Synonymous Keyword Search Over Encrypted Data in Cloud

Avani Konda, Sai Praneeth Gudimetla, Balaji T, Gopi Krishna Subramanyam, Usha Kiruthika
Department of Computer Science and Engineering,
SRM University, Ramapuram Campus,
Bharthi Salai, Ramapuram,
Chennai-089, Tamil Nadu

Abstract: In order to reduce cost in data management and to outsource the data to cloud servers, data owners in more amounts are motivated in cloud computing. Before outsourcing for privacy requirements, confidential data must be encrypted and data utilization such as keyword-based document has to be retrieved. To support dynamic update operations such as deletion and insertion process, a secure multi-keyword ranked search scheme over encrypted cloud data is produced. In general, the index construction and query generation of the vector space model and TF_IDF model that is widely used are combined. "Greedy Depth-first Search", a special tree-based index structure algorithm to provide multi-keyword ranked search that is efficient is proposed. To encrypt the index and query process, the secure KNN algorithm is used. Due to the use of our special tree-based index structure, the proposed scheme can achieve sub-linear search time and deal with the deletion and insertion of documents flexibly.

I INTRODUCTION:

Cloud computing store data into the cloud on demand high quality applications and services. Data owners can be relived from the burden of data storage and maintenance. If cloud server are not in the same trusted domain, the data may be at risk.

Data encryption makes effective data utilization that there could be large amount of outsourced data files. Users can retrieve files through keyword-based search instead of retrieving all the encrypted files back. Such technique can be applied in plain text

search scenarios. Data encryption restricts user's ability to perform keyword search. It also demands the protection of keyword privacy because few keywords contains important information about data files. Searchable encryption schemes build's index for each keyword and associate with the files. Searching input might be synonyms of the pre-defined keywords, not the exact or fuzzy [1] matching keywords due to the possible synonym substitution. Synonym based multi keyword ranked search over cloud data remains a very challenging problem.

For the effective search system, efficient and flexible searchable scheme which supports both multi keyword ranked search and synonym based search. Vector space model [2] (VSM) is used to build document index to address multi keyword search and result ranking. Each document is expressed as a vector where each dimension value is the term frequency (TF) [2]. The vector as the same dimension with document index where each dimension value is the inverse document

frequency (IDF) [2] weight. Cosine measure is used to compute similarity of a document search query and to improve search efficiency, tree based index structure is used. The proposed system solves the problem of synonym-based multi-keyword ranked search over encrypted cloud data. The practical and effective multi-keyword text search can be achieved by making contributions in two major aspects, supporting similarity-based ranking for more accurate search result and a tree-based search algorithm that achieves better-than linear search efficiency

II RELATED WORK

Searchable encryption**:** Traditional searchable encryption has been widely studied in the context of cryptography [3]. Among those works, most are focused on efficiency improvements and security definition formalizations. In the first construction of searchable encryption each word in the document is encrypted independently under a special two-layered encryption construction. In the index table, each entry consists of the trapdoor of a keyword and an encrypted set of file identifiers whose corresponding data files contain the keyword. Note that all these existing schemes support only exact keyword search, and thus are not suitable for Cloud Computing.

k-nearest neighbors algorithm**:** In pattern recognition, the K-nearest neighbors algorithm (KNN) [4] is a non-parametric method used for classification and regression. In both cases the input consists of the k closest training examples in the feature space. The output depends on whether KNN is used for classification or regression:

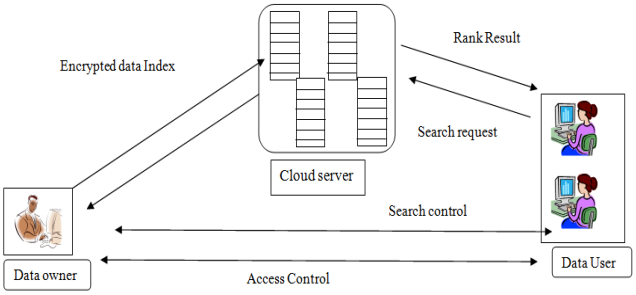
In KNN classification, the output is a class membership [5]. An object is classified by a majority vote of its neighbors, with the object being assigned to the class most common among its *k* nearest neighbors (*k* is a positive integer, typically small). If $k = 1$, then the object is simply assigned to the class of that single nearest neighbor.

Wildcard-based Fuzzy Set Construction (WFSC)**:** This scheme to enable fuzzy keyword search over encrypted cloud data. The key concept behind WFSC is maintaining an index that covers all possible variations of a keyword within a predefine edit distance. The number [1] of wildcard character used to modify the keyword is based on a predefine edit distance value. The drawback of this system is that with the increase in the edit distance value, the search quality is improved but there is huge increase in modified keyword set.

Dictionary-based Fuzzy Set (DFSC)**:** Instead of injecting keyword with the wildcard character, DFSC uses a dictionary to pull in only the valid words that are within the range of the predefine edit distance to form the modified keyword set. Even DFSC shows better storage usage than WFSC, DFSC inherited the [1] search quality degradation problem because more unrelated keywords can still be pulled into the modified key set from the dictionary as edit distance value increases. This problem decreases the search coverage of DFSC and makes it less accurate than WFSC. The major weakness of previously investigated schemes is that they are not considering user real intent of search.

Supports only identical keyword search, semantic search is not supported. Data owner has to perform all pre-processing tasks. Index encryption is sequential. To address multi-keyword search and result ranking, Vector Space Model (VSM) is used to build document index, that is to say, each document is expressed as a vector where each dimension value is the Term Frequency (TF) weight of its corresponding keyword. A new vector is also generated in the query phase. The vector has the same dimension with document index and its each dimension value is the Inverse Document Frequency (IDF) weight.

III ARCHITECTURE



Data Owner: has a collection of documents. He wants to store these documents on cloud storage. Data owner perform preprocessing, index construction, encryption and upload of documents.

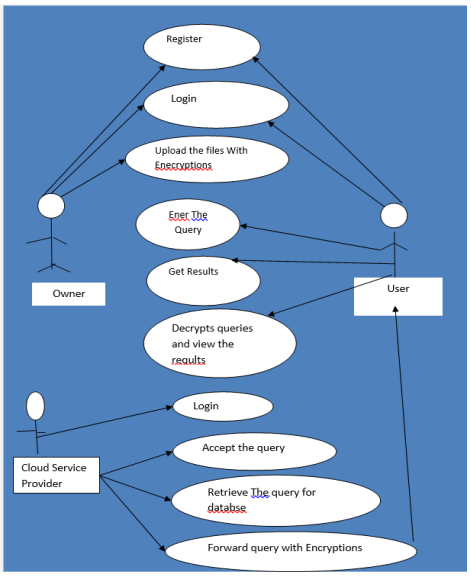
Data Users: wants to access these files. Using search keywords and secure key a trapdoor TD is generated. Trapdoor TD is used to search. After search processes, the cloud server sends most relevant k files to user. The retrieved documents are decrypted using secrete key.

Cloud Server: Upon receiving request form user, the cloud server searches the index tree and sends back top k relevant files to user.

Ranked Keyword Search**:** In this technique the search results are well ranked. This technique gives user most relevant document in the relevance order with query. And user gets relief [6] from sorting through every match in the content collection. This technique avoids unnecessary traffic. But they are only useful single keyword search

Multi-keyword ranked Search**:** This technique is to define and solve privacy preserving problem of multi-keyword ranked search and establish a variety of privacy requirements [1]. In this scheme the dictionary words are extracted from documents. And the queries and documents are expressed as vectors. The elements in the vector are the normalized TF values. The documents in the result are ranked by matching the vector co-ordinates.

IV ENCRYPTED DATA SEARCH



Keyword Extraction**:** TF-IDF stands for term frequency-inverse document frequency, and the TF-IDF weight is a weight often used in information retrieval and text mining. This weight is a statistical measure used to evaluate how important a word is to a document in a collection or corpus. The importance increases proportionally to the number of times a word appears in the document but is offset by the frequency of the word in the corpus. Variations of the TF-IDF weighting scheme are often used by search engines as a central tool in scoring and ranking a document's relevance given a user query.

Typically, the TF-IDF weight is composed by two terms: the first computes the normalized Term Frequency (TF), aka. The number of times a word appears in a document, divided by the total number of words in that document; the second term is the Inverse Document Frequency (IDF), computed as the logarithm of the number of the documents in the corpus divided by the number of documents where the specific term appears.

FREQUENCY:

Term frequency [5] measures how frequently a term occurs in a document. Since every document is different in length, it is possible that a term would appear much more times in long documents than shorter ones. In the case of the term frequency tf(t,d), the simplest choice is to use the *raw count* of a term in a document, i.e. the number of times that term t occurs in document d. If we denote the raw count by $f_{t,d}$, then the simplest tf scheme is tf(t,d) = $f_{t,d}$. Other possibilities include

- augmented frequency, to prevent a bias towards longer documents, e.g. raw frequency divided by the maximum raw frequency of any term in the document:

$$tf(t,d)=0.5+0.5.\frac{f_{t,d}}{\max\{f_{t,d}:t\in d\}}$$

Inverse document frequency [5] measures how important a term is. While computing TF, all terms are considered equally important. However it is known that certain terms, such as "is", "of", and "that", may appear a lot of times but have little importance. The inverse document frequency is a measure of how much information the word provides, that is, whether the term is common or rare across all documents. It is the logarithmically scaled inverse fraction of the documents that contain the word, obtained by dividing the total number of documents by the number of documents containing the term, and then taking the logarithm of that quotient.

$$IDF\ (T.D) = \log\frac{N}{|\{d\in D: t\in d\}|}$$

TF-IDF weighting**:** We now combine the definitions of term frequency and inverse document frequency, to produce a composite weight for each term in each document. The tf-idf weighting scheme assigns to term **t** a weight in document b given by

$$tf\text{-}idf_{t,d} = tf_{t,d}\ X\ idf_t$$

At this point, we may view each document as a vector with one component corresponding to each term in the dictionary, together with a weight for each component. For dictionary terms that do not occur in a document, this weight is zero. This vector form will prove to be crucial to scoring and ranking. As a first step, we introduce the overlap score measure: the score of a document *d* is the sum, over all query terms, of the number of times each of the query terms occurs in *d*. We can refine this idea so that we add up not the number of occurrences of each query term t in d, but instead the TF-IDF weight of each term in d.

$$Score\ (q,d)=\sum tf\text{-}idf_{t,d}$$

Synonym Expansion**:** The Words are analyzed in WordNet so that the related terms can be found for use in the index file. WordNet is a lexical database for the English language. A common use of WordNet is to determine the similarity between words. It groups English words into sets of synonyms called synonyms. All synonyms [2] are connected to other synonyms by means of semantic relations. Individual synonym members (words) can also be connected with lexical relations. For example, (one sense of) the noun "director" is linked to (one sense of) the verb "direct".

Two kinds of building blocks are distinguished in the source files: word forms and word meanings. Word forms are represented in their familiar orthography; word meanings are represented by synonym sets. Two kinds of relations are recognized: lexical and semantic. Lexical relations hold between word forms; semantic relations hold between word meanings.

Similarity Ranking**:** The sequential search process for keywords in a search request conducts as follows: the procedure starts from the root node and when arrives at an internal node u , if at least a keyword, then, it continues [5] to search both subtrees of u , otherwise stops searching in the

subtree because none of leaf node contains keyword in search query. Vector space model or term vector model is an algebraic model for representing text documents (and any objects, in general) as vectors of identifiers, such as, for example, index terms. It is used in information filtering, information retrieval, indexing and relevancy rankings. Documents and queries are represented as vectors.

$d_j = (w_{1,j} , w_{2,j} \ldots . w_{t,j})$

$q = (w_{1,q} , w_{2,q} \ldots . w_{t,q})$

Each dimension corresponds to a separate term. The cosine of two non-zero vectors can be derived by using the Euclidean dot product formula:

$d.q = ||d|| \, ||q|| \cos\theta$

Given two vectors of attributes, *D* and *Q*, the cosine similarity, *cos(θ)*, is represented using a dot product and magnitude as ,

$$\text{Similarity} = \cos(\theta) = \frac{D.Q}{||D||||Q||} = \frac{\sum_{i=1}^{n} DiQi}{\sqrt{\sum_{i=1}^{n} Di^2}\sqrt{\sum_{i=1}^{n} Qi^2}}$$

where D_i and Q_i are components of vector and respectively. The resulting similarity ranges from −1 meaning exactly opposite, to 1 meaning exactly the same, with 0 indicating orthogonality (decorrelation), and in-between values indicating intermediate similarity or dissimilarity.

V Graph Analysis

According to the pervious system algorithms had been used that are useful for proposed system. The improvements in the algorithms are also done. Here, are the some analysis to original and the improved system.

5.1 Analysis of Extraction method

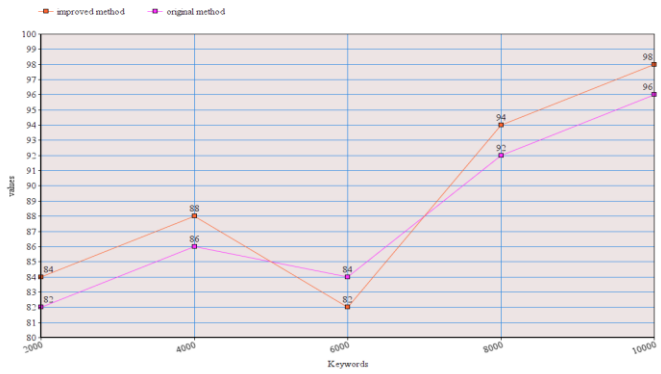


Figure 5.1Accuracy of Two Methods

The performance of the KNN and the IDF in the proposed system has to be verified. So, KNN algorithm for the train documents and the required result had achieved. The document is classified as per the similarity of the test document.

Accuracy of the both methods combining with the KNN algorithm are shown in the above Fig5.1. It shows that the values of the improved method is greater than the original method combining with KNN algorithm. The maximum value of original method is lower than the proposed method. Other than this there are additional things, the standard deviation of the proposed method is lower than original method, which shows that the proposed method reduces dependence on the number of features.

5.2 **Index tree performance**

The basic and improved schemes had measured based on the time cost and synonym based method of the index tree construction. From the previous system, the index tree construction is effected due to the number of keywords i.e., used in dictionary.

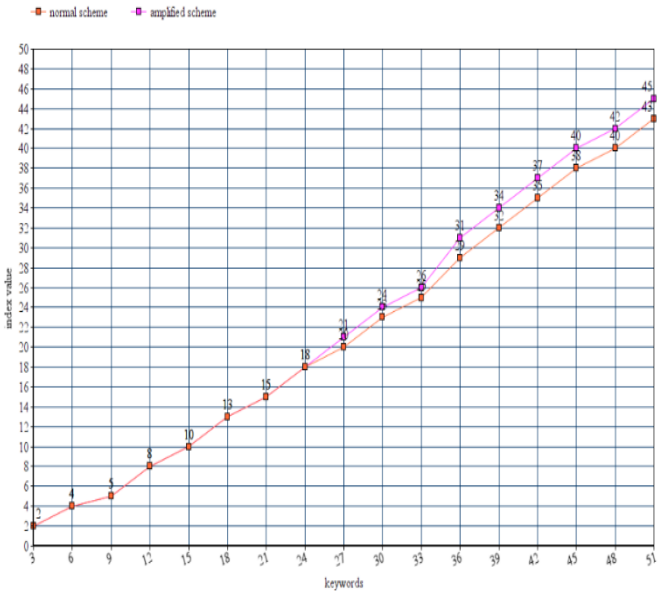


Figure 5.2 Analysis of time for both the schemes

The above graph indicates the time cost for constructing index tree is proportional to the number of keywords with the same dataset.

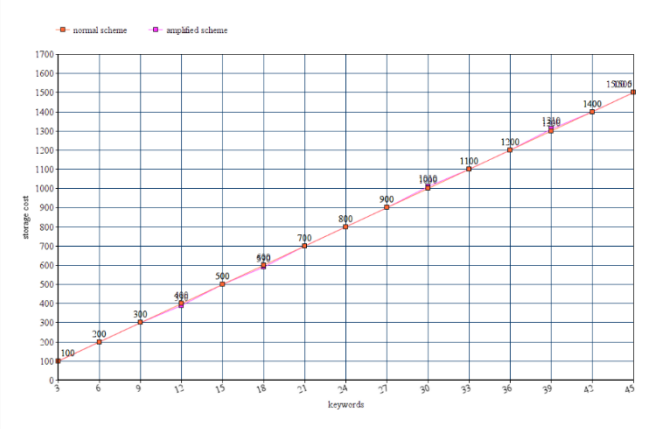


Figure 5.3 Analysis of Index Storage Cost

The above graph indicates that the sizes of index tree are very close in two schemes i.e., storage space is not a main problem in the cloud computing environment, because the index data only consume a small amount of storage space.

VI Future Work & Conclusion

The next work is to research semantics-based search approaches over encrypted cloud data that support other natural language processing technology. Natural language processing is a field of computer science, artificial intelligence, and linguistics concerned with the interactions between computers and human (natural) languages. In most of the cases this activity concerns processing human language texts by means of Information extraction. To improve the security access control policy will be enhanced. The term access control refers to the practice of restricting access to particular files. An access control system determines who is allowed to access the file uploaded. The access policy such as public and private are defined. The data owner who wishes to keep their file hidden from other user will provide access policy as private where the file will be stored in a separate folder and will be displayed only to that particular user. When the data owner provides the access policy as public to his uploaded file then the file will displayed to all data user who ever generates search query related to that file semantics. The data user will be provided the facility of rating the document being viewed. Once user provides the search query, all the documents that hold the exact query keyword or the semantically related keyword are listed. The user after accessing the file, if he/she finds the file relevant to their search query the user can provide the rating to that file. The user can provide either higher or the lower rating to the document based on the relevance he/she finds with the document. If same query is provided by any user for the next time, the document with higher rating will be resulted first. The aim of adding all the features to the cloud search service is that the cloud consumers can search the most relevant products or data by using the designed system.

VII References

[1] B. Wang, S. Yu, W. Lou, and Y. T. Hou, "Privacypreserving multi-keyword fuzzy search over encrypted data in the cloud," in IEEEINFOCOM, 2014.

[2] S.Menaka and N.Radha "Text Classification using Keyword Extraction Technique" International Journal of Advanced Research in Computer Science and Software Engineering Research Paper Volume 3, Issue 12, December 2013 ISSN: 2277 128X

[3] S. Kamara and K. Lauter, "Cryptographic cloud storage," in Financial Cryptography and Data Security. Springer, 2010, pp. 136– 149.

[4] W. K. Wong, D. W.-l. Cheung, B. Kao, and N. Mamoulis, "Secure knn computation on encrypted databases", in Proceedings of the 2009 ACM SIGMOD International Conference on Management of data. ACM, 2009, pp. 139152

[5] N.Veenasri, P.Rajesh "EFFECTIVE CLOUD SEARCH OVER ENCRYPTED DATA" International Journal of Science, Technology & Management www.ijstm volume No.03, Issue No. 12, December 2014 ISSN.

[6] P.A. Cabarcos, F.A. Mendoza, R.S. Guerrero, A.M. Lopez, and D. DiazSanchez, "SuSSo: seamless and ubiquitous single sign-on for cloud service continuity across devices," IEEE Trans. Consumer Electron.,vol.58, no. 4, pp. 1425-1433, 2012.

[7] W. Sun, B. Wang, N. Cao, M. Li, W. Lou, Y. T. Hou, and H. Li, "Privacy-preserving multi-keyword text search in the cloud supporting similarity-based ranking," in Proceedings of the 8th ACM SIGSAC symposium on Information, computer and communications security. ACMs, 2013, pp. 71–82.

[8] L. M. Vaquero, L. Rodero-Merino, J. Caceres, and M. Lindner, "A break in the clouds: towards a cloud definition," ACM SIGCOMM Comput. Commun. Rev., vol. 39, no. 1, pp. 50-55, 2009.

[9] S. G. Lee, D. Lee, and S. Lee, "Personalized DTV program recommendation system under a cloud computing environment," IEEE Trans. Consumer Electron., vol. 56, no. 2, pp. 1034-1042, 2010.

[10] C. Gentry, "A fully homomorphic encryption scheme," Ph.D. dissertation, Stanford University, 2009.

[11] N. Cao, C. Wang, M. Li, K. Ren, and W. Lou, "Privacypreserving multi-keyword ranked search over encrypted cloud data," in IEEE INFOCOM, April 2011, pp. 829– 837.

[12] D. Boneh, G. Crescenzo, R. Ostrovsky and G. Persiano, "Public key encryption with keyword search," in Advance in Cryptology Eurocrypt 2004.Springer,2004 pp. 506–522.

[13] C. Wang, N. Cao, K. Ren, and W. Lou, "Enabling secure and efficient ranked keyword search over outsourced cloud data," Parallel and Distributed Systems, IEEE Transactions on, vol. 23, no. 8, pp. 1467–1479, 2012.

[14] O. Goldreich and R. Ostrovsky, "Software protection and simulation on oblivious rams," Journal of the ACM (JACM), vol. 43, no. 3, pp. 431–473, 1996.

[15] D. Diaz-Sanchez, F. Almenarez, A. Marin, D. Proserpio, and P.A. Cabarcos, "Media cloud: an open cloud computing middleware for content management,"IEEE Trans. Consumer Electron., vol. 57, no. 2, pp. 970-978, 2011.