Covenant Journal of Informatics & Communication Technology. Vol. 5 No. 1, June, 2017

An Open Access Journal Available Online

Parameterised Key Diffusion Approach of AVK Based Cryptosystem

Shaligram Prajapat

International Institute of Professional Studies
Devi Ahilya University Indore
shaligram.prajapat@gmail.com

Abstract- The enhancement of security of information using symmetric key is the demand of industry and society. For sharing huge files and data symmetric key algorithms is first choice. This paper presents sparse matrix based approach (SAVK) for symmetric key based cryptosystem, exploiting location information for enciphering and deciphering of user's information. The sparse based method has been analyzed with three variants and performance of these approaches has been presented. The benefit of location information for encryption or decryption of information will find its applicability in moving low power gadgets, IOT components and auditing of cryptosystems.

Introduction

Secure information exchange is achieved by cryptographic algorithm. Asymmetric cryptic algorithms are not preferred for big sized files or in formations. Symmetric key based algorithms are suitable for recommended for cloud based information repositories. AES is the first choice as compared to DES, 3DES, BlowFish, RC6 ,Two Fish etc. To improve the performance and increase the degree of security Automatic Variable Key (AVK) based approach has been proposed by (C.T. Bhunia, 2008).Variety of approaches of AVK has been proposed in literature.(Prasun, 2008).Automatic Variable Key (AVK) approach is better alternative over longer sized key. AVK attempts to keep key-size constant and changes entire key in successive sessions. The optimum size of key been approximated up to 7 or 8 characters.(Shaligram and R.S. Thakur, 2015) . One approach of AVK is generation of key using Fibonacci-Q matrix (Shaligram, 2012) where by choosing various terms corresponding to different values of input parameters new keys of fixed length is generated. Another approach of for moving information sending and receiving equipment using , the location based AVK scheme has been discussed in this paper together with the realization and analysis.(Shaligram Prajapat, 2014) and

(Shaligram, 2016). The presented approach exploits the advantage of compact representation of sparse space with key variables using location information based (i, j) information for automatic variable key assuming the situation of moving transmitter and receiver for symmetric key based cryptosystem. The parameters *only* scheme have been used without exchanging entire key and the parameters have been diffused for key-construction using parameters only (shaligram,2016). The subsequent sections, present a scheme based on the sparse matrix approach for efficient and secure communication without using any key exchange.

Definition and Related Work

Sparse Matrix: A matrix of order very high dimension (m x n) having most of its member as zero. The sparse matrix representation schema records information of only non-zero members and discards zero members (Gilbert et. al. 1992). The threshold needed to fall a given square matrix to be "sparse-matrix" depends on the structure of the matrix, number of nonzero members and nature of operations to be performed on it. (W. H., Flannery et.al., 2014). By recording only nonzero member information provides substantial reduction in memory space. Depending on the number and distribution of the non-zero entries, different data structures can be used and yield huge savings in memory when compared to the conventional way of information representation. Consider matrix A of order 6 by 6.

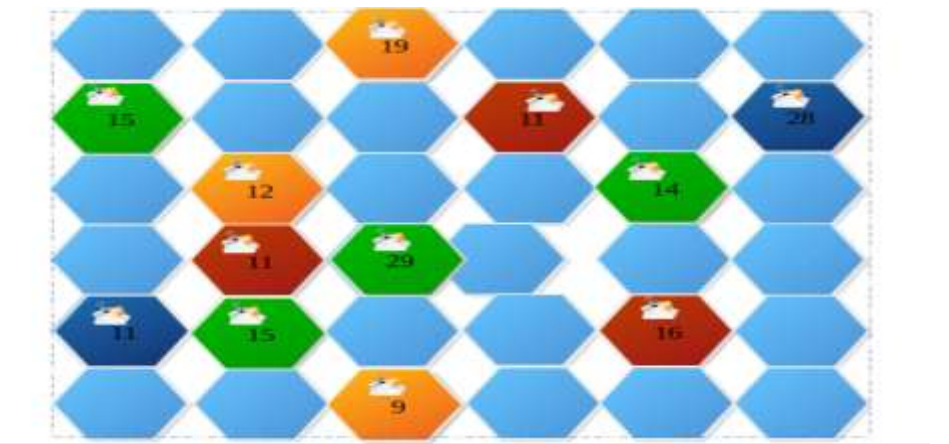

Figure 1: Physical location of data and corresponding data

In Figure 1 the partitioning of a geographical or physical area into small addressable location where moving devices at a location (i, j) exchanging data $d_{ij.}$ The corresponding logical equivalent matrix representation is shown as matrix A equation (1), where total 12 devices or information sharing nodes are currently located. The locations of these devices are represented as nonzero entry in the sparse matrix and only these information is recorded in compact

representation of sparse matrix. The equivalent sparse matrix of figure 1 .

$$A=\begin{bmatrix} 0 & 0 & 19 & 0 & 0 & 0 \\ 15 & 0 & 0 & 11 & 0 & 28 \\ 0 & 12 & 0 & 0 & 14 & 0 \\ 0 & 11 & 29 & 0 & 0 & 0 \\ 11 & 15 & 0 & 0 & 16 & 0 \\ 0 & 0 & 9 & 0 & 0 & 0 \end{bmatrix} \quad (1)$$

The Compact Sparse Matrix (CSM) representation of Sparse Matrix is equation (2).

$$CSM = \begin{matrix} \text{Header row} \rightarrow \\ \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \\ 7 \\ 8 \\ 9 \\ 10 \\ 11 \\ 12 \\ 13 \end{matrix} \begin{bmatrix} 6 & 6 & 12 \\ 0 & 2 & 19 \\ 1 & 0 & 15 \\ 1 & 6 & 28 \\ 1 & 3 & 11 \\ 1 & 5 & 18 \\ 2 & 1 & 12 \\ 2 & 4 & 14 \\ 3 & 1 & 11 \\ 3 & 2 & 29 \\ 4 & 0 & 11 \\ 4 & 1 & 15 \\ 4 & 4 & 16 \\ 5 & 2 & 19 \end{bmatrix} \quad (2)$$

By definition a sparse matrix may contain most of the member as zero, but actually what threshold is to be considered during implementation to be taken as a sparse matrix, is given by equation (4).

Necessary condition for being sparse matrix

Let ‘p’ denotes the count of nonzero members of A. For static memory allocation array A of size p would require 3-members (3-tuple) array storage for each nonzero member. To estimate reduction in storage space using compact way having only record of nonzero members using such representation let us compute total space required for storing A , m x n integer-members in a 2-D array. Obviously,

$$Numberofbytes\,required\ by\ A\ =\ \text{No.of rows (m)} * No.ofcolumns(n) * Size\_\ of\ \_\ integer \quad (3)$$

In CSM-version of spmat the memory efficiency can be achieved iff:

$$3\ *\ p *\ Size\_of\ \_integer\_inBytes\ \le m\ *\ n\ *\ Size\_of\ \_integer\_inBytes$$

$$p \le \frac{m*n}{3} \quad (4)$$

In other words, when total count of non-zero members is not exceeding 33% of the total count of members of entire matrix, then the compact storage representation of sparse structure would be beneficial. The matrix A may be represented more economically (in terms of space) using CSM if the conventional 2-D array representation of matrices is not used. Instead it is using representation of only non zero members only. Since 6 X 6 is the order of illustration of Figure 1. Using matrix "A", such that: no_of_rows(m) = 6 :no_of_columns(n) = 6 :no_of_member (Non zero entries p) = 12. This can be expressed also in the [0] position of following spmat array: csm [0][0]. Nonzero members of the array A from index [1...no_of_member] is denoted as:{0, 2, 19; 1, 0, 15; 1, 3, 11; 1, 5, 18; 2, 1, 12; 2, 4, 14; 3, 1, 11; 3, 2, 29; 4, 0, 11; 4, 1, 15; 4, 4, 16; 5, 2, 9 }. (Here notation '{ ' and '}' has been used for array to avoid confusion from reference /citation).CSM [ ][ ] information about a nonzero member has three parts:

1. An integer representing its row_index(i) or CSM[i][0].
2. An integer representing its column_index(j) or CSM[i][1].
3. The nonzero data associated with (i, j)_ location is dij or CSM[i][2].

Such a 3-tuple can be represented by a data structure with 3 fields:

$$csm[1]=[0,2,19],\ csm[2]=[1,0,15],\ csm[3]=[1,3,11],\ csm[4]=[1,5,18],$$
$$csm[5]=[2,1,12],\ csm[6]=[2,4,14],\ csm[7]=[3,1,11],\ csm[8]=[3,2,29],$$
$$csm[9]=[4,0,11],\ csm[10]=[4,1,15],\ csm[11]=[4,4,16],\ csm[12]=[5,2,9]\ \ldots$$

This representation not only saves memory but also stores key parameters that have been diffused in the first and second column of compact sparse matrix CSM[i][0] and CSM[i][1] respectively, where index i is variable subscript holding location of nonzero members in the compact sparse matrix CSM. The transformation of Sparse matrix into Compact Sparse Matrix notation is shown in Figure 2

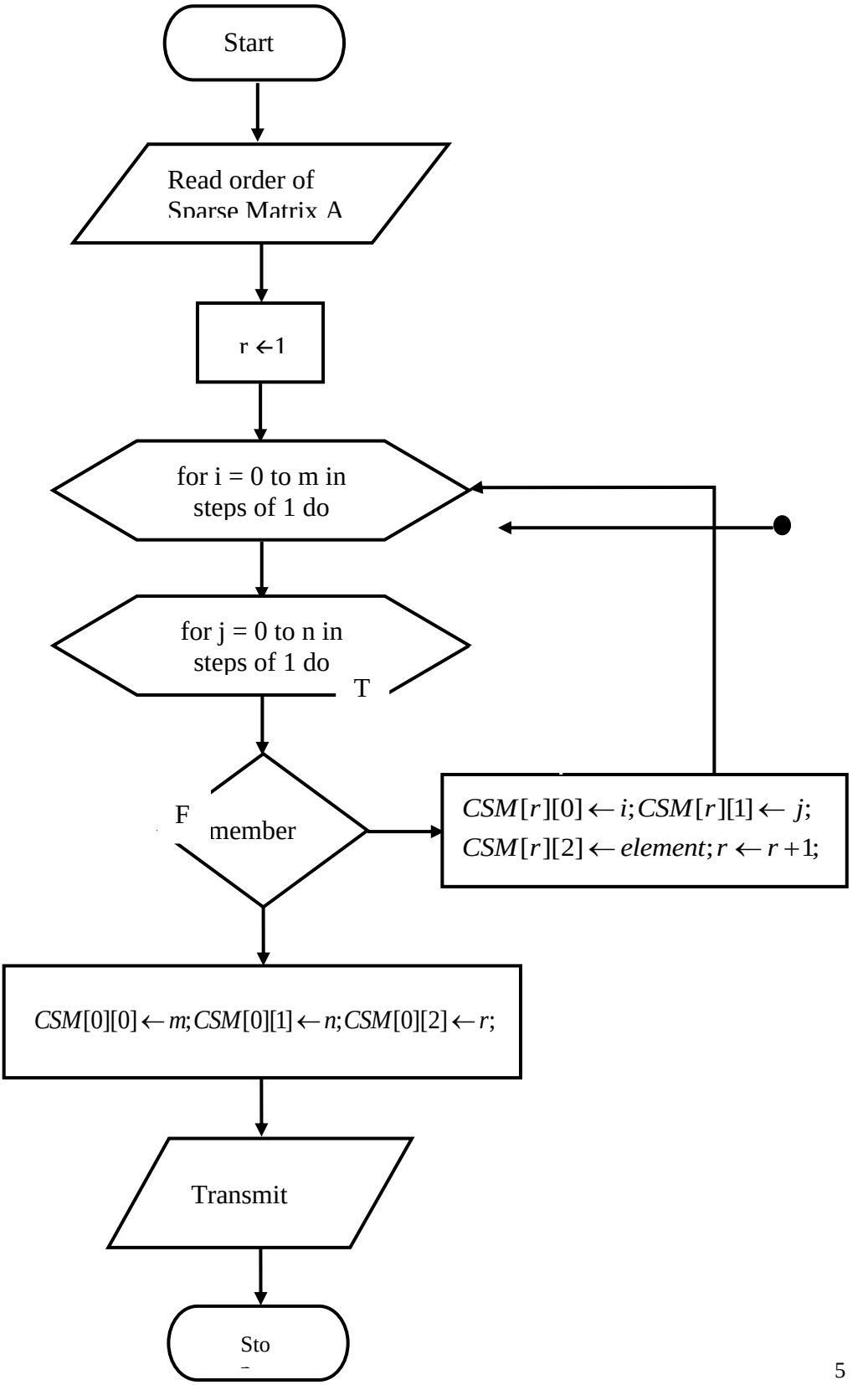
Start
Read order of
Sparse Matrix A
r ←1
for i = 0 to m in
steps of 1 do
for j = 0 to n in
steps of 1 do
T
F
member
CSM[r][0] ← i; CSM[r][1] ← j;
CSM[r][2] ← element; r ← r +1;
CSM[0][0] ← m; CSM[0][1] ← n; CSM[0][2] ← r;
Transmit
Sto

Figure 2: Transformation of Sparse Matrix into Compact Sparse Matrix (CSM)

Proposed Model of Encryption and Decryption (Diffused Parameters for Key SPARSE-AVK Algorithm)

This approach uses Linear Sparse based Symmetric AVK method (LSAVK) in Figure 3, Quadratic Sparse based Symmetric AVK method (QSAVK) in Figure 4 and Cubic Sparse based Symmetric AVK method (CSAVK) in Figure 5, for encryption and decryption using 1-dimensional,2-Dimensional,3-dimensional transformations using location parameters (i, j). Assuming the standard representation scheme the proposed algorithms works with row and column indexes starting from 1.

Proposed Algorithms for Ensuring Security

The transformations from plaintext to cipher text and recovery of plaintext from receiving cipher text will be done by from equations 5, to equation 10.

Linear Sparse AVK based Cryptic process

For a cipher generation of linear transmission using location parameters

$$ciphertext\_CSM'[i][2] = a + b * plaintext \quad (5)$$

For reconstruction of plain text from transformed linear ciphers using location parameters

$$plaintext\_CSM[i][2] \leftarrow \frac{CSM'[i][2] - CSM'[i][0]}{CSM'[i][1]} \quad (5)$$

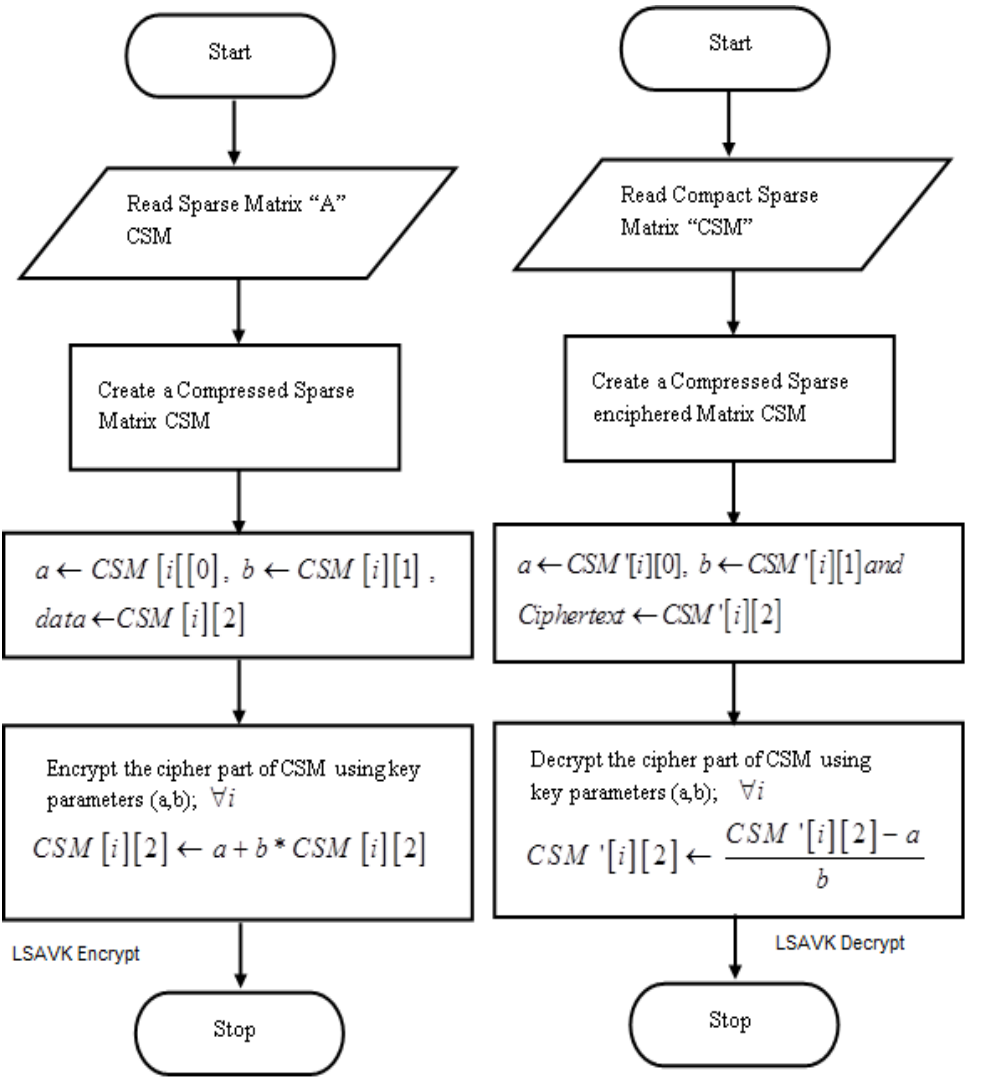

Figure 3: Proposed Cryptic Process for LSAVK

Quadratic Sparse based AVK Process (QSAVK)

For cipher generation of quadratic transformation location parameters

$$ciphertext = a + b * data + (a+b)^2 * data \quad (7)$$

For regeneration of plain text information from quadratic transformed cipher text

$$plain\ \text{text} = \frac{ciphertext - a}{b + (a+b)^2} \quad (8)$$

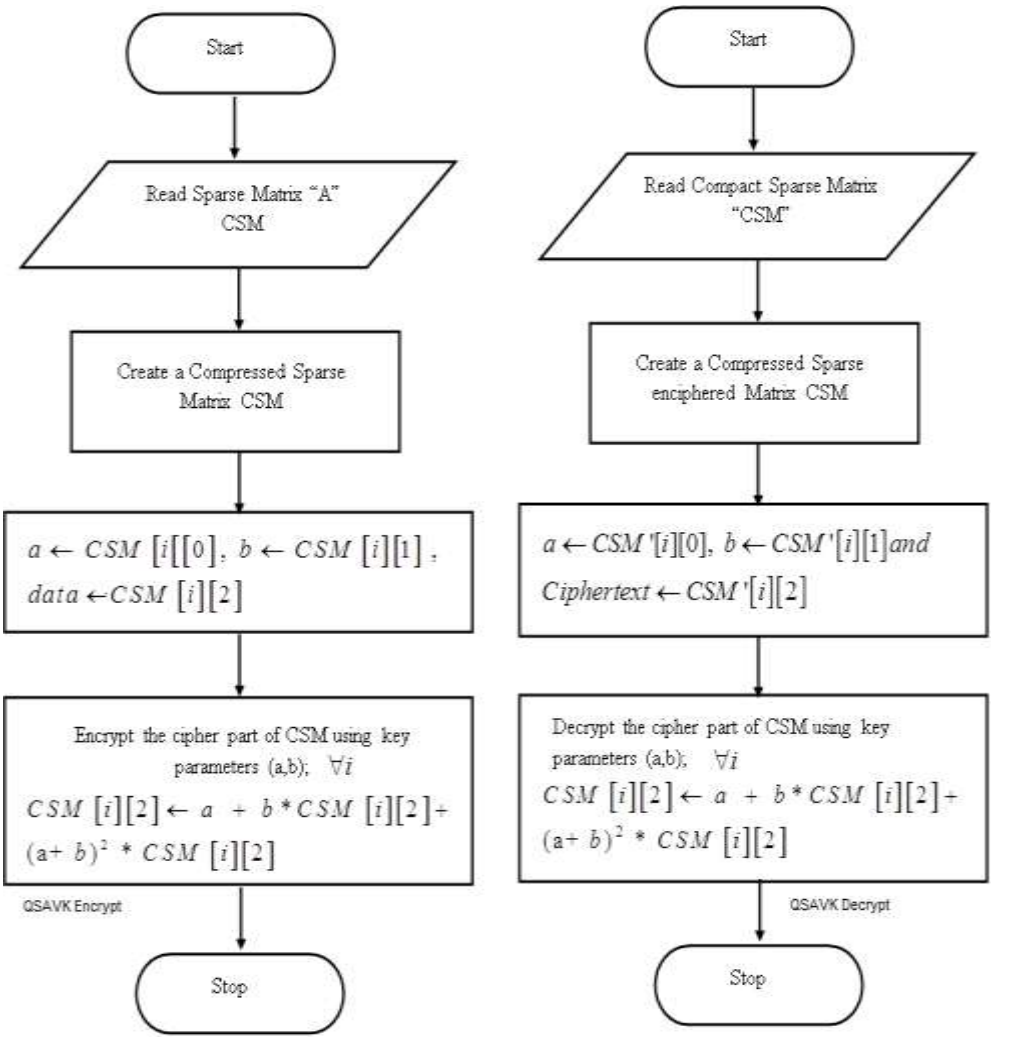

Figure 4: Proposed Process for transformation using QSAVK

Cubic AVK based Cryptic Process (CSAVK)

For cipher generation of cubic transformation location parameters

$$ciphertext = a + b * data + (a+b)^2 * data + (a+b)^3 * data \quad (9)$$

For regeneration of plain text information from cubic transformed cipher text

$$plain\ \text{text} = \frac{ciphertext - a}{b + (a+b)^2 + (a+b)^3} \quad (10)$$

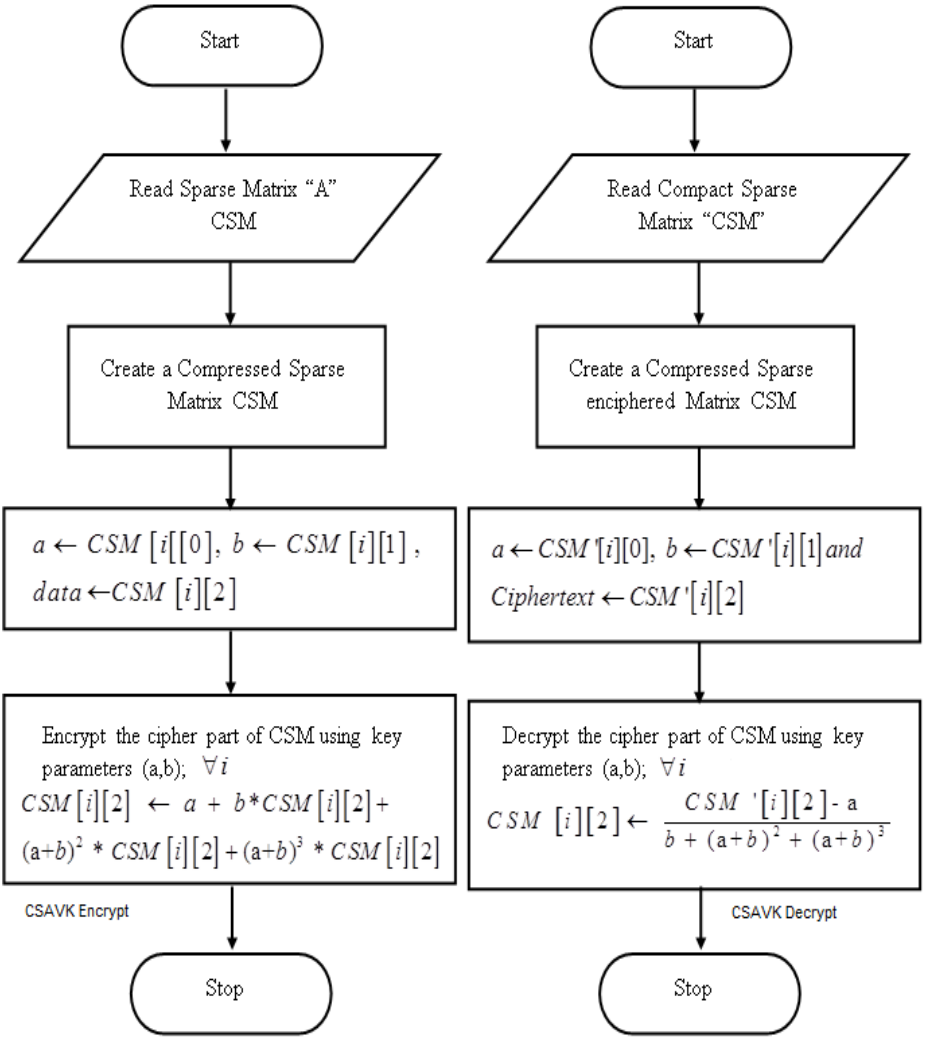

Figure 5: Proposed Process for transformation using CSAVK

Proposed Sparse Based AVK Algorithms (SAVK)

This section presents three cryptic algorithm based on linear(1-D),Quadratic(2-D) and Cubic (3-D) with encryption and decryption functionalities namely, LSAVKEncrypt( ) and LSAVKDecrypt( ) (Algorithm 1 and Algorithm 2) uses linear transformation ,QAVKEncrypt( ) and QSAVKDecrypt( ) (Algorithm 3 and Algorithm 4) uses quadratic transformation and CSAVKEncrypt( ) and CSAVKDecrypt( ) (Algorithm 5 and Algorithm 5) using cubic transformation under proposed Sparse based Symmetric Encryption /Decryption *(LSAVK Figure* 4, *QSAVK Figure* 5, *and CSAVK Figure* 5*)Schemes).*

Encryption Using Linear Transformations(1-D)

Algorithm1 LSAVK-Encrypt (Matrix CSM[ ])

{ // Receive plain text from sender with location information, generate cipher and transmit

for each i from 1 to CSM[0][3] in steps of 1 do

{ a ← CSM[i][0], b ← CSM[i][1], plaintext ← CSM[i][2];

Generate Cipher Text CSM'[i][2] ← a+b*plaintext;

Transmit Cipher Text(CSM'[ ])

}

}

Decryption Using Linear Transformations(1-D)

Sparse based Symmetric Decryption algorithms

Algorithm 2 LSAVK-Decrypt (Matrix CSM'[ ])

{ // Receive compact sparse matrix(cipher text) and recover plaintext information

for each i from 1 to CSM[0][2] in steps of 1 do

{ a ← CSM'[i][0], b ← CSM'[i][1], plaintext ← CSM'[i][2];

Generate Plain Text CSM'[i][3] ← (*CSM* '[*i*][3] – *CSM* '[*i*][0]) / *CSM* '[*i*][1];

return Plain-Text(CSM'[ ])

}

}

Encryption Using Quadratic Transformations(2-D)

Algorithm 3 QSAVK-Encrypt (matrix CSM[ ])

{ // Receive plain text from sender with location information

// and generate cipher and transmit

for each i from 1 to CSM[0][3] in steps of 1 do

{ a ← CSM[i][0], b ← CSM[i][1], plaintext ← CSM[i][2];

Generate Cipher Text CSM'[i][2] ← a+b*plaintext+$(a+b)^2$*plaintext;

Transmit Cipher Text(CSM'[ ])

}

}

Decryption Using Quadratic Transformations (2-D)

Algorithm 4 QSAVK-Decrypt (matrix CSM'[ ])

{

// *This algorithm accepts Compact Sparse Matrix with cipher text*

// and *recovers plain text in plaintext _ CSM'*[*i*]

for each i from 1 *to CSM* '[0][2] *do*

{

Set $a \leftarrow CSM'[i][0]$, $b \leftarrow CSM'[i][1]$, $plainText \leftarrow CSM'[i][2]$;

Generate plainText _ CSM $'[i] \leftarrow \frac{CSM'[i][2] - CSM'[i][0]}{CSM'[i][1] + (CSM'[i][0] + CSM'[i][1])^2}$;

return(*PlainText _ CSM* '[ ]);

}

}

Encryption Using Cubic Transformations(2-D)

Algorithm 5 CSAVK-Encrypt (matrix CSM[ ])

{ // Receive plain text from sender with location information

// and generate cipher and transmit

for each i from 1 to CSM[0][3] in steps of 1 do

{ a ← CSM[i][0], b ← CSM[i][1], plaintext ← CSM[i][2];

Generate Cipher Text CSM'[i][2] ← a+b*plaintext+$(a+b)^2$*plaintext+

$(a+b)^3$*plaintext;

Transmit Cipher Text(CSM'[ ])

}

}

Decryption using Cubic Transformations (2-D)

Algorithm 6 CSAVK-Decrypt (matrix CSM'[ ])

{

// *This algorithm accepts Compact Sparse Matrix with cipher text*

// and *recovers plain text in plaintext _ CSM'*$[i]$

for each i from 1 *to CSM* '$[0][2]$ *do*

{

Set a $\leftarrow$ *CSM* '$[i][0]$, *b* $\leftarrow$ *CSM* '$[i][1]$, *plainText* $\leftarrow$ *CSM* '$[i][2]$;

Generate plainText _ CSM '$[i] \leftarrow \frac{CSM'[i][2]-CSM'[i][0]}{CSM'[i][1]+(CSM'[i][0]+CSM'[i][1])^2+(CSM'[i][0]+CSM'[i][1])^3}$;

return(*PlainText _ CSM* '[ ]);

}

}

Analysis of LSAVK, QSAVK, CSAVK with Variation in Input File Size, Size of Parameters

The algorithm Linear AVK Encrypt (), Quadratic AVK Encrypt and Cubic AVK Encrypt accept compact form of sparse matrix entries and use location (index position) as a parameter for Cipher generation i.e. these algorithms utilize location information of nonzero member and converts the information into cipher text in linear, quadratic and cubic way. Similarly Linear AVK Decrypt () receives cipher text of the data item and based on its key (using the index position of the member as parameter) it recover, Quadratic AVK Decrypt ( ), Cubic AVK Decrypt ( ) original information. Since the key is not transmitted in the data transfer. So it becomes highly difficult to interpolate any information regarding plaintext or key. Table-1, Table 2 and Table 3 demonstrates the working of the proposed SAVK scheme. The sparse matrix recovered by Trudy (man in the middle) will be as follows:

{0,2,19; 1,0,15; 1,3,11; 1,5,18; 2,1,12; 2,4,14; 3,1,11; 3,2,29; 4,0,11; 4,1,15; 4,4,16; 5,2,9 }.

The data member corresponding to table-entries at row no. 3, 7 and 9 are same but after enciphering they are represented by different bit strings, this hides patterns of input plain text and making cryptic mining process hard. Since data enciphering is achieved by the location parameter (i, j) of device or data item, therefore the key would be variable and will change automatically for moving device. This variable key is making same data to become different cipher at different locations. Ciphers making position based variability in data items. LSAVK() is memory efficient due to storage of nonzero members only , $O(p+1) = O(p)$ and takes $O(n)$ time for processing, where p is number of nonzero items. Tables 1, Table 2 and Table 3 demonstrates working of cryptic algorithms LSAVK ( ), QSAVK ( ) and CSAVK( ) from sender (column-5 : Data Sent by Alice (Hex code)),Receiver(column-8: Data Received by BOB

(Recovered Text)) and man in middle (Column-6: Message bits on Noisy Channel).

Table 1: Illustration of LSAVK based Information Transmission

Index	i	j	M(i,j)	Data Sent by Alice (Hex code)	Message bits on Noisy Channel	Cipher on channel	Data Received by BOB (Recovered Text)
0	6	6	12	0C	00001110	4E	0C
1	1	3	19	13	00111010	3A	13
2	2	1	15	0F	00010001	11	0F
3	2	4	11	0B	00101110	2E	0B
4	2	6	18	12	01101110	6E	12
5	3	2	12	0C	00011011	1B	0C
6	3	5	14	0E	01001010	49	0E
7	4	2	11	0B	00011010	1A	0B
8	4	3	29	1D	01011011	5B	1D
9	5	1	11	0B	00010000	10	0B
10	5	2	15	0F	00100011	23	0F
11	5	5	16	10	01010101	55	10
12	6	3	09	09	00100001	21	09

Table 2: Illustration of QSAVK based Information Transmission

Index	I	j	M(i,j)	Data Sent by Alice (Hex Code)	Message bits on Noisy Channel	Cipher on Channel	Data Received by BOB (Recovered Hex Code)
0	6	6	12	0C	011100001110	70E	0C
1	1	3	19	13	000101101010	16A	13
2	2	1	15	0F	000010011000	098	0F
3	2	4	11	0B	000110111010	1BA	0B
4	2	6	18	12	010011101110	4EE	12
5	3	2	12	0C	000101000111	147	0C
6	3	5	14	0E	001111001001	3C9	0E
7	4	2	11	0B	000110100110	1A6	0B
8	4	3	29	1D	010111101000	5E8	1D
9	5	1	11	0B	000110011100	19C	0B

10	5	2	15	0F	001100000010	302	0F
11	5	5	16	10	011010010101	695	10
12	6	3	9	09	001011111010	2FA	09

Table 3: Illustration of CSAVK based Information Transmission

Index	I	j	M(i,j)	Data Sent by Alice (Hex Code)	Message bits on Noisy Channel	Cipher on channel	Data Received by BOB (Recovered Hex Code)
0	6	6	12	0C	0101100000001110	580E	0C
1	1	3	19	13	0000011000101010	062A	13
2	2	1	15	0F	0000001100111101	022D	0F
3	2	4	11	0B	0000101100000010	0B02	0B
4	2	6	18	12	0010100011101110	28EE	12
5	3	2	12	0C	0000011100100011	0723	0C
6	3	5	14	0E	0001111111001001	1FC9	0E
7	4	2	11	0B	0000101011101110	0AEE	0B
8	4	3	29	1D	0010110011010011	2CC3	1D
9	5	1	11	0B	0000101011100100	0AE4	0B
10	5	2	15	0F	0001011100011011	171B	0F
11	5	5	16	10	0100010100010101	4515	10
12	6	3	9	09	0001110010011011	1C9B	09

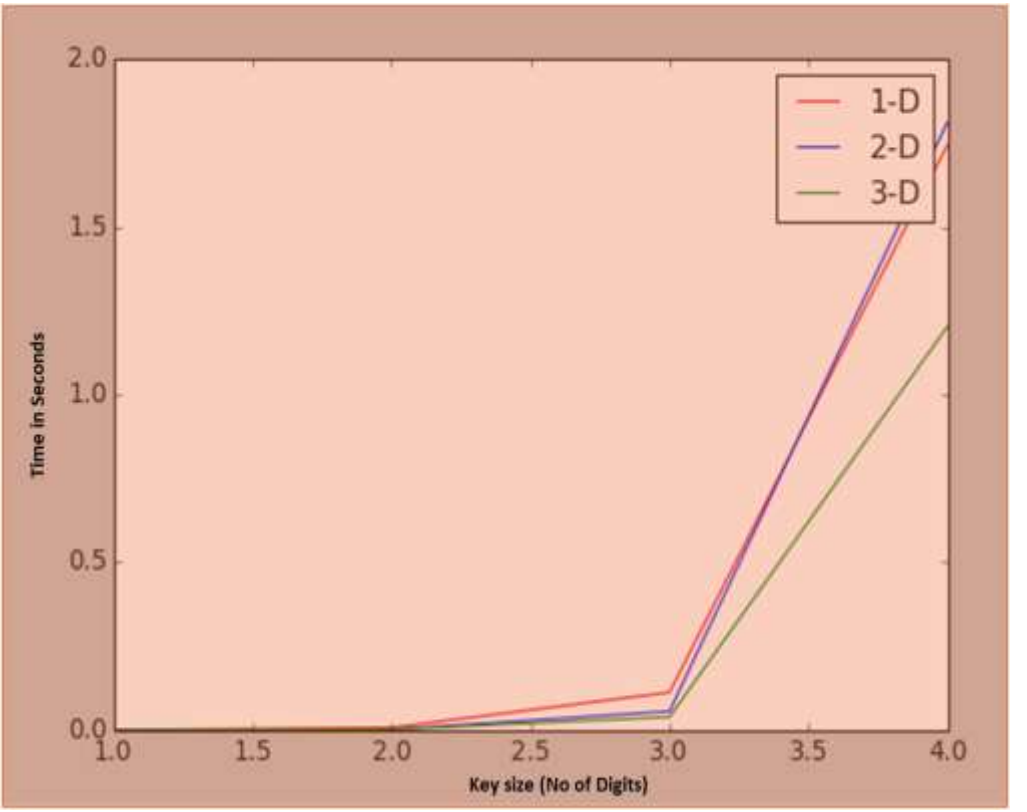

Figure 5: Key Size(X-axis) and corresponding time taken in Seconds(Y-axis) by LSAVK(),QSAVK(),CSAVK()

The performance of LSAVK, QSAVK and CSAVK are shown in figure 5, that shows with increasing the number of digits of parameters from 3 digits onwards the time taken by these algorithms varies fast. In Figure 7 the relative performance of the algorithm is presented, form both the figures it is clear that the performance of CSAVK is better.

Discussions and Future Directions

The encryption algorithms LSAVKEncrypt(), QSAVKEncrypt() and CSAVKEncrypt() use curves of linear, Quadratic and Cubic relationships that are parameterized by a and b. It is worth to check whether these relationships are manifested into produced ciphertext. If yes then it is better to apply rot-13 type of transformation before encryption process. The reverse of it can be applied at recipient end i.e. by LSAVKDecrypt(),QSAVKDecrypt()and CSAVKDecrypt().

Parameter Size (In Number of Digits) v/s. Max. Execution Time

The effect of increasing key size increasing non-linearly when numbers of digits are increased from 3.the table

Table 4: Key size (In Number of Digits) v/s. Execution Time

Key size (number of digits)	Sparse Matrix		
	1 degree decryption Execution Time (sec)	2 degree decryption Execution Time (sec)	3 degree decryption Execution Time (sec)
1	0.0003	0.0002	6.6996
2	0.0019	0.0019	0.0003
3	0.0337	0.0418	0.0346
4	0.7451	1.8633	0.7454

The Performance of Proposed Algorithms with Variable File Size.
In Table 5 demonstration of effect on the decryption time (in seconds) taken for various input size in Bytes. Whereas the Table 5 shows the decryption time (in seconds) taken for various input size in Bytes.

Table 5: Comparative Execution Times (In Sec) of Secret Key Algorithm

Input Size (Bytes)	LSAVK (Sec)	QSAVK (Sec)	CSAVK (Sec)
20527	0.00271	0.00306	0.00394
36002	0.00493	0.00479	0.00556
45911	0.0126	0.00602	0.00677
59862	0.00746	0.00889	0.00834
69646	0.00853	0.00913	0.00963
137325	0.02251	0.01904	0.01921
158959	0.02081	0.02173	0.02217
166364	0.02965	0.02603	0.02621
191383	0.03049	0.03154	0.03126
232398	0.03234	0.03326	0.03330

The result of table 5 is shown in figure 7 and it clears that time required by LSAVK (), QSAVK(), CSAVK( ) increases with increase in file size. The performance of LSAVK is sensitive towards file size whereas QSAVK() and CSAVK() are in close competition. CSAVK() performance is stable over LSAVK() and QSAVK().

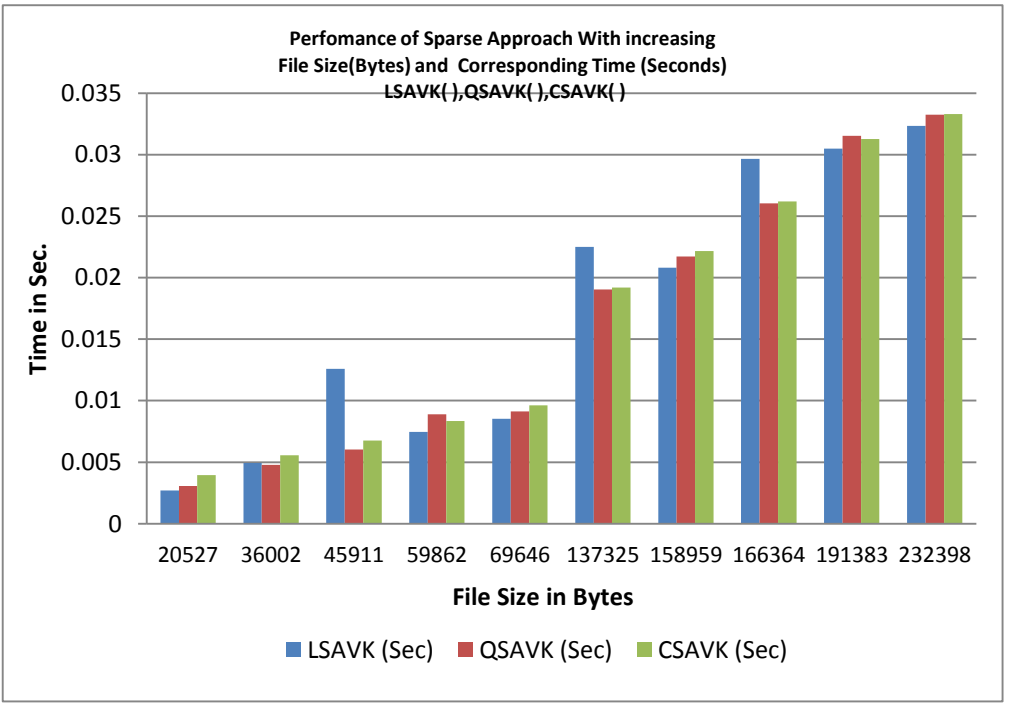

Figure 7: Relative comparison of LSAVK( ),QSAVK( ) and CSAVK( )

Conclusion

This paper presents six cryptic algorithms that apply three different degrees and performance comparison. The relative result comparison of the three approaches show better performance of CSAVK. The beauty of this approach that it does not require key exchange and diffused parameters set for key construction. The proposed technique open new perspectives for secure communication using AVK approach for low power devices together with saving the key computation time.

References

Shaligram Prajapat , Ramjeevan Singh Thakur , "Key Diffusion Approach for AVK based Cryptosystem", In proceedings of Second International Conference on Information and Communication Technology for Competitive Strategies, March 2016. Article No. 78,,ISBN: 978-1-4503-3962-9 doi:10.1145/2905055.2905288.

Shaligram Prajapat, Shashank Swami, Bhagirath Singroli, R. S. Thakur, Ashok Sharma, D. Rajput," Sparse Approach for Realizing AVK for Symmetric Key Encryption", International Journal of Recent Development in Engineering and

Technology ,Vol.2(4), pp. 15-18.June 2014.

Chakrabarti P., Bhuyan B., Chowdhuri A., and Bhunia C., A novel approach towards realizing optimum data transfer and Automatic Variable Key (AVK) in cryptography, IJCSNS International Journal of Computer Science and Network Security, Vol. 8, No. 5, pp. 241, 2008.

Chakrabarti P, "Application of. Automatic Variable Key (AVK) in RSA". Int'l J HIT Transactions on ECCN, Vol.2,. No. 5, Jan-Mar 2007, pp. 304-311.

Chakrabarti et al., Various New and Modified approaches for selective encryption (DES, RSA and AES) with AVK and their comparative study, published in International Journal HIT Transactions on ECCN, Vol 1, No.4, p. 236-244. 51

Bhunia, C. T., Application of AVK and selective encryption in improving performance of quantum cryptography and networks, United Nations Educational Scientific and Cultural Organization and International Atomic Energy Agency, retrieved, Vol.10, No. 12, pp. 200-210, 2006.

Bhunia C. T., New Approaches for Selective AES towards Trackling Error Propagation Effect of AES, Asian Journal of Information Technology, Pakistan, Vol. 5, No. 9, pp. 1017-1022, 2006.

Bhunia C. T., Chakrabarti P., Chowdhuri A. and Chandan T., Implementation of Automatic Variable Key with Choas Theory and Studied Thereof, J IUP Computer Science, Vol -5, No 4, pp. 22-32, 2011.

Bhunia C.T., Mondal G. and Samaddar S., Theories and Application of Time Variant Key in RSA and that with selective encryption in AES, Proc. EAIT, Elsevier Publications, Calcutta CSI-06, pp. 219-221, 2006.

Dutta M.P., Banerjee S. and Bhunia C., Two New Schemes to Generate Automatic Variable Key (AVK) to achieve the Perfect Security in Insecure Communication Channel, Proceedings of the 2015 International Conference on Advanced Research in Computer Science Engineering & Technology (ICARCSET 2015), pp.1-4, 2015.

Prajapat, Shaligram, D. Rajput, Ramjeevan Singh Thakur, "Time variant approach towards symmetric key", In proceedings of IEEE Science and Information Conference (SAI), London 2013. , pp.398-405, 2013.

Prajapat, Shaligram, Ramjeevan Singh Thakur. "Optimal Key Size of the AVK for Symmetric Key Encryption." In Covenant Journal of Information & Communication Technology,Vol.3(2), pp. 71-81. 2015.

Prajapat, Shaligram, Ramjeevan Singh Thakur. "Various Approaches towards Crypt-analysis." International Journal of Computer Applications, Vol. 127(14), pp. 15-24, 2015. (doi: 10.5120/ijca2015906518)

Prajapat, Shaligram, Ramjeevan Singh Thakur. "Cryptic Mining for Automatic Variable Key Based

Cryptosystem", Elsevier Procedia Computer Science, Vol. 78 (78C), pp. 199-209, 2016. (doi: doi:10.1016/j.procs.2016.02.034) .

Prajapat, Shaligram, Ramjeevan Singh Thakur."Cryptic Mining: Apriori Analysis of Parameterized Automatic Variable Key based Symmetric Cryptosystem."International Journal of Computer Science and Information Security, Vol. 14 (2), pp. 233- 246, 2016.

Prajapat, Shaligram, Ramjeevan Singh Thakur. "Realization of information exchange with Fibo-Q based Symmetric Cryptosystem." International Journal of Computer Science and Information Security ,Vol 14(2), pp. 216-223, 2016.

Prajapat, Shaligram, Thakur, A., Maheshwari, K., & Thakur, R. S., "Cryptic Mining in Light of Artificial Intelligence", IJACSA , Volume 6(8), pp. 62-69, 201510.14569/IJACSA.2015.060808) .

Prajapat shaligram (2016), Towards parameterized shared key for AVK approach, Chapter 4 ,Pattern and Data Analysis in Health care settings, IGI Global Publications.

W. H., Flannery, B. P., Teukolsky, S. A., Vetterling, W. T. "Sparse Linear Systems. in Numerical Recipes in FORTRAN: The Art of Scientific Computing", 2nd ed. Cambridge, England: Cambridge University Press, pp. 63-82, 1992.

Tim Davis, http://mathworld.Wolfram .com/ topics/ DavisTim.html, April 2014.

Gilbert, J. R, Moler, C. Schreiber, R., "Sparse Matrices in MATLAB: Design and Implementation", SIAM J. Matrix Anal. Appl. 13, 333-356, 1992.