Comparative Study of Arithmetic and Huffman Data Compression Techniques for Koblitz Curve Cryptography

O. Srinivasa Rao
Dept. of CSE
University College of Engineering
JNTUK-Vizianagaram
Andhra Pradesh, India-535 003

Dr S.Pallam Setty
Professor of CS&SE
University College of Engg.,
Andhra University, Visakhapatnam
Andhra Pradesh, India-530 003

ABSTRACT
Over the past 20 years, numerous papers have been written on various aspects of ECC implementation. In this paper we investigate the superiority of the Arithmetic data compression technique over the Huffman data compression technique in reducing the channel bandwidth and the transmission time. The main purpose of data compression is to reduce the memory space or transmission time, while that of cryptography is to ensure the security of the data. Applying Data compression techniques not only reduces the bandwidth but also enhances the strength of the cryptosystem. It is also observed that even if the given string is doubled i.e. AAAA (4A's) to AAAAAAAA (8A's), the compression ratio remains constant. Further in Arithmetic Data Compression the compression ratio is 50% more when compared to the Huffman Data Compression and the ratio increases with increasing string length.

General Terms
Your general terms must be any term which can be used for general classification of the submitted material such as Pattern Recognition, Security, Algorithms et. al.

Keywords
Elliptic curve cryptography, Koblitz curves, Huffman Data compression, Arithmetic Data Compression

1. INTRODUCTION
In 1985, Neal Koblitz [1] and Victor Miller [5] independently proposed using the group of points on an elliptic curve defined over a finite field in discrete logarithm cryptographic systems. In 1991, Koblitz[1] suggested using a special family of elliptic curves now popularly referred to as Koblitz curves. Koblitz curves have been widely studied in the academia and have been included in certain standards [2-4]. In the Koblitz curves point multiplication is considerably more efficient than on general curves.

The primary advantage that elliptic curve systems have over systems based on the multiplicative group of a finite field (and also over systems based on the intractability of integer factorization) is the absence of a sub exponential-time algorithm (such as those of "index-calculus") to find discrete logarithms in these groups. Consequently, one can use an elliptic curve group that is smaller in size and still maintain the same level of security. The result is smaller key sizes, bandwidth savings, and faster implementations—features that are especially attractive for security applications in smart cards, personal digital assistance, and wireless devices where computational power and integrated circuit space are limited. Elliptic curve cryptographic protocols for digital signatures, public-key encryption, and key establishment have been standardized by numerous standards organizations including:

- American National Standards Institute (ANSI X9.62 [6], ANSI X9.63 [7]).
- Institute of Electrical and Electronics Engineers (IEEE 1363-2000 [8]).
- International Standards Organization (ISO/IEC 15946-3 [9]).
- U.S. government's National Institute for Standards and Technology (FIPS 186-2 [2]).
- Internet Engineering Task Force (IETF PKIX [7], IETF OAKLEY [10]).
- Standards for Efficient Cryptography Group (SECG [3]).

Majority of the products and standards use RSA public-key cryptography for encryption and digital signatures. As seen, the bit length for secure RSA has increased over the years, thus putting a heavier processing load on applications. This burden has ramifications, especially for electronic commerce sites that conduct large numbers of secure transactions. Recently elliptic curve based cryptosystems have emerged as a competing alternative to RSA. [8,11,17]

2. ELLIPTIC CURVE CRYPTOGRAPHY
Elliptic curve cryptography makes use of elliptic curves in which the variables and coefficients are all restricted to elements of a finite field. Two families of elliptic curves are used in cryptographic applications: Prime curves defined over Zp and binary curves constructed over GF (2m). Fernandes[12] points out that prime curves are best suited for software applications, as the extended bit –fiddling operations needed by binary curves are not required; ,and that binary curves are best for hardware applications, where it takes remarkably few logic gates to create a powerful and fast cryptosystem. In this paper we used Koblitz curves which are a variant of binary curves for analysis purpose.

2.1 Koblitz curves
A koblitz curve E over F2m is an elliptic curve whose defining equation has coefficients in F2. There are two koblitz curves: y2+xy=x3+1 and y2+xy=x3+x2+1.These elliptic curves were first proposed for cryptographic use by koblitz [1].They have advantage over randomly selected curves over binary fields because the point multiplication operation in Koblitz curves involves no point doubling (See Solinas [14, 15, 19]).Koblitz curves have been standardized in NIST's FIPS (186-2[2]).

2.2 Elliptic Curves Arithmetic over F_2^m
A (non-super singular) elliptic curve E(F_2^m,) over F_2^m, defined by the parameters a, b ϵ F_2^m, b $\neq$ 0 , is the set of all solutions (x, y), x, y ϵ F_2^m, to the equation $y^2+xy=x^3+ax^2+b$, together with an extra point O, which is the point at infinity.

The set of points $E(F_2^m)$ forms a group with the following additional rules:

1. O+O =O

2. (x, y)+O=O+(x, y)=(x, y) for all (x, y) ϵ $E(F_2^m)$.

3. (x, y)+(x, x + y)=O for all (x, y) ϵ $E(F_2^m)$

(i.e., the negative of the point (x, y) is –(x, y)=(x, x +y)).

4. (Rules for adding two distinct points that are not inverse of each other) Let $P = (x_1, y_1) \epsilon E(F_2^m)$ and $Q=(x_2,y_2) \epsilon E(F_2^m)$ be two points such that $x_1 \neq x_2$.Then

$P+Q=(x_3,y_3)$,where

$x_3=\lambda^2+ \lambda+x_1+x_2+a$,

$y_3= \lambda(x_1+x_3) +x_3+y_1$, and

$\lambda = (y_2+y_1)/(x_2+x_1)$

5. (Rule for doubling a point)

Let $P=(x_1, y_1) \epsilon E (F_2^m)$ be a point with $x_1 \neq 0$. (If $x_1=0$ then P=-P, and so 2P=O) Then $2P =(x_3, y_3)$, where

$x_3= \lambda^2+ \lambda + a$

$y_3=x_1^2+ (\lambda+1) x_3$, and

$\lambda =x_1+ (y_1/x_1)$

3. DATA COMPRESSION TECHNIQUES

A data compression method is called *universal* if the compressor and decompressor do not know the statistics of the input stream. A universal method is *optimal* if the compressor can produce compression factors that asymptotically approach the entropy of the input stream for long inputs. *Compression performance*: Several quantities are commonly used to express the performance of a compression method. The Compression ratio is one of the quantities used to express compression efficiency and is defined as

Compression ratio =Size of the output stream/size of the input stream.

A value of 0.6 means that after compression the data occupies 60% of its original size. Values greater than 1 imply that the output stream is bigger than the input stream (negative compression). The compression ratio can also be called bpb (bit per bit), since it equals the number of bits in the compressed stream needed, on an average, to compress one bit in the input stream. In image compression, the same term, bpb stands for "bits per pixel."In modern, efficient text compression methods, it makes sense to talk about bpc (bits per character)—the number of bits it takes, on average, to compress one character in the input stream.

3.1 Huffman coding

A commonly used method for data compression is Huffman coding. It serves as the basis for several popular programs used in personal computers. Some of them use just the Huffman method, while others use it as one step in a multistep compression process. The Huffman method [15] is somewhat similar to the Shannon-Fano method. It generally produces better codes, and like the Shannon-Fano method, produces best code when the probabilities of the symbols are negative powers of 2. The main difference between the two methods is that Shannon-Fano constructs its codes from top to bottom (from the leftmost to the rightmost bits), while Huffman constructs a code tree from the bottom (builds the codes from right to left). Since its development, in1952, by D. Huffman, this method has been the subject of intense research in data compression.

The method starts by building a list of all the alphabet symbols in descending order of probabilities. It then constructs a tree, with a symbol at every leaf, from the bottom up in steps, where at each step the two symbols with the smallest probabilities are selected, added to the top of the partial tree, deleted from the list, and replaced with an auxiliary symbol representing the two symbols. When the list is reduced to just one auxiliary symbol (representing the entire alphabet), the tree is complete. The tree is then traversed to determine the codes for the symbols.

3.2 Arithmetic Coding

The Huffman method is simple, efficient, and produces the best codes for the individual data symbols. However, it is shown that the only case where it produces ideal variable-size codes (codes whose average size equals the entropy) is when the symbols have probabilities of occurrence that are negative powers of 2 (i.e., numbers such as1/2, 1/4, or 1/8). This is because the Huffman method assigns a code with an integral number of bits to each symbol of the alphabet. Information theory shows that a symbol with probability 0.4 should ideally be assigned a 1.32-bit code, since $-\log_2 0.4 \approx 1.32$.The Huffman method, however, normally assigns such a symbol a code of 1 or 2 bits. Arithmetic coding overcomes the problem of assigning integer codes to the individual symbols by assigning one (normally long) code to the entire input file. The method starts with a certain interval, it reads the input file symbol by symbol, and uses the probability of each symbol to narrow down the interval. Specifying a narrower interval requires more bits, so the number constructed by the algorithm grows continuously. To achieve compression, the algorithm is designed such that a high-probability symbol narrows the interval less than a low-probability symbol, with the result that high-probability symbols contribute fewer bits to the output .An interval can be specified by its lower and upper limits or by one limit and width (range). We use the latter method to illustrate how an interval's specification becomes longer as the interval narrows. The interval [0, 1] can be specified by the two 1-bit numbers 0 and1. The interval [0.1, 0.512] can be specified by the longer numbers 0.1 and 0.412. The very narrow interval [0.12575, 0.1257586] is specified by the long numbers 0.12575 and 0.0000086.

4. ECC ENCRYPTION AND DECRYPTION

Several approaches to encryption/ decryption using elliptic curves have been analyzed. This paper describes one of them. The first task in this system is to encode the plaintext message m to be sent as an x-y point Pm. It is the point Pm that will be encrypted to cipher text and subsequently decrypted. Note that we cannot simply encode the message as the x or y coordinate of a point, because not all such coordinates are in Ep(a, b). There are techniques for encoding. We developed a scheme that will be reported elsewhere. As with the key exchange system, an encryption/decryption system requires a point G and an elliptic group Ep(a, b) as parameters. Each user selects a private key n_A and generates a public key $P_A = n_A \times G$.

To encrypt and send a message Pm to B, A chooses a random positive integer x and produces the cipher text Cm corresponding to the pair of points ([7],[18],[20])

$$C_m= \{xG, P_m + xP_B\} \quad (1)$$

Note that A has used B's public key PB. To decrypt the cipher text, B multiplies the first point in the pair by B's secret key and subtracts the result from the second point:

$P_m + xP_B - n_B(xG) = P_m + x(n_BG) - n_B(xG) = P_m(2)$

A has masked the message P_m by adding xP_B to it. Nobody but A knows the value of x, so even though P_B is a public key, nobody can remove the mask xP_B. However, A also includes a "clue," which is enough to remove the mask if one knows the private key n_B. For an attacker to recover the message, he would have to compute x, given G and xG, which is hard.

5. Analytical study of Bandwidth for Koblitz Curve using Huffman Data Compression and Arithmetic Data compression

We have taken an irreducible polynomial x^7+x+1, and the Koblitz curve of $y^2+xy=x^3+x^2+1$ in the binary field. Then we generated the (x, y) points for the chosen koblitz curve. These points were mapped to the alphanumeric characters [16]. The mapped points are encrypted using equation (1). The co-ordinates of the koblitz curve, encryption and decryption of the input strings, Data compression algorithms are implemented in C and the results are shown in table1 and table2 and plot of these tables are shown Figure 1 and Figure 2. From these tables and graphs it is revealed that if the input string contains alphabets which are repeating, the compression ratio is high. The compressed data to be sent to the destination is much less in size and thus requires less bandwidth. The compression ratio in Arithmetic Data Compression is 50% more when compared to the Huffman Data Compression .Further the Compression increases with increase in string length. At the destination the data is uncompressed and original text is recovered by using the equation (2).

6. CONCLUSION

We compare the performance of arithmetic and Huffman Compression techniques for different input strings. It is observed that Arithmetic compression technique is more appropriate than Huffman data compression technique in reducing the channel bandwidth and the transmission time in Elliptic curve based cryptosystems.

Table 1

S.No	Input Data String	Encrypted data Size(bits)	Huffman Compressed Data Size (bits)	Arithmetic Compressed Data Size (bits)	Huffman Compression Ratio (Percentage)	Arithmetic Compression Ratio (Percentage)
1	AAAAA	70	10	7	14.285714	10
2	AAAAE	70	20	9	28.571429	12.857142
3	AAADA	70	20	9	28.571429	12.857142
4	AAADE	70	30	15	42.857143	21.428571
5	AACAA	70	20	9	28.571429	12.857142
6	AACAE	70	30	15	42.857143	21.428571
7	AACDA	70	30	15	42.857143	21.428571
8	AACDE	70	30	15	42.857143	21.428571
9	ABAAA	70	20	9	28.571429	12.857142
10	ABAAD	70	30	15	42.857143	21.428571
11	ABADA	70	30	15	42.857143	21.428571
12	ABADE	70	30	24	42.857143	34.285714
13	ABCAA	70	30	15	42.857143	21.428571
14	ABCDA	70	30	24	42.857143	34.285714
15	ABCDE	70	30	25	42.857143	35.714285
16	JNTU	56	24	20	42.857143	35.714285
17	JNTUKAKINADA	168	96	26	57.142857	15.476190
18	JNTUKAKIADAVIZIANAGARAM	336	240	26	71.428571	7.7380952

Table 2

S.No	Input Data String	Encrypted data Size(bits)	Huffman Compression		Arithmetic Compression	
			Compressed Data Size (bits)	Compression Ratio (Percentage)	Compressed Data Size (bits)	Compression Ratio (Percentage)
1	AAAAAAAAAAAA	168	24	14.285714	7	4.166666
2	AAAAAABBBBBB	168	48	28.571428	14	8.333333
3	AAAABBBBCCCC	168	72	42.857143	21	12.5
4	AAABBBCCCDDD	168	72	42.857143	24	14.285714

5	AABBCCDDEEFF	168	72	42.857143	26	15.47169
6	AABBCCDDDEEE	168	72	42.857143	26	15.47169
7	AAAABCCDDDD	154	66	42.857143	21	13.636363
8	ABCDEEEFFFG	168	96	57.142857	27	16.071428
9	ABCDEEEEEEF	154	88	57.142857	27	17.532467
10	DDEEFGHHHII	154	88	57.142857	27	17.532467
11	KLKLFGHIJJJ	154	88	57.142857	23	14.935064
12	AABCCCDHIJK	154	88	57.142857	26	16.883116
13	AAABBBCDEFJ	154	88	57.142857	24	15.584416
14	AAAABBBBCCC	154	66	42.857143	21	13.636363
15	IJKIJKIJKIJK	168	72	42.857143	19	11.309528
16	GHIJGHIJILAA	168	72	42.857143	25	14.880952
17	CCCDDDABCDEF	168	96	57.142857	20	11.904761
18	AABBBBKLKLKL	168	72	42.857143	24	14.285714

7. REFERENCES

[1] N.Koblitz, CM-curves with good cryptographic properties in: Advances in cryptology, CRYPTO'91, Lecture note in Computer Science,Vol.576, Springer 1991, pp 279-287

[2] National Institute of Standards and Technology (NIST), Digital Signature Standards(DSS), Federal information processing standard, FIPS PUB 186-2, January 27,2000.

[3] Certicom Research, SEC 1: Elliptic Curve Cryptography, Standards for efficient cryptography, September, 2000

[4] Certicom Research, SEC 2: Recommended Elliptic Curve domain parameters, Standards for efficient cryptography, September 20,2000

[5] V. Miller, "Uses of elliptic curves in cryptography", Advances in Cryptology– Crypto'85, Lecture Notes in Computer Science, 218 (1986), Springer-Verlag, 417-426.

[6] Certicom Corp., " An Introduction to Information Security", No. 1, March 1997.

[7] ANSI X9.63, Public Key Cryptography for the Financial Services Industry: Elliptic CurveKey Agreement and Key Transport Protocols, ballot version, May 2001.

[8] Internet Engineering Task Force, The OAKLEY Key Determination Protocol, IETF RFC 2412, November 1998.

[9] ISO/IEC 15946-3, Information Technology–Security Techniques– Cryptographic Techniques Based on Elliptic Curves, Part 3, Final Draft International Standard (FDIS), February 2001

[10] M. Jacobson, N. Koblitz, J. Silverman, A. Stein and E. Teske, "Analysis of the xednicalculus attack", Designs, Codes and Cryptography, 20 (2000), 41-64.

[11] S. Arita, "Weil descent of elliptic curves over finite fields of characteristic three", Advances in Cryptology–Asiacrypt 2000, Lecture Notes in Computer Science, 1976 (2000),Springer-Verlag, 248-259.

[12] Fernandes, A. "Elliptic Curve Cryptography", Dr.Dobb's journal, December 1999

[13] J.Solinas, "An improved algorithm for arithmetic on a family of elliptic curve", Advances in Cryptology - CRYPTO '97, Lecture Notes in Computer Science, 1997, Volume 1294/1997, 357-371, DOI: 10.1007/BFb0052248 ,1294(1997). Springer-Verlog, 357-371

[14] J.Solinas," Efficient arithmetic on koblitz Curves", Design codes and cryptography, 19(2000), 195-249

[15] Huffman, David (1952) "A Method for the Construction of Minimum Redundancy Codes," Proceedings of the IRE 40(9):1098–1101.

[16] O.Srinivasa Rao, S.Pallam Setty, "Efficient mapping methods of Elliptic Curve Crypto Systems" International Journal of Engineering Science and Technology, Vol. 2(8), 2010, pp. 3651-3656

[17] M.Prabu, R.Shanmugalakshmi "A Comparative and

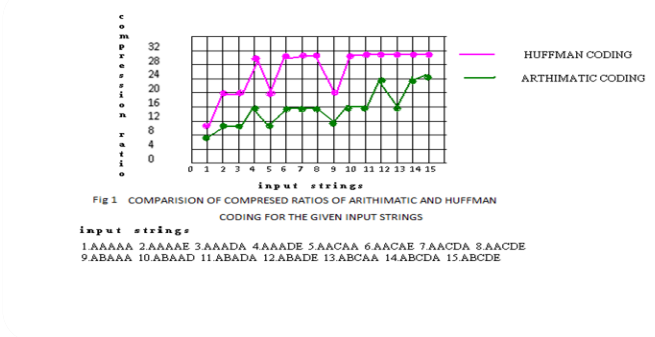

Fig 1 COMPARISION OF COMPRESED RATIOS OF ARITHIMATIC AND HUFFMAN CODING FOR THE GIVEN INPUT STRINGS

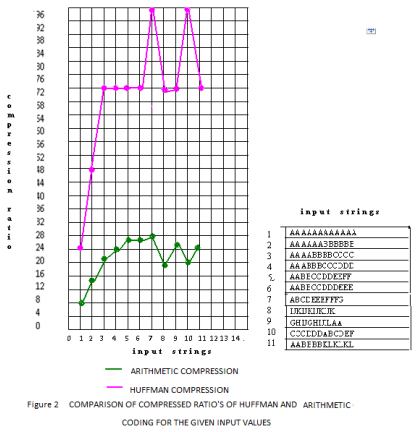

Figure 2 COMPARISON OF COMPRESSED RATIO'S OF HUFFMAN AND ARITHMETIC CODING FOR THE GIVEN INPUT VALUES

Overview Analysis of Elliptic Curve Cryptography over Finite Fields"2009, International Conference on Information and Multimedia Technology, IEEE computer society.

[18] Billy Bob Brumley and Kimmo U. Jarvinen, Member, IEEE "Conversion Algorithms and Implementations for Koblitz Curve Cryptography", IEEE Transactions on computers Vol.59, No.1, January 2010

[19] Yong-hee Jang, Yong-jin Kwon "Efficient Scalar Multiplication Algorithms Secure against Power Analysis Attacks for Koblitz Curve Cryptosystems" 2010, 10th Annual International Symposium on Applications and the Internet, IEEE Computer Society

[20] Chang Shu, Soonhak Kwon, and Kris Gaj "Reconfigurable Computing Approach for Tate Pairing Cryptosystems over Binary Fields" IEEE Transactions on computers Vol.58, No.8, September 2009