International Journal of Research Studies in Computer Science and Engineering (IJRSCSE)
Volume 3, Issue 3, 2016, PP 27-34
ISSN 2349-4840 (Print) & ISSN 2349-4859 (Online)
DOI: http://dx.doi.org/10.20431/2349-4859.0303004
www.arcjournals.org

Algorithms for String Perturbation

Geetha Mary A

School of Computing Science and Engineering, VIT University, Vellore, Tamil Nadu, India
geethamary.a@gmail.com

Abstract: *In this paper we propose a model in which the perturbation is done by randomization, where the data is generated in intervals in order to preserve the sensitive data. This paper explores the possibility of using random number generation algorithms to protect the most privacy sensitive data, hospital data records. The following tests with privacy preserving data mining and data perturbation show how random number generation algorithm show promising possibilities in preserving the sensitive hospital data records.*

Keywords: *Privacy, data mining, random number, perturbation, string operations.*

1. INTRODUCTION

Hospital Data Records are one of the most privacy-sensitive data. Therefore, we need to develop data-mining techniques sensitive to the privacy issues. This has fostered the development of a class of data mining algorithms that try to extract the data patterns without directly accessing the original data and guarantees that the mining process does not get sufficient information to reconstruct the original data. This paper considers a class of techniques for privacy-preserving data mining by randomly perturbing [6] the data while preserving the underlying probabilistic properties. It explores the random value perturbation based approach, a well-known technique for masking the data using randomizing the given entries of the data set.

The level of privacy desired by each individual differs based on the needs of an individual and the sensitivity of the data. Liu et al.[10] has discussed about two phase perturbation in which random data is added with original data to generate a dummy data. This dummy data is further randomized within a desired level of interval. The interval decided is directly proportional to the level of security desired by the individual. Once the randomization is attained, a reconstruction mechanism is done to retrieve back the original data with slightest possible deviation.

In Health Insurance Portability and Accountability Act i.e. HIPAA [9] , signed in 1996 by President of United States, the security rules were updated in 2006, which specifically stated about technical safe guard of patient details. According to the specifications of the Patient Safety and Quality Improvement Act of 2005 (PSQIA), the confidentiality of the data was emphasized upon with allowance of minimum disclosure of the same. 8.3 million people as of March 2011 and 10 million people as of April 2011 were affected by security breaches. Almost once in every 3 months, privacy attacks happen and personal health information theft takes place. Nowadays patient details like transcription information are outsourced from the USA to other countries like India, so while doing this patient details must be safeguarded.

In Li et al. [12], various forms of the perturbed data are generated based on the trust level denoted by the miner. They have also discussed about the diversity attack, where the miner could get any form of data and try to generate the oirginal data from the retrieval mechanisms. They have concluded by providing an algorithm which generates perturbed copies of a database on demand. These copies are generated by additive data perturbation.

Organizations follow some actions to limit disclosure of data while they go for publishing; they try to publish the data in such a manner that the sensitive information such as turnover of an organization, salary of an employee or diseases of a person should not be identified. In general to achieve this, personal identifiers like name, social security number, employee identity, voter's identity and hospital identity are eliminated from the published data. However, there is a possibility to re-identify an individual by using some background information and data linkage techniques. This limitation can be avoided by using privacy preserving data mining techniques (PPDM). Usually data is either distorted

or generalized in PPDM, but the main decisive factor is the level to which the data is to be distorted or generalized so that there is no extensive change in exactness of data or the knowledge developed from it.

Privacy Preserving Data Mining (PPDM) is the main focus of scientists in order to preserve the sensitive data of people. Starting from health care, banking, customer details to defence all the data is concerned to either an individual or a company and has different levels of disclosure. Nowadays privacy of Health care information of patients is hindered and is under constant attack, but at the same time this information needs to be analysed for research purposes. To distinguish between the disclosure of data for research and for unethical purposes, PPDM techniques plays a major role. Zhang[11] states that PPDM process is divided into three tiers, Data providers tier: where data collection takes place, Data warehouse tier: where data is converted into OLAP for easier processed data like aggregates, sum, average etc. and the top tier – Data Mining server: where Analysis is done according to the requirement of the research. The main concern is always with the Data Provider tier where Collection of data happens. In the collected data though the key fields like patient number, Social security number and name are discarded and given for analysis but they are still prone to lead to the disclosure of the identify of an individual by record linkage which is discussed primarily with many methods in certain papers. Again, these disclosures are handled by different methods like k-anonymity, l-diversity and t-closeness. T-closeness is the improved version of k-anonymity and l-diversity. These method leads to generalization and suppression of attributes which lead to major loss of data.

In [8,13, 14, 15, 16], various PPDM techniqes have been discussed. In [13] all the techniques are analyzed based on the privacy measures, kinds of algorithms used and data types on which the techniques are usually applied. In [8], Mohammadreza et al. concludes by specifying the merits and demerits of each the techniques given in [13].

Perturbation is also a major technique followed in PPDM introduced by Agarwal et al [10] in 2000. Random noise is added, now the noisy data along with the distribution is given to the data miner, who reconstructs the data for analysis. Reconstruction algorithm should be effective in such a way that loss of accuracy is as low as possible. Perturbation is mostly implemented by adding random data – additive data perturbation and multiplying with a matrix – multiplicative data perturbation. Perturbation are mostly implemented in two phases, first adding noise to original data, then checking for distribution, i.e. maintaining a mean of zero and allowed variance.

Given a hospital dataset with a description of patient diseases, we wish to compute the probability of occurrence of a disease and randomize it in order to preserve the privacy of the patients from the unauthorised users. We wish to deduce algorithms to convert tag values to string values and randomly arranging the diseases in the dataset.

2. ALGORITHMS

We now describe our algorithm that can be used to exactly (without approximation) attain a randomized set of data.

2.1. Shuffling

Let the diseases be divided into the different branches of medicine [7] say:

1. Neurology
2. Orthopedics
3. Psychiatry
4. Cardiology

Group the diseases into sub-branches of medicine. Let Neurology have 10, Orthopedics 7, Psychiatry 11 and Cardiology 9 diseases respectively. Make a table with rows equal to the total no of branches and columns equal to Maximum number of diseases (In this case 4*11). Arrange diseases of each branch in the given rows (left aligned).Now insert diseases at random intervals subjected to "Maximum Spread Rule" (Use Quadratic Hashing).Collect the entire cells column by column, discarding the empty cells [3].Append all the columns to get the randomly generated order of all diseases. Figure 1 presents the flowchart of the method.

The probability distribution of the above applied Maximum Spread Rule can be drawn to check the probable availability of each branch of medicine as per shuffling algorithm, which is depicted in Figure 5.

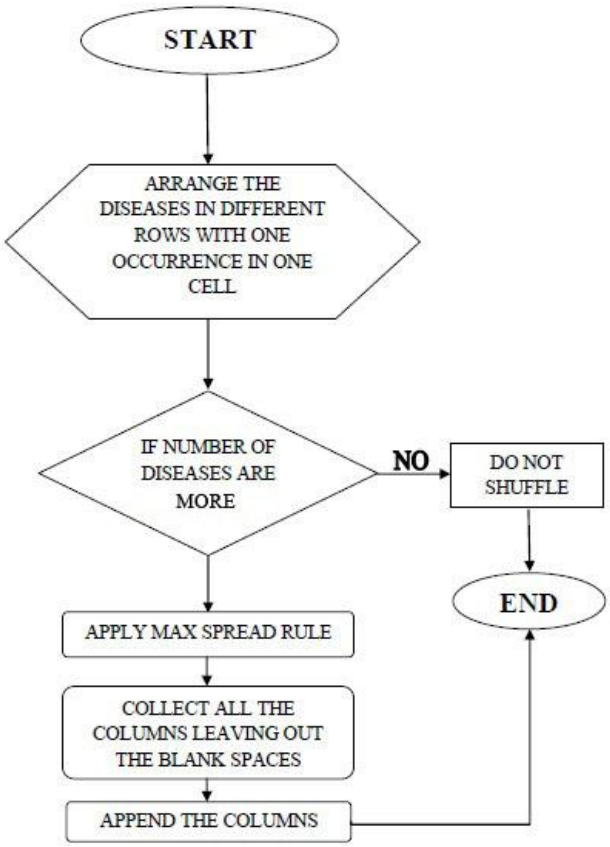

Figure1. *Flowchart for shuffling*

N	N	N	N	N	N	N	N	N	N	
O	O	O	O	O	O	O				
P	P	P	P	P	P	P	P	P	P	P
C	C	C	C	C	C	C	C	C		

Figure2. *Before Max Spread*

N	N	N	N		N	N	N	N	N	N
O		O	O	O		O		O		O
P	P	P	P	P	P	P	P	P	P	P
C		C	C	C	C		C	C	C	C

Figure3. *After Max Spread*

N	N	N	N		N	N	N	N	N	N
O	X	O	O	O		O		O		O
P	P	P	P	P	P	P	P	P	P	P
C	X	C	C	C	C		C	C	C	C

Figure4. *Taking Columns*

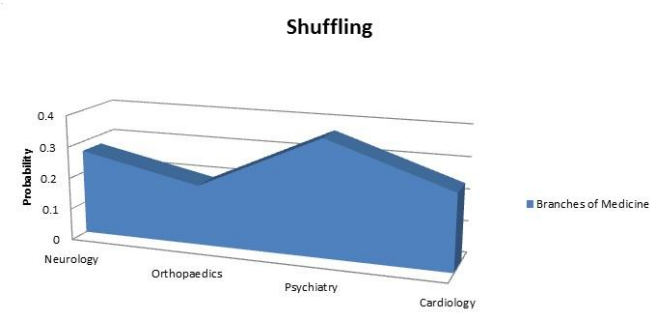

Figure5. *Probability distribution of the dataset*

2.2. Random Seed

Fetch all the diseases arranged under the different sub-branches of medicine. Arrange all the diseases in one row and number them.

The serial number allotted to the diseases can act as their key numbers and the last serial number is the table size. Multiply number 9 (This constant is the randomseed) with the system time in milliseconds (System time in milliseconds can be generated by getTimeInMillis()method found under java.util. Calendar).The above product is the new value of random seed. Break the random seed into groups of two digits and assign the number to each disease serial number wise. Write the random seed in cascading manner such that it is capable of assigning a new key to each disease. Perform Linear Probing for common key values and the random order will be generated.

```
import java.util.Calendar;

public class GetTimeInMilliSeconds {
public static void main(String[] args) {
Calendar cal = Calendar.getInstance();
System.out.println(cal.getTimeInMillis());
}
}
```

Figure6. *Java code for system time generation in miliseconds*

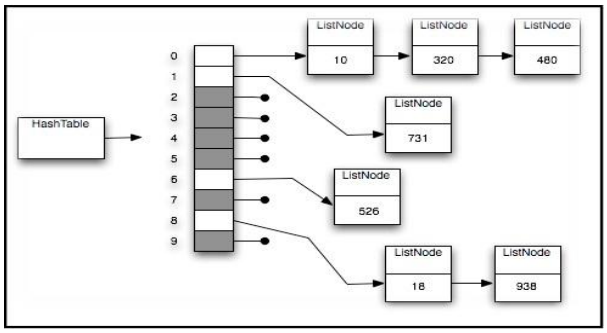

Figure7. *Linear Programming*

2.3. Arithmetic Coding

Arithmetic Coding [17] is a form of variable-length entropy encoding used in lossless data compression. Arithmetic coding, is entropy coder widely used, the only problem is it's speed, but compression tends to be better than Huffman can achieve. This presents a basic arithmetic coding implementation, if you have never implemented an arithmetic coder, this is the article which suits your needs, otherwise look for better implementations. Normally, a string of characters is represented using a fixed number of bits per character, as in ASCII code. When a string is converted to arithmetic encoding, frequently used characters will be stored with fewer bits and not-so-frequently occurring characters will be stored with more bits, resulting in fewer bits used in total.

Underflow occurs when both high and low get close to a number but theirs msb don't match: High = 0,300001 Low = 0,29997 if we ever have such numbers, and the continue getting closer and closer we'll not be able to output the msb, and then in a few iterations our 16 bits will not be enough, what we have to do in this situation is to shift the second digit (in our implementation the second bit) and when finally both msb are equal also output the digits discarded.

The tag values thus obtained are randomized by using Random() method and assigned to each patient. These tag values are then decoded.[2,4,5]

low = low + range x (low_range)
high = high + range x (high_range)
tag value = (lower_interval+upper_interval)/2

Symbol	A	B	C	D
Probability	0.4	0.2	0.1	0.3
Sub Range	(0.0-0.4)	(0.4-0.6)	(0.6-0.7)	(0.7-1.0)

Figure8. *Symbol, Probability and its sub range tabulation*

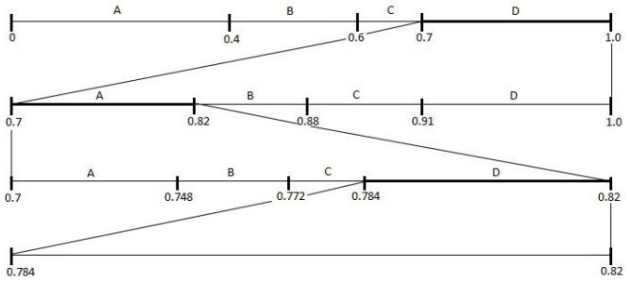

Figure9. *Computing lower and upper interval*

3. ANALYSIS

To demonstrate that the given algorithms can be used as a random source for protecting the privacy sensitive data, we must analyze the complexity and efficiency of the stated algorithms. By and large the running time of a random number algorithm is random but the complexity of all the tested algorithms was found out to be an order of n.

$T(n) = O(n)$

The probability distribution function $\Pr(X)$ also gives a result of unity which justifies the above proves

$$\sum_{u} \Pr(X = u) = 1$$

where u runs through all the possible values of *X*.

3.1. Shuffling

If a computer has access to purely random numbers, it is capable of generating a "perfect shuffle", a random permutation of the cards differs from "a perfectly executed single shuffle", notably a perfectly interleaving faro shuffle. The Fisher–Yates shuffle, popularized by Donald Knuth, is simple (a few lines of code) and efficient (O(n) on an n-card deck, assuming constant time for fundamental steps) algorithm for doing this. Shuffling can be seen as the opposite of sorting. There are other, less-desirable algorithms in common use. For example, one can assign a random number to each card, and then sort the cards in order of their random numbers. This will generate a random permutation, unless any of the random numbers generated are the same as any others (i.e. pairs, triplets etc.). This can be eliminated either by adjusting one of the pair's values randomly up or down by a small amount, or reduced to an arbitrarily low probability by choosing a sufficiently wide range of random number choices. If using efficient sorting such as mergesort or heapsort this is an O(n log n) average and worst-case algorithm

Shuffling can be done on the column with details of the Doctors. After applying the max spread on the column, the data takes up the following order:

N S N M N N S

Hence, the modified data is presented in Table 2.

3.2. Random Seed

Depending on the algorithm the seed can indeed affect the quality of the random numbers however due to the design of mersenne twister the choice of the seed doesn't really affect the length of the periodicity of the random numbers or their quality.

A random seed makes it more time-consuming to crack a list of perturbed data. However, it does not make dictionary attacks harder. If the attacker has access to both the randomized data and the random seed, the dictionary attack can be a brute force to retrieve the real data. To understand the difference

between cracking a single data entry and a set of them, consider a single password file that contains hundreds of usernames and passwords. Without a random seed, an attacker could compute hash(attempt[0]), and then check whether that hash appears anywhere in the file. The likelihood of a match, i.e. cracking one of the passwords with that attempt, increases with the number of passwords in the file. If salts are present, then the attacker would have to compute hash(salt[a] . attempt[0]), where "." denotes concatenation, compare against entry A, then hash(salt[b] . attempt[0]), compare against entry B, and so on. This defeats "reusing" hashes in attempts to crack multiple passwords.

The random seed when generated for the last column in the sample table produces Table 3.

3.3. Arithmetic Coding

Arithmetic coding, in conjunction with a suitable probabilistic model, can provide nearly optimal data compression. It has been analyzed that the effect of the model and the particular implementation of arithmetic coding also affect code length obtained. Periodic scaling is often used in arithmetic coding implementations to reduce time and storage requirements; it also introduces a regency effect, which can further affect compression. The concept of weighted entropy and it's usage in characterizing the effect that periodic scaling has on the code length is of much use. The analysis of why and by how much scaling increases the code length for files with a homogeneous distribution of symbols, and the characterization of reduction in code length due to scaling of files exhibiting locality of reference. A rigorous proof of the coding effects of rounding scaled weights, integer arithmetic, and encoding end-of-file are negligible.

Scaling is the process in which we periodically reduce the weights of all symbols. It allows us to use lower precision arithmetic at the expense of making the model more approximate, which can hurt compression when the distribution of symbols in the file is fairly homogeneous. Scaling also introduces a locality of reference (or regency) effect, which often improves compression when the distribution of symbols is variable. In this section we give a precise characterization of the effect of scaling on code length produced by an adaptive model. We express the code length in terms of the weighted entropies of the model. The weighted entropy is the Shannon entropy, computed using probabilities weighted according to the scaling process; the term "weighted entropy" is a notational convenience. Our characterization explains why and by how much scaling can hurt or help compression.

Table1. *Sample Patient data set*

Patient ID	Registration Date	Name	Address	Maritial Status	Gender	Father/ husband name	Age	Referred to
HDJ1101	13-01-2008	AbhaTyagi	Bhubaneshaw	Married	Female	Suresh Tyagi	38	Dr. M. Nair
HDJ13045	01-11-2008	Smriti Nanda	Delhi	Unmarried	Female	Pankaj Nanda	21	Dr. Neha Gaur
HDJ14033	10-02-2009	IshanVij	Chandigarh	Married	Female	Rajesh Vij	39	Dr.ShreyDatta
HDJ14214	23-02-2009	Madhur Vyas	Meerut	Married	Male	Ram C. Vyas	65	Dr.ShreyDatta
HDJ15005	02-05-2009	Damodar Pant	Delhi	Married	Male	Giriraj Pant	53	Dr. Neha Gaur
HDJ15016	19-09-2009	Devarshi Gandhi	Surat	Unmarried	Male	Sandeep Gandhi	28	Dr. Neha Gaur
HDJ15071	18-12-2009	LataParanjpe	Mumbai	Married	Female	Mohan Paranjpe	33	Dr. Neha Gaur

Table2. *Patient data after shuffling*

Patient ID	Registration Date	Name	Address	Maritial Status	Gender	Father/husband name	Age	Referred to
HDJ1101	13-01-2008	AbhaTyagi	Bhubaneshawar	Married	Female	Suresh Tyagi	38	Dr. Neha Gaur
HDJ13045	01-11-2008	Smriti Nanda	Delhi	Unmarried	Female	Pankaj Nanda	21	Dr.ShreyDatta
HDJ14033	10-02-2009	IshanVij	Chandigarh	Married	Female	Rajesh Vij	39	Dr. Neha Gaur
HDJ14214	23-02-2009	Madhur Vyas	Meerut	Married	Male	Ram C. Vyas	65	Dr. M. Nair
HDJ15005	02-05-2009	Damodar Pant	Delhi	Married	Male	Giriraj Pant	53	Dr. Neha Gaur
HDJ15016	19-09-2009	Devarshi Gandhi	Surat	Unmarried	Male	Sandeep Gandhi	28	Dr. Neha Gaur
HDJ15071	18-12-2009	LataParanjpe	Mumbai	Married	Female	Mohan Paranjpe	33	Dr.ShreyDatta

Table3. *Patient data set after applying random seed*

Patient ID	Registration Date	Name	Address	Maritial Status	Gender	Father/husband name	Age	Referred to
HDJ1101	13-01-2008	AbhaTyagi	Bhubaneshawar	Married	Female	Suresh Tyagi	38	Dr. M. Nair
HDJ13045	01-11-2008	Smriti Nanda	Delhi	Unmarried	Female	Pankaj Nanda	21	Dr. Neha Gaur
HDJ14033	10-02-2009	IshanVij	Chandigarh	Married	Female	Rajesh Vij	39	Dr.ShreyDatta
HDJ14214	23-02-2009	Madhur Vyas	Meerut	Married	Male	Ram C. Vyas	65	Dr. Neha Gaur
HDJ15005	02-05-2009	Damodar Pant	Delhi	Married	Male	Giriraj Pant	53	Dr.ShreyDatta
HDJ15016	19-09-2009	Devarshi Gandhi	Surat	Unmarried	Male	Sandeep Gandhi	28	Dr. Neha Gaur
HDJ15071	18-12-2009	LataParanjpe	Mumbai	Married	Female	Mohan Paranjpe	33	Dr. Neha Gaur

Table4. *Patient data set after arithmetic coding*

Patient ID	Registration Date	Name	Address	Maritial Status	Gender	Father/husband name	Age	Referred to
HDJ1101	13-01-2008	AbhaTyagi	Bhubaneshawar	Married	Female	Suresh Tyagi	38	Dr. Neha Gaur
HDJ13045	01-11-2008	Smriti Nanda	Delhi	Unmarried	Female	Pankaj Nanda	21	Dr.ShreyDatta
HDJ14033	10-02-2009	IshanVij	Chandigarh	Married	Female	Rajesh Vij	39	Dr.ShreyDatta
HDJ14214	23-02-2009	Madhur Vyas	Meerut	Married	Male	Ram C. Vyas	65	Dr. Neha Gaur
HDJ15005	02-05-2009	Damodar Pant	Delhi	Married	Male	Giriraj Pant	53	Dr. Neha Gaur
HDJ15016	19-09-2009	Devarshi Gandhi	Surat	Unmarried	Male	Sandeep Gandhi	28	Dr. Neha Gaur
HDJ15071	18-12-2009	LataParanjpe	Mumbai	Married	Female	Mohan Paranjpe	33	Dr. M. Nair

The given data when operated under arithmetic coding, gives the following probability and sub range graph is specified in Figure 10.

Symbol	N	S	M
Probability	0.57	.28	0.15
Sub Range	(0.00-0.57)	(0.57-0.85)	(0.85-1.00)

Figure10. *Symbol, Probability and its sub range tabulation under arithmetic coding*

Hence, after computing the lower and the upper interval, the data is obtained as shown in Table 4.

4. CONCLUSION

We have put forward a candidate methodology to randomly distribute the data of a given hospital dataset to protect the privacy and the sensitivity of the data. The patient health data, which is highly sensitive, can bring the identity of an individual under risk and the above implemented methodology is a way to safeguard it and minimize the risks tending towards PHI.

Using the notion of weighted entropy, we have precisely characterized the tradeoff between the overhead associated with scaling and the saving that it can be realized by exploiting locality of reference. The largest code length savings come from more sophisticated models and the scaling analysis extends accordingly. We have also proven that the computational effects on the code length using practical arithmetic coding implementations are small, so we can use the exact algorithm.

In this paper we have proposed improved version of the existing algorithms in order to overcome its limitations. We have taken a numerical example in order to analyze the improved versions of the algorithms with the existing versions. The results obtained for the improved versions are identified as better compared with existing algorithms. The results obtained also show the practical viability of the proposed research.

REFERENCES

[1] Charles W. O'Donnell, G. Edward Suh, and Srinivas Devadas, 2004. "PUF Based Random Number Generation", MIT CSAIL, Technical Memo.

[2] S.Jayaraman, S. Esakkirajan, T. Veerakumar, 2011. "Digital Image Processing", Tata McGraw-Hill Education.

[3] "The Art of Music Shuffling", KeyJ's blog. http://keyj.emphy.de/balanced-shuffle/

[4] P. G. Howard and J. S. Vitter, 1994. "Arithmetic Coding for Data Compression", Proceedings of the IEEE, vol. 82, No. 6, pp.857-865.

[5] Ian H., Neal,RadfordM.,and Cleary,John G., 1987. "Arithmetic Coding for Data Compression", Communications of the ACM, pp 520-540.

[6] HillolKargupta and Souptik Datta, Qi Wang and Krishnamoorthy Sivakumar, 2005. "Random Data Perturbation Techniques and Privacy Preserving Data Mining", Knowledge and Information Systems, Vol. 7, No. 4.

[7] UCI Machine Learning Repository. [online] http://archive.ics.uci.edu/ml/datasets.html (Accessed 17 April 2014).

[8] Mohammadreza K , Somayyeh S M., Classification and Evaluation the Privacy Preserving Data Mining Techniques by using a Data Modification-based Framework, International Journal on Computer Science and Engineering, vol.3, no.2, pp. 862-870, (2011).

[9] HIPPA, 2006, U.S. Department of Health and Human Services, http://www.hhs.gov/ocr/privacy/ hipaa/administrative/index.html (Accessed 17 April 2014).

[10] Liu L, Kantarcioglu M, Thuraisingham B , The applicability of the perturbation based privacy preserving data mining for real-world data, Data and Knowledge Engineering, vol.65, no.1, pp. 5-21, (2008).

[11] Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, Clifford Stein, 2009. "Introduction to Algorithms", The MIT Press, third edition.

[12] Yaping L, Minghua C, Qiwei L, Wei Z., Enabling Multi-Level Trust in Privacy Preserving Data Mining, IEEE Transactions on Knowledge and Data Engineering, vol. PP, no.99, pp. 1-17, (2011).

[13] Gayatri N, Swagatika D, A Survey On Privacy Preserving Data Mining: Approaches And Techniques, International Journal of Engineering Science and Technology, vol.3, no.3, pp. 2127-2133, (2011)

[14] Charu C. Aggarwal, Philip S. Yu,, 2008. "Privacy-preserving Data Mining: Models and Algorithms", Springer.

[15] Nan Z, Wei Z. , 2007. "Privacy Preserving Data Mining Systems", IEEE – Computer, Vol. 40, No. 4, pp.52-58.

[16] Geetha Mary A., D.P. Acharjya, N. Ch. S. N. Iyengar, "Improved Anonymization Algorithms for Hiding Sensitive Information in Hybrid Information System", International Journal of Computer Network and Information Security, Vol. 6, No.6, pp. 9-17.

[17] Paul G. Howard, Jeffrey Scott Vitter, Analysis of arithmetic coding for data compression, Information Processing & Management, Vol. 28, No. 6, pp. 749-763.