



UJM 14 (2) 2025

UNNES Journal of Mathematics

<http://journal.unnes.ac.id/sju/index.php/ujm>



Application Of Min-Plus Algebra in Determining the Shortest Route for Semarang Feeder Buses

Diana Ayu Lestari*, Isnaini Rosyida

Department of Mathematics, Faculty of Mathematics and Natural Science,
Universitas Negeri Semarang, Semarang, 50229, Indonesia

Article Info

Article History:

Received 18-06-2025

Accepted 20-10-2025

Published 31-10-2025

Keywords:

Min-plus algebra

Shortest route

Semarang feeder bus

Abstract

Feeder buses extend the Bus Rapid Transit (BRT) network of Semarang City to areas that the BRT cannot reach, yet their routes are rarely reviewed against travel time, which determines service efficiency and fuel use. This study applies min-plus algebra to find the shortest feeder bus routes because it represents a weighted directed network as a matrix and gives the earliest and the latest travel time to every stop in one computation, so that the stops lying on a shortest route can be identified. The aims were (1) to model two feeder routes as weighted directed simple graphs and adjacency matrices, (2) to determine the optimal route on each, and (3) to implement the computation in Python with Google Colab. Distances, travel times, and average bus speeds (18 and 17 km/h) were collected by field observation for the route from the BNI UNNES bus stop to Bandarjo Market (15 vertices) and for the route from the Semarang State Polytechnic to Banyumanik Terminal (20 vertices). The shortest route of the first network takes 35.63 minutes over 10.48 km, and that of the second takes 21.67 minutes over 6.14 km. Both differ from the current feeder bus routes and are proposed as route alternatives, to be validated against traffic conditions and passenger demand. The Python implementation reproduced the manual calculation and can be reused for larger networks.

How to cite:

Lestari, D. A., & Rosyida, I. (2025). Application Of Min-Plus Algebra in Determining the Shortest Route for Semarang Feeder Buses. *UNNES Journal of Mathematics*. 14 (2): 11-20

© 2025 Semarang State University

*Correspondence Address

E-mail: lestaridiana347@students.unnes.ac.id

p-ISSN 2252-6943

e-ISSN 2460-5859

1. Introduction

Population growth and economic activity in Semarang City increase the demand for public transport and make the efficiency of transport routes a pressing problem. Feeder buses play an important role in this system because they reach areas that the Bus Rapid Transit (BRT) cannot serve. A feeder route that is longer or slower than necessary increases operating time and fuel consumption and reduces the attractiveness of public transport. Current feeder routes therefore need to be reviewed to find out whether shorter alternatives exist. This need is evident on the two corridors studied here: the route from the Semarang State Polytechnic (Polines) to Banyumanik Terminal is used by many residents, whereas the corridor from the UNNES campus to Bandarjo Market in Ungaran passes many schools, places of worship, and markets but is not yet served by a feeder bus.

The problem can be modeled as a weighted directed simple graph whose arc weights are travel times. Min-plus algebra, the set $\mathbb{R} \cup \{+\infty\}$ with the operations $\oplus$ (minimum) and $\otimes$ (addition), suits this model because the adjacency matrix and its powers directly give the minimum travel time over paths of a given length. Computing these matrix powers by hand is laborious as the network grows, so the calculation was implemented in the Python programming language.

Previous studies of shortest routes in transport networks have mostly used other algorithms. Rifanti and Arifwidodo (2019) applied Floyd's algorithm to find the shortest routes between tourist attractions in Banyumas Regency. Pramudhita and Muljono (2018) developed an Android application that uses the A* method to find the nearest BRT stop in Semarang, and Khamami and Saputra (2019) built a Floyd–Warshall application for city transport routes in Semarang whose results agreed with manual calculation. These studies concentrate on distance and on application development. Min-plus algebra has been used to determine the fastest route for milk distribution (Suwanti et al., 2017) and for fish distribution (Sumianti et al., 2024), to compare angkot routes in Bandung with the Greedy algorithm, where it gave the better routes (Susilowati, 2018), and to find the shortest path to the airport by Trans Jogja bus (Susilowati & Fitriani, 2019).

To the authors' knowledge, min-plus algebra has not yet been applied to feeder bus routes in Semarang. The novelty of this study is threefold. First, two real feeder corridors are modeled with field-measured distances and travel times, and alternative routes are proposed, one of them on a corridor not yet served by a feeder bus. Second, both the earliest and the latest time vectors are used, so that the equality of the two identifies the stops of a shortest route and their difference shows how much time is lost at the other stops. Third, a reproducible Python implementation is provided for larger networks.

This study aims to determine the shortest feeder bus routes in Semarang City using min-plus algebra. The research questions are: (1) how can the two feeder bus routes be modeled as weighted directed simple graphs and solved as matrices using min-plus algebra; (2) what are the shortest routes obtained on the two feeder bus routes; and (3) how can the computation be implemented in the Python programming language? The basic definitions and theorems used are given below.

Definition 1.1 Definition Min-plus Algebra (Suroto, 2022)

The notation R_{min} is the set $\mathbb{R} \cup \{\varepsilon\}$, where R is the set of real numbers, $\varepsilon = +\infty$ and the neutral element $e = 0$. For $a, b \in R_{min}$, the operations are defined $\oplus$ dan $\otimes$ where $a \oplus b = \min(a, b)$ and $a \otimes b = a + b$. The set R_{min} with the operations $\oplus$ dan $\otimes$ is called the Min-plus Algebra and is denoted by

$$\mathcal{R}_{min} = (\mathbb{R} \cup \{+\infty\}, \oplus, \otimes).$$

Definition 1.2 Definition Network (Watanabe & Watanabe, 2014)

Suppose $G = (V, E)$ a graph with n vertices and m arcs and $N = (G, w)$ is a network on the graph G . Suppose $A(N) = (a_{ij})$ the directly related matrix of the network N is defined as the directly related matrix of the network N , $A = (a_{ij})$ with elements in

$$R_{min} \text{ as } a_{ij} = \begin{cases} c(v_i, v_j), & \text{jika } (v_i, v_j) \in E \\ \varepsilon(+\infty), & \text{jika } (v_i, v_j) \notin E \end{cases}$$

According to (Sumianti et al., 2024) in Definition 2.2.4 the network in question is a directed and weighted graph. In the shortest path problem, a_{ij} the weight of an arc (j, i) is the travel time to traverse the arc (j, i) .

Theorem 1.1 (Susilowati, 2018)

Given a network of paths in the same direction as n the points and A is a matrix directly related to the network. The minimum time vector when traversing the network from point 1 to point 2 n is given by:

$$x^e = A^* \otimes b^e = (I \oplus A \oplus A^{\otimes 2} \oplus A^{\otimes 3} \oplus \dots \oplus A^{\otimes n-1}) \otimes b^e$$

Where $b^e = [0 \ \varepsilon \ \varepsilon \ \dots \ \varepsilon]^T$. Further x_n^e is the minimum time when traversing the network.

Theorem 1.2 (Susilowati, 2018)

Given a matrix A which is a directly related matrix in a network in the direction of n a point. The maximum time vector when traversing the network from the starting point to the point n can be found using the equation

$$x^l = -((A^*)^T \otimes b^l) \text{ with } b^l = [\varepsilon \ \varepsilon \ \dots \ -x_n^e]^T$$

Definition 1.3 Trajectory Shortest (Suwanti et al., 2017)

A path $a_{ij} \in A$ in a one-way network S is called the shortest path if $x_i^e = x_i^l$ and $x_j^e = x_j^l$.

In this study the weights are the travel times between successive points. A route of minimum time is not necessarily the route of minimum distance, because the average speed differs between road segments; on both networks studied here, however, the minimum-time route also has the shortest distance among the candidate routes.

2. Method

This quantitative descriptive study applies min-plus algebra to two feeder bus routes in Semarang City: Route 1 from the BNI UNNES bus stop to Bandarjo Market, Ungaran, and Route 2 from the Semarang State Polytechnic (Polines) bus stop to Banyumanik Terminal. Primary data were collected by field observation, interviews, and documentation, namely the routes that can be taken, the stops and intersections on each route, the average bus speed, and the distance and travel time between consecutive points. Travel times were estimated from the distances and the average bus speed (18 km/h on Route 1 and 17 km/h on Route 2). The literature study covered min-plus algebra, network models, and shortest-route applications.

The analysis consisted of six steps: (1) the points of each route were listed and the roads connecting them were identified; (2) a weighted directed simple graph was drawn with travel time as the arc weight; (3) the $n \times n$ adjacency matrix A over min-plus algebra was formed according to Definition 1.2; (4) the minimum time vector $x^e = A^* \otimes b^e$ was computed according to Theorem 1.1; (5) the latest time vector $x^l = -((A^*)^T \otimes b^l)$ was computed according to Theorem 1.2; and (6) the shortest route was taken as the set of vertices with $x^e_i = x^l_i$ (Definition 1.3) and drawn on the network. Steps (3) to (6) were carried out manually and with Python in Google Colab, and the two results were compared.

3. Results and Discussion

3.1. Route 1: BNI UNNES Bus Stop to Bandarjo Market

Route 1 connects the BNI UNNES bus stop (v1) with Bandarjo Market, Ungaran (v15). Its network has 15 vertices (Table 1, Figure 1) and 17 arcs, each with a distance and a travel time based on an average bus speed of 18 km/h (Table 2). The data do not take road width, weather, small roads, or congestion into account. The network allows six candidate routes (Table 3).

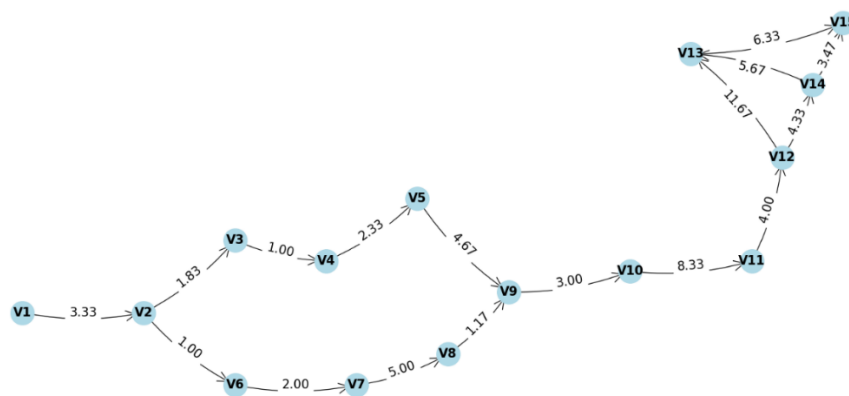


Figure 1 Weighted directed graph of Route 1 (BNI UNNES bus stop – Bandarjo Market)

Table 1 Vertices of Route 1 (BNI UNNES bus stop – Bandarjo Market)

Vertex	Location
v1	BNI UNNES bus stop
v2	Intersection of Jl. Sekaran Raya, Jl. Ampel Gading Raya, and Jl. Kolonel HR Hadijanto
v3	Intersection of Jl. Patemon, Gg. Kedawung I, and Gg. Selekor
v4	Intersection of Jl. Patemon and Jl. Kauman
v5	Intersection of Jl. Patemon and Jl. AMD
v6	UNNES Women's Flats
v7	Kalisegoro bus stop
v8	Intersection of Jl. Ampel Gading Raya and Jl. Muntal
v9	Intersection of Jl. Muntal and Jl. Muntal Raya
v10	Intersection of Jl. Patemon and Jl. Pakintelan II
v11	Intersection of Jl. Mr. Koesbiyono Tjondrowibowo and Jl. Ungaran-Cangkiran
v12	Intersection of Jl. Pramuka and Jl. Ungaran-Cangkiran
v13	Unyil Park, Ungaran
v14	Intersection of Jl. Kisarino Mangunpranoto and Jl. Patimura
v15	Bandarjo Market, Ungaran

Table 2 Arc distances and travel times of Route 1

Arc	Distance (km)	Time (min)	Arc	Distance (km)	Time (min)
v1 → v2	0.92	3.33	v9 → v10	0.85	3.00
v2 → v3	0.56	1.83	v10 → v11	2.50	8.33
v3 → v4	0.31	1.00	v11 → v12	1.20	4.00
v4 → v5	0.71	2.33	v12 → v13	3.30	11.67
v5 → v9	1.40	4.67	v12 → v14	1.27	4.33
v2 → v6	0.30	1.00	v13 → v15	2.08	6.33
v6 → v7	0.60	2.00	v14 → v13	1.54	5.67
v7 → v8	1.20	5.00	v14 → v15	1.04	3.47
v8 → v9	0.60	1.17			

Table 3 Candidate routes of Route 1

Route	Sequence of vertices	Distance (km)	Time (min)
1	v1-v2-v3-v4-v5-v9-v10-v11-v12-v13-v15	13.83	46.49
2	v1-v2-v3-v4-v5-v9-v10-v11-v12-v14-v15	10.76	36.29
3	v1-v2-v3-v4-v5-v9-v10-v11-v12-v14-v13-v15	13.34	44.82
4	v1-v2-v6-v7-v8-v9-v10-v11-v12-v13-v15	13.55	45.83
5	v1-v2-v6-v7-v8-v9-v10-v11-v12-v14-v15	10.48	35.63
6	v1-v2-v6-v7-v8-v9-v10-v11-v12-v14-v13-v15	13.06	44.16

The candidate routes differ by up to 10.86 minutes: the fastest takes 35.63 minutes and the slowest 46.49 minutes. The min-plus procedure in Section 3.3 identifies the fastest route without listing all candidates, which matters when the network is larger.

3.2. Route 2: Polines Bus Stop To Banyumanik Terminal

Route 2 connects the Polines bus stop (v1) with Banyumanik Terminal (v20) and was chosen because many residents use the feeder service on it. Its network has 20 vertices (Table 4, Figure 2) and 20 arcs, with an average bus speed of 17 km/h (Table 5). From v6 a bus can either continue through v7 to v11 and reach v15, or take the connecting road to v12 and continue through v13 and v14 to v15.

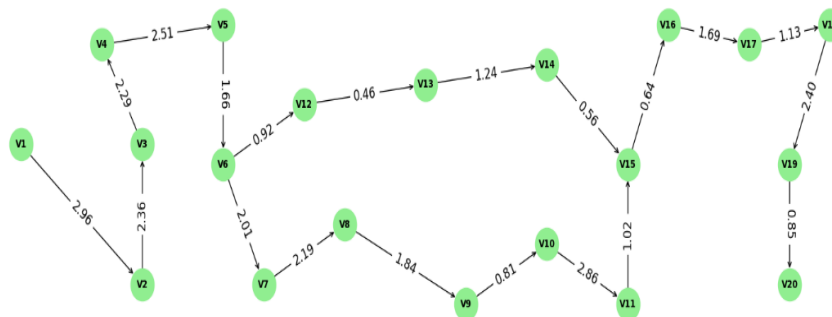


Figure 2 Weighted directed graph of Route 2 (Polines – Banyumanik Terminal)

Table 4 Vertices of Route 2 (Polines – Banyumanik Terminal)

Vertex	Location
v1	Polines bus stop
v2	Pedalangan Elementary School 1 bus stop
v3	Telogosari 1 bus stop
v4	Mulawarman bus stop
v5	Trans Marga Central Java bus stop
v6	Pedalangan Elementary School 2 bus stop
v7	Highway bus stop
v8	Villa Durian bus stop
v9	Intersection of Jl. Bina Remaja and Jl. Sukun Raya
v10	Regional Police Special Crimes bus stop
v11	Post Office bus stop
v12	Intersection of Jl. Tusam Timur Raya, Jl. Tusam Timur II, and Jl. Tusam III
v13	Intersection of Jl. Tusam Timur Raya, Jl. Tusam II, and Jl. Tusam I
v14	Intersection of Jl. Tusam Raya and Jl. Waru Raya
v15	Intersection of Jl. Jati Raya, Jl. Tusam Raya, Jl. Kanfer Raya, and Jl. Cemara Raya
v16	SMA 9 Semarang bus stop
v17	Grafika bus stop
v18	SMP Eka Sakti bus stop
v19	SMA 4 Semarang bus stop
v20	Banyumanik Terminal

Table 5 Arc distances and travel times of Route 2

Arc	Distance (km)	Time (min)	Arc	Distance (km)	Time (min)
v1 → v2	0.84	2.96	v11 → v15	0.29	1.02
v2 → v3	0.67	2.36	v6 → v12	0.26	0.92
v3 → v4	0.65	2.29	v12 → v13	0.13	0.46
v4 → v5	0.71	2.51	v13 → v14	0.35	1.24
v5 → v6	0.47	1.66	v14 → v15	0.16	0.56
v6 → v7	0.57	2.01	v15 → v16	0.18	0.64
v7 → v8	0.62	2.19	v16 → v17	0.48	1.69
v8 → v9	0.52	1.84	v17 → v18	0.32	1.13
v9 → v10	0.23	0.81	v18 → v19	0.68	2.40
v10 → v11	0.81	2.86	v19 → v20	0.24	0.85

3.3. Shortest Routes by Min-Plus Algebra

For each network the adjacency matrix A (15×15 for Route 1 and 20×20 for Route 2) was formed from the arc travel times following Definition 1.2, with a_{ij} the travel time of the arc from vertex j to vertex i and $\varepsilon = +\infty$ where no arc exists. The matrix $A^* = I \oplus A \oplus A^{\otimes 2} \oplus \dots \oplus A^{\otimes (n-1)}$ was computed up to the power $n - 1$ (14 for Route 1 and 19 for Route 2), and the vectors $x^e = A^* \otimes b^e$ and $x^l = -((A^*)^T \otimes b^l)$ were obtained from Theorems 1.1 and 1.2. Table 6 lists both vectors.

The adjacency matrix of Route 1 (15×15) is:

$$A = \begin{bmatrix} \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ 3,33 & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & 1,83 & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & 1,00 & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & 2,33 & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & 1,00 & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & 2,00 & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & 5,00 & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon & 4,67 & \varepsilon & \varepsilon & 1,17 & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & 3,00 & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & 8,33 & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & 4,00 & \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & 11,67 & \varepsilon & 5,67 & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & 4,33 & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & 6,33 & 3,47 \end{bmatrix}$$

and the adjacency matrix of Route 2 (20×20) is:

$$A = \begin{bmatrix} \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon \\ 2,96 & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon \\ \epsilon & 2,36 & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon \\ \epsilon & \epsilon & 2,29 & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon \\ \epsilon & \epsilon & \epsilon & 2,51 & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon \\ \epsilon & \epsilon & \epsilon & \epsilon & 1,66 & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon \\ \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & 2,01 & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon \\ \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & 2,19 & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon \\ \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & 1,84 & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon \\ \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & 0,81 & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon \\ \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & 2,86 & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon \\ \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & 0,92 & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon \\ \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & 0,46 & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon \\ \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & 1,24 & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon \\ \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & 1,02 & \epsilon & \epsilon & 0,56 & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon \\ \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & 0,64 & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon \\ \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & 1,69 & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon \\ \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & 1,13 & \epsilon & \epsilon & \epsilon & \epsilon \\ \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & 2,40 & \epsilon & \epsilon & \epsilon \\ \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & 0,85 & \epsilon & \epsilon \end{bmatrix}$$

Table 6 Minimum and latest time vectors of Route 1 and Route 2

Route 1			Route 2		
Vertex	x^e (min)	x^l (min)	Vertex	x^e (min)	x^l (min)
v1	0.00	0.00	v1	0.00	0.00
v2	3.33	3.33	v2	2.96	2.96
v3	5.16	4.50	v3	5.32	5.32
v4	6.16	5.50	v4	7.61	7.61
v5	8.49	7.83	v5	10.12	10.12
v6	4.33	4.33	v6	11.78	11.78
v7	6.33	6.33	v7	13.79	6.24
v8	11.33	11.33	v8	15.98	8.43
v9	12.50	12.50	v9	17.82	10.27
v10	15.50	15.50	v10	18.63	11.08
v11	23.83	23.83	v11	21.49	13.94
v12	27.83	27.83	v12	12.70	12.70
v13	37.83	29.30	v13	13.16	13.16
v14	32.16	32.16	v14	14.40	14.40
v15	35.63	35.63	v15	14.96	14.96
			v16	15.60	15.60
			v17	17.29	17.29
			v18	18.42	18.42
			v19	20.82	20.82
			v20	21.67	21.67

Note: bold rows are the vertices with $x^e = x^l$, i.e. the vertices of the shortest route.

The entry x^e_i is the minimum time needed to reach vertex i from the first vertex, and x^l_i equals x^e_n minus the minimum time from vertex i to the last vertex n . Hence $x^l_i \leq x^e_i$ for every vertex, with equality exactly at the vertices that lie on a shortest route, and the difference $x^e_i - x^l_i$ is the time a bus loses if it is routed through vertex i . On Route 1, equality holds at $v_1, v_2, v_6, v_7, v_8, v_9, v_{10}, v_{11}, v_{12}, v_{14}$, and v_{15} , so the shortest route runs from the BNI UNNES bus stop via the UNNES Women's Flats (v_6), the Kalisegoro bus stop (v_7), the intersection of Jl. Patemon and Jl. Pakintelan II (v_{10}), and the intersection of Jl. Kisarino Mangunpranoto and Jl. Patimura (v_{14}) to Bandarjo Market. Its travel time is 35.63 minutes, and its distance is $0.92 + 0.30 + 0.60 + 1.20 + 0.60 + 0.85 + 2.50 + 1.20 + 1.27 + 1.04 = 10.48$ km (Figure 3).

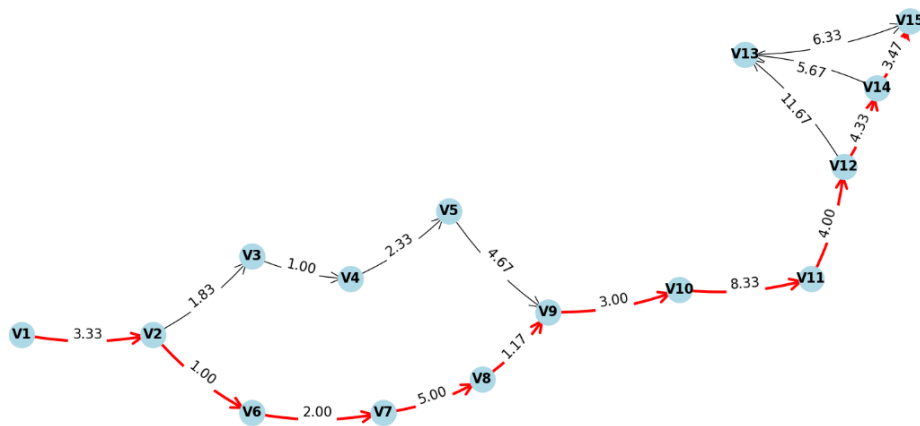


Figure 3 Shortest route of Route 1 (BNI UNNES bus stop – Bandarjo Market)

The red arcs in Figure 3 form the shortest route of Route 1. This route is not yet operated by a feeder bus although it passes schools, places of worship, and markets, so it can be proposed as a new feeder route from Semarang State University to Bandarjo Market.

On Route 2, equality holds at v_1 to v_6 and v_{12} to v_{20} , so the shortest route runs from the Polines bus stop through the stops Pedalangan Elementary School 1, Telogosari 1, Mulawarman, Trans Marga Central Java, and Pedalangan Elementary School 2 (v_2 to v_6), then through the connecting road to v_{12} (Jl. Tusam Timur Raya) and the Tusam, Waru, Jati, and Cemara intersections (v_{12} to v_{15}), and finally through the stops SMA 9 Semarang, Grafika, SMP Eka Sakti, and SMA 4 Semarang (v_{16} to v_{19}) to Banyumanik Terminal. The travel time is 21.67 minutes and the distance is $0.84 + 0.67 + 0.65 + 0.71 + 0.47 + 0.26 + 0.13 + 0.35 + 0.16 + 0.18 + 0.48 + 0.32 + 0.68 + 0.24 = 6.14$ km (Figure 4). This route differs from the route currently operated, so it can be proposed as a route alternative.

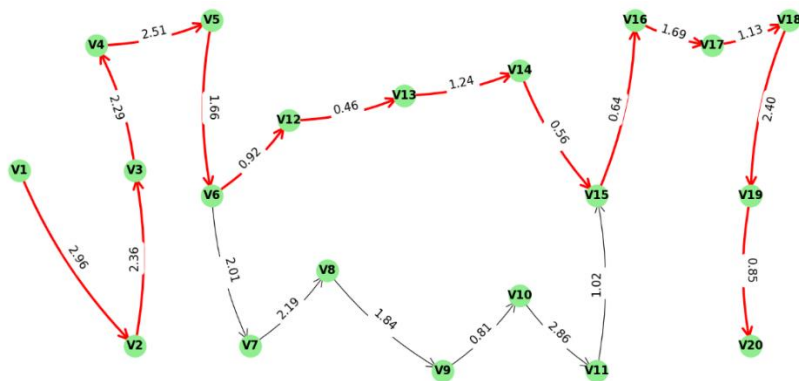


Figure 4 Shortest route of Route 2 (Polines – Banyumanik Terminal)

3.4. Python Implementation

The min-plus computation was implemented in Python on Google Colab with NumPy and NetworkX so that it is fast and reusable for larger networks. The program is shown for Route 1 in Figures 5 to 10. It defines a function for the min-plus product, in which sums are replaced by minima and products by sums, and enters the matrix A with `np.array()`, where `np.inf` stands for ϵ (Figure 5). A^* is accumulated from the powers of A up to $n - 1$ (Figure 6), the vectors x^e and x^l are computed according to Theorems 1.1 and 1.2 (Figures 7 and 8), and the vertices with $x^e = x^l$ are selected as the shortest route (Figure 9). The network and the route are drawn with `nx.DiGraph()` and `nx.draw()`. The output in Figure 10 agrees with the manual result in Table 6, and an independent check of both networks with the Bellman–Ford algorithm gave the same routes and times.

```

import numpy as np

def minplus_multiplication(A, B):
    """Operasi perkalian min-plus matrices"""
    n = len(A)
    C = np.full((n, n), float('inf'))
    for i in range(n):
        for j in range(n):
            C[i][j] = min(A[i][k] + B[k][j] if not (np.isinf(A[i][k]) or np.isinf(B[k][j])) else float('inf') for k in range(n))
    return C

def print_matrix(matrix, title):
    print(f"\n{title}")
    for row in matrix:
        formatted_row = [
            "inf" if np.isinf(x) else f"{x:.2f}"
            for x in row
        ]
        print("[ " + " ".join(f"{v:s}" for v in formatted_row) + " ]")

# Input Matriks A
A = np.array([
    [float('inf'), 2.0],
    [2.85 + (float('inf'))*19,
     (float('inf'), 2.16) + (float('inf'))*18,
     (float('inf'))*2 + (2.29) + (float('inf'))*17,
     (float('inf'))*3 + (2.51) + (float('inf'))*16,
     (float('inf'))*4 + (1.66) + (float('inf'))*15,
     (float('inf'))*5 + (2.01) + (float('inf'))*14,
     (float('inf'))*6 + (2.33) + (float('inf'))*13,
     (float('inf'))*7 + (1.84) + (float('inf'))*12,
     (float('inf'))*8 + (0.81) + (float('inf'))*11,
     (float('inf'))*9 + (2.86) + (float('inf'))*10,
     (float('inf'))*5 + (0.92) + (float('inf'))*14,
     (float('inf'))*11 + (0.46) + (float('inf'))*8,
     (float('inf'))*12 + (2.26) + (float('inf'))*7,
     (float('inf'))*10 + (1.02) + (float('inf'))*2 + (0.56) + (float('inf'))*6,
     (float('inf'))*14 + (0.64) + (float('inf'))*5,
     (float('inf'))*15 + (1.69) + (float('inf'))*4,
     (float('inf'))*16 + (1.13) + (float('inf'))*3,
     (float('inf'))*17 + (2.48) + (float('inf'))*2,
     (float('inf'))*18 + (0.85) + (float('inf'))*1,
     (float('inf'))*19
    ])

# Menampilkan Hasil
print_matrix(A, "Matriks A")

```

Figure 5 Python code for the min-plus product and the input of matrix A (Route 1)

```

# Perhitungan A* = (E ⊕ A ⊕ A^⊗2 ⊕ A^⊗3 ⊕ A^⊗4 ⊕ A^⊗5 ⊕ A^⊗6)
A_star = np.full((n, n), float('inf'))
for matrix in powers:
    for i in range(n):
        for j in range(n):
            A_star[i][j] = min(A_star[i][j], matrix[i][j])

# Menampilkan Matriks A*
print_matrix(A_star, "Matriks A*")

```

Figure 6 Python code for the calculation of A^*

```

# Membuat vektor b^e
b_e = np.array([0] + [float('inf')] * (A_star.shape[0] - 1)).reshape(-1, 1)

# Fungsi perkalian min-plus antara A_star dan b^e
def minplus_vector_multiplication(A, b):
    n = len(A)
    result = np.full((n, 1), float('inf'))
    for i in range(n):
        result[i] = min(A[i][j] + b[j][0] for j in range(n))
    return result

# Menghitung x^e
x_e = minplus_vector_multiplication(A_star, b_e)

# Menampilkan hasil x^e dalam bentuk transpose
x_e_transpose = x_e.T

# Menampilkan hasil
print("\nVektor x^e^T (Transpose dari A* ⊗ b^e):")
print("[ " + " ".join(f"{val:.2f}" if not np.isinf(val) else 'inf' for val in x_e_transpose[0]) + " ]")
print(x_e)

```

Figure 7 Python code for the minimum time vector

```

# Menghitung (Ae)T (Transpos dari Ae)
A_star_T = A_star.T

# Menampilkan Matriks (Ae)T
print_matrix(A_star_T, "(Ae)T")

# Menghitung xl = -((Ae)T ⊗ bl)
x_l = minplus_vector_multiplication(A_star_T, b_l)
x_l = - x_l

# Menampilkan hasil xl
print("\nVektor xl:")
print("[ " + " ".join(f"{val:.2f}" if not np.isinf(val) else 'inf' for val in x_l[:, 0]) + "]")
print(x_l)

```

Figure 8 Python code for the latest time vector

```

def find_strict_critical_path(A, x_e, x_l, start=1, end=8):
    from collections import defaultdict
    n = len(A)
    graph = defaultdict(list)

    # Buat graf dari matriks A
    for i in range(n):
        for j in range(n):
            if not np.isinf(A[i][j]):
                graph[i+1].append(j+1)

    x_e = x_e.flatten()
    x_l = x_l.flatten()

    # DFS hanya lewat simpul dengan xe == xl
    result_path = []

    def dfs(node, path):
        if x_e[node - 1] != x_l[node - 1]:
            return
        path.append(node)
        if node == end:
            result_path.append(path.copy())
        else:
            for neighbor in graph[node]:
                if x_e[neighbor - 1] == x_l[neighbor - 1]:
                    dfs(neighbor, path)
            path.pop()

    dfs(start, [])
    return result_path[0] if result_path else []

```

Figure 9 Python code for determining the shortest route

```

Vektor xe:
[0.00 2.96 5.32 7.61 10.12 11.78 13.79 15.98 17.82 18.63 21.49 12.70 13.16 14.40 14.96 15.60 17.29 18.42 20.82 21.67]

Vektor xl:
[-0.00 2.96 5.32 7.61 10.12 11.78 6.24 8.43 10.27 11.08 13.94 12.70 13.16 14.40 14.96 15.60 17.29 18.42 20.82 21.67]

Lintasan terpendek yang dilalui oleh bus feeder: v1 -> v2 -> v3 -> v4 -> v5 -> v6 -> v12 -> v13 -> v14 -> v15 -> v16 -> v17 -> v18 -> v19 -> v20

```

Figure 10 Output of the Python program: minimum time, latest time, and shortest route

3.5. Discussion

The results show where the time is gained. On Route 1 the two branches before v₉ (through v₃, v₄, v₅ or through v₆, v₇, v₈) differ by only 0.66 minutes, which is the value of $x^e - x^l$ at v₃ in Table 6, so the chosen branch is almost tied with the other. The decisive saving lies in the last segment, where $v_{12} \rightarrow v_{14} \rightarrow v_{15}$ takes 7.80 minutes against 18.00 minutes through v₁₃, and this explains why the slowest candidate in Table 3 takes 10.86 minutes longer than the shortest. Because the two upstream branches are nearly tied, the result was checked by recomputing all arc times as distance divided by 18 km/h; the total changes to 34.93 minutes but the route does not change. On Route 2 the connecting road $v_6 \rightarrow v_{12}$ (0.26 km) bypasses the longer branch through v₇ to v₁₁ (3.04 km); that branch would lengthen the trip by 7.55 minutes to 29.22 minutes, and the large difference between x^e and x^l at v₇ to v₁₁ in Table 6 shows the same.

Compared with previous studies, the routes found here follow the same logic as the min-plus applications to milk distribution (Suwanti et al., 2017), fish distribution (Sumianti et al., 2024), angkot routes in Bandung (Susilowati, 2018), and Trans Jogja bus routes (Susilowati & Fitriani, 2019): the vertices on the shortest route are those at which the earliest and latest times coincide (Definition 1.3). Floyd's algorithm (Rifanti & Arifwidodo, 2019), the Floyd–Warshall algorithm

(Khamami & Saputra, 2019), and the A* method (Pramudhita & Muljono, 2018) solve the same shortest-path problem, so on the same network they would be expected to give the same optimal route; the contribution of the min-plus formulation lies elsewhere. It gives, in one matrix computation, the minimum time to every stop and the time lost at every stop that is not on the route, and it is easily implemented in Python. Unlike the Semarang studies cited above, which produced route-search applications, this study uses the method to evaluate and propose feeder routes on real corridors.

The findings have implications for the feeder bus operator. The proposed Route 1 passes schools, places of worship, and markets on a corridor without feeder service, and the proposed Route 2 shortens the trip on a corridor that many residents already use. Both can be tested in the field before being adopted. Several limitations should be noted. The travel times are free-flow estimates from average speed and ignore congestion, traffic signals, and time of day; the networks contain only the roads selected by the authors; the time and distance of the current routes were not measured, so the saving relative to them cannot yet be quantified; and passenger demand and the suitability of the roads for buses were not assessed. Future research should use time-dependent weights, measure the current routes for a direct comparison, and add passenger demand to the model.

4. Conclusion

The three research questions are answered as follows. First, the two feeder bus routes were modeled as weighted directed simple graphs with travel time as the arc weight. Route 1, from the BNI UNNES bus stop to Bandarjo Market, has a 15×15 adjacency matrix, and Route 2, from the Polines bus stop to Banyumanik Terminal, has a 20×20 adjacency matrix. The minimum time vector $x^e = A^* \otimes b^e$ and the latest time vector $x^l = -((A^*)^T \otimes b^l)$ were computed from these matrices, and the shortest route consists of the vertices with $x^e_i = x^l_i$.

Second, the shortest route on Route 1 is $v_1 \rightarrow v_2 \rightarrow v_6 \rightarrow v_7 \rightarrow v_8 \rightarrow v_9 \rightarrow v_{10} \rightarrow v_{11} \rightarrow v_{12} \rightarrow v_{14} \rightarrow v_{15}$ (Table 1), with a travel time of 35.63 minutes and a distance of 10.48 km. It is not yet operated by a feeder bus and can be proposed as a new feeder route. The shortest route on Route 2 is $v_1 \rightarrow v_2 \rightarrow v_3 \rightarrow v_4 \rightarrow v_5 \rightarrow v_6 \rightarrow v_{12} \rightarrow v_{13} \rightarrow v_{14} \rightarrow v_{15} \rightarrow v_{16} \rightarrow v_{17} \rightarrow v_{18} \rightarrow v_{19} \rightarrow v_{20}$ (Table 4), with a travel time of 21.67 minutes and a distance of 6.14 km. It differs from the route currently operated and can be proposed as a route alternative.

Third, the computation was implemented in Python with `np.array()` for the matrix, a min-plus product function, the accumulation of A^* , the vectors x^e and x^l , the comparison $x^e_i = x^l_i$, and `nx.DiGraph()` with `nx.draw()` for the figures. The program reproduced the manual results and can be reused for larger networks. The operator is advised to test both proposed routes in the field, taking congestion and passenger demand into account.

REFERENCES

- Khamami, A., & Saputra, R. (2019). The shortest path search application based on the city transport route in Semarang using the Floyd-Warshall algorithm. *Journal of Physics: Conference Series*, 1217(1), Article 012116. <https://doi.org/10.1088/1742-6596/1217/1/012116>
- Pramudhita, A. C., & Muljono. (2018). Application of the nearest BRT bus stop search system in Semarang City using the Android-based A* method. *RESTI (Rekayasa Sistem dan Teknologi Informasi)*, 1(1), 19–25.
- Rifanti, U. M., & Arifwidodo, B. (2019). The implementation of Floyd algorithm in determining the shortest routes for tourism transportation. *Register: Jurnal Ilmiah Teknologi Sistem Informasi*, 5(2), 140–151. <https://doi.org/10.26594/register.v5i2.1683>
- Sumianti, Siswanto, & Wibowo, S. (2024). Application of min-plus algebra in determination of shortest route for distribution of pindang fish by Pindang Muri company. 6, 1–7.
- Suroto, S. (2022). Symmetrization of min-plus algebra. *Vygotsky*, 4(1), 35. <https://doi.org/10.30736/voj.v4i1.444>
- Susilowati, E. (2018). The comparison determining of some route of angkot in Bandung by using Greedy algorithm and min plus algorithm. *Lontar Komputer: Jurnal Ilmiah Teknologi Informasi*, 9(3), 182–191. <https://doi.org/10.24843/LKJITI.2018.v09.i03.p07>
- Susilowati, E., & Fitriani, F. (2019). Determining the shortest path between terminal and airport in Yogyakarta using Trans Jogja with min plus algorithm. *MUST: Journal of Mathematics Education, Science and Technology*, 4(2), 123–134. <https://doi.org/10.30651/must.v4i2.2966>
- Suwanti, V., Bintoto, P., & Dinullah, R. N. I. (2017). Application of min plus algebra in determining the fastest route for milk distribution. *Limits: Journal of Mathematics and Its Applications*, 14(2), 15. <https://doi.org/10.12962/limits.v14i2.3085>
- Watanabe, S., & Watanabe, Y. (2014). Min-plus algebra and networks. *RIMS Kôkyûroku Bessatsu*, 47, 41–54.