View metadata, citation and similar papers at core.ac.uk

brought to you by CORE

provided by Institute of Advanced Engineering and Science

International Journal of Electrical and Computer Engineering (IJECE)
Vol. 7, No. 1, February 2017, pp. 521~525
ISSN: 2088-8708, DOI: 10.11591/ijece.v7i1.pp521-525

New Heuristic Model for Optimal CRC Polynomial

Ahmed Salih Khirbeet, Ravie Chandren Muniyandi
Research Center for Software Technology and Management, Faculty of Technology and Information Science, University Kebangsaan Malaysia, Malaysia

Article Info

Article history:

Received Oct 23, 2016
Revised Dec 14, 2016
Accepted Dec 30, 2016

Keyword:

Cyclic redundancy codes
Data authentication
Data integrity
Genetic algorithms
Optimization

ABSTRACT

Cyclic Redundancy Codes (CRCs) are important for maintaining integrity in data transmissions. CRC performance is mainly affected by the polynomial chosen. Recent increases in data throughput require a foray into determining optimal polynomials through software or hardware implementations. Most CRC implementations in use, offer less than optimal performance or are inferior to their newer published counterparts. Classical approaches to determining optimal polynomials involve brute force based searching a population set of all possible polynomials in that set. This paper evaluates performance of CRC-polynomials generated with Genetic Algorithms. It then compares the resultant polynomials, both with and without encryption headers against a benchmark polynomial.

Copyright © 2017 Institute of Advanced Engineering and Science. All rights reserved.

Corresponding Author:

Ahmed Salih Khirbeet,
Research Center for Software Technology and Management,
Faculty of Technology and Information Science,
University Kebangsaan Malaysia, Malaysia.
Email: ahmed.salih89@siswa.ukm.edu.my

1. INTRODUCTION

Cyclic Redundancy Check (CRC) codes are a highly recognized data integrity mechanism in digital communications for industrial use. They act as a first line of defense in detecting data packet corruption between two nodes in a communication link [1]. Various CRC implementations exist in different applications, e.g. from embedded network applications [1] to wireless networks [2], to communication protocols such as USB 3 [3]. Their wide acceptance is due to their simplicity, good encoding and decoding performance, and good levels of error detection [4]. As data throughput increases, however, it becomes important to utilize optimized CRC implementations.

CRC optimization has been attempted from two angles, namely, polynomial selection, and hardware implementation. The latter is due to the constraints of traditional CRC implementations caused by increasing data rates. Different hardware implementations have been attempted to cope with this.

An attempt for better polynomial selection from a hardware perspective was described by [5]. They developed a pipelined implementation of polynomial division which helped form the basis of an effective CRC. It improved speed, while allowing for data rates from 1 Gbits/s to 4 Gbits/s on field-programmable gate array (FPGA) implementations with respect to the parallelization level (from 8 to 32 bits). However, faster data rates, i.e. 5 Gbits/s, were achieved by an 8-bit parallel CRC-32 proposed by [3] to match the high throughput of USB 3.0. It also proved that an 8-bit parallel CRC-32 was much faster than its serial counterpart. Despite the improvement, it is overshadowed by research done by [6] who propose a CRC circuit usable for 10Gbp, utilizing the state-space transformation method.

Improving CRC implementation from a software perspective requires increasing optimization of a population of polynomials or finding other ways such as reducing the initial search population set of polynomials. Choosing different polynomials for certain data word lengths and bit lengths offer different levels of optimization. A polynomial selection process for embedded network applications and a set of "good" general-purpose polynomials has been described by [1]. A collection of 35 new polynomials along with 13 previously published polynomials that provide good performance for CRCs between 3- and 16-bit for data word lengths up to 2048 bits were listed. It also discusses a couple of comparison case studies between a set of current protocols namely USB with ITU (for 5-bit CRCs) and different 8-bit polynomials. The case studies are used to prove how using different polynomials affect the performance of error detection and how higher levels of optimization may be achieved by using newer published CRC polynomials. The paper then provides an exhaustive survey of CRC polynomials from 3- to 15-bit along with a discussion of 16-bit polynomials. The paper also defines "good" polynomials as those achieving the maximum Hamming Distance for the longest data word along with other considerations. Despite the comprehensive nature of this research, CRC codes beyond 16 parity bits were avoided due to the large number of possibilities that had to be investigated. However, certain classes of 24- and 32-bit CRC codes were investigated by [7], whereas a comprehensive search for CRC codes with 32 parity bits was undertaken by [8].

Similar to [1], an analysis of CRC code standards, namely, CRC-12, ANSI and CCI7T X.25, proved that the standards were not on par with the best possible CRC implementations and that the situation seemed unlikely to change due to economic concerns [4]. This coupled with the use of CRCs in high-throughput applications such as USB 3 [3], has made parallel CRC circuitry design, an important area of research [9].

Performance of CRC implementations created by polynomials with a degree of 16 and above for error detection in communication systems was evaluated by [4]. The properties of minimum distance, "properness" and undetected error probability for binary symmetric channels (BSC) are calculated and compared with existing standards.

The core of CRC implementations depends on the polynomial chosen. Despite their importance in ensuring data integrity, a comprehensive research disclosed that most published CRC polynomials are either optimal when limited to certain message lengths or inferior to alternatives. The latter is supported by the fact that many applications utilize CRCs that offer much less error detection capability than possible for a certain CRC bit value. This may also be supported by the lack of "tools and data tables" for measuring polynomial performance [1].

With regards to polynomial selection, the current landscape suffers from three issues: (1) lacunae exist in the current published polynomials set (2) no specific guidance exists on the viability of certain polynomials in certain contexts, and (3) lacunae with respect to published quantitative analyses [1]. Research has shown that varying polynomials within a certain size as well as varying the size, produces different results with respect to error detecting performance [4].

2. METHODOLOGY

With regards to embedded networks, an attribute of concern with CRC implementation is the Hamming Distance (HD) [1]. HD is the least bit inversions that have to be integrated into a message to generate an error that is "undetectable by that message's CRC-based Frame Check Sequence". The probability of such a data corruption occurring is small but nonetheless, the least number of bit inversions in order to achieve the aforementioned undetectable errors is crucial in CRC polynomial design [1]; size of bits and data words are also to be considered since they affect error-detection performance [4].

Connected with HD is the attribute, Hamming Weight (HW). For a HW of N, N is the number of undetected error probabilities by a CRC implementation with respect to a specific polynomial. A collection of HWs captures performance for different numbers of bits corrupted in a message at a particular data word length. The first non-zero Hamming weight determines a code’s HD.

The goal of finding a good CRC polynomial is to ensure optimization, i.e., to maximize the HD and minimize the HW. The classical way to find such polynomials is to try all possible cases of polynomials and data bit errors (1-, 2-, 3-, message length errors), and then select the optimal one. This method is suitable for small values of data word lengths and polynomial degree. However, for larger values, computational complexity increases and thus makes solutions impossible on today's computers.

We propose solving the issue of reducing computational complexity by using the concept of Genetic Algorithms (GA), thus treating the problem as an optimization problem. GA is a type of evolutionary computation method, but has no agreed upon definition which differentiates it from other evolution-computation methods. However, there are some features which consistently present themselves in GA methods: “populations of chromosomes, selection according to fitness, crossover to produce new offspring, and random mutation of new offspring.”[10]. Also, genetic algorithms have been applied in various applications in communication [11], [12].

GAs help reduce the population of a certain problem set by performing a directed search [13]. This is in contrast to the classical method, which is random in nature and requires going through all members of the problem set's population.

GA allows movement from one set of data represented in binary ("chromosomes") in the search space to a new one by performing a type of "natural selection", as is present in nature. It utilizes genetics-inspired operators: crossover, mutation, and inversion. Similar to the natural inspiration it was derived from, each "chromosome" consists of "genes". "Genes" in turn represent an instance of a certain "allele", i.e., 0 or 1. "Chromosomes" are selected on their fitness to reproduce; fitter chromosomes on average produce more offspring. The Operators then define how new "chromosomes" are to be selected until an optimal solution is reached [10].

In relation to the case at hand, the highest possible degree for any polynomial is n. Therefore, the length of each chromosome is n and each chromosome is padded with “1”s which represent the constant in the polynomial.

HD has been used as a constraint, so each chromosome that does not satisfy the constraint is rejected. Before starting a search process, the user must determine bit error cases (from 1 to the required value of bit errors), the maximum number of combinations for each case, and the message length. Because binary representation has been used, the values' lower bound is zero (0) while the higher bound is one (1). The values are always integers (GF(2)). The target of the optimization process is to minimize the following function:

$$FV = \sum_{i=1}^{n} \frac{undetectedErrors_i}{Combinations_i} + \frac{1}{HD}$$

The Fitness Value (FV) equation clarifies that increasing the number of used combinations makes the resultant solution nearer to the best solution, but not without increasing the computational complexity.

Data integrity of the data packets transferred can be ensured by using encryption. One of secure cipher operation's principles is diffusion. It is satisfied when, if a single bit of the plain text is changed, every bit of the cipher text changes with a probability of 0.5 and vice versa. This is useful for the scenario at hand as any error that occurs in the message during transmission will cause a large number of errors in the message after decryption which in turn helps the next stage (CRC detector) to reduce the number of undetected errors. The number of undetected errors can be further reduced by ensuring each data packet contains a header which is known to both the sender and receiver. Thus, if the error is undetected using FCS, it can be detected by checking for changes in the header after decryption.

The Advanced Encryption Standard (AES) specification will be used for encryption functions relating to headers. The specification [14] defines it as a block cipher which can be used both ways, i.e., to encrypt information into ciphertext and also to decrypt the resultant ciphertext back into its original form. It is a symmetric-key algorithm, which means that the same key is used for both encryption and decryption functions. It can encrypt and decrypt data in 128-bit blocks by using 128, 192, and 256-bit cryptographic keys. We will be using a 128-bit cryptographic key for our application.

3. RESULTS AND DISSCUSSION

To validate the proposed method, we conducted experiments using the MATLAB environment with its built-in GA toolbox. The first experiment consisted of data packets of 48 bits, and where the highest degree of the generator polynomial was 16. Bit error cases ranged from 1 to 6f bits, and for each case, 10,000 combinations were tested at most.

The HD was chosen to be at minimum, 6. The benchmark chosen for this experiment was the polynomial found by [4], with a hex value of 0xC86C. An analysis by [1] found it to the best amongst many others for a data word size of 48 bits. Results are shown in Table 1.

Table 1. Resultant Polynomial Vs [4], Tested With 10,000 Combinations

HD	Polynomial	1 bit	2 bits	3 bits	4 bits	5 bits	6 bits	No. of Undetected Errors
Resultant 0x8411	5	0	0	0	0	183	1267	1450
[4] 0xC86C	6	0	0	0	0	0	2191	2191

Although the resultant polynomial has a lower HD than [4], the sum of undetected errors in 5-bit and 6-bit cases is lower than undetected errors in [4] for 6 bits. As previously mentioned, increasing the maximum number of combinations makes the solution nearer to the best solution. Therefore, another experiment was conducted; this time, making the maximum number of combinations, 300,000. The results are summarized in Table 2.

Table 2. Resultant Polynomial Vs [4], Tested With 300,000 Combinations

HD	Polynomial	1 bit	2 bits	3 bits	4 bits	5 bits	6 bits	No. of Undetected Errors
Resultant 0x9947	5	0	0	0	0	57	1163	1220
[4] 0xC86C	6	0	0	0	0	0	2191	2191

It is clear that the new resultant polynomial is able to detect more error cases than the previous one, and thus [4]'s as well. The number of undetected error probabilities has decreased significantly from the previous experiment, resulting in 57 compared with 183 from the previous experiment, for 5-bits. The improvement in 6-bits was not significant but nonetheless still better by 104 undetected error probabilities.

In order to validate the effect of encryption in lowering the number of undetected errors, another experiment was conducted. This time, a data word size of 112 was used and the highest possible polynomial degree was 16. Bit error cases ranged from 1 to 6 bits, and 400,000 combinations at most, were tested for each case. The HD constraint was left as is, with a minimum value of 6.

Table 3 shows the resultant polynomial, the number of undetected errors, the number of undetected errors with encryption, and the number of undetected errors with encryption along with a header of one-byte size, where 4 million combinations of 6-bit errors have been randomly chosen and tested. The data word size was 112 bit. The benefit of adding encryption as well as a header can be clearly seen. The number of undetected error probabilities without encryption and a header was 141. Adding encryption reduced it to 60 and adding a header along with encryption decreased it even further, enabling the CRC implementation to catch all possible errors.

Table 3. No. Of Undetected Error Probabilities for the Polynomial 0x8948

Polynomial	CRC	CRC and Encryption	CRC and Encryption with Header
0x8948	141	60	0

4. CONCLUSION AND FUTURE WORK

Our work showed how using a GA approach decreased the number of possible solution candidates for finding an optimal polynomial for use in a CRC implementation. We compared the resultant polynomials against a benchmark [4] for a data word length of 48 bits. We also evaluated the effect of encryption on data headers in a CRC implementation; we found that encryption improves performance of a CRC implementation.

Our evaluation, however, is not comprehensive which leaves room for future work. A more thorough evaluation of using GA for different data word lengths and CRC bit length would add strength to the proposal at hand. Similarly, a thorough benchmark analysis of resultant polynomials using GA for the various data word lengths and CRC bit length would prove the validity of our method across a wider scope of CRC implementations.

ACKNOWLEDGEMENTS

The researchers wish to thank Universiti Kebangsaan Malaysia (UKM) for supporting this work by research grants: Dana Impak Perdana (DIP-2014-037).

REFERENCES

[1] P. Koopman and T. Chakravarty, “Cyclic Redundancy Code (CRC) Polynomial Selection For Embedded Networks”, *Int. Conf. Dependable Syst. Networks*, 2004, pp. 1–11, 2004.

[2] V. Chea, M.V. Martin, and R. Liscano, “Hamming distance as a metric for the detection of crc-based side-channel communications in 802.11 wireless networks”, in *IEEE conference on communications and network security (cns)*, 2015, pp. 218–226.

[3] Y. Wu and Y. Qiu, “The 8-bit parallel crc-32 research and implementation in usb 3.0”, in *International conference on Computer science service system (csss)*, 2012, pp. 1079–1082.

[4] T. Baicheva, S. Dodunekov, and P. Kazakov, “Undetected error probability performance of cyclic redundancy-check codes of 16-bit redundancy”, in *IEE Proceedings - Communications*, vol. 147, no. 5, pp. 253–256.

[5] F. Monteiro, A. Dandache, A. M’Sir, and B. Lepley, “A polynomial division pipelined architecture for crc error detecting codes”, in *The 13th international conference on Microelectronics*, 2001, vol. 13, pp. 133–136.

[6] J.S. Lin, C.K. Lee, M.D. Shieh, and J.H. Chen, “High-speed crc design for 10 gbps applications”, in *IEEE international symposium on circuits and systems*, 2006, p. 4.
[7] G. Castagnoli, S. Brauer, and M. Herrmann, “Optimization of cyclic redundancy-check codes with 24 and 32 parity bits”, *IEEE Trans. Commun.*, vol. 41, no. 6, pp. 883–892, 1993.
[8] P. Koopman, “32-bit cyclic redundancy codes for internet applications”, in *International conference on dependable systems and networks*, 2002, pp. 459–468.
[9] K. Witzke and C. Leung, “A comparison of some error detecting crc code standards”, *IEEE Trans. Commun.*, vol. 33, no. 9, pp. 996–998, 1985.
[10] M. Mitchell, *An introduction to genetic algorithms*. Bradford Books, 1998.
[11] J. Mohammed, “Comparative Performance Investigations of Stochastic and Genetic Algorithms Under Fast Dynamically Changing Environment in Smart Antennas”, *Int. J. Electr. Comput. Eng.*, vol. 2, no. 1, pp. 98–105, 2012.
[12] N. Jiang, S. Jin, Y. Guo, and Y. He, “Localization of Wireless Sensor Network Based on Genetic Algorithm”, *Int. J. Comput. Commun. Control*, vol. 8, no. 6, p. 825, 2013.
[13] S. Sivanandam and S. Deepa, *Introduction to genetic algorithms*. Springer Berlin Heidelberg, 2007.
[14] “Specification for the advanced encryption standard (AES),” *Fed. Inf. Process. Stand. Publ.*, 2001.

BIOGRAPHIES OF AUTHORS

Ahmed Salih Khirbeet
Research Center for Software Technology and Management (SOFTAM)
Faculty of Technology and Information Science
University Kebangsaan Malaysia
ahmed.salih89@siswa.ukm.edu.my

Ravie Chandren Muniyandi
Research Center for Software Technology and Management (SOFTAM)
Faculty of Technology and Information Science
University Kebangsaan Malaysia.
ravie@ukm.edu.my