parfor and co-distributor parallel approaches for implementing fractal image compression based genetic algorithm. Science and Information Conference, pages 345–350, 2014.

[14] Franck Davoine. Compression d'images par fractales basée sur la triangulation de Delaunay. PhD thesis, Institut National Polytechnique de Grenoble, Avril 2010.

[15] A. Grossmann and B. Torrésani. Les Ondelettes. PhD thesis, CNRS-Luminy, Case 907, 13288 Marseille Cedex 09, 2001.

[16] Christophe R., Bertrand B., Olivier C., Jean-Michel G., Benoit D., and Jean-Paul Y. Compression par ondelettes de formulations intégrales sur architecture gpgpu. Conférence Européenne sur les Méthodes Numériques en Electromagnétisme, juillet 2012.

[17] Kubasov Denis. Codage de sources distribuées: nouveaux outils et application à la compression vidéo. PhD thesis, Université de Rennes 1, Décembre 2008.

[18] M. Sharma. Compression using huffman coding. International Journal of Computer Science and Network Security, May 2010.

[19] Amir Said. Introduction to arithmetic coding-theory and practice. HPL-2004-76, Palo Alto, April 2004.

[20] D. Slepian and J. Wolf. Noiseless coding of correlated information sources. IEEE Transactions on Information Theory, pages 471–480, July 1973.

[21] Aaron D. Wyner and Jacop Ziv. The rate-distortion function for source coding with side information at the decoder. IEEE Transactions on Information Theory, pages 1–10, January 1973.

[22] C.Y. Chong and S.P. Kumar. Sensor networks: Evolution, opportunities, and challenges. In Proceedings of the IEEE, pages 1247–1256, 2003.

[23] Merad Boudia O. R. Agregation des donnees et securite des reseaux de capteurs sans fil. PhD thesis, Universite de Tlemcen, 2014.

[24] R. Das. Performance and power optimization through data compression in network-on-chip architectures. IEEE, High Performance Computer Architecture, pages 215–225, Feb 2008.

[25] T. Arici, B. Gedik, Y. Altunbasak, and L. Liu. Pinco: a pipelined in-network compression scheme for data collection in wireless sensor networks. In Proceedings of 12th International Conference on Computer Communications and Networks, October 2003.

[26] D. Petrovic, R. C. Shah, K. Ramchandran, and J. Rabaey. Data funneling: Routing with aggregation and compression for wireless sensor networks. In Proceedings of First IEEE International Workshop on Sensor Network Protocols and Applications, May 2003.

[27] Agung B.W.R. Medical image watermarking with tamper detection and recovery using reversible watermarking with lsb modification and run length encoding (rle) compression. IEEE Communication, Networks and Satellite (ComNetSat), pages 167–171, July 2012.

[28] I. Witten, T. Bell, and J. Cleary. The calgary corpus. 1987.

[29] Krintz C. Adaptive on-the-fly compression. IEEE Transactions, Parallel and Distributed Systems, pages 15–24, Jan 2006.

[30] Ramdani Mohamed. Problème de sécurité dans les réseaux de capteurs avec prise en charge de l'énergie. PhD thesis, Université de Saad Dahlab de Blida, Novembre 2013.