JIKA (Jurnal of Informatics) Universitas Muhammadiyah Tangerang *P ISSN : 2549-0710*
Vol 6, No 3, October 2022, pp 297-309 *E ISSN : 2722-2713*
DOI : 10.31000/jika.v6i3.6528

PERBANDINGAN ALGORITMA *DIJKSTRA* DAN *FLOYD-WARSHALL* DALAM MENENTUKAN RUTE TERPENDEK STASIUN GUBENG MENUJU WISATA SURABAYA

Hendra[1)], Yosefina Finsensia Riti[2)]
[1,2] Program Studi Ilmu Informatika Fakultas Teknik, Universitas Katolik Darma Cendika,
Jl. Dr. Ir. H. Soekarno No.201, Surabaya, Jawa Timur
Co Responden Email: hendra@student.ukdc.ac.id

Article history
Received 07 Jul 2022
Revised 22 Sep 2022
Accepted 03 Oct 2022
Available 27 Oct 2022

Keywords
Algorithm Comparison,
Shortest-Path Problem,
Dijkstra,
Floyd-Warshall.

Abstract

The issue of determining the shortest-path problem has become a common thing to discuss with regard to the way it is solved using graphs. The solution is generally used with the aim of optimizing certain aspects, for example in terms of fuel use and travel time. This solution can be achieved by applying one of the components in the graph, called algorithm. This journal was written to help the public, especially tourists, in providing information related to the most efficient route from the starting point of the station to various tourist destinations in the city of Surabaya with the application of the Dijkstra algorithm and Floyd-Warshall algorithm. The techniques used in the calculations were carried out manually and the help of programs with language C. Through the research, the results obtained by the Dijkstra Algorithm had a higher efficiency in terms of calculations and the complexity of the program. The calculations carried out in the two algorithms show results in the form of the same and correct minimum weight value, as well as information related to the shortest trajectory that can be applied by the community to increase efficiency in traveling in the city of Surabaya. The shortest route obtained in the comparison of the two algorithms is on the journey with the starting point of Gubeng Station to Jalan Taman Apsari with a distance of 1.2 km.

Riwayat
Diterima 07 Jul 2022
Revisi 22 Sep 2022
Disetujui 03 Okt 2022
Terbit 27 Okt 2022

Kata Kunci
Perbandingan Algoritma,
Rute Terpendek,
Dijkstra,
Floyd-Warshall.

Abstrak

Persoalan penentuan rute terpendek sudah menjadi hal umum untuk dibahas berkaitan dengan cara penyelesaiannya yang menggunakan graf. Penyelesaian tersebut umumnya digunakan dengan tujuan untuk mengoptimalisasi aspek tertentu, contohnya dalam hal penggunaan bahan bakar dan waktu tempuh. Penyelesaian tersebut dapat diraih dengan penerapan salah satu komponen dalam graf yaitu algoritma. Jurnal ini ditulis guna membantu masyarakat terlebih para turis dalam memberikan informasi terkait rute yang paling efisien dari titik awal stasiun menuju ke berbagai destinasi wisata kota Surabaya dengan penerapan Algoritma Dijkstra dan Algoritma *Floyd-Warshall.* Teknik yang digunakan dalam perhitungan dilaksanakan secara manual dan bantuan program dengan Bahasa C. Melalui penelitian tersebut, hasil yang diperoleh Algoritma Dijkstra memiliki efisiensi yang lebih tinggi dari segi perhitungan dan kompleksitas programnya. Perhitungan yang dilaksanakan pada kedua algoritma menunjukkan hasil berupa nilai bobot minimum yang sama dan tepat, adapun juga informasi terkait lintasan terpendek yang dapat diterapkan masyarakat guna meningkatkan efisiensi dalam berwisata di kota Surabaya. Untuk rute terpendek yang diperoleh pada perbandingan dua algoritma adalah pada perjalanan dengan titik awal Stasiun Gubeng menuju Jalan Taman Apsari dengan jarak tempuh sejauh 1,2 km.

PENDAHULUAN

Graf merupakan struktur yang bertujuan untuk merepresentasikan suatu objek-objek diskrit serta hubungan dari objek-objek tersebut. Graf memiliki 2 himpunan, yaitu himpunan titik/simpul (*vertices*) dan himpunan sisi (*edges*) yang sekaligus menjadi komponen utama dalam adanya suatu Graf.

Konsep dari Graf ini tentunya juga dapat diimplementasikan ke berbagai aspek dalam kehidupan sehari-hari, salah satunya yaitu mengenai penentuan rute terpendek.

Persoalan penentuan rute terpendek (*shortest-path problem*) pada satu tempat awal menuju tujuan tertentu sudah menjadi hal yang umum dibahas dan penerapan graf berperan besar dalam menemukan penyelesaian terhadap persoalan ini. Dengan adanya penentuan rute yang paling efisien ini tentunya akan membawa dampak positif seperti halnya mampu meminimalisir suatu biaya operasional, selain itu optimalisasi waktu tempuh yang semakin baik antar destinasi, yang sekaligus menjadi alasan yang mendasari dibuatnya jurnal ini. Jurnal ini secara khusus membahas terkait proses dan hasil analisa perbandingan antara Algoritma *Dijkstra* dan *Floyd-Warshall* berdasarkan parameter yang telah ditentukan dengan objek penelitian yaitu penentuan rute terpendek dari Stasiun Gubeng menuju berbagai destinasi wisata di Kota Surabaya.

Penelitian terkait penentuan rute terpendek sudah banyak dilakukan, diantaranya seperti yang dilakukan oleh Masri, dkk. (Masri et al., n.d.). Membahas mengenai penentuan rute terpendek dengan algoritma *Dijkstra* yang dapat ditempuh pada suatu titik menuju tempat wisata di daerah objek pariwisata Danau Toba dan sekitarnya, yang diimplementasikan ke dalam aplikasi berbasis web dengan menggunakan *script PHP* dan *MySQL* untuk pengolahan databasenya. Hasil dari pencarian rute terpendek tersebut juga dilengkapi dengan jarak dan waktu tempuh dari satu titik menuju tujuan wisatawan.

Penelitian selanjutnya yaitu oleh Napiah, dkk. (Napiah & Darma Astuti, n.d.). Dimana penerapan algoritma ini diupayakan dalam meminimalisir kemacetan pada suatu kota dengan mobilitas yang tinggi dari wilayah Jelambar menuju kampus STMIK NUSA MANDIRI Cengkareng, dengan menghasilkan luaran (*output*) berupa rute tercepat dan terpendek dari tempat asal menuju tempat tujuan tanpa terhalang oleh kemacetan.

Penelitian serupa juga telah dilakukan oleh Delby T.Salaki (Salaki, n.d.), bahasan dalam penelitian tersebut membahas terkait mencari lintasan terpendek dari FMIPA menuju kantor rektorat dan ke fakultas lainnya. Hasil yang diperoleh dalam penelitian ini adalah berupa lintasan terpendek dari FMIPA dengan minimum lokal atau akses dengan jarak terdekat dari setiap lokasi yang kemudian digabungkan menjadi sebuah kumpulan lintasan dari satu lokasi menuju lokasi lain dengan jarak yang paling minimum menggunakan algoritma *Dijkstra* dan penerapan pada pemrograman *Pascal*.

Penelitian selanjutnya yaitu oleh Aldy, dkk. (Cantona, Fauziah, & Winarsih, 2020). Penelitian ini secara khusus membahas terkait pengkalkulasian jarak paling dekat dari satu titik menuju museum yang menjadi tempat tujuan di Jakarta. Hasil yang diperoleh dari penelitian ini berupa rute terpendek dari posisi pengguna menggunakan GPS (*Global Positioning System*) pada *smartphone* berbasis android. Teknik GPS ini memiliki peran untuk membantu guna meningkatkan kualitas dan kuantitas pada aktivitas yang secara khusus memerlukan teknik ini, penerapan GPS ini juga dinilai dapat menaikkan pendapatan dan kesejahteraan hidup pada aspek tertentu seperti pada penelitian yang dilakukan oleh Apriliani, dkk. (Apriliani, Herawati, Khan, Dewanti, & Rizal, 2018). Berdasarkan penelitian oleh Aldy, dkk, penggunaan GPS berbasis android tersebut dapat mempersingkat efektivitas waktu setiap pengguna untuk mencari museum di Jakarta.

Penelitian selanjutnya yang dilakukan oleh Andiany F & Hadikurniawati W. (Andiany & Hadikurniawati, 2018). Persoalan yang dibahas dalam penelitian ini adalah mengenai bagaimana rute terpendek serta estimasi biaya bakar bagi setiap pegawai perusahaan memperoleh fasilitas untuk menunjang pekerjaan, khususnya bagi pegawai yang mempunyai mobilitas tinggi. Hasil dari penelitian tersebut berupa algoritma untuk menentukan rute paling efisien serta dapat mengestimasi biaya bahan bakar yang kemudian algoritma tersebut diterapkan dalam web resmi sebagai sarana untuk melakukan peminjaman mobil dinas.

Penerapan selanjutnya juga telah dibahas oleh Muharrom M, (Muharrom, 2020) dijelaskan bahwa algoritma *Dijkstra* disini dapat digunakan dalam menentukan jarak menuju kampus terdekat dimana dalam pembahasan ini yaitu dari Bekasi menuju STMIK Nusa Mandiri, sehingga rute

perjalanan dapat ditempuh dengan cepat dan lebih efisien dalam waktu, tenaga, dan biaya bahan bakar di tengah-tengah kemacetan yang marak di kota Jakarta. Adapun faktor-faktor yang dinilai dapat diperhitungkan dalam mencari rute tercepat dalam pembahasan ini, seperti waktu tempuh dari satu persimpangan menuju persimpangan lainnya, adanya pedagang kaki lima, volume kendaraan yang digunakan, serta lalu lintas yang beroperasi.

Penerapan dari algoritma ini dapat pula diterapkan untuk proses evakuasi bencana, seperti halnya pada penelitian oleh Hanif Lyonnais. (Lyonnais, 2011). Algoritma *Dijkstra* berperan dalam mencari jalan dengan bantuan algoritma A* (*A-Star*) dari *node* awal hingga *node* tujuan pada graf. Dengan memanfaatkan algoritma tersebut, dapat dicari rute paling dekat menuju tempat yang aman saat evakuasi bencana. meningkatkan efisiensi dan memberikan solusi atas suatu permasalahan.

Adapun perbandingan yang telah dilakukan oleh beberapa penelitian sebelumnya, salah satu contohnya pada penelitian oleh Wildan, dkk. (Wildan et al., 2020). Tujuan dari penelitian tersebut adalah membandingkan performa dari penerapan algoritma *Dijkstra* dan algoritma A* (*A-Star*) untuk penyelesaian game *Pac-Man*. Hasil yang diperoleh berupa kesimpulan bahwa dari kedua algoritma yang dijadikan sebagai bahan perbandingan, algoritma A* memiliki performa yang lebih bagus dan mampu mendapatkan skor terbaik.

Penelitian serupa juga sudah pernah dilakukan oleh Rizaldy, dkk. (Rizaldy, n.d.). Bahasan terkait perbandingan pada algoritma *Dijkstra*, algoritma *Floyd-Warshall*, dan algoritma *Johnson* untuk memecahkan masalah pada topik bahasan, seperti mencari rute terpendek yang menghubungkan antara dua kota berlainan tertentu (*Single-source Single-destination Shortest Path*), mencari semua lintasan terpendek masing-masing dari suatu kota ke setiap kota lainnya. (*Single-source Shortest Path Problems*), dan mencari semua lintasan terpendek masing-masing antara setiap kemungkinan pasang kota yang berbeda. (*All-pairs Shortest Path Problems*).

Penelitian selanjutnya, dibahas oleh Wijaya, dkk. (Wijaya et al., 2016). Bahasan terkait penerapan sirkuit *Hamilton* untuk menentukan rute terpendek perjalanan salesman di PT. Health Wealth International (HWI). Penelitian ini menganalisis dan merancang aplikasi rute perjalanan salesman dengan sirkuit *Hamilton* yang mencakup pembelian, penjualan, dan rute perjalanan. Untuk metodologi yang digunakan adalah metode *Breadth First Search* (BFS) dan *Depth First Search* (DFS), serta untuk tools menggunakan *Data Flow Diagram* (DFD). Hasil dari penelitian ini menyatakan bahwa pencarian dengan metode BFS dinilai lebih efektif dan lebih cepat dibandingkan dengan metode DFS. Luaran (*output*) berupa aplikasi rute perjalanan dengan sirkuit *Hamilton* pada PT. Health Wealth International yang terkomputerisasi untuk menyediakan informasi rute perjalanan dan bertujuan untuk meningkatkan efektivitas pengiriman barang perusahaan.

Penelitian terkait algoritma *Dijkstra* juga ada seperti yang ditulis oleh Iqbal, dkk. (Iqbal, Zhang, Iqbal, & Tariq, 2018). Mendeskripsikan mengenai bagaimana penerapan dari algoritma *Dijkstra* yang kemudian dikembangkan menjadi algoritma MDSP (*modified Dijkstra's shortest path algorithms*). Pada penelitian tersebut, penulis menggunakan beberapa parameter untuk menemukan rute terpendek yang lebih valid lagi untuk jaringan pada jalan dengan penelitian yang telah dilakukan sebelumnya menggunakan algoritma *Dijkstra* yang dinilai masih membutuhkan banyak modifikasi untuk meningkatkan efisiensi serta mengurangi kompleksitas. Hasil yang diperoleh penulis adalah bahwa benar algoritma MDSP ini dinilai jauh lebih baik dan dapat memberikan informasi rute terpendek yang lebih valid dengan kompleksitas waktu yang minimum untuk beberapa lintasan jalan.

Penelitian terkait penentuan rute terpendek juga dapat dilakukan antara algoritma *Dijkstra* dengan algoritma lain seperti halnya yang ditulis oleh Fitro, dkk. (Fitro, Saeful Bachri, Ilham, Purnomo, & Frendianata, 2018) yaitu dengan algoritma *node combination* berbasis pada sistem GIS (*Geographic Information Systems*). Untuk data yang digunakan berlokasi pada Kecamatan Taman, Sidoarjo, Jawa Timur dengan 17 *node* dan 72 simpul. Jarak yang dihitung adalah berdasarkan nilai lintang dan bujur yang

diperoleh melalui *Google Maps API* dan dengan teknik pencarian dua algoritma, untuk menghasilkan produk berupa sistem pencarian rute terpendek yang tidak hanya optimal namun juga mampu memberikan efisiensi dari aspek *memory usage*. Berdasarkan penelitian yang telah dilaksanakan oleh penulis, didapatkan hasil bahwa kedua teknik pencarian dengan algoritma *Dijkstra* dan *node combination* tersebut berhasil menemukan jalur optimal dengan studi kasus di Kecamatan Taman, Sidoarjo, Jawa Timur dengan hasil berupa rute dari titik awal hingga titik akhir.

Pengembangan dari aplikasi algoritma *Dijkstra* selanjutnya oleh Kien Hua & Abdullah, (Kien Hua & Abdullah, 2018). Menjelaskan mengenai WSDA (*Weighted Sum-Dijkstra's Algorithm*) yang merupakan kombinasi dari metode WSM (*Weighted Sum Method*) dan algoritma *Dijkstra* yang digunakan pula untuk menyelesaikan persoalan pada *network* yang memiliki beberapa kriteria yang dinilai dapat bekerja lebih baik dan efektif dibandingkan dengan penggunaan algoritma *Dijkstra* saja yang hanya dapat menyelesaikan permasalahan dengan satu kriteria. Inovasi tersebut dapat meningkatkan efisiensi pada identifikasi rute.

Algoritma kedua terkait algoritma *Floyd-Warshall* memiliki beberapa implementasi seperti halnya menggunakan prosedur melalui program dan kode *MATLAB*. *MATLAB* merupakan *software* komputer yang dapat menyelesaikan persoalan pada berbagai materi di bidang matematika, antara lain seperti penyelesaian operasi dasar matematika, turunan, integral, limit, begitu pula dengan vektor, matriks, dan pembuatan grafik. (Estining, Lufianawati, Adipura, & Dan Masjudin, 2021). Implementasi *MATLAB* diterapkan seperti pada penelitian yang dilakukan oleh Sakharov, dkk. (Sakharov, Chernyi, Saburov, & Chertkov, 2021). Berdasarkan penelitian yang dilakukan oleh algoritma *Floyd-Warshall* merupakan algoritma yang dinamis dalam menentukan rute terpendek antara semua titik/simpul pada graf berarah yang tidak memiliki siklus dan dapat diimplementasikan pada graf yang memiliki bobot negatif.

Adapun implementasi algoritma *Floyd-Warshall* guna menyelesaikan suatu persoalan seperti pada penelitian oleh Sudarya Triana. (Sudarya Triana & Syahputri, 2018). Persoalan yang dibahas adalah terkait jalur garasi terpendek yang didasarkan karena tingginya jumlah kendaraan yang dimiliki suatu pengguna, tentunya kebutuhan akan garasi menjadi semakin lebih tinggi. Garasi disini sangat diperlukan khususnya ketika kendaraan mengalami masalah di lokasi yang tidak diketahui. Karenanya, algoritma *Floyd-Warshall* dapat diaplikasikan guna menyelesaikan persoalan tersebut dan pada penelitian yang dilaksanakan penerapan algoritma diimplementasikan pada aplikasi berbasis android yang dilengkapi sistem GPS dan *Google Maps*.

Algoritma *Floyd-Warshall* merupakan algoritma yang dari segi input adalah berupa graf matriks dan menghasilkan luaran berupa rute terpendek, pada penelitian tersebut dijelaskan bahwa dalam implementasi algoritma *Floyd-Warshall* ini memiliki dua kemungkinan, yakni rute terpendek sebenarnya berasal dari *node* yang berada diantara 1 dan k, dan biasanya terdiri dari beberapa baris dari simpul i ke k+1, begitu pula dengan k+1 ke j dalam penentuan rute terpendek. Diperoleh hasil bahwa dengan implementasi yang dilakukan penulis, algoritma *Floyd-Warshall* dapat digunakan untuk menyelesaikan persoalan yang bertujuan dalam hal penentuan rute terpendek, contohnya dalam hal mencari jarak garasi terpendek.

Permasalahan yang digambarkan pada Jurnal ini yaitu mengenai perbandingan algoritma tertentu untuk penentuan rute terpendek (*shortest-path problem*), dimana konsep penentuan rute terpendek ini merupakan suatu permasalahan dengan menemukan sebuah jalur antara 2 *node* dengan jumlah bobot seminimal mungkin. Kedua algoritma yaitu algoritma *Dijkstra* dan algoritma *Floyd-Warshall* dapat digunakan untuk mencari lintasan yang paling efektif dari suatu graf permasalahan. Cara kerja dari algoritma *Dijkstra* ini adalah dengan membandingkan bobot terkecil dari *node* awal sampai kepada *node* tujuan untuk menemukan lintasan yang paling efisien untuk dapat dilalui. (Masri et al., n.d.). Sedangkan, algoritma *Floyd-Warshall* merupakan algoritma yang bekerja di antara semua pasang simpul.

JIKA (Jurnal of Informatics) Universitas Muhammadiyah Tangerang *P ISSN : 2549-0710*
Vol 6, No 3, October 2022, pp 297-309 *E ISSN : 2722-2713*
DOI : 10.31000/jika.v6i3.6528

Stasiun Surabaya Gubeng (SGU) adalah salah satu stasiun yang terletak di daerah Gubeng, kota Surabaya, Jawa Timur. Dikenal juga sebagai Stasiun Gubeng, stasiun ini merupakan salah satu dari stasiun terpopuler yang sering menjadi tujuan warga antarkota. Adanya penelitian ini bertujuan untuk menghasilkan luaran (*output*) berupa rute maupun jalur yang paling efisien untuk dapat ditempuh dari Stasiun Gubeng menuju berbagai macam destinasi yang seringkali menjadi tujuan para wisatawan.

METODE PENELITIAN

Analisa Kebutuhan

Pada proses penulisan jurnal ini, diperlukan beberapa aplikasi tambahan guna mendukung implementasi graf, diantaranya seperti *draw.io* dalam pembuatan *flowchart*. *Google Earth*, *Google Maps* guna membuat tampilan graf. Kemudian, setelah perhitungan manual pada masing-masing algoritma, implementasi akan diterapkan dengan bantuan bahasa pemrograman C serta *source code* akan dieksekusi melalui *Online C Compiler* (https://www.onlinegdb.com/online_c_compiler), untuk *source code* yang digunakan selama penelitian ditampilkan pada pembahasan dalam jurnal ini.

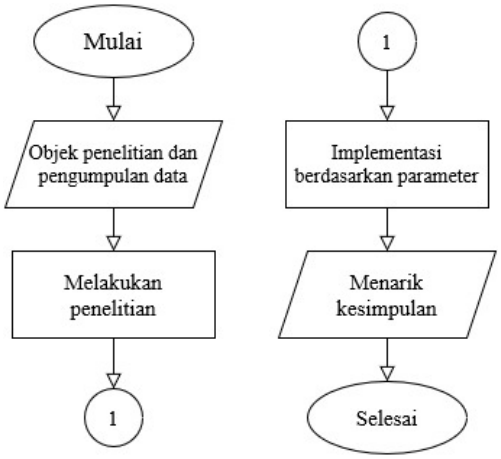
Tahap Penelitian

Penelitian ini menggunakan metode komparatif, dimana metode ini digunakan untuk perbandingan terhadap satu variabel terhadap satu maupun lebih variabel pembanding. Teknik pengumpulan data dilakukan secara kualitatif, dalam arti data tidak hanya berupa angka atau numerik, melainkan dapat berupa kata-kata dan data pendukung lainnya. Tujuan dari penelitian ini adalah untuk menghasilkan luaran (*output*) berupa perbandingan dari kedua algoritma yang digunakan dalam penerapan graf dengan titik awal Stasiun Surabaya Gubeng (SGU) menuju destinasi lokasi wisata yang cenderung populer di masyarakat. Perbandingan ini didasarkan pada dua aspek yang menjadi parameter terhadap hasil penelitian terhadap ketiga algoritma, diantaranya :

1. Perhitungan manual dan program
2. Kompleksitas

Berdasarkan tujuan tersebut, penelitian dilakukan dengan beberapa tahapan, diantaranya :

1. Menentukan objek penelitian serta mengumpulkan data dan informasi sebanyak mungkin.
2. Melakukan penelitian sesuai perhitungan yang telah ditentukan.
3. Melakukan penelitian dengan implementasi serta perbandingan berdasarkan parameter yang telah ditentukan.
4. Menarik kesimpulan dari hasil implementasi pada objek penelitian.



Gambar 1. *Flowchart* tahap penelitian

Untuk data yang diperoleh, dilansir dari www.javatravel.net terkait 10 destinasi wisata yang akan digunakan dalam penelitian ini, diantaranya Tugu Pahlawan, Ekowisata Mangrove Wonorejo, Surabaya North Quay, Kebun Bibit Wonorejo, Taman Bungkul, Kampung Bulak, Hutan Bambu Keputih, Pantai Ria Kenjeran, Jalan Taman Apsari, dan Kebun Binatang Surabaya.

Parameter Perbandingan

Untuk parameter pertama yaitu terkait perbandingan terhadap perhitungan secara manual dan secara program. Pada parameter ini, dilakukan perbandingan terhadap perhitungan manakah yang dinilai paling efektif untuk mencari rute terpendek berdasarkan titik awal dan tujuan tertentu. Untuk perhitungan secara manual dilakukan oleh penulis dengan menganalisa data, graf, dan proses perhitungan total secara manual. Kemudian, untuk perhitungan secara program akan dilakukan dengan bantuan bahasa

JIKA (Jurnal of Informatics) Universitas Muhammadiyah Tangerang *P ISSN : 2549-0710*
Vol 6, No 3, October 2022, pp 297-309 *E ISSN : 2722-2713*
DOI : 10.31000/jika.v6i3.6528

pemrograman C untuk melihat apakah hasil yang diperoleh sama besar dan perhitungan mana yang paling efektif dan efisien untuk digunakan dalam jangka waktu yang lebih cepat dan praktis.

Untuk parameter kedua, yakni mengenai kompleksitas. Pada perbandingan ini, ketiga algoritma yang digunakan akan dilihat seberapa efisien penggunaan algoritma tersebut untuk menentukan rute terpendek pada graf yang tersedia. Kompleksitas disini berbicara mengenai rumit atau tidaknya algoritma yang digunakan, sehingga indikasi yang dapat dijadikan yaitu seberapa efisien rute yang terbentuk oleh suatu algoritma yang dapat ditempuh oleh wisatawan, sehingga para wisatawan dapat melakukan perjalanan menuju destinasi tertentu dengan cepat dan dengan rute yang paling tepat.

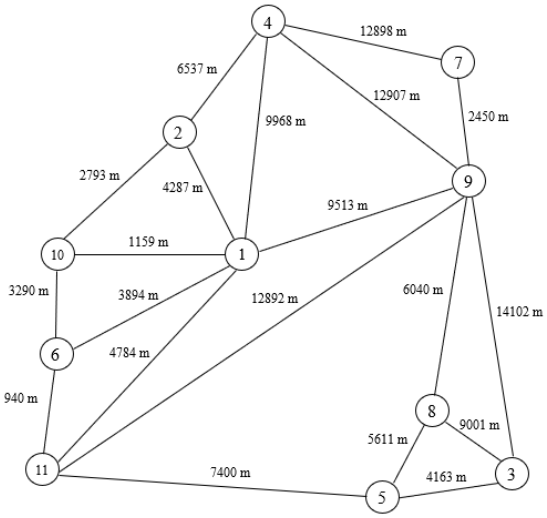
HASIL DAN PEMBAHASAN

Dilansir dari www.javatravel.net, adapun data yang akan digunakan untuk perhitungan :

Tabel 1. Data tempat dan keterangan

No	Nama Tempat	Keterangan
1	Stasiun Surabaya Gubeng	Titik awal
2	Tugu Pahlawan	Destinasi wisata
3	Ekowisata Mangrove Wonorejo	Destinasi wisata
4	Surabaya North Quay	Destinasi wisata
5	Kebun Bibit Wonorejo	Destinasi wisata
6	Taman Bungkul	Destinasi wisata
7	Kampung Bulak	Destinasi wisata
8	Hutan Bambu Keputih	Destinasi wisata
9	Pantai Ria Kenjeran	Destinasi wisata
10	Jalan Taman Apsari	Destinasi wisata
11	Kebun Binatang Surabaya	Destinasi wisata

Untuk graf yang digunakan dalam penelitian ini adalah graf berbobot dimana simpul dalam graf ini merepresentasikan nama tempat dari titik awal termasuk destinasi wisata, dan sisi/*edges* sebagai rute dengan nilai bobot (jarak) dalam satuan meter berdasarkan data pada Tabel 1. dan *software* tambahan yang digunakan, yaitu *Google Earth* dan *Google Maps*.



Gambar 2. Tampilan graf

Penerapan Algoritma *Dijkstra*

Perhitungan berdasarkan data dan graf yang telah dibuat dilakukan dengan 2 perhitungan, yaitu perhitungan secara manual dan perhitungan melalui bantuan bahasa pemrograman C. Berikut merupakan penerapan algoritma *Dijkstra* yang dilakukan secara manual. Langkah pertama diawali dengan membuat Matriks Ketetanggaan (*Adjacency Matrix*) berdasarkan simpul, sisi, dan nilai bobot pada graf (dalam kilometer).

Tabel 2. Matriks ketetanggaan

	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**
1	0	4	∞	10	∞	4	∞	∞	10	1	5
2	4	0	∞	7	∞	∞	∞	∞	∞	3	∞
3	∞	∞	0	∞	4	∞	∞	9	14	∞	∞
4	10	7	∞	0	∞	∞	13	∞	13	∞	∞
5	∞	∞	4	∞	0	∞	∞	6	∞	∞	7
6	4	∞	∞	∞	∞	0	∞	∞	∞	3	1
7	∞	∞	∞	13	∞	∞	0	∞	2	∞	∞
8	∞	∞	9	∞	6	∞	∞	0	6	∞	∞
9	10	∞	14	13	∞	∞	2	6	0	∞	13
10	1	3	∞	∞	∞	3	∞	∞	∞	0	∞
11	5	∞	∞	∞	7	1	∞	∞	13	∞	0

Setelah matriks ketetanggaan terbentuk, maka selanjutnya dapat dilakukan perhitungan secara manual dengan menerapkan algoritma *Dijkstra*. Hasil didapatkan melalui pencarian rute terpendek secara manual dari titik awal menuju titik akhir sesuai baris dan kolom tabel, kemudian rute yang paling pendek yang akan dijadikan sebagai nilai (dalam kilometer) dari masing-masing baris dan kolom. Untuk hasil rute terpendek direpresentasikan oleh baris pertama, dikarenakan fokus dari pembahasan ini adalah untuk mencari lintasan yang paling efisien dari titik awal Stasiun Surabaya Gubeng (SGU) yang diwakilkan oleh simpul nomor 1.

Tabel 3. Hasil penerapan algoritma dijkstra

	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**
1	0	4	16	10	12	4	12	16	10	1	5
2	4	0	21	7	16	6	19	20	13	3	7
3	16	20	0	26	4	13	17	9	14	16	12
4	10	7	25	0	21	13	13	19	13	9	14
5	12	16	4	21	0	8	14	6	12	12	7
6	4	6	13	13	8	0	16	14	13	3	1
7	12	19	17	13	14	16	0	8	2	13	15
8	16	20	9	19	6	14	8	0	6	17	13
9	10	13	14	13	12	13	2	6	0	11	13
10	1	3	16	9	12	3	13	17	11	0	4
11	5	7	12	14	7	1	15	13	13	4	0

Baris 1 merepresentasikan hasil pencarian rute terpendek dari Stasiun Surabaya Gubeng (SGU) menuju ke semua titik destinasi. Rute yang terbentuk terdiri dari beberapa lintasan dengan adanya pembulatan pada nilai bobot (jarak) nya untuk mempermudah penulisan.

1 → 1 = 0 km
1 → 2 = 1 – 10 – 2 = 3,9 = 4 km
1 → 3 = 1 – 11 – 5 – 3 = 16,3 = 16 km
1 → 4 = 1 – 4 = 9,9 = 10 km
1 → 5 = 1 – 11 – 5 = 12,1 = 12 km
1 → 6 = 1 – 6 = 3,9 = 4 km
1 → 7 = 1 – 9 – 7 = 11,9 = 12 km
1 → 8 = 1 – 9 – 8 = 15,6 = 16 km
1 → 9 = 1 – 9 = 9,5 = 10 km
1 → 10 = 1 – 10 = 1,2 = 1 km
1 → 11 = 1 – 11 = 4,8 = 5 km

Berdasarkan hasil pada Tabel 3. diperoleh bahwa rute terpendek (dalam kilometer) dari Stasiun Gubeng menuju beberapa destinasi wisata di Kota Surabaya, diantaranya dengan tujuan Tugu Pahlawan dapat ditempuh dengan jarak 4 km melalui rute dari Stasiun Gubeng melewati destinasi wisata Jalan Taman Apsari dan sampai pada Tugu Pahlawan. Kemudian, dengan tujuan Ekowisata Mangrove Wonorejo memiliki rute terpendek sejauh 16 km dari Stasiun Gubeng menuju ke Kebun Binatang Surabaya, kemudian ke arah Kebun Bibit Wonorejo hingga sampai pada tujuan. Untuk tujuan dari Stasiun Gubeng menuju Surabaya North Quay dapat ditempuh sejauh 10 km, Kebun Bibit Wonorejo sejauh 12 km dengan rute melalui Kebun Binatang Surabaya terlebih dahulu, Taman Bungkul 4 km, lalu dengan tujuan ke Kampung Bulak sejauh 12 km, Hutan Bambu Keputih memiliki rute terpendek sejauh 16 km dengan arah yang sama yaitu melalui Pantai Ria Kenjeran, untuk Pantai Ria Kenjeran sendiri dapat ditempuh dari Stasiun Gubeng sejauh 10 km, kemudian untuk tujuan Jalan Taman Apsari sejauh 1 km dan Kebun Binatang Surabaya sejauh 5 km. Untuk rute terpendek dari Stasiun Gubeng menuju wisata Kota Surabaya diperoleh pada rute 1-10, dengan keterangan titik awal Stasiun Gubeng dan tujuan ke Jalan Taman Apsari yang dapat ditempuh hanya sejauh 1,2 km atau 1 km (dalam pembulatan).

Kemudian, untuk perhitungan selanjutnya yaitu menggunakan bantuan program. Disini bahasa pemrograman yang digunakan adalah bahasa C, yang dimana proses eksekusi program dilakukan dengan *Online C Compiler*. Berikut merupakan kode program untuk implementasi algoritma *Dijkstra* terhadap data dan graf yang sudah tersedia.

JIKA (Jurnal of Informatics) Universitas Muhammadiyah Tangerang *P ISSN : 2549-0710*
Vol 6, No 3, October 2022, pp 297-309 *E ISSN : 2722-2713*
DOI : 10.31000/jika.v6i3.6528

main.c

```c
#include<stdio.h>
#include<stdbool.h>
#define inf 9999
int n;
int table[100][100];
int dijkstra (int start, int stop){
    int dist[n];
    bool visit[n];
    int i,j,v;
    for (i=1; i<=n; i++){
        dist[i] = inf;
    }
    for (i=1; i<=n; i++){
        visit[i] = false;
    }
    dist[start] = 0;
    while(true){
        int u = -1;
        int minDist = inf;
        for (i=1; i<=n; i++){
            if((visit[i] == false) && (dist[i] < minDist)){
                u = i;
                minDist = dist[i];
            }
        }
        if((u == -1) || (dist[u] == inf)){
            break;
        }
        visit[u] = true;
        for(v = 1; v <= n; v++){
```

Gambar 3. Program algoritma *dijkstra*

```c
            if(table[u][v] != 0){
                if(dist[v] > dist[u]+table[u][v]){
                    dist[v] = dist[u]+table[u][v];
                }
            }
            }
        }
        return dist[stop];
}
int main(){
    int start, stop;
    int i,j;
    printf("Masukkan banyak node : ");
    scanf("%d", &n);
    printf("Masukkan jarak node dalam matriks : \n");
    for(i=1; i<=n; i++){
        for(j=1; j<=n; j++){
            scanf("%d", &table[i][j]);
        }
    }
    int answer = 0, x;
    while(answer == 0){
        printf("\n");
        printf("Masukkan titik awal : ");
        scanf("%d", &start);
        printf("Masukkan titik tujuan : ");
        scanf("%d", &stop);
        printf("Bobot minimal dari titik %d ke %d: %d km \n"
        , start, stop, dijkstra(start,stop));
    }
}
```

Gambar 4. Program algoritma *dijkstra*

Melalui penerapan bahasa C tersebut, pertama akan diminta inputan mengenai seberapa banyak jumlah *node* yang akan digunakan, *node* disini berperan dalam menentukan seberapa banyak *node* yang akan membentuk ukuran matriks. *Node* yang telah diinputkan akan dilanjutkan dengan proses menginput jarak setiap *node* dalam bentuk matriks, bisa ditulis jarak menyerupai bentuk matriks dengan bantuan spasi pada *keyboard* hingga muncul inputan berikutnya. Kemudian, akan diminta inputan mengenai titik awal dan titik tujuan sehingga dapat diperoleh maksimal nilai bobot pada rute tersebut. Untuk penentuan hasil digunakan *node* sebanyak 11, kemudian untuk jarak *node* dalam bentuk matriks digunakan tampilan yang menyerupai seperti pada Tabel 3. Kemudian dilakukan perhitungan oleh program berdasarkan titik awal dan titik tujuan yang diinputkan.

```
Masukkan banyak node : 11
Masukkan jarak node dalam matriks :
0 4 0 10 0 4 0 0 10 1 5
4 0 0 7 0 0 0 0 0 3 0
0 0 0 0 4 0 0 9 14 0 0
10 7 0 0 0 0 13 0 13 0 0
0 0 4 0 0 0 0 6 0 0 7
4 0 0 0 0 0 0 0 0 3 1
0 0 0 13 0 0 0 0 2 0 0
0 0 9 0 6 0 0 0 6 0 0
10 0 14 13 0 0 2 6 0 0 13
1 3 0 0 0 3 0 0 0 0 0
5 0 0 0 7 1 0 0 13 0 0

Masukkan titik awal : 1
Masukkan titik tujuan : 1
Bobot minimal dari titik 1 ke 1: 0 km

Masukkan titik awal : 1
Masukkan titik tujuan : 2
Bobot minimal dari titik 1 ke 2: 4 km

Masukkan titik awal : 1
Masukkan titik tujuan : 3
Bobot minimal dari titik 1 ke 3: 16 km

Masukkan titik awal : 1
Masukkan titik tujuan : 4
Bobot minimal dari titik 1 ke 4: 10 km

Masukkan titik awal : 1
Masukkan titik tujuan : 5
Bobot minimal dari titik 1 ke 5: 12 km

Masukkan titik awal :
```

Gambar 5. Hasil program *dijkstra*

Berdasarkan program tersebut, dilakukan input untuk titik awal dari Stasiun Surabaya Gubeng (SGU) menuju ke beberapa tujuan wisata, hasil menyatakan bahwa diperoleh nilai bobot minimal yang sama seperti hasil pada perhitungan yang dilakukan secara manual.

Penerapan Algoritma *Floyd-Warshall*

Perhitungan selanjutnya adalah dengan menerapkan algoritma *Floyd-Warshall*, dengan algoritma tersebut akan dilakukan proses secara manual dan secara program menggunakan bahasa C. Perhitungan dengan algoritma *Floyd-Warshall* ini, terdiri dari 3 variabel diantaranya k, i, dan j. Dimana nantinya untuk perhitungan akan diselesaikan dengan menggunakan rumus :

$$X[i,j] > X[i,k] + X[k,j] \qquad (1)$$

Keterangan :
$[i,j]$ = baris dan kolom

JIKA (Jurnal of Informatics) Universitas Muhammadiyah Tangerang *P ISSN : 2549-0710*
Vol 6, No 3, October 2022, pp 297-309 *E ISSN : 2722-2713*
DOI : 10.31000/jika.v6i3.6528

k = matriks ke-n

Berdasarkan pada graf di Gambar 2. Langkah awal untuk menerapkan algoritma *Floyd-Warshall* ini adalah dengan membuat matriks hubung graf (K = 0) terlebih dahulu. Untuk variabel i, j, dan k diketahui :

$I = \{1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11\}$
$J = \{1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11\}$
$K = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11\}$

Berikut merupakan matriks hubung graf (K = 0) yang telah dibuat. Matriks berikut dibuat berdasarkan derajat yang mewakili setiap simpul pada graf (dalam kilometer).

Tabel 4. Matriks hubung graf

	1	2	3	4	5	6	7	8	9	10	11
1	0	4	∞	10	∞	4	∞	∞	10	1	5
2	4	0	∞	7	∞	∞	∞	∞	∞	3	∞
3	∞	∞	0	∞	4	∞	∞	9	14	∞	∞
4	10	7	∞	0	∞	∞	13	∞	13	∞	∞
5	∞	∞	4	∞	0	∞	∞	6	∞	∞	7
6	4	∞	∞	∞	∞	0	∞	∞	∞	3	1
7	∞	∞	∞	13	∞	∞	0	∞	2	∞	∞
8	∞	∞	9	∞	6	∞	∞	0	6	∞	∞
9	10	∞	14	13	∞	∞	2	6	0	∞	13
10	1	3	∞	∞	∞	3	∞	∞	∞	0	∞
11	5	∞	∞	∞	7	1	∞	∞	13	∞	0

Setelah diperoleh matriks hubung graf seperti pada Tabel 4. Langkah berikut adalah dengan menentukan masing-masing matriks, dengan ketentuan K = (1, 2, 3, …, n) dimana nilai dari K ini mewakilkan matriks ke-berapa yang dibuat serta baris dan kolom berapa yang diharuskan memiliki letak yang tetap pada matriks, nilai pada variabel K juga harus sama dengan jumlah simpul yang terdapat pada graf. Apabila nilai K = 1, maka baris dan kolom 1 haruslah berada pada angka yang sama, kemudian dilakukan perhitungan pada baris dan kolom lainnya dengan menggunakan Persamaan 1.

Tabel 5. Matriks K = 1

	1	2	3	4	5	6	7	8	9	10	11
1	0	4	∞	10	∞	4	∞	∞	10	1,2	4,8
2	4	0	∞	7	∞	8	∞	∞	14	3	9
3	∞	∞	0	∞	4	∞	∞	9	14	∞	∞
4	10	7	∞	0	∞	14	13	∞	13	11	15
5	∞	∞	4	∞	0	∞	∞	6	∞	∞	7
6	4	8	∞	14	∞	0	∞	∞	14	3	1
7	∞	∞	∞	13	∞	∞	0	∞	2	∞	∞
8	∞	∞	9	∞	6	∞	∞	0	6	∞	∞
9	10	14	14	13	∞	14	2	6	0	11	13
10	1	3	∞	11	∞	3	∞	∞	11	0	6
11	5	9	∞	15	7	1	∞	∞	13	6	0

Tabel 6. Matriks K = 2

	1	2	3	4	5	6	7	8	9	10	11
1	0	4	∞	10	∞	4	∞	∞	10	1	5
2	4	0	∞	7	∞	8	∞	∞	14	3	9
3	∞	∞	0	∞	4	∞	∞	9	14	∞	∞
4	#	7	∞	0	∞	14	13	∞	13	10	15
5	∞	∞	4	∞	0	∞	∞	6	∞	∞	7
6	4	8	∞	14	∞	0	∞	∞	14	3	1
7	∞	∞	∞	13	∞	∞	0	∞	2	∞	∞
8	∞	∞	9	∞	6	∞	∞	0	6	∞	∞
9	1	14	14	13	∞	14	2	6	0	11	13
10	1	3	∞	10	∞	3	∞	∞	11	0	6
11	5	9	∞	15	7	1	∞	∞	13	6	0

Pada Tabel 6. Ditemukan dua nilai yang mengalami perubahan dari matriks sebelumnya pada Tabel 5. Hal ini dikarenakan adanya nilai yang lebih kecil ketika menggunakan Persamaan 1. Dimana $X[i,k] + X[k,j]$ memiliki nilai yang lebih kecil dari $X[i,j]$ nya. Untuk algoritma *Floyd-Warshall* ini terdiri dari 11 matriks dimulai dari K = 1, sehingga diperoleh hasil pada saat matriks berada pada K = 11.

Tabel 7. Hasil algoritma floyd-warshall

	1	2	3	4	5	6	7	8	9	10	11
1	0	4	16	10	12	4	12	16	10	1	5
2	4	0	18	7	14	6	16	20	14	3	7
3	15	18	0	25	4	12	16	9	14	15	11
4	10	7	25	0	21	13	13	19	13	10	14
5	12	14	4	21	0	8	14	6	12	11	7
6	4	6	12	13	8	0	16	14	14	3	1
7	3	16	16	13	14	16	0	8	2	13	15
8	7	20	9	19	6	14	8	0	6	17	13
9	1	14	14	13	12	14	2	6	0	11	13
10	1	3	15	10	11	3	13	17	11	0	4
11	5	7	11	14	7	1	17	13	13	4	0

Dari Tabel 7. dapat diperoleh hasil rute terpendek dari titik awal Stasiun Surabaya Gubeng (SGU) yang direpresentasikan oleh

JIKA (Jurnal of Informatics) Universitas Muhammadiyah Tangerang *P ISSN : 2549-0710*
Vol 6, No 3, October 2022, pp 297-309 *E ISSN : 2722-2713*
DOI : 10.31000/jika.v6i3.6528

baris pertama menuju ke semua titik lainnya yang merupakan destinasi wisata di Kota Surabaya. Berikut merupakan hasil untuk rute dengan bobot paling minimum.

1 → 1 = 0 km
1 → 2 = 4 km
1 → 3 = 16 km
1 → 4 = 10 km
1 → 5 = 12 km
1 → 6 = 4 km
1 → 7 = 12 km
1 → 8 = 16 km
1 → 9 = 10 km
1 → 10 = 1 km
1 → 11 = 5 km

Berdasarkan Tabel 7. hasil yang diperoleh juga memiliki nilai bobot yang sama pada masing-masing rute dengan perhitungan pada algoritma sebelumnya, untuk jarak dari Stasiun Gubeng menuju Tugu Pahlawan memiliki rute terpendek sejauh 4 km, untuk tujuan ke Ekowisata Mangrove Wonorejo sejauh 16 km, Surabaya North Quay sejauh 10 km, Kebun Bibit Wonorejo sejauh 12 km, Taman Bungkul sejauh 4 km, kemudian untuk tujuan menuju Kampung Bulak dapat ditempuh sejauh 12 km, Hutan Bambu Keputih dengan jarak 16 km, Pantai Ria Kenjeran dengan jarak 10 km, dan untuk tujuan ke Jalan Taman Apsari sejauh 1 km dan Kebun Binatang Surabaya sejauh 5 km. Rute terpendek dan paling dekat yang diperoleh untuk mengunjungi destinasi wisata Surabaya dengan titik awal Stasiun Gubeng adalah pada rute 1-10 yaitu dari Stasiun Gubeng menuju Jalan Taman Apsari dengan total jarak minimum sebesar 1 km dalam pembulatan.

Hasil tersebut juga dilaksanakan pada pemrograman. Dengan perhitungan selanjutnya yaitu secara program, disini digunakan bahasa C untuk mengeksekusi code untuk melihat apakah hasil yang menjadi luaran sudah tepat dan sama seperti yang dikerjakan dengan cara manual. Untuk compiler, menggunakan *Online C Compiler* dan diperoleh hasil yang menyatakan bobot minimum dari inputan titik awal dan titik tujuan.

main.c

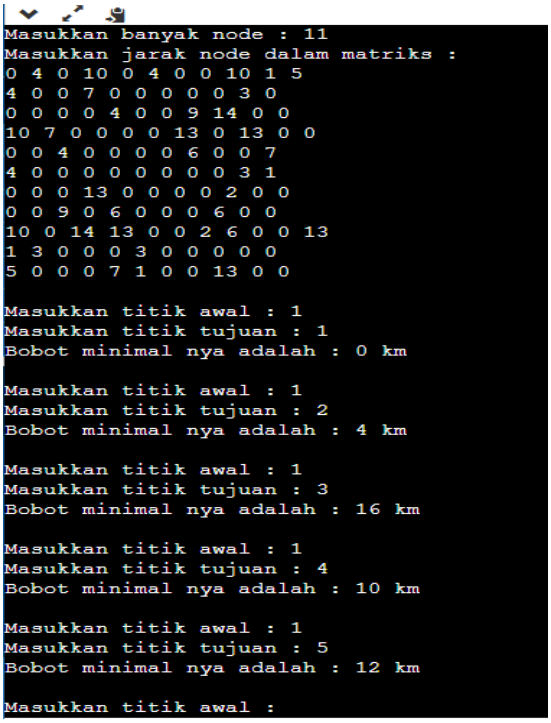
```c
#include<stdio.h>
#include<conio.h>
#define inf 9999
int table[100][100], pred[100][100];
int main()
{
    int i,j,k,n,bobot;
    printf("Masukkan banyak node : ");
    scanf("%d",&n);
    for(i=1; i<=n; i++)
    {
        for(j=1; j<=n; j++){
            pred[i][j] = i;
            table[i][j] = 0;
        }
    }
    for(i=1; i<=n; i++)
    {
        for(j=1; j<=n; j++)
        {
            if(i==j)
            {
                table[i][j] = 0;
            }
            else {
                printf("Masukkan jarak node dalam matriks :  \n");
                for(i=1; i<=n; i++){
                    for(j=1; j<=n; j++){
                        scanf("%d", &table[i][j]);
                    }
                }
            }
        }
    }
    for(i=1; i<=n; i++)
    {
        for(j=1; j<=n; j++)
        {
```

Gambar 6. Program algoritma *floyd-warshall*

```c
            if(i!=j && table[i][j] == 0)
            {
                table[i][j] = inf;
            }
        }
    }
    for(k=1; k<=n;k++)
    {
        for(i=1; i<=n; i++)
        {
            for(j=1; j<=n; j++)
            {
                if(table[i][k] + table[k][j] < table[i][j])
                {
                    table[i][j] = table[i][k] + table[k][j];
                    pred[i][j] = pred[k][j];
                }
            }
        }
    }
int answer = 0, x;
while(answer == 0){
    int start = 0, stop = 0;
    printf("\n");
    printf("Masukkan titik awal : ");
    scanf("%d", &start);
    printf("Masukkan titik tujuan : ");
    scanf("%d", &stop);
    printf("Bobot minimal nya adalah : %d km \n", table[start][stop]);
    }
}
```

Gambar 7. Program algoritma *floyd-warshall*

Code tersebut dieksekusi, sehingga luaran akan menampilkan beberapa inputan terlebih dahulu seperti halnya banyak *node*, jarak *node* dalam matriks guna mendata keseluruhan komponen sehingga dapat terbentuk menjadi sebuah matriks dan dapat diketahui bobot minimal pada titik awal dan tujuan dengan bantuan program. Dilakukan beberapa percobaan terkait inputan titik untuk melihat apakah bobot minimum yang dihasilkan sama dengan bobot pada pengerjaan dengan cara manual.



Gambar 8. Hasil program *floyd-warshall*

Untuk hasil yang diperoleh memiliki nilai bobot (jarak) minimal yang sama seperti hasil pada perhitungan secara manual pada titik awal dan titik tujuan tertentu.

Hasil Perbandingan Algoritma

Berdasarkan dua algoritma yang telah dikerjakan dengan cara manual dan program, yaitu algoritma *Dijkstra* dan algoritma *Floyd-Warshall*, diperoleh hasil yang mengacu pada parameter yang digunakan sebagai indikator untuk mengukur algoritma mana yang paling tepat dan efisien dalam menentukan rute terpendek dari Stasiun Surabaya Gubeng (SGU) menuju ke berbagai destinasi wisata di kota Surabaya. Untuk parameter pertama, adalah terkait perhitungan secara manual dan program, kedua algoritma dikerjakan dengan kedua perhitungan tersebut dengan baik, dan diperoleh hasil rute terpendek dan paling dekat yang sama baik dengan cara manual maupun dengan program yaitu pada perjalanan dengan titik awal Stasiun Gubeng menuju Jalan Taman Apsari dengan jarak yang harus ditempuh sejauh 1 km. Untuk efektivitas pengerjaan, algoritma *Floyd-Warshall* terutama dengan cara manual memerlukan waktu pengerjaan yang lebih lama serta memerlukan lebih banyak penggunaan tabel serta perhitungan jika dibandingkan dengan pengerjaan manual pada algoritma *Dijkstra*. Selain itu, diperlukan ketelitian yang tinggi untuk memperoleh penyelesaian dengan Algoritma *Floyd-Warshall*. Untuk *output* yang diperoleh, kedua algoritma memiliki hasil yang sama efektif dan tepat.

Sedangkan untuk kompleksitas program, algoritma *Floyd-Warshall* disini juga dinilai lebih kompleks jika dibandingkan dari segi baris *source code* pada penerapan algoritma *Dijkstra* seperti halnya pada Gambar 3. dan Gambar 4.

Tabel 5. Perbandingan algoritma berdasarkan parameter kompleksitas

Algoritma	Parameter pengujian
	Kompleksitas (Bahasa C)
Dijkstra	60 baris
Floyd-Warshall	69 baris

Berdasarkan tabel tersebut, maka algoritma *Dijkstra* memiliki kompleksitas pada *source code* yang lebih kecil dibandingkan dengan penerapan pada algoritma *Floyd-Warshall*, berarti bahwa algoritma *Dijkstra* memiliki efisiensi yang lebih tinggi dalam menentukan rute terpendek berdasarkan inputan titik awal dan titik tujuannya.

KESIMPULAN

Berdasarkan perhitungan yang telah dilaksanakan sebelumnya, diperoleh hasil bahwa perbandingan kedua algoritma, yakni algoritma *Dijkstra* dan algoritma *Floyd-Warshall* dalam penentuan rute terpendek dari Stasiun Surabaya Gubeng menuju berbagai destinasi wisata Kota Surabaya dilaksanakan dengan perhitungan manual dan menggunakan bantuan program. Hasil yang diperoleh menunjukkan bahwa dari kedua cara pengerjaan tersebut menghasilkan luaran yang tepat, sekaligus dengan penggunaan dua algoritma yang dapat memberikan output dengan nilai bobot sama. Data dan graf yang dibuat juga dapat diimplementasikan dengan baik. Dengan demikian, algoritma *Dijkstra* dan algoritma *Floyd-Warshall* dapat digunakan untuk menyelesaikan persoalan rute terpendek. Namun, tidak menutup kemungkinan akan hasil yang berbeda pada kasus lainnya.

Mengacu pada parameter yang telah dibuat, diperoleh kesimpulan bahwa algoritma *Dijkstra* merupakan algoritma yang lebih efisien dan lebih praktis serta dapat memberikan output dengan rute terpendek dan nilai bobot yang tepat pula. Penerapan algoritma *Dijkstra* juga lebih mudah dipahami dan diterapkan khususnya dalam menentukan rute terpendek pada suatu objek penelitian. Rute terpendek serta informasi jarak yang dihasilkan pun juga diharapkan dapat membantu masyarakat terutama para turis yang ingin mengunjungi berbagai destinasi wisata di Kota Surabaya dengan lebih efisien, selain itu guna untuk mengoptimalkan penggunaan kebutuhan seperti bahan bakar, estimasi waktu untuk menempuh perjalanan..

REFERENSI

Andiany, F. E., & Hadikurniawati, W. (2018). *Implementasi Algoritma Dijkstra Untuk Mencari Rute Terpendek Antar Kantor Dan Estimasi Penggunaan Bahan Bakar Kendaraan (Studi Kasus PT. Telkom Indonesia Regional IV Jateng-DIY)*.

Apriliani, I. M., Herawati, H., Khan, A. M., Dewanti, L. P., & Rizal, A. (2018). *Pengenalan Teknologi Global Positioning System (GPS) Sebagai Alat Bantu Operasi Penangkapan Ikan Di Pangandaran* (Vol. 7).

Cantona, A., Fauziah, & Winarsih. (2020). Implementasi Algoritma Dijkstra Pada Pencarian Rute Terpendek ke Museum di Jakarta. *Jurnal Teknologi Dan Manajemen Informatika*, *6*(1). Retrieved from http://http://jurnal.unmer.ac.id/index.php/jtmi

Estining, D., Lufianawati, T., Adipura, C., & Dan Masjudin, W. (2021). Pelatihan Software Matlab Untuk Penyelesaian Masalah Di Bidang Matematika. *Dharmakarya : Jurnal Aplikasi Ipteks Untuk Masyarakat*, *10*(1), 14–16. https://doi.org/10.24198/dharmakarya.v10i1.30015

Fitro, A., Saeful Bachri, O., Ilham, A., Purnomo, S., & Frendianata, I. (2018). Shortest Path Finding in Geographical Information Systems using Node Combination and Dijkstra Algorithm. *International Journal of Mechanical Engineering and Technology (IJMET*, *9*(2), 755–760. Retrieved from http://www.iaeme.com/IJMET/index.asp755http://www.iaeme.com/IJMET/issues.asp?JType=IJMET&VType=9&IType=2http://www.iaeme.com/IJMET/issues.asp?JType=IJMET&VType=9&IType=2http://www.iaeme.com/IJMET/index.asp756

Iqbal, M., Zhang, K., Iqbal, S., & Tariq, I. (2018). A Fast and Reliable Dijkstra Algorithm for Online Shortest Path. In *SSRG International Journal of Computer Science and Engineering (SSRG-IJCSE)* (Vol. 5). Retrieved from http://www.internationaljournalssrg.org

Kien Hua, T., & Abdullah, N. (2018). Weighted Sum-Dijkstra's Algorithm in Best Path Identification based on Multiple Criteria. *Journal of Computer Science & Computational Mathematics*, *8*(3), 107–113. https://doi.org/10.20967/jcscm.2018.04.008

Lyonnais, H. (2011). *Metode Pencarian Lintasan Terpendek Graf untuk Evakuasi Bencana*.

Masri, M., Kiswanto, A. P., Santri Kusuma, B., Dosen,), Alumni,), & Abstrak, U. (n.d.). *Implementasi Algoritma Dijkstra Dalam Perancangan Aplikasi Penentuan Rute Terpendek Pada Objek Pariwisata Danau Toba Dan Sekitarnya*. Retrieved from http://maps.google.com.Google

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. https://doi.org/10.21927/ijubi.v3i1.1229

Napiah, M., & Darma Astuti, R. (2022). *Implementasi Algoritma Dijkstra Menentukan Jarak Terdekat Jelambar Kampus STMIK Nusa Mandiri Cengkareng* (Vol. 7).

Rizaldy, M. R. (n.d.). *Pencarian Jalur Terpendek dalam GPS dengan Menggunakan Teori graf Using Graph Theory for Finding Shortest Path in GPS*.

Sakharov, V., Chernyi, S., Saburov, S., & Chertkov, A. (2021). Automatization Search for the Shortest Routes in the Transport Network Using the Floyd-warshell Algorithm. *Transportation Research Procedia*, *54*, 1–11. Elsevier B.V. https://doi.org/10.1016/j.trpro.2021.02.041

Salaki, D. T. (n.d.). *Penentuan Lintasan Terpendek Dari FMIPA Ke Rektorat Dan Fakultas Lain Di UNSRAT Manado Menggunakan Algoritma Dijkstra*.

Sudarya Triana, Y., & Syahputri, I. (2018). Implementation Floyd-Warshall Algorithm for the Shortest Path of Garage. In *International Journal of Innovative Science and Research Technology* (Vol. 3). Retrieved from www.ijisrt.com

Wijaya, E., & Vera. (2016). Penerapan Sirkuit Hamilton Untuk Menentukan Rute Terpendek Perjalanan Salesman PT Health Wealth International (HWI). In *Jurnal TIMES*.

Wildan, A., Ramadhan, R., & Udjulawa, D. (2020). Perbandingan Algoritma Dijkstra dan Algoritma A Star Pada Permainan Pac-Man. In *Jurnal Algoritme* (Vol. 1).