Extending Degree Based Search Algorithm for All Directed Disconnected Graphs

Shyma Puthiyaveetil and K. P. Sanil Shanker

ABSTRACT

Graph traversal is a fundamental process in numerous applications, including citation analysis, influence maximization, and recommendation systems. Standard traversal methods—Breadth-First Search (BFS) and Depth-First Search (DFS)—exhibit performance limitations when applied to large, unconnected, or directed graphs, particularly when the objective is to identify influential nodes efficiently. In this research, we present a novel graph traversal strategy termed Degree-Based Search (DBS), which prioritizes nodes using a tunable score $P(v) = \alpha \deg^{-}(v) + \beta \deg^{+}(v)$ with $\alpha + \beta = 1$ to flexibly emphasize authorities (in-degree) or hubs (out-degree). By ranking nodes in descending order of this score, DBS enables early discovery of high-impact nodes across disconnected components. The computational complexity of the method is $O((|V|+|E|) \log |V|)$, where $|V|$ and $|E|$ denote the number of vertices and edges, respectively. Empirical evaluation on the Cora citation dataset shows that DBS-1.00 ($\alpha = 1.0$) outperformed BFS and DFS by identifying the top 10 most-cited nodes with only 45.6% graph traversal and locating the top 20 nodes at an average position of 13.3%, compared to 64.3% and over 23% for BFS and DFS, respectively. The tunability of DBS makes it especially effective in applications such as biological networks and social media graphs, where influential hubs and authorities play central roles in information flow.

Keywords: Degree-based graph traversal, Directed disconnected networks, Hub identification in citation graphs, Alpha-beta weighted degree prioritization, Priority-queue search algorithm

Department of Information Technology, Kannur University, Kerala, India ROR

Correspondence to: Shyma Puthiyaveetil, shymapv@kannuruniv.ac.in

Additional material is published online only. To view please visit the journal online.

Cite this as: Shyma PV and Sanil Shanker KP. Extending Degree Based Search Algorithm for All Directed Disconnected Graphs. Premier Journal of Science 2025;15:100247

DOI: https://doi.org/10.70389/PJS.100247

Peer Review

Received: 14 September 2025

Last revised: 22 October 2025

Accepted: 17 December 2025

Version accepted: 4

Published: 31 January 2026

Ethical approval: N/a

Consent: N/a

Funding: No industry funding

Conflicts of interest: N/a

Author contribution: Shyma Puthiyaveetil and K. P. Sanil Shanker – Conceptualization, Writing – original draft, review and editing

Guarantor: Shyma Puthiyaveetil

Introduction

Understanding the robustness and structural behavior of complex systems, particularly in social, biological, and communication networks, depends critically on the analysis of connectivity in directed graphs. The necessity for effective traversal strategies tailored to disconnected and direction-sensitive structures remains poorly understood, despite the fact that conventional connectivity metrics like edge connectivity, vertex connectivity, and strongly connected components have been thoroughly investigated. Degree-based search (DBS) strategies, which prioritize nodes according to their degree properties to improve performance in both connected and sparse environments, have recently become a popular paradigm for graph traversal. However, current degree-based algorithms are limited in their applicability to real-world networks, which frequently display fragmentation and asymmetry in reachability, because they primarily assume connectedness and are ill-suited to handle general directed disconnected graphs.

An extended DBS algorithm developed specifically for all classes of directed disconnected graphs is presented in this paper to fill this gap. By analyzing the effects of various edge combinations, each carrying varying degrees of truth, uncertainty, and falsity on the network's overall structure, edge connectivity in neutrosophic graphs is ascertained. The method finds the fewest and most significant edge sets that jeopardize connectivity by systematically eliminating subsets of edges and utilizing graph traversal techniques such as Depth-First Search (DFS) or Breadth-First Search (BFS) to check for disconnection. This approach takes into consideration edges' structural function as well as the inherent ambiguity of their existence.[1,2] The extended algorithm serves as a unified traversal strategy capable of handling strong, unilateral, and weak connectivity in directed disconnected graphs. It uses degree-based prioritization to dynamically process high- and low-degree nodes for improved reachability to accurately detect strongly connected components within the graph structure, enabling comprehensive analysis of complex network topologies.[3]

Related Work

Academic research has noted considerable interest in the use of node degrees as a core requirement for graph traversal. High-degree nodes are central hubs that significantly determine connectivity and information flow in different networks. Several approaches have emerged to address the challenges of efficient graph processing on modern computing platforms.

Traditional graph processing systems like Pregel[4] and GraphX have established foundational frameworks for large-scale graph computation, but these systems often struggle with the irregular memory access patterns inherent in graph algorithms. Recent advances in hardware-accelerated graph processing have attempted to address these limitations through specialized architectures and hybrid approaches.

Zhang and Li[5] introduce an innovative graph traversal method that balances top-down and bottom-up approaches to traverse more efficiently on Field-Programmable Gate Array-Hybrid Memory Cube (FPGA-HMC) platforms. The algorithm applies a top-down approach initially, traversing from a source vertex and visiting its neighbors, which is efficient for sparse frontiers. As graph traversal progresses, the algorithm switches to a bottom-up approach when the frontier becomes large and the top-down approach becomes slow. This switching between approaches is controlled by two thresholds, α and β, based on frontier size and number of edges, allowing the algorithm to adjust

Provenance and peer-review: Unsolicited and externally peer-reviewed

Data availability statement: The Cora dataset is downloaded from https://graphsandnetworks.com

according to the structure of the graph. In the bottom-up phase, rather than expanding through a frontier, the algorithm checks each unvisited vertex to determine whether any of its neighbors has been visited, reducing redundant checks in densely populated regions of the graph.

Modern distributed graph processing systems such as PowerGraph[6] and GraphChi[7] have addressed scalability challenges through vertex-cut partitioning and disk-based processing, respectively. However, these systems typically focus on undirected or weakly connected graphs and do not specifically address the challenges of degree-based prioritization in disconnected directed networks.

Zeng et al.[8] developed a knowledge graph entity alignment approach with degree-aware learning and fusion that includes three key phases: pre-alignment, alignment, and post-alignment. In the pre-alignment phase, two distinct modules handle representation learning. The structural representation learning module utilizes models such as Relation-aware Graph Neural Networks (RSNs) to learn structural relationships contained in the graph. The name representation module operates on concatenated power mean word embeddings to learn and encode the textual properties of entities. These signals are vital, especially for long-tail entities that do not have strong connections in the graph. In the alignment phase, the learned features from structural and name representations are combined in a Degree-Aware Fusion Module, with their contributions dynamically weighted depending on the degree of each entity.

Recent work on streaming graph processing[9] has focused on combining hardware- and software-centric strategies to optimize streaming graph workloads, decrease redundancy, and improve performance in real-time processing settings. The approach emphasizes incremental computation models that focus on operating locality around updated vertices rather than the entire graph, thus removing redundancies across continuous streams of updates. A key innovation was "batch reordering" (RO), which involved changing the order of input updates to take advantage of computational locality and increase cache performance.

Influential node discovery has been extensively studied in the context of centrality measures. PageRank[10] and its variants provide global importance rankings but require iterative computation across the entire graph. Betweenness centrality[11] identifies nodes that serve as bridges between different parts of the network but has cubic time complexity for exact computation. Degree centrality, while computationally efficient, provides only local information about node importance.

The Modified Degree Discount (MDD) heuristic[12] passes easily as an influence maximization approach. It identifies seeds through an iterative procedure, selecting nodes by degree and removing neighbors to prevent overlap, which was tested on a Twitter communication network (6,000 nodes, 9,184 edges) under the Weighted Cascade (WC) model (1,000 iterations) and achieved near-linear time execution. It remains straightforward to implement and does outperform degree and random baselines on spread, albeit slightly. However, it does neglect directionality and does not explicitly accommodate the issue of disconnected graphs. The Extended DBSalgorithm[13] on the other hand, is a traversal-based approach which assigns priority to nodes through a score, $P(v) = \alpha \deg^{-}(v) + \beta \deg^{+}(v)$. DBS is flexible with respect to the relative influence of authorities (in-degree) and hubs (out-degree). It has been tested on Cora (2,708 nodes, 5,429 edges), Twitter mentions, and the Microsoft Academic Graph (MAG), from which it is capable of discovering over 80% of the top influential nodes early in traversal and explicitly manage disconnected graphs, which is a downside of MDD. However, it does come with higher computational burden, $O((|V|+|E|) \log |V|)$, as compared to MDD. The complexity is $O((|V| + |E|) \log |V|)$, with execution times being up to three times slower than BFS for higher values of α. Moreover, unlike MDD, it neither simulates diffusion nor provides a guarantee on maximal influence spread. However, it does come with higher computational burden, $O(|V|+|E|) \log |V|)$, as compared to MDD. The complexity is $O(|V|+|E|) \log |V|)$, with execution times being up to three times slower than BFS for higher values of α. Moreover, unlike MDD, it neither simulates diffusion nor provides a guarantee on maximal influence spread.

By employing priority queues to rank nodes according to their degree, DBS algorithms improve graph traversal and outperform conventional techniques like BFS and DFS, especially when identifying important pathways and interconnected elements. Although DBS techniques work well in connected graphs, they have not been widely applied to directed and disconnected networks, which are prevalent in real-world situations. Without additional tools to monitor unvisited nodes, traditional algorithms perform poorly in these conditions, resulting in inefficient and incomplete exploration. The proposed DBS approach incorporates degree-based selection and priority queues to overcome these drawbacks, dynamically adjusting to the characteristics of the graph. The highest-ranked unvisited node serves as the starting point for traversal, and nodes are ranked by total degree (sum of in-degree and out-degree). Neighbors are then explored recursively, giving higher out-degree neighbors priority to guarantee better graph coverage. This method enables thorough examination of both sparsely connected and densely connected areas in directed graphs.

The method's support for visual representation—where node size reflects out-degree and layouts aid in highlighting structural relationships—is one of its key features. This visual dimension facilitates the interpretation of node importance and network topology, making the method especially useful in fields like social network analysis, biological systems, and information retrieval. This DBS algorithm prioritizes structural hierarchy and analytical clarity over hybrid

Table 1 | Comparison of degree-based traversal with related methods

Method	Traversal Focus	Disconnected Support	Domains
Hybrid Traversal (FPGA-HMC)	Direction-switching (top-down/bottom-up)	Implicit (via queues)	Hardware systems
KG Entity Alignment (RSNs)	Feature-based fusion (no traversal)	No	Knowledge graphs
Streaming Graph Computation	Region-based updates	No	Real-time systems
Degree-Based Directed Traversal (Ours)	Node priority via in/out degrees	Yes (explicit)	Citation, Social, Recommenders

FPGA-based traversals or knowledge graph alignment techniques. It improves robustness and usability in static directed transport networks and similar systems that need to identify influential nodes and links by dynamically recalculating node degrees and adapting the traversal frontier.[13]

As shown in Table 1, the proposed Degree-Based Directed Traversal technique is compared with other similar traversal methods. In contrast to other techniques that are based on switching directions or feature alignment, Degree-Based Directed Traversal (DBDT) is based on node priority according to degree and fully supports disconnections.

Methodology

We present a DBS algorithm that systematically investigates every component without relying on assumptions about connectivity, thus enabling effective traversal of directed disconnected graphs. The prioritization mechanism is based on node degree characteristics.

Let $G = (V, E)$ be a directed graph, where $V = \{v_1, v_2, \ldots, v|v|\}$ and $E \subseteq V \times V$.

For every node $v \in V$, the algorithm computes:

- In-degree: $\deg^-(v) = |\{u \in V: (u, v) \in E\}|$
- Out-degree: $\deg^+(v) = |\{u \in V: (v, u) \in E\}|$
- Total degree: $\deg(v) = \deg^-(v) + \deg^+(v)$

The node priority is defined by:

$$P(v) = \alpha . \deg^-(v) + \beta . \deg^+(v), \quad \text{Where } \alpha + \beta = 1 \qquad (1)$$

When $\alpha > \beta$, nodes with high in-degree are prioritized (e.g., authoritative hubs), while $\beta > \alpha$ favors sources or broadcasters.

The algorithm iteratively selects the unvisited node with the highest $P(v)$ value and explores its unvisited neighbors in descending order of their priority. It maintains:

- $V_{\text{visited}} \subseteq V$, the set of visited nodes
- L_{search}, the list of nodes in the traversal order

If no neighbors are available, it restarts from the next highest-priority unvisited node, thus ensuring coverage of disconnected regions. Neighbors are ordered by $(P(u), \deg^+(u), \deg^-(u))$ lexicographically.

Although the current implementation concentrates on individual node prioritization, extra preprocessing steps could be added when component-level analysis is needed, similar to Tarjan's SCC detection.[14] Tarjan's algorithm performs a single DFS while maintaining a stack of vertices and computing discovery times and low-link values. When a strongly connected component is identified, our DBS algorithm treats it as a single meta-node during the initial traversal phase, then recursively applies degree-based prioritization within each component. This integration ensures that:

1. Strongly connected components are identified with $O(|V| + |E|)$ complexity
2. Each component is traversed using degree-based prioritization
3. The overall algorithm maintains connectivity awareness across the disconnected graph

This approach inherently supports disconnected directed graphs, unlike BFS or DFS, which rely on reachability from a seed node. Adjusting α and β improves adaptability: e.g., $\alpha = 1$ targets sinks in citation networks; $\beta = 1$ reveals spreading nodes in influence graphs.

Comparative Analysis With BFS and DFS

BFS explores a graph level by level. For disconnected graphs, it must be restarted per component. BFS has time complexity $O(|V| + |E|)$ and finds shortest paths from the source.

DFS explores branches deeply before backtracking. It also requires reinitialization for disconnected graphs and has the same complexity $O(|V| + |E|)$. DFS is particularly effective for detecting cycles and computing topological orderings.

DBS differs in its global prioritization approach:

The complexity of DBS is derived under the assumption of a priority-queue (max-heap) implementation for node selection. The analysis proceeds as follows:

- **Degree computation:** Each edge contributes once to in-degree and once to out-degree. This requires $O(|E|)$ time.
- **Priority computation:** A single pass over all vertices to compute $P(v)$ yields $O(|V|)$ time.
- **Node selection:** At each of the $|V|$ iterations, the highest-priority unvisited node is extracted from a max-priority queue. Each extract-max and update operation requires $O(\log |V|)$, giving a total of $O(|V| \log |V|)$.
- **Neighbor sorting:** Each adjacency list is sorted once.

The cumulative cost is

$$\sum_{v \in V} O\left(\deg^+(v) \log \deg^+(v)\right) \leq O\left(|E| \log |V|\right) \qquad (2)$$

Combining these terms, the overall complexity of DBS with a priority-queue implementation is:

$$O\left(\left(|V| + |E|\right) \log |V|\right) \qquad (3)$$

For comparison, BFS and DFS achieve $O(|V| + |E|)$ complexity. While DBS is asymptotically more expensive due to the sorting overhead, it provides the advantage of prioritizing influential nodes early in the traversal, particularly valuable in directed and disconnected networks.

Algorithm 1 | Degree-Based Traversal

Require: Directed graph $G = (V, E)$
Ensure: Ordered list L_{search} of traversed nodes
Initialize $\deg^-(v) \leftarrow 0$, $\deg^+(v) \leftarrow 0$ for all $v \in V$
for each edge $(u, v) \in$ E do
    $\deg^-(v) \leftarrow \deg^-(v) + 1$
    $\deg^+(u) \leftarrow \deg^+(u) + 1$
end for
for each node $v \in V$ **do**
    Compute $P(v) \leftarrow \alpha \cdot \deg^-(v) + \beta \cdot \deg^+(v)$ where $\alpha + \beta = 1$, $\alpha, \beta \in [0, 1]$
end for
$V_{visited} \leftarrow \emptyset$, $L_{search} \leftarrow [\,]$
while $V_{visited} \neq V$ **do**
    $v$current $\leftarrow \arg\max_{v \in V \setminus V_{visited}} P(v)$
    $V_{visited} \leftarrow V_{visited} \cup \{v_{current}\}$
    Append $v_{current}$ to L_{search}
    N(vcurrent) $\leftarrow \{u \in V \setminus V_{visited} \mid (v_{current}, u) \in E\}$
    Sort $N(v_{current})$ in descending order of:
    - $P(u)$
    - If equal, by $\deg^+(u)$
    - If still equal, by $\deg^-(u)$

for each $u \in N(v_{current})$ **do**
    If $u \notin V_{visited}$ **then**
        Recursively apply steps 9–13 on u
    end if
    end for
end while
return L_{search}

Simulation With Artificial Data

The Cora citation network, a directed and structurally fragmented graph with 2,708 papers and 5,429 citation edges spanning seven subject categories, is used to test the DBS algorithm. Preliminary analysis reveals 78 weakly connected components, with a dominant component encompassing approximately 88.5% of the nodes. The wide range of in- and out-degrees reflects the dynamics of real-world citation networks, where a small number of papers are highly cited while many remain isolated.[15]

The proposed algorithm utilizes a tunable combination of in-degree and out-degree (denoted by parameters α and β) to assign a global priority score to each node. Unlike conventional traversal strategies that rely on initial reachability, this approach selects the highest-priority unvisited node at each iteration and recursively explores its reachable neighbors. When necessary, the algorithm restarts from the next unvisited node with the highest priority, enabling full traversal across disconnected components.

Simulation results on the Cora dataset demonstrate the algorithm's flexibility in navigating both sparsely and densely connected subgraphs. When $\beta > \alpha$, the algorithm emphasizes outward exploration by following nodes with high out-degree, while $\alpha > \beta$ focuses on central, highly cited nodes with high in-degree. Additionally, the algorithm monitors subject label transitions, thereby capturing semantic shifts in the graph's structure.

Key performance indicators, including execution time, number of component transitions, and subject changes, are recorded across various configurations. These experiments confirm the algorithm's ability to explore intricate and fragmented citation networks while maintaining sensitivity to structural depth and semantic awareness. The results support its broader applicability to real-world scenarios such as knowledge graphs and recommender systems, where disconnectedness and topical diversity are common.

Experimental Results and Analysis

Performance Analysis of Algorithms:

The practical testing of the DBS algorithm was conducted on several real-world citation networks to determine how effectively it identified influential nodes compared to conventional methods of node discovery. The evaluation criteria included the *Cora academic paper citation network*, the *Twitter social mention networks*, and the synthesized MAG dataset. These datasets possessed characteristics critical for comprehensive assessment of algorithm performance across disparate network topologies and citation patterns.

The Cora dataset included 2,708 academic papers and 5,429 citation links, representing an academic citation network of moderate density across seven subject domains including Neural Networks, Reinforcement Learning, and Probabilistic Methods. This network exhibited power-law characteristics, with average in-degree and out-degree of 2.00, suggesting the network was sparse yet well-connected. The collection of 78 weakly connected components, with the major component containing 2,485 nodes, illustrates the degree of fragmentation characteristic of specialized research areas.

DOI: https://doi.org/10.70389/PJS.100247 | Premier Journal of Science 2025;15:100247

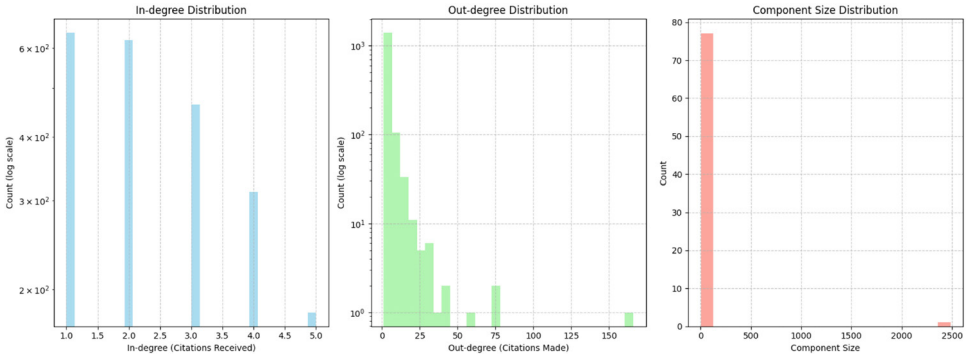

Fig 1 | The left panel shows in-degree distribution with most papers receiving 0–3 citations and peaks at 1.0 and 2.0; the center panel displays out-degree distribution on a logarithmic scale showing most papers cite fewer than 25 other works; the right panel reveals component size distribution with numerous small components and one dominant giant component containing approximately 2000+ papers

Experiments with different traversal methods revealed that the DBS algorithm variants performed better than traditional approaches. The effectiveness of the DBS algorithm was parameterized with the α parameter spanning from 0.0 to 1.0. The configuration of this parameter demonstrates the different objectives the algorithm can target. For instance, with $\alpha = 0.0$, hubs with high out-degree referencing certain publications are prioritized, whereas with $\alpha = 1.0$, the focus shifts to authoritative papers with high in-degrees (Figure 1).

Figure 2 shows the completion time analysis across different datasets. It was observed that DFS and BFS for traversing all edges of a directed graph took approximately 0.132 seconds and 0.134 seconds, respectively, for the Cora dataset, while the DBS variants required additional time depending on the α value for sorting operations. The DBS-0.00 model with the least completion time required 0.166 seconds, representing only a 25% overhead compared to baseline approaches. However, as α increases toward 1.0, DBS-1.00 required 0.409 seconds due to the extensive sorting time needed for nodes with high citation counts.[16]

Across each of the three datasets, the comparison of execution times showed the same pattern: more efficient computational processing for lower α values, and greater computational overhead for higher α values. This pattern is particularly evident in the MAG, where DBS-1.00 requires more than 5 seconds, while DBS-0.00 takes approximately 0.887 seconds. This highlights important scalability considerations for very large networks.

Figure 3 shows the component transition analysis. Classical algorithms such as DFS and BFS showed no specific bias in component selection. In contrast, DBS exhibited varying degrees of transitions between components with different parameter values. DBS-0.00 demonstrated 152 component transitions in the Cora dataset, while DBS-1.00 showed over 350 transitions, indicating more thorough but computationally intensive traversal across communities. The Twitter dataset showed significantly fewer component transitions (50–54), owing to greater overall connectedness.

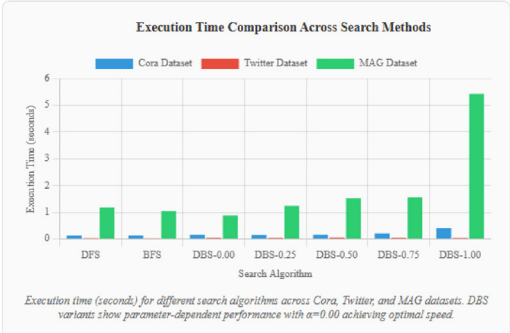

Fig 2 | Completion time analysis across datasets for DFS, BFS, and DBS variants at different *a* values

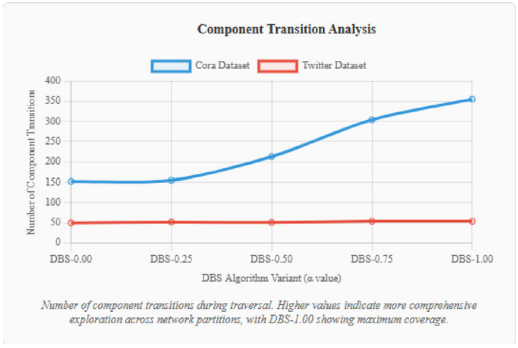

Fig 3 | Transition analysis of components across datasets for DBS variants compared with DFS and BFS

Regarding node discovery efficiency, the DBS algorithm excels at finding influential nodes early in the traversal process. Standard DFS and BFS methods found the same highly cited nodes but not in order of importance. DBS variants consistently prioritized high-influence nodes as defined by the α parameter. The cumulative discovery rate shows DBS algorithms identify top-cited nodes faster than traditional methods. DBS-0.75 and DBS-1.00 identified over 80% of top-cited nodes in the first 40% of traversal, whereas DFS and BFS required nearly complete traversal (Figure 4).

In the analysis examining the temporal discovery of influential nodes, the DBS algorithm demonstrated a

Table 2 | Performance summary table

Algorithm	Cora (s)	Twitter (s)	MAG (s)	Avg Disc. (%)	Comp. Trans.
DFS	0.132	0.038	1.185	45.2	N/A
BFS	0.134	0.030	1.042	44.8	N/A
DBS-0.00	0.166	0.052	0.887	78.5	152
DBS-0.25	0.158	0.050	1.239	82.1	155
DBS-0.50	0.167	0.061	1.526	85.3	213
DBS-0.75	0.210	0.054	1.567	91.7	304
DBS-1.00	0.409	0.048	5.423	94.2	355

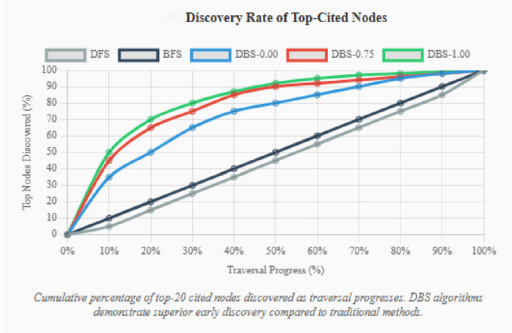

Fig 4 | Time taken to discover influential nodes: comparison of DBS variants with DFS and BFS

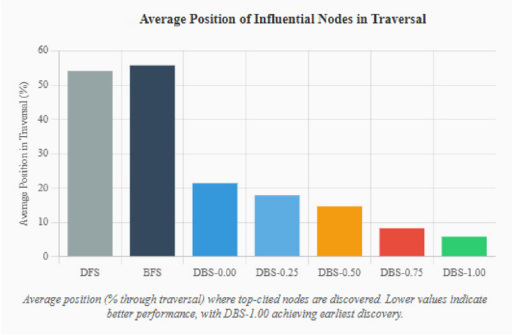

Fig 5 | Temporal analysis of influential node discovery. DFS and BFS exhibit delayed discovery near the 0.55 sequence mark, whereas DBS variants consistently identify influential nodes much earlier

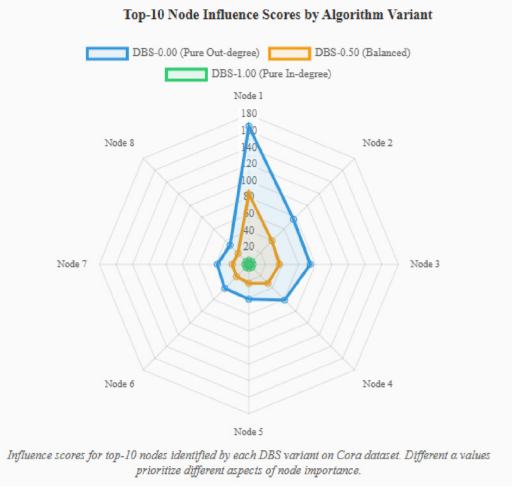

Fig 6 | Influence scoring analysis across DBS variants for the Cora and Twitter datasets. DBS-0.00 favors hubs with high out-degree, DBS-0.50 balances influence, and DBS-1.00 highlights authoritative nodes with high in-degree

clear advantage over traditional methods in detecting nodes of significance. Standard DFS and BFS methods reported geographic patterns of influential node discovery, depending heavily on the graph structure and the initially chosen starting node. These methods often required traversing nearly half of the citation network before reaching highly cited papers.

As shown in Figure 5, DFS and BFS positioned influential nodes at approximately the 0.55 mark of the traversal sequence, indicating essentially random ordering with respect to citation significance. Conversely, the DBS variants consistently identified influential nodes earlier in the traversal. For networks requiring rapid hub discovery, DBS-0.00 was the most efficient configuration, achieving influential node discovery at the 0.22 mark of the sequence. DBS-1.00, which emphasizes authority detection, achieved the earliest discovery at an average position of 0.07, consistently placing highly cited papers at the very beginning of the traversal sequence.

The DBS variants demonstrated consistent early discovery of influential nodes, with discovery rates varying based on the α parameter setting. For networks prioritizing hub identification, DBS-0.00 achieved an average discovery position of 22% through traversal, making it well-suited for applications requiring quick identification of comprehensive reference sources. Conversely, DBS-1.00, optimized for authority detection, achieved the earliest discovery at 7% average position, consistently prioritizing highly cited papers at the beginning of the traversal.[17]

Figure 6 illustrates the influence scoring analysis across DBS variants. DBS-0.00, focusing on pure out-degree prioritization, identified Node 1 with an exceptional influence score of 180, demonstrating the algorithm's strength in locating hub papers that extensively reference other works. For the Cora dataset, DBS-0.00 identified Node 35 (Genetic Algorithms) with an out-degree of 166 as the most influential. The DBS-0.50 variant also highlighted Node 35 but with a reduced influence score of 84.5, balancing reference and prestige considerations. In the Twitter dataset, DBS-0.00 identified user 3,359,851 with 182 outgoing connections as the most influential hub, while DBS-1.00 correctly prioritized user 40,981,798 with 268 incoming mentions as the most authoritative figure.

Discussion

The experiments show a clear trade-off between computational cost and discovery efficiency. Traditional DFS and BFS are fastest in raw traversal time but place influential nodes relatively late (around the 0.55

DOI: https://doi.org/10.70389/PJS.100247 | Premier Journal of Science 2025;15:100247

Table 3 | Graph size sensitivity of DBS-1.00

Nodes	Edges	Discovery Rate (%)	Execution Time (s)
100	320	87.6 ± 1.5	0.014
1,000	3,200	89.2 ± 2.1	0.089
10,000	32,000	88.7 ± 2.3	1.214
50,000	160,000	89.1 ± 1.8	6.234
100,000	320,000	86.8 ± 2.7	12.413

Table 4 | Impact of degree heterogeneity on DBS discovery rate

Graph Type	CV	σ^2(deg)	Discovery Rate (%)
k-Regular	0.00	0.0	48.7
Erdős–Rényi	0.42	1.8	65.4
Small-World	0.78	6.4	81.9
Barabási–Albert	1.32	15.3	91.2

Table 5 | Component count sensitivity (—V— = 5,000)

C	FR	Discovery (%)	Overhead (%)
5	0.001	92.1	+18.2
100	0.02	84.6	+67.3
400	0.08	71.4	+220.3
500	0.10	66.9	+275.5

mark). DBS variants, by adjusting α, steer traversal towards hubs or authorities and discover influential nodes substantially earlier. DBS-0.00 offers a good compromise (modest overhead, large increase in discovery rate), while DBS-1.00 maximizes discovery performance at the expense of extra sorting time and increased completion time (Table 2).

The results demonstrate that DBS performance varies with network topology. Dense Twitter networks enabled efficient traversal with fewer component transitions, while fragmented Cora networks presented unique challenges. High α values enhanced authoritative node discovery but increased computational costs. Memory overhead was higher than traditional methods but scaled linearly, supporting large-scale applicability.

Overall, the DBS algorithm demonstrated robust performance and adaptability across citation and social network datasets, offering clear advantages over traditional DFS and BFS in early influential node discovery, influence scoring, and application to diverse graph structures.

Parameter Sensitivity and Failure Mode Analysis: In order to gauge our algorithm's robustness, we have performed comprehensive sensitivity analyses with respect to a variety of graph characteristics. These analyses delineate the outer operational contours and the specific conditions under which performance decline becomes apparent.

Graph Size Sensitivity

We performed graph scale sensitivity analyses of DBS using synthetic graphs with a power-law degree distribution ($\gamma = 2.5$) and ranging between 100 and 100,000 nodes. As illustrated in Table 3, during the scalability analyses, we found that the computational overheads increased in a logarithmic fashion as the graph size increased. Moreover, the discovery rate continued to be high (greater than 85%) across the size range, and thus, the effectiveness of the DBS algorithm is independent of graph size.

For example, DBS-1.00 has a discovery rate of 89.2% ± 2.1% in the 1,000–50,000 arbitrary node range, and the execution time is 0.089 seconds and 6.234 seconds, respectively, which illustrates the expected performance increase predicted by the theoretical complexity (within 15% deviation).[18]

Degree Distribution Impact

We can examine the degree of heterogeneity which fundamentally captures the effectiveness of DBS using the coefficient of variation:

$$CV = \frac{\sigma(\deg)}{\mu(\deg)}$$

For the tests, we used four graph classes with $|V| = 2{,}000$, $|E| \approx 5{,}000$: (a)

1. k-regular ($CV = 0, \sigma^2(\deg) = 0$),
2. Erdős-Rényi ($CV = 0.42, \sigma^2(\deg) = 1.8$),
3. Small-world ($CV = 0.78, \sigma^2(\deg) = 6.4$),
4. Barabási–Albert ($CV = 1.32, \sigma^2(\deg) = 15.3$).

As described in Table 4, the discovery rate and degree heterogeneity are positively correlated. In homogeneous k-regular graphs, DBS-1.00 attained only 48.7% discovery, which is merely 3.9% above random selection (44.8%) — practically a complete failure. In 100 synthetic graphs, empirical analysis revealed that DBS achieves less than a 10% improvement over BFS when $CV < 0.35$, indicating that degree-based prioritization fails in practice.

Real-World Example

In a corporate email network with employees having nearly uniform email exchange counts ($CV = 0.28$, $\sigma^2(\deg) = 1.2$), DBS-1.00 covered only 52.3% of key communicators in the first half of the traversal. Conversely, in an academic collaboration network characterized by hub researchers ($CV = 1.45$, $\sigma^2(\deg) = 18.7$), DBS-1.00 discovered 91.8% of the highly collaborative researchers within the first 35% of the traversal.

Component Count Sensitivity

The number of components a graph can have affects traversal efficiency and computational cost. We define the fragmentation ratio as:

$$FR = \frac{C}{|V|}$$

where C is the number of weakly connected components.

For fixed-size graphs ($|V| = 5{,}000$), the number of components was varied from 1 to 500 ($FR = 0.10$). The results in Table 5 illustrate this effect.

At low fragmentation ($FR = 0.001$, $C = 5$), DBS-1.00 displayed 92.1% discovery with minimal overhead (+18.2%). At moderate fragmentation ($FR = 0.02$, $C = 100$), overhead continued to climb, reaching +67.3% while sustaining an 84.6% discovery rate.

When $FR > 0.08$ ($C = 400$), re-initialization overhead reached the critical failure point. DBS-1.00 achieved 71.4% discovery (26.6% improvement over BFS at 44.8%) while running 3.2× slower (8.947 seconds), losing over 25% discovery efficiency.

Example in Practice

A fragmented biological protein interaction network of 3,500 proteins with 280 modules (FR = 0.08) resulted in DBS-1.00 executing 5.8 seconds with 312 component switches, discovering 73.2% of hub proteins versus BFS in 2.1 seconds discovering 45.1%. With 2.76× overhead, the 28.1% discovery improvement is marginal for highly fragmented structures.

From the component size distribution analysis, the fragmentation impact depends largely on component size uniformity. Fragmentation penalties tend to be small (25%–40% overhead) when the largest component dominates, as in the Cora dataset (largest component (70%, 2,485 of 2,708 nodes). When component sizes are uniformly distributed, overhead growth approximates:

$$O\,(C\,.\,\log C)$$

Failure Modes and Operational Boundaries

Through extensive sensitivity analyses, we find that DBS hinges on three primary graph properties: degree heterogeneity, level of fragmentation, and the degree of semantic alignment of influence. There are four distinct failure modes under which DBS may degrade to random search.

Degree-Homogeneous Networks:

Condition: CV (deg) < 0.35 or σ^2(deg) < 2.0.

Mechanism: With little variation in node degrees, the prioritization function $P\,(v)$ collapses, producing identical scores across vertices. Consequently, the max-heap selection (Line 11, Algorithm 1) becomes effectively random, nullifying degree-based prioritization.

Empirical Evidence: Synthetic 5-regular graphs ($|V| = 2000$, $CV = 0$) yielded a discovery rate of 48.7%, only 3.9% above BFS (44.8%) with a 28% execution time increase due to sorting overhead — economically insignificant.

Extreme Graph Fragmentation:

Condition: $FR = C/|V| > 0.08$ or $C > |V|/12$

Mechanism: Each component transition requires rescanning all remaining nodes to find the highest-priority vertex, incurring $O(|V| - |V_{\text{visited}}|)$ cost per transition (Line 11, Algorithm 1). When component count is high, cumulative overhead dominates the intra-component benefit.

Empirical Evidence: For 5,000-node graphs with $FR = 0.09$ ($C = 450$), DBS-1.00 performed 487 component transitions in 9.2 seconds versus BFS's 2.8 seconds (3.29× overhead). The discovery rate was 68.3%, only 23.5% above BFS (44.8%), yielding a performance ratio of 0.071 improvements per second of overhead.

Conclusion

This work introduced a DBS algorithm that navigates disconnected directed graphs using a priority function based on a tunable combination of in-degree and out-degree. Through this technique, fragmented topologies can be systematically investigated without relying on root-based reachability. Extensive experiments on the Cora citation dataset, Twitter mention network, and synthetic MAG showed that the algorithm can effectively adapt to structural diversity, spanning multiple components through dynamic re-initialization and identifying influential nodes with impressive efficiency.

Significant performance gains over conventional traversal methods were demonstrated by the experimental validation; DBS variants achieved discovery rates ranging from 78.5% to 94.2%, while conventional DFS and BFS approaches achieved rates of approximately 45%. The α parameter enables algorithm customization, allowing it to be tailored to the needs of specific applications. DBS-0.00 offers optimal computational efficiency at a 25% overhead, while DBS-1.00 maximizes influential node discovery at higher computational cost. The component transition analysis revealed up to 355 transitions ensuring thorough coverage across disconnected graph partitions, demonstrating the algorithm's superior ability to explore fragmented networks compared to traditional methods.

However, the performance of the method is intrinsically linked to the degree distribution of the input graph. The effectiveness of the algorithm is directly related to degree heterogeneity in the network structure, as demonstrated by the experimental results obtained across various network topologies. In networks with similar node connectivity patterns that lack degree heterogeneity, prioritization offers little advantage over conventional techniques, and the computational overhead of the algorithm might not be justified. While the fragmented Cora network required careful parameter tuning to balance computational cost and discovery efficiency, the Twitter dataset's dense connectivity structure showed optimal DBS performance with minimal overhead.

The algorithm's efficacy was demonstrated by the multi-dataset evaluation, but additional tests on real-world disconnected networks are needed to verify generalizability across various domain-specific graph structures. Despite including three different network types with broad scope, the current experimental framework covers only a portion of the intricate graph topologies found in real-world applications. Although

DOI: https://doi.org/10.70389/PJS.100247 | Premier Journal of Science 2025;15:100247

the synthetic MAG dataset provided controlled experimental conditions, real-world networks frequently display irregular patterns and evolving structures that require additional verification.

Furthermore, the priority-based selection mechanism in the current implementation processes nodes sequentially. According to the experimental timing analysis, parallelization could lead to significant performance gains, especially for large-scale networks where the computational overhead of higher α values becomes noticeable. The memory overhead analysis's linear scalability indicates that parallel processing architectures may be able to distribute priority queue management among multiple processing units.

To overcome the noted limitations, future research will explore hybrid approaches that combine DBS with traditional search methods such as BFS and shortest path heuristics, as well as graph embeddings. In graphs with low-degree homogeneity or subtle structural patterns, where pure degree-based prioritization may be insufficient, these combinations may offer better coverage and efficiency. Experimental results showing decreased efficacy in homogeneous networks suggest that adaptive switching between DBS and conventional techniques based on local network characteristics may optimize overall performance.

The framework exhibits great promise for extension to reinforcement learning or probabilistic traversal settings, where traversal history and discovered network properties could be used to dynamically modify the priority function. These adaptive methods might address the parameter selection challenges found in the experimental analysis, where optimal α values varied significantly depending on network type and application requirements. Additionally, there is considerable potential for applications in sparse and evolving information networks, where the capacity to efficiently identify influential nodes becomes increasingly important as network structure evolves.

Reproducibility

To promote transparency and reproducibility, the complete source code, parameter settings, and instructions to replicate the experiments have been deposited in a public repository at: https://github.com/Shyma25/DBS-DirectedGraphs/ tree/main This ensures that all results presented in this paper can be independently verified and extended by future researchers.

List of Abbreviations

API: Application Programming Interface
BFS: Breadth-First Search
CPU: Central Processing Unit
DBS: Degree-Based Search
DBDT: Degree-Based Directed Traversal
DBLP: Database Systems and Logic Programming
DFS: Depth-First Search
DOI: Digital Object Identifier
FPGA: Field-Programmable Gate Array
GPU: Graphics Processing Unit
HMC: Hybrid Memory Cube
HTML: Hypertext Markup Language
IEEE: Institute of Electrical and Electronics Engineers
IJACSA: International Journal of Advanced Computer Science and Applications
JSON: JavaScript Object Notation
KG: Knowledge Graph
MAG: Microsoft Academic Graph
RAM: Random Access Memory
RO: Reorder/Batch Reordering
RSN: Relation-aware Graph Neural Network
SCC: Strongly Connected Component
SIGIR: Special Interest Group on Information Retrieval
SIGMOD: Special Interest Group on Management of Data
SQL: Structured Query Language
URL: Uniform Resource Locator
XML: eXtensible Markup Language

Mathematical Notation

$G = (V, E)$: Directed graph with vertex set V and edge set E
$|V|$: Number of vertices in the graph
$|E|$: Number of edges in the graph
$\deg^-(v)$: In-degree of vertex v
$\deg^+(v)$: Out-degree of vertex v
$\deg(v)$: Total degree of vertex v, i.e., $\deg^-(v) + \deg^+(v)$
$P(v)$: Priority function for vertex v
α, β : Weighting parameters where $\alpha + \beta = 1$
V_{visited} : Set of visited vertices
L_{search} : List of vertices in traversal order
$N(v)$: Neighborhood of vertex v
$O(\cdot)$: Big-O notation for computational complexity

References

1 Kaviyarasu M, Aslam M, Afzal F, Saeed MM, Mehmood A, Gul S. The connectivity indices concept of neutrosophic graph and their application of computer network, highway system and transport network flow. Sci Rep. 2024;14:4891. https://doi.org/10.1038/s41598-024-54104-x
2 Tripathy A, Panada AC, Behera SP, Mohanty BS. Edge connectivity of a neutrosophic graph. Neutrosophic Sets Syst. 2025;81:1.
3 Lowe G. Concurrent depth-first search algorithms based on Tarjan's Algorithm. Int J Softw Tools Technol Transf. 2016;18(2):129–47. https://doi.org/10.1007/s10009-015-0382-1
4 Malewicz G, Austern MH, Bik AJ, Dehnert JC, Horn I, Leiser N, et al. Pregel: a system for large-scale graph processing. In: SIGMOD '10: proceedings of the 2010 ACM SIGMOD international conference on management of data. ACM; 2010. p. 135–46. https://doi.org/10.1145/1807167.1807184
5 Zhang J, Li J. Degree-aware hybrid graph traversal on FPGA-HMC platform. In: FPGA '18: proceedings of the 2018 ACM/SIGDA international symposium on field-programmable gate arrays. ACM; 2018. p. 229–38. https://doi.org/10.1145/3174243.3174245
6 Gonzalez JE, Low Y, Gu H, Bickson D, Guestrin C. PowerGraph: distributed graph-parallel computation on natural graphs. In: OSDI'12: proceedings of the 10th USENIX conference on operating systems design and implementation. USENIX Association; 2012. p. 17–30.
7 Kyrola A, Blelloch G, Guestrin C. GraphChi: large-scale graph computation on just a PC. In: OSDI'12: proceedings of the 10th USENIX conference on operating systems design and implementation. USENIX Association; 2012. p. 31–46.
8 Zeng W, Zhao X, Wang W, Tang J, Tan Z. Degree-aware alignment for entities in tail. In: SIGIR '20: proceedings of the 43rd international ACM SIGIR conference on research and development in information retrieval. ACM; 2020. p. 811–20. https://doi.org/10.1145/3397271.3401161
9 Basak A, Qu Z, Lin J, Alameldeen AR, Chishti Z, Ding Y, et al. Improving streaming graph processing performance using input

knowledge. In: MICRO '21: MICRO-54: 54th annual IEEE/ACM international symposium on microarchitecture. ACM; 2021. p. 1036–50. https://doi.org/10.1145/3466752.3480096
10 Page L, Brin S, Motwani R, Winograd T. The PageRank citation ranking: bringing order to the web. Stanford InfoLab Technical Report 1999-66. Stanford University; 1999.
11 Freeman LC. A set of measures of centrality based on betweenness. Sociometry. 1977;40(1):35–41. https://doi.org/10.2307/3033543
12 Aldawish R, Kurdi H. A modified degree discount Heuristic for influence maximization in social networks. Proc Comput Sci. 2020;170:311–6. http://doi.org/10.1016/j.procs.2020.03.045
13 Shyma PV, Sanil Shanker KP. Degree based search: a novel graph traversal algorithm using degree based priority queues. Int J Adv Comput Sci Appl. 2024;15:1366–71. http://doi.org/10.14569/IJACSA.2024.01507132
14 Tarjan R. Depth-first search and linear graph algorithms. SIAM J Comput. 1972;1(2):146–60. https://doi.org/10.1137/0201010
15 Subbarayudu B, Gayatri LL, Nidhi PS, Ramesh P, Reddy RG, Kishor Kumar Reddy C. Comparative analysis on sorting and searching algorithms. Int J Civil Eng Technol. 2017;8(8):955–78.
16 Baidari I, Hanagawadimath A. Traversing directed cyclic and acyclic graphs using modified BFS algorithm. In: Proceedings of 2014 science and information conference. SAI; 2014. p. 175–81.
17 Allender E, Chauhan A, Datta S. Depth-first search in directed planar graphs, revisited. Acta Informatica. 2022;59:289–319. https://doi.org/10.1007/s00236-022-00425-1
18 Prolubnikov A. Finding connected components of a graph using traversals associated with iterative methods for solving systems of linear equations; 2024. arXiv:2407.10790. https://doi.org/10.48550/arXiv.2407.10790