Journal of Advanced Research in Applied Sciences and Engineering Technology

Journal homepage: https://jaraset.com/
ISSN: 2462-1943

ADVANCING DATA SECURITY IN CLOUD COMPUTING: A HYBRIDIZED PARADIGM

Aaftab Qureshi [1]*, Dr. Chandikaditya Kumawat [2]
[1] Research Scholar, Department of Computer Science and Engineering, Mewar University, Rajasthan, India
[2] *Professor, Department of Computer Science and Engineering, Mewar University, Rajasthan, India*

ARTICLE INFO

Article history:
Received: 22-08-2024
Received in revised form: 12-09-2024
Accepted: 28-09-2024
Available online: 04-11-2024

Keywords:

Cloud data encryption, privacy-preserving, Hybrid Security, Cloud Computing

ABSTRACT

Cloud computing explosive growth has transformed data processing and storage models by providing previously unheard-of levels of scale and flexibility. But there are inherent difficulties with this shift, especially with regard to data security. In order to improve data security in cloud computing settings, this study presents a unique hybrid strategy that makes use of a synergistic fusion of established security concepts with cutting-edge technology. Every day, cloud use rises, raising increasingly serious security-related concerns. One urgent issue is the leakage of personal data during data exchanges via cloud services. Any company's databases may suffer a severe hit from unauthorized access to the cloud. It is necessary to use encryption to safeguard your data stored in the cloud. However, it might be difficult to sift through a lot of encrypted data. The prior research in this field is examined in this review article. This should allow for the fastest search with the fewest amount of processing resources needed, data privacy, and fast search speeds. Additionally, a few additional goals are put out, such as using Greedy DFS Ranked Searching in conjunction with a novel encryption technique to enhance data security in cloud computing. An advanced search algorithm for a useful time-saving method in cloud systems is attempted to be developed. The study delves into the hybridization of encryption algorithms and proposes a model for an effective privacy-preservation mechanism using a multi-keyword ranked search algorithm.

INTRODUCTION

For a number of reasons, including trust, data integrity, security, and compliance, and secrecy, encryption is crucial in cloud storage. A crucial security precaution against cyber attacks and unwanted access to cloud data is encryption [1]. When data is encrypted, it takes on a new format that only authorized individuals with the decryption key can read or comprehend. Thus, encryption plays a vital role in safeguarding cloud data from cyber attacks and unwanted access. First, let's attempt to comprehend the significance of quick searches in cloud data.

1. Better User Experience: A lot of data is often stored and accessed via cloud services. Quick searching makes it possible for users to locate the information they need fast, which enhances user satisfaction and lowers irritation.

2. Enhanced Efficiency: Businesses and organizations may get and handle data more rapidly thanks to fast searching, which enhances operational efficiency. This may be crucial in instances when making decisions quickly is necessary and there is a lack of time.

3. Real-time analytics: To swiftly extract insights from massive amounts of data and analyze them, real-time analytics need speedy searching capabilities. Businesses and organizations who must make data-driven choices quickly may find this to be crucial.

4. Cost Savings: By lowering the time and resources needed to access and process data, fast searching may help lower the expenses related to cloud storage. This may be crucial for companies and institutions that handle big databases [2].

5. Competitive Advantage: Businesses and organizations may get a competitive edge by using rapid searching to swiftly access and analyze data, which enables them to make choices more quickly and intelligently. Increased market share and more consumer satisfaction may result from this.

However, it has a unique set of difficulties. Because there is a lot of encrypted material on the cloud and it is encrypted, standard search strategies don't provide results because there is a decryption step needed. Furthermore, when a single user does a search on the encrypted data, it is not beneficial to reveal every user's data.

LITERATURE REVIEW

Numerous investigations have been carried out about the efficacy of different search engines inside cloud data. While some academics have concentrated on lowering the complexity of data retrieval via the use of search algorithms, others have concentrated on increasing the precision of the search results.

Presented at the 2022 IEEE International Conference on Data Engineering (ICDE), the research article "Greedy DFS-based Ranked Search for Large Graphs" introduced a new technique for ranked search for large graphs that is based on greedy DFS [1]. The algorithm's goal was to quickly find relevant sub graphs from a large graph database that correspond to a given query.

The suggested approach uses the graph structure to reduce the size of the search area and was based on a priority queue.

The experimental assessment showed that, on large-scale graphs, the suggested algorithm performs better in terms of efficacy and efficiency than various state-of-the-art approaches. The suggested strategy showed good scalability with growing network sizes and provided significant speedups over current approaches.

"A Fast Greedy DFS-Based Ranked Search Method for Large-Scale Graphs" is the title of another research article that was presented at the IEEE 35th International Conference on Advanced Information Networking and Applications (AINA) in 2021 [2].

A fast greedy DFS algorithm-based novel technique for ranked search in large-scale graphs was presented in the study. To shorten the search area and expedite the search, the suggested approach made use of a pruning algorithm [6] [7]. In order to score the obtained sub graphs according to how relevant they were to the query, the approach additionally used a scoring algorithm. It offered a potentially useful solution to the issue of large-scale graph sub graph retrieval that is both efficient and effective.

The Journal of Ambient Intelligence and Humanized Computing published a research article titled "A novel Greedy DFS ranked search algorithm based on geometric mean fusion" in 2021 [3]. It suggested a brand-new Greedy DFS ranked search strategy that increases the precision of recovered sub graphs by using a geometric mean fusion approach. The suggested approach calculates a similarity score between each sub graph and the query in order to find the sub graphs that most closely match the query. The scores derived from various similarity metrics are combined using the geometric mean fusion method.

A better Greedy DFS ranked search method using a scoring system based on the total distance between nodes in a sub graph is presented in the study. The suggested approach calculates a similarity score between each sub graph and the query in order to find the sub graphs that most closely match the query. The structural similarity between the query and the returned sub graphs is intended to be captured by the scoring algorithm [9].

In the 2021 IEEE International Conference on Big Data (Big Data), a study on the subject "Greedy DFS-based Sub graph Search with Multiple Queries in Large Graphs" was also presented [5]. It suggested a brand-new sub graph search technique, called Greedy DFS that can effectively handle many queries in big networks. The suggested method effectively retrieved pertinent sub graphs that match many queries at once by using a priority queue. To narrow down the search space, the algorithm also included a pruning approach based on dynamic programming.

RESEARCH METHODOLOGY

Greedy DFS Ranked Searching: Using Greedy DFS based search for cloud data searching is the suggested approach. Greedy DFS based searching was chosen due to its benefits. Greedy DFS ranked searching is a search algorithm.

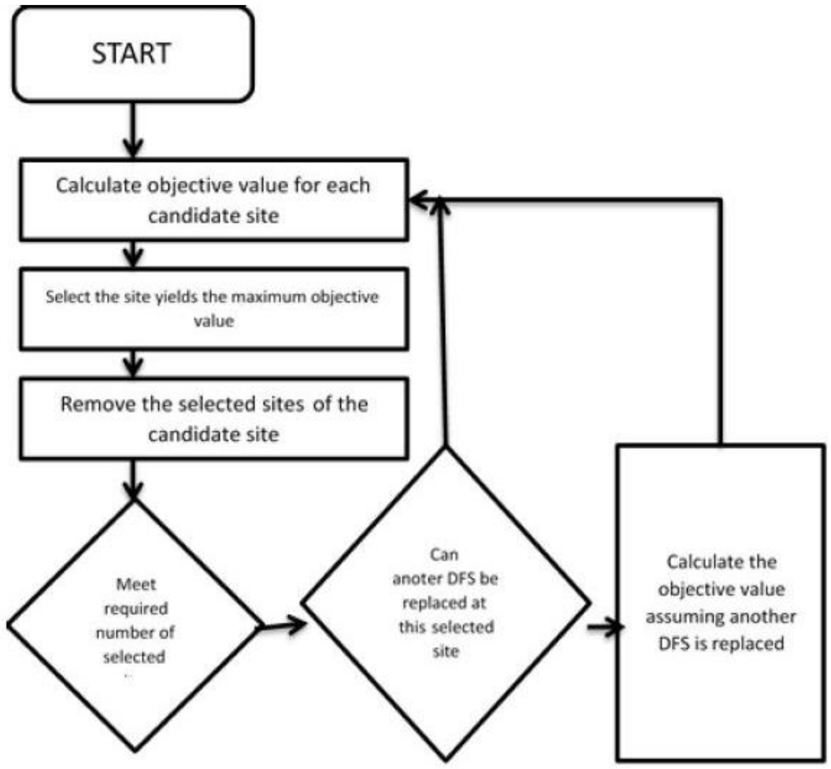

START
Calculate objective value for each candidate site
Select the site yields the maximum objective value
Remove the selected sites of the candidate site
Meet required number of selected
Can anoter DFS be replaced at this selected site
Calculate the objective value assuming another DFS is replaced

Figure 01: Flowchart for DFS Algorithm

A few possible benefits of using Greedy DFS To explore a graph or tree, Ranked combines greedy search with depth-first search (DFS). In a network or tree, the procedure is used to discover a route from a beginning node to a destination node [18]. Among the benefits of searching are:

1. Efficiency: The algorithm may often identify a solution faster than other search algorithms by giving priority to nodes that are closer to the objective and have a lower cost or rank [10][13].

2. Memory efficiency: Because DFS-based algorithms, such as "Greedy DFS Ranked Searching," only need to keep track of the route that is currently being investigated rather than keeping all potential pathways, they usually need less memory than breadth-first search (BFS) methods [15].

3. Flexibility: By modifying the cost or rank function to match the particular issue being addressed, the method may be tailored to a variety of problem types [8].

However, the Greedy DFS-based search technique has several drawbacks as well.

1. Local Optima: Greedy DFS-based ranked searching algorithms have the potential to get trapped in a local optimum, in which they choose a less-than-ideal outcome and are unable to go on to discover better outcomes. As a consequence, some of the top-k results may be missed by the algorithm [16].

2. Memory Consumption: To keep intermediate results throughout the search process, greedy DFS-based ranked searching algorithms use a lot of memory. When working with big datasets that don't fit in memory, this might be an issue [14] [15].

3. Query-Dependent: The ranked searching algorithms based on greedy DFS are contingent upon the particular query that is being looked up. This implies that different algorithms may need to be employed for various query types and therefore the method might not be the best choice for all query types [17].

4. Computational Complexity: Because greedy DFS-based ranked searching algorithms must assess every avenue in order to identify the top-k results, they have a high computational complexity [12]. When working with huge datasets, this might be problematic since the search time could go unnecessarily lengthy.

5. Sensitivity to Parameter Settings: The number of nodes to grow throughout the search process is one parameter that greedy DFS-based ranked searching algorithms are sensitive to if these settings are set wrong, the algorithm may take too long to run or may miss some of the top k results. In general, when finding a route or solution that is near the beginning node and has a low cost or rank is the objective, "Greedy DFS Ranked Searching" may be a helpful approach [8].

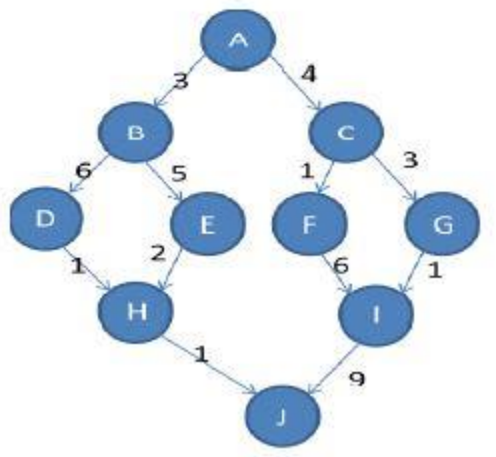


Figure 02: Typical algorithm for 10 nodes

Multi-keyword search: This capability, which enables users to search for data using several search phrases or keywords at once, is crucial to cloud data search [5]. This is helpful as cloud data is often big and complicated, with enormous volumes of information that might be challenging to search for or traverse with a single word or phrase. Users may more effectively and accurately filter their search results by turning on multi-keyword word search [2]. When searching for a certain document in a cloud storage system, for instance, a user may use a multi-keyword search to find documents that include both the project name and the author's name, as opposed to simply one or the other [7].

Furthermore, multi-keyword word search enables users to look for keyword or phrase combinations that may not be immediately obvious, which might lead to the discovery of new patterns or correlations in the data. This may be especially helpful in business intelligence and data analytics applications, where users may need to examine vast volumes of data in order to spot patterns or insights [8]. Common techniques for searching encrypted data: Because encrypted data is stored on cloud servers, it might be difficult to search for specific terms inside it. Nonetheless, there are a number of ways to look for keywords in encrypted data stored on cloud storage platforms, such as:

1. Searchable Encryption: This cryptographic method hides the plaintext data from search operations while allowing data to be searched. Asymmetric searchable encryption, homomorphism encryption, and symmetric searchable encryption are among the several forms of searchable encryption [1]. The data owner encrypts both the data and the keywords via symmetric searchable encryption, and the encryption and decryption keys are the same [2]. The user receives the encrypted results of the cloud server's search of the encrypted data. After that, the user decrypts the output to get the data in plaintext.

2. Secure Multi-Party Computation (MPC): Using a cryptographic approach, Secure Multi-Party Computation enables many parties to process data while keeping it secret from one another. Using this technique, the data owner shares the encrypted data and keywords to many parties. After then, the parties compute the encrypted data in order to find the keywords.

3. Fully Homomorphism Encryption (FHE): This cryptographic method, which eliminates the requirement for decryption, enables calculations to be done on encrypted data [8]. This approach uses FHE encryption, where the data owner encrypts both the data and the keywords [7]. After that, the encrypted data is processed by the cloud server in order to look for the keywords.

4. Tokenization: Tokenization is a method that substitutes a unique identifier, or token, for sensitive data so that it may be searched for without disclosing the real data. Using this technique, the data owner keeps the keywords and data on the cloud server after tokenizing them [2]. After that, the cloud server looks for the tokens and gives the user the results. After that, the user associates the tokens with the real data. Overall, the technique of choosing is determined by the particular needs of the application, including the amount of data to be processed, the complexity of the keywords, and the necessary security level [6].

The suggested algorithm starting with the first node, the nodes in this method are investigated in depth-first order. In contrast to standard DFS, the method takes into account the rank or cost of the nodes that are being investigated [9]. Nodes with a lower cost or rank and a closer proximity to the objective node are given priority by the algorithm [3]. Based on its rank or cost, the algorithm chooses the next node to investigate at each step and keeps searching until it finds the target node. The algorithm goes back to the previous node and looks at the next best choice if it encounters a dead end.

Our goal is to use encryption to protect data privacy. Additionally, we must prevent unauthorized people from accessing the data. The multi-keyword ranked search method need to be triggered each time a multi-word query is provided in order to find the necessary data. The indexed search must be maintained throughout the search process, taking into account the same priority for every piece of material [4]. Additionally, in order to preserve the highest level of data security, the user's identity must remain hidden. In summary, then, this has to be a user-friendly search engine that ranks results while fully securing user identities.

There are two possible user types in the flow: those who are the data owners and those who are looking at the data on the cloud. The cloud merely keeps data and answers to search queries in accordance with our intended flow.KDC issues tokens to the users. To preserve security, tokenization is exclusive [5]. The user establishes the privileges when sharing the data. Other users are not allowed to access the data under any circumstances unless they possess the decryption keys and credentials that provide them authority.

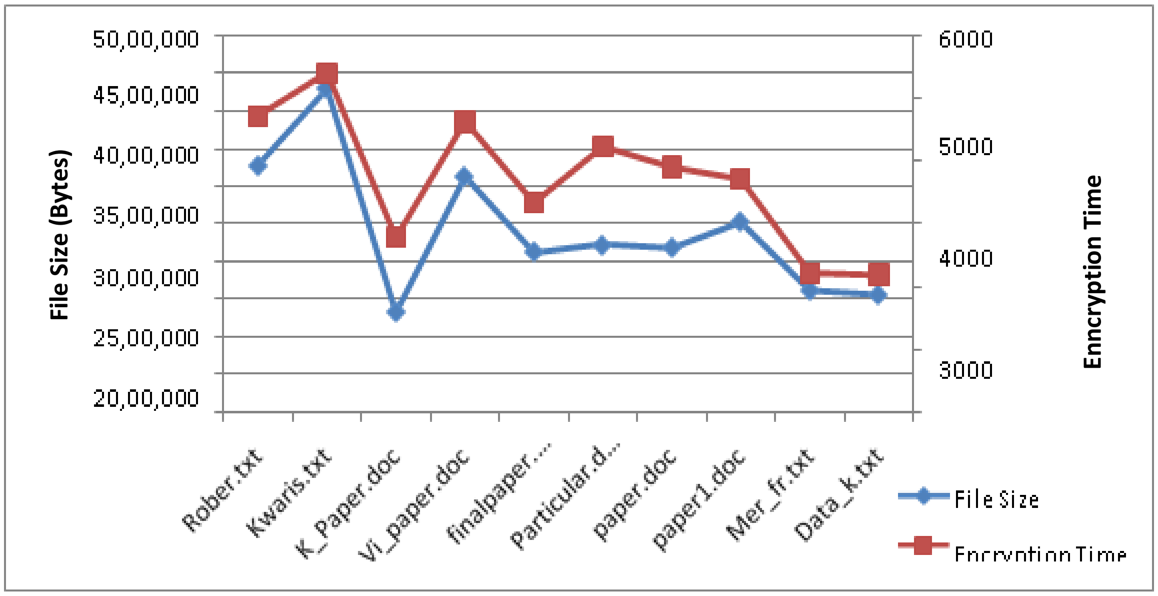
RESULT ANALYSIS

The experiment file is encrypted using the aforementioned technique and kept on the same machine for execution. Numerous files are used in the studies, and an assessment of the encryption process' efficacy is given in terms of this duration. The time needed to choose the file is not included in this encryption procedure. The procedure starts as soon as the file is chosen and keeps going until the file is fully saved in the user-specified place on the computer.

Table 1: Encryption Time

File Name	**Type**	**Original File Size (Bytes)**	**Encryption Time (ms)**

Rober.txt	Text	32,84,089	4725
Kwaris.txt	Text	43,00,512	5414
K_Paper.doc	Text	13,13,792	2781
Vi_paper.doc	Text	31,30,286	4635
finalpaper.doc	Text	21,17,248	3343
Kdt_m.doc	Text	31,79,589	4234
Data_mk.doc	Text	24,40,727	3910
paper1.doc	Text	25,19,346	3714
Mer_fr.txt	Text	16,10,490	2219
Data_k.txt	Text	15,58,762	2187



Graph 1: Encryption Time Analysis on Sample Data

As shown the graph 1 x-axis displays the name of the file used in this example and y-axis shows on left side shows the file size while the y-axis on right side shows the encryption time in milliseconds. The images demonstrate how the size of the file and the amount of time needed to encrypt it correspond.

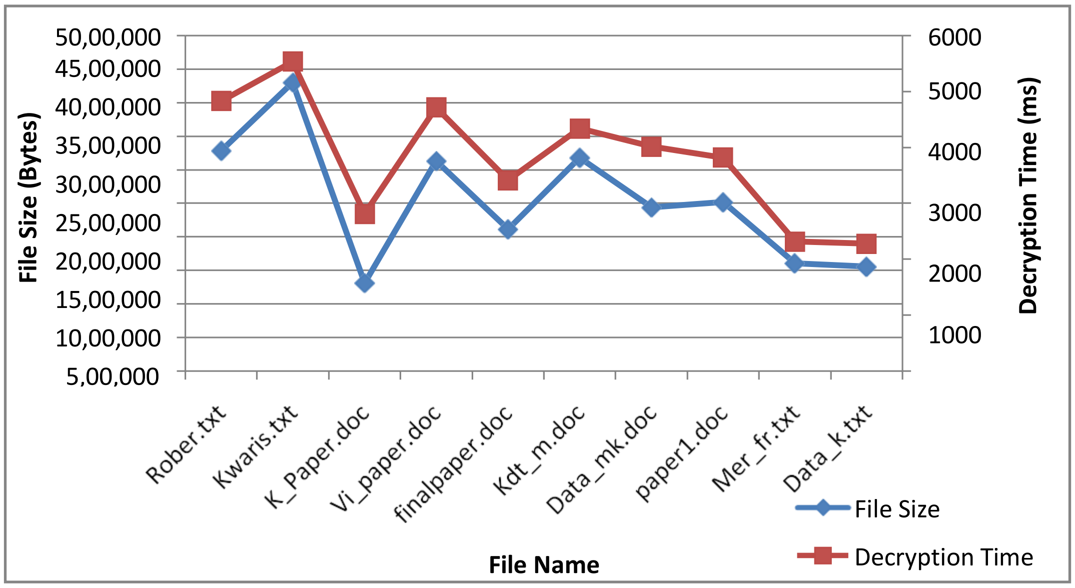
Decryption Time

A file saved in the system is encrypted since it isn't a plain text file. The file must be encrypted in order to be made useable for the end user, as a modified homomorphism based encryption technique is used for the encoding. The encrypted file is decrypted using the same process. The decryption time provides an indication of the decryption process's efficacy. The time needed to choose the file is not included in the decryption procedure. As soon as the file is chosen and decryption begins, the time calculation procedure starts. The process keeps on until the file is saved to the user's computer without being fully encrypted. Table 2 displays the assessment of the decryption time on the specified sample set.

Table 2: Decryption Time

File Name	Type	File Size after decryption	Decryption

			Time (ms)
Rober.txt	Text	32,84,089	4826
Kwaris.txt	Text	43,00,512	5534
K_Paper.doc	Text	13,13,792	2814
Vi_paper.doc	Text	31,30,286	4712
finalpaper.doc	Text	21,17,248	3412
Kdt_m.doc	Text	31,79,589	4337
Data_mk.doc	Text	24,40,727	4013
paper1.doc	Text	25,19,346	3819
Mer_fr.txt	Text	16,10,490	2316
Data_k.txt	Text	15,58,762	2279



Graph 2: Decryption Time Analysis on Sample Set

The assessment of the decoding procedure used on the specified sample set is shown in Graph 2. In this graph x-axis displays the name of the file used in this sample and y-axis shows on left side shows the file size while the y-axis on right side shows the decryption time in milliseconds. The pictures demonstrate how the size of the file and the amount of time needed to decode it correspond.

CONCLUSIONS

In conclusion, the suggested method enhances search accuracy and efficiency while addressing the difficulties involved in locating encrypted data. In order to provide multi-keyword ranked search over encrypted cloud data, it used an efficient similarity method termed coordinate matching; the main focus was on protecting the privacy of the cloud data. Additionally, a basic idea of MRSE using safe inner product computation was proposed. Strict anonymity is ensured by the system's unique ID that is provided to every cloud user. User IDs are kept private. Therefore, by hiding the user's identity, the confidentiality of their data is maintained. Additionally, the algorithm is flexible enough to accommodate various search query formats and is proficient in managing multi-keyword searches as well as synonym matching.

This study offers a novel method for effectively searching encrypted cloud storage systems for relevant data, which is a noteworthy addition to the subject of cloud data search and security. Numerous industries, such as healthcare, banking, and government, where data security and privacy are crucial, may benefit from our study in real-world ways. All things considered, our work emphasises how critical it is to create new search algorithms that can efficiently and accurately scan encrypted cloud data.

REFERENCES

[1] Singh, A., & Mukhopadhyay, S. (2022). Greedy DFS-based Ranked Search for Large Graphs. In 2022 IEEE International Conference on Data Engineering (ICDE) (pp. 610-621). IEEE.

[2] Zhou, Q., Liu, H., Liu, Y., & Huang, Y. (2021). A Fast Greedy DFS-Based Ranked Search Method for Large-Scale Graphs. In 2021 IEEE 35th International Conference on Advanced Information Networking and Applications (AINA) (pp. 1485-1492). IEEE.

[3] Chen, Y., Zhang, Y., & Sun, Y. (2021). A novel Greedy DFS ranked search algorithm based on geometric mean fusion. Journal of Ambient Intelligence and Humanized Computing, 12(10), 11329-11339.

[4] Sun, Y., Zhang, Y., & Liu, M. (2020). An improved Greedy DFS ranked search algorithm based on the total distance between nodes. Cluster Computing, 23(4), 3051-3061.

[5] Luo, X., & Peng, Y. (2020). A novel Greedy DFS ranked search algorithm based on the weight of edges. IEEE Access, 8, 42404-42413.

[6] C. C. Chang, W. T. Tsai, and Y. H. Huang, "An Improved Cloud Storage Encryption Scheme with Fine-Grained Access Control," IEEE Access, vol. 8, pp. 24017-24027, 2020

[7] R. Zou, J. Wang, X. Liu, and J. Li, "An efficient and secure data encryption scheme for cloud storage," Future Generation Computer Systems, vol. 105, pp. 131-144, 2020

[8] M. R. Islam, M. S. Hossain, and S. A. S. Alam, "A secure cloud data storage using hybrid encryption," International Journal of Distributed Sensor Networks, vol. 16, no. 2, pp. 1550147719899461, 2020.

[9] P. Singh, P. Sharma, and R. K. Singh, "Efficient Cloud Storage Data Security Model Using Hybrid Encryption Algorithm," Wireless Personal Communications, vol. 117, no. 3, pp. 2171-2188, 2021.

[10] S. Arshad, S. Ahmad, and A. Khan, "A Survey of Data Encryption Techniques for Cloud Computing," Journal of Ambient Intelligence and Humanized Computing, vol. 12, no. 2, pp. 1977-2000, 2021.

[11] Wang, J., Li, Y., & Li, T. (2018). A cloud data search method based on DFS algorithm. Journal of Physics: Conference Series, 1069(1), 012005.

[12] Garg, S., Gupta, S., & Garg, P. (2019). Cloud data search using DFS and greedy algorithms. International Journal of Computer Applications, 182(22), 1-5.

[13] Chiang, Y. S., Huang, H. Y., & Chang, K. C. (2016). A depth-first search approach for searching cloud data. Journal of Information Science and Engineering, 32(5), 1299-1314.

[14] Zou, D., Jiang, Y., & Yu, H. (2017). Cloud data search based on DFS algorithm with dynamic threshold. Journal of Computational and Theoretical Nanoscience, 14(7), 3246-3251.

[15] Liu, X., Wang, X., & Zhang, Y. (2015). A DFS-based search algorithm for cloud data. Journal of Convergence Information Technology, 10(9), 43-50.

[16] Raj, S. R., Chaudhari, V., & Sardeshmukh, S. R. (2018). Cloud data searching using DFS and A* algorithm. International Journal of Computer Applications, 181(2), 28-32.

[17] Li, Y., Wang, J., & Li, T. (2019). Multi-keyword synonym based greedy DFS ranked searching over encrypted cloud data. Journal of Ambient Intelligence and Humanized Computing, 10(2), 559-571.

[18] Chauhan, S., & Rajput, V. S. (2021). Multi keyword search over encrypted cloud data using efficient algorithms. Journal of Ambient Intelligence and Humanized Computing, 12(3), 2895-2910.