A New Factorization Method to Factorize RSA Public Key Encryption

B R Ambedkar[1] and S S Bedi[2]

[1] Department of CS and IT, MJP Rohilkhand University

Bareilly, Uttar Pradesh 243006, India

[2] Department of CS and IT, MJP Rohilkhand University

Bareilly, Uttar Pradesh 243006, India

Abstract

The security of public key encryption such as RSA scheme relied on the integer factoring problem. The security of RSA algorithm is based on positive integer *N*, because each transmitting node generates pair of keys such as public and private. Encryption and decryption of any message depends on *N*. Where, *N* is the product of two prime numbers and pair of key generation is dependent on these prime numbers. The factorization of *N* is very intricate. In this paper a New Factorization method is proposed to obtain the factor of positive integer *N*. The proposed work focuses on factorization of all trivial and nontrivial integer numbers and requires fewer steps for factorization process of RSA modulus *N*. The New Factorization method is based on Pollard rho factorization method. Experimental results shown that factorization speed is fast as compare existing methods.

Keywords: *Attacks, Pollard rho method, Public Key Cryptography, RSA Algorithm***.**

1. Introduction

Public key cryptography is one of the mathematical applications that are valuable in sending information via insecure channel. RSA algorithm is a public key encryption algorithm. RSA has become most popular cryptosystem in the world because of its simplicity. According to number theory, it is easy to finds two big prime number, but the factorization of the product of two big prime numbers is very difficult. The difficulty of computing the roots *N*, where *N* is the product of two large unknown primes, it is widely believed to be secure for large enough *N*. Since RSA can also be broken by factoring *N*, the security of RSA is often based on the integer factorization problem [1]. The integer factorization problem is a well-known topic of research within both academia and industry. It consists of finding the prime factors for any given large modulus. Currently, the best factoring algorithm is the general number field sieve or GNFS for short. On December 12, 2009 a small group of scientists used the previously mentioned approach to factor a RSA-768 bit modulus, that is, a composite number with 232 decimal digits. Their achievement required more than two years of collaborative work and used many hundreds of computing machines. Hence, factoring large primes is a laborious and complex task [2]. A method for factoring algorithm (specially designed) for semi primes based on new mathematical ideas. Since this method is relatively simple and scalable, it can be suitable for parallel processing [2]. A new algorithm which attacks the RSA scheme. But the main condition of this algorithm is that to break RSA modulus firstly we should have public key and modulus *N*. On the basis of this public key *(e, N)* proposed algorithm disclose the private key [3]. An attacker has access to the public key *e* and *N* and the attacker wants the private key *d*. To get *d, N* needs to be factored (which will yield *p* and *q*, which can then be used to calculate *d*). Factoring n is the best known attack against RSA to date. (Attacking RSA by trying to deduce *(p-1) (q-1*) is no easier than factoring *N*, and executing an exhaustive search for values of d is harder than factoring *N*.) Some of the algorithms used for factoring are as follows [12]: Trial division oldest and least efficient Exponential running time. Try all the prime numbers less than sqrt *(N)*. Quadratic Sieve (QS)**:** The fastest algorithm for numbers smaller than 110 digits. Multiple Polynomial Quadratic Sieve (MPQS): Faster version of QS. Double Large Prime Variation of the MPQS Faster still. Number Field Sieve (NFS) Currently the fastest algorithm known for numbers larger than 110 digits. Was used to factor the ninth Fermat number. These algorithms [12] represent the state of the art in warfare against large composite numbers against RSA. We can identify many approaches to attacking RSA mathematically, factor N into two prime factors. A lot of algorithm has been proposed regarding factorization, the Pollard rho algorithm [7], and the Pollard (*p*-1) algorithm

[8], Brent's method [9], are probabilistic, and may not finish, for small values of *N*, Trial division algorithm [12] and Fermat method can finish [11]. In this paper we are proposing New Factorization (NF) method which is based on Pollard rho Factorization (PRF) method [7]. By using NF method we can factorize quickly all integer number. It is very suitable for factorization against RSA algorithms, because that have a product of two prime number, so there is no more than two factors, NF method factorize all numbers with minimum elapsed time shown in Fig. 1. MATLAB environment is used for various analyses.

2. Attacks

The security of the RSA cryptosystem is not perfect. An attacker can resort to different approaches to attack RSA algorithm. Among them, Brute force, Mathematical attacks, Timing attacks and Chosen Cipher text attacks are described in following:

2.1 Brute Force Attacks on RSA

Brute force attacks, involves trying all possible public and private keys. The attacker tries all possible combinations to guess the private key. RSA with short secret key is proven insecure against brute force attack [13]. This attack can be easily circumvented by choosing large key. However, the larger key will make the encryption and decryption process little slow as it will require greater computations in key generation as well as in encryption/decryption algorithm. The key length used in the encryption determines the practical feasibility of performing a brute-force attack, with longer keys exponentially more difficult to crack than shorter ones. Brute-force attacks can be made less effective by obfuscating the data to be encoded, something that makes it more difficult for an attacker to recognize when he/she has cracked the code. One of the measures of the strength of an encryption system is how long it would theoretically take an attacker to mount a successful brute-force attack against it.

2.2 Mathematical Attacks on RSA

Mathematical attacks focus on attacking the underlying structure of RSA function. The first intuitive attack is the attempt to factor the modulus *N*. There are several approaches, all equivalent to in effect to factoring the product of two primes *N*. Our objective is to provide RSA attacks that decrypts message by factoring *N*.

2.3 Timing Attack

In RSA, the attacker can exploit the timing variation of the modular exponentiation implementations and able to determine d by calculating the time it takes to compute $C^d \pmod N$ for a given cipher text *C* [13].

2.4 Factorization Attack

The security of RAS is based on the idea that the modulus is so large that is infeasible to factor it in reasonable time. Bob selects *P* and *Q* and calculate $N=P\times Q$. Although *N is* public, *P* and *Q* are secret. If Eve can factor *N* and obtain *P* and *Q*, Eve then can calculate $d = e^{-1} \bmod \varphi(N)$ because *e* is public. The private exponent *d* is the trapdoor that Eve uses to decrypt any encrypted message.
The factorization attack is a extremely giant dispute for security of RSA algorithm. Some existing factorization algorithms can be generating public and private key of RSA algorithm, by factorization of modulus *N*. But they are taking huge time for factorization of *N*, in case of *P* and *Q* very large. We are focusing on factorization speed and proposing new factorization method to enhance the speed of factorization. Related works for factorization of modulus *N* are following.

3. Related Work

The RSA Factoring Challenge was started in March 1991 by RSA Data Security to keep abreast of the state of the art in factoring. Since its inception, well over a thousand numbers have been factored, with the factories returning valuable information on the methods they used to complete the factorizations. The Factoring Challenge provides one of the largest test-beds for factoring implementations and provides one of the largest collections of factoring results from many different experts worldwide. In short, this vast pool of information gives us an excellent opportunity to compare the effectiveness of different factoring techniques as they are implemented and used in practice. Since the security of the RSA public-key cryptosystem relies on the inability to factor large numbers of a special type, the cryptographic significance of these results is self-evident. Some factorization methods are explored in following paragraphs:

J. Carlos et al [2] proposed a method for factoring algorithm is: gcd $[N, (k \,.\, \check{N}) + \Delta]$ and gcd $[N, (k \,.\, \check{N}) - \Delta]$ result in nontrivial factors of *N* for different values of Δ where Ň is the reverse of *N* and k is a positive integer ranging from one to infinity.

Sattar J Aboud [3] was introducing a method that breaking the RSA scheme based on the knowing public key *(e, N)*. This method will work efficiently if the decryption key d is small. It possible therefore, to factor the modulus *N*.

L. Scripcariu, M.D. Frunze [4] was commencing some weak points of RSA algorithm and proposed a method for secure public key. The main concept of this paper is for encryption *(e, N)* change every time public key for encryption.

In 1978, RSA developed a public key cryptosystem that is based on the difficulty of integer factoring. The RSA public key encryption scheme is the first example of a provable secure public key encryption scheme against chosen message chosen attacks [5].The RSA scheme is as follows [6]:

Key generation algorithm, to generate the keys entity A must do the following:

1. Randomly and secretly select two large prime numbers *p* and *q*.
2. Compute the $N=p*q$.
3. Compute (φ) phi $(N) = (p-1)(q-1)$.
4. Select random integer e, $1<e<N$, where gcd $(e, \varphi) =1$.
5. Compute the secret exponent d, $1 < d <$ phi, such that $e*d \equiv 1 (\text{mod } (\text{phi}))$.
6. The public key is (N, e) and the private key is (N, d). Keep all the values d, p, q and phi secret.

Public key encryption algorithm, entity *A* encrypt a message m for entity B which entity decrypt.

Encryption: Entity *B* should do the following:

Obtain entity *A*'s public key *(N,e)*, represent the message M as an integer in the interval *[0….N-1]*. Compute $C= M^e \text{ mod } N$.Send the encrypted message e to entity *A*.

Decryption: To recover the message m from the cipher text c. Entity A do the following: Obtain the cipher text c from entity B. Recover the message $M = C^d \text{ mod } N$.

Fermat Factorization [11] was discovered by mathematician Pierre de Fermat in the 1600s. Fermat factorization rewrites a composite number *N* as the difference of squares as follows: $N=x^2 – y^2$, this difference of squares leads immediately to the factorization of *N: N=(x+y)(x-y)*, Assume that *s* and *t* are nontrivial odd factors of *N* such that $st = N$ and $s \leq t$. We can find *x* and *y* such that $s = (x - y)$ and $t = (x + y)$

Example 1:

Let N=95{Product of any two prime numbers.}
Decimal digits: 2 Bits: 7
Let factors := P,Q
Compute X :=10
Compute Y :=2.236 (is not integer number)
(Go to step 4)
X :=11
Y: =5.09 (is not integer number)
(Go to step 4)
X :=12
Y: =7 (is an integer number)
P: = 5
Q: =19
End

Example 2:

Let N= 2320869986411928544793
Decimal digits: 22 Bits: 73
Let factors := P,Q
Compute X := 48175408524
Compute Y :=219488.97 (is not integer number)
(Go to step 4)
X := X + 1,……,X+17073029192103
X: = 17121204600627
Y: = 17121136822844 (is an integer number)
P: = 67777783
Q: =34242341423471
End

4. Pollard Rho Factorization

Pollard rho Factorization [7] method is a probabilistic method for factoring a composite number *N* by iterating a polynomial modulo *N*. The method was published by J.M. Pollard in 1975. Suppose we construct the sequence: $X_0 \equiv 2 \text{ mod } N$, $X_{n+1} \equiv X^2_n + 1 (\text{mod } N)$. This sequence will eventually become periodic. It can be shown that the length of the cycle is less than or equal to *N* by a proof by contradiction: assume that the length *L* of the cycle is greater than *N*, however we have only *N* distinct *xn* values in our cycle of length $L > N$, so there must exist two *xn* values are congruent, and these can be identified as the .starting points. of a cycle with length less than or equal to *N*. Probabilistic arguments show that the expected time for this sequence (mod *N*) to fall into a cycle and expected length of the cycle are both proportional to N , for almost all *N* [8]. Other initial values and iterative functions often have similar behavior under iteration, but the function $f(N) = (X^2_n + 1)$ has been found to work well in practice for factorization. Assume that *s* and *t* are nontrivial factors of *N* such that $st = N$ and $s \leq t$. Now suppose that we have found nonnegative integers *i, j* with $i < j$ such that $X_i \equiv X_j \pmod{s}$ but $X_i \neq X_j \pmod{N}$. Since $s \mid (X_i – X_j)$, and $s \mid N$, we have that $s \mid \gcd(X_i – X_j, N)$. By assumption $s \geq 2$, thus $\gcd(X_i – X_j, N) \geq 2$. By definition we know that $\gcd(X_i – X_j, N) \mid N$. However, we have that $N \mid (X_i – X_j)$, and thus that $\gcd(X_i – X_j, N) \mid N$. So we have that $N \mid \gcd(X_i – X_j,$

$N)$, gcd $(X_i - X_j, N) > 1$, and gcd $(X_i - X_j, N) | N$. Therefore gcd $(X_i - X_j, N)$ is a nontrivial factor of N.

Example 3:

Consider the Pollard rho algorithm for $N = 21 = 7*3$.
The sequence of Xn values generated by the algorithm is: $X_0 = 2$, $X_1 = 5$, $X_2 = 5$,……, for n≥1,
If $n \geq 1$, $X_{2n} - X_n = 0$.
The algorithm at each step for n = 1,2,…
Compute gcd $(X_{2n} - X_n, N)$ = gcd $(0, 21) = 21$
We are showing this example, the Pollard rho algorithm is a probabilistic, and may not finish, even for small values of N.

5. New Factorization Method

The new factorization method (NF) based on Pollard rho method and square root of N. The NF method is focusing on Mathematical and Factorization Attacks on RSA. Shown in e.g. 3, the Pollard rho factorization method may not finish small value of N. We are using RSA algorithm for public key cryptography, it is asymmetric encryption technique, by using this technique we cannot encrypt and decrypt message by same key. The generation of public and private key is dependent on the factorization of N. By using Pollard rho factorization method, we can't factorize all integer numbers N see e.g. 3. By using NF method we factorize all integer numbers N. The NF method is a independent of periodic sequence and probabilistic method. For factorization of all integer number, we are proposing to Pollard rho factorization method to modifying and removing the concept of periodic sequence. NF method as follows:

1. Let any positive integer is N.
2. Compute the square root of N.
3. Take ceiling function of step two.
4. Decrement by one in square of step three.
5. Compute the greatest Common divisor of step three and step number one.
6. If step five is grater than one.
7. Step five is a factor of step one.
8. Compute other factor of N divided by step seven.
9. Otherwise increment the value of steps three by one and continues step three to eight, till step four is grater than one.

By using these steps we factorize any positive integer as follows:

Example 4:

Let N=21
S=5
P=gcd(24,21)=3>1
Q=21/3=7
P and Q is factor of N.

Example5:

Let N= 999962000357
Decimal digits=12 Bits= 40
S= 999981
P=gcd (999962000360, 999962000357) = 1
P=gcd (999964000323, 999962000357) = 999983>1
Q= 999962000357/999983=999979
P and Q is a factor of N.

6. Simulation and Results

We are factorized all trivial and nontrivial number shown in e.g. 4. The NF method is a appropriate and essential for factorization of product of any two prime numbers because that have only two factors as shown in simulated result Fig. 1, factors of 14 decimal digit by using NF method elapsed time is 94ms shown in Fig. 1. That domino effect simulated by MATLAB version 7.0 and Intel(R), Core(TM)2 Quad CPU 2.66GHz, 3.24 GB of RAM. As per the results shows factorization becomes more rapidly by NF method.

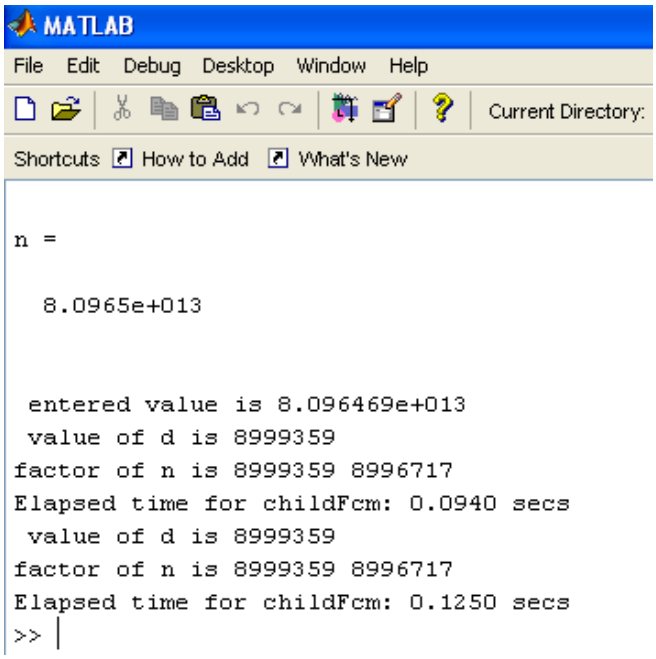

Fig. 1 Factors of 80964686104403 by using NF method (Elapsed time in sec.= 0.094) simulation shows.

The elapsed time for prime factorization shown in Fig. 2 with respect digits and Fig. 3 with respect bits process elapsed times is awfully tiny as compare to existing method.

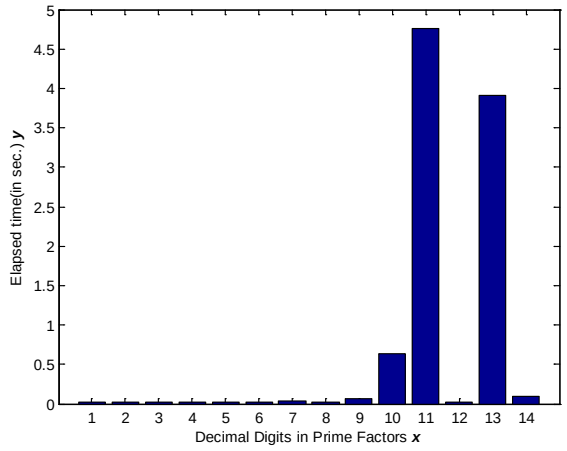

Fig. 2 NF method(Elapsed time vs Decimal digits in prime factors)

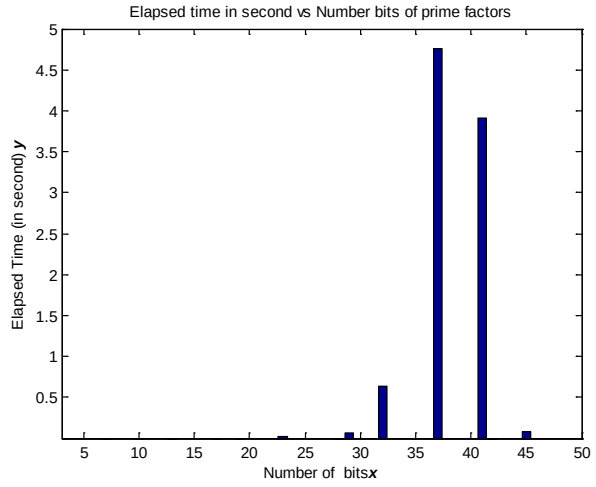

Fig. 3 NF method(Elapsed time vs Total number of Bits of prime factors)

Shown in Fig. 4, a plot of time elapsed and number of digits for given number by using traditional existing Pollard rho, the factorization of 7 digit decimal number elapsed time is 0.5s.

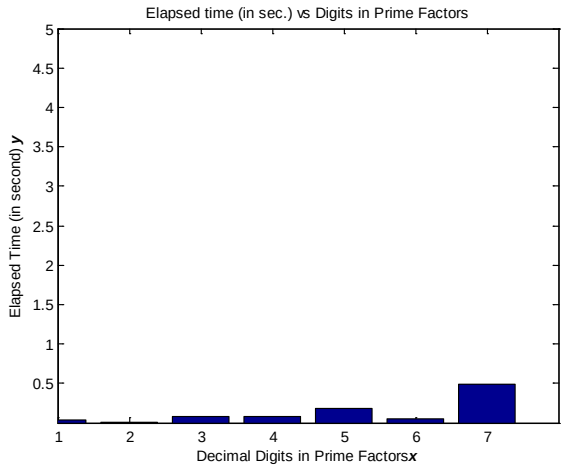

Fig. 4 Pollard rho method (Elapsed time vs Decimal digits in prime factors)

Table 1 shows a comparison of some factorization methods vs NF method, The trial division, Fermat factorization method and NF method always terminate, and upper bounds can be derived for the running times of these algorithms in terms of *N*, the number to be factored. The Pollard rho algorithm, Brent's method, and the Pollard *p*-1algorithm are probabilistic, and may not finish, even for small values of *N*.

Table 1: Comparison of few factorization methods vs MPRF method

Factorization Method	For all numbers	Technique used
Trial	Can factorize	Division based
Fermat	Can factorize	$N=x^2-y^2$
Pollard Rho	Can't factorize	Periodic sequences
Pollard (p-1)	Can't factorize	$(a^{k!}-1) \bmod n$
Brent's	Can't factorize	$X_{n+1}=X_n^2+2(\text{mod}$
NF	Factorize	Square root based

The shown in Fig. 5 comparison between existing algorithms and NF method. The factorization of 7 digit decimal number by NF method the elapsed time is 0.032s. NF method produced all factors of any integer number. The algorithm was executed using MATLAB tool.

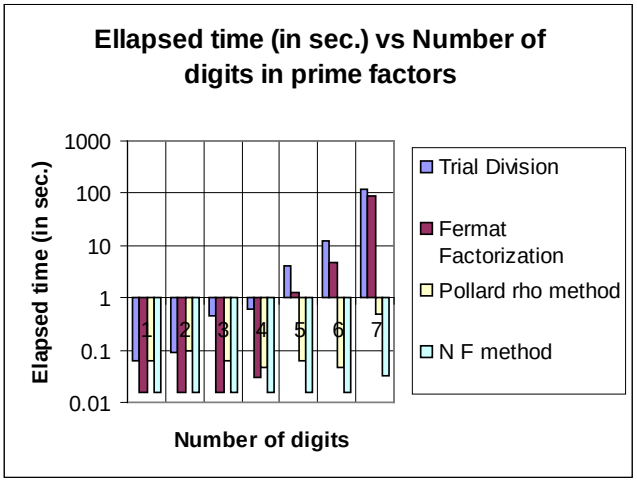

Fig. 5 Number of digits vs. Elapsed time in Prime Factors

7. Conclusions

In this paper New Factorization method is proposed for RSA modulus *N* factorization. During simulation, Pollard rho factorization method is found not suitable to factor all trivial numbers as shown in example 3. Although NF method factorizes all the integer numbers as described in example 4. The elapsed time with respect to decimal digits in prime factors is found less as compared to considered existing methods. The speed to compute the prime factors is also found more as compare to existing methods. Therefore the proposed method is relatively simple, fast and scalable as compare to existing methods.

References

[1] Hung-Min Sun, Mu-En Wu, Wei-Chi Ting, and M. Jason Hinek, "Dual RSA and Its Security Analysis", IEEE Transactions on Information Theory, Vol. 53, No. 8, Aug. 2007.

[2] Joao Carlos Leandro da Silva, "Factoring Semi primes and Possible Implications", IEEE in Israel, 26th Convention, pp. 182-183, Nov. 2010 .

[3] Sattar J Aboud, "An efficient method for attack RSA scheme", ICADIWT Second International Conference, pp 587-591,4-6 Aug 2009.

[4] L. Scripcariu, M.D. Frunza, "A New Character Encryption Algorithm", ICMCS 2005, pp. 83 - 86, Sept., 2005.

[5] Hongwei Si *et al*, "An Improved RSA Signature Algorithm based on Complex Numeric Operation Function", International Conference on Challenges in Environmental Science and Computer Engineering , vol. 2, pp. 397-400, 2010.

[6] B. Schneier, Applied cryptography, second edition, NY: John Wiley & Sons, Inc., 1996.

[7] J. Pollard, "Monte Carlo methods for index computation (mod p)",Math. Comp., Vol. 32, pp.918-924, 1978.

[8] J. Pollard, "Theorems on factorization and primality testing", Proc. Cambridge Philos.Soc., Vol. 76, pp.521-528, 1974.

[9] R. P. Brent, "An improved Monte Carlo factorization algorithm", BIT 20 (1980), 176-184. MR 82a:10007, Zbl 439.65001. rpb051 .

[10] R. Rivest, A. Shamir, L. Adleman, "A Method for Obtaining Digital for Signatures and Public-Key Cryptosystems", Communications of the ACM, vol. 21 (2), pp. 120-126, 1978 .

[11] Bell, E. T., "The Prince of Amateurs: Fermat." New York: Simon and Schuster, pp. 56-72, 1986.

[12] Ms. Divya Bansal, Ajay Goel, "Interrelation Among RSA Security, Strong Primes, and Factoring"

[13] M. Wiener, "Cryptanalysis of short rsa secret exponents",IEEE Transactions on Information Theory, 160:553–558, March 1990.

Bhagwant Ram Ambedkar received the B. Tech. Degree in Electronics Engineering from Institute of Engineering and Technology Lucknow, India in 2001 and M. Tech. with specialization in Wireless Communication and Computing from Indian Institute of Information Technology, Allahabad, India in 2004. Presently he is working **as** Assistant Professor in Department of Computer Science and Information Technology, MJP Rohilkhand University Bareilly, Uttar Pradesh, India.

Dr. Sarabjeet Singh Bedi received the M.E. degree in Computer Science and Engineering from Thapar Institute of Engineering and Technology, Punjab, India in 2002. He has received his Ph.D. from Indian Institute of Information Technology and Management, Gwalior, India. He has teaching and research experience of 16 years. His research interest is Network Management and Security. Currently He is teaching at M.J.P. Rohilkhnad University, Bareilly, India. He is involved in various academic and research activities. He is member of selection, advisor, governing and technical committees of various Universities and Technical Institution bodies. Dr. S. S. Bedi received Institute Medal from TIET, Punjab, India, 2002, Rastriya Shikshak Ratan Award from AIBDA, New Delhi in 2002 and Best Paper award at World Congress on Engineering at London, 2008.