Geospatial Business Analytics using Dijkstra's Algorithm and BFS on OpenStreetMap Road Networks

Leonardus Brain Fatolosja - 13524146
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jalan Ganesha 10 Bandung
E-mail: leonardusbrain0@gmail.com , 13524146@std.stei.itb.ac.id

***Abstract*— Location is one of the most important aspects in establishing a successful business. Poor location selection and intense market competition may reduce business profitability and long-term sustainability. This paper proposes a geospatial business analytics approach using Dijkstra's algorithm and Breadth-First Search (BFS) on OpenStreetMap road networks.**

Keywords*— *Dijkstra Algorithm, Business Analytics, Breadth-First Search

I. Introduction

Location constitutes one of the most important considerations in establishing a business. Numerous businesses have failed as a direct result of poor location selection. Beyond poor location selection, intense market competition represents another significant factor contributing to business failure. When a business is established in an unstrategic location coupled with a highly competitive market environment, elevated operational and transportation costs substantially erode profit margins. Therefore, the probability of business survival diminishes over time. This phenomenon demonstrates that location selection and competitive market conditions in a given area are critical factors that warrant careful analysis prior to business establishment.

Dijkstra's algorithm can be applied to support business location analysis by modelling road networks as weighted graphs. In this study, OpenStreetMap is used as the primary source of road network and geospatial data. The road network obtained from OpenStreetMap is transformed into a graph structure, where Dijkstra's algorithm is used to calculate the shortest reachable distance from a candidate business location to surrounding roads, residential areas, and competing businesses.

In addition to Dijkstra's algorithm, Breadth-First Search (BFS) is used as a baseline comparison. BFS explores graph nodes level by level without considering edge weights, meaning that every road segment is treated as having equal cost. Although BFS is useful for unweighted graph traversal, it is less realistic for road network analysis because road segments have different lengths. By comparing BFS and Dijkstra's algorithm, this research demonstrates the importance of weighted shortest-path analysis in geospatial business analytics. Dijkstra's algorithm is expected to produce a more accurate representation of real-world accessibility, while BFS provides a simpler comparison model.

Therefore, this paper proposes a geospatial business analytics approach using Dijkstra's algorithm and BFS on OpenStreetMap road networks. The objective is to analyze the potential of a business location by evaluating accessibility, surrounding competitors, residential density, and road network coverage. The expected result is a system that can assist business owners or analysts in making more informed location decisions before establishing a business.

II. Theory

A. Graph

Formally, a graph can be written as:

$$G = (V, E)$$

where V is set of nodes and E is set of edges.

In an undirected graph, edges have no orientation. In a directed graph, edges are ordered pairs. For a weighted graph, a weight function ($w: E \to \mathbb{R}$) is employed. Consequently, the shortest path problem becomes the search for a path with minimum total weight rather than merely minimum edge count. If a node is unreachable from the source, its distance is considered undefined or infinite (∞), depending on the formulation employed.

Based on edge structure, a simple graph contains neither loop nor multiple edge. A multigraph permits multiple edge, meanwhile a pseudograph permits loop. Graphs are classified by orientation into directed and undirected graph. Furthermore, graphs are classified by weight into weighted and unweighted. The choice of model is far from merely cosmetic.

Standard Breadth First Search computes shortest paths in terms of edge count on unweighted graph, meanwhile Dijkstra's Algorithm is applicable to weighted graph with non-negative weight. For shortest path problems, a useful notation is $\delta(s, v)$ which denotes the shortest path from source s to node v. In the single-source shortest paths problem, one is given a single source s and must compute distances to all other reachable nodes. On an unweighted graph, "path length" refers to the number of edges. On a weighted graph, "path length" refers to the sum of weights on the edges comprising the path. Notably, any subpath of a shortest path is itself a shortest path with respect

to its endpoints. This optimal substructure property is precisely what enables the correctness of both BFS and Dijkstra's algorithm to be formally proven.

B. Dijkstra Algorithm

Dijkstra solve single source shortest path of non-negative weighted graph. Its input consists of a graph ($G = (V, E, w)$), where (V) represents the set of nodes, (E) represents the set of edges, and (w) is a weight function that assigns a non-negative cost to each edge. The algorithm also requires a source node (s), from which the shortest distance to all other nodes is computed. The output typically consists of two main data structures: dist[v], which stores the shortest known distance from the source node to each node (v), and parent[v], which stores the predecessor of each node in the shortest-path tree. The parent structure enables reconstruction of the actual shortest path from the source to a target node.

The algorithm begins by initializing the distance of the source node to zero and the distance of every other node to infinity. This represents the assumption that, before the algorithm starts, no path from the source to other nodes has been discovered. All nodes are initially marked as unvisited or unfinalized. In most efficient implementations, a priority queue is used to repeatedly select the unfinalized node with the smallest tentative distance.

At each iteration, Dijkstra's algorithm selects the unfinalized node (u) with the minimum tentative distance. This node is then finalized, meaning that the current value of dist[u] is guaranteed to be the shortest possible distance from the source to (u). After selecting (u), the algorithm relaxes all outgoing edges from (u). Edge relaxation is the process of checking whether a shorter path to a neighboring node (v) can be obtained through (u). If the value of dist[u] + w(u, v) is smaller than the current value of dist[v], then dist[v] is updated and parent[v] is set to (u).

```
Input:
        G = (V, E, w)
        s = source vertex
Output:
        dist[v]
        parent[v]
Begin
        for each vertex v in V do
        dist[v] ← ∞
        parent[v] ← NIL
        visited[v] ← false
        end for
        dist[s] ← 0
        Q ← priority queue ordered by dist[v]

        while Q is not empty do
        u ← vertex in Q with the smallest dist[u]
        remove u from Q
        if visited[u] = true then
        continue
        end if
        visited[u] ← true

        for each edge (u, v) in E do
        if visited[v] = false then
        new_distance ← dist[u] + w(u, v)
        if new_distance < dist[v] then
        dist[v] ← new_distance
        parent[v] ← u
        update priority of v in Q
        end if
        end if
        end for
        end while
        return dist, parent
End
```

C. BFS

Breadth-first search (BFS) is a graph traversal algorithm that visits nodes in order of their distance layers from the source. For unweighted graphs, BFS computes the minimum distance in terms of edge count from a source node s to every reachable node. On directed graphs, only out-neighbors are considered. on undirected graphs, all neighbors are examined. BFS furthermore serves as the foundation for constructing BFS trees, identifying connected components, testing bipartiteness, and numerous other algorithmic applications.

If level 0 contains the source node s, then level 1 contains all unvisited neighbors of s, level 2 contains all nodes reachable via exactly two edges but not fewer, and so on. A queue data structure enforces the first-in, first-out (FIFO) ordering that aligns with layer-by-layer exploration. Since all edges are treated as having uniform weight, a node discovered first is indeed discovered via a path with minimum edge count.

BFS begins by placing the source node (s) into a queue and marking it as visited. The algorithm then repeatedly removes the front node from the queue and examines all of its adjacent vertices. If an adjacent vertex has not been visited, it is marked as visited, its distance is updated as one more than the distance of the current node, and it is inserted into the queue. This process continues until the queue becomes empty or until a certain search limit is reached.

```
Input:
        G = (V, E)
        s = source vertex
Output:
        dist[v]
        parent[v]
Begin
        for each vertex v in V do
          dist[v] ← ∞
          parent[v] ← NIL
          visited[v] ← false
        end for
        visited[s] ← true
        dist[s] ← 0
        Q ← empty queue
        enqueue(Q, s)
        while Q is not empty do
          u ← dequeue(Q)

          for each neighbor v of u do
            if visited[v] = false then
              visited[v] ← true
              dist[v] ← dist[u] + 1
              parent[v] ← u
              enqueue(Q, v)
            end if
          end for
        end while
        return dist, parent
End
```

The main strength of BFS is its simplicity and optimality on unweighted graphs. Because nodes are expanded in increasing order of edge count, BFS guarantees that the first time a node is discovered, it is reached through the minimum number of edges from the source.

III. IMPLEMENTATION

This section describes the implementation design of the geospatial business analytics system. The system uses

OpenStreetMap road network data to model the surrounding area of a candidate business location as a graph. Dijkstra's algorithm is used as the main method to compute a realistic service area based on accumulated road distance, while Breadth-First Search (BFS) is used as a baseline method to show how the result differs when road segment weights are ignored. The implementation focuses on analyzing a candidate business location by measuring accessibility, nearby competitors, residential building distribution as a customer proxy, and road network density.

A. Preparation Data

The system is designed as a geospatial analysis program that receives a candidate business location as input. The location can be represented using latitude and longitude coordinates or obtained from a selected point on a map. The system then retrieves the surrounding road network and geospatial features from OpenStreetMap. OpenStreetMap is used because it provides open geospatial data, including roads, buildings, and points of interest such as restaurants, cafes, shops, schools, and other public facilities.

In relation to the algorithms, OSMnx provides the road network graph, NetworkX runs Dijkstra and BFS on that graph, and GeoPandas processes the spatial output generated from both algorithms. Dijkstra's algorithm depends on road segment length as edge weight, while BFS only depends on graph connectivity. Therefore, both algorithms operate on the same road network graph but produce different interpretations of reachability.

B. Graph Construction

The first step in the implementation is constructing a graph from the OpenStreetMap road network. The selected study area is downloaded from OpenStreetMap using OSMnx. The road network is represented as a graph ($G = (V, E)$), where (V) represents road nodes and (E) represents road segments. Road nodes may represent intersections, road endpoints, or simplified road geometry points, while road segments represent the connections between those nodes.

Each edge in the graph contains several attributes, with road length being the most important attribute for this research. The road length is used as the edge weight for Dijkstra's algorithm. If the road is one-way, the graph represents it as a directed edge. If the road allows two-way movement, the graph can contain two directed edges in opposite directions. This representation is suitable for road network analysis because it reflects the structure and direction of movement in real road systems.

The candidate business location is not directly used as a graph node. Instead, its geographic coordinate is mapped to the nearest road node in the OpenStreetMap graph. This process is necessary because graph algorithms operate on nodes and edges, while the business location is originally just a coordinate point. After the nearest road node is identified, that node becomes the source node for both Dijkstra's algorithm and BFS.

C. Dijkstra's Algorithm Usage

Dijkstra's algorithm is used as the main algorithm for computing the service area of the candidate business location. The service area represents all road nodes that can be reached from the candidate location within a specified road-network distance, such as five kilometers. In this implementation, before Dijkstra's algorithm is executed, the candidate business location is mapped to the nearest road node in the graph. This is necessary because the input location is originally a latitude-longitude coordinate, while graph algorithms operate on nodes and edges. After the nearest node is found, it becomes the source node for Dijkstra's algorithm. A distance threshold is used to limit the search. All nodes whose shortest distance from the source is less than or equal to the threshold are included in the Dijkstra-based service area.

The flow of Dijkstra's Algorithm usage in this system is as follows:

- The program receives a candidate business location as latitude and longtitude coordinates and downloads the surrounding road networks from OpenStreetMap.
- The road network is converted into weighted graph. Road intersections or road points become graph nodes, road segments become graph edges, and each edges contain a weight based on road segment's length.
- The candidate business location is mapped to the nearest road node (graph node). The node will be used as the source node.
- Dijkstra's Algorithm is executed from the source node. It calculates the shortest accumulated road distance to surrounding nodes.
- Node with distance less than or equal to threshold are selected. The selected nodes form the reachable service area.
- The service area is converted into a spatial area (polygon) and the information inside the area is extracted, such as business points, competitors, residential buildings, and road segments.
- The extracted information is used to calculate business analytics metrics.

The dijkstra's algorithm itself operates as follows: it starts from the source node and expands through the road network by always selecting the unvisited node with the smallest accumulated distance. For each selected node, the algorithm relaxes its outgoing edges and updates the shortest known distance to neighboring nodes.

The main advantage of using Dijkstra's algorithm is that it considers actual road distance. Therefore, the resulting service area is more realistic than a simple circular radius or unweighted traversal. In the context of business analytics, this allows the system to estimate which competitors, residential buildings, and road segments are realistically reachable from the candidate business location through the road network.

D. Breadth-First Search (BFS) Usage

Breadth-First Search is used as a baseline algorithm for comparison. Unlike Dijkstra's algorithm, BFS does not consider road length or any other edge weight. BFS explores the graph level by level based on the number of edges from the source node. The source node is placed at level zero, its direct neighbors are placed at level one, the neighbors of those nodes are placed at level two, and so on.

The flow of BFS usage in this system is as follows:

- The program receives the same candidate business location used by Dijkstra and The same OpenStreetMap road network graph.
- The source node is inserted into a queue. BFS starts exploring neighboring nodes level by level.
- Each road segment is treated as one step and the algorithm does not consider the actual length of each road segment.
- Nodes at level (n) are visited after all level (n-1) nodes are processed. The process continues until the queue is empty or the maximum traversal level is reached.
- All visited nodes within the level threshold are collected. The collected nodes form the BFS-based service area.
- Business points, competitors, residential buildings, and roads inside this area are extracted and The result is compared with the Dijkstra-based service area.

In BFS, every road segment is treated as having equal cost. This means that a short alley and a long main road are counted equally as one edge. As a result, BFS can identify nodes reachable within a certain number of graph steps, but it cannot accurately represent real world travel distance.

The purpose of BFS in this research is not to replace Dijkstra's algorithm, but to demonstrate the limitation of unweighted traversal in road network analysis. By comparing BFS and Dijkstra, the system can show how ignoring road length affects the resulting service area and business analytics output. BFS is expected to be faster and simpler, but less realistic for analyzing actual accessibility.

E. Business Analytics and Evaluation Metrics

After the service area is generated, the system performs business analytics using geospatial features located within the reachable area. The first metric is competitor count, which measures the number of businesses of the same category within the service area. The second metric is residential building count, which is used as a proxy for potential customers. Since OpenStreetMap does not provide actual customer population or income data, residential buildings such as houses, apartments, and residential complexes are used as an approximate indicator of potential demand.

The third metric is road density, which is calculated from the total length of road segments inside the service area divided by the area size. The fourth metric is accessibility, which is derived from the number of reachable nodes, reachable road length, and the structure of the surrounding road network. These metrics are then combined into a business potential score. A possible scoring model is:

$$Business\_Potential_{score} = 0.35 \times CustomerScore + 0.25 \times AccessibilityScore + 0.20 \times RoadDensityScore + 0.20 \times CompetitionScore$$

For the notes, the scoring model may not be accurate and can be adjusted depending on the type of business or the type of people (economic class) in that area.

The evaluation compares the results of Dijkstra's algorithm and BFS using both algorithmic and business-oriented metrics. From the algorithmic perspective, the system evaluates execution time, number of visited nodes, number of reachable nodes, and service area size. These metrics are used to compare the computational behavior of both algorithms.

From the business analysis perspective, the system compares the number of competitors, number of residential buildings, total road length, road density, and business potential score obtained from each algorithm's service area. If Dijkstra and BFS produce different reachable areas, the business analytics results may also differ. This comparison is important because it shows how algorithm selection affects business interpretation.

F. Output and Expected Results

The system produces several outputs. The first output is a map visualization showing the candidate business location, Dijkstra-based service area, BFS-based service area, nearby competitors, residential buildings, and relevant road network segments. The second output is a summary table containing the number of competitors, number of residential buildings, road density, accessibility score, and final business potential score. The third output is a comparison between Dijkstra and BFS, showing how the two algorithms differ in terms of coverage, runtime, and business interpretation.

The expected algorithmic result is that BFS will generally be simpler and faster because it only uses a queue and ignores edge weights. Dijkstra's algorithm is expected to require more computation because it uses a priority queue and considers accumulated road distance. However, Dijkstra is expected to produce more realistic results because road segment length is included in the calculation.

In the context of business location analysis, the Dijkstra-based result is expected to be more suitable for decision-making. It can show which competitors and residential buildings are realistically reachable within a selected distance threshold. Meanwhile, BFS serves as a useful baseline to

demonstrate why weighted graph algorithms are more appropriate for geospatial business analytics on road networks.

IV. RESULT AND ANALYSIS

This chapter capture the results obtained from the implementation of the geospatial business analytics program. The analysis focuses on comparing the output of Dijkstra's algorithm and Breadth-First Search (BFS) when applied to OpenStreetMap road networks. The comparison is conducted to evaluate how weighted and unweighted graph traversal affect business location analysis. The results include the generated service area, reachable road network, number of competitors, residential building count, road density, execution time, and business potential interpretation.

A. Output

The program produced several outputs. The first output is a map visualization showing the candidate business location, the Dijkstra-based service area, the BFS-based service area, nearby competitors, residential buildings, and reachable road segments. The second output is a tabular summary containing the analytical metrics obtained from each algorithm. The third output is a comparison between Dijkstra and BFS in terms of reachability, business interpretation, and computational performance.

The generated map provides a visual comparison between the service areas produced by Dijkstra and BFS. The Dijkstra-based service area represents roads reachable within the selected road-network distance threshold, while the BFS-based service area represents roads reachable within a certain traversal depth. This output allows visual inspection of how both algorithms behave when applied to the same candidate business location.

For example, The case study area used in this research is Bandung. This area was selected because it contains various types of roads, residential zones, and business points of interest. The candidate business location was represented using latitude and longitude coordinates and then mapped to the nearest road node in the OpenStreetMap road network. The business category is Pharmacy.

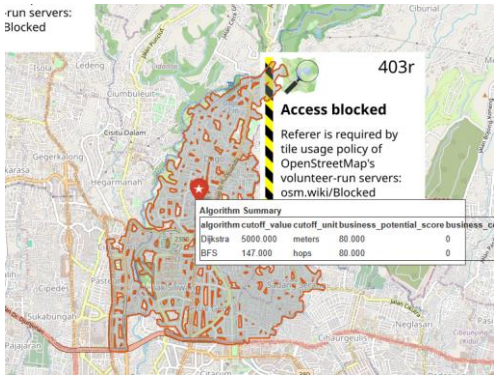

Figure 1. Generated Map of Dijkstra and BFS area

B. Dijkstra Result

The Dijkstra-based result represents realistic accessibility because each road segment contributes according to its actual length. Road nodes were included in the service area only if their shortest accumulated road distance from the candidate location was less than or equal to the selected threshold. Therefore, the generated service area follows the structure of the road network rather than forming a simple circular buffer.

The result of Dijkstra's algorithm is summarized below.

Metric	Value
Reachable Nodes	18591
Reachable Edges	42255
Candidate Customers	5970
Road Length (km)	2931.681159201908
Road Density	39.60631977917862
Competitors	0

These values were then used to compute the business analytics metrics. A higher number of residential buildings indicates a stronger potential customer proxy, while a higher number of competitors indicates stronger market competition.

The Dijkstra result is considered the primary result of this research because it uses road length as the edge weight. This makes the output more suitable for analyzing real business accessibility. In real-world conditions, customers and suppliers travel through roads with different lengths, so a weighted shortest-path algorithm provides a more accurate representation than an unweighted traversal method.

C. BFS Result

Breadth-First Search was executed on the same OpenStreetMap road network graph. However, unlike Dijkstra's algorithm, BFS did not use road length as edge weight. Every road segment was treated as having equal traversal cost. The source node was the same nearest road node used in the Dijkstra analysis.

BFS explored the road network level by level. Nodes directly connected to the source were placed in the first level, nodes reachable through two edges were placed in the second level, and so on. All nodes reached within this depth were included in the BFS-based service area.

The BFS-based result is summarized below.

Metric	Value
Reachable Nodes	25613
Reachable Edges	59118
Candidate Customers	8852
Road Length (km)	4010.2877875616114
Road Density	40.88914393233394
Competitors	0

The BFS result shows how far the graph can be explored when all roads are treated equally. This method is useful as a baseline because it demonstrates the behavior of unweighted graph traversal. However, because BFS ignores road length, its service area may not represent actual travel distance. A long road segment and a short road segment are both counted as one step, even though their real-world distance can be very different.

Therefore, the BFS result is not used as the main basis for business location decision-making. Instead, it is used to compare against Dijkstra's algorithm and to show why weighted graph analysis is more appropriate for OpenStreetMap road networks.

D. Comparison Between Dijkstra and BFS

The comparison between Dijkstra and BFS shows the effect of using edge weights in road network analysis. Dijkstra calculates reachability based on accumulated road length, while BFS calculates reachability based on the number of edges traversed. Because road segments in real road networks have different lengths, the two algorithms produce different service areas and different business analytics results.

Metric	Dijkstra	BFS
Reachable Nodes	18591	25613
Reachable Edges	42255	59118
Candidate Customers	5970	8852
Road Length (km)	2931.681159201908	4010.2877875616114
Road Density	39.60631977917862	40.88914393233394
Competitors	0	0

From the table earlier, Dijkstra is able to produce a more realistic service area because it considers actual road distance. BFS may produce a service area that is either too broad or spatially distorted because it ignores edge length. For example, BFS may reach a far location through only a few long road segments, while Dijkstra may exclude that location if the accumulated road distance exceeds the threshold. Therefore, BFS result produce more reachable nodes, reachable edges, and candidate customers.

In terms of performance, BFS is generally faster because it only uses a queue and visits nodes based on traversal level. Its time complexity is $(O(|V| + |E|))$. Dijkstra is computationally more expensive because it uses edge weights and usually requires a priority queue. With a binary heap implementation, its time complexity is $(O((|V| + |E|) * log\ |V|))$. However, the additional computation in Dijkstra is justified because the result better represents real road-network accessibility.

E. Analysis of Program Success and Performance

The implementation can be considered successful because the system is able to perform the complete workflow from data acquisition to business interpretation. The system was able to retrieve the road network from OpenStreetMap, construct the road network graph, map the candidate business location to the nearest graph node, execute Dijkstra and BFS, generate service areas, extract business and residential data, compute analytics metrics, and visualize the result.

From an algorithmic perspective, the system is successful because both Dijkstra and BFS can produce reachable service areas from the same source node. The system is also successful if the resulting service areas can be used to extract competitors, residential buildings, and road network metrics.

From a performance perspective, BFS run faster than Dijkstra because it does not process edge weights. However, BFS has lower analytical quality for this problem because it ignores actual road distance. Dijkstra requires more computation but produces a more meaningful result for business location analysis. Therefore, the performance trade-off is acceptable because the objective of this research is not only speed, but also realistic accessibility analysis.

The result also shows that algorithm selection affects business interpretation. If the service area is generated using BFS, the number of competitors and residential buildings may differ from the Dijkstra result. This difference occurs because the two algorithms define reachability differently. Dijkstra defines reachability based on accumulated road distance, while BFS defines reachability based on edge count. As a result, the business potential score may also differ between both methods.

The only limitation with this program is the absence of competitors retrieved from OpenStreetMap. This limitation is not caused by the graph algorithm itself, but by limitation that OSM has. OSM is an open source map and the availability of points of interest depends heavily on the number of contributors and the mapping activity in a particular region. Therefore, the information of existing business in Indonesia is also limited. The alternative way to solve this problem is using Google Maps, which generally provides more complete and frequently updated business information. However, this alternative introduces several trade-offs. Google Maps data is not open-source, requires API access, and may involve usage costs depending on the number of requests. Therefore, this research uses OpenStreetMap because it is open source, free to access, and suitable for an academic prototype.

V. CONCLUSIONS AND RECOMMENDATIONS

A. Conclusions

In conclusion, this research shows that graph algorithms can be used to support geospatial business location analysis using OpenStreetMap road network data. Dijkstra's algorithm provides a more realistic service area because it considers actual road segment length as edge weight, while BFS is useful as a comparison method to demonstrate the limitations of unweighted graph traversal. Based on the implementation results, the system successfully generated service areas, measured reachable road networks, counted residential buildings as customer proxies, calculated road density, and compared algorithm performance. Although competitor data from OpenStreetMap was limited in the case study, the system still demonstrates that road-network-based analysis can provide more meaningful business accessibility insights than simple radius-based analysis. Therefore, Dijkstra's algorithm is more suitable as the main method for business location analysis, while BFS can be used as a baseline to understand the effect of ignoring road distance.

B. Recommendations

- Future research should use a more complete business data source to improve competitor analysis. OpenStreetMap is suitable for an academic prototype, but its business data may be incomplete in some regions. Another alternative choice is Google Maps.
- The scoring model should be adjusted based on the type of business being analyzed. Different businesses may require different weights for customer density, accessibility, road density, and competition.
- Future implementation can include travel time as an edge weight instead of only road length. This would make the service area more realistic because travel time may be affected by road type and speed.

VIDEO LINK AT YOUTUBE

https://youtu.be/7ApqwyDqNlA

SOURCE CODE AT GITHUB

https://github.com/leains/STIMA_MAKALAH

ACKNOWLEDGMENT

The author is deeply thank you to God, Lecturer, and others that support author to make this paper.

REFERENCES

[1] Algorithms for Competitive Programming, "Dijkstra - finding shortest paths from given vertex," CP-Algorithms, Sep. 24, 2023. [Online]. Available: https://cp-algorithms.com/graph/dijkstra.html. [Accessed: Jun. 17, 2026].

[2] GeeksforGeeks, "Dijkstra's Algorithm," GeeksforGeeks, Jan. 21, 2026. [Online]. Available: https://www.geeksforgeeks.org/dsa/dijkstras-shortest-path-algorithm-greedy-algo-7/. [Accessed: Jun. 17, 2026].

[3] R. Munir, "Graf (Bag. 1)," Bahan Kuliah IF2120 Matematika Diskrit, Program Studi Teknik Informatika, STEI-ITB, 2020. [Online]. Available: https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2020-2021/Graf-2020-Bagian1.pdf. [Accessed: Jun. 17, 2026].

[4] Stanford CS161, "Lecture 11: Dijkstra and Bellman-Ford," CS161: Design and Analysis of Algorithms, Stanford University, Winter 2022. [Online]. Available: https://stanford-cs161.github.io/winter2022/assets/files/lecture11-notes.pdf. [Accessed: Jun. 17, 2026].

[5] E. Demaine and S. Devadas, "Lecture 13: Breadth-First Search (BFS)," MIT OpenCourseWare, 6.006 Introduction to Algorithms, Massachusetts Institute of Technology, Fall 2011. [Online]. Available: https://ocw.mit.edu/courses/6-006-introduction-to-algorithms-fall-2011/resources/lecture-13-breadth-first-search-bfs/. [Accessed: Jun. 18, 2026].

[6] [6] R. Mutianisa and R. R. Cahyani, "Pemilihan Lokasi Usaha Terhadap Kesuksesan Usaha," JUKERDI: Jurnal Kewirausahaan Cerdas dan Digital, vol. 1, no. 2, pp. 10–17, Apr. 2024, doi: 10.61132/jukerdi.v1i2.71. [Online]. Available: https://ejournal.arimbi.or.id/index.php/JUKERDI/article/download/71/76/278. [Accessed: Jun. 17, 2026].

[7] K. Thevrajah, "Barriers of Failure of Small Business in Jaffna District: Conceptual Analysis," European Journal of Business and Management, vol. 7, no. 7, pp. 400–404, 2015. [Online]. Available: https://iiste.org/Journals/index.php/EJBM/article/viewFile/20550/21495. [Accessed: Jun. 17, 2026].

[8] OpenStreetMap Foundation, "OpenStreetMap: Copyright and License," OpenStreetMap. [Online]. Available: https://www.openstreetmap.org/copyright. [Accessed: Jun. 18, 2026].

[9] NetworkX Developers, "dijkstra_path," NetworkX 3.6.1 documentation. [Online]. Available: https://networkx.org/documentation/stable/reference/algorithms/generated/networkx.algorithms.shortest_paths.weighted.dijkstra_path.html. [Accessed: Jun. 18, 2026].

PERNYATAAN

Dengan ini saya menyatakan bahwa makalah yang saya tulis ini adalah tulisan saya sendiri, bukan saduran, atau terjemahan dari makalah orang lain, dan bukan plagiasi.

Bandung, 19 Juni 2026

Leonardus Brain Fatolosja
13524146