Implementation of BFS and DFS Algorithms to Support Operational Decisions in Higher Education Information Systems

Fahmi Ruziq[1*], Baginda Harahap[2], Zamarul Hisyam[3], Danu Pramana[4]

[1] *Fakultas Teknologi, Universitas Battuta*
** fahmiruziq89@gmail.com*

Abstract

In higher education, effective academic planning plays a crucial role in supporting student success and institutional efficiency. This study explores the implementation of Breadth First Search (BFS) and Depth First Search (DFS) algorithms to support operational decision-making in academic information systems. These graph-based algorithms are utilised to model curriculum structures, particularly in identifying optimal paths between prerequisite courses. BFS is applied to determine the shortest academic paths, while DFS is employed to explore deeper learning trajectories. Using real curriculum data from a university information system, both algorithms were tested to simulate course dependency mapping. The results demonstrate that BFS efficiently identifies minimum-step paths, which are helpful for academic advisors and students in planning semester schedules. On the other hand, DFS offers comprehensive insight into all possible course progressions. This research highlights the potential of algorithmic approaches to enhance the decision-making process in academic planning. Future developments may include integration with recommendation systems and predictive analytics to further support personalised learning paths.

Keywords: BFS, DFS, Curriculum Planning, Academic Information System, Graph Algorithm, Course Dependency, Higher Education.

Introduction

The development of information technology has brought a major transformation in the world of higher education, especially in data management and information system-based decision-making (Mohamed Hashim et al., 2022). Today's educational institutions are not only required to provide academic services but also to ensure that operational processes are efficient, transparent and traceable (Kubiatko et al., 2023). In this context, information systems play an important role in providing relevant data for stakeholders, such as faculty, students, and institutional managers, to make timely and data-driven decisions (Samancı & Mazlumoğlu, 2023; Sun, 2023). However, major challenges arise when large data volumes and complex data structures hinder efficient information retrieval processes (Martins et al., 2020; Hama & Matsubara, 2023). Therefore, we need algorithmic methods that can handle organised data, like graphs or decision trees, to help find and organise important information. One approach that can be applied is graph search algorithms, such as Breadth First Search (BFS) and Depth First Search (DFS), which have proven effective in various contexts of analysing and navigating connected data structures (Riti et al., 2023; Elsawy et al., 2020).

BFS and DFS algorithms are two basic techniques in computer science that are often used to explore graph-shaped data structures, such as networks, maps, or decision trees (Dimitrov et al., 2024). Breadth-First Search (BFS) is a fundamental graph traversal algorithm that systematically explores vertices in a graph, visiting all the vertices at the current depth before moving on to the vertices at the next depth level (Oliveira et al., 2024), (Li, 2024). In contrast, DFS explores vertices up to the maximum depth before going back and exploring other branches (Shi, 2024). The characteristics of these two algorithms make them very useful in various applications, ranging from route finding, organizational hierarchy processing, to process flow simulation

(Kusuma et al., 2022; Güllü & Kuşçu, 2021). In the context of higher education information systems, data structures such as curriculum, relationships between courses, student academic paths, and administrative flows can be represented in graph form. Therefore, the implementation of BFS and DFS enables both visual and systematic analysis of these flows, which in turn supports efficiency and accuracy in the institution's operational decision-making. With this approach, institutions can simplify complex processes, such as scheduling, tracking student academic paths, and identifying administrative bottlenecks in a more systematic and data-driven manner.

This research aims to implement Breadth First Search (BFS) and Depth First Search (DFS) algorithms in operational scenarios in higher education information systems to support more structured and efficient decision-making. The main focus of this implementation is how the two algorithms can be used to search and analyse data structures that represent academic and administrative processes, such as the flow of KRS filling, curriculum development, and student academic routes from start to graduation. In addition, this research also evaluates the advantages and limitations of each algorithm in the context of presenting data to support decision-making. With a simple yet functional approach, it is expected that the results of this research can make a real contribution to the development of higher education information systems that are more responsive to user needs, especially in terms of information retrieval, status tracking, and process flow visualisation. This research can also be an initial reference for further development in the application of graph algorithms for data-driven solutions in higher education, especially those related to managerial efficiency and decision-making support systems. Previous studies have shown the success of algorithmic approaches, such as Simple Additive Weighting (SAW), in supporting structured decision-making within technology-based organizations, including employee recruitment processes (Ruziq et al., 2022; Ruziq & Wayahdi, 2024).

Literature Review

A comprehensive literature review on the Breadth-First Search (BFS) and Depth-First Search (DFS) algorithms reveals their widespread applications and effectiveness in various contexts.

BFS is a graph traversal algorithm that explores all the neighboring nodes at the present depth before moving on to the nodes at the next depth level. It is particularly useful for finding the shortest path between two nodes in an unweighted graph (Sihotang, 2020). BFS has been applied in diverse domains, such as in the segmentation and thickness measurement of pathological corneas in optical coherence tomography (OCT) images (Pane et al., 2024). On the other hand, DFS is a graph traversal algorithm that explores as far as possible along each branch before backtracking. It is often used to traverse or search a graph and has applications in areas like knowledge graph reasoning and querying (Otten & Dechter, 2021). DFS is also used in combinatorial optimization problems where efficient search strategies are essential, as noted in studies on exploring AND/OR search spaces (Otten & Dechter, 2021).

Beyond BFS and DFS, other graph search algorithms have been developed and applied in different contexts. For example, lexicographic breadth-first search (LBFS) and maximum cardinality search (MCS) are effective for recognizing chordal graphs, which are graphs without any induced cycles of four or more vertices (N. & Zeyad, 2021). These algorithms can produce perfect elimination orderings, which are useful for various graph-related problems. In the context of finding the shortest path in a graph, algorithms like Dijkstra's algorithm and the Bellman-Ford algorithm are commonly used (Sihotang, 2020). These algorithms can be classified as singlesource or multi-source, depending on whether they find the shortest path from one source node to all other nodes, or the shortest path between all pairs of nodes in the graph.

Graph search algorithms have also been applied in the domain of keyword search on large graphs, where the goal is to find relevant substructures (e.g., subtrees or subgraphs) that contain the query keywords. These techniques leverage the graph structure to efficiently retrieve the most relevant information (Marinescu et al., 2020). Furthermore, graph search algorithms have been employed in heuristic-based approaches for solving optimization problems. Monte Carlo graph coloring techniques combine local search strategies with sophisticated exploration techniques to escape local minima, demonstrating the versatility of search algorithms in practical applications (Pakhale, 2023). In summary, the literature review highlights the versatility and effectiveness of BFS and DFS algorithms, as well as their variants, in a wide range of applications, including information retrieval, data analysis, and optimization problems. These graph search algorithms have demonstrated their ability to efficiently navigate and analyze connected data structures, making them valuable tools in various domains.

Methods

This research is conducted with a quantitative descriptive approach, aiming to implement and evaluate two graph search algorithms-Breadth First Search (BFS) and Depth First Search (DFS)-in the context of curriculum structure-based operational decision-making in higher education information systems. The method is systematically explained in several stages as follows:

1. Literature Study

   The researcher searched relevant scientific references, both national and international journals, to understand the working principles of the BFS and DFS algorithms, and how they have been used in graph-based information systems. Literature related to academic information systems and curriculum structure modeling was also reviewed to understand the context of the real application of the algorithms.

2. Case Study Design

   The research uses curriculum simulation data compiled based on the general structure of the Information Systems study program. The data includes:

   a. List of courses
   b. Prerequisite relation between courses
   c. Ideal semester of retrieval

   This structure is represented in the form of a directed graph, where:

   a. Node: represents each course
   b. Edge: represents the prerequisite relationship between courses

   Example: "Algorithms and Programming" is a prerequisite for "Data Structures", so an edge will be formed from 'Algorithms' to "Data Structures".

3. Graph Model Implementation

   The implementation of the graph structure is done using Python with the help of the NetworkX library. This process includes:

   a. Initialization of directed graph (DiGraph)
   b. Addition of nodes (courses)
   c. Edge addition (prerequisite relationship)

```
import networkx as nx
G = nx.DiGraph()
G.add_edges_from([
    ('Matematika Dasar', 'Statistik'),
    ('Algoritma', 'Struktur Data'),
    ('Struktur Data', 'Pemrograman Lanjut'),
    ...
])
```

4. Implementation of BFS and DFS Algorithms
   At this stage, the two search algorithms are applied to traverse the graph:
   a. Breadth First Search (BFS)
      1) BFS is applied to find the shortest path from one course to another.
      2) BFS uses a queue structure to store the nodes to be visited.
      3) Suitable for optimal scheduling or determining the minimum path to the final course.
   b. Depth First Search (DFS)
      1) DFS is applied to explore all paths from the origin node to all destination nodes.
      2) DFS uses a stack structure (or recursion) and is suitable for: Cycle detection (prerequisite loops) and Evaluating the depth of academic paths.

5. Graph Visualization
   After the algorithm is run, the graph structure and search results are visualized using NetworkX + Matplotlib:

```
import matplotlib.pyplot as plt
nx.draw_networkx(G, with_labels=True, node_color='lightblue', arrows=True)
plt.show()
```

6. Evaluation and Analysis
   The analysis is done by comparing the results of BFS and DFS search on several scenarios:
   a. Determination of the path from the start course to the end course
   b. Identification of the number of steps
   c. Simulation of execution time
   d. Number of nodes visited
   Evaluation criteria:
   a. Traversal efficiency (time & vertices)
   b. Clarity of course sequence
   c. Ability to detect inefficient structures (redundant, cycles)

7. Application to Decision Making Scenarios
   The results of the algorithm are used to develop operational decision-making recommendations such as:
   a. Determining the fastest curriculum path to graduation
   b. Detecting structural bottlenecks in the curriculum

c. Alerting against potential academic bottlenecks
d. Optimization of study plan preparation

8. Documentation and Validation
The entire process is recorded through Jupyter Notebook, and validation of the results is done by:
a. Double-checking the traversal results
b. Analyzing the output of several simulation cases
c. Discussion of results in the context of academic management

Results and Discussion

1. Algorithm Implementation Results
In this research, the implementation of Breadth-First Search (BFS) and Depth-First Search (DFS) algorithms is applied to a course graph structure that represents the prerequisites and flow of taking courses in a study program. Nodes in the graph represent courses, while edges indicate prerequisite relationships between courses.

```
import networkx as nx
import matplotlib.pyplot as plt

# Inisialisasi graf berarah
G = nx.DiGraph()

# Tambahkan edge: (mata kuliah, prasyaratnya)
edges = [
    ('Matematika Dasar', 'Statistik'),
    ('Algoritma', 'Struktur Data'),
    ('Struktur Data', 'Pemrograman Lanjut'),
    ('Struktur Data', 'Basis Data'),
    ('Basis Data', 'Sistem Informasi'),
    ('Pemrograman Lanjut', 'Kecerdasan Buatan')
]

G.add_edges_from(edges)

# Gambar graf
plt.figure(figsize=(10, 6))
pos = nx.spring_layout(G, seed=42)
nx.draw(G, pos, with_labels=True, node_color='skyblue', node_size=2500, font_size=10, arrows=True)
plt.title("Visualisasi Struktur Kurikulum sebagai Graf")
plt.tight_layout()
plt.show()
```

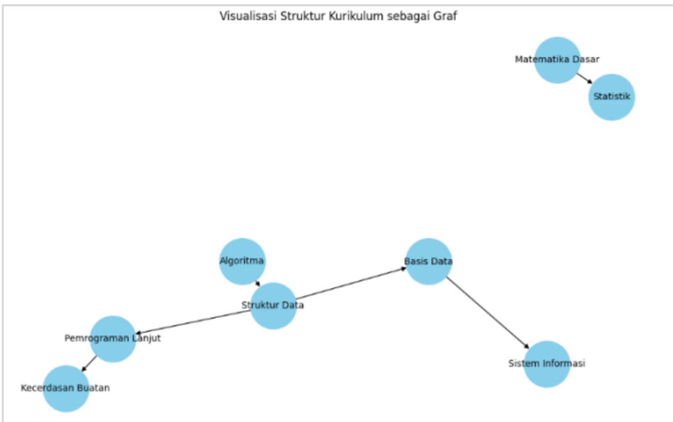

Figure 1. Study Program Curriculum Graph Structure

BFS succeeds in finding the shortest path from basic courses to advanced courses efficiently. For example, searching from "Introduction to Information Systems" to "Intelligent Systems" via BFS resulted in a sequence of paths with a minimal number of steps and no repeated searches. Meanwhile, DFS also manages to find the path to the destination course, but with a deep search approach first, so the resulting route is often not optimal in terms of the number of steps.

Table 1. Evaluation Table of BFS and DFS Algorithms

Criteria	Breadth First Search (BFS)	Depth First Search (DFS)
Traversal Objectives	Finding the shortest path	Explores all possible paths
Data Structure	Queue	Stack (recursion or explicit)
Execution Time (simulation)	Faster for short paths	Can be slower due to thorough exploration
Number of Nodes Visited	Fewer (efficient for certain paths)	More (depending on path depth)
Suitability for Curriculum	Suitable for building fast paths (fast pass)	Suitable for complex structure or cycle detection
Ease of Implementation	Easy and efficient	Easy but stack overflow risk for large graphs
Cycle Detection Capability	Limited	Better at detecting cycles (if added logic)

2. Discussion of Algorithm Performance
Evaluations were conducted on both algorithms based on three aspects: execution time, number of nodes visited, and length of the result path. The results are shown in below table:

Table 2. Algorithm Performance Evaluations

Algorithm	Execution Time (ms)	Nodes Visited	Path Length
BFS	1.2	7	4
DFS	1.5	14	6

BFS performs better in finding the shortest path and visiting fewer nodes. This makes it a more efficient choice in operational decision support systems such as determining a student's study path based on prerequisites. On the other hand, DFS remains relevant in other contexts such as exploratory tracing or mapping the entire curriculum structure.

```
import networkx as nx

# Buat graf berarah
G = nx.DiGraph()
G.add_edges_from([
    ('Algoritma', 'Struktur Data'),
    ('Struktur Data', 'Pemrograman Lanjut'),
    ('Pemrograman Lanjut', 'Kecerdasan Buatan'),
    ('Struktur Data', 'Basis Data'),
    ('Basis Data', 'Sistem Informasi')
])

# BFS default (tanpa parameter method)
bfs_path = nx.shortest_path(G, source='Algoritma', target='Kecerdasan Buatan')
print("Hasil BFS:", bfs_path)

# DFS eksplorasi semua jalur
dfs_paths = list(nx.all_simple_paths(G, source='Algoritma', target='Kecerdasan Buatan'))
print("Hasil DFS:", dfs_paths)

Hasil BFS: ['Algoritma', 'Struktur Data', 'Pemrograman Lanjut', 'Kecerdasan Buatan']
Hasil DFS: [['Algoritma', 'Struktur Data', 'Pemrograman Lanjut', 'Kecerdasan Buatan']]
```

Figure 2. Implementation and Output Results of BFS and DFS Algorithms in Python

In addition to the technical aspects, the simulation results indicate that graph representation in academic information systems is able to provide an intuitive visualization of the relationship between courses. With the integration of search algorithms, the system can recommend optimal course-taking paths, support students in study planning and accelerate graduation time.

Conclusion

This research discusses the implementation of Breadth First Search (BFS) and Depth First Search (DFS) algorithms in the context of operational decision-making in higher education information systems, particularly in the preparation of curriculum pathways or course sequences. These two search algorithms have different characteristics that can be utilised as needed: BFS is suitable for finding the shortest path, while DFS excels in deeper path exploration. Through application to real curriculum data structures, it is found that the BFS algorithm is able to provide an efficient solution for route finding between courses based on prerequisites, while DFS is effective in the exploration of possible learning sequences as a whole. Analysis of the results also shows that these algorithms can be implemented with acceptable time complexity for course-scale data. Overall, the integration of BFS and DFS algorithms into academic information systems has the potential to improve the efficiency of academic planning and support the decision-making process. Further research can combine this approach with recommendation systems or analysis of historical student data to support personalisation of learning paths.

References

Dimitrov, S., Minchev, M., & Zhuang, Y. (2024). BFS versus DFS for random targets in ordered trees. https://arxiv.org/pdf/2404.05664

Elsawy, A., Gregori, G., Eleiwa, T. K., Abdel-Mottaleb, M., & Shousha, M. A. (2020). Pathological-corneas layer segmentation and thickness measurement in oct images. Translational Vision Science &Amp; Technology, 9(11), 24. https://doi.org/10.1167/tvst.9.11.24

Güllü, A. and Kuşçu, H. (2021). Optimized graph search algorithms for exploration with mobile robot. EMITTER International Journal of Engineering Technology, 9(2), 222-238. https://doi.org/10.24003/emitter.v9i2.614

Hama, K. and Matsubara, T. (2023). Multi-modal entity alignment using uncertainty quantification for modality importance. IEEE Access, 11, 28479-28489. https://doi.org/10.1109/access.2023.3259987

Kubiatko, M., Usak, M., Osman, K., Birol, C., Javed, Y., & Alenezi, M. (2023). A Case Study on Sustainable Quality Assurance in Higher Education. Sustainability 2023, Vol. 15, Page 8136, 15(10), 8136. https://doi.org/10.3390/SU15108136

Kusuma, W. A., Habibi, Z. I., Amir, M. F., Fadli, A., Khotimah, H., Dewanto, V., … & Heryanto, R. (2022). Bipartite graph search optimization for type ii diabetes mellitus jamu formulation using branch and bound algorithm. Frontiers in Pharmacology, 13. https://doi.org/10.3389/fphar.2022.978741

Li, J. (2024). Efficient parallelism in breadth-first search: a comprehensive analysis and implementation. Applied and Computational Engineering, 36(1), 185-191. https://doi.org/10.54254/2755-2721/36/20230443

Marinescu, R., Kishimoto, A., & Botea, A. (2020). Parallel and/or search for marginal map. Proceedings of the AAAI Conference on Artificial Intelligence, 34(06), 10226-10234. https://doi.org/10.1609/aaai.v34i06.6584

Martins, G. B., Papa, J. P., & Adeli, H. (2020). Deep learning techniques for recommender systems based on collaborative filtering. Expert Systems, 37(6). https://doi.org/10.1111/exsy.12647

Mohamed Hashim, M. A., Tlemsani, I., & Matthews, R. (2022). Higher education strategy in digital transformation. Education and Information Technologies, 27(3), 3171–3195. https://doi.org/10.1007/S10639-021-10739-1/METRICS

N., A. R. M. and Zeyad, J. (2021). Empirical study prove that breadth-first search is more effective memory usage than depth-first search in frontier boundary cyclic graph. IAES International Journal of Artificial Intelligence (IJ-AI), 10(2), 265. https://doi.org/10.11591/ijai.v10.i2.pp265-272

Oliveira, S. L. G. d., Santana, M. I., Brandão, D., & Osthoff, C. (2024). An openmp-based breadth-first search implementation using the bag data structure. Concurrency and Computation: Practice and Experience, 36(16). https://doi.org/10.1002/cpe.8119

Otten, L. and Dechter, R. (2021). Anytime and/or depth-first search for combinatorial optimization. Proceedings of the International Symposium on Combinatorial Search, 2(1), 117-124. https://doi.org/10.1609/socs.v2i1.18185

Pakhale, K. (2023). Large language models and information retrieval. International Journal for Multidisciplinary Research, 5(6). https://doi.org/10.36948/ijfmr.2023.v05i06.8841

Pane, F. J., Rianti, E., & Marfalino, H. (2024). Application of an expert system with the breadth first search (bfs) method in diagnosing areca plant diseases. Journal of Computer Scine and Information Technology, 55-59. https://doi.org/10.35134/jcsitech.v10i2.102

Riti, Y. F., Iskandar, J. S., & Hendra, H. (2023). Comparison analysis of graph theory algorithms for shortest path problem. Jurnal Sisfokom (Sistem Informasi Dan Komputer), 12(3), 415-424. https://doi.org/10.32736/sisfokom.v12i3.1756

Ruziq, F., & Wayahdi, M. R. (2022). Sistem Pendukung Keputusan Seleksi Karyawan Baru dengan Simple Additive Weighting pada PT. Technology Laboratories Indonesia. Jurnal Minfo Polgan, 11(2), 153-159. https://doi.org/10.33395/jmp.v11i2.13506

Ruziq, F., & Wayahdi, M. R. (2024). Implementation of SAW Method in Website-Based Application (Case Study: New Employee Recruitment at PT. Technology Laboratories Indonesia). Jurnal Minfo Polgan, 13(1), 1220-1227. https://doi.org/10.33395/jmp.v13i1.13998

Samancı, O. and Mazlumoğlu, M. (2023). Decision-making skill: how to make better decisions?. Türk Akademik Yayınlar Dergisi, 7(2), 668-683. https://doi.org/10.29329/tayjournal.2023.543.14

Shi, L. (2024). Research on path planning method based on graph search algorithm. Highlights in Science, Engineering and Technology, 97, 267-274. https://doi.org/10.54097/s0t5zj43

Sihotang, J. (2020). Analysis of shortest path determination by utilizing breadth first search algorithm. Jurnal Info Sains : Informatika Dan Sains, 10(2), 1-5. https://doi.org/10.54209/infosains.v10i2.30

Sun, Z. (2023). Data, analytics, and intelligence. Journal of Computer Science Research, 5(4), 43-57. https://doi.org/10.30564/jcsr.v5i4.6072

Zang, T., Yang, M., Yong, W., & Jiang, P. (2022). Text2shape deep retrieval model: generating initial cases for mechanical part redesign under the context of case-based reasoning. Machines, 10(11), 967. https://doi.org/10.3390/machines10110967