A Simulink model for modified fountain codes

Nadhir Ibrahim Abdulkhaleq[1], Nahla Abdul Jalil Salih[2], Rajaa Salih Mohammed Hasan[2], Ihsan Jabbar Hasan[2]
[1]Department of Mobile Computing and Communication Engineering, College of Engineering, University of Information Technology and Communication (UoITC), Baghdad, Iraq
[2]Department of Electronics, Institute of Medical Technology, Middle Technical University, Baghdad, Iraq

Article Info

Article history:

Received Feb 05, 2022
Revised Oct 19, 2022
Accepted Oct 29, 2022

Keywords:

Fountain codes
Generator matrix
LDHS-LT
Short length files
Simulink model

ABSTRACT

This paper introduces a Simulink model design for a modified fountain code. The code is a new version of the traditional Luby transform (LT) codes. The design constructs the blocks required for generation of the generator matrix of a limited-degree-hopping-segment Luby transform (LDHS-LT) codes. This code is especially designed for short length data files which have assigned a great interest for wireless sensor networks. It generates the degrees in a predetermined sequence but random generation and partitioned the data file in segments. The data packets selection has been made serialy according to the integer generated from both degree and segment generators. The code is tested using Monte Carlo simulation approach with the conventional code generation using robust soliton distribution (RSD) for degree generation, and the simulation results approve better performance with all testing parameter.

Corresponding Author:

Nadhir Ibrahim Abdulkhaleq
Department of Mobile Computing and Communication Engineering, College of Engineering
University of Information Technology and Communication (UoITC), Baghdad, Iraq
Email: nadhir.abdulkhaleq@uoitc.edu.iq

1. INTRODUCTION

Forward error correction codes represent the core of channel coding techniques [1]. Usually, the code is generated in a length (N) greater than the actual data length (k). For the conventional block codes (N) is decided to be able to face the worst channel conditions. In the end of nineties [2], [3] invents the principles of fountain codes. The idea is simple, instead of predetermined the code length, they decide to have a code with variable length. By flooding the channel continuously with coded symbols and wait for the receiver to acknowledge the sender of collecting the sufficient number of coded symbols to recover the data file. The brilliant design for fountain codes presented by Karp *et al*. [3] with the name of Luby transform (LT) codes are the practical version of such type of codes.

Wireless networks are designed to send data files in packets, which in turn makes it more suitable for many applications [4]-[11]. Following this fact, the LT encoder is first truncated the data file into packets $(p_{d1}, p_{d2}, \dots \dots, p_{dk})$ as k represent the length of the data file. The encoding process summarized into three main phases.

Phase 1: degree generation, the encoder follows certain distribution $\Psi(d)$ to generate a set of integers, called degrees. With the first version of LT code a well-known robust soliton distribution has been used. Then several modifications are introduced to maintain the best distribution parameters which are (size of degree 1, minimum number of edges, contribution of all data packets, ability to break the decoding stuck). So, this phase is so important in the design of the fountain codes because it's affected the ability of fast data recovery in the receiver part. In this paper, we introduce a deterministic degree generation with limited set of integers in which a high probability of breaking the decoder stuck is achieved using our previous sequential decoding approach [12]-[14].

Phase 2: packets selection, the degree d which generated from phase 1 means that d packets have to be selected and combined to form the coded packet. In the conventional LT code, the packets are selected uniformly at random while in our proposed design, the selection is based on sequential progress. Which impose a relation between encoded packets that could be used to regenerate degree one again.

Phase 3: encoded packets generation, the combination of the selected packets is done using binary ex-or operations. Keep in mind that, each packet may have certain number of binary bits, the same ex-or operations will be done and the main job of this phase is to combine these packets to create the coded packets. These three main phases are theoretically repeated in endless loop till an acknowledgment has been arrived to inform full reception of the file packets. Since 2002, fountain codes witnessed many attempts to overcome performance limitations. Short length data files applications for such codes is a big challenge. In [15]-[17], the digital fountain codes are designed with a generator matrix that could deal with file length less than (10^3) packets. The idea is based on presenting new degree distributions that mitigate the limitations of the conventional robust soliton distribution (RSD) used for LT codes. Chang *et al.* [18], use a non-equal selection probability for the data packets. The idea merge between the high degree output packets with some low degree one to avoid decoding block. Belief propagation algorithm is modified to find a solution for an empty ripple size for short length blocks is presented in [19]-[25]. From these previous works, it is evident that the short length data files need an adjustment for the degrees of the encoded packets as well as the way of selection to the data packets required to form the encoded one. This is the idea presented in our work where we provide what we called limited-degree-hopping-segment Luby transform (LDHS-LT) to give limited range of small degrees and with hopping selection for the data segment to ensure participation of all data packets in the encoding process.

The paper is arranged to have the proposed LDHS-LT code generation in section 2. In section 3, the Simulink model for the generator matrix is presented. Comparing our proposed code with the onventional LT code is analyzed in section 4. The conclusion is reported in section 5.

2. GENRATOR MATRIX FOR LDHS-LT CODE

The LDHS-LT code is constructed in a very simple way and based on the well-known LT code. The efficiency of such fountain codes depends on the quality of the generator matrix. The conventional LT code has a generator matrix which constructed after fulfilling the above three main phases. Big consideration appears for the conventional LT code with short data length and that’s why our LDHS-LT code try to overcome it. Let’s try to construct this code in a similar way and present the steps of generation. The short data files in our design is chosen to be 32 packets. So, to generate our encoded packets, we have to do the following.

2.1. Segmentation

To ensure the contribution of all data packets in the encoding process and also provide certain kind of randomly choice, the data file is sliced into several S segments. The number of segments is determined:

$$S_n = mod(k/S_s) \tag{1}$$

Where S_s is the size of the data segment and it will be in the range of (3-5) packets. So, for 32 packets data file and $S_s = 4$ packets we shall have $S_n = 8$.

2.2. Degree generation

As explained in phase 1 above, the degree generation follows certain random distribution $\Psi(d)$. In this proposed code, the degree values are limited to follow the range of (1: S_s). Again for $S_s = 4$ we have the degrees of $(1, 2, 3, and\ 4)$. It is worth to mention that the generation of such degrees is random without replacement.

2.3. Degree generation

Figure 1, illustrates the data packets selection. In this example we have a prototype data file of size (k = 9) and if (S_s = 3) which according to that we have (d =1, 2 and 3). The degrees are generated randomly, like (3, 1, 2) and we have three segments (S_1, S_2, and S_3).

Our data file consists of 9 packets denoted by (p_{di}). Where the suffix ($i \in (1{:}9)$). These data packets are distributed into three segments (S_1, S_2, and S_3) in a sequential assigning. The distinguish between segments has been done via the type of the line that forming the circles which illustrates the data packets in the figure. For instance, the solid line circuits denoted the data packets in the second segment. Now, the random generator produces two random numbers, one for the degree and the second for the segment. In our example of Figure 1, the random generator produces the numbers (32, 23, 11, 21) to generates the three coded packets denoted by (P_{ci}) which have a rectangular shape in the figure. According to such formation, the first coded packet is generated from the result of ($P_{c1} = p_{d4} \oplus p_{d5} \oplus p_{d6}$). Consequently, ($P_{c2} = p_{d7} \oplus p_{d8}$), ($P_{c3} = p_{d1}$) and ($P_{c4} = p_{d1} \oplus p_{d3}$).

It worths to note that, there is no random selection for the data packets in our design. The data packets are selected in sequential manner. That's means if the degree is 1, the first data packet is selected. While for degree 2, the first two data packets are selected and so on for higher degrees. This sequential selection has great effect in breaking the decoder halt when the coded packet of degree 1 is lost either in the beginning of the decoder work or at any decoding step before full data file recovery. To illustrate this feature, suppose that at any step in the decoding mission no degree one is found to proceed and we have two coded packets of the form ($P_{cl} = p_{d4} \oplus p_{d5} \oplus p_{d6}$) and ($P_{cm} = p_{d4} \oplus p_{d5}$) then by ex-or these two coded packets, new degree 1 is generated and resume decoding process again.

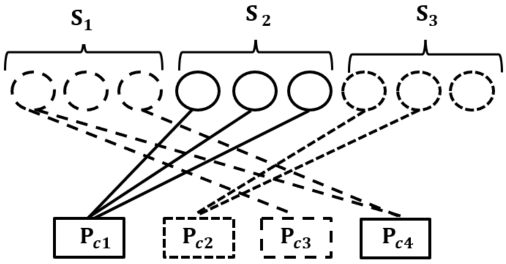


Figure 1. Data packets selection for LDHS-LT code

3. LDHS-LT DECODER

The nature of the fountain codes enables the decoder to starts its job after collecting sufficient number (N) of coded packets. This number mainly characterized as ($N = k(1+\in)$, where ($\in$) represent the amount of overhead coded packets. The decoder has three stages to fully recover the data file packets. These stages are described.

3.1. Copying of degree one

The decoder fetch for the most valuable coded packet which is the degree one. These coded packets are stored in the ripple box. The size of this ripple is a compromise issue because getting large means that encoding require large number of coded packets to cover all the data packets and this will increase the overhead that required to recover the data file. On the other hand, decreasing the ripple size means facing decoding stuck by loosing degree one coded packet quickly. So, in this step simply the value of degree one coded packet will be copied to the connected data packet, for instance in our Figure 1, that's mean:

$$\widehat{P_{d1}} = P_{c3} \tag{2}$$

3.2. Updating the adjacent coded packets

In this stage, the decoder updates the connected coded packets values by using the recovered data packet value. Again, referring to Figure 1, we had estimated the value of P_{d1} but this packet is connected to P_{c4} also, so updating its value by:

$$P_{c4} = P_{c4} \oplus p_{d1} \tag{3}$$

3.3. Erasing all participated edges

All the edges that connected to the evolved coded packet in a single decoding step are removed. In our Figure 1, two edges are erased one for connection between (p_{d1}, P_{c3}) and the other that connect (p_{d1}, P_{c4}). After that, it is clear that P_{c4} has single connection with p_{d3} which would continue the decoding process for the next round.

3.4. Breaking the decoding stuck

Sometimes, the ripple box is empty during the decoding process or even in the beginning of it. The ripple box is the place that the decoder can find a degree-one coded packet in it. There are many reasons that cause the ropple box to vanish. One of the main reasons is the bad channel conditions that causes to lost some of the transmitted coded packets. Another possible reason is the bad generator matrix or degree generation that make a critical number for degree-one coded packts. In such cases, the decoder fetches for certain patterns of coded packets that when ex-ored together this will produce new degree one coded packet and the operation of the decoder resumes. Our LDHS-LT encoding offer many patterns of such coded packets that prevents decoding stuck.

These four prementioned stages which represent the decoder job have to be continued till recovering all the data packets. In many times, the decoder succeeded in its job which in turn an acknowledgment message will be sent to the source that indicates a successful decoding. However, if the decoder stucks just as mentioned in section 3.4, it means that the decoder needs to make further processing or collect more coded packets to have new degree-one coded packet to start with and resume decoding.

4. SIMULINK MODEL FOR (LDHS-LT) ENCODER

The core idea of the Simulink design is shown in Figure 2. Two main random generators are used to generate the degree (d) and the data segment (S), while the data packets are selected sequentially. That's mean, after having a number from degree generator and a number represents the assigned sgment it will be determined for the encoder to take a serial number (degree) of data packets from the assigned segment to join them using ex-or oprations.

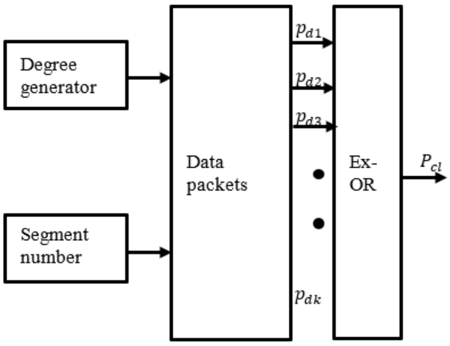


Figure 2. Block diagram for LDHS-LT encoder

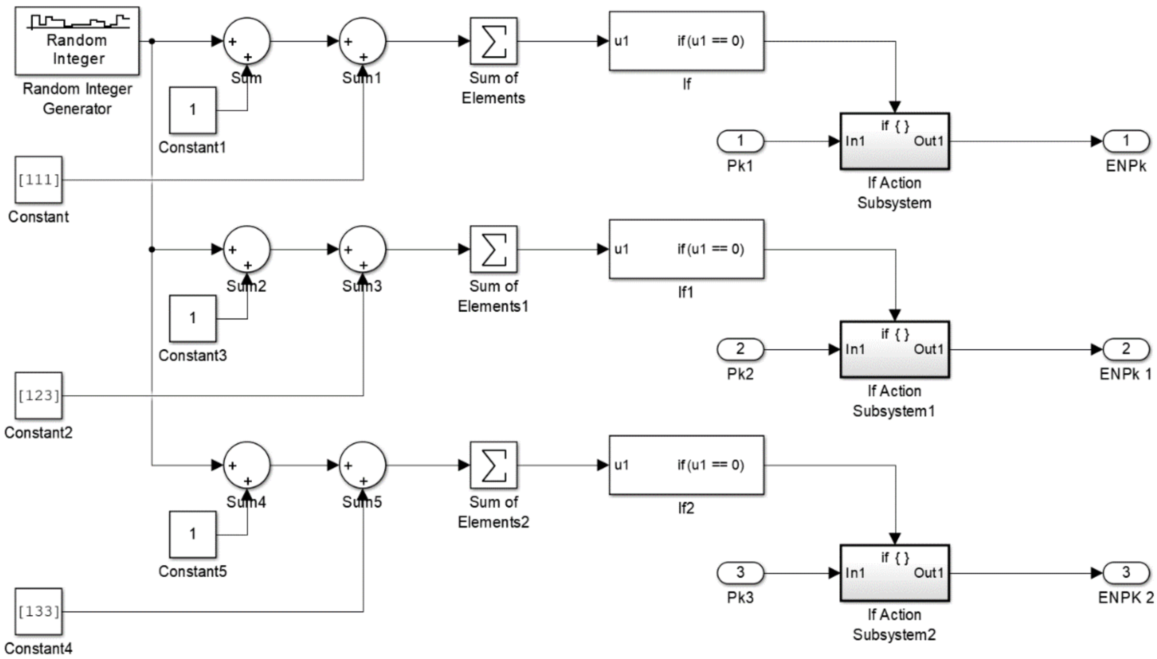


Figure 3. Simulink design for degree 1 coded packets

To demonestrate the idea of Figure 2, let's return to our prototype case of the size ($k = 9$), so the random generation of the degree is ($d \in \{1,2,3\}$) and also the segments are generated randomly as ($S \in \{1,2,3\}$). Table 1, lists some possible sets for these generators. For the first coded packet, it is generated from combining three sequented data packets in the second segment. While for the coded packet no. 4, and according to the random generation, it is created from combining the first two data packets in the second segment. The Simulink model for generation of three coded packets for three different degrees and segments are illustrated in Figure 3, Figure 4, and Figure 5.

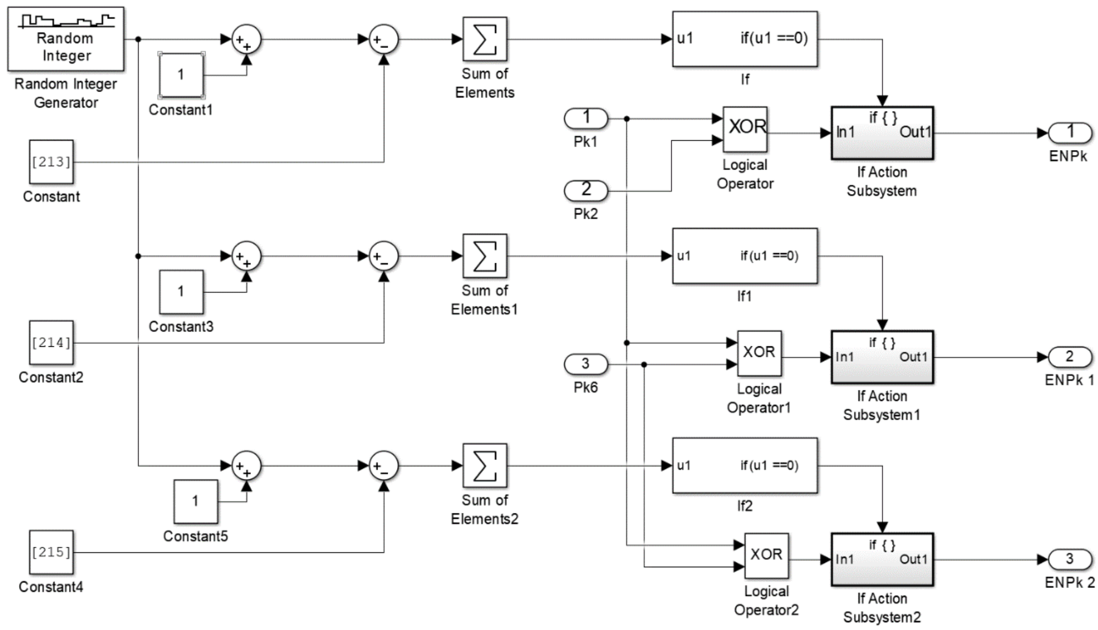


Figure 4. Simulink design for degree 2 coded packets

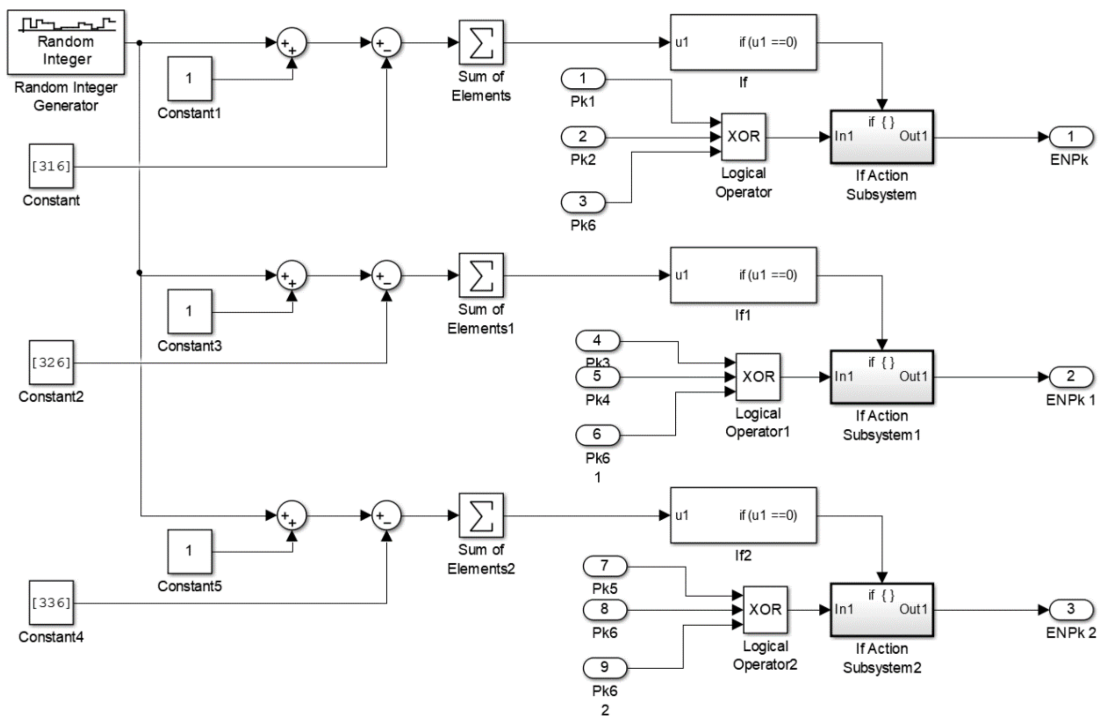


Figure 5. Simulink design for degree 3 coded packets

Table 1. Generation sets of LDHS-LT encoder

Degree generator	Segment generator	Selected data packets
3	2	4,5,6
1	1	1
3	1	1,2,3
2	3	7,8

5. RESULTS AND DISCUSSION

The performance of the proposed LDHS-LT encoder is tested with respect to the ability of data recovery, the amount required overhead as well as the average packet operations. The performance comparison is made with the conventional RSD-LT encoder by using a data file contains (40) packets, keep in mind that each packet may has any number of bits according to the application, while in our simulation each packet is represented by single bit. The test is made for (1000) run and the comparison is made to test several parameters. The percent of file recovery ($P_{Success}$) which represent the average recovered data packet from the total sent packlets. The amount of additional (overhead) packets required for full recovery success ($OVHD_P$).

Let's first define the RSD characteristics, this distribution is mainly made for large data files and is given by:

$$\mu(d) = \frac{\tau(d) + \rho(d)}{\beta} \tag{4}$$

Where the normalization factor $\beta = \sum_{d=1}^{k}(\,\rho(d) + \tau(d))$ and $\tau(d)$ is given as:

$$\tau(d) = \begin{cases} \frac{R}{dk}, d = 1,2,..,\frac{k}{R} - 1 \\ \frac{R}{k} ln\frac{R}{\delta}, d = \frac{k}{R} \\ 0, d = \frac{k}{R} + 1, \ldots\ldots, k. \end{cases} \tag{5}$$

Where $R = c.ln(k/\delta)\sqrt{k}\,)$ is the average amount of the degree 1 coded packets. while c represents a constant has a positive quantity usually is less than 1. The letter δ is an indication for the probability of decoder stuck and failure.

The test is made for very short data length of size (40 packets). The challenge of such length is required especially for its application in the field of smart cities and their wireless sensor networks [13]-[20]. Table 2 shows the result for the simulation test of our LDHS-LT code with parameters (d = 1, 2, 3 or 4) and (S = 10), compared with traditional RSD-LT code with parameters (c = 0.02, c = 0.02, δ = 0.1). Table 2 shows the results for a scenario of erasure probability of (α = 0.5).

According to Table 2, for (1000) run each run afile of (k = 40 packets) is sent. The decoder try to recover the sent packets gradually starting from (N = 40 packets) ending with (N = 90 packets). The results shown for best performance, LDHS-LT code succeeds to get all the packets for (960 run) and when it fails to do so there is only (1 lost packet) in average, and this is done with an extra packet of the order of (20 packets). On the other hand, the traditional RSD-LT did its best with total recovery of (907 run) and the remaining run has an average of (2 lost packets) with an overhead of (26 packets).

Table 2. Data recovery performance

Code type	Full recovery	Unrecovered packets	Average degree	Overhead percent
LDHS-LT	960	1	2	20
RSD-LT	907	2	2	26

6. CONCLUSION

Fountain codes have an opportunity of being an adaptive channel codes for their property of variable code length. LT codes are designed for large data files and due to thet the degree generation for the coded packets are generated using certain random distribution. The new design for short length LDHS-LT code had approved its superiority for file recovery performance with respect of full data recovery and average unrecovered packets and amount of overhead. The Simulink design for such code is made for prototype one and could represents the first step in hardware design. An FPGA design for our LDHS-LT is thye next step to present the code for practical applications.

REFERENCES

[1] F. Li and H. Imai, “An overview of forward error-correction coding for spread spectrum,” *Proceedings of ISSSTA'95 International Symposium on Spread Spectrum Techniques and Applications*, 1996, pp. 1193-1197, vol. 3, doi: 10.1109/ISSSTA.1996.563494.

[2] J. W. Byers, M. Luby, and M. Mitzenmacher, “A digital fountain approach to asynchronous reliable multicast,” in *IEEE Journal on Selected Areas in Communications,* vol. 20, no. 8, pp. 1528-1540, 2002, doi: 10.1109/JSAC.2002.803996.

[3] R. Karp, M. Luby, and A. Shokrollahi, “Finite length analysis of LT codes,” *International Symposium on Information Theory, 2004. ISIT 2004. Proceedings.*, 2004, doi: 10.1109/ISIT.2004.1365074.

[4] J. Boiko, I. Pyatin, and O. Eromenko, “Design and evaluation of the efficiency of channel coding LDPC codes for 5G information technology,” *Indonesian Journal of Electrical Engineering and Informatics (IJEEI),* vol. 9, no. 4, 2021, doi: 10.52549/ijeei.v9i4.3188.

[5] W. Alam, D. Sarma, R. J. Chakma, M. J. Alam, and S. Hossain, “Internet of things based smart vending machine using digital payment system,” *Indonesian Journal of Electrical Engineering and Informatics (IJEEI),* vol. 9, no. 3, 2021, doi: 10.52549/.v9i3.3133.

[6] E. Y. T. Adesta, D. Agusman, and Avicenna, “Internet of Things (IoT) in Agriculture Industries,” *Indonesian Journal of Electrical Engineering and Informatics (IJEEI),* vol. 5, no. 4, 2017, doi: 10.52549/ijeei.v5i4.373.

[7] C. V. Nguyen *et al*., “ZigBee based data collection in wireless sensor networks,” *International Journal of Informatics and Communication Technology (IJ-ICT),* vol. 10, no. 3, pp. 211-224, 2021, doi: 10.11591/ijict.v10i3.pp212-224.

[8] S. A. Alabady, “An efficient forward error correction code for wireless sensor networks,” *International Journal of Informatics and Communication Technology (IJ-ICT),* vol. 10, no. 2, pp. 104-115, 2021, doi: 10.11591/ijict.v10i2.pp104-115.

[9] J. A. Aldhaibani, and N. A. Al-Shareefi, “Free Space Optics Backhaul Link For Small Cells Of 5G Cellular Networks,” *Journal of Engineering Science and Technology,* vol. 15, no. 3, pp. 1685-1697, 2020. [Online]. Available: https://jestec.taylors.edu.my/Vol%2015%20issue%203%20June%202020/15_3_15.pdf

[10] I. J. Hasan, B. M. Waheib, N. A. J. Salih, and N. I. Abdulkhaleq, “A global system for mobile communications-based electrical power consumption for a non-contact smart billing system,” *International Journal of Electrical and Computer Engineering (IJECE),* vol. 11, no. 6, pp. 4659-4666, 2021, doi: 10.11591/ijece.v11i6.pp4659-4666.

[11] S. A. Abdulhussein and R. A. Abdulhussein, “Handover evaluation of UMTS-WiMAX networks,” *TELKOMNIKA (Telecommunication, Computing, Electronics and Control)*, vol. 19, no. 5, pp. 1456-1465, 2021, doi: 10.12928/telkomnika.v19i5.20411.

[12] N. I. Abdulkhaleq and O. Gazi, “Decoding of LT-like codes in the absence of degree-one code symbols,” *ETRI Journal*, vol. 38, no. 5, pp. 896–902, 2016, doi: 10.4218/etrij.16.0116.0107.

[13] N. I. Abdulkhaleq and O. Gazi, “A sequential coding approach for short length LT codes over AWGN channel,” *2017 4th International Conference on Electrical and Electronic Engineering (ICEEE)*, 2017, pp. 267-271, doi: 10.1109/ICEEE2.2017.7935831.

[14] N. I. Abdulkhaleq, I. J. Hasan and N. A. J. Salih, “Mitigation of Packet Erasure using New Fountain Code Design,” *2018 IEEE International Multidisciplinary Conference on Engineering Technology (IMCET*), 2018, pp. 1-7, doi: 10.1109/IMCET.2018.8603036.

[15] J. K. Zao, M. Hornansky and P. -L. Diao, “Design of optimal short-length LT codes using evolution strategies,” *2012 IEEE Congress on Evolutionary Computation*, 2012, pp. 1-9, doi: 10.1109/CEC.2012.6256147.

[16] Z. -K. Chong, B. -M. Goi, H. Ohsaki, C. -K. Bryan Ng, and H. -T. Ewe, “Design of short-length message fountain code for erasure channel transmission,” *2012 IEEE Conference on Sustainable Utilization and Development in Engineering and Technology (STUDENT),* 2012, pp. 239-241, doi: 10.1109/STUDENT.2012.6408411.

[17] Z. Huang, J. Jiao, K. Zhang, S. Wu, and Q. Zhang, “A Novel Weight Coefficient PEG Algorithm for Ultra-Reliable Short Length Analog Fountain Codes,” in *IEEE Wireless Communications Letters*, vol. 8, no. 1, pp. 125-128, 2019, doi: 10.1109/LWC.2018.2863289.

[18] L. -J. Chang, C. -H. Wang, and J. K. Zao, “An error-floor reduction technique for short-length LT codes,” *2014 International Symposium on Information Theory and its Applications*, 2014, pp. 279-283. [Online]. Available: https://ieeexplore.ieee.org/document/6979848

[19] H. Cheong, J. Eun, H. Kim, and K. J. Kim, “Belief propagation decoding assisted on-the-fly Gaussian elimination for short LT codes,” *Cluster Computing*, vol. 19, pp. 309–314, 2016, doi: 10.1007/s10586-015-0522-0.

[20] D. -D. Yan, X. -K. Fan, Z. -Y. Chen, and H. -Y. Ma, “Low-loss belief propagation decoder with Tanner graph in quantum error-correction codes,” *Chinese Physics B*, vol. 31, no. 1, 2022, doi: 10.1088/1674-1056/ac11cf.

[21] A. A. Zamzami, K. -C. Chao, and S. -C. Lin, “Stopping-Set Assisted Reinforced Belief Propagation for Decoding Short LDPC Codes,” in *IEEE Communications Letters,* vol. 26, no. 3, pp. 505-508, 2022, doi: 10.1109/LCOMM.2021.3137222.

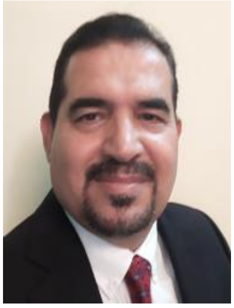
[22] F. Z. Zenkouar, M. E. Alaou, and S. Najah, “GF(q) LDPC encoder and decoder FPGA implementation using group shuffled belief propagation algorithm,” *International Journal of Electrical and Computer Engineering (IJECE)*, vol. 12, no. 3, pp. 2184-2193, 2022, doi: 10.11591/ijece.v12i3.pp2184-2193.

[23] S. -Y. Zhang and B. Shahrrava, “Belief propagation polar decoding for wireless communication systems with noisy channel estimates,” *Wireless Personal Communications*, vol. 123, pp. 1365–1377, 2022, doi: 10.1007/s11277-021-09184-7.

[24] R. Sun, M. Zhao, J. Liu, J. Sun, H. Guan and Z. Zhao, “Piggybacking Belief Propagation Decoding for Rateless Codes Based on RA Structure,” *2018 IEEE/CIC International Conference on Communications in China (ICCC),* 2018, pp. 237-241, doi: 10.1109/ICCChina.2018.8641154.

[25] P. Shi, Z. Wang, and D. Li, “Low overhead growth degree coding scheme for online fountain codes with limited feedback,” in *International Conference on Wireless and Satellite Systems*, 2022, pp. 456–467, doi: 10.1007/978-3-030-93398-2_43.

BIOGRAPHIES OF AUTHORS



Nadhir Ibrahim Abdulkhaleq had his BSC in Electrical Engineering from Cairo-Egypt in 1990 and MSc. in Radar and Communication from the Technology University in Baghdad-Iraq in 1995, the PhD in Electronic and Communication Engineering from Cankaya University in Turkey in 2017. He currently works as an Asst. Prof. in Department of Mobile Computing and Communications Engineering-College of EngineeringUniversity of Information Technology&Communications (UoITC). He had published several papers in coding theory and wireless sensor networks. He can be contacted at email: nadhir.abdulkhaleq@uoitc.edu.iq.

Nahla Abdul Jalil Salih is a member of the department of Electronic Techniques in the Institute of Technology in Middle Technical University in Iraq. she received his BSc in control and systems Engineering from university of technology, Baghdad, Iraq in 1992, her interests are control and systems, fractal coding technique, digital image coding. She can be contacted at email: nahla.alchalaby@yahoo.com.

Rajaa Salih Mohammed Hasan is a member of department of forensic techniques in the Institute of Technology at The Middle Technical University in Iraq. She received her BSC in Statistics from University of Baghdad Iraq in 1997, she received her MSC in Information Technology from University of Technology Bagdad Iraq in 2005 her interests is new trends in Information Technology. She can be contacted at email: Raja.Salih14@yahoo.com.

Ihsan Jabbar Hasan was born in Baghdad in1966. He received BSc in electrical engineering from Cairo, Egypt in 1990 and MSc in electrical power engineering from the Foundation of Technical Education-College of Electrical and Electronic Techniques, Baghdad, Iraq in 2007, the PhD degree from the Universiti Teknikal Malaysia Melaka (UTeM) in 2015. He currently works as an Asst. Prof. in the Middle Technical College, Baghdad, Iraq. He had published several papers in the field of power distribution network, demand response techniques, power system planning, network planning and optimization. He can be contacted at email: ihsan_hssn@yahoo.com.