Enhanced Depth-First Search Algorithm for Improving the Efficiency of Route Construction in Data Center Networks

Ayam Mohsen Abbass
Computer Engineering Department, College of Engineering, Mustansiriyah University, Baghdad, Iraq
ayammohsen@uomustansiriyah.edu.iq (corresponding author)

Ahmed Yousif Falih Saedi
Computer Engineering Department, College of Engineering, Mustansiriyah University, Baghdad, Iraq
ahmed.yousif@uomustansiriyah.edu.iq

Jaafar Qassim Kadhim
Electrical Engineering Department, College of Engineering, Mustansiriyah University, Baghdad, Iraq
jaafar80@uomustansiriyah.edu.iq

Received: 21 December 2024 | Revised: 8 February 2025 and 22 February 2025 | Accepted: 24 February 2025

Licensed under a CC-BY 4.0 license | Copyright (c) by the authors | DOI: https://doi.org/10.48084/etasr.10005

ABSTRACT

Data communication is a critical part of Data Center Networks (DCNs), ensuring the efficient and reliable transfer of data between nodes. Using the shortest path for data forwarding is a common approach in DCNs because it helps minimize latency. This method can present challenges, commonly related to network congestion and increased vulnerability to node failures. Given the inherent self-similarity and multipath routing characteristics of the fat-tree topology, the work at hand proposes an enhanced search method that aims to improve the efficiency of the Depth-First Search (DFS) method. The proposed algorithm is compared with the original algorithm on the typical network architecture found in data centers. The results highlight the advantages of the proposed enhanced DFS method, demonstrating its improved scalability and effectiveness in minimizing latency. The proposed method consistently reduces energy consumption under various network load conditions.

Keywords-data center network; fat-tree topology; depth-first search; uninformed search algorithm

I. INTRODUCTION

The evolution of data centers is driven primarily by the growth of Internet of Things (IoT) environments [1]. IoT technology and its rapid growth are expected to play an increasing important role in the future [2]. Cloud computing has transformed the shape of the IT industry with its scalability, flexibility, and cost-effectiveness [3, 4], while supporting the development of emerging technologies such as Artificial Intelligence (AI) and IoT [5], which may be distinct, but can be integrated to enable each other's functions [6].

Data communication is a critical part of DCNs and the fat-tree network architecture is widely recognized as the dominant choice. Fat-tree networks can form a Data Center Network (DCN) architecture that provides a scalable and efficient solution for transporting high-bandwidth data [7]. The architectural design incorporates a multi-tiered hierarchical topology with multiple leaf nodes and intermediate switches, allowing for a high degree of adaptability and expandability [8]. The fat-tree network architecture has tree-shaped branches and no blocking, as in Clos networks. In a fat-tree network, the core layer is at the top, followed by the aggregation layer, switches at the edge of the network, and servers at the base layer. The edge switches and the aggregation layer form pods that connect to the base layer, which contains switches and servers [9]. The branches are data links of varying thickness (bandwidth) that are used efficiently with their underlying technology being considered [10]. In addition, the edge switches act as the connection point for the servers, commonly referred to as hosts. Although fat-tree networks are widely used in DCNs, finding the shortest path for data transfer can present some challenges [11]. Complex topology, load balancing issues, fault recovery procedures, and configuration overhead can all affect their reliability and efficiency [12]. These challenges can lead to suboptimal routing decisions, increased latency, and reduced network efficiency [13].

II. LITERATURE REVIEW

Various routing algorithms have been proposed for DCNs, focusing on adaptive routing, reliability, low response time, and fault tolerance. These requirements are best met by the widely used multipath routing schemes [14-17]. Today's large-scale networks and data centers have a hierarchical structure consisting of routing and switching components with a large number of alternative paths. The use of routing protocols automates the creation of routing tables and enables the discovery of new routes in response to network changes, such as network failures or new links and routers [18]. Tharun and Kottilingam [13] proposed a very effective framework, the Dynamic Cluster-topology with a triangle model and dynamic sub topology Load Balancing (DCLB). This framework aims to optimize the scheduling of network flows in fat-free DCNs by dynamically incorporating additional IP header route information. It reduces memory usage, selects the least-cost path to optimize routing, and effectively manages traffic load balancing. Kulakov et al. [19] proposed and justified the organization of DCNs with fat-tree topology based on software configurable network technology using an improved deep search algorithm. The algorithm reduces the construction time of such networks by taking into account the self-similarity of the DCN topology, thereby improving the efficiency of large-scale DCNs. In addition, it allows the optimization of paths based on specified metrics during the construction process. Satotani and Takahashi [20] used a Depth-First Search (DFS) algorithm to find a Generalized Moore Graph (GMG) of a given degree and order. They found that the algorithm worked very well for small graphs, but was time consuming for large degree and order values. Liu et al. [21] proposed an exceptional set of rules for hybrid DCNs to properly schedule the circuit switch so that it handles most of the traffic demand, leaving little to the slower packet switch. They found that the proposed Best First Fit (BFF) method with partial reconfigurability significantly outperforms Eclipse with much lower computational complexity.

Authors in [22] proposed the fitted fat-tree architecture, a flexible DCN architecture whose topology and performance can be easily adjusted according to the traffic demand. The proposed architecture provided the same performance as the original fat-tree, but with significantly reduced computational cost. Isaia and Guan [23] used Mininet applications with an advanced fat-tree topology to overcome traffic bottlenecks in the network by assigning weights to the nodes and the communication links according to their capabilities, resulting in reduced packet loss and jitter. Biswas and Hussain [24] introduced a novel design that reduces the network size by removing the highest switch level while maintaining the scalability of MIN fat-tree algorithm. The proposed network architecture significantly reduces the delay and the number of switches. Their experiments showed that approximately 33.33% of the switches were reduced for 8 clients and 14.28% for 128 clients. Further research can determine if the network width can also be reduced while maintaining its scalability.

This study proposes an enhanced search method for a typical DCN to improve the efficiency of DFS and compares it with the original algorithm. The results indicate that the proposed algorithm reduces latency and energy consumption.

III. MATERIALS AND METHODS

This section consists of two parts, the first part describes the uninformed search algorithms in DCN and the second part describes the proposed enhanced algorithm based on DFS.

A. *Overview of Uninformed Search Algorithms in Data Center Networks*

Uninformed search is a state-space solution strategy that does not use any additional state information other than that provided in the problem definition and is capable of developing successors and distinguishing the target state from the non-target state. Uninformed and informed search algorithms are shown in Figure 1. DFS, Breadth-First Search (BFS), and Uniform Cost Search (UCS) are three different algorithms used to search and traverse graph structures. A comparison of these algorithms based on different aspects is given in Table I [25, 26].

BFS is a state-space decision-making strategy that first explores the root node, then all of the descendants of the root node, then the descendants of those descendants, and so on. All nodes at a given depth in the search tree are explored before any other nodes are expanded, and if all have the same cost BFS is optimal [27, 28]. BFS uses a First-In-First-Out (FIFO) queue, starting at the root node, and the current node is retrieved from the top of the queue at each iteration. The search stops when the current node is the destination. BFS is a level-wise traversal technique that explores nodes layer by layer, ensuring that all nodes are visited in the shortest possible number of steps [29].

UCS is a generalization of the BFS algorithm that takes cost into account [30]. In cost-based search, nodes are explored from most costly to least costly depending on their distance from the root node. The nodes are stored in a priority queue, and at each algorithm step, the lowest-cost node is explored [31]. This algorithm is optimal for finding the shortest path when edge costs are different and non-negative. When the costs are equal, the search is identical to the BFS. However, the search procedure can enter an infinite loop if it explores a node that has a zero-cost action that again points to the same state. Thus, it is possible to guarantee the completeness and optimality of the search if the costs of all actions are strictly positive [31].

DFS is an algorithm that follows the "go deep, head first" concept. DFS analyzes the first path from the current node in the list, then its second path, and so on. Explored nodes are removed from the path so the search continues from the next deepest node with unexplored paths. The DFS method can be implemented using a Last-In-First-Out (LIFO) stack or a recursive function [32, 33].

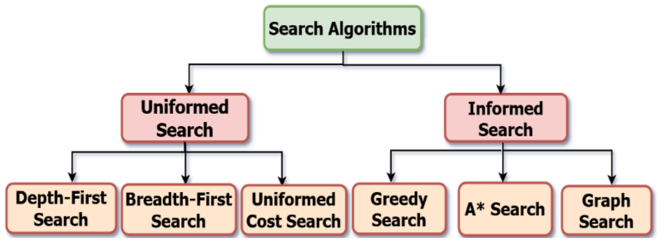

Fig. 1. Uninformed and informed search algorithms.

TABLE I. UNINFORMED SEARCH ALGORITHMS

Metrics	DFS	BFS	UCS
Searching strategy	Explorer nodes along each branch	Visit all the vertices of a graph	Visit vertices in order of increasing cost
Memory complexity	Less memory compared to BFS, UCS	Less memory compared to UCS	More memory than DFS, BFS
Time complexity	O(n + m), where n = vertices number and m = edges number	O(n + m), where n = vertices number and m = edges number	O(E log V), where E = edges number, V = vertices number
Optimality	shortest route assignment in an unweighted graph	Determine the minimum distance between two nodes in a directed graph with weights	Does not guarantee optimality
Use cases	Suited for solving mazes and generating random spanning trees	Suited for finding shortest paths in unweighted graphs	Suited for finding the shortest path in a weighted graph and optimal solutions in decision trees
Topology	Used on trees or mazes	Used on trees, mazes, and grids	Used on graphs where edge weights have a meaningful interpretation
Drawbacks	Takes an exponential amount of time in the worst case	Requires a lot of memory	It can be slow in the case of large graphs

B. Enhanced Algorithm based on a Depth-First Search

DFS is an iterative algorithm that uses the backtracking method. It defines a node as visited or unvisited. Assuming that no other nodes have been visited, it sends a packet to the destination node, flags it as visited, and examines the data of its immediate neighbors. The packet moves to the next forward node until it finds no forward nodes on the current path, and then moves backward on the same path to find a traversable node on another path. This algorithm visits all nodes in the specified path, identifies any unwanted nodes, traverses them all, and then selects the next path. The transition process between nodes can continue in a loop and go into an infinite loop. This can affect the search for the desired (target) node, and the algorithm may not be able to find it. Path selection can be achieved using one of three primary methods:

- The worst-fit method always allocates the most significant available bandwidth to the channel associated with the node in the hope that the remaining bandwidth will be helpful in serving a future request.
- The first-fit method is the simplest technique for selecting a path within any frequency range from among all frequencies that meet the requirements. This method accepts the request to assign nodes for the subsequent process by following the pointer as it moves through the network, tracing the route for all accessible nodes.
- The best-fit method is a technique that uses the channel bandwidth that effectively meets the requirements.

It is worth noting that when using the DFS method with multiple paths in a DCN, the worst-fit method is considered more effective than the other methods. DFS uses a centralized process to manage the bandwidth of each channel and the routing of all data packets within the DCN. The topology defines the two nodes to be connected as the sender and receiver nodes and the level of the two nodes, thus eliminating unnecessary transitions between levels.

The proposed enhanced DFS algorithm, shown in Figure 2, introduces a global variable that indicates the direction of the nodes in the network to speed up the transition between levels: If the variable is equal to 1, the selected transition path is up the tree. If the variable is equal to -1, the selected transition path is down the tree. If the sender and receiver nodes pass on a connectable level, the variable is equal to -1. This means that the traversal has reached the local limit and must go down the tree. If the traversal reaches a switch representing an edge node that isn't connected to the receiver, the next variable is set to 1 and the traversal returns to the previous level. A comparison of recent research with our proposed algorithm based on different aspects is given in Table II.

TABLE II. COMPARISON BETWEEN ENHANCED DFS ALGORITHM AND RECENT RESEARCH

Recent researches	Shortest path	Least cost	Fast tree exploration
[13]		✓	
[19]	✓	✓	
[20]			✓
[21]	✓	✓	
[22]		✓	
[23]	✓	✓	
Enhanced DFS	✓	✓	✓

IV. RESULTS AND DISCUSSION

The proposed method was evaluated using the OMNET++ version 5.6.2 emulator with MATLAB version R2016a. The traditional DCN architecture with a proposed modification to the topology was used to analyze and confirm the effectiveness of the proposed method in finding a set of paths. The simulated system is illustrated in Figure 3 and the results of the shortest path search are shown in Table III. The algorithm produces 4 paths between the vertices S0 and S4 and the average path length is (4+5+8+8)/4 = 6.25, since the lengths of the paths are 4, 5, 8, and 8, respectively [34].

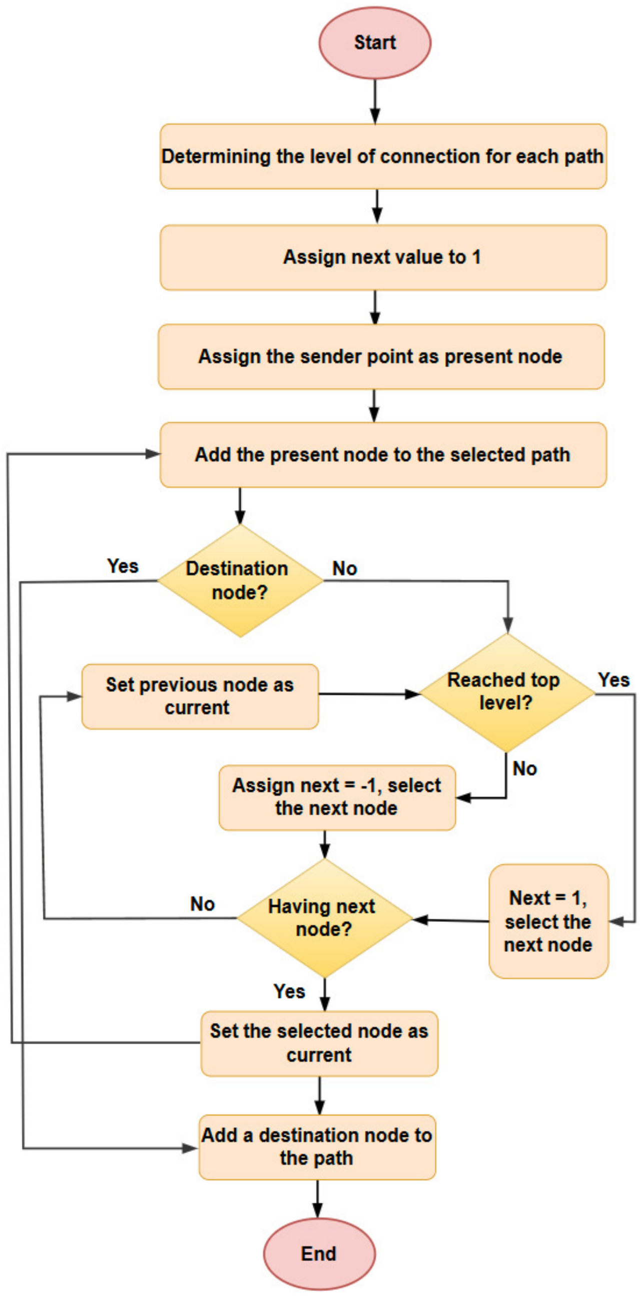
Start
Determining the level of connection for each path
Assign next value to 1
Assign the sender point as present node
Add the present node to the selected path
Destination node?
Yes
No
Set previous node as current
Reached top level?
Yes
No
Assign next = -1, select the next node
Next = 1, select the next node
Having next node?
No
Yes
Set the selected node as current
Add a destination node to the path
End

Fig. 2. Enhanced DFS algorithm.

TABLE III. PATHS BETWEEN S0 AND S4 VERTICES

Path number	Path direction	Length
Path 1	S0,N0,N6,N2,S4	4
Path 2	S0,N6,N8,N7,N2,S5	5
Path 3	S0,N0,N4,N1,N5,N3,N7,N2,S5	8
Path 4	S0,N0,N4,N9,N5,N3,N7,N2,S5	8

Figures 4 and 5 show the relationship between the number of paths and the number of DCN nodes for the basic and modified DFS algorithms, respectively. It can be observed that the number of paths increases significantly as the number of DCN nodes increases for the basic DFS. On the other hand, the increase is smaller for the modified DFS. This is because the basic DFS performs a sequential enumeration of all paths. In the modified DFS, the self-similar network topology is taken into account, allowing the optimal path to be selected each time, thus reducing the number of iterations. Figure 6 shows the relationship between the total number of DCN nodes and the number of iterations required by both DFS methods. As can be seen, the advantage of the proposed algorithm increases as the number of DCN nodes increases.

Calculating the network delay helps reduce response time, energy consumption, and improve network performance. The delay depends on the number of hops, the transmission delay between nodes and the network load. The delay increases gradually as the network load increases, especially at higher values, which is expressed by a logarithmic relationship. On the other hand, the energy consumption depends on the number of nodes and the network load in a common logarithmic relationship, because network systems start with a significant increase in consumption and then the consumption increase slows down with the increase of nodes and network load [35]. The average delay can be calculated using (1), taking into account the number of hops and the traffic load:

$$Average\ delay = N \times D \times \log(1 + NL) \qquad (1)$$

where N is the number of hops required to transfer data, D represents the time between two nodes to transfer data, and NL is the amount of data transferred through the network. Figure 7 illustrates a comparison of the average packet delays for four resource allocation techniques: modified DFS, best-it, worst-fit, and first-fit, under varying traffic loads. The modified DFS approach outperforms the others, maintaining consistently low delays regardless of traffic intensity. This reflects its strong ability to handle traffic efficiently and reduce congestion. Conversely, the worst-fit strategy exhibits the highest delays, especially under heavy traffic, indicating poor resource allocation. Best-fit and first-fit perform moderately well, but their delays increase steadily with increasing traffic load, indicating limited efficiency in high-demand scenarios. Overall, the results highlight the clear advantage of modified DFS, demonstrating its scalability and effectiveness in minimizing latency when traditional methods struggle with increased traffic. This underscores the importance of utilizing advanced optimization techniques to improve network performance. The values obtained from the experiments are shown in Table IV.

TABLE IV. AVERAGE DELAY OBTAINED FROM TRAFFIC LOAD EXPERIMENTS

Traffic load	First-fit	Best-fit	Worst-fit	Enhanced DFS
0.1	0.7500	0.7500	0.7500	0.0500
02	0.7667	0.7685	0.7685	0.0556
0.3	0.7971	0.8241	0.8024	0.0611
0.4	0.8366	0.9167	0.8462	0.0667
0.5	0.8833	1.0463	0.8981	0.0722
0.6	0.9363	1.2130	0.9570	0.0778
0.7	0.9949	1.4167	1.0222	0.0833
0.8	1.0587	1.6574	1.0930	0.0889
0.9	1.1271	1.9352	1.1690	0.0944
1.0	1.2000	2.2500	1.2500	0.1000

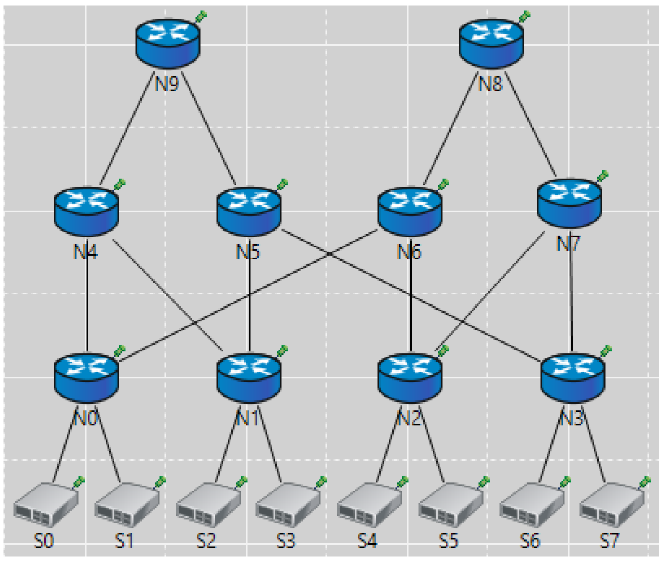

Fig. 3. Paths between DCN nodes.

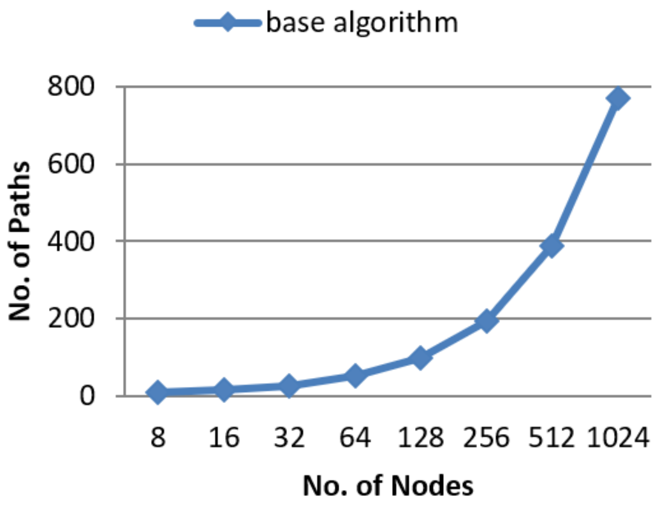

Fig. 4. Number of paths in base DFS algorithm.

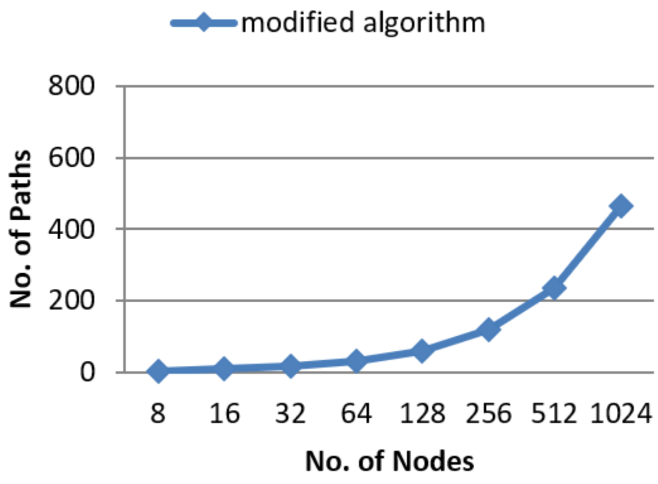

Fig. 5. Number of paths in modified DFS algorithm.

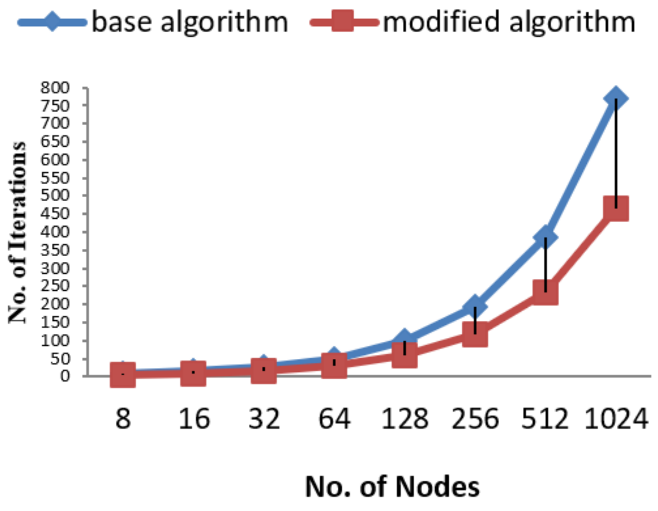

Fig. 6. Iterations dependency on the number of DCN nodes.

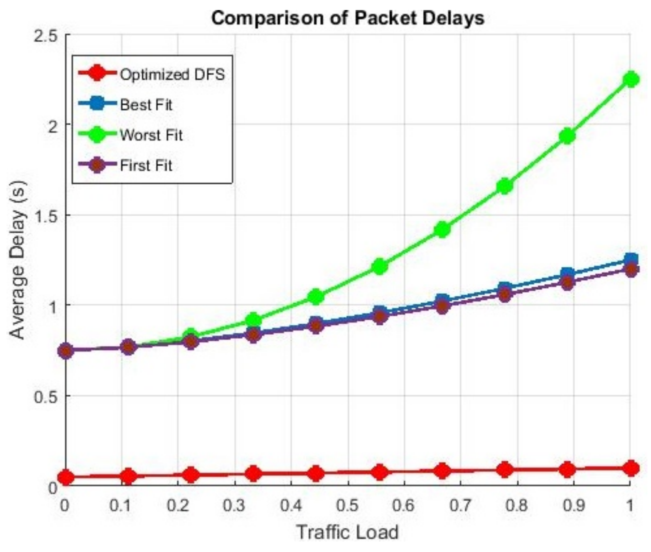

Fig. 7. Comparison of packet delay versus traffic load.

Determining energy consumption is critical because it helps in determine operating costs, emissions, and environmental footprint. The goal is to design a greener network by using advanced routing algorithms that reduce energy consumption, which is reflected in network performance, especially in power-sensitive networks, including Wi-Fi sensor networks. The energy consumption can be calculated using (2):

$$Energy\ consumption = EC \times \log(1 + NL \times N) \quad (2)$$

where EC is a factor that determines the efficiency of each algorithm and its energy consumption characteristics, NL is the amount of data transferred through the network, and N is the number of hops required to transfer the data. In Figure 8, the results show that the modified DFS consistently outperforms alternative allocation methods in terms of efficiency, maintaining stable and minimal energy consumption despite increasing network load. This efficiency is likely due to its superior resource allocation technique, which can reduce fragmentation and improve resource utilization. The stability of modified DFS makes it suitable for energy sensitive networks, such as IoT systems or data centers, where energy costs are a significant constraint, especially under high load conditions. Worst-fit exhibits exponential growth in energy consumption as load increases, indicating significant inefficiencies likely due to severe fragmentation, making it inappropriate for high-demand environments. Best-fit and first-fit are reasonable options, with energy consumption growing more slowly than worst-fit, although they still fall short of the efficiency and scalability of modified DFS. These techniques may be appropriate for networks with moderate loads but are less effective in conditions that require sophisticated power control. In summary, modified DFS is the most effective method for maintaining low energy consumption under varying network loads, whereas worst-fit is the least efficient. Best-fit and first-fit provide intermediate solutions, but exhibit reduced adaptability under high load conditions. The values obtained from the experiments are shown in Table V.

TABLE V. ENERGY CONSUMPTION OBTAINED FROM TRAFFIC LOAD EXPERIMENTS

Traffic load	**First-fit**	**Best-fit**	**Worst-fit**	**Enhanced DFS**
0.1	1.0474	1.0200	1.0025	1.0048
02	1.0671	1.0400	1.0100	1.0091
0.3	1.0822	1.0600	1.0225	1.0131
0.4	1.0949	1.0800	1.0400	1.0168
0.5	1.1061	1.1000	1.0625	1.0203
0.6	1.1162	1.1200	1.0900	1.0235
0.7	1.1255	1.1400	1.1255	1.0265
0.8	1.1342	1.1600	1.1600	1.0294
0.9	1.1423	1.1800	1.2025	1.0321
1.0	1.1500	1.2000	1.2500	1.0347

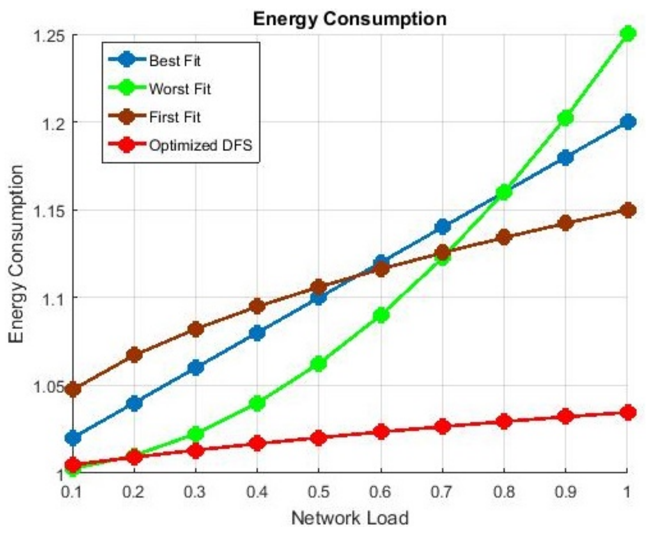

Fig. 8. Comparison of energy consumption versus traffic load.

V. CONCLUSION

This study proposes a modified Depth-First Search (DFS) algorithm for finding the optimal route configuration in Data Center Networks (DCNs). The proposed algorithm optimally exploits the characteristics of the fat-tree network topology and increases the efficiency of searching for multiple data transmission paths. The application of DFS within a fat-tree network topology facilitates the construction of multiple paths with improved load balancing. The results highlight the distinct advantages of the modified DFS, demonstrating its scalability and effectiveness in minimizing latency, whereas conventional approaches struggle under high traffic load conditions. This underscores the need for advanced optimization techniques to improve network performance. The proposed modified DFS is the most effective strategy for keeping power consumption low under varying network loads, while worst-fit is the least effective. Best-fit and first-fit perform moderately well, although they exhibit reduced adaptability in high load scenarios. Future research may include the exploration of dynamic topologies, where the network adapts primarily based on actual usage, rather than using a static topology.

ACKNOWLEDGMENT

The authors would like to thank Mustansiriyah University, Baghdad, Iraq for the support in the present work.

REFERENCES

[1] P. J. Roig, S. Alcaraz, K. Gilly, C. Bernad, and C. Juiz, "Arithmetic Study about Efficiency in Network Topologies for Data Centers," *Network*, vol. 3, no. 3, pp. 298–325, Sep. 2023, https://doi.org/10.3390/network3030015.

[2] B. E. Sabir, M. Youssfi, O. Bouattane, and H. Allali, "Towards a New Model to Secure IoT-based Smart Home Mobile Agents using Blockchain Technology," *Engineering, Technology & Applied Science Research*, vol. 10, no. 2, pp. 5441–5447, Apr. 2020, https://doi.org/10.48084/etasr.3394.

[3] S. M. Altowaijri and Y. E. Touati, "Securing Cloud Computing Services with an Intelligent Preventive Approach," *Engineering, Technology & Applied Science Research*, vol. 14, no. 3, pp. 13998–14005, Jun. 2024, https://doi.org/10.48084/etasr.7268.

[4] A. Sunyaev, *Internet Computing: Principles of Distributed Systems and Emerging Internet-Based Technologies*, 2nd ed. Cham, Switzerland: Springer, 2024, https://doi.org/10.1007/978-3-031-61014-1.

[5] N. Chen, T. Qiu, X. Zhou, K. Li, and M. Atiquzzaman, "An Intelligent Robust Networking Mechanism for the Internet of Things," *IEEE Communications Magazine*, vol. 57, no. 11, pp. 91–95, Nov. 2019, https://doi.org/10.1109/MCOM.001.1900094.

[6] M. Humayun, N. Tariq, M. Alfayad, M. Zakwan, G. Alwakid, and M. Assiri, "Securing the Internet of Things in Artificial Intelligence Era: A Comprehensive Survey," *IEEE Access*, vol. 12, pp. 25469–25490, 2024, https://doi.org/10.1109/ACCESS.2024.3365634.

[7] M. Hayamizu and K. Makino, "Ranking Top-k Trees in Tree-Based Phylogenetic Networks," *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, vol. 20, no. 3, pp. 2349–2355, May 2023, https://doi.org/10.1109/TCBB.2022.3229827.

[8] W. Yang, C. Chen, and J. Yu, "Topology-Aware Node Allocation on Supercomputers with Hierarchical Network," in *2022 IEEE 24th Int Conf on High Performance Computing & Communications; 8th Int Conf on Data Science & Systems; 20th Int Conf on Smart City; 8th Int Conf on Dependability in Sensor, Cloud & Big Data Systems & Application*, Hainan, China, 2022, pp. 1095–1100, https://doi.org/10.1109/HPCC-DSS-SmartCity-DependSys57074.2022.00173.

[9] A. C. Castillo, "Various Network Topologies and an Analysis Comparative Between Fat-Tree and BCube for a Data Center Network: An Overview," in *2022 IEEE Cloud Summit*, Fairfax, VA, USA, 2022, pp. 1–8, https://doi.org/10.1109/CloudSummit54781.2022.00007.

[10] E. Jo, D. Pan, J. Liu, and L. Butler, "A simulation and emulation study of SDN-based multipath routing for fat-tree data center networks," in *Proceedings of the Winter Simulation Conference 2014*, Savannah, GA, USA, 2014, pp. 3072–3083, https://doi.org/10.1109/WSC.2014.7020145.

[11] X. Zhang and B. Hu, "A Power-Aware Routing Algorithm in Fat-tree Date Center Networks," in *2019 IEEE 20th International Conference on High Performance Switching and Routing*, Xi'an, China, 2019, pp. 1–6, https://doi.org/10.1109/HPSR.2019.8807996.

[12] A. Hammadi and L. Mhamdi, "A survey on architectures and energy efficiency in Data Center Networks," *Computer Communications*, vol. 40, pp. 1–21, Mar. 2014, https://doi.org/10.1016/j.comcom.2013.11.005.

[13] K. Siva Tharun and K. Kottilingam, "Optimization Load Balancing over Imbalance Datacenter Topology," in *New Trends in Computational Vision and Bio-inspired Computing*, S. Smys, A. M. Iliyasu, R. Bestak, and F. Shi, Eds. Cham, Switzerland: Springer, 2020, pp. 397–407, https://doi.org/10.1007/978-3-030-41862-5_38.

[14] J. J. Wilke and J. P. Kenny, "Opportunities and limitations of Quality-of-Service in Message Passing applications on adaptively routed Dragonfly and Fat Tree networks," in *2020 IEEE International Conference on Cluster Computing*, Kobe, Japan, 2020, pp. 109–118, https://doi.org/10.1109/CLUSTER49012.2020.00021.

[15] W.-H. Liang *et al.*, "Dynamic Flow Scheduling Technique for Load Balancing in Fat-Tree Data Center Networks," *International Journal of Performability Engineering*, vol. 17, no. 6, pp. 491–593, Jun. 2021, https://doi.org/10.23940/ijpe.21.06.p1.491503.

[16] H. Abdulrab, F. A. Hussin, A. Abd Aziz, A. Awang, I. Ismail, and P. A. M. Devan, "Reliable Fault Tolerant-Based Multipath Routing Model for Industrial Wireless Control Systems," *Applied Sciences*, vol. 12, no. 2, Jan. 2022, Art. no. 544, https://doi.org/10.3390/app12020544.

[17] S. Liu, H. Xu, and Z. Cai, "Low Latency Datacenter Networking: A Short Survey." arXiv, Jul. 31, 2014, https://doi.org/10.48550/arXiv.1312.3455.

[18] S. Supittayapornpong, P. Namyar, M. Zhang, M. Yu, and R. Govindan, "Optimal Oblivious Routing for Structured Networks," in *IEEE INFOCOM 2022 - IEEE Conference on Computer Communications*, London, United Kingdom, 2022, pp. 1988–1997, https://doi.org/10.1109/INFOCOM48880.2022.9796682.

[19] Y. Kulakov, S. Kopychko, and V. Gromova, "Organization of Network Data Centers Based on Software-Defined Networking," in *1st International Conference on Computer Science, Engineering and Education Applications*, Kiev, Ukraine, 2018, pp. 447–455, https://doi.org/10.1007/978-3-319-91008-6_45.

[20] Y. Satotani and N. Takahashi, "Depth-First Search Algorithms for Finding a Generalized Moore Graph," in *TENCON 2018 - 2018 IEEE Region 10 Conference*, Jeju, South Korea, 2018, pp. 0832–0837, https://doi.org/10.1109/TENCON.2018.8650418.

[21] L. Liu, L. Gong, S. Yang, J. Xu, and L. Fortnow, "Best First Fit (BFF): An Approach to Partially Reconfigurable Hybrid Circuit and Packet Switching," in *2018 IEEE 11th International Conference on Cloud Computing*, San Francisco, CA, USA, 2018, pp. 426–433, https://doi.org/10.1109/CLOUD.2018.00060.

[22] J. Duan and Y. Yang, "Fitted Fat-Tree for Localized Traffic in Data Center Networks," in *2018 IEEE International Conference on Communications*, Kansas City, MO, USA, 2018, pp. 1–6, https://doi.org/10.1109/ICC.2018.8422567.

[23] P. Isaia and L. Guan, "Distributed Mininet placement algorithm for fat-tree topologies," in *2017 IEEE 25th International Conference on Network Protocols*, Toronto, Canada, 2017, pp. 1–6, https://doi.org/10.1109/ICNP.2017.8117599.

[24] A. Biswas and A. Hussain, "Design and Analysis of a New Reduced Switch Scalable MIN Fat Tree Topology," *Computación y Sistemas*, vol. 26, no. 2, pp. 949–962, Jun. 2022, https://doi.org/10.13053/cys-26-2-3902.

[25] M. Aliyan *et al.*, "Analysis and Performance Evaluation of Various Shortest Path Algorithms," in *2024 3rd International Conference for Innovation in Technology*, Bangalore, India, 2024, pp. 1–16, https://doi.org/10.1109/INOCON60754.2024.10512150.

[26] C. Ulloa, J. Baier, W. Yeoh, V. Bulitko, and S. Koenig, "A Learning-Based Framework for Memory-Bounded Heuristic Search: First Results," *Proceedings of the International Symposium on Combinatorial Search*, vol. 10, no. 1, pp. 178–179, 2019, https://doi.org/10.1609/socs.v10i1.18479.

[27] A. K. Mishra and P. C. Siddalingaswamy, "Analysis of tree based search techniques for solving 8-puzzle problem," in *2017 Innovations in Power and Advanced Computing Technologies*, Vellore, India, 2017, pp. 1–5, https://doi.org/10.1109/IPACT.2017.8245012.

[28] Y. Sun, X. Yi, and H. Liu, "The Communication Analysis of Implementation in Breadth First Search Algorithm," in *2016 IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData)*, Chengdu, China, 2016, pp. 754–759, https://doi.org/10.1109/iThings-GreenCom-CPSCom-SmartData.2016.159.

[29] F. Zhang *et al.*, "An adaptive breadth-first search algorithm on integrated architectures," *The Journal of Supercomputing*, vol. 74, no. 11, pp. 6135–6155, Nov. 2018, https://doi.org/10.1007/s11227-018-2525-0.

[30] R. Kanniga Devi, M. Gurusamy, and P. Vijayakumar, "An Efficient Cloud Data Center Allocation to the Source of Requests," *Journal of Organizational and End User Computing (JOEUC)*, vol. 32, no. 3, pp. 23–36, 2020, https://doi.org/10.4018/JOEUC.2020070103.

[31] O. A. Rotimi, A. O. Olowoye, R. A. Ganiyu, S. O. Olabiyisi, and O. T. Amumeji, "An Exploratory study of Critical Factors Affecting the Efficiency of Uninformed Tree based Search Algorithms," *IOSR Journal of Mathematics*, vol. 14, no. 3, pp. 6–11, Jun. 2018.

[32] T. A. Akanmu, S. O. Olabiyisi, E. O. Omidiora, C. A. Oyeleye, M. A. Mabayoje, and A. O. Babatunde, "Comparative Study of Complexities of Breadth – First Search and Depth-First Search Algorithms, using Software Complexity Measures," in *Proceedings of the World Congress on Engineering*, London, United Kingdom, 2010, vol. 1.

[33] B. J. Saleh, A. Y. F. Saedi, A. T. Q. Al-Aqbi, and L. Abdalhasan Salman, "Optimum Median Filter Based on Crow Optimization Algorithm," *Baghdad Science Journal*, vol. 18, no. 3, Sep. 2021, Art. no. 8, https://doi.org/10.21123/bsj.2021.18.3.0614.

[34] X. Ying, "Design of the Data Center Network Architecture under the Background of Cloud Computing: A Review," *World Journal of Innovation and Modern Technology*, vol. 7, no. 4, pp. 55–61, Jul. 2024, https://doi.org/10.53469/wjimt.2024.07(04).07.

[35] S. Maudet, G. Andrieux, R. Chevillon, and J.-F. Diouris, "Refined Node Energy Consumption Modeling in a LoRaWAN Network," *Sensors*, vol. 21, no. 19, Oct. 2021, Art. no. 6398, https://doi.org/10.3390/s21196398.