International Journal Of Mathematics And Statistics Invention (IJMSI)
E-ISSN: 2321 – 4767 P-ISSN: 2321 - 4759
Www.Ijmsi.Org Volume 2 Issue 09 || September. 2014 || PP-16-18

Decoding Parity Code Error(S): Analysis Of Algorithmic Method Of Hamming Code Techniques On Integer Sequence A128984.

1,Afolabi Godfrey, 2,Zaid Ibrahim , 3,Muhammad .S. Magami & 4,Ibrahim A.A
Mathematics Department, Joda International School, P.M.B 1031, Nigeria.
Department of Mathematics, Usmanu Danfodiyo University, Nigeria.

ABSTRACT: *The Algorithmic method of Hamming code techniques was discussed in this paper. Since the probability of error(s) occurring in any bit position of encoded data is the same, decoding parity code error(s) was therefore the major area of concern in this work. Some selected integers from integer sequence A128984 were analyzed and consequently, useful results were obtained which will form a base for ascertaining computational error(s) in binary codes.*

KEY WORDS*: Algorithmic method, Encoded data, Hamming codes and Integer sequence A128984.*

I. INTRODUCTION

Most often than none, not all computations are without error(s). There are quite a number of reliable codes that can be used to handle such error(s) when they occur. This is simply to make results obtained from computations more reliable and efficient wherever and whenever in use. Hamming code is one of such codes that can be used to encode results obtained [1]. Hamming code makes use of the concept of parity and parity bits, which are bits added to the original result (data) to form encoded data. This is to ascertain the efficiency and reliability of the result (data) being checked. The bit positions are numbered in sequence from 1 to $n + k$ (k-parity bits, n-bit data) with the positions numbered with powers of two(s) reserved as the parity code bits while the other positions as data bit [2]. Among the Hamming code techniques is the Algorithmic method [3]. The computation in this paper is on some selected integers from the integer sequence A128984 [4]. This A128984 is constructed using a degree of the special variety of sub-graphs of cayley graphs in order for a study of degree/ diameter problem [5] and they are 2, 4, 6, 10, 12, 16, ... [6]. The results from this analysis shall in great measure contribute to the general knowledge of coding theory.

II. DEFINITION OF BASIC TERMS

The following terms as used in this paper are defined to make the work self contained:

[1] Algorithmic method: The algorithmic method involves the following steps [7]; (a) number the bits starting from 1 (b) write their respective binary equivalences (c) mark all bit positions that are powers of two (2^n, n = 0, 1, …) as parity code bits (d) mark all other bit positions other than the powers of two as data bits (e) for each parity bit position, calculate the parity code, the position of the parity code bit determine the sequence of bits that it alternately checks and skips and (f) set parity bit to '1' if the total numbers of 1's in the positions it checks is odd, otherwise set it to '0' if even. Thus, this gives the parity code.

[2] Encoded data: This is the addition of parity (correction) bit to the original data (result) obtained to ascertain the occurrence of error(s) [8].

III. METHOD AND PROCEDURE

The Algorithmic method of encoding data involves the following steps [9]:

[1] Number the bits starting from 1: bit 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, etc.
[2] Write the binary equivalence of the bits numbered in (1) above: 1, 10, 11, 100, 101, 110, 111, 1000, etc.
[3] Mark all bit positions that are powers of two as parity bits: 1, 2, 4, 8, 16, 32, 64, 128, etc.
[4] Mark all other bit positions other than powers of two as data to be encoded: 3, 5, 6, 7, 9, 10, 11, 12, 13, etc.
[5] For each parity bit calculates the parity for some of the bits in the code word. The position of the parity bit determines the sequence of bits that it alternately checks and skips:
[6] Parity position 1: start with 1, check 1 bit, skip 1 bit, etc (1, 3, 5, 7, 9, 11, 13, 15, 17, 19, 21, 23, 25, etc).
[7] Parity position 2: start with 2, check 2 bits, skip 2 bits, etc (2, 3, 6, 7, 10, 11, 14, 15, 18, 19, 22, 23, 26, etc).
[8] Parity position 4: start with 4, check 4 bits, skip 4 bits, etc (4, 5, 6, 7, 12,13,14,15, 20, 21, 22, 23, etc).
[9] Parity position 8: start with 8, check 8 bits, skip 8 bits, etc (8-15, 24-31, 40-47, 56-63, etc).
[10] Parity position 16: start with 16, check 16 bits, skip 16 bits, etc (16-31, 48-63, 80-95, 112-127, etc).

[11] Parity position 32: start with 32, check 32 bits, skip 32 bits, etc (32-63, 96-127, 160-191, 224-255, etc),etc.

[12] Set a parity bit to '1' if the total number of 1's in the positions it checks is odd. Otherwise, set a parity bit to '0' if the total number of 1's in the position it checks is even. Thus, the parity code is obtained and an encoded data is formed when the parity codes are placed in their respective positions.

a. These encoded data (results) are either transmitted, used or stored for future purposes with ascertain probability of error(s) sure. If the sent data (results) differs from the one received, signifying that an error has occurred, their respective parity code bits are compared and the bit positions added. The sum of the addition gives the position of the corrupted bit in the data (result) received and traceable in the parity code bit. To correct this error, flip the corrupted bit either from '0' to '1' or from '1' to '0' as the case may be.

IV. DATA PRESENTATION

The encoded data (results) obtained from the analysis on some selected integers from the integer sequence A128984 are shown in table 4.0 below:

Table 4.0 (Encoded data obtained from the analysis on some selected integers from the integer sequence *A128984*)

Selected Integer	Binary Equivalence (code)	Parity Code	Encoded Data
2	10	**110**	**1**1**1**00
4	100	**110**	**1**1**1**000
6	110	**011**	**01**111**0**
10	1010	**101**	1**0**11**01**0
12	1100	**011**	**01**1**1**100
…	…	**…**	…

Source: [10]

The respective parity codes and encoded data of the selected integers from the sequence *A128984* were obtained by using the Algorithmic method as follows:

2: [10]: x x 1 x 0, position 1: check bits 1, 3, 5: we have, ! x 1 x 0: odd parity so set position 1 to a '**1**'. Position 2: check bits 2, 3: we have, **1** ! 1 x 0: odd parity so set position 2 to a '**1**'. Position 4: check bits 4, 5: we have, **1 1** 1 ! 0: even parity so set position 4 to a '**0**'. The parity code is **1 1 0**, thus the encoded data is **1 1** 1 **0** 0.

4: [100]: x x 1 x 0 0, position 1: check bits 1, 3, 5: we have, ! x 1 x 0 0: odd parity so set position 1 to a '**1**'. Position 2: check bits 2, 3, 6: we have, **1** ! 1 x 0 0: odd parity so set position 2 to a '**1**'. Position 4: check bits 4, 5. 6: we have, **1 1** 1 ! 0 0: even parity so set position 4 to a '**0**'. The parity code is **1 1 0** and the encoded data is **1 1** 1 **0** 0 0.

6: [110]: x x 1 x 1 0, position 1: check bits 1, 3, 5: we have, ! x 1 x 1 0: even parity so set position 1 to a '**0**'. Position 2: check bits 2, 3, 6: we have, **0** ! 1 x 1 0: odd parity so set position 2 to a '**1**'. Position 4: check bits 4, 5. 6: we have, **0 1** 1 ! 1 0: odd parity so set position 4 to a '**1**'. The parity code is **0 1 1**, thus the encoded data is **0 1** 1 **1** 1 0.

10: [1010]: x x 1 x 0 1 0, position 1: check bits 1, 3, 5, 7: we have, ! x 1 x 0 1 0: odd parity so set position 1 to a '**1**'. Position 2: check bits 2, 3, 6, 7: we have, **1** ! 1 x 0 1 0: even parity so set position 2 to a '**0**'. Position 4: check bits 4, 5. 6, 7: we have, **1 0** 1 ! 0 1 0: odd parity so set position 4 to a '**1**'. The parity code is **1 0 1**, thus the encoded data is **1 0** 1 **1** 0 1 0.

12: [1100]: x x 1 x 1 0 0, position 1: check bits 1, 3, 5, 7: we have, ! x 1 x 1 0 0: even parity so set position 1 to a '**0**'. Position 2: check 2, 3, 6, 7: we have, **0** ! 1 x 1 0 0: odd parity so set position 2 to a '**1**'. Position 4: check 4, 5, 6, 7: we have, **0 1** 1 ! 1 0 0: odd parity so set position 4 to a '**1**'. The parity code is **0 1 1**, thus the encoded data is **0 1** 1 **1** 1 0 0.

V. ANALYSIS OF DECODING PARITY CODE ERROR(S)

The analysis of decoding parity code error(s) using the algorithmic method of hamming code techniques on some selected integers from integer sequence A128984 against probable erroneous results are shown in table 5.0 below: (EP = Error Position)

Table 5.0 (Results of sent encoded data and received encoded data of integer sequence *A128984*)

Selected Integer	Binary Equivalence (code)	Parity Code Sent	Encoded Data Sent	Parity Code Received	Encoded Data Received	EP
2	10	**110**	**1**1**1**00	**111**	**11110**	**4**
4	100	**110**	**1**1**1**000	**100**	**101000**	**2**
6	110	**011**	**01**111**0**	**010**	**011010**	**4**
10	1010	**101**	1**0**11**01**0	**111**	1111010	2
12	1100	**011**	**01**1**1**100	**111**	**1111100**	**1**
…	…	**…**	…	…	…	…

Source: Researchers' computation

From TABLE 5.0 above:
{2}: The sent encoded data differs from the received encoded data, by comparing their respective parity codes and adding the bit positions, we have 0 + 0 + 4 = 4. Thus, position 4 of the received data (information) is erroneous which equivalent is parity code position 3. To correct it therefore, flip it from '**1**' to '**0**'.
{4}: The sent encoded data differs from the received one, by comparing their respective parity codes and adding its parity bit positions, we have: 0 + 2 + 0 = 2. Thus, position 2 of the received data (information) is corrupted and is equivalent to parity code position 2. To correct it therefore, flip it from '**0**' to '**1**'.
{6}: The sent encoded data is different from that received, by comparing their respective parity codes and adding its bit positions, we have: 0 + 0 + 4 = 4. Thus, bit position 4 of the received data (information) is corrupted and is equivalent to parity code position 3. To correct it, flip it from '**0'** to '**1**'.
{10}: The sent encoded data differs from the one received, thus we compare their respective parity codes and add the bits position. We have, 0 + 2 + 0 = 2. Bit position 2 in the message received as corrupted, this is also bit position 2 in the parity code received. To correct it therefore, flip it from '**1**' to '**0**'.
{12}: The sent encoded data differs from the one received, thus we compare their respective parity codes and add the bits position. We have, 1 + 0 + 0 = 1. This imply that bit position 1 of the received data (information) was corrupted which is also the position of sent parity code bit that as altered. To correct it therefore, flip it from '**1**' to '**0**'.

VI. CONCLUSION

Generally, since the results obtained from any analysis is either for present or future purposes, therefore, encoding such results in other to ascertain any alterations in form of error(s) when they occur becomes of great importance. More so, since the probability of alteration in form of error(s) on any bit of the original encoded data (information) is the same, therefore, the results of using the Algorithmic method of Hamming code techniques in decoding parity code error(s) with analysis on some selected integers from integer sequence A128984 as presented in this paper, which can conveniently apply to any analysis in binary operation, has further contributed to the general knowledge of Error Detection and Correction.

REFERENCES

[1] I. Koren, *Computer arithmetic algorithms* (Natrick MA): A. K. Peters, 2002.

[2] R. W. Hamming, Bell system Technology, *Journal of Error Detecting and Correcting Codes vol. 29, April, 1950, pp. 147-160.*

[3] M. Subhasish, and J. M. Edward, "*Which concurrent error detection scheme to choose?"* Center for Reliable Computing, Stanford University, 2000.

[4] A. A. Ibrahim, Mathematics Association of Nigeria, *the journal On the Combinatorics of A Five-element sample Abacus of vol. 32, 2005, No. 2B: 410-415.*

[5] A. A. Ibrahim, *A Counting Scheme And Some Algebraic Properties of A class of Special Permutation Patterns, Journal of Discrete Mathematcal Sciences and Cryptography, Taru Publishers, vol. 10, 2007, No. 4: 537- 546, New Delhi, India*.

[6] A. A. Ibrahim and M. S. Audu, *Some Group Theoretical Properties of Certain Class of (123) and (132)-Avoiding Patterns of Numbers: An Enumeration Scheme: An Enumeration Scheme, African Journal of Natural Sciences, vol. 8, 2005, 2 (4): 334 – 340.*

[7] A. Nirandi, *"Computational complexity of a longest path algorithm with a recursive method,* 2010.

[8] J. F. Ziegler, *"Automatic Recognition and Classification of Forward Error Correcting Codes". An M.Sc Thesis submitted to the faculty of Information Technology and Engineering of George Mason University.* Spring 2000, George Mason University Fairfax, Virginia.

[9] B. Sklar, "*Digital Communication: Fundamentals and Applications".* (Second Edition, Prentice-Hall, 2001).

[10] A. Godfrey & A. A. Ibrahim, *IOSR Journals of Mathematics (IOSR- JM) e – ISSN: 2278, p – ISSN: 2319 – 7676. Volume 9, Issue 2 (Nov. – Dec. 2013) pp 33 -37.* www.iosrjournals.org.