IMPLEMENTASI ALGORITMA DIJKSTRA DALAM PENENTUAN JALUR TERPENDEK MENUJU OBJEK WISATA DI KABUPATEN TEGAL

Syefudin[1)], Ahmad Muzakky Zain[2)], Gunawan[3)]
[1,2,3]Program Studi Teknik Informatika, STMIK YMI Tegal, Jl. Pendidikan No.1
Pesurungan Lor Margadana, Kota Tegal, Jawa Tengah, Indonesia
Email: syefudin5@gmail.com[1)] , muzakkyzain@gmail.com[2)]
gunawan.gayo@gmail.com[3)]
Nomor Telp : 085742711806[1)], 081225681248[2)], 08170990360[3)]
Asal Negara: Indonesia

ABSTRAK

Pencarian jalur terpendek pada sebuah graf berbobot mengacu pada pencarian jalur dengan jumlah bobot simpul terkecil, yang membentuk jalur tersebut. Oleh karena itu, jalur yang ditemukan merupakan jalur yang memiliki jumlah bobot simpul atau jarak yang paling minim. Pencarian jalur terpendek dilakukan pada pencarian objek wisata di Kabupaten Tegal yang akan membentuk suatu graf berarah dan berbobot. Sebelum menjalankan algoritma *Dijkstra*, dibutuhkan data jarak dari setiap simpul terlebih dahulu untuk menentukan jalur terpendek. Tujuan wisata yang menjadi favorit adalah lokasi yang diminati oleh penduduk setempat maupun para wisatawan. Berdasarkan hasil survei, terdapat satu objek wisata yang paling populer di Kabupaten Tegal, yaitu Objek Wisata Guci. Bobot tiap simpul diambil dari *geolocation* objek wisata Guci. Dari hasil proses algoritma *Dijkstra* dapat diketahui jalur yang dihasilkan oleh algoritma *Dijkstra* sudah cukup akurat atau tepat. Kemudian dari data jarak yang didapatkan, kemudian dihitung waktu tempuh menuju objek wisata Guci dengan memasukkan variabel bobot kemacetan dan rata-rata kecepatan yang diasumsikan dari kendaraan roda 2 dan roda 4. Pada penelitian ini, perhitungan jalur terpendek rute terpendek yang didapatkan dari titik awal di Taman Rakyat Slawi menuju Objek Wisata Guci sejauh 28,7 km. Sedangkan data waktu tempuh didapat dari kondisi jalan yang diasumsikan normal tanpa hambatan kecuali bobot kemacetan.

Kata kunci: Algoritma *Dijkstra*, Jalur Terpendek, Objek Wisata, Graf, Waktu Tempuh

ABSTRACT

Searching for the shortest path in a weighted graph refers to finding the path with the smallest number of node weights that make up the path. Therefore, the path found is the path that has the minimum number of node weights or distances. The search for the shortest path is carried out in the search for tourist objects in Tegal Regency which will form a directed and weighted graph. Before running Dijkstra's algorithm, distance data from each node is needed to determine the shortest path. The intended tourist object is a tourist object that is of interest to residents or tourists, there is one tourist object that is most in demand in Tegal Regency according to the survey results, namely, the Jar Tourism Object. The weight of each node is taken from the geolocation of the Guci tourist attraction. From the process results of Dijkstra's algorithm, it can be seen that the path generated by Dijkstra's algorithm is quite accurate or precise. Then from the distance data obtained, then the travel time to the Guci tourist attraction is calculated by entering the variable congestion weight and the assumed average speed of 2-wheeled and 4-wheeled vehicles. In this study, the shortest route calculation was obtained from the starting point at Slawi People's Park to the Jar Tourism Object as far as 28.7 km. While the travel time data is obtained from road conditions which are assumed to be normal without obstacles except for the weight of congestion.

Keywords: Climate Change, Community Environmental Disputes, Global Warming, Pollution, Environmental Degradation.

1. PENDAHULUAN

Objek wisata adalah suatu tempat yang dikunjungi oleh seseorang untuk menikmati keindahan alam semesta dan melepaskan kejenuhan dari rutinitas dalam kehidupan sehari-hari, tempat-tempat wisata ini juga menjadi pilihan yang sempurna untuk berlibur bersama keluarga(Pratama et al., 2020). Dengan banyaknya objek wisata yang ada di Kabupaten Tegal, tentunya perlu adanya informasi mengenai jalur terpendek untuk menuju objek wisata yang diinginkan. Namun, masalah yang sering dihadapi adalah sulitnya wisatawan untuk menemukan jalur terpendek menuju objek wisata yang dituju. Hal ini dapat berdampak pada menurunnya minat wisatawan untuk berkunjung ke objek wisata yang ada di Kabupaten Tegal (Pintoko & L., 2018)

Untuk mencari jalur terpendek terdapat beberapa algoritma yang dapat digunakan, seperti penelitian tentang penerapan algoritma *Dijkstra* dan

algoritma semut atau *Ant Colony* untuk membandingkan kedua algoritma yang diterapkan menghasilkan akurasi algoritma *Dijkstra* dan algoritma *Ant Colony* sama-sama baiknya, namun algoritma *Dijkstra* memerlukan bobot antar titik tetapi proses perhitunganya lebih cepat yaitu dibandingkan algoritma *Ant Colony* yang tidak memerlukan bobot dan mengunjungi tiap titik untuk mencari data jarak tetapi waktu perhitunganya lebih lama (Sunardi et al., 2019). Algoritma *Dijkstra* dapat digunakan untuk mengoptimalkan penjadwalan pengiriman catering ke 5 pabrik dalam satu kali pengantaran mengacu pada jarak terpendek, dengan menghitung jarak menggunakan persamaan *Haversine Formula* (Putri et al., 2020). Penerapan algoritma *Dijkstra* juga dilakukan untuk merancang aplikasi C-Hos (Covid-19 Hospital) berbasis android yang bertujuan untuk memberikan informasi yang membantu masyarakat dalam mencari jalur terpendek menuju rumah sakit rujukan Covid-19 (Ismail et al., 2021). Selain itu, algoritma *Dijkstra* juga dapat menjadi pilihan yang tepat untuk diterapkan dalam pembuatan peta UNTAR karena dapat menghitung jarak terpendek antara dua titik tertentu untuk memberikan jalur tercepat bagi user ketika menggunakan peta tersebut (Sutrisno et al., 2020).

Algoritma *Dijkstra* merupakan metode yang dipakai dalam menangani permasalahan jarak terpendek pada suatu graf yang memiliki arah (Cantona et al., 2020). Pada algoritma *Dijkstra*, dilakukan operasi pembuatan sebuah pohon jalur terpendek dari titik pertama hingga mencapai titik akhir (Yosua et al., 2021). Pemilihan algoritma *Dijkstra* untuk menentukan jalur terpendek menjadi pilihan yang tepat, karena algoritma *Dijkstra* merupakan sebuah metode yang sangat efektif dalam menemukan jarak terpendek antara dua lokasi tertentu dengan prinsip dasar dari algoritma *Dijkstra* yaitu menentukan dua jalur terpendek yang ada di antara lokasi awal dan lokasi tujuan (Galih & Krisdiawan, 2018). Namun, algoritma *Dijkstra* hanya dapat digunakan untuk menentukan jalur terpendek jika bobot yang digunakan adalah bilangan positif, jika terdapat bobot yang bernilai negatif, maka algoritma ini tidak dapat digunakan (Herlambang et al., 2022). Pencarian rute terpendek algoritma *Dijkstra* hanya bisa mencari 1 rute tujuan dalam satu proses pencarian (Yuswardi & Balia, 2022).

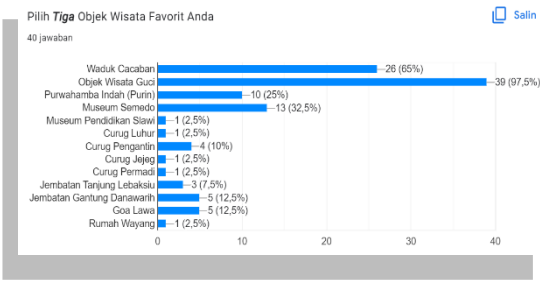
Dari permasalahan yang ada, karakteristik algoritma *Dijkstra* perlu dianalisa untuk diketahui bagaimana penerapanya dalam mencari rute terpendek. Hasil penerapan algoritma *Dijkstra* dalam menentukan jalur terpendek menuju objek wisata di Kabupaten Tegal diharapkan tidak hanya memberikan rincian jarak terpendeknya saja, melainkan juga informasi waktu tempuh ke titik tujuan atau wisata yang dituju. Sehingga dengan adanya informasi jalur terpendek, biaya transportasi dan waktu akan lebih efisien (Erik Prasetyo & Marisa, 2021).

2. METODE

Penelitian ini dilakukan di Kabupaten Tegal, dimana objek yang akan diteliti yaitu, objek wisata yang ada di Kabupaten Tegal. Dengan banyaknya objek wisata diperlukan informasi bagi wisatawan yang ingin menuju objek wisata favoritnya atau objek wisata yang ingin dituju. Oleh karena itu, dibutuhkan observasi data dan pengumpulan data yang kemudian akan diimplementasikan ke dalam algoritma *Dijkstra* untuk mendapatkan informasi yang dibutuhkan, yaitu berupa data jalur terpendek dan waktu tempuh yang dapat digunakan sebagai bahan pertimbangan untuk menuju objek wisata.

2.1 Pengumpulan Data

Penelitian ini diawali dengan menganalisa kebutuhan data, yaitu data mengenai objek wisata di Kabupaten Tegal yang sering didatangi oleh masyarakat. Oleh karena itu, survey dilakukan dengan menggunakan tools *Google Form*.



Gambar 1. Tools google form

Penggunaan *Google Form* memudahkan observasi atau pengumpulan data objek wisata terfavorit atau yang ingin dikunjungi. Pada gambar 1, terlihat hasilnya terdapat 40 responden yang memilih 1 dari 13 objek wisata yang dijadikan pilihan yaitu Waduk Cacaban, Objek Wisata Guci, Pantai Purwahamba Indah (Purin), Museum Semedo, Museum Pendidikan Slawi, Curug Luhur, Curug Pengantin, Curug Jejeg, Curug Permadi Jembatan Gantung Danawarih, Jembatan Tanjung Lebaksiu, Goa Lawa dan Rumah Wayang. Terdapat satu objek wisata yang sering dikunjungi atau diminati masyarakat, yaitu 97,5% orang memilih Objek Wisata Guci.

2.2 Jalur Terpendek (*Shortest Graph*)

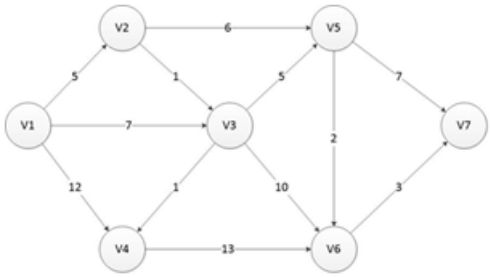
Jalur terpendek adalah jalur yang menghubungkan simpul asal dan simpul tujuan dengan mempertimbangkan jarak dan waktu tempuh tercepat (Darnita & Toyib, 2019). Shortest path problem (SPP) merupakan penyelesaian masalah yang bertujuan untuk menemukan jalur terpendek antara dua atau lebih simpul pada graf berbobot, bobot sisi-sisi yang dilewati harus dijumlahkan sehingga menghasilkan total bobot yang paling minimum (Hutapea et al., 2021). Untuk menghitung jalur terpendek, digunakan graf yang memiliki bobot

(*weighted graph*), bobot graf dihitung berdasarkan jarak dan waktu tempuh antara simpul-simpul yang terdapat dalam graf (Marlim et al., 2020). Sesuai dengan hasil pengumpulan data pada gambar 1, jalur terpendek yang akan dicari yaitu jalur terpendek menuju Objek Wisata Guci. Pencarian jalur terpendek menjadi kasus yang sering diterapkan bersama dengan algoritma *Dijkstra*, karena algoritma *Dijkstra* adalah suatu metode yang efektif dalam menemukan jalur terpendek.

2.3 Algoritma *Dijkstra*

Algoritma *Dijkstra* dapat dimanfaatkan dalam menangani masalah jarak terpendek pada sebuah graf yang memiliki arah (Chatrin Bunaen et al., 2022). Algoritma *Dijkstra* juga merupakan algoritma yang umumnya dimanfaatkan untuk menyelesaikan masalah yang berkaitan dengan optimasi atau biasa disebut algoritma *Greedy* (Chatrin Bunaen et al., 2022).

Edsger W. *Dijkstra* merupakan penemu teori algoritma *Dijkstra* melalui artikelnya dengan judul "A Note on Two Problems in Connexion with Graphs" diterbitkan di jurnal Numerische Mathematik, Volume 1, Halaman 269-271 pada tahun 1959 (Marlim et al., 2020).



Gambar 1. Ilustrasi Algoritma *Dijkstra*

Pencarian rute terpendek menggunakan algoritma *Dijkstra* memerlukan sebuah bobot setiap simpulnya. Gambar 2 memperlihatkan sebuah graf pencarian rute terpendek dari V1 menuju ke V7 dengan beberapa titik rute yang memiliki bobot pada setiap simpul yang menghubungkan setiap titik.

Algoritma *Dijkstra* menjalankan suatu algoritma dengan menemukan rute terpendek ke satu simpul optimum pada tiap tahapnya, dimana pada tahap ke-n, akan ada n simpul yang memiliki jalur terpendek yang diketahui, algoritma secara bertahap memperbarui jalur terpendek ke setiap simpul saat bergerak dari simpul ke simpul untuk menemukan jalur terpendek dari titik awal ke simpul tujuan (Ismail et al., 2021). Untuk lebih jelasnya, berikut ini urutan untuk menghitung jarak terpendek menggunakan algoritma *Dijkstra*:

1. Tentukan simpul awal dan akhir yang ingin dicari jarak terpendeknya.
2. Inisialisasi semua simpul dengan jarak tak terhingga (inf) kecuali simpul awal, yang diinisialisasi dengan jarak 0.
3. Tentukan simpul dengan jarak terdekat dari simpul awal yang belum dikunjungi. Simpul ini akan menjadi simpul saat ini.
4. Untuk setiap simpul tetangga dari simpul saat ini, hitung jarak total dari simpul awal ke simpul tetangga. Jika jarak total lebih kecil dari jarak sebelumnya, perbarui jarak simpul tetangga.
5. Tandai simpul saat ini sebagai telah dikunjungi.
6. Ulangi langkah 3 - 5 sampai semua simpul telah dikunjungi atau simpul akhir sudah ditemukan.
7. Jika simpul akhir ditemukan, jarak terpendek adalah jarak dari simpul awal ke simpul akhir. Jika tidak, tidak ada jalur dari simpul awal ke simpul akhir.

Dari langkah-langkah di atas, setelah diketahui titik yang saling terhubung pada graf dan bobot tiap simpulnya, maka kita dapat menghitung bobot terpendek menuju titik tujuan rumus:

$$\boldsymbol{D(vi) = \min(D(vj), D(vi) + W(vi, vj))}$$

Rumus 1. Bobot Terpendek

Rumus 1 menunjukan bagaimana suatu bobot akan dihitung untuk menentukan titik mana yang akan disimpan sebagai tujuan berikutnya, dimana $D(v_i)$ merupakan jumlah bobot terkecil dari v_i ke v_j dan $W(v_i,v_j)$ merupakan bobot simpul dari v_i ke v_j yang akan dihitung dengan menghitung minimal bobot terkecil dari $D(v_i)$ menuju $D(v_j)$. Rumus tersebut diterapkan pada setiap titik yang ada untuk kemudian dicari bobot yang terkecil. Untuk menerapkan rumus tersebut, dalam mencari jalur terpendek menggunakan algoritma *Dijkstra* terapkan:

1. Definisikan L = {} dan V = {V1, V2, V3, V4, ….,Vn},
2. Untuk iterasi berikutnya lakukan $D(v_i) = W(v_i,v_j)$,
3. Apabila Vn belum menjadi titik tetap atau Vn ∉ L maka :
   a. Pilih titik $v_k \in V - L$ dengan $D(v_k)$ terkecil, lalu $L = L \cup \{v_k\}$ maka jadikan v_k sebagai titik permanen.
   b. Untuk setiap $v_j \in V - L$ terapkan, jika $D(v_k) + W(v_k,v_j) < D(v_j)$ ubah $D(v_j)$ dengan $D(v_k) + W(v_k,v_j)$.

2.4 Waktu Tempuh

Pengambilan keputusan dalam memilih jalur menuju suatu tempat tidak hanya berpatokan pada pencarian rute atau jalur terpendeknya saja, perhitungan waktu tempuh menjadi hal yang penting dalam mengestimasi waktu menuju sebuah titik atau lokasi yang ingin dituju. Waktu tempuh menjadi dapat digunakan untuk mengetahui efisiensi sebuah jalur yang akan dilalui setelah mengetahui jalur terpendek menuju suatu tempat.

$$Waktu = \frac{Jarak}{(Kecepatan\ x\ Bobot\ Kemacetan)}\ x\ 60$$

Rumus 2. Waktu Tempuh

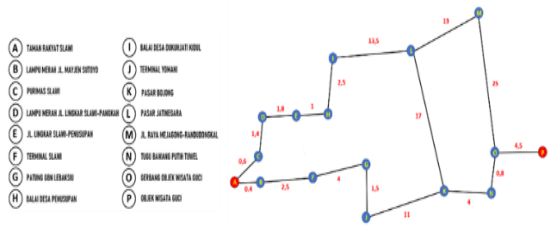
Setelah jarak terpendek diketahui, selanjutnya untuk menentukan waktu tempuh jarak akan dibagi dengan rata-rata kecepatan dikalikan dengan bobot kemacetan seperti yang ditunjukan pada rumus 2.

3. HASIL DAN PEMBAHASAN

3.1 Pembentukan Graf

Dari hasil survey kemudian dilakukan *geolocation* atau pembentukan jalur menggunakan Google Maps dengan titik awal di Taman Rakyat Slawi yang akan diuji pada algoritma *Dijkstra* dengan pengujian pada 2 objek wisata yang paling diminati.

Pembentukan jalur atau graf menuju Objek Wisata Guci, menunjukan 16 titik dengan titik A sebagai titik awal yaitu di Taman Rakyat Slawi hingga ke titik P sebagai tujuan akhir yaitu Objek Wisata Guci, dimana setiap titik disimbolkan dengan huruf yang mendefinisikan setiap tempat yang akan dilalui.



Gambar 2. Jalur Menuju Objek Wisata Guci

Setelah dilakukan *geolocation* menggunakan Google Maps maka akan terlihat bahwa, dari titik ke titik memiliki jarak yang berbeda-beda. Satuan jarak yang digunakan adalah kilometer (km). Ketiga data jalur beserta jaraknya kemudian akan diproses kedalam algoritma.

3.2 Implementasi Algoritma *Dijkstra*

Dari pembentukan graf yang telah dilakukan untuk objek wisata pilihan sesuai data yang didapatkan, proses selanjutnya yaitu mencari jalur terpendek suatu simpul secara bertahap hingga menemukan simpul terpendek menggunakan rumus seperti pada rumus 1 untuk menghitung bobot terpendek tiap simpul.

Data bobot jalur yang akan dilalui menuju objek wisata Guci dapat dilihat pada graf pada gambar 3. Dengan demikian dapat dilakukan perhitungan dengan perhitungan sebagai berikut:

1. L = {A}

D(A)	= 0	D(E)	= ∞ D(I) = ∞D(M) =∞
D(B)	= ∞	D(F)	= ∞ D(J)= ∞ D(N) = ∞
D(C)	= ∞	D(G)	= ∞ D(K)= ∞D(O) = ∞
D(D)	= ∞	D(H)	= ∞ D(L)= ∞D(P) = ∞

Rute terpendek dari titik A adalah titik B dan C dengan $D_{(A)} = 0$, $D_{(B)} = \infty$ dan $D_{(C)} = \infty$ maka:

$D_{(B)}$ = min($D_{(B)}$,$D_{(A)}$ + $W_{(A,B)}$) = min(∞,0+0,4) = min(∞,0,4) = 0,4

$D_{(C)}$ = min($D_{(C)}$,$D_{(A)}$ + $W_{(A,C)}$) = min(∞,0+0,6) = min(∞,0,6) = 0,6

Nilai $D_{(B)} < D_{(C)}$ sehingga titik B terpilih sebagai titik selanjutnya.

2. Iterasi 1:
   L = {A,B}
   $D_{(F)}$ = min($D_{(F)}$,$D_{(B)}$ + $W_{(B,F)}$) = min(∞,0,4+2,5) = min(∞,2,9) = 2,9
   Dari hasil iterasi hanya titik F yang terhubung pada titik B dengan nilai $D_{(F)}$ = 2,9 maka titik F terpilih sebagai titik selanjutnya.
3. Iterasi 2:
   L = {A,B,F}
   $D_{(G)}$ = min($D_{(G)}$, $D_{(F)}$ + $W_{(F,G)}$) = min(∞,2,9+4) = min(∞,6,9) = 6,9
   Dari hasil iterasi hanya titik G yang terhubung pada titik F dengan nilai $D_{(G)}$ = 6,9 maka titik G terpilih sebagai titik selanjutnya.
4. Iterasi 3 :
   L = {A,B,F,G}
   $D_{(J)}$ = min($D_{(J)}$, $D_{(G)}$ + $W_{(G,J)}$) = min(∞,6,9+1,5) = min(∞,8,4) = 8,4
   Dari hasil iterasi hanya titik J yang terhubung pada titik G dengan nilai $D_{(J)}$ = 8,4 maka titik J terpilih sebagai titik selanjutnya.
5. Iterasi 4 :
   L = {A,B,F,G,J}
   $D_{(K)}$ = min($D_{(K)}$, $D_{(J)}$ + $W_{(J,K)}$) = min(∞,8,4+11) = min(∞,19,4) = 19,4
   Dari hasil iterasi hanya titik K yang terhubung pada titik J dengan nilai $D_{(K)}$ = 19,4 maka titik K terpilih sebagai titik selanjutnya.
6. Iterasi 5 :
   L = {A,B,F,G,J,K}
   $D_{(L)}$ = min($D_{(L)}$, $D_{(K)}$ + $W_{(K,L)}$) = min(∞,19,4+17) = min(∞,36,4) = 36,4
   $D_{(N)}$ = min($D_{(N)}$, $D_{(K)}$ + $W_{(K,N)}$) = min(∞,19,4+4) = min(∞,23,4) = 23,4
   Dari hasil iterasi terdapat 2 titik yang terhubung dengan titik K yaitu titik L dan N dengan nilai $D_{(N)} < D_{(L)}$ maka titik N terpilih sebagai titik selanjutnya.
7. Iterasi 6 :
   L = {A,B,F,G,J,K,N}
   $D_{(O)}$ = min($D_{(O)}$, $D_{(N)}$ + $W_{(N,O)}$) = min(∞,23,4+0,8) = min(∞,24,2) = 24,2
   Dari hasil iterasi hanya titik O yang terhubung pada titik N dengan nilai $D_{(O)}$ =

24,2 maka titik O terpilih sebagai titik selanjutnya.

8. Iterasi 7 :
   L = {A,B,F,G,J,K,N,O}
   $D_{(P)}$ = min($D_{(P)}$, $D_{(O)}$ + $W_{(O,P)}$) = min(∞,24,2+4,5) = min(∞,28,7) = 28,7

Dari hasil iterasi hanya titik P yang terhubung pada titik O dengan nilai D(P) = 28,7 yang sekaligus menjadi jalur terakhir menuju objek wisata Guci di titik P dengan L = {A, B, F, G, J, K, N, O, P}, maka jalur terpendek menuju objek wisata Guci adalah dari titik A→B→F→G→J→K→N→O→P dengan jarak tempuh 28,7 km.

3.3 Analisis Hasil

Setelah mengetahui seluruh data jalur terpendek menuju objek wisata Guci di Kabupaten Tegal, selanjutnya akan dihitung waktu tempuh perjalanan menuju objek wisata dengan ketentuan bobot kemacetan yang dikelompokkan menjadi 3 kategori yaitu lancar, ramai lancar dan macet dengan asumsi bobot kemacetan.

Tabel 1. Bobot Kemacetan

Kondisi Jalan	Bobot Kemacetan
Lancar	1
Ramai Lancar	0,75
Macet	0,5

Dari kategori asumsi bobot kemacetan yang sudah ada pada Tabel 1, kemudian dengan melihat kondisi jalur yang ada diasumsikan bahwa kecepatan rata-rata berkendara untuk kendaraan roda 2, yaitu 60 km/jam dengan jalan lancar, 50 km/jam dengan jalan ramai lancar dan 40km/jam dengan jalan macet. Kecepatan rata-rata berkendara untuk kendaraan roda 4 yaitu 50km/jam dengan jalan lancar, 40km/jam dengan jalan ramai lancar dan 30km/jam dengan jalan macet. Waktu tempuh dapat dihitung setelah mengetahui jarak tempuh, bobot kemacetan dan kecepatan rata-rata berkendara dengan rumus 2.

Setelah dilakukan penerapan algoritma *Dijkstra* untuk menemukan jalur terpendek menuju objek wisata Guci yang paling diminati di Kabupaten Tegal, didapati hasil jalur yang akan dilewati oleh wisatawan beserta jenis kendaraan dan waktu tempuhnya.

Tabel 2. Jalur Terpendek dan Waktu Tempuh Roda 2

Jenis Kendaraan	Objek Wisata	Jalur Terpendek	Jarak (km)	Kondisi Jalan	Waktu (Menit)
Roda 2	Objek Wisata Guci	A→B→F→G→J→K→N→O→P	28,7	Lancar	29
				Ramai Lancar	46
				Macet	86
Roda 4	Objek Wisata Guci	A→B→F→G→J→K→N→O→P	28,7	Lancar	34
				Ramai Lancar	57
				Macet	115

Untuk menuju 2 objek wisata Guci menggunakan kendaraan roda 2 dan roda 4, jarak akan selalu berbanding lurus dengan waktu tempuh, ketika jarak semakin jauh maka waktu tempuhnya juga akan semakin lama, namun juga waktu akan dipengaruhi oleh kecepatan berkendara dan bobot kemacetan yang ada. Tabel 2 menunjukan hasil perhitungan waktu tempuh setelah jarak terpendek diketahui. Perhitungan waktu tempuh yang ada merupakan perhitungan dengan nilai kecepatan berkendara dan kondisi jalan (bobot kemacetan) yang diasumsikan.

Hasil dari penelitian ini menunjukan pencarian jalur terpendek menggunakan algoritma *Dijkstra* dapat dihitung dengan akurasi yang baik, dengan bantuan *geolocation* dari Google Maps dalam melakukan pembobotan tiap simpul. Untuk memberikan informasi yang lebih mengenai objek wisata terdekat, disajikan hasil perhitungan waktu tempuh yang berbeda-beda tergantung pada jarak yang ditempuh dan bobot kemacetanya.

4. KESIMPULAN DAN SARAN

Penerapan algoritma *Dijkstra* pada kasus penentuan jalur terpendek terbukti berhasil. Dengan pembobotan pada graf yang kemudian setiap simpul diminimalisasikan kemudian setiap simpul terpendek akan dipilih sebagai simpul berikutnya hingga terhubung ke simpul akhir. Bobot yang digunakan adalah bobot dengan nilai positif, jika bernilai negatif maka jalur terpendek tidak dapat dihitung. Dalam penelitian ini, dari perhitungan jalur terpendek Rute terpendek yang didapatkan dari titik awal di Taman Rakyat Slawi menuju Objek Wisata Guci yaitu sejauh 28,7 km.

Dengan data rute terpendek yang sudah didapatkan maka dapat ditentukan waktu tempuh dengan bobot kemacetan dan kecepatan rata-rata yang diasumsikan. Data waktu tempuh yang didapatkan merupakan data waktu tempuh perjalanan secara umum atau dengan kondisi jalan yang normal tanpa hambatan dengan bobot kemacetan dan kecepatan rata-rata yang

diasumsikan, sehingga perlu data yang lebih akurat mengenai hambatan-hambatan dijalan dan yang akan mempengaruhi dalam menentukan waktu tempuh. Diharapkan perhitungan algoritma *Dijkstra* yang telah diterapkan dapat diimplementasikan pada sebuah sistem untuk dapat digunakan dikemudian hari.

DAFTAR PUSTAKA

Cantona, A., Fauziyah, & Winarsih. (2020). Implementasi Algoritma Dijkstra Pada Pencarian Rute Terpendek ke Museum di Jakarta. *Jurnal Teknologi Dan Manajemen Informatika*, *6*(1), 27–34.

Chatrin Bunaen, M., Pratiwi, H., & Finsensia Riti, Y. (2022). Penerapan Algoritma Dijkstra Untuk Menentukan Rute Terpendek Dari Pusat Kota Surabaya Ke Tempat Bersejarah. *4*(1), 213–223. https://doi.org/https://doi.org/10.47233/jteksis.v4i1.407 ISSN

Darnita, Y., & Toyib, R. (2019). Penerapan Algoritma Greedy Dalam Pencarian Jalur Terpendek Pada Instansi-Instansi Penting Di Kota Argamakmur Kabupaten Bengkulu Utara. *Jurnal Media Infotama*, *15*(2), 57. https://doi.org/https://dx.doi.org/10.37676/jmi.v15i2.867

Erik Prasetyo, L., & Marisa, F. (2021). Sistem Optimasi Pendistribusian Bahan Makanan dan Snack dengan Algoritma Ant Colony Optimization (ACO). *AITI: Jurnal Teknologi Informasi*, *18*(1), 88–96. https://doi.org/https://dx.doi.org/10.24246/aiti.v18i1.88-96

Galih, E. C., & Krisdiawan, R. A. (2018). Implementasi Algoritma Dijkstra Pada Aplikasi Wisata Kuningan Berbasis Android. *Nuansa Informatika*, *12*(1). https://doi.org/10.25134/nuansa.v12i1.1344

Herlambang, B. A., Pradana, S. O., & Waliyansyah, R. R. (2022). Sistem Informasi Geografis Pencarian Rute Optimum Lokasi Obyek Wisata Kota Semarang Menggunakan Algoritma Dijkstra. *Journal of Applied Science and Technology*, *2*(01), 16. https://doi.org/10.30659/jast.2.01.16-25

Hutapea, Y. P., Montolalu, C. E. J. C., & Komalig, H. A. H. (2021). Algoritma Dijkstra untuk Penentuan Lintasan Terpendek Pada Kasus Tujuh Hotel di Kota Manado Menuju Bandara Sam Ratulangi Manado. *D'Cartesian*, *9*(2), 158. https://doi.org/10.35799/dc.9.2.2020.29146

Ismail, M. T., Andryana, S., & Gunaryati, A. (2021). Implementasi Algoritma Dijkstra untuk Pencarian Rute Terpendek Menuju Rumah Sakit pada Aplikasi C-Hos (Covid-19 Hospital). *Jurnal Media Informatika Budidarma*, *5*(3), 888. https://doi.org/10.30865/mib.v5i3.3077

Marlim, Y. N., Jollyta, D., & Saputra, F. (2020). *Analisis Sistem Jalur Terpendek Menggunakan Algoritma Djikstra dan Evaluasi Usability*. *6*(1), 54–60. https://doi.org/http://dx.doi.org/10.26418/jp.v6i1.37627

Pintoko, B. M., & L., K. M. (2018). Analisis Sentimen Jasa Transportasi Online pada Twitter Menggunakan Metode Naive Bayes Classifier. *E-Proceeding of Engineering*, *5*(3), 8121–8130.

Pratama, Z., Hartama, D., Ridwan Lubis, M., Retno Andani, S., & Okta Kirana, I. (2020). Penerapan Metode Dijkstra untuk Menentukan Jalur Lintasan Terpendek Kota Kisaran Menuju Objek Wisata Simalungun. *Rekayasa Teknik Informatika Dan Informasi*, *1*(2), 68–77.

Putri, T. D., Sugeng, W., & Safitri, E. (2020). Algoritma Dijkstra untuk Penentuan Jarak Tempuh Terpendek Pengantaran Katering Pabrik. *Journal MIND Journal*, *5*(2), 108–120. https://doi.org/https://doi.org/10.26760/mindjournal.v5i2.108

Sunardi, S., Yudhana, A., & Kadim, A. A. (2019). Implementasi Algoritma Dijkstra dan Algoritma Semut Untuk Analisis Rute Transjogja Berbasis Android. *IT JOURNAL RESEARCH AND DEVELOPMENT*, *4*(1), 1–9. https://doi.org/10.25299/itjrd.2019.vol4(1).2483

Sutrisno, T., Yosua, Nugraha, M. A., & Joni. (2020). Penerapan Algoritma Dijkstra Dalam Pembuatan Peta Untar. *Jurnal Komputer Dan Informatika*, *15*(1), 308–316. https://doi.org/https://journal.untar.ac.id/index.php/JKI/article/view/8699

Yosua, S., Sigalingging, C., Jipesya, J., & Jumaryadi, Y. (2021). Implementasi Algoritma Dijkstra dalam Pencarian Klinik Hewan Terdekat. *Jurnal Ilmiah FIFO*, *13*(1), 85. https://doi.org/10.22441/fifo.2021.v13i1.009

Yuswardi, & Balia. (2022). Distribusi Gas LPG Dengan Mencari Rute Terpendek Perbandingan Algoritma Dijkstra Dan Ant Colony Wilayah Kabupaten Pidie. *Jurnal Sains Riset* , *12*(1), 132–138. https://doi.org/10.47647/jsr.v10i12