# Applying Slime Mold's Behaviour as a Heuristic in A* Algorithm Pathfinding for Jakarta's Light Rail Transit (LRT) Network

Nicholas Andhika Lucas - 13523014
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jalan Ganesha 10 Bandung
E-mail: realandhikalucas@gmail.com, 13523014@std.stei.itb.ac.id

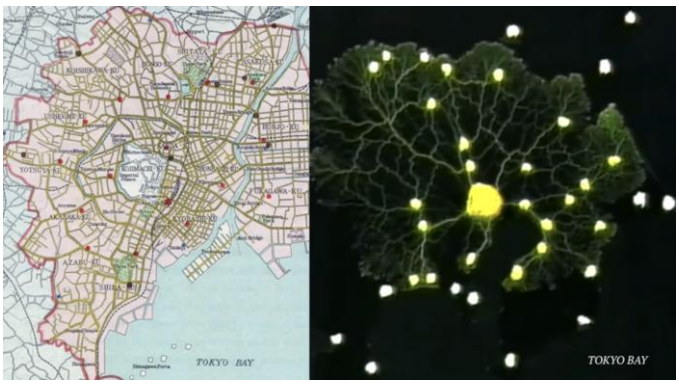
***Abstract*—Transportation network optimization is a critical aspect of urban planning aimed at enhancing efficiency by minimizing travel time and operational costs. This research explores the application of a bio-inspired heuristic, namely from the behavior of the slime mold *Physarum polycephalum in* its process of foraging for food. The paper aims to analyse and compare the effects of implement this additional heuristic for a deterministic path-finding algorithm like A* Algorithm, conducted on a modified Jakarta's Light Rail Transit (LRT) network map. The proposed changes were adding a weighted value mimicking nutritional value that plays a factor for slime mold to find food sources. The results demonstrate that the modified algorithm did not alter the chosen path when tested using the scaling factor of 10, but increased the number of explored nodes. This shows that the added heuristic plays a part in the algorithm's decision making which may be more suitable for real life transport system scenarios.**

***Keywords—pathfinding; Jakarta LRT; A* Algorithm; urban planning; slime mold; bio-inspired heuristic***

## I. INTRODUCTION

Transportation network optimization is an important aspect of urban planning. One way to improve the efficiency of transportation networks is to reduce the distance between cities or stations. This can be achieved by applying a path-finding algorithm that identifies the shortest distance between points. This will, in turn, minimize travel time and operational costs.

In 2010, a group of scientists conducted an experiment with slime mold. Despite their seemingly primitive appearance, slime molds have evolved for millions of years, optimizing their natural behavior to forage for nutrients, even without a brain. So, the group of scientists arranged the slime mold and a few pieces of food sources in a way that replicates Japan's then capital subway system in Tokyo. The slime was placed at the central point, representing Tokyo's subway station, while other food sources were placed to represent nodes of Japan's major cities nearby Tokyo. The aim of the experiment simply was to capture what the resulting behavior would be when the slime mold forages for nearby food sources. The resulting network of the slime molds ended up being very similar to the actual railroad network that connects the cities located near Tokyo. This experiment brought attention to the phenomenon now called *biologically inspired adaptive network design*, demonstrated by the efficiency of natural network formation by biological species. The experiment gave insight into how the biologically inspired models may optimize and improve human-designed transport systems such as subway networks. [1].




Picture 1. Result of the slime mold experiment conducted compared to Tokyo's actual subway network. Source: [5]

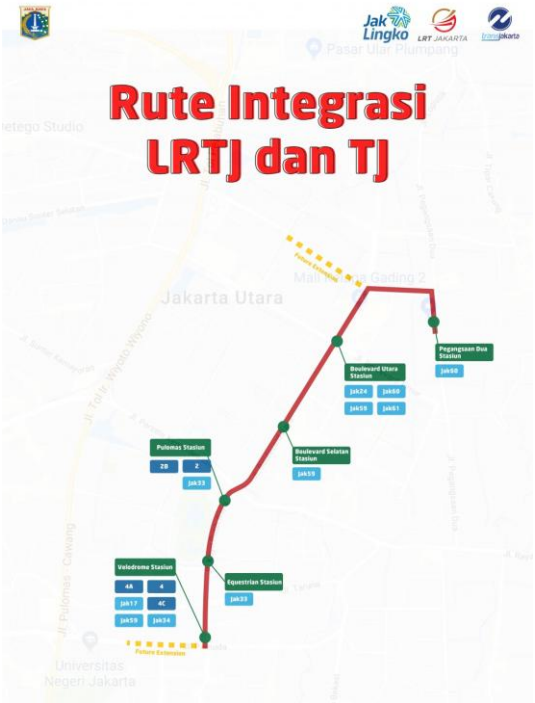
The slime mold species used in the experiment mentioned before is *physarum polycephalum*. It is a single-celled amoeboid organism that grows as a system of veins, notably with its greenish-yellow color (as shown in the picture above). These veins are used to transfer nutrients throughout the entire organism. As quoted from the experiment mentioned before:

> *"In its vegetative phase, the slime mold Physarum polycephalum "slimes" its way through the world seeking food. As it explores, it links previously found food sources with tubular structures." [1]*

Slime molds find food sources by first foraging broadly over an area, searching for the nearest food source. When a food source is detected, it will slowly refine the tubular

network, or in other words optimize the vertex linked to the food source. This process that defines how slime molds expand and forage food sources, while seemingly hard to recreate, has been recreated or imitated using computer models by multiple authors.

The Jakarta LRT is a light rapid transit system, one of the integrated public transportations system available for citizens in Jakarta. It only consist of the South Line, but is planned to expand up to the Northern Line. The only operational line right now, Phase 1, consists of 6 stations that stretch 5.8 km from North to East Jakarta, which are Pegangsaan Dua, Boulevard Utara, Boulevard Selatan, Pulomas, Equestarian, and Velodrome [6].




Picture 2. Jakarta LRT Route. Source: Adapted from [7]

Inspired by the slime mold experiment, the author aims to apply some of the insights from the experiment to further improve human-designed transport systems. In this context, the author aims to implement some of the behaviors showcased by slime mold when foraging for food sources as a heuristic for regular deterministic path-finding algorithms. The path-finding algorithm uses the A* algorithm, one of the most effective deterministic path-finding algorithms for all scenarios, compared to other algorithms such as Greedy Best First Search and Uniform Cost Search. The author will implement and analyze this technique for the Light Rapid Transit (LRT) Route in Jakarta which spans 6 stations from North Jakarta to East Jakarta. After implementing the algorithm, the research will discuss the effect of the adding the bio-inspired heuristic, inspired by the slime mold, to the existing A* algorithm, comparing the difference between the two.

## II. THEORETICAL FOUNDATION

### A. *A* Algorithm*

The A* (pronounced "A-star") algorithm is a widely used pathfinding and graph traversal algorithm, renowned for its performance and accuracy in finding the shortest path between two points, or nodes. As an informed search algorithm, it uses information about the distance to the goal to guide its search process, making it more efficient than uninformed methods. A* strategically avoids expanding paths that are already determined to be high-cost, thereby saving computational resources. The core of the algorithm is its evaluation function, f(n), which determines the priority of nodes to be explored. This function is defined as the sum of two components:

$$f(n) = g(n) + h(n)$$

In the equation, g(n) represents the actual, known cost of the path accumulated from the starting node to the current node n, while h(n) is a heuristic function that estimates the cost of the cheapest path from n to the destination. Therefore, f(n) provides an estimated total cost for the lowest-cost path from the start to the goal that passes through node n.

To manage the search process, the A* algorithm typically utilizes a priority queue. This data structure stores discovered but unexplored nodes, ordering them based on the lowest f(n) value. By always selecting the node with the smallest f(n) from the queue, A* prioritizes the most promising paths. A critical requirement for the A* algorithm to guarantee an optimal (shortest) path is that its heuristic function, h(n), must be admissible. A heuristic is considered admissible if, for every node n, its estimate h(n) is less than or equal to the true cost to reach the goal, h*(n), meaning it is always optimistic and never overestimates the cost. In the context of route planning on a map, the straight-line distance (Euclidean distance) between the current node and the destination is a common and effective admissible heuristic, as the shortest possible path between two points is a straight line.

Furthermore, the A* algorithm can be enhanced by incorporating a potential function into the priority key of each node in the queue. This potential function for a node v is an estimate of the distance to reach the target from v, which allows A* to more effectively prioritize nodes that are likely closer to the destination. For more complex applications, such as in bidirectional search or time-dependent graphs where edge costs can change, the selection of this potential function is crucial to ensure it provides a valid estimate despite varying network conditions. [8]

### B. *Bio-Inspired Heuristics*

One of the experiments used as reference in this research, authored by Wenxiao Zhang [2], modeled the slime mold using a bio-inspired heuristic. One of the characteristics displayed is using pheromones as a sense of direction or can also be described as a heuristic. It mimics how slime molds work by utilizing chemical signals, similar to pheromones. Food sources that have a higher nutrient value are given higher priority in the foraging process. Other characteristics include prioritizing paths that efficiently connect multiple

stations over other paths. This can be implemented in the form of adding weighted values for nodes in the transport system. The idea is that a node -- or nutrient source, in this analogy -- may be prioritized first if its value is big enough.

## III. IMPLEMENTATION AND EXPERIMENT

### *A. Implementation on Python*

The Jakarta LRT Network with 6 stations will be modeled using a weighted graph structure, where the nodes represent stations and edges represent track segments or direct routes to stations. The implementation will use Python as its programming language.

Extracting the nodes from the actual LRT stations is done by locating the stations based on its longitude and latitude position. This can be done by specifically retrieving the value from Google Maps or other third-party sources. The graph is then modeled by creating a list of connections between nodes of the stations. For reference, a snippet of the initial data structure is as such:

```
"PEG": {
        "id": "PEG", "name": "Pegangsaan Dua",
        "coordinates": (-6.1519, 106.8906), "line": "Line 1",
        "connections": ["BPB"]
    },
    "BPB": {
        "id": "BPB", "name": "Boulevard Utara",
        "coordinates": (-6.1580, 106.8875), "line": "Line 1",
        "connections": ["PEG", "BVN"]
    },
    "BVN": {
        "id": "BVN", "name": "Boulevard Selatan",
        "coordinates": (-6.1650, 106.8845), "line": "Line 1",
        "connections": ["BPB", "PUL"]
    },
    "PUL": {
        "id": "PUL", "name": "Pulomas",
        "coordinates": (-6.1720, 106.8815), "line": "Line 1",
                "connections": ["BVN", "KG"]
    },
    "KG": {
        "id": "KG", "name": "Kayu Putih",
        "coordinates": (-6.1790, 106.8785), "line": "Line 1",
        "connections": ["PUL", "VEL"]
    },
    "VEL": {
        "id": "VEL", "name": "Velodrome",
        "coordinates": (-6.2183, 106.8697), "line": "Line 1",
        "connections": ["KG"]
    }
```

Then, we first implement the A* Algorithm calculation without the additional heuristic. In theory, the implementation doesn't deviate from its most common implementation, that is by using a heuristic of the actual distance of the node to the destination, using Euclidian's distance to calculate the heuristic cost. This is then implemented in Python, with functions such as to get the actual cost or the heuristic and to reconstruct the path taken using the A* Algorithm. A thing to note here is that we implement the cost calculation not by distance, but using time. It is calculated with the distance cost divided by the speed of LRT in average. Here, we assume the average speed of LRT is 35 km/h.

To implement the slime mold heuristic, we apply the standard A* Algorithm logic with a reward system inspired by the slime mold behavior. The implementation is as follows: each station has a defined nutrient value, or in other words, weighing system. There are no set criterias or formulas used to count the value here, but is a rough estimate from the author to value the importance of each station.

1. 'Pegangsaan Dua': 0.7
2. 'Boulevard Utara': 0.9
3. 'Boulevard Selatan': 0.9
4. 'Pulomas': 0.6
5. 'Kayu Putih': 0.6
6. 'Velodrome': 0.8

For example, Boulevard Utara and Boulevard Selatan is weighed higher because the station resides in a highly-populated residential area, compared to Pulomas and Kayu Putih which is less dense and 'important'.

These evaluations affect the reward system that is defined with this function:

$$f(n) = g(n) + h(n) - nutrient\_reward$$

$$nutrient\_reward = nutrient\_value\_of\_neighbor * scaling\_factor$$

In short, the calculation considers the nutrient value, or the weigh of a node that might be visited with its scaling factor. The scaling factor here varies according to the input, it may be 2, 3, 10, or even 100 depending on the context of the problem. The higher the value of a station, the smaller the total function cost it has. In short, if a station is valued higher, it will reduce the total cost of it, with the significance relying on the scaling factor. So, the A* Algorithm calculation may consider choosing a "longer distance" option because of the reward system implemented here.

### *B. Experiments and Results*

The research goal now is to test the algorithm in the context of the Jakarta LRT, comparing the difference between the modified slime mould inspired A* Algorithm with its standard form. The test case initially used are basic tests from station to station as such:

1. Pegangsaan Dua -> Velodrome
2. Pegangsaan Dua -> Boulevard Selatan
3. Velodrome -> Boulevard Utara
4. Velodrome -> Boulevard Selatan

In each test run, we collect and analyse these key metrics:

1. The chosen path of the station
2. The total travel cost (in minutes) from the chosen path, and
3. The total nodes considered and explored before finding the goal

The result of the initial test case is described below:

```
Pegangsaan Dua → Velodrome:
  Weighted A*:
    Path: Pegangsaan Dua → Boulevard Utara → Boulevard Selatan → Pulomas → Equestrian → Velodrome
    Cost: 13.8 minutes
    Nodes explored: 6
  Standard A*:
    Path: Pegangsaan Dua → Boulevard Utara → Boulevard Selatan → Pulomas → Equestrian → Velodrome
    Cost: 13.8 minutes
    Nodes explored: 6

Pegangsaan Dua → Boulevard Selatan:
  Weighted A*:
    Path: Pegangsaan Dua → Boulevard Utara → Boulevard Selatan
    Cost: 5.9 minutes
    Nodes explored: 3
  Standard A*:
    Path: Pegangsaan Dua → Boulevard Utara → Boulevard Selatan
    Cost: 5.9 minutes
    Nodes explored: 3

Velodrome → Boulevard Utara:
  Weighted A*:
    Path: Velodrome → Equestrian → Pulomas → Boulevard Selatan → Boulevard Utara
    Cost: 11.0 minutes
    Nodes explored: 5
  Standard A*:
    Path: Velodrome → Equestrian → Pulomas → Boulevard Selatan → Boulevard Utara
    Cost: 11.0 minutes
    Nodes explored: 5

  Weighted A*:
    Path: Pegangsaan Dua → Boulevard Utara → Boulevard Selatan
    Cost: 5.9 minutes
    Nodes explored: 3
  Standard A*:
    Path: Pegangsaan Dua → Boulevard Utara → Boulevard Selatan
    Cost: 5.9 minutes
    Nodes explored: 3
```

After the initial test case, the author realized that the research goal will not be reflected properly in the current sample test case using Jakarta's LRT Route. This is due to the orientation of LRT stations being placed in a straight line, which would produce a straight-forward result of the closest route to the station literally being as if taking the station as it is. So, changes were made to the sample test case.

We now experiment by extending the current Jakarta LRT route. A new branch route is created from Pulomas, the 4th station in order from Pegangsaan Dua. Now, there are 2 ways to reach the end station of Velodrome, which is:

*Pulomas -> Rawamangun -> Pemuda -> Velodrome,*
*Pulomas -> Kayu Putih -> Velodrome*

The proper location data of Rawamangun and Pemuda is added according to real-life data. The nutritional or weigh values are 0.95 and 0.7 respectively, considering Rawamangun is a major commercial hub which boosts its value higher than other stations. These new values are updated, along with the new connection between nodes that changed between Pulomas, Velodrome, and the new added Nodes of Rawamangun and Pemuda.

Now, the test case for the updated sample is:

1. Boulevard Utama -> Velodrome
2. Velodrome -> Boulevard Utama
3. Pegangsaan Dua -> Rawamangun
4. Pegangsaan Dua -> Kayu Putih

Using a scaling factor of 10.

```
Using a SCALING_FACTOR of 10 for weighted tests.


--- Test Case: Boulevard Utara → Velodrome ---
Standard Path:   Boulevard Utara → Boulevard Selatan → Pulomas → Kayu Putih → Velodrome
                 (Cost: 15.8 mins, Nodes: 6)
Weighted Path:   Boulevard Utara → Boulevard Selatan → Pulomas → Kayu Putih → Velodrome
                 (Cost: 15.8 mins, Nodes: 7)

--- Test Case: Velodrome → Boulevard Utara ---
Standard Path:   Velodrome → Kayu Putih → Pulomas → Boulevard Selatan → Boulevard Utara
                 (Cost: 15.8 mins, Nodes: 7)
Weighted Path:   Velodrome → Kayu Putih → Pulomas → Boulevard Selatan → Boulevard Utara
                 (Cost: 15.8 mins, Nodes: 7)

--- Test Case: Pegangsaan Dua → Rawamangun ---
Standard Path:   Pegangsaan Dua → Boulevard Utara → Boulevard Selatan → Pulomas → Rawamangun
                 (Cost: 11.4 mins, Nodes: 5)
Weighted Path:   Pegangsaan Dua → Boulevard Utara → Boulevard Selatan → Pulomas → Rawamangun
                 (Cost: 11.4 mins, Nodes: 5)

--- Test Case: Pegangsaan Dua → Kayu Putih ---
Standard Path:   Pegangsaan Dua → Boulevard Utara → Boulevard Selatan → Pulomas → Kayu Putih
                 (Cost: 9.5 mins, Nodes: 5)
Weighted Path:   Pegangsaan Dua → Boulevard Utara → Boulevard Selatan → Pulomas → Kayu Putih
                 (Cost: 9.5 mins, Nodes: 5)
```

## IV. ANALYSIS

### *A. Initial Data Description*

Average Travel Cost

- Standard A*: 13.13 minutes
- Modified A*: 13.13 minutes
- Difference: 0

Average Nodes Explored

- Standard A*: 5.75 nodes
- Modified A*: 6 nodes
- Difference: +0.25 nodes

Based on the initial reading on the key metrics for analysis, there are no impact on the final path cost. This means that the calculated reward for taking a higher weight or nutrient value node was not sufficient enough to make the algorithm select a physically longer path. Both algorithms took the most optimal route in all the test cases.

The only difference that happened was an increase in nodes explored, which was about a 4.35% using the modified A* algorithm. This means that using the modified algorithm version takes up more computational effort, although in this current test case, the difference is very minimal and may be described better if using a bigger and more complex test case. However, insights that can be inferred here is that the modified A* algorithm actively calculates the high reward routes and takes it into account, even if the option is then discarded.

An important thing to highlight is that this analysis is based on the scaling factor of 10 used in the computation. Scaling factor plays a significant role in determining the selected station. For example, changing the scaling factor to 50 (which is a very big increase) would cause a difference in the route taken. For the first test case from Boulevard Utara to Velodrome, the chosen path of modified A* algorithm is now the path to Rawamangun (the fictional extended path) instead of Kayu Putih (our original Jakarta LRT route).

### B. Test Case Explanation

Let's take a look at the first test case example, that is to find the fastest route from the start to end station, Pegangsaan Dua to Velodrome. With scaling factor 10, the function is as such:

$$f(n) = g(n) + h(n) - nutrient\_reward$$

$$nutrient\_reward = nutrient\_value\_of\_neighbor * 10$$

The calculation between the 2 algorithms will differ in choosing the faster path to reach Velodrome from Pulomas, so we will describe the calculation there:

*Path A: via Kayu Putih*

1. g(n) (actual cost):

   Time to Pulomas (4.8 mins) + time from Pulomas to Kayu Putih (2.4 mins) = 7.2 mins

2. h(n) (estimated cost to destination):

   Calculated heuristic cost = 7.6 mins

3. nutrient_reward:

   The nutrient value of Kayu Putih (0.6) multiplied by the scaling factor (10) = 6.0.

Final cost = 7.2 + 7.6 - 6.0 = **8.8 mins**

*Path B: via Rawamangun*

1. g(n) (actual cost):

   Time to Pulomas (4.8 mins) + time from Pulomas to Rawamangun (4.3 mins) = 9.1 mins.

2. h(n) (estimated cost to destination):

   Calculated heuristic cost = 6.4 mins

3. nutrient_reward:

   The nutrient value of Rawamangun (0.95) multiplied by the scaling factor (10) = 9.5.

Final cost = 9.1 + 6.4 - 9.5 = **6.0 mins**

Because the new path B has a lower final cost value than path A, it is placed at a higher priority for the modified A* Algorithm to explore, which causes an increase in node exploration. Although, after considering both paths, the algorithm ultimately chooses the path via Kayu Putih (the original LRT route without being extended) because it has a total final cost value lower than the other.

### C. Modified A* Algorithm Application

The slime-mold inspired A* algorithm has many benefits in taking account weighing factors to calculate transport routes. One of the most beneficial addition from this heuristic is to add other factors into decision-making, although in reality, route-planning may use a higher form of algorithm instead of using a deterministic path-finding algorithm. These benefits include:

1. Factoring aspects outside of distance cost, such as the possibility of delays
2. Taking account high visitor volume per station, which would accommodate a larger amount of visitors opposed to choosing the fastest route
3. Enabling future route extension plans with respect to the "nutritional" value per station

## V. CONCLUSION

This research discusses about the application of the biological inspired network design using slime mold's food foraging process as a heuristic in a deterministic path-finding algorithm such as A* Algorithm. The heuristic chosen in this manner mimics how slime mold's prioritize higher nutrient food sources, which in our experiment translates into stations that has a weighted value, based off visitor numbers, delays, or even amount of stations connected to. The main goal of the experiment was to compare the difference between the modified and standard algorithm and find insights from the tests.

Key findings in this experiment are the additional considerations taken by the algorithm in calculating its cost function. It accounts for the weighted value of each station by subtracting the current A* algorithm function with the "nutrient reward", which is the weighted value of the considered node, multiplied by its scaling factor. The higher the scaling factor, the bigger the significance in the decision-making process of the algorithm. It may even influence the modified A* Algorithm to take a physically longer route because of the added weighted value.

Still, there are many limitations during this experiment that could be improved. Most significantly is the small network sample used. Applying this modified algorithm to a more complex and larger network may produce more results and insights. Then, another point of improvement is creating a basis in determining the scaling factor value or the weighted value for each node, removing subjective assumptions for a more objective decision-making.

In conclusion, this paper demonstrated how taking inspiration from biology may be used to improve existing transport systems.


## ACKNOWLEDGMENT

As the author of this paper, I would like to express my sincere gratitude to all parties who have provided support and inspiration throughout the writing process, enabling me to successfully complete this research titled "Applying Slime Mold's Behaviour as a Heuristic in A* Algorithm Pathfinding for Jakarta's Light Rail Transit (LRT) Network." I would like to thank:

1. Mr. Dr. Rinaldi Munir, Mrs. Dr. Nur Ulfa Maulidevi, and Mr. Monterico Adrian, as the lecturers for the IF2211 Algorithm Strategies course, for the materials and lessons shared in the Informatics Engineering class during the 2nd Semester of the 2024/2025 academic year.

2. My beloved parents, who have always provided moral support and prayers. Their presence and positive

encouragement gave me the extra motivation to complete this work to the best of my ability.

3. My dearest girlfriend, for her endless support, patience, and help. Her encouragement was a constant source of strength during the challenges of writing this paper.

4. The community of scholars, researchers, and creators whose published works provided the essential foundation for this paper. The referenced journals, articles, and videos were invaluable, offering critical insights that significantly deepened my comprehension of the subject matter.

## REFERENCES

[1] A. Tero et al., "Rules for biologically inspired adaptive network design," Science, vol. 327, no. 5964, pp. 439–442, Jan. 2010. doi: 10.1126/science.1177894. [Online]. Available: https://www.science.org/doi/10.1126/science.1177894.

[2] Wenxiao, "SlimeMould." github.com. https://github.com/MoeBuTa/SlimeMould (accessed Jun. 23, 2025).

[3] Verge Science. What Self-Driving Cars Can Learn From Brainless Slime Mold. (May 1, 2018). Accessed: Jun. 23, 2025. [Online Video]. Available: https://www.youtube.com/watch?v=40f7_93NIgA

[4] T. Irving. "A virtual slime mold that can design a subway network." phys.org. https://phys.org/news/2022-01-virtual-slime-mold-subway-network.html (accessed Jun. 23, 2025).

[5] B. Roemmele. Fungal Intelligence: Lead The Way To Improved Technological Systems. (Jan. 1, 2019). Accessed: Jun. 23 2025. [Online Video]. Available: https://www.youtube.com/watch?v=RVe94qa1ar4

[6] A. Tamtomo., "Uji Publik LRT Jakarta, Ini Informasi dan Cara Pendaftaran" kompas.com. https://megapolitan.kompas.com/read/2019/06/18/10143171/infografik-uji-publik-lrt-jakarta-ini-informasi-dan-cara-pendaftaran(accessed Jun. 24, 2025).

[7] "LRT". Jakarta Kita. jakartakita.com/lrt (accessed Jun. 24, 2025).

[8] R. Munir, "Route Planning (2025) Bagian2", informatika.stei.itb.ac.id. https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2024-2025/22-Route-Planning-(2025)-Bagian2.pdf (accessed Jun. 24, 2025).

## 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, 24 Juni 2025

Nicholas Andhika Lucas
13523014