Polynom : Journal in Mathematics Education
Volume 3, Nomor 2, Juli 2023
Available online at: http://ejournal.uin-suka.ac.id/tarbiyah/polynom
Print ISSN : 2828-8521, Online ISSN : 2808-0777

Penentuan Rute Terpendek Stasiun Lempuyangan ke UIN Sunan Kalijaga dengan Algoritma Dijkstra

Indri Dwi Cahyaningtyas[1], Gading Julio Perkasa[*], Miranti [3], Vindy Antia[4], Jasmine Nurul Izzah[5], Burhanuddin Latif[6],
[1,2,3,4,5,6] Program Studi Pendidikan Matematika, Universitas Islam Negeri Sunan Kalijaga Yogyakarta
* Corresponding Author. E-mail: gadingjulio123@gmail.com

ABSTRAK

Penelitian ini bertujuan untuk menentukan rute terpendek dari Stasiun Lempuyangan ke UIN Sunan Kalijaga Yogyakarta menggunakan algoritma Dijkstra. Data berupa jarak aktual antar lokasi diambil dari Google Maps dan direpresentasikan dalam bentuk graf berbobot, dengan simpul sebagai lokasi dan sisi sebagai jarak antar lokasi. Algoritma Dijkstra bekerja dengan mencari jarak terpendek dari simpul awal ke simpul lainnya secara iteratif hingga mencapai simpul tujuan. Proses perhitungan dilakukan dengan bantuan aplikasi berbasis VB.Net. Hasil penelitian menunjukkan bahwa rute terpendek dari Stasiun Lempuyangan ke UIN Sunan Kalijaga adalah melalui A → B → D → E → H → K dengan bobot total 3,45. Implementasi algoritma Dijkstra terbukti efisien dan akurat dalam menentukan jalur terpendek, sehingga dapat menjadi solusi optimal untuk sistem navigasi dan pemetaan.
Kata Kunci: Algoritma Dijkstra, graf berbobot, rute terpendek, Stasiun Lempuyangan, UIN Sunan Kalijaga Yogyakarta.

ABSTRACT

This research aims to determine the shortest route from Lempuyangan Station to UIN Sunan Kalijaga Yogyakarta using Dijkstra's algorithm. Data in the form of actual distance between locations is taken from Google Maps and represented in the form of a weighted graph, with vertices as locations and edges as distances between locations. Dijkstra's algorithm works by finding the shortest distance from the starting node to other nodes iteratively until it reaches the destination node. The calculation process is done with the help of a VB.Net-based application. The results showed that the shortest route from Lempuyangan Station to UIN Sunan Kalijaga is through A → B → D → E → H → K with a total weight of 3.45. The implementation of Dijkstra's algorithm proved to be efficient and accurate in determining the shortest path, so it can be an optimal solution for navigation and mapping systems.
Keywords: *Dijkstra's algorithm; weighted graph; shortest route; Lempuyangan Station; UIN Sunan Kalijaga Yogyakarta*

http://dx.doi.org/10.14421/polynom.2023.32.56-62

PENDAHULUAN

Yogyakarta adalah salah satu provinsi yang terletak di Pulau Jawa yang juga merupakan salah satu kota tujuan pendidikan yang banyak menarik minat para perantau untuk datang dan melanjutkan pendidikan ke berbagai perguruan tinggi yang terdapat di Yogyakarta (Devinta dkk., 2024). Tidak heran jika Yogyakarta dikenal dengan sebutan Kota Pelajar. Menurut data Badan Pusat Statistik Provinsi Daerah Istimewa Yogyakarta tahun 2023, terdapat sebanyak 126 perguruan tinggi negeri maupun swasta yang ada di Yogyakarta. Banyaknya perguruan tinggi yang diyakini memiliki kualitas baik tersebut tentunya membuat mahasiswa yang datang untuk merantau ke Yogyakarta semakin banyak.

Mahasiswa yang datang ke Yogyakarta untuk melanjutkan pendidikan tentunya juga mempertimbangkan transportasi. Salah satu sarana transportasi yang digunakan oleh mahasiswa untuk datang ke Yogyakarta adalah kereta api. Stasiun Lempuyangan, sebagai salah satu pusat transportasi utama di Yogyakarta, menjadi titik awal perjalanan banyak pengguna kereta api, termasuk mahasiswa, tenaga pengajar, dan masyarakat umum. Salah satu tujuan populer dari lokasi ini adalah Universitas Islam Negeri (UIN) Sunan Kalijaga Yogyakarta, sebuah lembaga pendidikan yang berperan penting dalam pengembangan ilmu pengetahuan dan budaya.

Lokasi Stasiun Lempuyangan dan UIN Sunan Kalijaga Yogyakarta terletak di kawasan yang padat aktivitas dan fasilitas, sehingga terdapat berbagai rute yang dapat dilalui dari Stasiun Lempuyangan ke UIN Sunan Kalijaga Yogyakarta. Oleh karena itu perlunya mengetahui rute terpendek untuk sampai ke UIN Sunan Kalijaga Yogyakarta agar perjalanan lebih efektif. Menentukan jalur terpendek sangat penting untuk mengoptimalkan waktu yang digunakan serta memberikan efisiensi di berbagai bidang lainnya sehingga perjalanan menjadi lebih efektif, cepat, serta dapat menghemat biaya (Harahap & Khairina, 2017). Untuk menentukan rute terpendek maka dibutuhkan suatu algoritma.

Algoritma adalah serangkaian tahapan yang sering diterapkan dalam berbagai aktivitas sehari-hari (Panggabean dkk., 2021). Algoritma erat kaitannya dengan masalah rute terpendek, karena dalam menentukan rute terpendek, terdapat serangkaian tahapan yang disusun secara sistematis (Pane dkk., 2024). Algoritma yang paling sering digunakan untuk mencari jalur atau lintasan terpendek yaitu algoritma djikstra (Adinda, 2022). Oleh karena itu algoritma yang digunakan pada penelitian ini untuk menentukan jalur terpendek dari Stasiun Lempuyangan ke UIN Sunan Kalijaga Yogyakarta yaitu algoritma dijkstra.

Algoritma Dijkstra adalah salah satu metode yang digunakan untuk menemukan jalur terpendek dari suatu simpul ke simpul lainnya dalam graf yang hanya memiliki bobot positif (Muharrom, 2020). Dalam proses penentuan jalur terpendek pada suatu graf menggunakan algoritma Dijkstra, akan mendapatkan jalur yang terbaik karena pada waktu penentuan jalur yang akan dipilih, akan dianalisis bobot dari simpul yang belum terpilih, lalu dipilih simpul dengan bobot yang terkecil (Arga dkk., 2021). Algoritma Dijkstra dapat menghasilkan jalur terbaik, sehingga algoritma tersebut dipilih untuk menentukan rute terpendek pada penelitian ini.

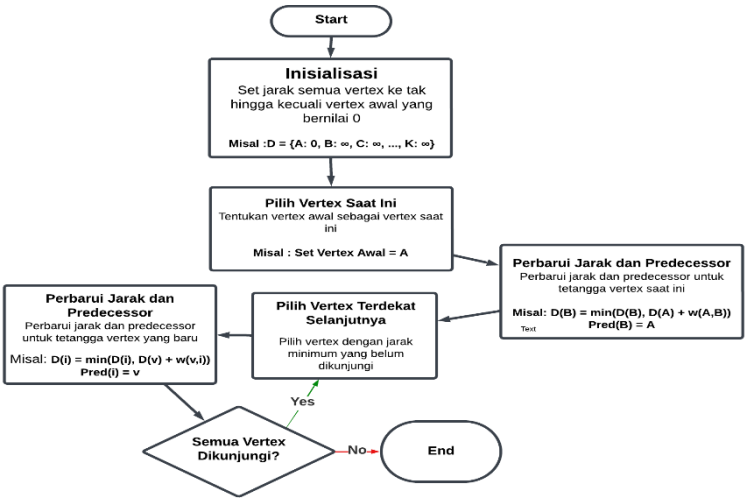
Berdasarkan uraian diatas, maka mahasiswa perlu untuk mengetahui rute terpendek dari Stasiun Lempuyangan ke UIN Sunan Kalijaga Yogyakarta. Pada penelitian ini akan dicari rute terpendek yang menghubungkan kedua tempat tersebut. Oleh karena itu penelitian berjudul "Penentuan rute terpendek Stasiun Lempuyangan ke UIN Sunan Kalijaga Yogyakarta dengan Algoritma Dijkstra. Tujuan dari penelitian ini adalah untuk menentukan rute terpendek dari Stasiun Lempuyangan ke UIN Sunan Kalijaga Yogyakarta dengan menerapkan algoritma dijkstra

METODE

Penelitian ini bertujuan untuk menyelesaikan persoalan jarak terpendek menggunakan Algoritma Dijkstra. Algoritma Dijkstra adalah salah satu algoritma yang paling efektif untuk menentukan rute terpendek dari suatu titik ke titik lainnya (Inayah dkk., 2023). Lokasi penelitian antara Stasiun Lempuyangan dan UIN Sunan Kalijaga, Yogyakarta. Data berupa jarak aktual diperoleh melalui Google Maps dalam satuan kilometer (km) dan direpresentasikan sebagai graf berbobot, di mana setiap node merepresentasikan lokasi, dan setiap edge memiliki bobot berupa jarak antar node.

Langkah-langkah penyelesaian dengan Algoritma Dijkstra menurut [2] meliputi:

1. Inisialisasi daftar jarak (tak hingga kecuali vertex awal bernilai nol), vertex sebelumnya, dan vertex yang belum dikunjungi.
2. Menetapkan vertex awal sebagai vertex saat ini, lalu memperbarui jarak dan vertex sebelumnya berdasarkan vertex yang dapat dicapai.
3. Memilih vertex berikutnya dengan jarak terpendek, mengulangi proses hingga semua vertex telah dikunjungi.
4. Hasil disimpan dan divisualisasikan pada peta untuk menampilkan jalur terpendek.



Gambar 1 Flowchart Algoritma Dijkstra

Untuk memastikan efisiensi, sistem memeriksa apakah jalur telah dihitung sebelumnya dalam database. Jika belum, proses dilakukan menggunakan Algoritma Dijkstra, hasilnya disimpan, lalu ditampilkan. Dengan metode ini, penelitian ini menawarkan solusi praktis untuk menentukan jalur terpendek secara efisien, sekaligus menjadi model pengembangan sistem informasi berbasis graf.

HASIL DAN PEMBAHASAN

A. *Analisis Algoritma Dijkstra*

Algoritma Dijkstra dianalisis dan diterapkan melalui perancangan aplikasi menggunakan bahasa pemrograman VB.Net. Aplikasi ini dirancang untuk mendukung proses penentuan jalur terpendek dengan cara yang sistematis dan efisien. Pada penerapannya, algoritma Dijkstra tidak hanya memperhitungkan simpul dan sisi dalam graf, tetapi juga mempertimbangkan beberapa faktor penting yang dapat mempengaruhi hasil pencarian jalur terpendek. Dalam proses mencari jalur terpendek menggunakan algoritma Dijkstra, terdapat dua faktor yang perlu diperhatikan agar hasilnya optimal untuk menemukan jalur dengan jarak total terkecil.

1. *Input Graph*

   Graf dapat didefinisikan sebagai pasangan himpunan (V, E), di mana V merupakan kumpulan simpul (node) seperti {v1, v2, v3,...,vn}, dan E adalah kumpulan sisi (edge) yang menghubungkan simpul-simpul tersebut, ditulis sebagai (e1, e2, …, en} (Fauzi, 2011). Berdasarkan arah pada sisi-sisinya, graf dapat dibagi menjadi dua tipe utama: graf berarah (*directed graph*), di mana setiap sisi memiliki orientasi tertentu, dan graf tak berarah (*undirected graph*), di mana sisi tidak menunjukkan arah khusus. Selain itu, graf juga dapat diklasifikasikan berdasarkan keberadaan bobot, yaitu graf berbobot (*weighted graph*), di mana sisi memiliki nilai yang menunjukkan bobot seperti jarak atau waktu, dan graf tak berbobot (*unweighted graph*), di mana sisi tidak memiliki nilai bobot (Kristo & Shandi, 2022)(Anshoriy, 2023).

   Dalam konteks aplikasi yang dirancang, graf yang digunakan harus disesuaikan dengan peta lokasi yang telah ditentukan melalui analisis sebelumnya. Setiap simpul merepresentasikan titik lokasi tertentu, sedangkan sisi mencerminkan hubungan antar simpul dengan nilai bobot yang relevan. Pendekatan ini sangat penting untuk memastikan algoritma Dijkstra dapat bekerja secara optimal dalam menentukan rute terpendek.

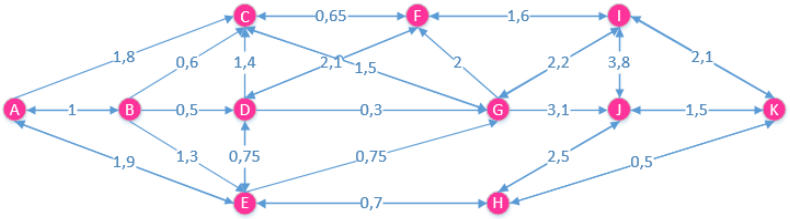
2. *Proses Graph*

   Pada algoritma Dijkstra, node menjadi elemen utama yang digunakan karena algoritma ini memanfaatkan struktur diagram pohon (*tree*) untuk menentukan jalur terpendek (Fauzi, 2011). Algoritma ini bekerja pada graf berarah (*directed graph*) untuk menghitung jarak minimum. Proses dimulai dengan mencari jarak terpendek dari node awal ke node terdekat, dilanjutkan ke node-node berikutnya hingga seluruh jalur dianalisis. Sebelum iterasi (i) dilakukan, algoritma telah mengidentifikasi jarak terpendek untuk (i-1) node yang paling dekat. Jika semua node memiliki bobot positif, algoritma dapat menemukan node terdekat berikutnya dari node asal selama node tersebut memiliki hubungan langsung dengan node (Ti), yaitu node yang saat ini sedang diproses.

Node-node yang memiliki keterhubungan langsung dengan node (Ti) inilah yang kemudian menjadi kandidat untuk langkah selanjutnya (Sembiring, 2021). Algoritma Dijkstra akan memilih node terdekat berikutnya berdasarkan bobot terkecil untuk melanjutkan proses pencarian jalur terpendek (Anwar dkk., 2024).

Proses pada graf ini melibatkan analisis terhadap struktur graf yang telah diinput sebelumnya, di mana setiap simpul (*node)* dan sisi (*edge*) dianalisis berdasarkan bobot yang menghubungkannya. Algoritma dimulai dengan menentukan simpul awal sebagai titik awal pencarian, kemudian menghitung jarak dari simpul tersebut ke simpul-simpul tetangga yang terhubung langsung. Setelah itu, simpul dengan jarak terkecil dipilih sebagai langkah berikutnya untuk melanjutkan proses ke simpul-simpul lain yang terhubung dengannya. Proses ini dilakukan secara iteratif hingga semua simpul dalam graf telah dianalisis, atau hingga simpul tujuan tercapai. Selama proses ini, algoritma memastikan bahwa setiap simpul yang telah dihitung jaraknya tidak akan diperiksa ulang, sehingga meminimalkan waktu dan sumber daya yang digunakan. Dengan pendekatan ini, algoritma Dijkstra dapat mengidentifikasi jalur terpendek secara sistematis dan optimal berdasarkan bobot yang ada pada graf (Pane dkk., 2024).

Graf dari Stasiun Lempuyangan sampai UIN Sunan Kalijaga diperoleh sebagai berikut :



Gambar 2 Graf dari Stasiun Lempuyangan ke UIN Sunan Kalijaga

Tabel 1. Node lokasi peta

Node	**Lokasi**
A	Stasiun Lempuyangan
B	UKDW
C	Jl. Herman
D	Jl. Kusbini
E	Jl. Bimokuthing
F	Jl. Colombo
G	Jl. Langensari
H	Jl. Timoho
I	Jl. Affandi
J	Jl. Laksda
K	UIN Sunan Kalijaga Yogyakarta

Langkah-langkah untuk menentukan jarak terpendek dari A ke F dengan menggunakan algoritma Dijkstra adalah sebagai berikut :

1. Awalnya, status setiap node yang belum dipilih diatur menjadi "0", sedangkan node yang sudah dipilih diatur menjadi "1", dimulai dari node A.
2. Bobot node yang memiliki hubungan langsung dengan node sumber, yaitu node A, ditentukan. Misalnya, bobot dari node A ke B adalah 1, dari A ke G adalah 1,8 dari A ke J adalah 1,9. Untuk node C, D, E, F, H, I, J dan K bobotnya diinisialisasi dengan "-" karena tidak ada lintasan langsung (arc) yang menghubungkan node-node tersebut dengan node A
3. *Predecessor* (pendahulu) dari node A, B, G, dan J adalah node A, karena jarak dihitung dari node A. Dengan demikian, node A disebut sebagai predecessor (node sumber). Sementara itu, untuk node C, D, E, F, H, I, J dan K predecessor-nya diinisialisasi dengan "-" karena tidak ada lintasan langsung yang menghubungkan node-node tersebut dengan node A, sehingga tidak ada jarak yang dapat dihitung.

Tabel 2. Hasil Iterasi

Iterasi ke-1											
Node	A	B	C	D	E	F	G	H	I	J	K
Status	1	0	0	0	0	0	0	0	0	0	0
Bobot	-	1	1,8	-	1,9	-	-	-	-	-	-
Predecessor	A	A	A	-	A	-	-	-	-	-	-
Iterasi ke-2											
Node	A	B	C	D	E	F	G	H	I	J	K
Status	1	1	0	0	0	0	0	0	0	0	0
Bobot	-	1	0,6	0,5	1,3	-	-	-	-	-	-
Predecessor	A	A	B	B	B	-	-	-	-	-	-
Iterasi ke-3											
Node	A	B	C	D	E	F	G	H	I	J	K
Status	1	1	0	1	0	0	0	0	0	0	0
Bobot	-	1	0,6	0,5	0,75	2,1	0,3	-	-	-	-
Predecessor	A	A	B	B	D	D	D	-	-	-	-
Iterasi ke-4											
Node	A	B	C	D	E	F	G	H	I	J	K
Status	1	1	0	1	0	0	1	0	0	0	0
Bobot	-	1	0,6	0,5	0,75	2,1	0,3	-	2,2	3,1	-
Predecessor	A	A	B	B	D	D	D	-	G	G	-
Iterasi ke-5											
Node	A	B	C	D	E	F	G	H	I	J	K
Status	1	1	0	1	0	0	1	0	1	0	0
Bobot	-	1	0,6	0,5	0,75	2,1	0,3	-	2,2	3,1	-
Predecessor	A	A	B	B	D	D	D	-	G	G	-
Iterasi ke-6											
Node	A	B	C	D	E	F	G	H	I	J	K
Status	1	1	0	1	0	0	1	0	1	0	1
Bobot	-	1	0,6	0,5	0,75	2,1	0,3	-	2,2	3,1	2,1
Predecessor	A	A	B	B	D	D	D	-	G	G	I
Iterasi ke-7											
Node	A	B	C	D	E	F	G	H	I	J	K
Status	1	1	1	1	0	0	1	0	1	0	1
Bobot	-	1	0,6	0,5	0,75	0,65	0,3	-	2,2	3,1	2,1
Predecessor	A	A	B	B	D	C	D	-	G	G	I
Iterasi ke-8											
Node	A	B	C	D	E	F	G	H	I	J	K
Status	1	1	1	1	0	1	1	0	1	0	1
Bobot	-	1	0,6	0,5	0,75	0,65	0,3	-	1,6	3,1	2,1
Predecessor	A	A	B	B	D	C	D	-	F	G	I
Iterasi ke-9											
Node	A	B	C	D	E	F	G	H	I	J	K
Status	1	1	1	1	1	1	1	0	1	0	1
Bobot	-	1	0,6	0,5	0,75	0,65	0,3	0,7	1,6	3,1	2,1
Predecessor	A	A	B	B	D	C	D	H	F	G	I
Iterasi ke-10											
Node	A	B	C	D	E	F	G	H	I	J	K
Status	1	1	1	1	1	1	1	1	1	0	1
Bobot	-	1	0,6	0,5	0,75	0,65	0,3	0,7	1,6	3,1	2,1
Predecessor	A	A	B	B	D	C	D	H	F	G	I
Iterasi ke-11											
Node	A	B	C	D	E	F	G	H	I	J	K
Status	1	1	1	1	1	1	1	1	1	1	1
Bobot	-	1	0,6	0,5	0,75	0,65	0,3	0,7	1,6	2,5	0,5
Predecessor	A	A	B	B	D	C	D	E	F	H	H

Proses iterasi dimulai dengan menetapkan node A sebagai node awal (status 1). Pada iterasi pertama, bobot awal dihitung untuk semua node yang terhubung langsung ke A, yaitu B dengan bobot 1, C dengan bobot 1.8, dan E dengan bobot 1.9, dengan A sebagai predecessor. Node dengan bobot terkecil, yaitu B, dipilih untuk iterasi berikutnya. Pada iterasi kedua, node B ditambahkan ke rute, dan bobot diperbarui untuk node yang terhubung dengannya. Bobot C diperbarui menjadi 0.6, D menjadi 0.5, dan E menjadi 1.3, semua dengan B sebagai predecessor. Node dengan bobot terkecil, yaitu D, dipilih untuk iterasi berikutnya.

Pada iterasi ketiga, node D ditambahkan ke rute, dan bobot diperbarui untuk node yang terhubung dengannya. Bobot E diperbarui menjadi 0.75, F dihitung menjadi 2.1, dan G menjadi

0.3, semua dengan D sebagai predecessor. Node G, dengan bobot terkecil, dipilih untuk iterasi berikutnya. Pada iterasi keempat, node G ditambahkan, dan bobot untuk node yang terhubung dengannya diperbarui. Bobot I menjadi 2.2 dan J menjadi 3.1, dengan G sebagai predecessor. Node dengan bobot terkecil berikutnya, yaitu I, dipilih. Iterasi dilanjutkan hingga iterasi kesebelas, di mana semua node telah diproses dan bobot terkecil untuk mencapai K telah ditemukan, yaitu 2.1 dengan H sebagai predecessor.

Berdasarkan iterasi terakhir, rute terpendek dari A ke K dapat ditentukan dengan melacak nilai Predecessor. Rute yang diperoleh adalah A → B → D → E → H → K, dengan bobot total 3.45. Proses ini menunjukkan bahwa algoritma berhasil menemukan rute terpendek melalui pembaruan bobot secara iteratif berdasarkan jalur dengan nilai minimum.

KESIMPULAN

Berdasarkan analisis dan implementasi algoritma Dijkstra dalam pencarian jalur terpendek dari node A (Stasiun Lempuyangan) ke node K (UIN Sunan Kalijaga), diperoleh hasil berupa rute terpendek A → B → D → E → H → K dengan bobot total 3.45. Proses iterasi menunjukkan efisiensi algoritma ini dalam menghitung jarak minimum secara sistematis, di mana setiap simpul (node) diproses satu per satu berdasarkan bobot terkecil hingga seluruh graf dianalisis. Implementasi algoritma Dijkstra yang mempertimbangkan graf berbobot memungkinkan pemilihan jalur optimal sesuai dengan bobot yang merepresentasikan jarak atau waktu, sehingga algoritma ini sangat cocok diterapkan dalam sistem navigasi atau pemetaan rute terpendek. Untuk penelitian selanjutnya, dapat dipertimbangkan penggunaan algoritma lain, seperti algoritma A* atau Bellman-Ford, guna membandingkan efisiensi dan akurasi pencarian jalur terpendek. Selain itu, arah balik dari UIN Sunan Kalijaga ke Stasiun Lempuyangan atau rute dengan titik awal yang berbeda, seperti stasiun lain atau universitas lain di Yogyakarta, dapat menjadi fokus penelitian lanjutan untuk memperkaya analisis dan meningkatkan aplikasinya dalam sistem transportasi atau navigasi berbasis graf yang lebih luas..

UCAPAN TERIMAKASIH

Kami sebagai penulis mengucapkan terima kasih kepada dosen mata kuliah Pemodelan Matematika atas bimbingan dan ilmu yang diberikan. Terima kasih juga disampaikan kepada Kelompok 2, yaitu Vindy, Gading, Miranti, Jasmine, dan Tyas, atas dukungan dan kerja sama yang luar biasa. Semoga karya ini bermanfaat dan dapat menjadi kontribusi positif dalam pengembangan ilmu pengetahuan.

Daftar Pustaka

Adinda, P. R. (2022). Aplikasi Algoritma Dijkstra Untuk Penentuan Jalan Terpendek. *Portaldata.org*, *2*(9), 1–11.

Anshoriy, W. F. (2023). *Penerapan metode algoritma Clarke and Wright Savings pada penentuan rute terpendek: Studi kasus di Kantor Pos Pemeriksa Kabupaten Blitar*. Universitas Islam Negeri Maulana Malik Ibrahim.

Anwar, S., Katili, M. R., & Padiku, I. R. (2024). Penerapan algoritma dijkstra dalam perancangan sistem informasi pencarian dan penyewaan kamar kost berbasiskan web. *Diffusion: Journal of System Information Technology*, *4*(2), 185–195.

Arga, E. S., Firmansyah, G. G., Imam, K., & Fauzi, M. (2021). Penerapan Algoritma Djikstra Pada Pencarian Jalur Terpendek. *Jurnal Bayesian : Jurnal Ilmiah Statistika dan Ekonometrika*, *1*(2), 134–142. https://doi.org/10.46306/bay.v1i2.15

Devinta, M., Hidayah, N., & Hendrastomo, G. (2024). Fenomena Culture Shock pada Mahasiswa Perantauan di Yogyakarta. *JKOMDIS : Jurnal Ilmu Komunikasi Dan Media Sosial*, *4*(2), 557–565. https://doi.org/10.47233/jkomdis.v4i2.1874

Fauzi, I. (2011). *Penggunaan algoritma dijkstra dalam pencairan rute tercepat dan rute terpendek: studi kasus pada jalan raya antara wilayah Blok M dan Kota*.

Harahap, M. K., & Khairina, N. (2017). Pencarian Jalur Terpendek dengan Algoritma Dijkstra. *SinkrOn*, *2*(2), 18–23. https://doi.org/10.33395/sinkron.v2i2.61

Inayah, A. M., Resti, N. C., & Ilmiyah, N. F. (2023). Analisa perbandingan algoritma floyd-warshall dan algoritma dijkstrauntuk penentuan rute terdekat. *Jurnal Ilmiah Matematika Realistik (JI-MR)*, *4*(2), 146–155.

Kristo, M., & Shandi, Y. J. (2022). Perancangan aplikais pencarian rute perjalanan angkutan kota (angkot) di kota bandung berbasis web menggunakan algotima a*. *Media Informatika*, *21*(2), 133–143.

Muharrom, M. (2020). Implementasi Algoritma Dijkstra Dalam Penentuan Jalur Terpendek Studi Kasus Jarak Tempat Kuliah Terdekat. *Indonesian Journal of Business Intelligence (IJUBI)*, *3*(1), 25–30. https://doi.org/10.21927/ijubi.v3i1.1229

Pane, J. A., Fitriani, I., & Lestari, M. (2024). Implementasi Algoritma Dijkstra Dalam Menentukan Rute Terpendek Menuju Museum Di Jakarta. *JIPETIK:Jurnal Ilmiah Penelitian Teknologi Informasi & Komputer*, *5*(1), 11–18. https://doi.org/10.26877/jipetik.v5i1.18583

Panggabean, S., Gata, W., Syarif, A. R., Rahmadani, S., & Widianto, T. (2021). Implementasi Algoritma Dijkstra Untuk Menentukan Jalur Terpendek Wilayah Pasar Minggu Dan STMIK Nusamandiri Jakarta. *Swabumi*, *9*(1), 78–85. https://doi.org/10.31294/swabumi.v9i1.9574

Sembiring, A. P. (2021). *Penggunaan Algoritma Djikstra Dalam Menentukan Rute Tercepat Kampus UMA 2 Menuju Kampus UMA 1*. Universitas Medan Area.