REDUCING TO FAULT ERRORS IN COMMUNICATION CHANNELS SYSTEMS

[1]Shiv Kumar Gupta and [2]Rajiv Kumar

[1]*Research Scholar Dept. of Computer Science, Manav Bharti University Solan, (H.P.) India*
[2] *Asstt. Professor Dept. of ECE, Jaypee University of Inf. Tech. Wakanghat Distt. Solan (H.P.) India*

ABSTRACT
In this paper we introduce error-control techniques for improving the error-rate performance that is delivered to an application in situations where the inherent error rate of a digital transmission system is unacceptable. The acceptability of a given level of bit error rate depends on the particular application. For examples, certain types of digital speech transmission are tolerant to fairly high bit error rates. Other types of applications such as electronic funds transfer require essentially error-free transmission. For example, FEC is used in the satellite and deep-space communications. A recent application is in audio CD recordings where FEC is used to provide tremendous robustness to errors so that clear sound reproduction is possible even in the presence of smudges and scratches on the disk surface.

KEYWORDS*: ARQ, FEC, Detection System, Parity check code.*

I. INTRODUCTION

In most of the communication channels a certain level of noise and interface is unavoidable. With the advent of digital systems, transmission has been optimized. However, bit errors in transmission will occur with some small but nonzero probability. For example, typical bit error rates for systems that use copper wires are in the order of 10^{-6} i.e. one in a million. Modern optical fiber systems have bit error rates of 10^{-9} or less. In contrast, [3] wireless transmission systems can experience error rate as high as 10^{-3} or worse. There are two basic approaches to error control. The first approach involves the detection of errors and an automatic retransmission request (ARQ) when errors are detected. This approach presupposes the availability of a return channel over which the retransmission request can be made. For example, ARQ is widely used in computer communication systems that use telephones lines. The seconds approach, forward error correction (FEC)[1][5], involves the detection of errors followed by processing that attempts to correct the errors. FEC is appropriate when a return channel is not available, retransmission requests are not easily accommodated, or a large amount of data is sent and retransmission to correct a few errors is very inefficient. Error detection is the first step in both ARQ and FEC. The difference between ARQ and FEC is that ARQ wastes the bandwidth by using retransmission, whereas FEC requires additional redundancy in the transmitted information and incurs significant processing complexity in performing the error correction.

II. DETECTION SYSTEM TECHNIQUES

Here, the idea of error detection has been discussed by using the single parity check code as an example throughout the discussion. As illustrated in Figure 1.1, the basic idea in performing error detection is very simple. The information produced by an application is encoded so that the stream that is input the communication channel satisfies a specific pattern or condition [2][7]. The receiver checks the stream coming out of communication channel to see whether the pattern is satisfied or not. If it is not, the receiver can be certain that an error has occurred and therefore sets an alarm to alert the user. This certainty stems from the fact that no such pattern would have been transmitted by the encoder.

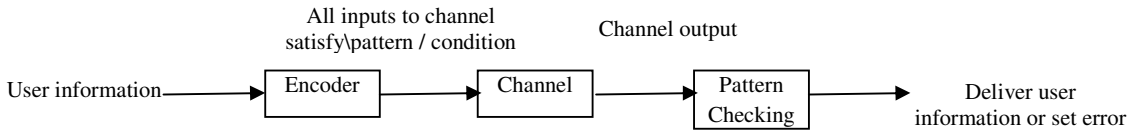


Figure 1.1 General error-detection systems

The simplest code is the single party check code that takes k information bits and appends a single check bit to form a codeword. The parity check ensures that total number of 1's in the codeword is even; that is, the codeword has even party. The check bit in this case is called a parity bit. This error detection is used in ASCII where characters are represented by seven bits and the eighth bit consists of a parity bit. This code is an example of the so-called linear codes because the parity bit is calculated as the modulo 2 sum of the information bits:

$$b_{k+1} = b_1 + b_2 + \cdots + b_k \qquad modulo\ 1$$

Where $b_1, b_2, \ldots, b_k$ are the information bits.

Recall that in modulo 1 arithmetic $0 + 0 = 0, 0 + 1 = 1, 1 + 0 = 1$ and $1 + 1 = 0$. Thus, if the information bits contain an even number of 1s, then the parity bit will be 0; and if they contain an old number, then the parity bit will be 1. Consequently, the above rule will assign the parity bit a value that will produce a codeword that always contains an even number of 1s.

2.1 Single Parity Check Code

This pattern defines the single parity check code. If a codeword undergoes a single error during transmission, then the corresponding binary block at the output of the channel will contain an odd number of 1s and the error will be detected. More generally, if the codeword undergoes an odd number of errors, the corresponding output block will also contain an odd number of 1s. Therefore, the single parity bit allows us to detect all error patterns that introduce an odd number of errors. On the other hand, the single parity bit will fail to detect any error patterns that introduce an even number of errors, since the resulting binary vector will have even parity. Nonetheless, the single parity bit provides a remarkable amount of error-detection capability, since the addition of a single check bit results in making half of all possible error patterns detectable, regardless of the value of k. Figure 1.2 shows an alternative way of looking at the operation of this example. [6][4] At the transmitter a checksum is calculated from the information bits and transmitted along with the information. At the receiver, the checksum is recalculated, based on the received information. The received and recalculated checksums are compared, and the error alarm is set if they disagree.

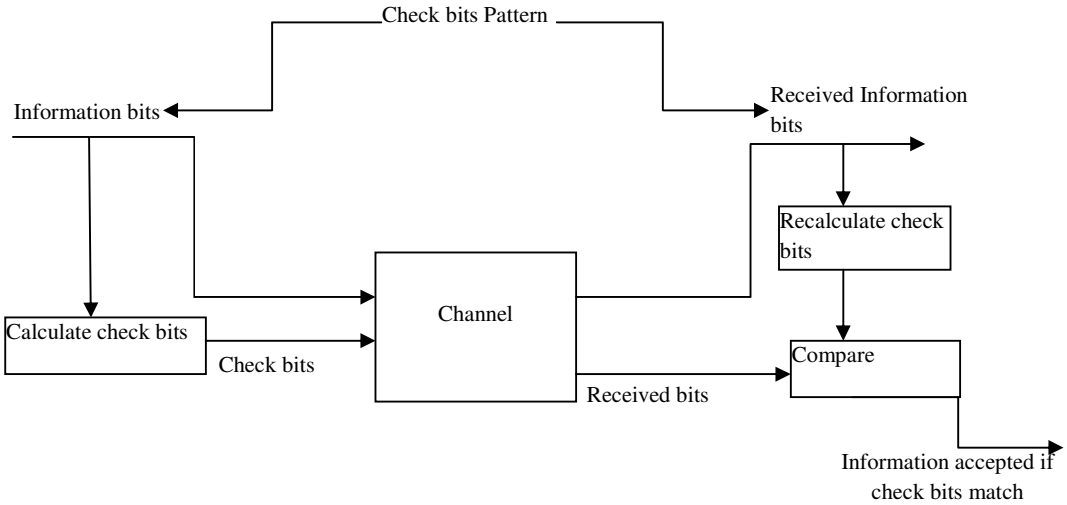
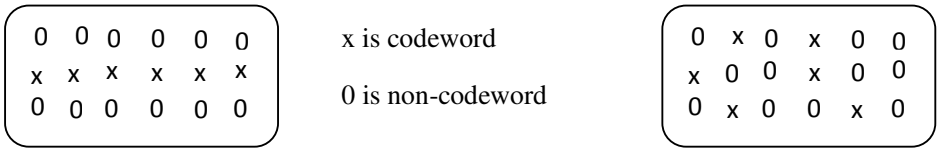


Figure 1.2 Error-detection system using check bits

This simple example can be used to present two fundamental observations about error detection. The first observation is that error detection requires redundancy in that the amount of information that is transmitted is over and above the required minimum. For a single parity check code of length $k+1$, k bits are information bits, and one bit is the parity bit. Therefore, the fraction $1/(k+1)$ of the transmitted bits is redundant.

The second fundamental observation is that every error-detection technique will fail to detect some errors. In particular, an error-detection technique will always fail to detect transmission errors that convert a valid codeword into another valid codeword. For the single parity check code, an even number of transmission errors will always convert a valid codeword to another valid codeword.

The objective in selecting an error-detection code is to select the codeword that reduce the likelihood of the transmission channel converting one valid codeword into another [8]. To visualize how this is done, suppose we depict the set of all possible binary block as the space shown in Figure 1.3, with code words shown by xs in the space and noncodewords by os. To minimize the probability of error-detection failure, we want the code words to be selected so that they are spaced as far away from each other as possible. Thus the code in Figure 1.3a is a poor code because the code words are close to each other. On the hand, the code in Figure 1.3b is good because the distance between code words is maximized. The effectiveness of a code clearly depends on the types of error that are introduced by the channel. We next consider how the effectiveness is evaluated for the example of the single parity check code.



(a). A code with poor distance properties (b). A code with good distance properties

Figure 1.3 Distance properties of codes

2.2 Effectiveness of Error-Detection Codes

The effectiveness of an error-detection code is measured by the probability that the system fails to detect an error. To calculate this probability of error-detection failure, we need to know the probabilities with which various errors occur. These probabilities depends on the particular properties of the given communication channel. We will consider three models of error channels [18]; the random error vector model, the random bit error model, and burst errors.

Suppose us transmission a codeword that has n bits. Defines the error vector $\vec{e} = (e_1, e_2, \ldots, e_n)$ where $e_i = 1$ if an error occurs in the ith transmitted bit and $e_i = 0$ otherwise. In one extreme case, the random error vector model, all 2^n possible error vectors are equally likely to occur. In this channel model the probability of $\vec{e}$ does not depend on the number of errors it contains. Thus the error vector (1, 0, …, 0) has the same probability of occurrence as the error vector (1, 1, … , 1). The single parity check code will fail when the error vector has an even number of 1s. Thus for the random error vector channel model, the probability of error detection failure is 1/2.

Now consider the random bit error model where the bit errors occur independently of each other. Satellite communication provides an example of this type of channel [9][10]. Let p be the probability of an error in a single-bit transmission. The probability of an error vector that has j errors is $p^j(1-p)^{n-j}$, since each of the j errors occurs with probability p and each of the $n-j$ correct transmissions occurs with probability$1-p$. By rewriting this probability we obtain:

$$p[\vec{e}] = (1-p)^{n-w(\vec{e})}p^{w(\vec{e})} = (1-p)^n\left(\frac{p}{1-p}\right)^{w(g)} \quad (1)$$

Where the weight $w(\overrightarrow{e})$ is defined as the number of 1s in $\overrightarrow{e}$. For any useful communication channel. The probability of bit error is much smaller than 1, and so $p < \frac{1}{2}$ and $\left(\frac{p}{1-p}\right) < 1$. This implies that

for the random bit error channel the probability of $\vec{e}$ decreases as the number of errors (1s) increases; that is, an error pattern with a given number of bit errors is more likely than an error pattern with a large number of bit errors. Therefore this channel tends to map a transmitted codeword into binary blocks that are clustered around the codeword.

The single parity check code will fail if the error pattern has an even number of 1s. therefore, in the random bit error model:

$$p[error\ detection\ failure] = p[undetectable\ error\ pattern]$$

$$= p[error\ patterns\ with\ even\ number\ of\ 1s] \quad (2)$$

$$= \binom{n}{2}p^2(1-p)^{n-2} + \binom{n}{4}p^4(1-p)^{n-4} + \ \ldots$$

where the number of terms in the sum extends up to the maximum possible even number of errors. In the preceding equation we have used the fact that the number of distinct binary $n-tuples$ with j ones and $n-j$ zeros is given by

$$\binom{n}{j} = \frac{n!}{j!\,(n-j!)}$$

In any useful communication system, the probability of a single-bit error p is much smaller than 1. We can then use the following approximation:

$$p^i(1-p)^i \approx p^i(1-pj) \approx p^i.$$

For example, if $p = 10^{-3}$ then $p^2(1-p)^{n-2} \approx 10^{-6}\ and\ p^4(1-p)^{n-4} \approx 10^{-12}$. Thus the probability of detection failure is determined by the first term in equation 4. For example, suppose $n = 32\ and\ p = 10^{-4}$. Then the probability of error detection failure is 5×10^{-6} a reduction of nearly two orders of magnitude.

We have observed that a wide gap exists in the performance achieved by the two preceding channel models. Many communication channels combine aspects of these two channels in that errors occur in bursts. Period of low error rate transmission are interspersed with periods in which clusters of error occur. The periods of low error rate are similar to the random bit error model, and the periods of error burst are similar to the random error vector model, the probability of error-detection failure for the single parity check code will be between those of the two channel models. In general, measurement studies are required to characterize the statistics of the burst occurrence in specific channels.

III. TWO-DIMENSIONAL PARITY CHECKS

A simple method to improve the error-detection capability of a single parity check code is to arrange the information bits in columns of *k* bits, as shown in Figure1.4. The last bit in each column is the check bit for the information bits in the column. [11][13] Note that in effect the last column is a "check codeword" over the previous *m* columns. The right-most bit in each row is the check bit of the other bits in the row. The resulting encoded matrix of bits satisfies the pattern that all rows have even parity and all columns have even parity. If one, two, or three errors occur anywhere in the matrix of bits during transmission, then at least one row or parity check will fail, as shown in Figure 1.5. However, some patterns with four errors are not detectable, as shown in figure. The two-dimensional parity check code is another example of a liner code. It has the property that error-detecting capabilities can be identified visually, but it does not have particularly good performance.

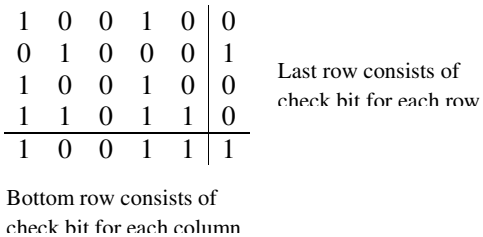


Figure 1.4 Two –dimensional parity check code

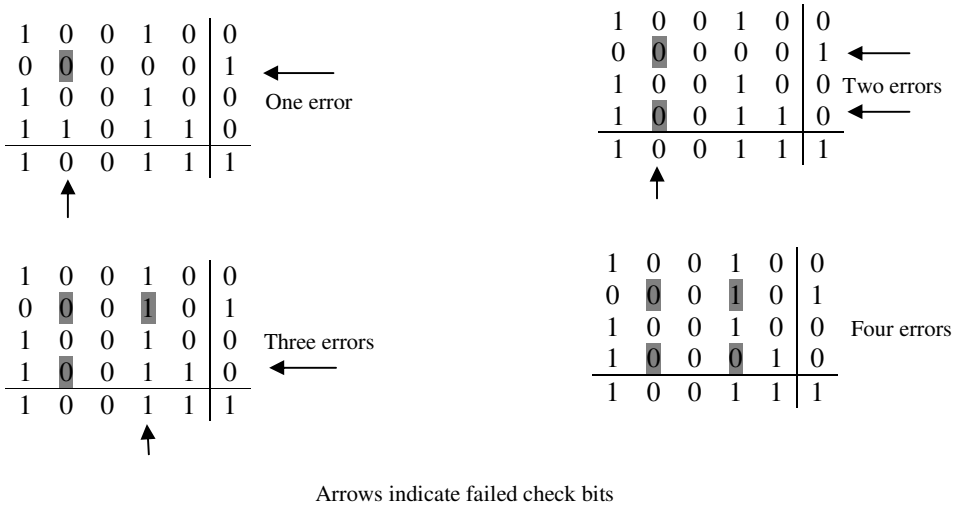


Figure 1.5 Detectable and undetectable error patterns for two-dimensional code

IV. PERFORMANCE OF LINEAR CODES

In figure 1.3 we showed qualitatively that we can minimize the probability of error detection failure by spacing code words apart in the sense that it is unlikely for errors to convert one codeword into another. In this paper we show that the error-detection performance of a code is determine by the distances between code-words. The Hamming distance $d(\underline{b_1}, \underline{b_2})$ between the binary vectors $\underline{b_1}$ and $\underline{b_2}$ is defined as the number of components in which they differ. Thus the Hamming distance between two vectors increases as the number of bits in which they differ increases. Consider the modulo 2 sum of two binary $n-tuples$ $\underline{b_1} + \underline{b_2}$. The components of this sum will equal one when the corresponding components in $\underline{b_1}$ and $\underline{b_2}$ differ, and they will be zero otherwise [14],[15]. Clearly this result is equal to the number of 1s in $\underline{b_1} + \underline{b_2}$, so

$$d\left(\underline{b_1}, \underline{b_2}\right) = w(\underline{b_1} + \underline{b_2}) \qquad (3)$$

where w is the weight function introduced earlier. The extent to which error vectors with few errors are more likely than error vectors with many errors suggests that we should design linear codes that have code words that are far apart in the sense of Hamming distance.

Define the minimum distance d_{mn} of a code as follows

$$d_{mn} = distance\ between\ two\ closest\ distinct\ codewords$$

For any given linear code [16], [17], the pair of closest codewords is the most vulnerable to transmission error, so d_{mn} can be used as a worst-case type of measure. Form equation 9 we have that if $\underline{b_1}$ and $\underline{b_2}$

Is also a codeword. To find d_{mn}, we need to find the pair of distance codewords $\underline{b_1}$ and $\underline{b_2}$ that minimize$d(\underline{b_1}, \underline{b_2})$. By equation 10, this is equivalent to finding the nonzero codeword with the smallest weight. Thus

$$d_{mn} = weight\ fo\ the\ nonzero\ codeword\ with\ the\ smallest\ number\ of\ 1s$$

From Table 1.1, we see the Hamming (7,4) code has $d_{mn} = 3$.

Information				Codeword							weight
b_1	b_2	b_3	b_4	b_1	b_2	b_3	b_4	b_5	b_6	b_7	$w(\underline{b})$
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	1	1	1	1	4
0	0	1	0	0	0	1	0	1	0	1	3
0	0	1	1	0	0	1	1	0	1	0	3
0	1	0	0	0	1	0	0	0	1	1	3
0	1	0	1	0	1	0	1	1	0	0	3
0	1	1	0	0	1	1	0	1	1	0	4
0	1	1	1	0	1	1	1	0	0	1	4
1	0	0	0	1	0	0	0	1	1	0	3
1	0	0	1	1	0	0	1	0	0	1	3
1	0	1	0	1	0	1	0	0	1	1	4
1	0	1	1	1	0	1	1	1	0	0	4
1	1	0	0	1	1	0	0	1	0	1	4
1	1	0	1	1	1	0	1	0	0	1	4
1	1	1	0	1	1	1	0	0	0	0	3
1	1	1	1	1	1	1	1	1	1	1	7

Table 1.1 Hamming (7, 4) code

If we start changing, the bits in a codeword one at a time until another codeword is obtained, then we will need to change at least d_{mn} bits before we obtain another codeword. This situation implies that all error vectors with $d_{min} - 1$ or fewer errors are detectable. We say that a code is $t - error$ detecting if

$$d_{min} \geq t + 1.$$

Finally, let us consider the probability of error-detecting failure for general linear code. In the case of the random error vector channel model, all 2^n possible error patterns are equally probable. A linear (n, k) code fails to datect only the $2^k - 1$ error vectors that correspond to nonzero codewords. We can state then that the probability of error-detection failure for the random error vector channle model is

$\frac{2^k-1}{2^n} \approx 1/2^{n-k}$. [6][12] Furthermore, we can decrease the probability of detection failure by increasing the number of parity bits $n - k$. Consider now the random bit error channel model. The probability of detection failure is given by

$$p[detection\ failure] = p[\underline{e}\ is\ a\ nonzero\ codeword]$$

$$= \sum_{nonzero\ codewords\ \underline{b}} (1-p)^{n-w(\underline{b})} p^{w(\underline{b})}$$

$$= \sum_{w=d_{min}}^{d_{max}} N_w (1-p)^{n-w} p^w$$

$$\approx N_{d_{min}} p^{d_{min}} \; for \; p \ll 1$$

The second summation adds the probability of all nonzero code words. The third summation combines all code words of the same weight. So N_w is the total number of codewords that have weight w. The approximation results from the fact that the summation is dominated by the leading term when p is very small. Consider the $(7,4)$ Hamming code as an example once again. For the random error vector model, the probability of error-detection failure is $\frac{1}{2^3} = \frac{1}{8}$. On the other hand, for the random bit error channel the probabiltity of error-detection failure is approximation $7p^3$, since $d_{min} = 3$ and seven codewords have this weight. If $p = 10^{-4}$, then the probabiltiy of error detection failure is 7×10^{-12}. Compared to the single parity check code, the Hamming code yields a tremendous improvement in error-detection capability.

V. RESULTS AND DISCUSSION

This paper has proposed coding techniques that are applicable in error control for improving the error-rate performance. The effectiveness of an error-detection code observed that a wide gap in terms of errors occur in bursts by the two preceding channel models. Then performance of linear codes are extent to which error vectors with few errors are more likely than error vectors with many errors that should design linear codes that apart in the sense of Hamming distance.

VI. CONCLUSION

Result shows that a wide gap exists in the performance achieved by the two preceding channel models. Many communication channels combine aspects of these two channels in that errors occur in bursts. Period of low error rate transmission are interspersed with periods in which clusters of error occur. The effectiveness of an error-detection code is measured by the probability that the system fails to detect an error. To calculate this probability of error-detection failure, one needs to know the probabilities with which various errors occur. These probabilities depends on the particular properties of the given communication channel.

REFERENCES

[1]. L. Cong, H. L. Qin. Design and simulation of JTIDS/BA/INS/GPS navigation processor software. Journal of Astronautics, 2008, 29(4): 1233–1238. (in Chinese)
[2]. P. Y. Cui, R. C. Zang, H. T. Cui. Fault isolation and recovery based on improved federated filter. Systems Engineering and Electronics, 2007, 29(5): 832–834. (in Chinese)
[3]. S.Seshu, "'On an Improved Diagnosis Problem"', IEEE Transactions on Electronic Computers., vol EC-14, no 1, pp 76-79, Feb 1965.
[4]. M.L.Bushnell and V.D.Agrawal Essentials of Electronic Testing for Digital, Memory and Mixed-Signal VLSI circuits Kluwer Academic Publishers., 2000
[5]. C. Hoare. Communicating Sequential Processes. Prentice Hall, 1985.
[6]. Math Works, 14 April 1999. Personal communication via e-mail from J. Regensburger.
[7]. Leon-Garcia, A., Probability And Random Process For Electrical Engineering, Addision-Wesley, Reading, Massachusetts, 1994.
[8]. Lin, S and D. J. Costello, Error control Coding: Fundamentals and Applications, Prentice Hall, Englewood Cliffs, NJ, 1983.
[9]. M. A. Breuer, S. K. Gupta, and T. M. Mak, "Defect and Error Tolerance in the Presence of Massive Numbers of Defects," IEEE Design and Test Magazine, pp. 216–227, May-June 2004.
[10]. X. Zhang, N. Gupta, and R. Gupta, "Locating Faults Through Automated Predicate Switching," in Proc. of the 28th Int'l Conf. on Software Engineering, pp. 272–281, May 2006.
[11]. X. Li and D. Yeung, "Application-Level Correctness and its Impact on Fault Tolerance," in Proc. of the 13 Int'l Conf. on High-Performance Computer Architecture, pp. 181–192, Feb. 2007.

[12]. Wicker, S. B., Error Control Systems for Digital Communication and Storage, Prentice-Hall, Inc., 1995.
[13]. Peterson, L. L. and B. S. Davie, Computer Networks – A Systems Approach, 2nd Ed, Morgan Kaufmann, 2000.
[14]. Siewiorek, D. P. and R. S. Swarz, Reliable Computer Systems – Design and Evaluation, 3rd Ed, Digital Press, 2000.
[15]. Wilken, K., and J.P. Shen, "Concurrent Error Detection Using Signature Monitoring and Encryption: Low-Cost Concurrent-Detection of Processor Control Errors," Dependable Computing for Critical Applications, Springer-Verlag, A. Avizienis, J.C. Laprie (eds), Vol. 4, pp. 365-384, 1989.
[16]. Rakesh kumar katare & Shiv Kumar Gupta "Realization Through ABFT in Cloud Computing" on International Conference on Challenges of. Globalization & Strategy for Competitiveness. (ICCGC-2011). 14-15 January, 2011
[17]. Varavithya, V., Upadhyay, J., Mohapatra, P.: An Efficient Fault-Tolerant Routing Scheme for Two-Dimensional Meshes. In: International Conference on High Performance Computing, pp. 773–778 (1995)
[18]. Duato, J., Yalamanchili, S., Ni, V.: Interconnection Networks, an Engineering Approach. Morgan Kaufmann Publishers, USA (2003)

Authors

Shiv Kumar Gupta has done M. Sc (Mathematics), M.C.A., M. Phil (CS) and currently pursing Ph.D. Computer Science from Manav Bharti University, Solan (H.P.) India. He is life membership of "MATERIALS RESEARCH SOCIETY OF INDIA".

Rajiv Kumar received his B. Tech. in 1994, in Electrical Engineering, from College of Technology, G.B. Pant University of Agriculture & Technology, Pantnagar and M. Tech. from Regional Engineering College, Kurukshetra (Kurukshetra University) with specialization in Control System. He started his career as a teaching associate from National Institute of Technology (NITT), Kurukshetra. He is presently with the Department of Electronics and Communication Engineering at Jaypee University of Information Technology, Waknaghat, Solan. He obtained his Ph.D.in network reliability in the year 2010 from NIT under the supervision of Prof. Krishna Gopal. Rajiv Kumar is member of IEEE and Life Member of ISTE, IETE, Forum of Inter disciplinary Mathematics and System Society of India. His areas of research interests are computing, cyber- physical systems and network reliability.