IMPLEMENTASI ALGORITMA *FLOYD WARSHALL* DALAM PENCARIAN RUTE TERPENDEK LOKASI *TOWER BASE TRANSCEIVER STATION* (BTS) PADA PT CITRA AKSES INDONUSA

SKRIPSI

OLEH
BELLA NAFA SAVITRI
NIM. 17610120

PROGRAM STUDI MATEMATIKA
FAKULTAS SAINS DAN TEKNOLOGI
UNIVERSITAS ISLAM NEGERI MAULANA MALIK IBRAHIM
MALANG
2022

IMPLEMENTASI ALGORITMA *FLOYD WARSHALL* DALAM PENCARIAN RUTE TERPENDEK LOKASI *TOWER BASE TRANSCEIVER STATION* (BTS) PADA PT CITRA AKSES INDONUSA

SKRIPSI

Diajukan Kepada
Fakultas Sains dan Teknologi
Universitas Islam Negeri Maulana Malik Ibrahim Malang
untuk Memenuhi Salah Satu Persyaratan dalam
Memperoleh Gelar Sarjana Matematika (S.Mat)

Oleh
Bella Nafa Savitri
NIM. 17610120

PROGRAM STUDI MATEMATIKA
FAKULTAS SAINS DAN TEKNOLOGI
UNIVERSITAS ISLAM NEGERI MAULANA MALIK IBRAHIM
MALANG
2022

IMPLEMENTASI ALGORITMA *FLOYD WARSHALL* DALAM PENCARIAN RUTE TERPENDEK LOKASI *TOWER BASE TRANSCEIVER STATION* (BTS) PADA PT CITRA AKSES INDONUSA

SKRIPSI

Oleh
Bella Nafa Savitri
NIM. 17610120

Telah Diperiksa dan Disetujui Untuk Diuji
Malang, 24 Juni 2022

Dosen Pembimbing I	Dosen Pembimbing II
Mohammad Nafie Jauhari, M.Si.	Evawati Alisah, M.Pd.
NIDT. 19870218201608011056	NIP. 197206041999032001

Mengetahui,
Ketua Program Studi Matematika

Dr. Elly Susanti, S.Pd., M.Sc.
NIP. 197411292000122005

IMPLEMENTASI ALGORITMA *FLOYD WARSHALL* DALAM PENCARIAN RUTE TERPENDEK LOKASI *TOWER BASE TRANSCEIVER STATION* (BTS) PADA PT CITRA AKSES INDONUSA

SKRIPSI

Oleh
Bella Nafa Savitri
NIM. 17610120

Telah Dipertahankan di Depan Dewan Penguji Skripsi
Dan Dinyatakan Diterima Sebagai Salah Satu Persyaratan
Untuk Memperoleh Gelar Sarjana Matematika (S. Mat)
Tanggal 25 Juni 2022

Ketua Penguji	: Muhammad Khudzaifah, M.Si.	.....
Anggota Penguji 1	: Hisyam Fahmi M.Kom	.....
Anggota Penguji 2	: Muhammad Nafie Jauhari, M.Si.	.....
Anggota Penguji 3	: Evawati Alisah, M.Pd.	.....

Mengetahui,
Ketua Program Studi Matematika

Dr. Elly Susanti, S.Pd., M.Sc.
NIP. 197411292000122005

PERNYATAAN KEASLIAN TULISAN

Saya yang bertanda tangan di bawah ini:

Nama	:	Bella Nafa Savitri
NIM	:	17610120
Program Studi	:	Matematika
Fakultas	:	Sains dan Teknologi
Judul Skripsi	:	Implementasi Algoritma *Floyd Warshall* Dalam Pencarian Rute Terpendek Lokasi *Tower Base Transceiver Station* (BTS) Pada PT Citra Akses Indonusa

Menyatakan dengan sebenarnya bahwa skripsi yang saya tulis ini benar-benar merupakan hasil karya sendiri, bukan merupakan pengambilan data, tulisan, atau pikiran orang lain yang saya akui sebagai hasil tulisan atau pikiran saya sendiri, kecuali dengan mencantumkan sumber cuplikan pada daftar rujukan. Apabila dikemudian hari terbukti atau dapat dibuktikan skripsi ini hasil jiplakan, maka saya bersedia menerima sanksi atas perbuatan tersebut.

Malang, 20 juni 2022

Yang membuat pernyataan,

Bella Nafa Savitri

NIM. 17610120

HALAMAN MOTTO

“Selesaikan Apa Yang Telah Dimulai”

HALAMAN PERSEMBAHAN

Skripsi ini penulis persembahkan untuk:
Ayahanda Sungkono, ibunda Afifah Hikmawati dan adik Ahmad Dhani Wafiy, beserta seluruh keluarga yang selalu mendoakan, memberi semangat, nasihat dan kasih sayang yang tak terhingga. Teman-teman yang selalu membantu dalam menyelesaikan kesulitan penulis. Dosen-dosen yang telah membantu menyelesaikan skripsi penulis. Sehingga menjadikan alasan bagi penulis untuk selalu bersemangat dalam berproses.

KATA PENGANTAR

Assalamu'alaikum Warahmatullahi Wabarakatuh

Segala puji bagi Allah Swt. yang telah melimpahkan rahmat, taufik dan hidayah-Nya, sehingga penulis mampu menyelesaikan skripsi ini yang diberi berjudul "Implementasi Algoritma *Floyd Warshall* dalam Pencarian Rute Terpendek Lokasi *Tower Base Transceiver Station* (BTS) pada PT Citra Akses Indonusa", sebagai syarat untuk memperoleh gelar sarjana dalam bidang matematika di Fakultas Sains dan Teknologi, Universitas Islam Negeri Maulana Malik Ibrahim Malang.

Dalam penyusunan skripsi ini tidak lepas bimbingan dan arahan baik secara langsung maupun tidak langsung dari berbagai pihak. Untuk itu ucapan terima kasih yang sebesar-besarnya dan penghargaan yang setinggi-tingginya penulis sampaikan terutama kepada:

1. Prof. Dr. H. M. Zainuddin, M.A., selaku rektor Universitas Islam Negeri Maulana Malik Ibrahim Malang.
2. Dr. Sri Harini, M.Si., selaku dekan Fakultas Sains dan Teknologi Universitas Islam Negeri Maulana Malik Ibrahim Malang.
3. Dr. Elly Susanti, S.Pd., M.Sc., selaku ketua Program Studi Matematika Fakultas Sains dan Teknologi, Universitas Islam Negeri Maulana Malik Ibrahim Malang.
4. Mohammad Nafie Jauhari, M.Si., selaku dosen pembimbing I yang telah banyak memberikan arahan, nasihat, motivasi serta pengalaman baru dan berharga kepada penulis.
5. Evawati Alisah, M.Pd., selaku dosen pembimbing II yang telah memberikan arahan, nasihat dan motivasi kepada penulis.
6. Muhammad Khudzaifah, M.Si., selaku Ketua Penguji dalam ujian skripsi yang telah memberikan masukkan dan arahan kepada penulis.
7. Hisyam Fahmi, M.Kom., selaku Anggota Penguji 1 dalam ujian skripsi yang telah memberikan masukkan dan arahan kepada penulis.

8. Seluruh dosen dan segenap sivitas akademika Program Studi Matematika, Fakultas Sains dan Teknologi, Universitas Islam Negeri Maulana Malik Ibrahim Malang atas segala ilmu dan bimbingan yang diberikan
9. Ayahanda Sungkono dan Ibunda Afifah Hikmawati serta adik Ahmad Dhani Wafiy dan seluruh keluarga yang selalu mendoakan, memberikan semangat, dukungan dan motivasi kepada penulis.
10. Seluruh mahasiswa jurusan Matematika angkatan 2017 dan semua pihak yang tidak disebutkan satu-persatu yang telah memberikan bantuan secara langsung maupun tidak langsung kepada penulis dalam menyelesaikan skripsi ini.

Semoga Allah Swt. memberikan karunia dan rahmat-Nya kepada kita semua. Penulis berharap semoga skripsi ini dapat memberikan manfaat khususnya bagi penulis dan para pembaca. *Aamiin*

Wassalamu'alaikum Warahmatullahi Wabarakatuh.

Malang, 19 Juni 2022

Penulis

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

DAFTAR LAMPIRAN

ABSTRAK

Savitri, Bella Nafa. 2022. **Implementasi Algoritma *Floyd Warshall* Dalam Pencarian Rute Terpendek Lokasi *Tower Base Transceiver Station* (BTS) Pada PT Citra Akses Indonusa**. Skripsi. Program Studi Matematika, Fakultas Sains dan Teknologi, Universitas Islam Negeri Maulana Malik Ibrahim Malang. Pembimbing (1): Mohammad Nafie Jauhari, M.Si, Pembimbing (2): Evawati Alisah M.Pd.

Kata Kunci: Graf, rute terpendek, *Floyd Warshal*, *tower*, PT Citra Akses Indonusa.

PT Citra Akses Indonusa merupakan perusahaan yang berjalan pada keahlian layanan teknologi informasi di Provinsi Banten. Salah satu layanan yang disajikan oleh perusahaan tersebut membutuhkan pembangunan *tower Base Transceiver Station* (BTS). *Tower* tersebut membutuhkan pemeliharaan jika terdapat kerusakan pada sinyal jaringan. Akibatnya dibutuhkan rute terpendek untuk memudahkan karyawan menuju lokasi *tower* agar lebih efektif. Terdapat delapan lokasi *tower* BTS di Kabupaten Tangerang dan tiga lokasi *tower* BTS di Kota Tangerang. Proses pencarian rute terpendek pada penelitian ini menggunakan algoritma *Floyd Warshall*, yang memiliki keunikan mencari rute terpendek dengan membandingkan setiap sisi dari seluruh sisi yang dilewati. Proses pengujian rute terpendek dilakukan dengan memilih titik awal, kemudian memilih beberapa lokasi *tower* BTS. Selanjutnya akan dicari rute terpendek menggunakan algoritma *Floyd Warshall* dari setiap titik tujuan lokasi *tower* BTS, kemudian akan dipilih lokasi *tower* BTS yang akan dikunjungi terlebih dahulu begitu pun seterusnya hingga tujuan terakhir. Keefektifan pencarian rute terpendek ini melibatkan perbandingan rute yang disajikan oleh *Google Maps*. Berdasarkan hasil 30 kali percobaan secara acak terhadap lokasi *tower* BTS, didapatkan rata-rata efektivitas rute terpendek sebesar 25.54% dibandingkan rute yang dihasilkan *Google Maps*. Hal ini dikarenakan adanya penyeleksian pada tiap lokasi tujuan *tower* BTS sehingga menjadikan rute lebih efektif.

ABSTRACT

Savitri, Bella Nafa. 2022. **On The Implementation of Floyd Warshall Algorithm to Determine The Shortest Path of Base Transceiver Station (BTS) Tower Location in PT Citra Akses Indonusa**. Thesis. Department of Mathematics, Faculty of Science and Technology, Universitas Islam Negeri Maulana Malik Ibrahim Malang. Advisor (1): Mohammad Nafie Jauhari, M.Si, Advisor (2): Evawati Alisah M.Pd.

Keywords: Graf, shortest path, Floyd Warshal, tower, PT Citra Akses Indonusa.

PT Citra Akses Indonusa is a company that operates on the expertise of information technology services in Banten Province. One of the services provided by the company requires the construction of a Base Transceiver Station (BTS) tower. The tower requires maintenance if there is damage to the network signal. As a result, the shortest route is needed to make it easier for employees to reach the tower location to be more effective. There are eight BTS tower locations in Tangerang Regency and three BTS tower locations in Tangerang City. The process of finding the shortest route in this study uses the Floyd Warshall algorithm, which is unique in finding the shortest route by comparing each edge of all edges that are passed. The process of testing the shortest route is done by selecting the starting point, then selecting several BTS tower locations. Next, the shortest route will be searched using the Floyd Warshall algorithm from each point of destination for the BTS tower location, then the BTS tower location will be selected first and so on until the last destination. The effectiveness of this shortest route search involves a comparison of the routes presented by Google Maps. Based on the results of 30 randomized trials on BTS tower locations, the average shortest route effectiveness was 25.54% compared to the route generated by Google Maps. This is due to the selection at each BTS tower destination location so as to make the route more effective.

مستخلص البحث

سفطري, بيلا نافا. ٢٠٢٢. **حول تنفيذ خوارزمية (*Floyd Warshall*) لتحديد أقصر مسار لموقع برج محطة الإرسال والاستقبال الأساسية (BTS) في (PT Citra Akses Indonusa)**. البحث الجامعي. قسم الرياضيات، كلية العلوم والتكنولوجيا، جامعة مولاناماالك إبراهيم الإسلامية الحكومية مالانج. المشرف (١): محمدنافع جوهر، الماجستير (٢): إيفاواتي اليسا, الماجستير.

الكلمة المفتاحية: الرسم البياني، مسار ات قصيرة، (*Floyd Warshall*)، أبراج، (PT Citra Akses Indonusa).

(PT Citra Akses Indonusa) هي شركة تعمل على خبرة خدمات تكنولوجيا المعلومات في مقاطعة بانتين. تتطلب إحدى الخدمات التي تقدمها الشركة بناء برج محطة الإرسال والاستقبال الأساسية (BTS). يتطلب البرج صيانة إذا كان هناك تلف في إشارة الشبكة. نتيجة لذلك، هناك حاجة إلى أقصر طريق لتسهيل وصول الموظفين إلى موقع البرج ليكونوا أكثر فعالية. هناك ثمانية مواقع برج (BTS) في (Tangerang *Regency*) وثلاثة مواقع برج (BTS) في مدينة (Tangerang). تستخدم عملية العثور على أقصر طريق في هذه الدراسة خوارزمية (Floyd Warshall)، وهي فريدة من نوعها في إيجاد أقصر طريق من خلال مقارنة كل حافة من جميع الحواف التي يتم تمريرها. تتم عملية اختبار أقصر مسار عن طريق اختيار نقطة البداية، ثم اختيار العديد من مواقع برج (BTS). بعد ذلك، سيتم البحث عن أقصر طريق باستخدام خوارزمية (Floyd Warshall) من كل نقطة وجهة لموقع برج (BTS)، ثم سيتم اختيار موقع برج (BTS) أولاً وهكذا حتى الوجهة الأخيرة. تتضمن فعالية هذا البحث الأقصر عن المسار مقارنة المسارات التي تقدمها (*Google Maps*). بناءً على نتائج ٣٠ تجربة عشوائية على مواقع برج (BTS)، كان متوسط فعالية المسار الأقصر ٢٥،٥٤٪ مقارنة بالمسار الذي تم إنشاؤه بواسطة (*Google Maps*). ويرجع ذلك إلى الاختيار في كل موقع وجهة برج (BTS) لجعل الطريق أكثر فعالية.

BAB I

PENDAHULUAN

1.1 Latar Belakang

Teori yang berkaitan erat dengan pencarian rute terdekat adalah teori tentang graf. Graf merupakan cabang dari salah satu ilmu matematika yang memuat informasi tertentu dengan menggambarkan berbagai macam struktur yang bertujuan untuk memvisualisasikan objek-objek agar mudah dimengerti. Objek graf tersebut berupa titik dan sisi yang menghubungkan titik tersebut, yang di mana bentuk matematis pada graf G dapat dinyatakan sebagai $G = (V, E)$, di mana V merupakan himpunan titik-titik dan E merupakan himpunan sisi yang menghubungkan titik-titik. Terdapat berbagai macam algoritma-algoritma yang bisa digunakan untuk menyelesaikan masalah dalam mencari rute terpendek.

Bersamaan dengan pertumbuhan pada era globalisasi, perkembangan pada bidang teknologi informasi juga mengalami kemajuan yang cukup cepat, dimulai dengan terciptanya perangkat canggih komputer. Teknologi informasi merupakan salah satu bentuk kemudahan akses dalam memperoleh informasi secara relevan, akurat dan tepat waktu, di mana sistem kerjanya berupa pengolahan data yang terdiri dari mendapatkan, menyusun, memproses, memanipulasi dan menyimpan data untuk diolah menjadi informasi sesuai dengan pencapaian yang diinginkan. Teknologi informasi yang berkembang pada saat ini memiliki penerapan dalam bisnis perusahaan, sebagai contoh penerapan dalam dukungan penjualan dan jasa, penyediaan informasi pasar dan pengembangan produk baru dengan tujuan utama untuk memberikan layanan terbaik kepada para *customer*.

PT Citra Akses Indonusa merupakan salah satu perusahaan yang berjalan pada keahlian layanan teknologi informasi di Provinsi Banten. Pendirian perusahaan ini dilakukan pada tahun 2017 dan memiliki pelanggan yang tersebar di Provinsi Banten. Layanan yang disajikan terdiri dari *networking*, CCTV, *server* dan *storage*, serta *seat management*. Layanan *networking* merupakan layanan utama yang disajikan oleh perusahaan tersebut di mana dalam pemasangan layanan *networking* membutuhkan pembangunan *tower Base Transceiver Station* (BTS).

BTS merupakan sebuah alat dalam jaringan telekomunikasi berupa *tower* yang memiliki suatu antena yang memancarkan sinyal dan berfungsi sebagai penguat sinyal daya yang menghubungkan antara jaringan operator telekomunikasi dengan para *customer*. Selain mendirikan BTS, dibutuhkan juga suatu perangkat yang bertugas untuk menjaga kualitas sinyal jaringan agar tetap terjaga dengan baik. *Tower* tersebut juga membutuhkan *maintenance* atau pengaturan kembali jika terdapat kerusakan pada sinyal jaringan. Agar memudahkan akses karyawan untuk sampai ke lokasi BTS dan melakukan *maintenance* dengan optimal, dibutuhkan rute terpendek.

Sebagai pekerja yang melaksanakan pekerjaannya dalam memperoleh jarak terpendek dari tugas yang diperoleh tidak boleh berputus asa. Putus asa itu sendiri memiliki arti tidak mau kembali berusaha dalam memperoleh suatu hal dan merupakan sikap yang tercela. Dalam agama Islam perilaku putus asa merupakan perilaku yang dilarang bagi semua orang mukmin, dikutip dalam Quran:

يٰبَنِيَّ اذْهَبُوْا فَتَحَسَّسُوْا مِنْ يُّوْسُفَ وَاَخِيْهِ وَلَا تَايْـَٔسُوْا مِنْ رَّوْحِ اللّٰهِ ۗاِنَّهٗ لَا يَايْـَٔسُ مِنْ رَّوْحِ اللّٰهِ اِلَّا الْقَوْمُ الْكٰفِرُوْنَ

Artinya: "*Wahai anak-anakku! Pergilah kamu, carilah (berita) tentang Yusuf dan saudaranya dan jangan kamu berputus asa dari rahmat Allah. Sesungguhnya yang berputus asa dari rahmat Allah, hanyalah orang-orang kafir*" (Q.S Yusuf:87).

Berdasarkan ayat tersebut dijelaskan tentang larangan berputus asa, sebagaimana yang dikisahkan lewat pencarian Nabi Yusuf. Nabi Yakub memerintahkan putra-putranya untuk tidak berputus asa dan selalu berharap akan menemukan Nabi Yusuf dan saudaranya karena orang-orang yang berputus asa merupakan orang-orang yang keluar dari jalan Allah swt. Pada dasarnya orang-rang kafir tidak mengetahui akan adanya kebahagiaan dan kesuksesan setelah ujian dan cobaan yang ada.

Salah satu algoritma graf dalam pencarian rute terpendek adalah algoritma *Floyd Warshall*. Algoritma *Floyd Warshall* adalah algoritma yang menerapkan pemrograman yang dinamis, yaitu metode penyelesaian permasalahan dengan melihat dari hasil yang hendak dicapai sebagai suatu kesimpulan yang saling berhubungan. Algoritma ini akan mencapai keberhasilan dalam penemuan solusi rute terpendek karena membandingkan semua rute yang memungkinkan pada graf untuk setiap sisi dari semua sisi yang dilewati. Hal tersebut yang menjadi dasar mengapa dipilih algoritma *Floyd Warshall* untuk diterapkan pada penelitian pencarian rute terpendek lokasi BTS milik PT Citra Akses Indonusa.

Beberapa penelitian mengenai algoritma *Floyd Warshall*, contohnya penelitian oleh Vera Apriliani Nawagusti pada tahun 2018 yang membahas "Penerapan Algoritma *Floyd Warshall* dalam Aplikasi Penentuan Rute Terpendek Mencari Lokasi BTS (*Base Tower Station*) pada PT. GCI Palembang"

menunjukkan bahwa algoritma ini memiliki hasil yang akurat dalam menentukan rute terpendek. Selain itu, penelitian yang dilakukan oleh Dewi Isnaini (2019) dengan pembahasan "Pencarian Rute Terpendek *Non-Player Character* (*NPC*) dengan Metode *Floyd Warshall* pada *Game* Wisata Batu" menunjukkan hasil uji coba dengan algoritma tersebut akurat sebesar 94.44% dalam menentukan jarak terpendek. Paparan penelitian-penelitian tersebut menyatakan bahwa hasil implementasi algoritma *Floyd Warshall* cukup baik, sedangkan pada penelitian kali ini akan dilakukan penerapan Algoritma *Floyd Warshall* dalam pencarian rute atau jalur terpendek lokasi *tower Base Transceiver Station* (BTS) pada PT. Citra Akses Indonusa di Kota Tangerang dan Kabupaten Tangerang.

Beberapa uraian di atas melatarbelakangi penulis melakukan penelitian dengan judul "Implementasi Algoritma *Floyd Warshall* Dalam Pencarian Rute Terpendek Lokasi *tower Base Transceiver Station* (BTS) Pada PT Citra Akses Indonusa". Diharapkan hasil penelitian ini mampu dijadikan sebagai sarana penunjang pada perusahaan PT Citra Akses Indonusa sebagai bentuk efektivitas para pegawai dalam menemukan rute terpendek dalam melakukan pemeliharaan BTS.

1.2 Rumusan Masalah

Dari paparan latar belakang di atas didapatkan perolehan rumusan masalah, yaitu bagaimana implementasi algoritma *Floyd Warshall* dalam menentukan rute/jalur terpendek lokasi *tower* BTS PT Citra Akses Indonusa di Kabupaten dan Kota Tangerang?

1.3 Tujuan Penelitian

Pada penelitian ini, tujuan yang ingin diperoleh sesuai dengan rumusan masalah yang sudah disebutkan adalah untuk mengetahui hasil dari implementasi algoritma *Floyd Warshall* dalam menentukan rute/jalur terpendek lokasi *tower* BTS PT Citra Akses Indonusa di Kabupaten dan Kota Tangerang.

1.4 Manfaat Penelitian

Berdasarkan latar belakang di atas yang telah dijabarkan, didapatkan beberapa manfaat dari penelitian ini adalah:

1. Untuk menambah dan memperdalam wawasan tentang algoritma *Floyd Warshall* sebagai pencarian rute/jalur terpendek.
2. Pihak PT Citra Akses Indonusa dapat mengetahui rute/jalur terpendek lokasi BTS di Kabupaten dan Kota Tangerang.
3. Dapat digunakan oleh penelitian selanjutnya sebagai materi untuk mempelajari dan mengaplikasikan algoritma *Floyd Warshall* dalam pencarian rute/jalur terpendek.
4. Mengembangkan sarana keilmuan matematika dan komputasi bagi Fakultas Sains dan Teknologi, Universitas Islam Negeri Maulana Malik Ibrahim Malang.

1.5 Batasan Masalah

Batasan-batasan permasalahan yang akan diberikan dalam penelitian ini, yaitu:

1. Penelitian ini mengambil jalur pertigaan atau perempatan yang dilalui sebagai titik, tetapi tidak dijadikan sebagai titik tujuan.

2. Ruang lingkup objek yang digunakan adalah lokasi *tower* BTS milik PT Citra Akses Indonusa yang berada di daerah Kabupaten dan Kota Tangerang.
3. Penentuan bobot rute terdekat yang digunakan tidak berpengaruh terhadap waktu.
4. Rute yang diambil pada penelitian ini merupakan rute jalan umum yang bisa dilalui oleh kendaraan roda empat.

BAB II

TINJAUAN PUSTAKA

2.1 Graf

Leonhard Euler pertama kali mempublikasikan teori tentang graf pada tahun 1736 untuk menyelesaikan solusi dari permasalahan jembatan Konigsberg (Cipta,2019). Selanjutnya satu abad kemudian di tahun 1874 seorang matematikawan bernama G.E Kirchof berhasil mengembangkan *Thoery of Trees* atau yang dikenal dengan teori pohon yang digunakan di dalam persoalan jaringan listrik.

Graf G merupakan pasangan $(V(G), E(G))$ dengan $V(G)$ merupakan himpunan tidak kosong dan berhingga dari objek-objek yang disebut titik dan $E(G)$ adalah himpunan (mungkin kosong) pasangan takberurutan dari titik-titik berbeda di $V(G)$ yang disebut sisi. Banyaknya objek di $V(G)$ disebut *order* dari G dan dituliskan dengan $p(G)$, dan banyaknya objek di $E(G)$ disebut ukuran dari G dan dituliskan dengan $q(G)$. Jika graf yang dibahas hanya graf G, maka ukuran dan order dari G masing-masing cukup ditulis p dan q. (Chartrand G. & Lesniak L., 2000)

Sisi $e = (u, v)$ dikatakan menghubungkan titik u dan v. Jika $e = (u, v)$ merupakan sisi di graf G, maka u dan v disebut terhubung langsung (*adjacent*), v dan e serta u dan e disebut terkait langsung (*incident*), dan titik u dan v disebut ujung dari e. Dua sisi berbeda e_1 dan e_2 disebut terhubung langsung (*adjacent*) jika terkait langsung pada satu titik yang sama. Selanjutnya, sisi $e = (u, v)$ akan ditulis $e = uv$ (Chartrand G. & Lesniak L., 2000).

Berdasarkan jenis sisinya, graf dapat dibagi menjadi dua, yaitu graf berarah (*directed graph*) dan graf tak berarah (*undirected graph*). Graf berarah adalah graf yang semua sisinya memiliki arah. Apabila semua sisinya tidak memiliki arah, maka graf tersebut disebut sebagai graf tak berarah. Jika disebutkan dengan kata graf saja, maka yang dimaksudkan adalah graf tak berarah. Suatu graf biasanya digambarkan secara grafis, dengan setiap titik digambarkan sebagai titik atau lingkaran kecil, dan setiap sisi $e = uv$ digambarkan dengan sebuah sisi atau kurva yang menghubungkan u dan v (Budayasa, 2016).

Dalam menyelesaikan persoalan rute terpendek dengan graf, terdapat proses-proses yang dilakukan untuk menggambarkan lokasi-lokasi rute, yaitu (Surendro, 2007):

1. Gabungkan setiap titik lokasi menjadi graf terhubung.
2. Berikan arah sisi pada graf yang membentuk graf berarah dari graf terhubung yang ada.
3. Berikan informasi jarak dari rute yang ditempuh menjadi bobot jarak. Gunakan bobot jarak yang ada menjadi bobot dari graf berarah sehingga menjadi graf berbobot.

2.2 Algoritma Floyd Warshall

Pada tahun 1962 Robert Floyd menemukan algoritma yang diberi nama dengan *Floyd Warshall*. Algoritma ini merupakan salah satu bagian dari pemrograman dinamis, yaitu metode penyelesaian permasalahan dengan melihat dari hasil yang hendak dicapai sebagai suatu kesimpulan yang saling berhubungan. *Input* dari algoritma *Floyd Warshall* adalah graf berarah dan graf

berbobot. Serta *output* atau hasil dari algoritma ini adalah rute terpendek dari semua pasangan titik (Saputra, 2011).

Algoritma *Floyd Warshall* tidak hanya mencari jarak terpendek antara dua buah titik tertentu, tetapi langsung membuat tabel sisi terpendek antar titik (Rully,dkk.,2018). Menurut penelitian Novandi (2007), algoritma *Floyd Warshall* adalah sebuah algoritma analisis graf untuk mencari bobot minimum dari graf berarah. Dalam pengertian lain, algoritma ini merupakan suatu metode yang melakukan pemecahan masalah dengan memandang solusi yang akan diperoleh sebagai suatu keputusan yang saling terkait. Artinya solusi-solusi tersebut dibentuk dari solusi yang berasal dari tahap sebelumnya dan ada kemungkinan solusi tersebut lebih dari satu.

Kelebihan dari algoritma *Floyd Warshall* (Mukti, dkk., 2008), yaitu:

1. Algoritma *Floyd Warshall* dapat digunakan untuk mencari jarak terpendek (*shortest path*) dari setiap pasangan node
2. Algoritma *Floyd Warshall* menggunakan matriks bobot berukuran $n \times n$ sebagai masukan, di mana n merupakan jumlah titik
3. Algoritma *Floyd Warshall* dapat mentoleransi sisi negatif (*negative edge*).

Dalam menyelesaikan permasalahan algoritma ini, digunakan matriks ketetanggaan antar titik yang berukuran $n \times n$ di mana n merupakan jumlah titik. Jika terdapat n titik, maka matriks tersebut diproses mulai dari iterasi $k = 1$ hingga $k = n$. Selanjutnya untuk setiap iterasi matriks diproses kembali sesuai dengan titik yang dijadikan titik ketetanggaan. Dalam pemrosesan satu matriks digunakan rumus sebagai berikut (Letivany,2018):

$$F_{[i,j]}^{k} = \min\left\{F_{[i,j]}^{k-1}, F_{[i,k]}^{k-1} + F_{[k,j]}^{k-1}\right\} \quad (1)$$

Di mana $F_{[i,j]}^{k}$ menyatakan entri matriks F ketetanggaan berbobot, baris ke-i, kolom ke-j di iterasi ke-k.

Tahapan-tahapan dari algoritma *Floyd Warshall* untuk mencari rute terpendek secara umum adalah sebagai berikut (Letivany,2018):

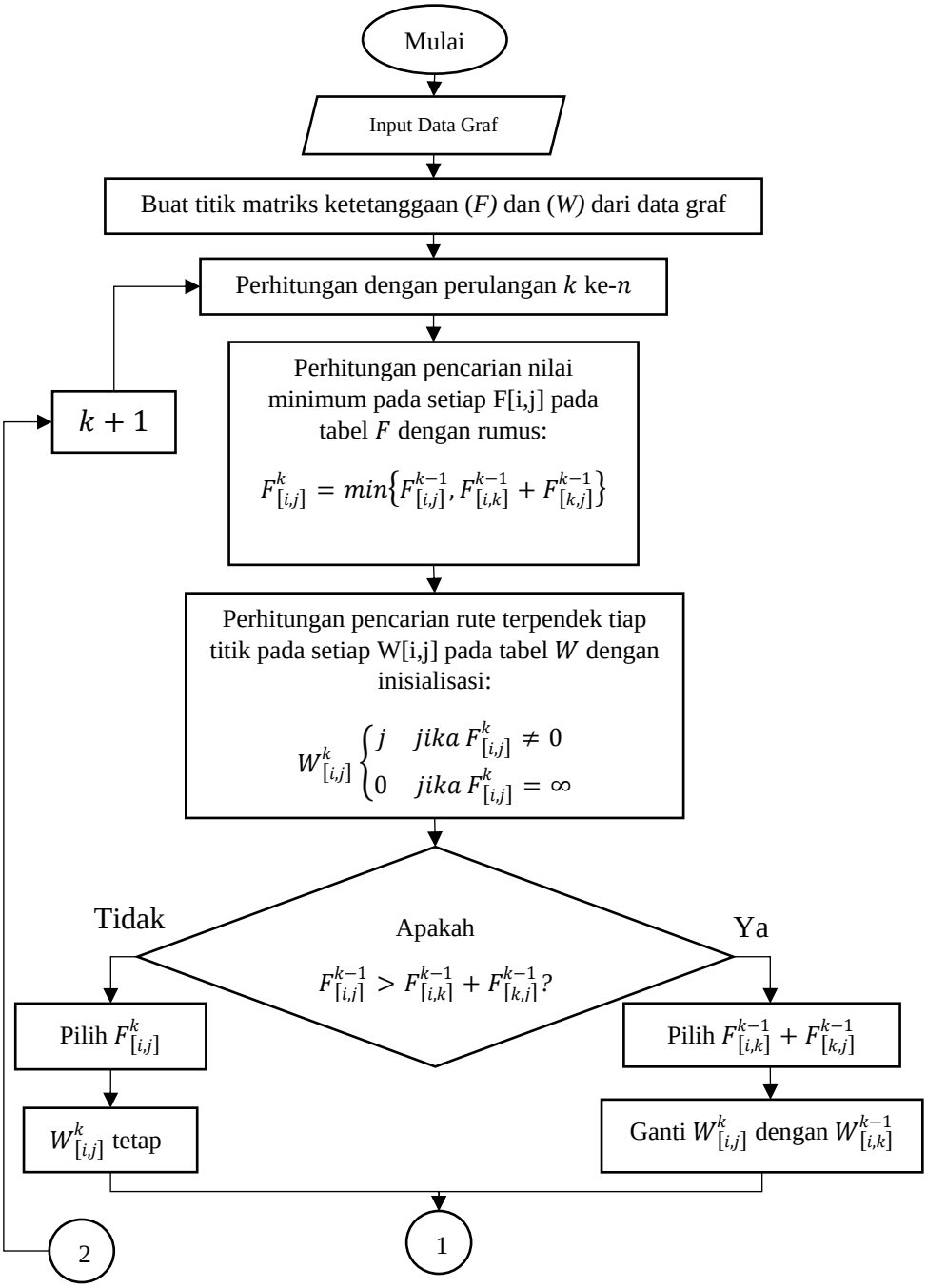
1. Menentukan semua bobot jarak dari titik yang akan dilalui sesuai pada graf.
2. Merepresentasikan semua bobot jarak tiap titik ke dalam bentuk matriks berukuran $n \times n$.
3. Menghitung semua rute terpendek dengan membandingkan nilai pada $F_{[i,j]}$ dengan penjumlahan $F_{[i,k]} + F_{[k,j]}$. Berdasarkan dua nilai tersebut maka akan dipilih nilai yang paling minimum.
4. Melakukan perhitungan berulang hingga $k = n$ atau sebanyak *titiknya*.
5. Menentukan jarak terpendek dari rute yang dicari dari hasil perhitungan.

Langkah-langkah berikut hanya mendapatkan hasil nilai jarak terpendek dari tiap-tiap titik lokasi *tower* BTS. Maka berdasarkan hal tersebut diperlukan modifikasi tambahan untuk menentukan jalur/rute terpendek. Modifikasi yang dilakukan adalah dengan menambahkan satu matriks $n \times n$ yang berdampingan dengan matriks nilai jarak terpendek, yang disusun sebagai berikut:

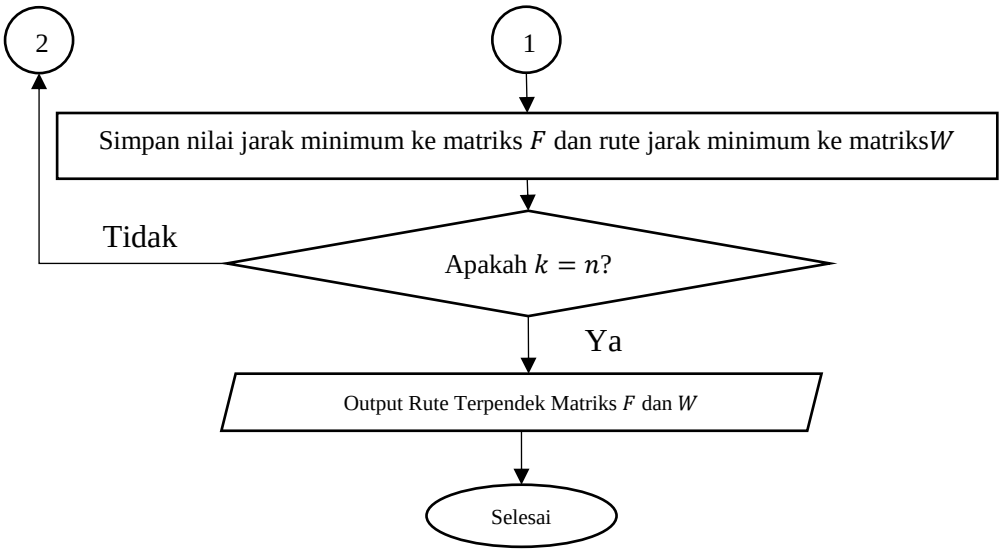
$$W_{[i,j]}^{k} = \begin{cases} j & jika\ F_{[i,j]}^{k} \neq 0 \\ 0 & jika\ F_{[i,j]}^{k} = \infty \end{cases} \quad (2)$$

Di mana $W_{[i,j]}^{k}$ menyatakan entri matriks W yang berisi simbol lokasi titik, baris ke-i, kolom ke-j di iterasi ke-k.

Berdasarkan modifikasi tersebut, ketika dalam iterasi ke-k nilai $F_{[i,j]}$ lebih besar dari $F_{[i,k]} + F_{[k,j]}$, maka akan dipilih nilai $F_{[i,k]} + F_{[k,j]}$ yang juga dilakukan dengan mengganti $W_{[i,j]}^{k}$ dengan $W_{[i,k]}^{k-1}$. Berikut merupakan gambar *flowchart* dari algoritma *Floyd Warshall*:

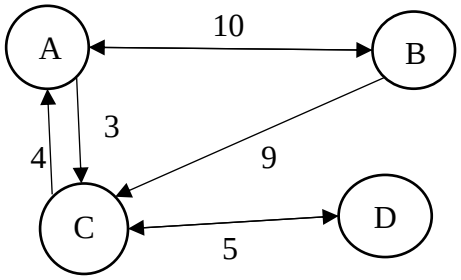


Gambar 2.1 *Flowchart Algoritma Floyd Warshall Bagian 1*



Gambar 2.2 *Flowchart Algoritma Floyd Warshall Bagian 2*

Contoh pengerjaan algoritma *Floyd Warshall* adalah sebagai berikut:



Gambar 2.3 *Contoh Graf Berarah*

Berdasarkan gambar graf tersebut dengan satuan bobot kilometer, kemudian masuk ke dalam proses mencari rute terpendek dengan menjumlahkan tiap rute yang ingin dikunjungi dari titik awal lokasi hingga ke lokasi yang menjadi tujuan. Dimisalkan akan ditentukan rute terpendek dari lokasi A ke lokasi D.

1. Langkah pertama diawali dengan membuat matriks ketetanggaan F^0 dan W^0 dari graf berukuran $n \times n$ yang di mana n merupakan banyak titik lokasi yang digunakan. Kemudian masukkan bobot jarak ke dalam matriks F^0 dan kita masukkan juga simbol titik yang dilalui ke dalam matriks W^0. Kita dapatkan hasil sebagai berikut:

Tabel 2.1 *Matriks F^0dan W^0*

F^0	A	B	C	D
A	0	10	3	∞
B	10	0	9	∞
C	4	∞	0	5
D	∞	∞	5	0

W^0	A	B	C	D
A	0	B	C	∞
B	A	0	C	∞
C	A	∞	0	D
D	∞	∞	C	0

2. Pada langkah kedua, mulai perhitungan iterasi *k* ke-1 dari matriks $\boldsymbol{F^0}$ **dan** $\boldsymbol{W^0}$. Pada proses ini akan dibuat matriks baru $\boldsymbol{F^1}$ dan $\boldsymbol{W^1}$ yang di mana setiap elemen pada baris ke-1 dan kolom ke-1 dari matriks $\boldsymbol{F^1}$ dan $\boldsymbol{W^1}$ sama dengan elemen pada baris dan kolom ke-1 di $\boldsymbol{F^0}$ dan $\boldsymbol{W^0}$.

Tabel 2.2 *Perhitungan Matriks F^1 dan W^1*

F^1	A	B	C	D
A	0	10	3	∞
B	10	0		
C	4		0	
D	∞			0

W^1	A	B	C	D
A	0	B	C	∞
B	A	0		
C	A		0	
D	∞			0

Selanjutnya, untuk baris dan kolom dari matriks $\boldsymbol{F^1}$ yang masih kosong, akan dicari pencarian nilai minimum menggunakan rumus algoritma *Floyd Warshall* $F^1_{[i,j]} = \min\{F^0_{[i,j]}, F^0_{[i,1]} + F^0_{[1,j]}\}$:

Untuk elemen matriks [2,3]:
$F^1_{[2,3]} = \min\{F^0_{[2,3]}, F^0_{[2,1]} + F^0_{[1,3]}\}$
$F^1_{[2,3]} = \min\{9, 10 + 3\}$
$F^1_{[2,3]} = 9$

Untuk elemen matriks [3,2]:
$F^1_{[3,2]} = \min\{F^0_{[3,2]}, F^0_{[3,1]} + F^0_{[1,2]}\}$
$F^1_{[3,2]} = \min\{\infty, 4 + 10\}$
$F^1_{[3,2]} = 14$

Untuk elemen matriks [4,2]:
$F^1_{[4,2]} = \min\{F^0_{[4,2]}, F^0_{[4,1]} + F^0_{[1,2]}\}$
$F^1_{[4,2]} = \min\{\infty, \infty + 10\}$
$F^1_{[4,2]} = \infty$

Untuk elemen matriks [2,4]:
$F^1_{[2,4]} = \min\{F^0_{[2,4]}, F^0_{[2,1]} + F^0_{[1,4]}\}$
$F^1_{[2,4]} = \min\{\infty, 10 + \infty\}$
$F^1_{[2,4]} = \infty$

Untuk elemen matriks [3,4]:
$F^1_{[3,4]} = \min\{F^0_{[3,4]}, F^0_{[3,1]} + F^0_{[1,4]}\}$
$F^1_{[3,4]} = \min\{5, 4 + \infty\}$
$F^1_{[3,4]} = 5$

Untuk elemen matriks [4,3]:
$F^1_{[4,3]} = \min\{F^0_{[4,3]}, F^0_{[4,1]} + F^0_{[1,3]}\}$
$F^1_{[4,3]} = \min\{5, \infty + 3\}$
$F^1_{[4,3]} = 5$

Dikarenakan adanya perubahan nilai matriks $\boldsymbol{F^1}$ pada elemen [3,2], maka nilai simbol rute di matriks $\boldsymbol{W^1}$ pada elemen [3,2] juga berubah sesuai dengan kaidah modifikasi rumus algoritma *Floyd Warshall*. Didapatkan hasil untuk elemen-elemen matriks $\boldsymbol{F^1}$ dan $\boldsymbol{W^1}$ dengan $k = 1$ adalah sebagai berikut:

Tabel 2.3 *Matriks F^1 dan W^1*

$\boldsymbol{F^1}$ $k = 1$	**A**	**B**	**C**	**D**
A	0	10	3	∞
B	10	0	9	∞
C	4	14	0	5
D	∞	∞	5	0

$\boldsymbol{W^1}$ $k = 1$	**A**	**B**	**C**	**D**
A	0	B	C	∞
B	A	0	C	∞
C	A	A	0	D
D	∞	∞	C	0

3. Langkah selanjutnya proses untuk $k = 2$ pada matriks $\boldsymbol{F^1}$ dan $\boldsymbol{W^1}$. Pada proses ini akan dibuat matriks baru $\boldsymbol{F^2}$ dan $\boldsymbol{W^2}$ yang di mana pada proses ini mengulang kembali proses pertama tetapi dengan perbedaan elemen yang digunakan yaitu pada baris dan kolom yang kedua dari matriks $\boldsymbol{F^1}$ dan $\boldsymbol{W^1}$.

Tabel 2.4 *Perhitungan Matriks F^2 dan W^2*

$\boldsymbol{F^2}$	**A**	**B**	**C**	**D**
A	0	10		
B	10	0	9	∞
C		14	0	
D		∞		0

$\boldsymbol{W^2}$	**A**	**B**	**C**	**D**
A	0	B		
B	A	0	C	∞
C		A	0	
D		∞		0

Kemudian, nilai minimum untuk baris dan kolom dari matriks $\boldsymbol{F^2}$ yang masih kosong akan dicari menggunakan rumus algoritma *Floyd Warshall* $F^2_{[i,j]} = \min\{F^1_{[i,j]}, F^1_{[i,2]} + F^1_{[2,j]}\}$:

Untuk elemen matriks [1,3]:
$F^2_{[1,3]} = \min\{F^1_{[1,3]}, F^1_{[1,2]} + F^1_{[2,3]}\}$
$F^2_{[1,3]} = \min\{3, 10 + 9\}$
$F^2_{[1,3]} = 3$

Untuk elemen matriks [1,4]:
$F^2_{[1,4]} = \min\{F^1_{[1,4]}, F^1_{[1,2]} + F^1_{[2,4]}\}$
$F^2_{[1,4]} = \min\{\infty, 10 + \infty\}$
$F^2_{[1,4]} = \infty$

Untuk elemen matriks [3,1]:
$F^2_{[3,1]} = \min\{F^1_{[3,1]}, F^1_{[3,2]} + F^1_{[2,1]}\}$
$F^2_{[3,1]} = \min\{4, 14 + 10\}$
$F^2_{[3,1]} = 4$

Untuk elemen matriks [3,4]:
$F^2_{[3,4]} = \min\{F^1_{[3,4]}, F^1_{[3,2]} + F^1_{[2,4]}\}$
$F^2_{[3,4]} = \min\{5, 14 + \infty\}$
$F^2_{[3,4]} = 5$

Untuk elemen matriks [4,1]:
$F^2_{[4,1]} = \min\{F^1_{[4,1]}, F^1_{[4,2]} + F^1_{[2,1]}\}$
$F^2_{[4,1]} = \min\{\infty, \infty + 10\}$
$F^2_{[4,1]} = \infty$

Untuk elemen matriks [4,3]:
$F^2_{[4,3]} = \min\{F^1_{[4,3]}, F^1_{[4,2]} + F^1_{[2,3]}\}$
$F^2_{[4,3]} = \min\{5, \infty + 13\}$
$F^2_{[4,3]} = 5$

Dihasilkan elemen-elemen matriks $\boldsymbol{F^2}$ dan $\boldsymbol{W^2}$ dengan $k = 2$ adalah sebagai berikut:

Tabel 2.5 *Matriks F^2 dan W^2*

$\boldsymbol{F^2}$ $k = 2$	**A**	**B**	**C**	**D**
A	0	10	3	∞
B	10	0	9	∞
C	4	14	0	5
D	∞	∞	5	0

$\boldsymbol{W^2}$ $k = 2$	**A**	**B**	**C**	**D**
A	0	B	C	∞
B	A	0	C	∞
C	A	A	0	D
D	∞	∞	C	0

4. Langkah berikutnya proses untuk $k = 3$ pada matriks $\boldsymbol{F^2}$ dan $\boldsymbol{W^2}$ akan dibuat matriks baru $\boldsymbol{F^3}$ dan $\boldsymbol{W^3}$ yang di mana pada proses ini mengulang kembali seperti proses sebelumnya tetapi dengan perbedaan elemen yang digunakan yaitu pada baris dan kolom yang ketiga dari matriks $\boldsymbol{F^2}$ dan $\boldsymbol{W^2}$.

Tabel 2.6 *Perhitungan Matriks F^3dan W^3*

$\boldsymbol{F^3}$	**A**	**B**	**C**	**D**
A	0		3	
B		0	9	
C	4	14	0	5
D			5	0

$\boldsymbol{W^3}$	**A**	**B**	**C**	**D**
A	0		C	
B		0	C	
C	A	A	0	D
D			C	0

Selanjutnya, akan dicari nilai minimum untuk baris dan kolom dari matriks $\boldsymbol{F^3}$ yang masih kosong, menggunakan rumus algoritma *Floyd Warshall* $F^3_{[i,j]} = \min\{F^2_{[i,j]}, F^2_{[i,3]} + F^2_{[3,j]}\}$:

Untuk elemen matriks [1,2]:
$F^3_{[1,2]} = \min\{F^2_{[1,2]}, F^2_{[1,3]} + F^2_{[3,2]}\}$
$F^3_{[1,2]} = \min\{10,3 + 14\}$
$F^3_{[1,2]} = 10$

Untuk elemen matriks [1,4]:
$F^3_{[1,4]} = \min\{F^2_{[1,4]}, F^2_{[1,3]} + F^2_{[3,4]}\}$
$F^3_{[1,4]} = \min\{\infty, 3 + 5\}$
$F^3_{[1,4]} = 8$

Untuk elemen matriks [2,1]:
$F^3_{[2,1]} = \min\{F^2_{[2,1]}, F^2_{[2,3]} + F^2_{[3,1]}\}$
$F^3_{[2,1]} = \min\{10,9 + 4\}$
$F^3_{[2,1]} = 10$

Untuk elemen matriks [2,4]:
$F^3_{[2,4]} = \min\{F^2_{[2,4]}, F^2_{[2,3]} + F^2_{[3,4]}\}$
$F^3_{[2,4]} = \min\{\infty, 9 + 5\}$
$F^3_{[2,4]} = 14$

Untuk elemen matriks [4,1]:
$F^3_{[4,1]} = \min\{F^2_{[4,1]}, F^2_{[4,3]} + F^2_{[3,1]}\}$
$F^3_{[4,1]} = \min\{\infty, 5 + 4\}$
$F^3_{[4,1]} = 9$

Untuk elemen matriks [4,2]:
$F^3_{[4,2]} = \min\{F^2_{[4,2]}, F^2_{[4,3]} + F^2_{[3,2]}\}$
$F^3_{[4,2]} = \min\{\infty, 5 + 14\}$
$F^3_{[4,2]} = 19$

Dikarenakan adanya perubahan nilai matriks $\boldsymbol{F^3}$ pada elemen [1,4], [2,4], [4,1] dan [4,2], maka nilai simbol rute di matriks $\boldsymbol{W^3}$ pada elemen tersebut juga berubah seperti pada proses iterasi ke-1. Diperoleh hasil elemen-elemen matriks $\boldsymbol{F^3}$ dan $\boldsymbol{W^3}$ dengan $k = 3$ sebagai berikut:

Tabel 2.7 *Matriks F^3 dan W^3*

$\boldsymbol{F^3}$ $k = 3$	**A**	**B**	**C**	**D**
A	0	10	3	8
B	10	0	9	14
C	4	14	0	5
D	9	19	5	0

$\boldsymbol{W^3}$ $k = 3$	**A**	**B**	**C**	**D**
A	0	B	C	C
B	A	0	C	C
C	A	A	0	D
D	C	C	C	0

5. Kemudian untuk proses $k = 4$ pada matriks $\boldsymbol{F^3}$ dan $\boldsymbol{W^3}$ akan dibuat matriks baru $\boldsymbol{F^4}$ dan $\boldsymbol{W^4}$ yang di mana pada proses ini mengulang kembali proses pertama, kedua dan ketiga tetapi dengan perbedaan elemen yang digunakan yaitu pada baris dan kolom yang keempat dari matriks $\boldsymbol{F^3}$ dan $\boldsymbol{W^3}$.

Tabel 2.8 *Perhitungan Matriks* F^4 *dan* W^4

F^4	A	B	C	D
A	0			8
B		0		14
C			0	5
D	9	19	5	0

W^4	A	B	C	D
A	0			C
B		0		C
C			0	D
D	C	C	C	0

Proses pencarian nilai minimum untuk baris dan kolom dari matriks $\boldsymbol{F^4}$ yang masih kosong, digunakan rumus algoritma *Floyd Warshall* $F^4_{[i,j]} = \min\{F^3_{[i,j]}, F^3_{[i,4]} + F^3_{[4,j]}\}$:

Untuk elemen matriks [1,2]:
$F^4_{[1,2]} = \min\{F^3_{[1,2]}, F^3_{[1,4]} + F^3_{[4,2]}\}$
$F^4_{[1,2]} = \min\{10, 8 + 19\}$
$F^4_{[1,2]} = 10$

Untuk elemen matriks [1,3]:
$F^4_{[1,3]} = \min\{F^3_{[1,3]}, F^3_{[1,4]} + F^3_{[4,3]}\}$
$F^4_{[1,3]} = \min\{3, 8 + 5\}$
$F^4_{[1,3]} = 3$

Untuk elemen matriks [2,1]:
$F^4_{[2,1]} = \min\{F^3_{[2,1]}, F^3_{[2,4]} + F^3_{[4,1]}\}$
$F^4_{[2,1]} = \min\{10, 14 + 9\}$
$F^4_{[2,1]} = 10$

Untuk elemen matriks [3,1]:
$F^4_{[3,1]} = \min\{F^3_{[3,1]}, F^3_{[3,4]} + F^3_{[4,1]}\}$
$F^4_{[3,1]} = \min\{4, 5 + 9\}$
$F^4_{[3,1]} = 4$

Untuk elemen matriks [2,3]:
$F^4_{[2,3]} = \min\{F^3_{[2,3]}, F^3_{[2,4]} + F^3_{[4,3]}\}$
$F^4_{[2,3]} = \min\{9, 14 + 5\}$
$F^4_{[2,3]} = 9$

Untuk elemen matriks [3,2]:
$F^4_{[3,2]} = \min\{F^3_{[3,2]}, F^3_{[3,4]} + F^3_{[4,2]}\}$
$F^4_{[3,2]} = \min\{14, 5 + 19\}$
$F^4_{[3,2]} = 14$

Didapatkan hasil untuk elemen-elemen matriks $\boldsymbol{F^4}$dengan $k = 4$ adalah sebagai berikut:

Tabel 2.9 *Matriks* F^4 *dan* W^4

F^4 $k = 4$	A	B	C	D
A	0	10	3	8
B	10	0	9	14
C	4	14	0	5
D	9	19	5	0

W^4 $k = 4$	A	B	C	D
A	0	B	C	C
B	A	0	C	C
C	A	A	0	D
D	C	C	C	0

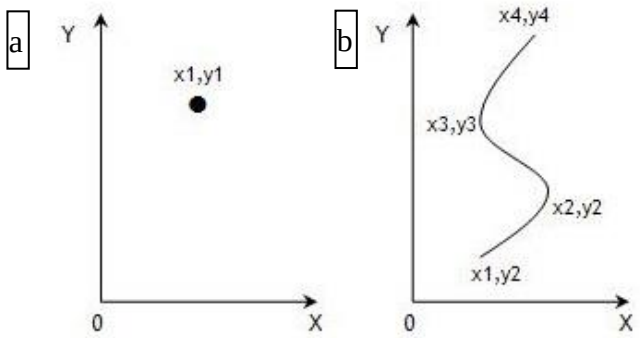
6. Setelah perhitungan iterasi $k = n$, yang pada contoh pengerjaan algoritma *Floyd Warshall* ini $k = 4$, maka didapatkan seluruh hasil rute terpendek antar titik pada matriks $\boldsymbol{F}$ dan $\boldsymbol{W}$ dari matriks $\boldsymbol{F^4}$ dan $\boldsymbol{W^4}$. Kemudian kita mencari rute terpendek dari rute yang ingin kita cari yaitu dari lokasi A lokasi D. Berdasarkan hasil matriks $\boldsymbol{F}$ didapatkan bahwa jarak rute terpendek dari lokasi A ke lokasi D adalah 8 Km yang di mana rute yang dilewati berdasarkan matriks $\boldsymbol{W}$ adalah A – C – D.

2.3 Sistem Informasi Geografis (SIG)

Sistem informasi geografis adalah suatu sistem yang menyajikan informasi yang menggunakan peta sebagai *interface* ke dalam bentuk grafis. Peta pada SIG ini tersusun atas konsep beberapa layer dan relasi yang memproses dan menayangkan data spasial digital. Menurut Aronoff (1989), Sistem Informasi Geografis atau yang disingkat dengan SIG merupakan suatu sistem komputer yang mempunyai kemampuan untuk pemasukan, pengambilan, analisis dan tampilan data geografis yang diutamakan untuk sebuah pengambilan keputusan. Sistem ini didukung oleh tiga komponen komputer yaitu perangkat keras (*hardware*), lunak (*software*) dan sumber daya manusia (*brainware*) yang dirancang secara efisien untuk memasukkan, memanipulasi, memperbaharui, menganalisis, menyimpan dan menyajikan semua informasi yang orientasinya berupa geografis (Nugroho,2020).

Data spasial merupakan salah satu jenis data geografi pada SIG. Data tersebut memiliki arti data yang berkaitan dengan lokasi, posisi dan area pada koordinat tertentu (Sulistiyanto, 2021). Data spasial dapat direpresentasikan dalam dua format, salah satu format tersebut adalah format data vektor. Dalam

membangun objek spasialnya data vektor dibagi lagi menjadi beberapa macam data, diantara-Nya data titik (*point*) dan data sisi (*line/polyline*)(Awangga, 2019). Data titik pada umumnya digunakan untuk menggambarkan objek stasiun, curah hujan, alamat *customer*, kota, dan lain sebagainya. Data sisi dapat digunakan untuk menggambarkan jalan, sungai, jaringan listrik, dan lain sebagainya. Pada struktur sebuah data vektor, posisi dari objek ditulis pada sistem koordinat (koordinat Cartesius). Berikut merupakan contoh ilustrasi tentang data titik dan sisi pada data vektor:



Gambar 2.4 *Contoh Data Vektor, a) Titik dan b) Sisi*

2.4 *Base Transceiver Station* (BTS)

BTS merupakan sebuah alat dalam jaringan telekomunikasi berupa *tower* yang memiliki suatu antena yang memancarkan sinyal dan berfungsi sebagai penguat sinyal daya yang menghubungkan antara jaringan operator telekomunikasi dengan para *customer*. BTS memiliki daerah yang luas cakupannya sesuai dengan kuat lemahnya pancaran sinyal dari daya dikirimkan ke para pengguna. Sistem yang digunakan sebagian besar dari mereka adalah sistem GSM (*Global System For Mobile Communication*) (Nanang Ismail, dkk., 2015).

Tower telekomunikasi seluler/*tower* BTS adalah alat yang berfungsi untuk menempatkan antena pemancar sinyal (jaringan akses) untuk memberikan layanan kepada pelanggan di sekitar tower. Penentuan lokasi *tower* BTS untuk

jaringan telepon seluler menjadi masalah yang sering dihadapi oleh pihak operator penyedia jaringan komunikasi seluler. Operator dituntut untuk dapat menentukan lokasi *tower* BTS yang potensial agar semua wilayah dapat terjangkau sinyalnya. Berbagai parameter menjadi bahan pertimbangan dalam menentukan perencanaan pembangunan *tower* BTS, baik itu dari segi teknis dan keadaan sosial kemasyarakatan (Arif dan Bambang, 2007).

2.5 Kajian Keislaman

Hemat memiliki pengertian sifat yang penuh perhitungan, cermat dan berhati hati dalam melakukan segala suatu hal yang berhubungan dengan manusia. Dalam konteks keislaman merupakan suatu sikap yang berarti tidak kikir atau pelit dan tidak berlebih-lebihan pada sesuatu atau boros. Dalam ajarannya agama Islam telah menganjurkan bahwa setiap umat manusia harus hidup hemat. Salah satu contoh kasus dalam hidup hemat adalah ketika sedang melakukan suatu perjalanan kita memilih rute terpendek yang di mana dapat menghemat tenaga dan bahan bakar kendaraan yang digunakan.

Dalam Al-Quran yang menjelaskan tentang sifat boros yang merupakan perbuatan merugikan bagi manusia yaitu terdapat pada surat Al-Isra ayat 27:

إِنَّ ٱلْمُبَذِّرِينَ كَانُوٓاْ إِخْوَٰنَ ٱلشَّيَٰطِينِ ۖ وَكَانَ ٱلشَّيْطَٰنُ لِرَبِّهِۦ كَفُورًا

Artinya: "*Sesungguhnya pemboros-pemboros itu adalah saudara-saudara setan dan setan itu adalah sangat ingkar kepada Tuhannya*." (Q.S Al-Isra:27)

Sudah jelas bahwa ayat ini menjelaskan tentang larangan bagi umat muslim melakukan pemborosan, karena pemborosan merupakan perbuatan yang dilarang oleh Allah Swt. yang di mana perbuatan tersebut tidak membawa manfaat. Bagi manusia yang melakukan perbuatan tersebut diumpamakan bersaudara dengan setan dan ingkar kepada Allah Swt. (Hamka, 1982).

Berdasarkan Tafsir Ibnu Katsir jilid 5 (2003) ayat ini mengandung penjelasan bahwa mereka yang melakukan pemborosan menjadi orang yang serupa dengan setan dan bersaudara dalam keborosan, kebodohan, kemaksiatan dan pengabaian terhadap ketaatan kepada Allah swt. Oleh karena itu yang melakukan hal tersebut benar-benar ingkar yang berarti setan itu telah mengingkari nikmat yang diberikan oleh Allah Swt. dan sama sekali tidak mau berbuat taat kepada-Nya bahkan cenderung durhaka dan menyalahi-Nya. Berdasarkan hal tersebut menandakan kita untuk senantiasa berhemat dan tidak melakukan hal-hal yang berlebihan, sebagaimana dalam melakukan suatu perjalanan dilakukan dengan mencari rute terpendek untuk mencapai definisi berhemat yang dianjurkan oleh Islam.

BAB III

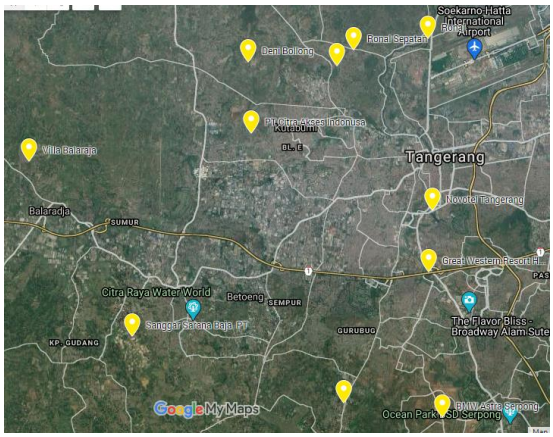
METODE PENELITIAN

3.1 Pendekatan Penelitian

Penelitian ini menggunakan pendekatan studi lapangan dan penelitian studi evaluasi kuantitatif. Studi lapangan merupakan acuan utama dalam pengambilan titik lokasi yang diperoleh atau bersumber dari PT Citra Akses Indonusa. Penelitian studi evaluasi kuantitatif berarti mengukur keberhasilan suatu kegiatan/program dengan mengumpulkan data berupa angka-angka atau data numerik yang bisa dihitung sesuai dengan aturan penelitian.

3.2 Sumber Data

Data pada penelitian ini menggunakan data sekunder lokasi *tower Base Transceiver Station* (BTS) yang bersumber dari PT Citra Akses Indonusa. Data lokasi *tower* BTS ini diambil berdasarkan pengguna yang bekerja sama dengan PT Citra Akses Indonusa pada tahun 2020. Lokasi titik penelitian ini terdiri atas delapan titik yang berada di Kabupaten dan tiga titik di Kota Tangerang. Data yang sudah didapatkan tersebut kemudian diimplementasikan ke dalam suatu titik melalui aplikasi *Google Maps* sebagai berikut:



Gambar 3.1 *Titik Lokasi Tower BTS*

3.3 Variabel Penelitian

Pada penelitian ini menggunakan variabel-variabel yang disajikan dengan keterangan dan penjelasan definisi dalam bentuk tabel sebagai berikut:

Tabel 3.1 *Variabel Penelitian*

No.	Variabel	Keterangan Variabel	Definisi
1	$V(G)$	Himpunan titik-titik $\{x_1, x_2, \dots, x_n\}$	Titik-titik yang menyimbolkan tiap lokasi
2	$W_{E(G)}$	Himpunan bobot jarak antar titik $\{d(x_1, x_2), \dots, d(x_m, x_n)\}$	Nilai atau jarak dari tiap titik yang berhubungan.
3	$F_{[i,j]}^{k}$	Himpunan pengujian rute yang berisikan nilai bobot	Nilai entri matriks ketetanggaan berbobot baris ke-i dan kolom ke-j di iterasi ke-k
4	$W_{[i,j]}^{k}$	Himpunan pengujian rute yang berisikan simbol titik lokasi	Nilai entri matriks simbol lokasi titik baris ke-i dan kolom ke-j di iterasi ke-k

3.4 Tahapan Penelitian

Pada penelitian ini dilakukan tahapan-tahapan sebagai berikut:

3.4.1 Peninjauan Langsung

Langkah ini dilakukan dengan mengamati secara langsung objek yang akan digunakan sebagai bahan penelitian. Hal-hal yang ditinjau yaitu titik lokasi *tower* BTS yang akan dijadikan titik graf, rute yang akan digunakan sebagai sisi pada graf dan mengamati apakah rute tersebut bisa dilalui kendaraan roda empat, satu arah atau dua arah.

3.4.2 Pengumpulan Data

Tahap ini merupakan kumpulan data lokasi *tower Base Tranceiver Station* (BTS) yang disusun dalam bentuk tabel dengan menerangkan nama

lokasi dan alamat lokasi. Pengumpulan data ini hanya mengambil rute yang dilewati secara umum atau rute yang melewati jalan raya dikarenakan rute ini memungkinkan untuk dilalui kendaraan roda 4. Selanjutnya dari data titik tersebut diambil jarak dari rute yang akan digunakan sebagai bobot sisi pada graf. Diperoleh data alamat lokasi *tower* BTS sebagai berikut:

Tabel 3.2 *Data Lokasi Tower BTS*

No.	**Nama Lokasi *Tower* BTS**	**Alamat**
1	PT SSB	Millennium Industrial Estate, Jl. Millennium Raya Blok F1 (Tigaraksa, Peusar, Kec. Panongan, Tangerang, Banten 15720
2	PT Citra Akses Indonusa	Jl. Perum Puri Jaya Taman Permai, Sukamantri, Kec. Ps. Kemis, Tangerang, Banten 15560
3	Novotel	Tangcity Superblock, Jl. Jenderal Sudirman No.1, RT.001/RW.005, Babakan, Tangerang, Tangerang City, Banten 15117
4	BMW Astra Serpong	Astra Biz Center No. 11A Jl. BSD Raya Utama BSD City, Lengkong Kulon, Kec. Serpong, Tangerang, Banten 15331
5	Perumahan Dinas Korem	Legok, Kec. Legok, Tangerang, Banten 15820
6	Great Western Resort	Jalan MH. Thamrin Km 2,7, RT.007/RW.001, Panunggangan Utara, Kec. Serpong, Kota Tangerang, Banten 15143
7.	Ronal (Selapajang)	Tangerang, RT.004/RW.003, Selapajang Jaya, Neglasari, Tangerang City, Banten 15127
8.	Ari	Jl. Al Barokah, Pd. Jaya, Kec. Sepatan, Tangerang, Banten 15520
9.	Ronal (Sepatan)	Lebak Wangi, Sepatan Timur, Tangerang, Banten 15520
10.	Deni	Kp. Bolang, Sukasari, Kec. Rajeg, Tangerang, Banten 15540
11.	Villa Balaraja	Saga, Balaraja, Tangerang, Banten

3.4.3 Penentuan Titik Lokasi

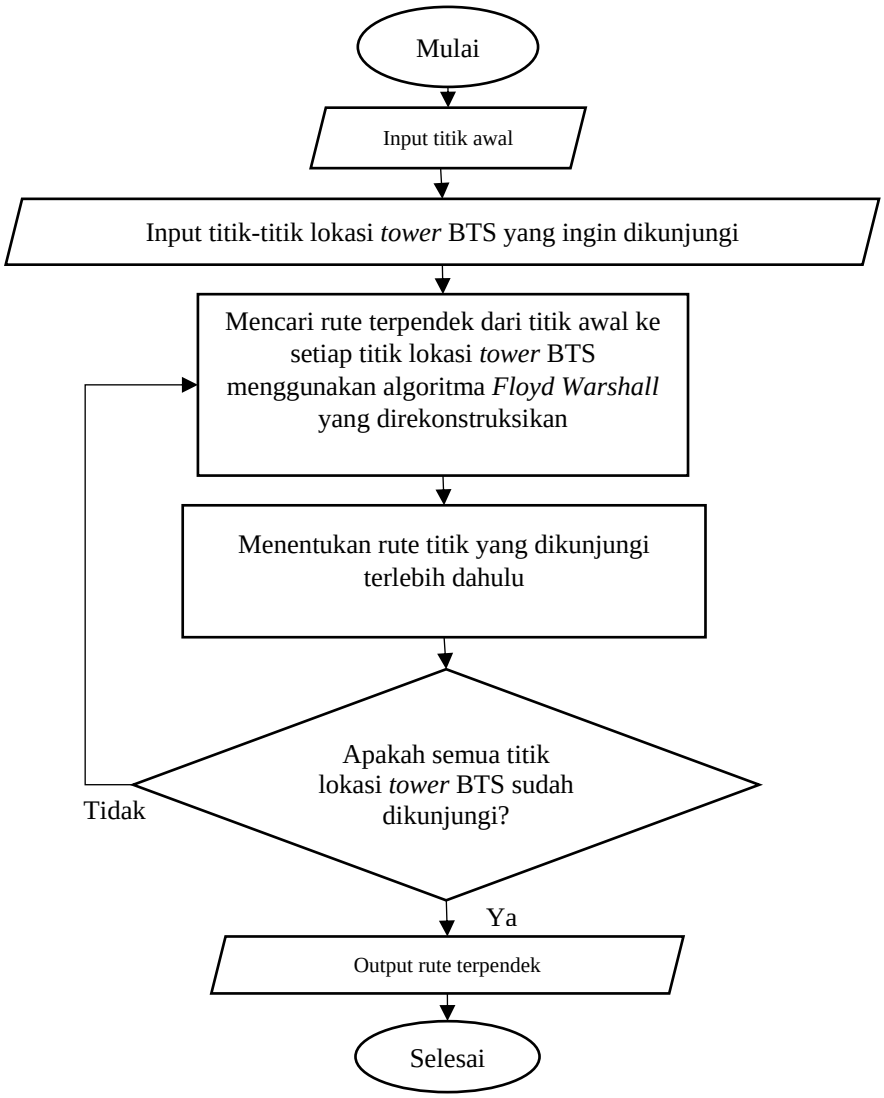
Setiap titik diperoleh berdasarkan titik lokasi koordinat alamat *tower Base Transceiver Station* (BTS) dengan mengambil titik *longitude* dan *latitude* pada *Google Earth Pro*. Kemudian, hasil pengambilan titik tersebut dimuat ke dalam *file* dengan format .txt untuk dapat diolah ke dalam bentuk pencarian rute terpendek menggunakan algoritma *Floyd Warshall* yang dimuat ke dalam aplikasi *Python*.

3.4.4 Membuat Graf

Pembuatan graf lokasi *tower* BTS ini melibatkan penggunaan program *Python* dalam menjalankan algoritma yang digunakan. Pada program ini akan memuat dan menggambarkan data kumpulan titik lokasi *tower* BTS (titik) dan persimpangan jalan yang dilabelkan dan memiliki bobot jarak. Penelitian ini menggunakan satuan jarak Km (kilometer).

3.4.5 Melakukan Pengujian

Langkah selanjutnya setelah proses pembuatan graf adalah melakukan pengujian pencarian rute terpendek menggunakan algoritma *Floyd Warshall*. Namun sebelum melakukan pengujian pencarian rute terpendek, akan dibuat terlebih dahulu sebuah program algoritma *Floyd Warshall* yang disesuaikan dengan data yang telah dimuat ke dalam bentuk graf menggunakan aplikasi *Python*. Pengujian yang dilakukan akan memasukkan titik awal dan beberapa titik tujuan lokasi *tower* BTS, kemudian akan dipilih rute mana yang lebih dekat terlebih dahulu dengan titik awal kemudian dicari titik tujuan berikutnya hingga ditemukan titik tujuan terakhir. Berikut ini merupakan *flowchart* bagaimana proses pencarian rute terpendek menggunakan algoritma *Floyd Warshall*:



Gambar 3.2 *Flowchart Pengujian Pencarian Rute Terpendek*

3.4.6 Evaluasi Hasil

Mengevaluasi hasil pengujian dari rute terpendek menggunakan algoritma *Floyd Warshall* merupakan langkah terakhir pada penelitian ini. Langkah ini akan mengkaji tentang efektivitas hasil rute yang dihasilkan. Faktor yang akan dijadikan keefektifan pencarian rute terpendek ini adalah rute dan jarak yang dilalui ke beberapa titik lokasi *tower* BTS menggunakan algoritma *Floyd Warshall* yang dimodifikasikan kemudian dibandingkan dengan *Google Maps*, jika algoritma bekerja lebih baik maka tingkat keefektifan algoritma tersebut juga baik.

BAB IV

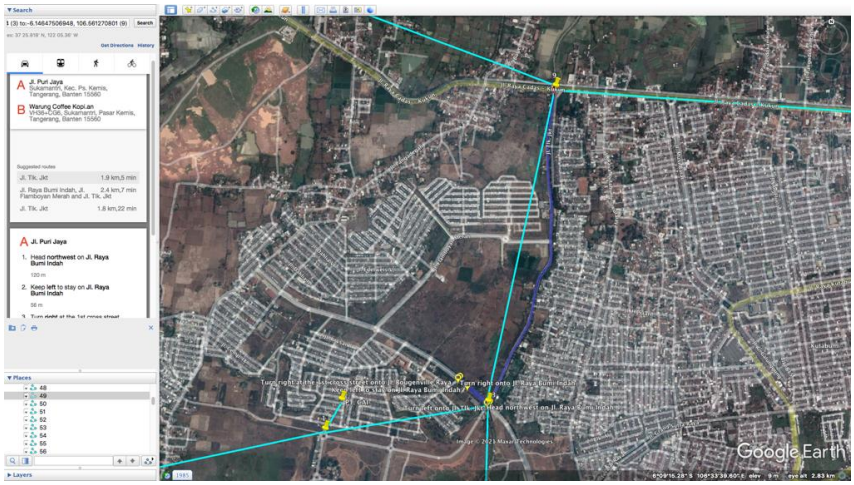
PEMBAHASAN

4.1 Proses Pengumpulan Sumber Data

Pada proses pengumpulan data ini menjelaskan penggunaan aplikasi *Google Earth Pro* sebagai penanda titik dan sisi jalan yang dilalui serta penulisan format data graf berdasarkan data yang dikumpulkan.

4.1.1 Pengambilan Data Dengan Aplikasi *Google Earth Pro*

Pada aplikasi ini diambil data titik yang akan dilewati dan menarik sisi dari titik-titik tersebut sesuai jalan yang dilalui. Penandaan titik pada aplikasi *Google Earth Pro* dilakukan dengan membuat tanda letak pada titik lokasi jalan berupa ikon *pin* yang diberi nama sebagai simbol dari titik tersebut. Kemudian dilakukan pengukuran jarak dari tiap titik yang ada sebagai bobot titik.



Gambar 4.1 *Pengambilan Data dengan Aplikasi Google Earth Pro*

4.1.2 Penulisan Format Data Graf

Bobot dari tiap titik yang dilalui dituliskan ke dalam aplikasi *Notepad* dengan format.txt yang di mana menggunakan spasi sebagai pemisah tiap titik dan bobot. Masukkan bobot tersebut ke dalam data yang sudah ada sesuai dengan titik yang dilalui. Satuan data bobot yang diambil pada aplikasi ini

dengan menggunakan satuan Km (kilometer). Data graf yang diambil pada penelitian ini akan membentuk data graf berarah.

```
11 12 2.2
11 29 0.45
12 11 2.2
12 13 2
12 58 0.2
13 12 2
13 14 1.8
13 91 0.75
14 13 1.8
14 58 1
14 RONAL 0.75
```

Gambar 4.2 *Potongan Data Graf dalam Format File.Txt*

4.2 Deskripsi Data

Penelitian ini menggunakan data titik lokasi yang dikelompokkan menjadi data titik lokasi *tower* BTS dan data titik lokasi persimpangan jalan. Berikut merupakan deskripsi dari masing-masing kelompok data titik lokasi *tower* BTS.

4.2.1 Data Titik Lokasi *Tower* BTS

PT Citra Akses Indonusa memiliki *tower* BTS yang tersebar di daerah Kabupaten dan Kota Tangerang. Lokasi persebaran *tower* BTS yang ada dibentuk ke dalam tabel yang berisikan nama lokasi dan simbol dari *tower* BTS yang dikelompokkan menjadi bagian Kota Tangerang dan Kabupaten Tangerang. Berikut merupakan tabel deskripsi data dari *tower* yang dimiliki oleh PT CAI:

1. **Kota Tangerang**

Tabel 4.1 *Data Titik Lokasi Tower BTS Kota Tangerang*

No.	Nama Lokasi *Tower* BTS	Simbol	*Latitude*	*Longitude*
1.	Novotel	NOVOTEL	6°11'39.72"S	106°38'4.16"E
2.	Great Western Resort	GWR	6°13’19.27”S	106°37’57.34”E
3.	Ronal (Selapanjang)	RS	6° 7’1.14”S	106°37’56.55”E

2. KabupatenTangerang

Tabel 4.2 *Data Titik Lokasi Tower BTS Kabupaten Tangerang*

No.	Nama Lokasi *Tower* BTS	Simbol	*Latitude*	*Longitude*
1.	PT SSB	SSB	6°15'4.28"S	106°29'54.25"E
2.	PT Citra Akses Indonusa	CAI	6° 9'34.86"S	106°33'7.90"E
3.	BMW Astra Serpong	BMW	6°17'13.35"S	106°38'20.15"E
4.	Perumahan Dinas Korem Legok	PDKL	6°16'52.10"S	106°35'39.03"E
5.	Ari	ARI	6° 7’44.99”S	106°35’28.16”E
6	Ronal (Sepatan)	RONAL	6° 7'19.20"S	106°35'54.70"E
7.	Deni	DENI	6° 7'39.01"S	106°33'3.84"E
8.	Villa Balaraja	VB	6°10'21.16"S	106°27'6.66"E

4.2.2 Data Titik Lokasi Persimpangan Jalan

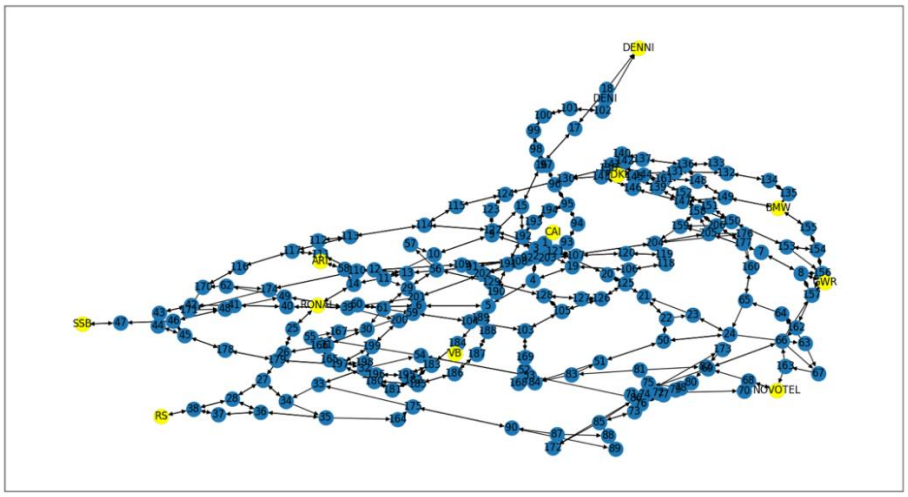
Data titik lokasi persimpangan jalan diambil berdasarkan jenis jalan yang berwarna kuning muda, kuning tua dan abu-abu dengan ukuran ketebalan yang lebih besar seperti pada contoh gambar 4.1. Berdasarkan fungsi dari jalan raya, warna kuning muda memiliki arti jalan kolektor primer dan arteri sekunder. Warna kuning tua memiliki arti jalan arteri primer serta warna abu-abu dengan ketebalan lebih besar memiliki arti jalan kolektor sekunder dan jalan lokal primer. Pemilihan jenis jalan tersebut berlandaskan pada jalan umum atau jalan yang sering dilalui oleh pengguna berkendara roda empat.

Tabel 4.3 *Penulisan Format Data Graf*

Titik 1	Titik 2	Bobot (Km)
11	29	0.45

Pada proses pengumpulan data sebelumnya, didapatkan titik lokasi persimpangan jalan sebanyak 218 titik dan 517 sisi yang akan dimuat ke dalam bentuk data graf berarah seperti pada gambar 4.3. Pada gambar tersebut terdapat

data “11 29 0.45” yang di mana memiliki arti titik persimpangan jalan pertama dengan simbol “11” menuju ke titik persimpangan jalan kedua dengan simbol “29” memiliki bobot sebesar “0.45” Km yang dideskripsikan seperti pada tabel 4.3. Apabila data tersebut divisualisasikan ke dalam bentuk graf, akan memuat gambar sebagai berikut yang menggunakan program *Python* dengan modul *networkx* dan *matplotlib.pyplot*:



Gambar 4.3 *Visualisasi Data Graf*

4.3 Program Pencarian Rute Terpendek

4.3.1 Pembacaan *File* Data

Dalam pembuatan graf, dibutuhkan suatu fungsi pembacaan data untuk mengambil data dari *file* yang ada. Gambar 4.4 menjelaskan fungsi pembacaan data yang mengambil *list* data titik dari nama *file* penyimpanan data yang kemudian dibentuk ke dalam format data untuk membuat graf berarah. Digunakan modul *networkx* sebagai *nx* untuk membantu pembacaan *file* data.

```
g=nx.read_edgelist('nama_file',create_using=nx.DiGraph(),data=[('bobot',float)])
```

Gambar 4.4 *Fungsi Membaca Data Graf*

4.3.2 Pendefinisian Program Algoritma Floyd Warshall

Algoritma *Floyd Warshall* merupakan algoritma yang menggunakan matriks untuk menemukan rute terpendek. Dari data yang sudah terbaca pada tahapan pembacaan *file* data pada program *Python* akan didefinisikan algoritma *Floyd Warshall* dengan memasukkan fungsi graf dan bobot yang diketahui. Selanjutnya dibuat *adjacency matrix* sebagai penyajian data dari graf.

```
for i, j, k in graph.edges(data=True):
    bobot = k.get(weight,1.0)
    jarak[i][j] = min(bobot,jarak[i][j])
    titik[i][j] = i

for w in graph:
    jarak_w = jarak[w]  # simpan perhitungan
    for i in graph:
        jarak_i = jarak[i]  # simpan perhitungan
        for j in graph:
            k = jarak_i[w] + jarak_w[j]
            if jarak_i[j] > k:
                jarak_i[j] = k
                titik[i][j] = titik[w][j]

return dict(titik), dict(jarak)
```

Gambar 4.5 *Potongan Program Algoritma Floyd Warshall*

Dari pendefinisian algoritma tersebut, didapatkan hasil rute terpendek dan bobot dari seluruh rute yang dilalui dari tiap titik. Pada tabel 4.4 akan ditampilkan bentuk matriks yang berisikan bobot rute terpendek dari tiap-tiap lokasi titik *tower*.

Tabel 4.4 *Hasil Bobot Rute Terpendek Tiap-Tiap Lokasi Titik Tower*

TITIK	CAI	ARI	SSB	GWR	BMW	RS	VB	PDKL	DENI	RONAL	NOVOTEL
CAI	0	8.82	16.84	17.6	26.69	15.28	18.18	20.36	7.1	10.5	14.17
ARI	10.12	0	24.95	16.17	26.17	6.58	26.06	21.16	7.14	1.81	12.25
SSB	16.79	25.13	0	21.96	26.95	29.59	14.51	19.8	22.45	26.81	23.43
GWR	16.09	15.44	22.68	0	10.79	13.39	25.74	10.32	19.92	15.27	4.5
BMW	25.43	24.72	26.75	9.74	0	23	31.44	7.05	29.2	24.86	14.09
RS	16.58	6.58	29.22	13.7	23.69	0	31.11	22.71	12.2	4.77	9.78
VB	17.25	25.6	13.98	24.97	31.73	31.11	0	24.58	20.66	26.35	27.85
PDKL	20.57	21.85	20.05	10.22	7.15	21.13	25.36	0	25.22	22.96	12.27
DENI	7.1	7.14	22.5	21.18	31.19	12.2	20.66	24.97	0	7.43	17.27
RONAL	11.80	1.81	26.63	16.63	26.63	4.77	26.35	22.84	7.43	0	12.71
NOVOTEL	15.09	12.5	24.34	4.5	14.5	10.09	28.89	13.62	17	11.96	0

4.3.3 Pendefinisian Program Rekonstruksi Jalur

Program algoritma *Floyd Warshall* diketahui hanya mencari seluruh jalur terpendek dari setiap titik-titik. Dibutuhkan program tambahan untuk mencari pencarian dari satu titik awal menuju satu titik tujuan. Berikut merupakan gambar dari program rekonstruksi jalur sebagai program tambahan pencarian jalur:

```
#Bentuk pencarian rute dengan dimulai titik awal hingga titik tujuan
def rekonstruksi_jalur(t_awal, t_tujuan, predecessors):
    if t_awal == t_tujuan:
        return []
    t_sebelum = predecessors[t_awal]
    t_sekarang = t_sebelum[t_tujuan]
    jalur = [t_tujuan, t_sekarang]
    while t_sekarang != t_awal:
        t_sekarang = t_sebelum[t_sekarang]
        jalur.append(t_sekarang)
    return list(reversed(jalur))
```

Gambar 4.6 *Pendefinisian Program Rekonstruksi Jalur*

Pada gambar dijelaskan bahwa bentuk pencarian jalur dari titik awal hingga titik tujuan dicari berdasarkan mengurutkan hasil rute dari algoritma *Floyd Warshall* yang dimulai dengan titik tujuan hingga titik awalnya. Selanjutnya, hasil pencarian tersebut akan dicerminkan sehingga mendapatkan rute yang dimulai dari titik awal hingga titik tujuan.

4.3.4 Pendefinisian Program Rute Terpendek Terlebih Dahulu

Program ini akan dibuat suatu pencarian rute terpendek dari satu titik awal menuju beberapa titik tujuan. Sesuai dengan nama pendefinisiannya dari beberapa titik tujuan tersebut akan dipilih rute paling terpendeknya diikuti dengan rute terpendek selanjutnya. Gambar 4.7 menampilkan potongan gambar program rute terpendek terlebih dahulu.

```
else :
    for i in titiktujuan:
        lintasan=rekonstruksi_jalur(X,i,semuajalur)
        jarak=semuajarak[X][i]
        penyeleksian.append([lintasan,jarak])

    A1=min(penyeleksian, key=lambda x:x[1] )
    penyeleksian.clear()
    rute.append(A1[0])
    jarak1=A1[1]
    totaljarak+=jarak1
    A2=(A1[0][-1])
    titiktujuan.remove(A2)
    return penyeleksianrtd(A2,titiktujuan,rute,jarak,totaljarak)


penyeleksianrtd(X,titiktujuan,rute,jarak,totaljarak)
```

Gambar 4.7 *Potongan Program Rute Terpendek Terlebih Dahulu*

Pada gambar 4.7 dijelaskan bahwa dalam melakukan pencarian rute terpendeknya berdasarkan pada program pendefinisian rekonstruksi jalur. Dari program rekonstruksi jalur tersebut akan dilakukan pencarian berulang hingga ditemukan titik terakhir dari titik tujuan yang dimasukkan. Kemudian akan dilakukan penjumlahan dari hasil setiap titik yang dituju tersebut untuk mendapatkan total jarak yang ditempuh.

4.4 Pengujian Pencarian Rute Terpendek

Proses pengujian pencarian rute terpendek dilakukan sesuai dengan tahapan penelitian algoritma *Floyd Warshall* yang telah dijelaskan pada bab 3. Berikut merupakan tahapan pengujian pencarian rute terpendek:

4.4.1 Menentukan Titik Awal dan Titik Tujuan

Pada tahapan ini akan dimasukkan titik awal dan titik akhir sebagai tujuan yang akan dibentuk ke dalam rute mana saja yang akan dilalui. Titik awal dan titik tujuan dapat dipilih secara bebas sesuai dengan titik lokasi yang terdapat pada data yang ada. Titik tujuan dapat dipilih dengan memasukkan beberapa titik lokasi *tower* BTS dengan tidak memasukkan titik lebih dari data yang ada. Format penulisan untuk memasukkan beberapa titik tujuan pada program dipisahkan dengan tanda koma dan spasi (,). Contoh format memasukkan titik awal dengan beberapa titik tujuan disajikan pada tabel 4.5.

Tabel 4.5 *Contoh Format Penulisan Memasukkan Titik Awal dan Titik Tujuan*

Titik Awal	CAI
Titik Tujuan	RONAL, ARI, RS

4.4.2 Pengujian Rute Terpendek

Tahapan pengujian rute terpendek ini akan diseleksi titik awal terhadap semua lokasi titik tujuan yang dipilih menggunakan program pencarian rute terpendek terlebih dahulu yang diambil berdasarkan nilai di algoritma *Floyd Warshall*. Pemilihan rute terpendek tersebut dipilih berdasarkan bobot terpendek yang ada kemudian dicari rute yang disesuaikan dengan program rekonstruksi jalur. Berikut merupakan contoh pengujian rute terpendek berdasarkan contoh titik pada tabel 4.5. Pada pengujian penyeleksian rute terdekat terlebih dahulu ini akan dicari rute tiap-tiap titik awal ke dalam daftar *list* tutup. Apabila sudah ditemukan rute terpendeknya, maka akan dimasukkan ke dalam daftar *list* rute.

- **Penentuan Rute CAI → RONAL**

  Berdasarkan hasil dari penyelesaian pencarian rute terpendek menggunakan algoritma *Floyd Warshall* diketahui bahwa bobot dari titik CAI menuju titik RONAL yaitu 10.50 Km. Proses penentuan ini diawali dengan memasukkan titik RONAL ke dalam daftar *list* tutup.

$$List\ tutup\ = (`RONAL')$$

  Dari titik CAI menuju titik RONAL, titik yang dilalui adalah titik 14. Kita masukkan titik 14 tersebut ke dalam *list* tutup.

$$List\ tutup = (`RONAL',`14')$$

Selanjutnya titik yang dituju dari titik CAI menuju titik 14 adalah titik 58, masukkan ke dalam *list*.

$$List\ tutup = (‘RONAL’,’14’,’58’)$$

Kemudian dari titik CAI, titik yang dilalui menuju titik 58 adalah titik 12. Masukkan titik 12 tersebut ke dalam *list* tutup.

$$List\ tutup\ = (‘RONAL’,’14’,’58’,’12’)$$

Titik yang dituju dari titik CAI menuju titik 12 adalah titik 11. Titik 11 kita masukkan ke dalam *list*.

$$List\ tutup\ =\ (‘RONAL’,’14’,’58’,’12’,’11’)$$

Dari titik CAI ke titik 11 titik yang dilalui adalah titik 10. Masukkan titik tersebut ke dalam *list* tutup.

$$List\ tutup\ =\ (‘RONAL’,’14’,’58’,’12’,’11’,’10’)$$

Setelah proses sebelumnya, diketahui titik 9 merupakan titik yang dilalui dari titik CAI ke titik 10. Masukkan titik 9 ke dalam *list*.

$$List\ tutup\ = (‘RONAL’,’14’,’58’,’12’,’11’,’10’,’9’)$$

Kemudian, titik CAI melalui titik 3 untuk menuju ke titik 9. Masukkan titik 41 ke dalam *list* tutup.

$$List\ tutup\ =\ (RONAL’,’14’,’58’,’12’,’11’,’10’,’9’,’3’)$$

Titik yang dituju dari titik CAI menuju titik 3 adalah titik 1, masukkan ke dalam *list*.

$$List\ tutup\ =\ (RONAL’,’14’,’58’,’12’,’11’,’10’,’9’,’3’,’1’)$$

Sama dengan proses sebelumnya, titik yang dilalui untuk menuju titik CAI dari titik 1 adalah langsung titik CAI. Maka telah ditemukan rute terpendeknya dengan memasukkan titik CAI ke dalam *list*.

$$List\ tutup\ = (\text{'}RONAL\text{'}, \text{'}14\text{'}, \text{'}58\text{'}, \text{'}12\text{'}, \text{'}11\text{'}, \text{'}10\text{'}, \text{'}9\text{'}, \text{'}3\text{'}, \text{'}1\text{'}, \text{'}CAI\text{'})$$

Selanjutnya akan dicerminkan *list* tutup tersebut untuk membuat rute dari titik awal menuju titik tujuan sesuai dengan yang dicari. Didapatkan hasil yaitu:

$$List\ tutup\ = (\text{'}CAI\text{'}, \text{'}1\text{'}, \text{'}3\text{'}, \text{'}9\text{'}, \text{'}10\text{'}, \text{'}11\text{'}, \text{'}12\text{'}, \text{'}58\text{'}, \text{'}14\text{'}, \text{'}RONAL\text{'})$$

- **Penentuan Rute CAI → ARI**

Pada hasil dari penyelesaian pencarian rute terpendek menggunakan algoritma *Floyd Warshall* diketahui bahwa bobot dari titik CAI menuju titik ARI yaitu 8.82 Km. Berdasarkan bobot tersebut akan ditelusuri rute mana saja yang dilalui dengan memulai dari titik tujuan sampai titik awal kemudian dicerminkan. Maka kita masukkan titik ARI sebagai titik awal ke dalam *list* tutup.

$$List\ tutup\ = (\text{'}ARI\text{'})$$

Titik yang dituju setelah titik ARI berdasarkan algoritma *Floyd Warshall* yang dimulai dari titik CAI menuju titik ARI adalah titik 58. Kemudian titik 58 dimasukkan ke dalam *list* tutup.

$$List\ tutup = (\text{'}ARI\text{'}, \text{'}58\text{'})$$

Selanjutnya titik yang dituju dengan menelusuri dari titik CAI menuju titik 58 adalah titik 12. Masukkan ke dalam *list*.

$$List\ tutup = (\text{'}ARI\text{'}, \text{'}58\text{'}, \text{'}12\text{'})$$

Titik yang dituju dari titik CAI ke titik 12 adalah titik 11. Titik 11 kita masukkan ke dalam, *list* tutup.

$$List\ tutup = (\text{'}ARI\text{'}, \text{'}58\text{'}, \text{'}12\text{'}, \text{'}11\text{'})$$

Melalui proses yang sama titik yang dilalui dari titik CAI menuju titik 11 adalah titik 10 yang selanjutnya dimasukkan ke dalam *list*.

$$List\ tutup = (\text{'}ARI\text{'}, \text{'}58\text{'}, \text{'}12\text{'}, \text{'}11\text{'}, \text{'}10\text{'})$$

Dari *list* sebelumnya, titik yang dilalui dari titik CAI menuju titik 10 adalah titik 9. Masukkan titik 9 ke dalam *list* tutup.

$$List\ tutup = (\text{'}ARI\text{'}, \text{'}58\text{'}, \text{'}12\text{'}, \text{'}11\text{'}, \text{'}10\text{'}, \text{'}9\text{'})$$

Selanjutnya, titik CAI yang terhubung berdasarkan bobot terpendek dari algoritma *Floyd Warshall* untuk menuju titik 9 adalah titik 3. Masukkan titik 3 ke dalam *list*.

$$List\ tutup = (\text{'}ARI\text{'}, \text{'}58\text{'}, \text{'}12\text{'}, \text{'}11\text{'}, \text{'}10\text{'}, \text{'}9\text{'}, \text{'}3\text{'})$$

Titik yang dilalui untuk menuju titik 3 dari titik CAI adalah titik 1. Masukkan titik 1 ke dalam *list*.

$$List\ tutup = (\text{'}ARI\text{'}, \text{'}58\text{'}, \text{'}12\text{'}, \text{'}11\text{'}, \text{'}10\text{'}, \text{'}9\text{'}, \text{'}3\text{'}, \text{'}1\text{'})$$

Dari titik CAI, titik yang dilalui menuju titik 1 adalah langsung titik CAI itu sendiri. Maka telah ditemukan rute terpendeknya dengan memasukkan titik CAI ke dalam *list*.

$$List\ tutup = (\text{'}ARI\text{'}, \text{'}58\text{'}, \text{'}12\text{'}, \text{'}11\text{'}, \text{'}10\text{'}, \text{'}9\text{'}, \text{'}3\text{'}, \text{'}1\text{'}, \text{'}CAI\text{'})$$

Selanjutnya dari *list* tutup tersebut akan dibuat rute dari titik awal menuju titik tujuan sesuai dengan yang dicari dengan membalikkan *list* tutup tersebut. Didapatkan hasil yaitu:

$$List\ tutup = (\text{'}CAI\text{'}, \text{'}1\text{'}, \text{'}3\text{'}, \text{'}9\text{'}, \text{'}10\text{'}, \text{'}11\text{'}, \text{'}12\text{'}, \text{'}58\text{'}, \text{'}ARI\text{'})$$

- **Penentuan Rute CAI → RS**

Dari hasil penyelesaian pencarian rute terpendek menggunakan algoritma *Floyd Warshall* diketahui bobot dari titik CAI ke titik RS adalah 15.28 Km.

Kemudian akan ditelusuri rute mana saja yang dilalui dengan memulai dari titik tujuan sampai titik awal kemudian dicerminkan dari bobot yang diketahui tersebut. Kita masukkan titik RS sebagai titik awal ke dalam *list* tutup.

$$List\ tutup\ = (\text{'}RS\text{'})$$

Titik yang dilalui untuk menuju titik RS dari titik CAI adalah titik 38. Masukkan titik 38 tersebut ke dalam *list* tutup.

$$List\ tutup\ = (\text{'}RS\text{'},\text{'}38\text{'})$$

Sama dengan proses penentuan rute sebelumnya, dilakukan pencarian rute hingga ditemukan titik CAI sebagai titik terakhir dalam *list*. Maka ditemukan rute terpendeknya dengan memasukkan titik CAI ke dalam *list* tutup.

$$List\ tutup$$
$$= (\text{'}RS\text{'},\text{'}38\text{'},\text{'}28\text{'},\text{'}27\text{'},\text{'}26\text{'},\text{'}25\text{'},\text{'}RONAL\text{'},\text{'}14\text{'},\text{'}58\text{'},\text{'}12\text{'},\text{'}11\text{'},\text{'}10\text{'},\text{'}9\text{'},\text{'}3\text{'},\text{'}1\text{'},\text{'}CAI\text{'})$$

Kemudian untuk membuat rute yang tersusun dari titik awal menuju titik tujuan adalah dengan mencerminkan *list* tutup tersebut. Didapatkan hasil:

$$List\ tutup$$
$$= (\text{'}CAI\text{'},\text{'}1\text{'},\text{'}3\text{'},\text{'}9\text{'},\text{'}10\text{'},\text{'}11\text{'},\text{'}12\text{'},\text{'}58\text{'},\text{'}14\text{'},\text{'}RONAL\text{'},\text{'}25\text{'},\text{'}26\text{'},\text{'}27\text{'},\text{'}28\text{'},\text{'}38\text{'},\text{'}RS\text{'})$$

Berdasarkan penentuan rute di atas yang dimulai dari titik awal CAI menuju masing-masing titik tujuan didapatkan titik lokasi dengan rute terdekat untuk dikunjungi adalah titik ARI. Kemudian dapat kita masukkan titik ARI ke dalam *list* rute.

$$List\ rute = (\text{'}CAI\text{'},\text{'}1\text{'},\text{'}3\text{'},\text{'}9\text{'},\text{'}10\text{'},\text{'}11\text{'},\text{'}12\text{'},\text{'}58\text{'},\text{'}ARI\text{'})$$

- **Penentuan ARI→ RONAL**

Diketahui bobot dari titik ARI ke titik RONAL adalah 1.81 Km. Kita masukkan titik RONAL sebagai titik awal ke dalam *list* tutup.

$$List\ tutup\ = (\text{'}RONAL\text{'})$$

Selanjutnya titik yang dituju dari titik ARI menuju titik RONAL adalah titik 14, masukkan ke dalam *list*.

$$List\ tutup\ = (\text{'}RONAL\text{'},\text{'}14\text{'})$$

Titik yang dituju setelah titik 14 yang dimulai dari titik ARI menuju titik 14 adalah titik 58. Kemudian titik 58 dimasukkan ke dalam *list* tutup.

$$List\ tutup\ = (\text{'}RONAL\text{'},\text{'}14\text{'},\text{'}58\text{'})$$

Kemudian, titik ARI merupakan titik yang dilalui dari titik ARI ke titik 58 yang berarti telah ditemukan rute terpendeknya. Masukkan titik ARI ke dalam *list*.

$$List\ tutup\ = (\text{'}RONAL\text{'},\text{'}14\text{'},\text{'}58\text{'},\text{'}ARI\text{'})$$

Setelah memasukkan titik terakhir ke dalam *list* tutup, akan dicerminkan *list* tersebut agar menjadi suatu bentuk hasil pencarian yang dimulai dari titik awal hingga titik tujuan.

$$List\ tutup\ = (\text{'}ARI\text{'},\text{'}58\text{'},\text{'}14\text{'},\text{'}RONAL\text{'})$$

- **Penentuan ARI → RS**

Dari hasil algoritma *Floyd Warshall*, bobot titik ARI menuju titik RS diketahui 6.58 Km. Kemudian masukkan titik RS sebagai titik awal ke dalam *list*.

$$List\ tutup\ = (\text{'}RS\text{'})$$

Selanjutnya akan dicari titik dengan bobot minimum yang menghubungkan titik ARI ke titik RS. Didapatkan titik 38, masukkan ke dalam *list* tutup.

$$List\ tutup\ = (\text{'}RS\text{'},\text{'}38\text{'})$$

Titik yang dituju dari titik ARI menuju titik 38 adalah titik 28. Titik 28 kita masukkan ke dalam *list*.

$$List\ tutup\ =\ (\text{'}RS\text{'},\text{'}38\text{'},\text{'}28\text{'})$$

Dari *list* sebelumnya, titik yang dilalui dari titik ARI menuju titik 28 adalah titik 27. Masukkan titik 27 ke dalam *list* tutup.

$$List\ tutup\ =\ (\text{'}RS\text{'},\text{'}38\text{'},\text{'}28\text{'},\text{'}27\text{'})$$

Melalui proses yang sama titik yang dilalui dari titik ARI menuju titik 27 adalah titik 26 yang selanjutnya dimasukkan ke dalam *list*.

$$List\ tutup\ =\ (\text{'}RS\text{'},\text{'}38\text{'},\text{'}28\text{'},\text{'}27\text{'},\text{'}26\text{'})$$

Kemudian, titik ARI melalui titik 25 untuk menuju ke titik 26. Masukkan titik 26 ke dalam *list* tutup.

$$List\ tutup\ =\ (\text{'}RS\text{'},\text{'}38\text{'},\text{'}28\text{'},\text{'}27\text{'},\text{'}26\text{'},\text{'}25\text{'})$$

Didapatkan titik RONAL yang merupakan titik yang dilalui dari titik ARI menuju titik 25.

$$List\ tutup\ =\ (\text{'}RS\text{'},\text{'}38\text{'},\text{'}28\text{'},\text{'}27\text{'},\text{'}26\text{'},\text{'}25\text{'},\text{'}RONAL\text{'})$$

Dari titik ARI, titik yang melalui titik tujuan RONAL adalah titik 14. Masukkan ke dalam *list*.

$$List\ tutup\ =\ (\text{'}RS\text{'},\text{'}38\text{'},\text{'}28\text{'},\text{'}27\text{'},\text{'}26\text{'},\text{'}25\text{'},\text{'}RONAL\text{'},\text{'}14\text{'})$$

Selanjutnya titik yang dilalui dari titik ARI ke titik 14 adalah titik 58.

$$List\ tutup\ =\ (\text{'}RS\text{'},\text{'}38\text{'},\text{'}28\text{'},\text{'}27\text{'},\text{'}26\text{'},\text{'}25\text{'},\text{'}RONAL\text{'},\text{'}14\text{'},\text{'}58\text{'})$$

Diketahui titik ARI merupakan titik terakhir yang dimasukkan ke dalam *list*, karena titik yang dilalui titik ARI menuju titik 58 adalah titiknya sendiri. Didapatkan hasil:

$$List\ tutup\ =\ (\text{'}RS\text{'},\text{'}38\text{'},\text{'}28\text{'},\text{'}27\text{'},\text{'}26\text{'},\text{'}25\text{'},\text{'}RONAL\text{'},\text{'}14\text{'},\text{'}58\text{'},\text{'}ARI\text{'})$$

Setelah didapatkan hasil *list* tutup sebelumnya, kita cerminkan agar membentuk suatu pencarian yang urut dimulai dari titik awal hingga titik tujuan.

$$List\ tutup\ =\ (\text{'}ARI\text{'},\text{'}58\text{'},\text{'}14\text{'},\text{'}RONAL\text{'},\text{'}25\text{'},\text{'}26\text{'},\text{'}27\text{'},\text{'}28\text{'},\text{'}38\text{'},\text{'}RS\text{'})$$

Berdasarkan penentuan rute di atas yang dimulai dengan titik awal ARI menuju masing-masing titik tujuan yang tersisa, didapatkan titik lokasi dengan rute terdekat untuk dikunjungi adalah titik RONAL. Kemudian dapat kita masukkan *list* tutup dari titik ARI menuju titik RONAL ke dalam *list* rute.

$$List\ rute = (\text{'}CAI\text{'},\text{'}1\text{'},\text{'}3\text{'},\text{'}9\text{'},\text{'}10\text{'},\text{'}11\text{'},\text{'}12\text{'},\text{'}58\text{'},\text{'}ARI\text{'},\text{'}58\text{'},\text{'}14\text{'},\text{'}RONAL\text{'})$$

- **Penentuan RONAL → RS**

Diketahui bobot sebesar 4.77 Km merupakan bobot dari titik RONAL ke titik RS. Proses penentuan ini sama halnya dengan proses yang dilakukan sebelumnya. Kita masukkan titik RS ke dalam *list*.

$$List\ tutup\ =\ (\text{'}RS\text{'})$$

Sama dengan proses sebelumnya, dilakukan pencarian rute hingga ditemukan titik RONAL sebagai titik terakhir di dalam *list*. Didapatkan hasil:

$$List\ tutup\ =\ (\text{'}RS\text{'},\text{'}38\text{'},\text{'}28\text{'},\text{'}27\text{'},\text{'}26\text{'},\text{'}25\text{'},\text{'}RONAL\text{'})$$

Kita cerminkan hasil dari pencarian rute dari titik RONAL ke titik RS supaya menjadi suatu bentuk urut dari titik awal hingga titik tujuan.

$$List\ tutup\ =\ (\text{'}RONAL\text{'},\text{'}25\text{'},\text{'}26\text{'},\text{'}27\text{'},\text{'}28\text{'},\text{'}38\text{'},\text{'}RS\text{'})$$

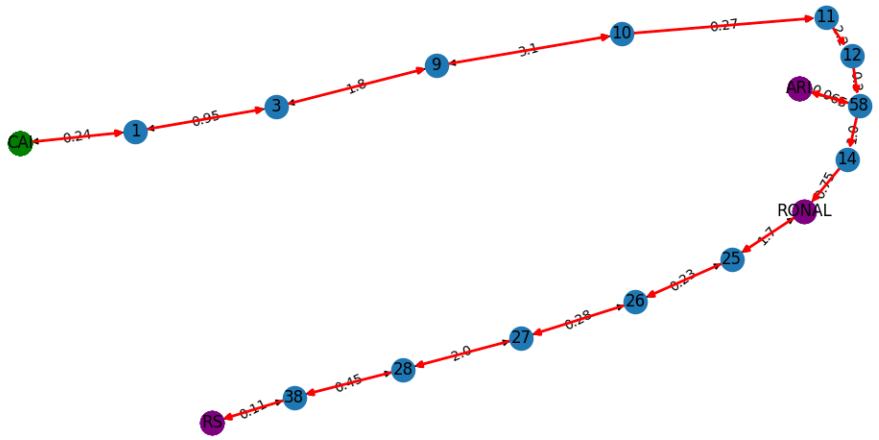
Dari seluruh penentuan rute yang telah dilakukan, didapatkan hasil *list* rute yang di mana telah dilalui tiap-tiap titik tujuan yang dimasukkan:

$$\boldsymbol{List\ rute} = (\text{'}CAI\text{'},\text{'}1\text{'},\text{'}3\text{'},\text{'}9\text{'},\text{'}10\text{'},\text{'}11\text{'},\text{'}12\text{'},\text{'}58\text{'},\text{'}ARI\text{'},\text{'}58\text{'},\text{'}14\text{'},\text{'}RONAL\text{'}$$
$$,\text{'}25\text{'},\text{'}26\text{'},\text{'}27\text{'},\text{'}28\text{'},\text{'}38\text{'},\text{'}RS\text{'})$$

List rute di atas menjelaskan bahwa rute yang dilalui pertama kali merupakan rute yang memiliki bobot yang paling terkecil dan dilanjutkan hingga semua titik tujuan sudah dilalui. Titik tujuan yang dituju pertama kali dari titik awal CAI adalah titik ARI. Kemudian dari titik ARI dilanjutkan menuju titik RONAL dan diakhiri dengan titik RS. Total jarak dari seluruh rute yang dilalui adalah 15.40 Km. Apabila titik tujuan tersebut dimasukkan secara acak, akan tetap menghasilkan hasil yang sama.

4.4.3 Visualisasi Rute

Hasil dari pencarian rute dari titik awal CAI hingga titik tujuan akhir RS di visualisasikan dan ditampilkan pada gambar 4.8.

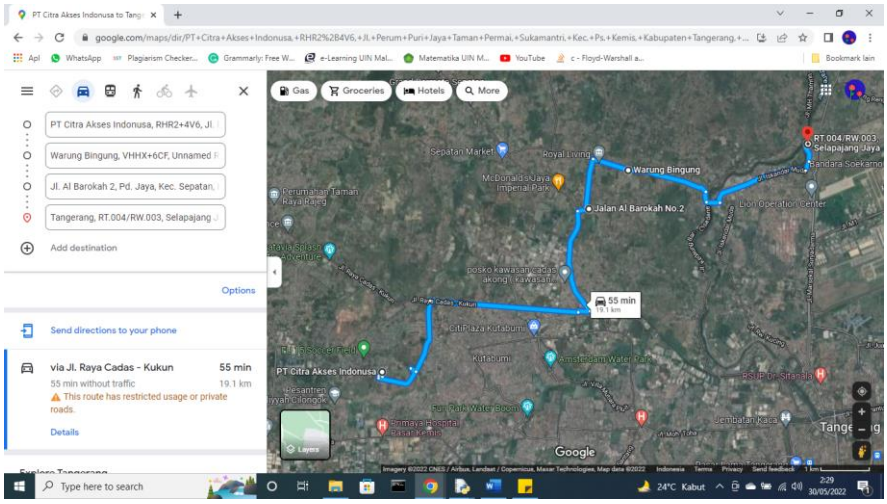


Gambar 4.8 *Visualisasi Hasil Pengujian Rute*

Pada gambar ditampilkan titik CAI yang berwarna hijau yang memiliki arti sebagai titik awal. Titik yang berwarna ungu memiliki arti titik tujuan dan titik yang berwarna biru merupakan warna dari seluruh titik. Sisi panah berwarna merah menunjukkan graf berarah dari sisi rute yang dituju. Terdapat juga sisi panah berwarna hitam yang menunjukkan seluruh sisi dari tiap-tiap titik.

4.5 Hasil Pengujian

Penelitian ini menguji rute terpendek pada beberapa lokasi *tower* BTS menggunakan algoritma *Floyd Warshall* yang akan dikaji dengan *Google Maps*. Pada hasil pengujian pencarian rute sebelumnya didapatkan hasil rute yang diawali dengan CAI menuju ARI, RONAL dan RS. Pada *Google Maps* akan dilakukan pencarian rute yang sama sesuai urutan titik tujuan yang dimasukkan pada tabel 4.5.



Gambar 4.9 *Hasil Pencarian Rute Google Maps*

Gambar hasil pencarian rute pada *Google Maps* menampilkan rute yang berurutan sesuai dengan titik tujuan yang dimasukkan. Didapatkan total jarak 19.1 Km yang lebih panjang 3.37 Km dibandingkan hasil pencarian menggunakan algoritma *Floyd Warshall*.

Proses pengujian ini akan dilakukan dengan acak secara menyeluruh sebanyak 30 kali. Jadi, seluruh titik lokasi *tower* BTS akan mendapatkan bagian untuk dilakukan proses pengujian sesuai dengan program yang telah dibuat sebelumnya. Akan dibandingkan hasil dari titik awal dan urutan titik tujuan yang dimasukkan menggunakan algoritma *Floyd Warshall* dan *Google Maps*. Hasil proses pengujian rute terpendek ini akan ditampilkan pada tabel 4.6.

Tabel 4.6 *Hasil Pengujian Rute Bagian 1 (1-15)*

No.	Titik Awal	Titik Tujuan	Hasil Rute Penelitian	Hasil Jarak Penelitian (Km)	Rute *Google Maps*	Jarak *Google Maps* (Km)
1	SSB	ARI GWR	SSB, GWR, ARI	37.40	SSB, ARI, GWR	41.8
2	NOVOTEL	RONAL, RS	NOVOTEL, RS, RONAL	14.86	NOVOTEL, RS, RONAL	16.5
3	VB	CAI, GWR	VB, CAI, GWR	34.86	VB, CAI, GWR	41.6
4	RONAL	BMW, CAI	RONAL, CAI, BMW	38.4	RONAL, BMW, CAI	53
5	DENI	NOVOTEL, CAI	DENI, CAI, NOVOTEL	21.27	DENI, NOVOTEL, CAI	32.8
6	CAI	SSB, PDKL	CAI, SSB, PDKL	36.64	CAI, SSB, PDKL	38.4
7	BMW	NOVOTEL, ARI	BMW, NOVOTEL, ARI	26.62	BMW, NOVOTEL, ARI	27.1
8	RS	BMW, VB	RS, BMW, VB	55.14	RS, BMW, VB	62.6
9	PDKL	VB, DENI	PDKL, DENI, VB	45.89	PDKL, VB, DENI	52.1
10	VB	RONAL, DENI	VB, DENI, RONAL	28.09	VB, RONAL, DENI	34.5
11	ARI	PDKL, CAI	ARI, CAI, PDKL	30.38	ARI, PDKL, CAI	43.3
12	GWR	DENI, SSB	GWR, DENI, SSB	42.43	GWR, DENI, SSB	45.3
13	RONAL	GWR, RS	RONAL, RS, GWR	18.47	RONAL, GWR, RS	30.3
14	CAI	RONAL, ARI, RS	CAI, ARI, RONAL, RS	15.40	CAI, RONAL, ARI, RS	19.1
15	SSB	NOVOTEL, RONAL, VB	SSB, VB, RONAL, NOVOTEL	53.58	SSB, NOVOTEL, RONAL, VB	63.6

Tabel 4.7 *Hasil Pengujian Rute Bagian 2 (4-18)*

No.	Titik Awal	Titik Tujuan	Hasil Rute Penelitian	Hasil Jarak Penelitian (Km)	Rute *Google Maps*	Jarak *Google Maps* (Km)
16	VB	RS, GWR, DENI	VB, DENI, RS, GWR	46.56	VB, RS, GWR, DENI	79.9
17	DENI	PDKL, ARI, CAI	DENI, CAI, ARI, PDKL	37.09	DENI, PDKL, ARI, CAI	63.9
18	RONAL	VB, PDKL, GWR	RONAL, GWR, PDKL, VB	52.32	RONAL, VB, PDKL, GWR	68.4
19	RS	PDKL, DENI, BMW	RS, DENI, PDKL, BMW	44.32	RS, PDKL, DENI, BMW	86.1
20	ARI	NOVOTEL, SSB, CAI	ARI, CAI, NOVOTEL, SSB	48.63	ARI, NOVOTEL, SSB, CAI	60.2
21	NOVOTEL	BMW, CAI, GWR	NOVOTEL, GWR, BMW, CAI	40.66	NOVOTEL, BMW, CAI, GWR	62.4
22	GWR	CAI, RS, SSB	GWR, RS, CAI, SSB	46.81	GWR, CAI, RS, SSB	61.4
23	BMW	DENI, SSB, ARI	BMW, ARI, DENI, SSB	54.37	BMW, DENI, SSB, ARI	81.3
24	PDKL	ARI, NOVOTEL, VB	PDKL, NOVOTEL, ARI, VB	50.86	PDKL, ARI, NOVOTEL, VB	68.8
25	ARI	DENI, RONAL, RS	ARI, RONAL, RS, DENI	18.78	ARI, DENI, RONAL, RS	19.5
26	NOVOTEL	BMW, GWR, PDKL	NOVOTEL, GWR, PDKL, BMW	21.97	NOVOTEL, BMW, GWR, PDKL	35.9
27	PDKL	CAI, VB, BMW, RONAL	PDKL, BMW, RONAL, CAI, VB	62.01	PDKL, CAI, VB, BMW, RONAL	104

Tabel 4.8 *Hasil Pengujian Rute Bagian 3(28-30)*

No.	Titik Awal	Titik Tujuan	Hasil Rute Penelitian	Hasil Jarak Penelitian (Km)	Rute *Google Maps*	Jarak *Google Maps* (Km)
28	BMW	DENI, ARI, VB, NOVOTEL	BMW, NOVOTEL, ARI, DENI, VB	54.42	BMW, DENI, ARI, VB, NOVOTEL	106
29	SSB	PDKL, RS, BMW, ARI	SSB, PDKL, BMW, RS, ARI	56.53	SSB, PDKL, RS, BMW, ARI	96.2
30	CAI	NOVOTEL, SSB, DENI, GWR	CAI, DENI, NOVOTEL, GWR, SSB	51.56	CAI, NOVOTEL, SSB, DENI, GWR	89.1

4.6 Evaluasi Pengujian

Setelah melakukan pengujian sebanyak 30 kali didapatkan hasil bahwa pengujian menggunakan algoritma *Floyd Warshall* memiliki hasil jarak yang lebih pendek daripada jarak yang *Google Maps*. Efektivitas hasil dari jarak tempuh akan dimuat ke dalam bentuk persentase menggunakan rumus (Mukti, 2018):

$$Nilai\ efektivitas = \frac{JG - JF}{JG} \times 100\%. \qquad (3)$$

Keterangan:

JG = jarak rute pada *Google Maps*

JF = jarak rute terpendek algoritma *Floyd Warshall*

Hasil dari perhitungan nilai efektivitas ditampilkan pada tabel 4.9.

Tabel 4.9 *Hasil Perhitungan Nilai Efektivitas*

Pengujian Ke-	**Persentase (%)**	**Pengujian Ke-**	**Persentase (%)**
1	10,53	**16**	41,73
2	9,94	**17**	41,96
3	16,20	**18**	23,51
4	27,55	**19**	48,52
5	35,15	**20**	19,22
6	4,58	**21**	34,84
7	1,77	**22**	23,76
8	11,92	**23**	33,12
9	11,92	**24**	26,08
10	18,58	**25**	3,69
11	29,84	**26**	38,80
12	6,34	**27**	40,38
13	39,04	**28**	48,66
14	19,37	**29**	41,24
15	15,75	**30**	42,13
Rata-rata			**25,54**

Tabel 4.9 menunjukkan bahwa pengujian dengan algoritma *Floyd Warshall* memiliki jarak tempuh dengan rata-rata persentase keefektifan sebesar 25,54 % dibandingkan dengan *Google Maps*. Persentase keefektifan terbesar terjadi pada pengujian ke-28 sebesar 48,66%. Persentase keefektifan terkecil terjadi pada pengujian ke-7 sebesar 1,77%. Hasil persentase tersebut menyatakan bahwa algoritma *Floyd Warshall* merupakan algoritma yang efektif dikarenakan memiliki jarak yang lebih pendek dibandingkan dengan pengujian menggunakan *Google Maps*. Jika nilai efektivitas bernilai positif, maka nilai rute terpendek tersebut bersifat efektif. Jika bernilai negatif, maka nilai rute terpendek tersebut bersifat tidak efektif.

Alasan yang membuat hasil pengujian rute terpendek tersebut menjadi efektif adalah penyeleksian rute lokasi *tower* BTS yang dikunjungi di prioritaskan ke rute yang lebih dekat jaraknya terlebih dahulu dari seluruh tujuan yang dipilih. Apabila dimasukkan dua atau lebih titik lokasi *tower* BTS yang akan dituju, maka

akan diseleksi setiap titik tujuan tersebut menggunakan program pencarian rute terpendek terlebih dahulu yang disesuaikan dengan algoritma *Floyd Warshall* yang direkonstruksikan. Selanjutnya akan dipilih titik lokasi *tower* BTS mana yang akan dijadikan tujuan pertama dan berlanjut hingga ditemukan titik tujuan terakhir.

Dari proses pendefinisian program tersebut didapatkan pengujian rute yang lebih efektif dibandingkan dengan pengujian rute menggunakan *Google Maps*. Pengujian yang dilakukan menggunakan *Google Maps* tidak menyeleksi titik tujuan mana yang harus dituju terlebih dahulu. Apabila kita memasukkan titik tujuan lokasi *tower* BTS PT SSB, NOVOTEL dan RONAL, *Google Maps* akan menampilkan titik tujuan yang harus dituju pertama kali adalah titik lokasi *tower* BTS PT SSB, NOVOTEL dan diakhiri dengan titik RONAL. Jika menggunakan program pencarian rute terpendek terlebih dahulu tersebut akan dipilih titik tujuan terpendek dari titik awal kemudian dilanjutkan dengan titik terpendek selanjutnya hingga titik terpendek tujuan terakhir. Sehingga dapat dikatakan bahwa bentuk program ini memiliki peran dalam menjadikan pencarian rute terpendek ini efektif.

4.7 Kajian Keislaman Terhadap Pencarian Rute Terpendek

Pencarian rute terpendek merupakan suatu pencarian yang bertujuan untuk melakukan kegiatan berhemat. Hemat berarti hati-hati dalam membelanjakan uang, tidak boros, disesuaikan dengan pendapatan dan kemampuan, maka dari itu kita diharuskan untuk tidak melakukan pemborosan. Sebagaimana yang telah dikaji pada bagian 2.5, ayat tersebut menjelaskan tentang larangan bagi umat muslim melakukan pemborosan, karena pemborosan

merupakan perbuatan yang dilarang oleh Allah Swt. yang di mana perbuatan tersebut tidak membawa manfaat.

Selain melakukan kegiatan berhemat tersebut kita juga dituntut untuk tidak terlalu pelit terhadap apa yang kita lakukan maka kita perlu melakukannya dengan cukup, yang berarti tidak kurang dan tidak lebih. Seperti pada hadis riwayat Tirmidzi, Ibnu Majah dan Ahmad, Rasulullah saw. bersabda:

"*Tidaklah anak Adam memenuhi wadah yang lebih buruk dari perut. Cukuplah bagi anak Adam memakan beberapa suapan untuk menegakkan punggungnya. Namun jika ia harus (melebihinya), hendaknya sepertiga perutnya (diisi) untuk makanan, sepertiga untuk minuman, dan sepertiga lagi untuk bernafas*".

Berdasarkan hadis tersebut memiliki kaitan dengan penelitian ini yaitu jika menempuh suatu perjalanan, harap senantiasa mencari rute terpendek agar menghemat jarak dan juga bahan bakar, yang juga hal tersebut dilakukan dengan cukup, tidak pelit dan tidak boros.

BAB V

PENUTUP

5.1 Kesimpulan

Berdasarkan pembahasan penelitian yang telah dilakukan dapat diambil kesimpulan bahwa algoritma *Floyd Warshall* membutuhkan program tambahan untuk mencari rute dari titik awal ke titik tujuan yang diinginkan. Pencarian rute terpendek lokasi *tower* BTS PT Citra Akses Indonusa di Kabupaten dan Kota Tangerang menggunakan algoritma *Floyd Warshall* memiliki hasil rute terpendek dengan hasil yang lebih efektif. Hal ini ditunjukkan dari hasil pengujian pencarian rute secara acak sebanyak 30 kali terhadap titik-titik lokasi *tower* BTS. Didapatkan rata-rata efektivitas jarak rute terpendek sebesar 25.54% dibandingkan jarak rute yang dihasilkan *Google Maps*. Hasil efektivitas tertinggi diperoleh pengujian ke-28 sebesar 48,66%. Persentase keefektifan terkecil terjadi karena memiliki hasil rute yang sama dengan *Google Maps,* yaitu pada pengujian ke-7 sebesar 1,77% yang berarti pencarian menggunakan algoritma *Floyd Warshall* memiliki hasil yang lebih efektif.

Keefektifan dari hasil pencarian rute terpendek pada penelitian ini terjadi karena adanya program pencarian rute terpendek terlebih dahulu. Tujuan program tersebut adalah untuk menentukan titik tujuan lokasi *tower* BTS yang memiliki jarak terdekat dengan titik awal untuk dikunjungi terlebih dahulu dari beberapa titik tujuan yang dipilih. Program pencarian rute terpendek terlebih dahulu ini menggunakan algoritma *Floyd Warshall* yang sudah dilakukan rekonstruksi jalur dalam pencariannya. Hasil penelitian ini membuat pencarian rute terpendek lebih efektif dibandingkan pencarian menggunakan *Google Maps,* karena pencarian

pada *Google Maps* tidak melakukan penyeleksian pencarian rute terpendek terlebih dahulu dan hanya menampilkan rute sesuai dengan urutan titik tujuan yang dimasukkan.

5.2 Saran

Penelitian ini menghasilkan program modifikasi tambahan dalam pencarian rute terpendek lokasi *tower* BTS pada PT Citra Akses Indonusa menggunakan algoritma Floyd Warshall. Penelitian selanjutnya diharapkan untuk menjadikan penelitian ini sebagai referensi dalam melakukan pencarian rute terpendek dan mengembangkan tampilan visualisasi grafnya agar terlihat lebih rapi dan sesuai dengan tampilan gambar peta pada *Google Maps*.

DAFTAR PUSTAKA

Abdullah. 2003. *Tafsir Ibnu Katsir Jilid 5*. Bogor: Pustaka Imam asy-Syafi'i.

Al-Quran Terjemahan. 2015. *Departemen Agama RI*. Bandung: CV Darus Sunnah.

Aldina, letivany. 2018. *Contoh Penerapan Program Dinamis dalam Algoritma Floyd-Warshall*. Makalah: Sekolah Teknik Informatika Institut Teknologi Bandung.

Aronoff, S. 1989. *Geographic Information System: A Management Perspective*. Canada, Ottawa: WDL Publication.

Awangga, Rolly Maulana. 2019. *Pengantar Sistem Informasi Geografis: Sejarah, Definisi dan Konsep Dasar*. Bandung: Kreatif Industri Nusantara.

Chartrand, L dan Lesniak, L. 2000. *Graph & Digraph*. California: Chapman and Hall.

HR At-Tirmidzi (2380), Ibnu Majah (3349), Ahmad (4/132), dan lain-lain. Hadits ini dinilai shahiholeh Al-Albani dalam As-Silsilah Ash-Shahihah (2265).

Hamka. 1982. *Tafsir al-Azhar: juz. XIV*. Jakarta: PT Pustaka Panjimas,.

Isnaini, Dewi. 2019. Pencarian Rute Terpendek Non-Player Character (NPC) dengan Metode Floyd Warshall pada Game Wisata Batu. Skripsi. Malang: Universitas Islam Negeri Malang.

K. Surendro. 2007. Pemanfaatan Enterprise Architecture Planning Untuk Perencanaan Strategis Sistem informasi. *Jurnal Informatika*, 08(01),1–9.

Laila Nugraha, A., & Sudarsono, B. (2012). Survei Topografi Untuk Menentukan Garis Tampak Pandang Base Transceiver Station (BTS). *Teknik*, 28(1), 55-60.

Mukti, M Ridwan dan Mulyono. 2018. Menentukan Rute Terpendek Menggunakan Algoritma Floyd Warshall Dalam Pendistribusian Barang Pada PT. Rapy Ray Putratama. *Karismatika*, 04(01),39-53.

Nanang Ismail, Maharoni & Innel Lindra. 2015. Analisis Perencanaan Pembangunan BTS (Base Transceiver Station) Berdasarkan Faktor Kelengkungan Bumi dan Daerah Fresnel di Regional Project Sumatera Bagian Selatan. *Jurnal Teknik Elektro*, 09(01),104-121.

Novandi, R. A. D. 2007. Perbandingan Algoritma Dijkstra dan Algoritma FloydWarshall dalam Penentuan Sisi Terpendek (Single Shortest Path). Bandung: Institut Teknologi Bandung.

Nugroho, Feri. 2020. *Sistem Informasi Geografis Membuat Peta dengan Citra Satelit di ArcGIS 10.8*. Bandung: Media Sains Indonesia.

Rully Mujiastuti, Rizky Purwanto & Sugiartowo. 2018. Pencarian Lokasi Apotek Terdekat Menggunakan Algoritma Floyd-Warshal. Jurnal Sistem Informasi, Teknologi Informatika dan Komputer. 09(01):55-63.

Saputra, Ragil. 2011. Sistem Geografis Pencarian Rute Optimum Obyek Wisata Kota Yogyakarta Dengan Algoritma Floyd-Warshall. *Jurnal Matematika*, 14(01),19-24.

Sulistiyanto. 2021. *Sistem Informasi Geografis Teori dan Praktik dengan Quantum GIS*. Malang: Ahlimedia Press.

Vera Apriliani Nawagusti. 2018. Penerapan Algoritma Floyd Warshall dalam Aplikasi Penentuan Rute Terpendek Mencari Lokasi BTS (Base Tower Station) pada PT.GCI Palembang. *Jurnal Nasional Teknologi dan Sistem Informasi*,04(02),081-088.

LAMPIRAN

Lampiran 1: Data Titik dan Bobot (1-64)

No.	Titik 1	Titik 2	Bobot	No.	Titik 1	Titik 2	Bobot
1	CAI	1	0.24	**33**	5	188	1.9
2	SSB	47	2	**34**	6	5	2.5
3	NOVOTEL	68	2.9	**35**	6	39	0.45
4	GWR	8	1.24	**36**	6	201	0.7
5	GWR	156	0.5	**37**	7	8	0.6
6	GWR	157	0.7	**38**	7	205	1.3
7	BMW	135	2.2	**39**	8	7	1.14
8	BMW	149	2.8	**40**	8	156	0.5
9	BMW	155	3.3	**41**	8	157	0.6
10	PDKL	130	1.5	**42**	8	GWR	0.87
11	PDKL	131	0.7	**43**	9	3	1.8
12	DENI	18	0.16	**44**	9	10	3.1
13	DENI	102	0.05	**45**	9	15	2.3
14	ARI	58	0.1	**46**	10	9	3.1
15	RONAL	14	0.75	**47**	10	11	0.27
16	RONAL	25	1.7	**48**	11	12	2.2
17	RS	38	0.16	**49**	11	29	0.45
18	VB	184	0.8	**50**	12	11	2.2
19	1	CAI	0.24	**51**	12	13	2
20	1	2	2.52	**52**	12	58	0.2
21	1	3	0.95	**53**	13	12	2
22	2	1	2.52	**54**	13	14	1.8
23	2	4	0.6	**55**	13	91	0.75
24	2	192	4.3	**56**	14	13	1.8
25	3	1	0.95	**57**	14	58	1
26	3	9	1.9	**58**	14	RONAL	0.75
27	3	19	2	**59**	15	9	2.3
28	4	2	0.6	**60**	15	16	1.3
29	4	5	1.4	**61**	15	192	2.2
30	4	19	3.2	**61**	16	15	1.3
31	5	4	1.4	**63**	16	17	0.5
32	5	6	2.5	**64**	16	98	0.7

No.	Titik 1	Titik 2	Bobot	No.	Titik 1	Titik 2	Bobot
65	17	16	0.5	**100**	31	26	4.3
66	17	18	0.61	**101**	31	30	0.07
67	18	17	0.61	**102**	31	165	1.35
68	18	DENI	0.16	**103**	32	33	0.45
69	19	3	1.76	**104**	32	54	0.18
70	19	4	3.2	**105**	33	32	0.45
71	19	106	2	**106**	33	34	0.05
72	20	21	2.7	**107**	34	27	4.6
73	20	121	0.5	**108**	34	35	4
74	20	125	0.1	**109**	35	36	0.2
75	21	20	2.7	**110**	35	164	4.1
76	21	22	0.25	**111**	36	28	0.22
77	22	21	0.25	**112**	36	35	0.2
78	22	23	0.4	**113**	36	37	0.4
79	22	50	1.6	**114**	37	36	0.4
80	23	21	0.2	**115**	37	38	0.23
81	23	24	1.9	**116**	38	28	0.45
82	24	23	1.9	**117**	38	37	0.23
83	24	50	2	**118**	38	RS	0.16
84	24	63	0.24	**119**	39	6	0.45
85	25	RONAL	1.7	**120**	39	40	0.88
86	25	26	0.23	**121**	39	49	2.7
87	26	25	0.23	**122**	39	200	0.6
88	26	27	0.28	**123**	40	39	0.88
89	26	31	4.3	**124**	40	41	0.28
90	27	26	0.28	**125**	41	40	0.28
91	27	28	2.1	**126**	41	42	0.85
92	27	34	4.6	**127**	42	41	0.85
93	28	27	2.1	**128**	42	43	0.65
94	28	36	0.22	**129**	43	42	0.65
95	28	38	0.45	**130**	43	44	1.2
96	29	56	0.35	**131**	44	43	1.2
97	29	61	2	**132**	44	45	0.55
98	30	31	0.05	**133**	45	44	0.55
99	30	59	0.19	**134**	45	46	0.24

No.	Titik 1	Titik 2	Bobot	No.	Titik 1	Titik 2	Bobot
135	45	178	2.7	**169**	61	30	0.8
136	46	45	0.24	**170**	62	170	0.35
137	46	47	3.2	**171**	62	174	0.11
138	46	48	1.3	**172**	63	64	0.75
139	47	46	3.2	**173**	63	67	0.35
140	47	SSB	1.3	**174**	64	65	0.35
141	48	46	1.3	**175**	64	162	0.5
142	48	49	0.4	**176**	65	24	0.7
143	49	39	2.7	**177**	65	160	2.4
144	49	62	0.5	**178**	66	67	0.5
145	50	22	1.6	**179**	66	69	0.5
146	50	24	2	**180**	66	157	2.7
147	50	51	0.06	**181**	67	163	0.4
148	51	50	0.06	**182**	68	69	0.14
149	51	84	0.04	**183**	68	NOVOTEL	2.9
150	52	168	0.04	**184**	69	66	0.5
151	52	169	0.03	**185**	69	71	0.4
152	53	54	0.45	**186**	70	68	0.4
153	53	86	0.45	**187**	71	72	0.35
154	54	53	0.45	**188**	71	74	0.2
155	54	55	0.45	**189**	72	70	0.3
156	55	166	0.06	**190**	72	73	0.35
157	55	167	0.05	**191**	73	72	0.35
158	56	29	0.35	**192**	73	85	1.7
159	56	57	0.5	**193**	74	75	1
160	56	129	1.9	**194**	74	76	0.13
161	57	10	0.27	**195**	75	74	1
162	58	12	0.2	**196**	75	81	0.8
163	58	14	1	**197**	76	73	0.5
164	58	ARI	0.1	**198**	76	77	0.14
165	59	60	0.85	**199**	77	76	0.14
166	59	104	0.5	**200**	77	78	0.6
167	60	61	0.19	**201**	78	75	0.03
168	61	29	2	**202**	79	77	0.9

No.	Titik 1	Titik 2	Bobot	No.	Titik 1	Titik 2	Bobot
203	80	78	0.45	**238**	98	97	0.45
204	80	79	0.4	**239**	98	99	0.21
205	81	82	0.07	**240**	99	98	0.21
206	81	83	0.35	**241**	99	100	0.1
207	82	80	0.21	**242**	100	99	0.1
208	83	51	0.05	**243**	100	101	0.12
209	83	84	0.04	**244**	101	100	0.12
210	84	52	0.75	**245**	101	102	0.04
211	85	73	1.7	**246**	102	101	0.04
212	85	86	0.6	**247**	102	DENI	0.05
213	85	172	0.55	**248**	103	104	0.5
214	86	172	0.03	**249**	103	105	1.5
215	86	173	0.8	**250**	103	169	1.3
216	87	88	0.25	**251**	104	59	0.5
217	87	90	0.26	**252**	104	103	0.5
218	88	85	0.4	**253**	105	103	1.5
219	89	88	0.12	**254**	105	126	0.09
220	90	89	0.23	**255**	105	128	0.21
221	90	175	0.8	**256**	106	19	2
222	91	13	0.75	**257**	106	118	0.19
223	91	92	1.1	**258**	106	125	2.4
224	92	91	1.1	**259**	107	108	2.3
225	92	93	0.3	**260**	107	120	1
226	93	92	0.3	**261**	107	121	1.6
227	93	94	1	**262**	108	107	2.3
228	94	93	1	**263**	108	109	0.4
229	94	95	0.16	**264**	108	202	1.4
230	95	94	0.16	**265**	108	203	0.9
231	95	96	0.35	**266**	109	108	0.4
232	95	194	3.1	**267**	109	110	0.45
233	96	95	0.35	**268**	109	202	1.7
234	96	97	0.25	**269**	109	203	1.3
235	97	96	0.25	**270**	110	109	0.45
236	97	98	0.45	**271**	110	111	3.3
237	98	16	0.7	**272**	110	174	4.6

No.	Titik 1	Titik 2	Bobot	No.	Titik 1	Titik 2	Bobot
273	111	110	3.3	**308**	125	20	0.1
274	111	112	0.3	**309**	125	106	2.4
275	111	117	0.45	**310**	125	126	2.5
276	112	111	0.3	**311**	126	105	0.09
277	112	113	0.28	**312**	126	125	2.5
278	113	112	0.28	**313**	126	127	0.3
279	113	114	0.15	**314**	127	126	0.3
280	113	117	0.65	**315**	127	128	0.5
281	114	113	0.15	**316**	128	105	0.21
282	114	115	0.6	**317**	128	127	0.5
283	114	122	0.6	**318**	128	129	0.25
284	115	114	0.6	**319**	129	56	1.9
285	115	124	0.8	**320**	129	128	0.25
286	116	117	4.6	**321**	130	124	3.2
287	116	170	1.5	**322**	130	138	1.1
288	117	111	0.45	**323**	130	PDKL	01.05
289	117	113	0.65	**324**	131	132	3
290	117	116	4.6	**325**	131	PDKL	0.7
291	118	106	0.19	**326**	132	131	3
292	118	119	1.1	**327**	132	133	0.27
293	119	118	1.1	**328**	132	134	0.5
294	119	120	0.05	**329**	133	132	0.27
295	120	107	1	**330**	133	136	1.1
296	120	119	0.05	**331**	134	132	0.5
297	121	20	0.5	**332**	134	135	0.65
298	121	107	1.6	**333**	135	134	0.65
299	121	122	6.8	**334**	135	BMW	2.3
300	122	114	0.6	**335**	136	133	1.1
301	122	121	6.8	**336**	136	137	2.4
302	122	123	0.65	**337**	136	148	0.9
303	123	122	0.65	**338**	137	136	2.4
304	123	124	0.65	**339**	137	138	0.27
305	124	115	0.8	**340**	137	140	0.85
306	124	123	0.65	**341**	138	130	1.1
307	124	130	3.2	**342**	138	137	0.27

No.	Titik 1	Titik 2	Bobot	No.	Titik 1	Titik 2	Bobot
343	138	144	1.1	**378**	150	151	1.5
344	139	145	0.35	**379**	150	153	3.2
345	139	147	0.45	**380**	151	150	1.5
346	139	161	0.26	**381**	151	152	0.24
347	140	137	0.85	**382**	152	144	0.75
348	140	141	0.15	**383**	152	151	0.24
349	140	142	0.4	**384**	152	158	2
350	141	140	0.19	**385**	153	150	3.2
351	141	143	0.35	**386**	153	154	1.4
352	141	144	0.65	**387**	153	156	1.3
353	142	140	0.4	**388**	154	153	0.85
354	142	145	0.3	**389**	154	155	6.1
355	143	141	0.35	**390**	154	156	1.1
356	143	146	0.5	**391**	155	154	6.1
357	144	138	1.1	**392**	155	BMW	3.3
358	144	141	0.65	**393**	156	8	0.95
359	144	152	0.75	**394**	156	154	1.1
360	145	139	0.35	**395**	156	157	0.6
361	145	142	0.3	**396**	156	GWR	0.3
362	145	146	0.35	**397**	157	8	0.45
363	145	161	0.5	**398**	157	156	0.6
364	146	143	0.5	**399**	157	162	1.9
365	146	145	0.35	**400**	157	GWR	0.9
366	146	147	0.35	**401**	158	152	2
367	147	139	0.45	**402**	158	159	0.8
368	147	146	0.35	**403**	158	206	0.22
369	147	149	1.8	**404**	159	158	0.45
370	147	150	0.35	**405**	159	204	0.95
371	148	136	0.9	**406**	160	65	3
372	148	149	1.6	**407**	160	176	0.27
373	148	161	0.45	**408**	161	139	0.26
374	149	147	1.8	**409**	161	145	0.5
375	149	148	1.6	**410**	161	148	0.45
376	149	BMW	3	**411**	162	64	0.5
377	150	147	0.35	**412**	162	66	0.85

No.	Titik 1	Titik 2	Bobot	No.	Titik 1	Titik 2	Bobot
413	163	66	0.23	**448**	182	183	0.6
414	163	NOVOTEL	0.4	**449**	182	195	1
415	164	175	0.15	**450**	183	182	0.28
416	165	32	0.17	**451**	183	184	2.3
417	165	166	0.03	**452**	183	195	0.75
418	166	165	0.03	**453**	184	183	2.3
419	166	167	0.06	**454**	184	189	1.7
420	167	30	1.2	**455**	184	VB	0.8
421	168	53	0.7	**456**	185	180	2.9
422	169	103	1.3	**457**	185	186	0.8
423	169	168	0.04	**458**	186	185	0.8
424	170	116	1.5	**459**	186	187	1.1
425	170	171	0.4	**460**	187	186	1.1
426	171	48	0.85	**461**	187	188	0.5
427	172	87	0.17	**462**	188	5	1.9
428	173	82	0.08	**463**	188	187	1.1
429	174	110	4.6	**464**	189	184	1.7
430	174	171	0.14	**465**	189	190	3.3
431	175	33	0.13	**466**	190	189	3.3
432	175	90	0.8	**467**	190	191	3.8
433	176	177	0.4	**468**	191	190	3.8
434	176	205	0.65	**469**	191	192	7.2
435	177	160	0.17	**470**	192	2	4.3
436	177	176	0.4	**471**	192	15	2.2
437	177	206	0.4	**472**	192	191	7.2
438	178	45	2.7	**473**	192	193	0.65
439	178	179	0.7	**474**	193	192	0.65
440	179	178	0.7	**475**	193	194	0.19
441	179	197	0.17	**476**	194	95	3.1
442	180	181	0.35	**477**	194	193	0.19
443	180	185	2.9	**478**	195	182	1
444	180	197	0.35	**479**	195	183	0.75
445	181	180	0.35	**480**	195	196	1.2
446	181	182	1.9	**481**	196	195	1.4
447	182	181	1.9	**482**	196	198	2.7

No.	Titik 1	Titik 2	Bobot
483	197	179	0.17
484	197	180	0.35
485	197	198	1.2
486	197	199	2.1
487	198	196	2.7
488	198	197	2
489	198	199	0.7
490	199	197	1.7
491	199	198	0.7
492	199	200	2.7
493	200	6	0.7
494	200	199	2.7
495	200	201	0.8
496	201	39	0.65
497	201	200	0.8
498	201	202	4.5
499	202	108	1.3
500	202	109	1
501	202	201	4.5
502	202	203	0.8
503	203	108	1.6
504	203	109	1.9
505	203	202	0.8
506	203	204	4.5
507	204	176	1.8
508	204	203	4.5
509	204	205	1.4
510	205	7	1.3
511	205	158	1.4
512	205	159	1.5
513	205	204	1.4
514	205	206	0.95
515	206	158	0.22
516	206	159	0.9
517	206	177	0.4

Lampiran 2: Skrip Program

```
#Pendefinisian algoritma Floyd Warshall

def SFW (graph,weight):
    jarak = defaultdict(lambda: defaultdict(lambda: float('inf')))
    titik = defaultdict(dict)
    TidakBerarah = not graph.is_directed()

    for i in graph:
        jarak[i][i] = 0

    for i, j, k in graph.edges(data=True):
        bobot = k.get(weight)
        jarak[i][j] = min(bobot,jarak[i][j])
        titik[i][j] = i

    for w in graph:
        jarak_w = jarak[w]  # simpan perhitungan
        for i in graph:
            jarak_i = jarak[i]  # simpan perhitungan
            for j in graph:
                k = jarak_i[w] + jarak_w[j]
                if jarak_i[j] > k:
                    jarak_i[j] = k
                    titik[i][j] = titik[w][j]

    return dict(titik), dict(jarak)

#Pendefinisian rekonstruksi jalur

def rekonstruksi_jalur(t_awal, t_tujuan, predecessors):
    if t_awal == t_tujuan:
        return []
    t_sebelum = predecessors[t_awal]
    t_sekarang = t_sebelum[t_tujuan]
    jalur = [t_tujuan, t_sekarang]
    while t_sekarang != t_awal:
        t_sekarang = t_sebelum[t_sekarang]
        jalur.append(t_sekarang)
    return list(reversed(jalur))

#Pendefinisian penyeleksian rute terpendek terlebih dahulu

def penyeleksianrtd(X,titiktujuan,rute,totaljarak):
    penyeleksian=[]

    if len(titiktujuan)==0:
        print('Rute yang dilalui : ',rute)
        print('Total jarak tempuh : ',totaljarak, 'Km')
```

```
        for i in rute:
            for j in range (len(i)-1):

                nx.draw_networkx_edges(g,pos,edgelist=[[i[j],i[j+1
                ]]],width=2, edge_color='red')
                nx.draw_networkx_edges(g, pos, edgelist=[i[-1]],
                edge_color='purple')
                nx.draw_networkx_edges(g, pos,
                edgelist=[rute[0][0]], edge_color='green')
        plt.show()

    else :
        for i in titiktujuan:
            sisi=rekonstruksi_jalur(X,i,semuajalur)
            jarak=semuajarak[X][i]
            penyeleksian.append([sisi,jarak])

        A1=min(penyeleksian, key=lambda x:x[1] )
        penyeleksian.clear()
        rute.append(A1[0])
        jarak1=A1[1]
        totaljarak+=jarak1
        A2=(A1[0][-1])
        titiktujuan.remove(A2)
        return penyeleksianrtd(A2,titiktujuan,rute,totaljarak)
```

Lampiran 3: Hasil Seluruh Pencarian Rute Terpendek algoritma *Floyd Warshall*

TITIK	RUTE YANG DILALUI
CAI	{'1': 'CAI', '2': '1', '3': '1', '4': '2', '192': '2', '9': '3', '19': '3', '5': '4', '15': '9', '191': '192', '193': '192', '10': '9', '106': '19', '6': '5', '188': '5', '39': '6', '201': '6', '187': '188', '40': '39', '49': '39', '200': '39', '202': '108', '11': '10', '16': '15', '12': '11', '29': '11', 'ARI': '58', '58': '12', 'RONAL': '14', '14': '58', '25': 'RONAL', '13': '12', '26': '25', '56': '29', '61': '29', '91': '92', '92': '93', '17': '16', '98': '16', '18': '17', 'DENI': '102', '97': '98', '99': '98', '125': '106', '118': '106', '20': '125', '21': '20', '121': '20', '22': '21', '126': '105', '107': '120', '122': '121', '23': '22', '50': '22', '24': '23', '51': '50', '63': '24', '64': '63', '67': '63', '27': '26', '31': '30', '28': '27', '34': '33', '30': '61', '165': '31', 'RS': '38', '38': '28', '36': '28', '37': '36', '35': '36', '57': '56', '129': '56', '59': '30', '60': '59', '104': '59', '32': '165', '166': '165', '33': '32', '54': '32', '53': '54', '55': '54', '164': '35', '175': '90', '41': '40', '62': '49', '199': '200', '42': '41', '43': '42', '44': '43', '45': '44', '46': '45', '178': '179', 'SSB': '47', '47': '46', '48': '171', '179': '197', '170': '62', '174': '62', '84': '51', '52': '84', '168': '169', '169': '103', '103': '104', '86': '53', '172': '86', '173': '86', '167': '166', '128': '129', '116': '170', '171': '174', '110': '109', '65': '64', '162': '64', '163': '67', '160': '65', '66': '163', '176': '160', 'GWR': '156', '8': '157', '156': '157', '157': '66', '7': '205', '205': '204', '69': '66', '158': '206', '154': '156', '159': '205', '206': '205', '204': '203', '71': '69', 'NOVOTEL': '163', '68': '70', '72': '73', '74': '75', '70': '72', '73': '85', '75': '78', '76': '77', '85': '88', '81': '75', '77': '79', '82': '173', '83': '81', '78': '80', '79': '80', '80': '82', '87': '172', '88': '87', '90': '87', '89': '90', '93': '94', '94': '95', '95': '96', '96': '97', '194': '193', '100': '99', '101': '100', '102': '101', '105': '128', '127': '128', '119': '118', '108': '107', '120': '119', '109': '108', '203': '108', '111': '110', '112': '111', '117': '111', '113': '112', '114': '113', '115': '114', '124': '115', '123': '122', 'PDKL': '130', '130': '124', '131': 'PDKL', '138': '130', '132': '131', '137': '138', '144': '152', '133': '136', '134': '132',

	'136': '137', 'BMW': '149', '135': '134', '149': '147', '155': '154', '148': '161', '147': '150', '140': '141', '161': '145', '141': '144', '142': '140', '152': '158', '139': '147', '145': '142', '146': '143', '150': '151', '143': '141', '151': '152', '153': '154', '177': '176', '197': '199', '180': '185', '198': '199', '181': '180', '185': '186', '182': '181', '186': '187', '183': '182', '195': '182', 'VB': '184', '184': '183', '189': '184', '196': '198', '190': '191'}
ARI	{'58': 'ARI', '14': '58', '12': '58', 'RONAL': '14', '25': 'RONAL', '13': '12', '26': '25', '11': '12', '29': '11', '56': '29', '61': '29', '91': '13', '92': '91', '27': '26', '31': '30', '28': '27', '34': '33', '30': '61', '165': '31', 'RS': '38', '38': '28', '36': '28', '37': '36', '35': '36', '57': '56', '129': '56', '59': '30', '60': '59', '104': '59', '32': '165', '166': '165', '33': '32', '54': '32', '53': '54', '55': '54', '164': '35', '175': '90', '86': '53', '172': '86', '173': '86', '167': '166', 'CAI': '1', '1': '3', 'SSB': '47', '47': '46', '18': '17', '2': '1', '3': '9', '4': '2', '192': '193', '9': '10', '19': '3', '5': '4', '6': '5', '188': '5', '39': '6', '201': '6', '10': '57', '15': '16', '16': '98', '17': '16', '98': '97', 'DENI': '102', '106': '125', '20': '125', '21': '20', '121': '20', '125': '126', '22': '21', '23': '22', '50': '51', '24': '50', '63': '24', '40': '39', '49': '48', '200': '39', '41': '40', '42': '41', '43': '42', '44': '43', '45': '46', '46': '48', '178': '179', '48': '171', '62': '49', '51': '83', '84': '83', '52': '84', '168': '169', '169': '103', '170': '62', '174': '110', '64': '63', '67': '66', '97': '96', '99': '98', '103': '104', '126': '105', '118': '106', '107': '121', '202': '108', '122': '121', '179': '197', '187': '188', '191': '192', '193': '194', '199': '200', '128': '129', '116': '117', '171': '174', '110': '109', '65': '64', '162': '64', '163': '67', '160': '65', '66': '69', '176': '160', 'GWR': '156', '8': '157', '156': '157', '157': '66', '7': '8', '205': '176', '69': '68', '158': '206', '154': '156', '159': '206', '206': '177', '204': '203', '71': '69', 'NOVOTEL': '163', '68': '70', '72': '73', '74': '75', '70': '72', '73': '85', '75': '78', '76': '77', '85': '88', '81': '75', '77': '79', '82': '173', '83': '81', '78': '80', '79': '80', '80': '82', '87': '172', '88': '87', '90': '87', '89': '90', '93': '92', '94': '93', '95': '94', '96': '95', '194': '95', '100': '99', '101':

	'100', '102': '101', '105': '128', '127': '128', '119': '120', '108': '107', '120': '107', '109': '108', '203': '108', '111': '110', '112': '113', '117': '113', '113': '114', '114': '122', '115': '114', '124': '123', '123': '122', 'PDKL': '130', '130': '124', '131': 'PDKL', '138': '130', '132': '133', '137': '138', '144': '152', '133': '136', '134': '132', '136': '148', 'BMW': '149', '135': '134', '149': '147', '155': '154', '148': '161', '147': '150', '140': '141', '161': '139', '141': '144', '142': '140', '152': '158', '139': '147', '145': '146', '146': '147', '150': '153', '143': '141', '151': '152', '153': '154', '177': '176', '197': '199', '180': '185', '198': '199', '181': '180', '185': '186', '182': '181', '186': '187', '183': '182', '195': '182', 'VB': '184', '184': '189', '189': '190', '196': '198', '190': '191'}
SSB	{'47': 'SSB', '46': '47', 'CAI': '1', '1': '2', '18': '17', 'ARI': '58', '58': '12', 'RONAL': '14', '14': '58', '25': '26', 'RS': '38', '38': '37', '2': '4', '3': '1', '4': '5', '192': '2', '9': '3', '19': '4', '5': '6', '6': '39', '188': '5', '39': '49', '201': '6', '10': '9', '15': '192', '11': '10', '12': '11', '29': '56', '13': '91', '91': '92', '16': '15', '17': '16', '98': '16', 'DENI': '102', '106': '118', '20': '121', '21': '20', '121': '107', '125': '20', '22': '21', '23': '22', '50': '22', '24': '23', '63': '24', '26': '31', '27': '26', '31': '30', '28': '36', '34': '33', '36': '35', '56': '129', '61': '60', '30': '61', '59': '104', '165': '166', '32': '165', '33': '32', '54': '53', '35': '34', '164': '35', '37': '36', '40': '41', '49': '48', '200': '39', '41': '42', '42': '43', '43': '44', '44': '45', '45': '46', '178': '45', '48': '46', '62': '49', '51': '50', '53': '168', '55': '54', '166': '55', '57': '56', '129': '128', '60': '59', '104': '103', '64': '63', '67': '63', '175': '90', '92': '93', '97': '96', '99': '98', '126': '125', '118': '119', '107': '108', '202': '109', '122': '114', '187': '186', '191': '190', '193': '192', '199': '198', '179': '178', '170': '62', '174': '62', '84': '51', '52': '84', '168': '52', '169': '52', '103': '105', '86': '53', '172': '86', '173': '86', '167': '55', '128': '105', '116': '170', '171': '174', '110': '174', '65': '160', '162': '64', '163': '67', '160': '177', '66': '163', '176': '204', 'GWR': '156', '8': '7', '156': '8', '157': '8', '7': '205', '205': '204', '69': '66', '158': '206', '154': '156',

	'159': '205', '206': '205', '204': '203', '71': '69', 'NOVOTEL': '163', '68': '70', '72': '71', '74': '71', '70': '72', '73': '72', '75': '78', '76': '74', '85': '88', '81': '75', '77': '76', '82': '173', '83': '81', '78': '80', '79': '80', '80': '82', '87': '172', '88': '87', '90': '87', '89': '90', '93': '94', '94': '95', '95': '194', '96': '95', '194': '193', '100': '99', '101': '100', '102': '101', '105': '126', '127': '126', '119': '120', '108': '109', '120': '107', '109': '110', '203': '109', '111': '117', '112': '111', '117': '116', '113': '117', '114': '113', '115': '114', '124': '115', '123': '122', 'PDKL': '130', '130': '124', '131': 'PDKL', '138': '130', '132': '131', '137': '138', '144': '138', '133': '136', '134': '132', '136': '137', 'BMW': '135', '135': '134', '149': '147', '155': '154', '148': '161', '147': '146', '140': '137', '161': '145', '141': '140', '142': '140', '152': '144', '139': '145', '145': '142', '146': '143', '150': '147', '143': '141', '151': '152', '153': '154', '177': '176', '197': '179', '180': '197', '198': '197', '181': '180', '185': '180', '182': '181', '186': '185', '183': '182', '195': '182', 'VB': '184', '184': '183', '189': '184', '196': '198', '190': '189'}
GWR	{'8': '157', '156': 'GWR', '157': 'GWR', '7': '8', '154': '156', '162': '157', '205': '7', '158': '206', '159': '205', '206': '205', '204': '205', 'CAI': '1', '1': '3', 'SSB': '47', '47': '46', '18': '17', 'ARI': '58', '58': '12', 'RONAL': '25', '14': 'RONAL', '25': '26', 'RS': '38', '38': '37', '2': '4', '3': '19', '4': '19', '192': '15', '9': '10', '19': '106', '5': '4', '6': '39', '188': '5', '39': '201', '201': '202', '10': '57', '15': '9', '11': '10', '12': '11', '29': '61', '13': '12', '91': '13', '16': '15', '17': '16', '98': '16', 'DENI': '102', '106': '125', '20': '21', '21': '23', '121': '20', '125': '20', '22': '21', '23': '24', '50': '24', '24': '65', '63': '24', '26': '27', '27': '34', '31': '30', '28': '36', '34': '33', '36': '35', '56': '129', '61': '60', '30': '167', '59': '104', '165': '166', '32': '165', '33': '175', '54': '53', '35': '34', '164': '35', '37': '36', '40': '39', '49': '48', '200': '201', '41': '40', '42': '41', '43': '42', '44': '43', '45': '46', '46': '48', '178': '179', '48': '171', '62': '49', '51': '50', '84': '51', '52': '84', '168': '52', '169': '52', '53': '168', '86': '85', '55': '54', '166': '55', '167': '55', '57': '56', '129':

	'128', '60': '59', '104': '103', '170': '62', '174': '110', '64': '162', '67': '66', '65': '64', '160': '177', '66': '162', '163': '67', '172': '85', '173': '86', '175': '90', '92': '91', '97': '98', '99': '98', '103': '169', '126': '105', '128': '105', '118': '106', '107': '121', '202': '203', '110': '109', '122': '123', '116': '117', '171': '174', '179': '197', '187': '188', '191': '192', '193': '192', '199': '200', '176': '177', '69': '66', '71': '69', 'NOVOTEL': '163', '68': '70', '72': '71', '74': '71', '70': '72', '73': '72', '75': '78', '76': '74', '85': '73', '81': '75', '77': '76', '82': '81', '83': '81', '78': '77', '79': '80', '80': '82', '87': '172', '88': '87', '90': '87', '89': '90', '93': '92', '94': '93', '95': '94', '96': '97', '194': '193', '100': '99', '101': '100', '102': '101', '105': '103', '127': '126', '119': '120', '108': '203', '120': '107', '109': '202', '203': '204', '111': '112', '112': '113', '117': '113', '113': '114', '114': '115', '115': '124', '124': '130', '123': '124', 'PDKL': '130', '130': '138', '131': 'PDKL', '138': '137', '132': '133', '137': '140', '144': '152', '133': '136', '134': '132', '136': '148', 'BMW': '149', '135': '134', '149': '147', '155': '154', '148': '161', '147': '150', '140': '141', '161': '139', '141': '143', '142': '145', '152': '158', '139': '147', '145': '146', '146': '147', '150': '153', '143': '146', '151': '152', '153': '154', '177': '206', '197': '199', '180': '197', '198': '199', '181': '180', '185': '186', '182': '181', '186': '187', '183': '182', '195': '196', 'VB': '184', '184': '183', '189': '184', '196': '198', '190': '189'}
BMW	{'135': 'BMW', '149': 'BMW', '155': 'BMW', '134': '135', '148': '149', '147': '149', '154': '155', 'CAI': '1', '1': '3', 'SSB': '47', '47': '46', 'NOVOTEL': '163', '68': '70', 'GWR': '156', '8': '156', '156': '153', '157': '156', 'PDKL': '131', '130': '138', '131': '132', '18': '17', '102': '101', 'ARI': '58', '58': '12', 'RONAL': '25', '14': 'RONAL', '25': '26', 'RS': '38', '38': '37', '2': '4', '3': '19', '4': '5', '192': '15', '9': '10', '19': '106', '5': '6', '6': '39', '188': '5', '39': '201', '201': '202', '7': '8', '205': '176', '10': '57', '15': '9', '11': '10', '12': '11', '29': '61', '13': '12', '91': '13', '16': '15', '17': '16', '98': '16', 'DENI': '102', '106': '125', '20': '21', '21': '23', '121': '20', '125': '20', '22': '21', '23': '24',

	'50': '24', '24': '65', '63': '24', '26': '27', '27': '34', '31': '30', '28': '36', '34': '33', '36': '35', '56': '129', '61': '60', '30': '167', '59': '104', '165': '166', '32': '165', '33': '175', '54': '53', '35': '34', '164': '35', '37': '36', '40': '39', '49': '48', '200': '201', '41': '40', '42': '41', '43': '42', '44': '45', '45': '46', '46': '48', '178': '45', '48': '171', '62': '49', '51': '50', '84': '51', '52': '84', '168': '52', '169': '52', '53': '168', '86': '85', '55': '54', '166': '55', '167': '55', '57': '56', '129': '128', '60': '59', '104': '103', '170': '116', '174': '110', '64': '162', '67': '66', '65': '160', '162': '157', '160': '177', '66': '162', '69': '66', '163': '67', '71': '69', '70': '72', '72': '71', '74': '71', '73': '72', '85': '73', '75': '78', '76': '74', '81': '75', '77': '76', '78': '77', '79': '80', '80': '82', '82': '81', '83': '81', '172': '85', '173': '86', '87': '172', '88': '87', '90': '87', '89': '90', '175': '90', '92': '91', '93': '92', '94': '93', '95': '94', '96': '97', '194': '193', '97': '98', '99': '98', '100': '99', '101': '100', '103': '169', '105': '103', '126': '105', '128': '105', '118': '106', '107': '108', '108': '203', '120': '107', '109': '202', '202': '203', '203': '204', '110': '111', '111': '112', '112': '113', '117': '113', '113': '114', '114': '115', '115': '124', '122': '123', '124': '130', '116': '117', '119': '120', '123': '124', '127': '126', '138': '137', '132': '134', '133': '132', '136': '133', '137': '140', '144': '141', '158': '152', '159': '158', '206': '158', '204': '159', '176': '177', '171': '170', '179': '197', '187': '188', '191': '192', '193': '192', '199': '200', '140': '141', '161': '148', '141': '143', '142': '145', '152': '151', '139': '147', '145': '146', '146': '147', '150': '147', '143': '146', '151': '150', '153': '150', '177': '206', '197': '199', '180': '197', '198': '199', '181': '180', '185': '180', '182': '181', '186': '187', '183': '182', '195': '196', 'VB': '184', '184': '183', '189': '184', '196': '198', '190': '189'}
RS	{'38': 'RS', '28': '38', '37': '38', 'ARI': '58', '58': '14', 'RONAL': '25', '14': 'RONAL', '25': '26', '11': '12', '12': '58', '29': '11', '13': '14', '91': '13', '26': '27', '27': '28', '31': '26', '34': '33', '36': '37', '56': '29', '61': '60', '30': '31', '165': '166', '92': '91', '35': '36', '57': '56', '129': '128', '59': '30', '60': '59', '104': '59', '32': '33', '166': '55', '33': '175', '54': '32', '53': '54', '55': '54', '164': '35', '175': '164', '86': '53', '172':

	'86', '173': '86', '167': '55', 'CAI': '1', '1': '3', 'SSB': '47', '47': '46', '18': '17', '2': '1', '3': '9', '4': '2', '192': '193', '9': '10', '19': '106', '5': '4', '6': '5', '188': '5', '39': '201', '201': '202', '10': '57', '15': '16', '16': '98', '17': '16', '98': '97', 'DENI': '102', '106': '125', '20': '125', '21': '22', '121': '20', '125': '126', '22': '50', '23': '22', '50': '51', '24': '50', '63': '24', '40': '39', '49': '48', '200': '201', '41': '40', '42': '41', '43': '42', '44': '45', '45': '46', '46': '48', '178': '45', '48': '171', '62': '49', '51': '83', '84': '83', '52': '84', '168': '169', '169': '103', '170': '62', '174': '110', '64': '63', '67': '66', '97': '96', '99': '98', '103': '104', '126': '105', '118': '106', '107': '121', '202': '108', '122': '121', '179': '197', '187': '188', '191': '192', '193': '194', '199': '200', '128': '105', '116': '117', '171': '174', '110': '109', '65': '64', '162': '64', '163': '67', '160': '65', '66': '69', '176': '160', 'GWR': '156', '8': '157', '156': '157', '157': '66', '7': '8', '205': '7', '69': '68', '158': '206', '154': '156', '159': '205', '206': '177', '204': '205', '71': '69', 'NOVOTEL': '163', '68': '70', '72': '73', '74': '75', '70': '72', '73': '85', '75': '78', '76': '77', '85': '88', '81': '75', '77': '79', '82': '173', '83': '81', '78': '80', '79': '80', '80': '82', '87': '172', '88': '89', '90': '175', '89': '90', '93': '92', '94': '93', '95': '94', '96': '95', '194': '95', '100': '99', '101': '100', '102': '101', '105': '103', '127': '126', '119': '120', '108': '107', '120': '107', '109': '108', '203': '108', '111': '110', '112': '113', '117': '113', '113': '114', '114': '122', '115': '114', '124': '123', '123': '122', 'PDKL': '130', '130': '138', '131': 'PDKL', '138': '144', '132': '133', '137': '138', '144': '152', '133': '136', '134': '132', '136': '148', 'BMW': '149', '135': '134', '149': '147', '155': '154', '148': '161', '147': '150', '140': '141', '161': '139', '141': '144', '142': '145', '152': '158', '139': '147', '145': '146', '146': '147', '150': '153', '143': '146', '151': '152', '153': '154', '177': '176', '197': '199', '180': '197', '198': '199', '181': '180', '185': '186', '182': '181', '186': '187', '183': '182', '195': '196', 'VB': '184', '184': '189', '189': '190', '196': '198', '190': '191'}
VB	{'184': 'VB', '183': '184', '189': '184', 'CAI': '1', '1': '2', 'SSB': '47',

	'47': '46', 'NOVOTEL': '163', '68': '70', 'GWR': '156', '8': '7', '156': '8', '157': '8', 'BMW': '135', '135': '134', '149': '147', '155': '154', 'PDKL': '130', '130': '124', '131': 'PDKL', '18': '17', '102': '101', 'ARI': '58', '58': '12', 'RONAL': '14', '14': '13', '25': 'RONAL', 'RS': '38', '38': '28', '2': '4', '3': '1', '4': '5', '192': '191', '9': '3', '19': '4', '5': '188', '6': '200', '188': '187', '39': '6', '201': '200', '7': '205', '205': '204', '10': '9', '15': '192', '11': '10', '12': '11', '29': '11', '13': '91', '91': '92', '16': '15', '17': '16', '98': '16', 'DENI': '102', '106': '19', '20': '121', '21': '20', '121': '107', '125': '20', '22': '21', '23': '22', '50': '22', '24': '23', '63': '24', '26': '25', '27': '26', '31': '30', '28': '27', '34': '33', '36': '28', '56': '29', '61': '29', '30': '61', '59': '104', '165': '31', '32': '165', '33': '32', '54': '53', '35': '36', '164': '35', '37': '36', '40': '39', '49': '48', '200': '199', '41': '42', '42': '43', '43': '44', '44': '45', '45': '178', '46': '45', '178': '179', '48': '46', '62': '49', '51': '50', '84': '51', '52': '84', '168': '52', '169': '52', '53': '168', '86': '53', '55': '54', '166': '165', '167': '166', '57': '56', '129': '128', '60': '59', '104': '103', '170': '62', '174': '62', '64': '162', '67': '63', '65': '160', '162': '157', '160': '177', '66': '162', '69': '66', '163': '67', '71': '69', '70': '72', '72': '71', '74': '71', '73': '72', '85': '88', '75': '78', '76': '74', '81': '75', '77': '76', '78': '80', '79': '80', '80': '82', '82': '173', '83': '81', '172': '86', '173': '86', '87': '172', '88': '87', '90': '87', '89': '90', '175': '90', '92': '93', '93': '94', '94': '95', '95': '194', '96': '95', '194': '193', '97': '96', '99': '98', '100': '99', '101': '100', '103': '105', '105': '126', '126': '125', '128': '105', '118': '106', '107': '108', '108': '202', '120': '107', '109': '202', '202': '201', '203': '202', '110': '174', '111': '117', '112': '111', '117': '116', '113': '117', '114': '113', '115': '114', '122': '114', '124': '115', '116': '170', '119': '120', '123': '122', '127': '126', '138': '130', '132': '131', '133': '136', '134': '132', '136': '137', '137': '138', '148': '161', '140': '137', '144': '138', '139': '145', '145': '142', '147': '146', '161': '145', '141': '140', '142': '140', '143': '141', '146': '143', '152': '158', '150': '151', '151': '152', '153': '154', '158': '206', '154': '156', '159': '205', '206': '205', '204': '203', '176': '204', '171':

	'174', '177': '176', '179': '197', '197': '180', '180': '181', '181': '182', '185': '180', '182': '183', '195': '183', '186': '185', '187': '186', '191': '190', '193': '192', '198': '197', '199': '198', '196': '195', '190': '189'}
PDKL	{'130': 'PDKL', '131': 'PDKL', '124': '130', '138': '130', '132': '131', 'CAI': '1', '1': '3', 'SSB': '47', '47': '46', 'NOVOTEL': '163', '68': '70', 'GWR': '156', '8': '7', '156': '153', '157': '156', '18': '17', '102': '101', 'ARI': '58', '58': '12', 'RONAL': '25', '14': '58', '25': '26', 'RS': '38', '38': '37', '2': '4', '3': '19', '4': '19', '192': '2', '9': '3', '19': '106', '5': '6', '6': '39', '188': '5', '39': '49', '201': '202', '7': '205', '205': '176', '10': '57', '15': '9', '11': '10', '12': '11', '29': '56', '13': '12', '91': '13', '16': '15', '17': '16', '98': '16', 'DENI': '102', '106': '118', '20': '121', '21': '23', '121': '122', '125': '20', '22': '21', '23': '24', '50': '24', '24': '65', '63': '24', '26': '31', '27': '34', '31': '30', '28': '36', '34': '33', '36': '35', '56': '129', '61': '60', '30': '167', '59': '104', '165': '166', '32': '165', '33': '32', '54': '53', '35': '34', '164': '35', '37': '36', '40': '39', '49': '48', '200': '201', '41': '40', '42': '43', '43': '44', '44': '45', '45': '46', '46': '48', '178': '45', '48': '171', '62': '49', '51': '50', '84': '51', '52': '84', '168': '52', '169': '52', '53': '168', '86': '53', '55': '54', '166': '55', '167': '55', '57': '56', '129': '128', '60': '59', '104': '103', '170': '116', '174': '110', '64': '63', '67': '63', '65': '160', '162': '157', '160': '177', '66': '163', '69': '66', '163': '67', '71': '69', '70': '72', '72': '71', '74': '71', '73': '72', '85': '73', '75': '78', '76': '74', '81': '75', '77': '76', '78': '77', '79': '80', '80': '82', '82': '81', '83': '81', '172': '86', '173': '86', '87': '172', '88': '87', '90': '87', '89': '90', '175': '90', '92': '91', '93': '92', '94': '93', '95': '96', '96': '97', '194': '193', '97': '98', '99': '98', '100': '99', '101': '100', '103': '169', '105': '126', '126': '125', '128': '105', '118': '119', '107': '108', '108': '109', '120': '107', '109': '110', '202': '109', '203': '109', '110': '111', '111': '112', '112': '113', '117': '113', '113': '114', '114': '115', '115': '124', '122': '123', '116': '117', '119': '120', '123': '124', '127': '126', '158': '152', '154': '153', '159': '158', '206': '158', '204': '159', '176': '177', '171': '170', '179': '178', '187': '188', '191': '192', '193': '192', '199': '200', '137':

	'138', '144': '138', '133': '132', '134': '132', '136': '133', 'BMW': '135', '135': '134', '149': '147', '155': 'BMW', '148': '161', '147': '146', '140': '137', '161': '145', '141': '140', '142': '140', '152': '144', '139': '145', '145': '142', '146': '143', '150': '147', '143': '141', '151': '152', '153': '150', '177': '206', '197': '179', '180': '197', '198': '197', '181': '180', '185': '180', '182': '181', '186': '185', '183': '182', '195': '182', 'VB': '184', '184': '183', '189': '184', '196': '198', '190': '189'}
DENI	{'18': 'DENI', '102': 'DENI', '17': '18', '101': '102', 'CAI': '1', '1': '3', 'ARI': '58', '58': '12', 'RONAL': '14', '14': '13', '25': 'RONAL', '2': '192', '3': '9', '4': '2', '192': '15', '9': '15', '19': '3', '5': '4', '6': '5', '188': '5', '39': '6', '201': '6', '10': '9', '15': '16', '11': '10', '12': '13', '29': '11', '13': '91', '91': '92', '16': '98', '98': '99', '106': '19', '26': '25', '56': '29', '61': '29', '40': '39', '49': '39', '200': '39', '92': '93', '202': '108', '187': '188', '191': '192', '193': '194', '97': '98', '99': '100', '125': '106', '118': '106', '20': '125', '21': '20', '121': '20', '22': '21', '126': '105', '107': '120', '122': '121', '23': '22', '50': '51', '24': '23', '51': '83', '63': '24', '64': '63', '67': '66', '27': '26', '31': '30', '28': '27', '34': '33', '30': '61', '165': '31', 'RS': '38', '38': '28', '36': '28', '37': '36', '35': '36', '57': '56', '129': '56', '59': '30', '60': '59', '104': '59', '32': '165', '166': '165', '33': '32', '54': '32', '53': '54', '55': '54', '164': '35', '175': '90', '41': '40', '62': '49', '199': '200', '42': '41', '43': '42', '44': '43', '45': '44', '46': '45', '178': '179', 'SSB': '47', '47': '46', '48': '171', '179': '197', '170': '62', '174': '62', '84': '83', '52': '84', '168': '169', '169': '103', '103': '104', '86': '53', '172': '86', '173': '86', '167': '166', '128': '129', '116': '170', '171': '174', '110': '109', '65': '64', '162': '64', '163': '67', '160': '65', '66': '69', '176': '160', 'GWR': '156', '8': '157', '156': '157', '157': '66', '7': '8', '205': '204', '69': '68', '158': '206', '154': '156', '159': '205', '206': '205', '204': '203', '71': '69', 'NOVOTEL': '163', '68': '70', '72': '73', '74': '75', '70': '72', '73': '85', '75': '78', '76': '77', '85': '88', '81': '75', '77': '79', '82': '173', '83': '81', '78': '80', '79': '80', '80': '82', '87': '172', '88': '87', '90': '87', '89': '90', '93': '94', '94': '95', '95': '96', '96': '97',

	'194': '95', '100': '101', '105': '128', '127': '128', '119': '118', '108': '107', '120': '119', '109': '108', '203': '108', '111': '110', '112': '111', '117': '111', '113': '112', '114': '113', '115': '114', '124': '115', '123': '122', 'PDKL': '130', '130': '124', '131': 'PDKL', '138': '130', '132': '131', '137': '138', '144': '152', '133': '136', '134': '132', '136': '137', 'BMW': '149', '135': '134', '149': '147', '155': '154', '148': '161', '147': '150', '140': '141', '161': '139', '141': '144', '142': '140', '152': '158', '139': '147', '145': '142', '146': '143', '150': '153', '143': '141', '151': '152', '153': '154', '177': '176', '197': '199', '180': '185', '198': '199', '181': '180', '185': '186', '182': '181', '186': '187', '183': '182', '195': '182', 'VB': '184', '184': '189', '189': '190', '196': '198', '190': '191'}
RONAL	{'14': 'RONAL', '25': 'RONAL', 'ARI': '58', '58': '14', '12': '58', '13': '14', '26': '25', '11': '12', '29': '11', '56': '29', '61': '29', '91': '13', '92': '91', '27': '26', '31': '26', '28': '27', '34': '27', '30': '31', '165': '31', 'RS': '38', '38': '28', '36': '28', '37': '36', '35': '36', '57': '56', '129': '56', '59': '30', '60': '59', '104': '59', '32': '165', '166': '165', '33': '32', '54': '32', '53': '54', '55': '54', '164': '35', '175': '164', '86': '53', '172': '86', '173': '86', '167': '166', 'CAI': '1', '1': '3', 'SSB': '47', '47': '46', 'DENI': '102', '18': 'DENI', '102': '101', '2': '1', '3': '9', '4': '2', '192': '193', '9': '10', '19': '3', '5': '4', '6': '5', '188': '5', '39': '6', '201': '6', '10': '57', '15': '16', '16': '98', '17': '16', '98': '97', '106': '125', '20': '125', '21': '20', '121': '20', '125': '126', '22': '21', '23': '22', '50': '51', '24': '50', '63': '24', '40': '39', '49': '48', '200': '39', '41': '40', '42': '41', '43': '42', '44': '43', '45': '46', '46': '48', '178': '179', '48': '171', '62': '49', '51': '83', '84': '83', '52': '84', '168': '169', '169': '103', '170': '62', '174': '110', '64': '63', '67': '66', '97': '96', '99': '98', '101': '100', '103': '104', '126': '105', '118': '106', '107': '121', '202': '108', '122': '121', '179': '197', '187': '188', '191': '192', '193': '194', '199': '200', '128': '129', '116': '117', '171': '174', '110': '109', '65': '64', '162': '64', '163': '67', '160': '65', '66': '69', '176': '160', 'GWR': '156', '8': '157', '156': '157', '157': '66', '7': '8', '205':

	'176', '69': '68', '158': '206', '154': '156', '159': '206', '206': '177', '204': '205', '71': '69', 'NOVOTEL': '163', '68': '70', '72': '73', '74': '75', '70': '72', '73': '85', '75': '78', '76': '77', '85': '88', '81': '75', '77': '79', '82': '173', '83': '81', '78': '80', '79': '80', '80': '82', '87': '172', '88': '87', '90': '87', '89': '90', '93': '92', '94': '93', '95': '94', '96': '95', '194': '95', '100': '99', '105': '128', '127': '128', '119': '120', '108': '107', '120': '107', '109': '108', '203': '108', '111': '110', '112': '113', '117': '113', '113': '114', '114': '122', '115': '114', '124': '123', '123': '122', 'PDKL': '130', '130': '124', '131': 'PDKL', '138': '144', '132': '133', '137': '138', '144': '152', '133': '136', '134': '132', '136': '148', 'BMW': '149', '135': '134', '149': '147', '155': '154', '148': '161', '147': '150', '140': '141', '161': '139', '141': '144', '142': '140', '152': '158', '139': '147', '145': '146', '146': '147', '150': '153', '143': '141', '151': '152', '153': '154', '177': '176', '197': '199', '180': '185', '198': '199', '181': '180', '185': '186', '182': '181', '186': '187', '183': '182', '195': '196', 'VB': '184', '184': '189', '189': '190', '196': '198', '190': '191'}
NOVOTEL	{'68': 'NOVOTEL', '69': '68', 'CAI': '1', '1': '3', 'SSB': '47', '47': '46', 'GWR': '156', '8': '157', '156': '157', '157': '66', 'DENI': '102', '18': '17', '102': '101', 'ARI': '58', '58': '12', 'RONAL': '25', '14': 'RONAL', '25': '26', 'RS': '38', '38': '37', '2': '4', '3': '19', '4': '19', '192': '15', '9': '10', '19': '106', '5': '4', '6': '39', '188': '5', '39': '201', '201': '202', '7': '8', '205': '7', '10': '57', '15': '9', '11': '10', '12': '11', '29': '61', '13': '12', '91': '13', '16': '15', '17': '16', '98': '16', '106': '125', '20': '21', '21': '22', '121': '20', '125': '20', '22': '50', '23': '22', '50': '51', '24': '50', '63': '24', '26': '27', '27': '34', '31': '30', '28': '36', '34': '33', '36': '35', '56': '129', '61': '60', '30': '167', '59': '104', '165': '166', '32': '33', '33': '175', '54': '53', '35': '34', '164': '35', '37': '36', '40': '39', '49': '48', '200': '201', '41': '40', '42': '41', '43': '42', '44': '45', '45': '46', '46': '48', '178': '45', '48': '171', '62': '49', '51': '83', '84': '83', '52': '84', '168': '52', '169': '52', '53': '168', '86': '85', '55': '54', '166': '55', '167': '55', '57': '56', '129': '128', '60': '59', '104':

	'103', '170': '62', '174': '110', '64': '162', '67': '66', '65': '64', '162': '157', '160': '177', '66': '69', '163': '67', '71': '69', '172': '85', '173': '86', '175': '90', '92': '91', '97': '98', '99': '98', '101': '100', '103': '169', '126': '105', '128': '105', '118': '106', '107': '121', '202': '203', '110': '109', '122': '121', '116': '117', '158': '206', '154': '156', '159': '205', '206': '205', '204': '205', '176': '177', '171': '174', '179': '197', '187': '188', '191': '192', '193': '192', '199': '200', '72': '71', '74': '71', '70': '72', '73': '72', '75': '78', '76': '74', '85': '73', '81': '75', '77': '76', '82': '81', '83': '81', '78': '77', '79': '80', '80': '82', '87': '172', '88': '87', '90': '87', '89': '90', '93': '92', '94': '93', '95': '94', '96': '97', '194': '193', '100': '99', '105': '103', '127': '126', '119': '120', '108': '107', '120': '107', '109': '108', '203': '204', '111': '110', '112': '113', '117': '113', '113': '114', '114': '122', '115': '114', '124': '130', '123': '122', 'PDKL': '130', '130': '138', '131': 'PDKL', '138': '144', '132': '133', '137': '138', '144': '152', '133': '136', '134': '132', '136': '148', 'BMW': '149', '135': '134', '149': '147', '155': '154', '148': '161', '147': '150', '140': '141', '161': '139', '141': '143', '142': '145', '152': '158', '139': '147', '145': '146', '146': '147', '150': '153', '143': '146', '151': '152', '153': '154', '177': '206', '197': '199', '180': '197', '198': '199', '181': '180', '185': '186', '182': '181', '186': '187', '183': '182', '195': '196', 'VB': '184', '184': '183', '189': '184', '196': '198', '190': '191'}

Lampiran 4: Hasil Pengujian Rute Penelitian

Pengujian Ke-	Hasil
1	Masukkan titik awal: SSB Masukkan titik tujuan: ARI, GWR Rute yang dilalui: [['SSB', '47', '46', '48', '49', '62', '174', '110', '109', '203', '204', '205', '7', '8', '156', 'GWR'], ['GWR', '157', '162', '64', '65', '24', '50', '51', '84', '52', '169', '103', '105', '128', '129', '56', '57', '10', '11', '12', '58', 'ARI']] Total jarak tempuh: 37.405 Km
2	Masukkan titik awal: NOVOTEL Masukkan titik tujuan: RONAL, RS Rute yang dilalui: [['NOVOTEL', '68', '69', '71', '72', '73', '85', '172', '87', '90', '175', '33', '34', '35', '36', '37', '38', 'RS'], ['RS', '38', '28', '27', '26', '25', 'RONAL']] Total jarak tempuh: 14.869999999999997 Km
3	Masukkan titik awal: VB Masukkan titik tujuan: CAI, GWR Rute yang dilalui: [['VB', '184', '183', '182', '181', '180', '185', '186', '187', '188', '5', '4', '2', '1', 'CAI'], ['CAI', '1', '3', '19', '106', '125', '20', '21', '22', '23', '24', '63', '67', '163', '66', '157', '156', 'GWR']] Total jarak tempuh: 34.86 Km
4	Masukkan titik awal: RONAL Masukkan titik tujuan: BMW, CAI Rute yang dilalui: [['RONAL', '14', '58', '12', '11', '29', '56', '57', '10', '9', '3', '1', 'CAI'], ['CAI', '1', '3', '19', '106', '118', '119', '120', '107', '108', '203', '204', '205', '206', '158', '152', '151', '150', '147', '149', 'BMW']] Total jarak tempuh: 38.400000000000006 Km
5	Masukkan titik awal: DENI Masukkan titik tujuan: NOVOTEL, CAI

	Rute yang dilalui: [['DENI', '18', '17', '16', '15', '9', '3', '1', 'CAI'], ['CAI', '1', '3', '19', '106', '125', '20', '21', '22', '23', '24', '63', '67', '163', 'NOVOTEL']] Total jarak tempuh: 21.270000000000003 Km
6	Masukkan titik awal: CAI Masukkan titik tujuan: SSB, PDKL Rute yang dilalui: [['CAI', '1', '2', '4', '5', '6', '39', '40', '41', '42', '43', '44', '45', '46', '47', 'SSB'], ['SSB', '47', '46', '48', '49', '62', '170', '116', '117', '113', '114', '115', '124', '130', 'PDKL']] Total jarak tempuh: 36.64 Km
7	Masukkan titik awal: BMW Masukkan titik tujuan: NOVOTEL, ARI Rute yang dilalui: [['BMW', '149', '147', '150', '153', '156', '157', '162', '66', '67', '163', 'NOVOTEL'], ['NOVOTEL', '68', '69', '71', '74', '76', '77', '78', '75', '81', '83', '84', '52', '169', '103', '105', '128', '129', '56', '57', '10', '11', '12', '58', 'ARI']] Total jarak tempuh: 26.625 Km
8	Masukkan titik awal: RS Masukkan titik tujuan: BMW, VB Rute yang dilalui: [['RS', '38', '37', '36', '35', '164', '175', '90', '89', '88', '85', '73', '72', '70', '68', '69', '66', '157', '156', '154', '153', '150', '147', '149', 'BMW'], ['BMW', '149', '147', '150', '151', '152', '158', '159', '204', '203', '202', '201', '200', '199', '197', '180', '181', '182', '183', '184', 'VB']] Total jarak tempuh: 55.14 Km
9	Masukkan titik awal: PDKL Masukkan titik tujuan: VB, DENI Rute yang dilalui: [['PDKL', '130', '124', '115', '114', '113', '112', '111', '110', '109', '108', '107', '120', '119', '118', '106', '19', '3', '9', '15', '16', '17', '18', 'DENI'], ['DENI', '18', '17', '16', '15', '192', '191', '190', '189', '184', 'VB']] Total jarak tempuh: 45.89 Km

10	Masukkan titik awal: VB Masukkan titik tujuan: RONAL, DENI Rute yang dilalui: [['VB', '184', '189', '190', '191', '192', '15', '16', '17', '18', 'DENI'], ['DENI', '102', '101', '100', '99', '98', '97', '96', '95', '94', '93', '92', '91', '13', '14', 'RONAL']] Total jarak tempuh: 28.09 Km
11	Masukkan titik awal: ARI Masukkan titik tujuan: PDKL, CAI Rute yang dilalui: [['ARI', '58', '12', '11', '29', '56', '57', '10', '9', '3', '1', 'CAI'], ['CAI', '1', '3', '19', '106', '118', '119', '120', '107', '108', '109', '110', '111', '112', '113', '114', '115', '124', '130', 'PDKL']] Total jarak tempuh: 30.384999999999998 Km
12	Masukkan titik awal: GWR Masukkan titik tujuan: DENI, SSB Rute yang dilalui: [['GWR', '157', '162', '64', '65', '24', '50', '51', '84', '52', '169', '103', '105', '128', '129', '56', '57', '10', '9', '15', '16', '17', '18', 'DENI'], ['DENI', '18', '17', '16', '15', '192', '2', '4', '5', '6', '39', '40', '41', '42', '43', '44', '45', '46', '47', 'SSB']] Total jarak tempuh: 42.43 Km
13	Masukkan titik awal: RONAL Masukkan titik tujuan: GWR, RS Rute yang dilalui: [['RONAL', '25', '26', '27', '28', '38', 'RS'], ['RS', '38', '37', '36', '35', '164', '175', '90', '89', '88', '85', '73', '72', '70', '68', '69', '66', '157', '156', 'GWR']] Total jarak tempuh: 18.47 Km
14	Masukkan titik awal: CAI Masukkan titik tujuan: RONAL, ARI, RS Rute yang dilalui: [['CAI', '1', '3', '9', '10', '11', '12', '58', 'ARI'], ['ARI', '58', '14', 'RONAL'], ['RONAL', '25', '26', '27', '28', '38', 'RS']] Total jarak tempuh: 15.409999999999998 Km

15	Masukkan titik awal: SSB Masukkan titik tujuan: NOVOTEL, RONAL, VB Rute yang dilalui: [['SSB', '47', '46', '45', '178', '179', '197', '180', '181', '182', '183', '184', 'VB'], ['VB', '184', '189', '190', '191', '192', '193', '194', '95', '94', '93', '92', '91', '13', '14', 'RONAL'], ['RONAL', '25', '26', '31', '165', '32', '54', '53', '86', '172', '87', '88', '85', '73', '72', '70', '68', 'NOVOTEL']] Total jarak tempuh: 53.58 Km
16	Masukkan titik awal: VB Masukkan titik tujuan: RS, GWR, DENI Rute yang dilalui: [['VB', '184', '189', '190', '191', '192', '15', '16', '17', '18', 'DENI'], ['DENI', '102', '101', '100', '99', '98', '97', '96', '95', '94', '93', '92', '91', '13', '14', 'RONAL', '25', '26', '27', '28', '38', 'RS'], ['RS', '38', '37', '36', '35', '164', '175', '90', '89', '88', '85', '73', '72', '70', '68', '69', '66', '157', '156', 'GWR']] Total jarak tempuh: 46.56 Km
17	Masukkan titik awal: DENI Masukkan titik tujuan: PDKL, ARI, CAI Rute yang dilalui: [['DENI', '18', '17', '16', '15', '9', '3', '1', 'CAI'], ['CAI', '1', '3', '9', '10', '11', '12', '58', 'ARI'], ['ARI', '58', '12', '11', '29', '56', '129', '128', '105', '126', '125', '20', '121', '122', '123', '124', '130', 'PDKL']] Total jarak tempuh: 37.09 Km
18	Masukkan titik awal: RONAL Masukkan titik tujuan: VB, PDKL, GWR Rute yang dilalui: [['RONAL', '25', '26', '31', '165', '32', '54', '53', '86', '172', '87', '88', '85', '73', '72', '70', '68', '69', '66', '157', '156', 'GWR'], ['GWR', '8', '7', '205', '206', '158', '152', '144', '138', '130', 'PDKL'], ['PDKL', '130', '124', '115', '114', '113', '117', '116', '170', '171', '48', '46', '45', '178', '179', '197', '180', '181', '182', '183', '184', 'VB']] Total jarak tempuh: 52.32000000000001 Km

19	Masukkan titik awal: RS Masukkan titik tujuan: PDKL, DENI, BMW Rute yang dilalui: [['RS', '38', '28', '27', '26', '25', 'RONAL', '14', '13', '91', '92', '93', '94', '95', '96', '97', '98', '99', '100', '101', '102', 'DENI'], ['DENI', '18', '17', '16', '15', '9', '3', '19', '106', '118', '119', '120', '107', '108', '109', '110', '111', '112', '113', '114', '115', '124', '130', 'PDKL'], ['PDKL', '131', '132', '134', '135', 'BMW']] Total jarak tempuh: 44.32999999999999 Km
20	Masukkan titik awal: ARI Masukkan titik tujuan: NOVOTEL, SSB, CAI Rute yang dilalui: [['ARI', '58', '12', '11', '29', '56', '57', '10', '9', '3', '1', 'CAI'], ['CAI', '1', '3', '19', '106', '125', '20', '21', '22', '23', '24', '63', '67', '163', 'NOVOTEL'], ['NOVOTEL', '68', '69', '71', '74', '76', '77', '78', '75', '81', '83', '51', '50', '22', '21', '20', '121', '107', '108', '109', '110', '174', '171', '48', '46', '47', 'SSB']] Total jarak tempuh: 48.635000000000005 Km
21	Masukkan titik awal: NOVOTEL Masukkan titik tujuan: BMW, CAI, GWR Rute yang dilalui: [['NOVOTEL', '68', '69', '66', '157', '156', 'GWR'], ['GWR', '156', '154', '153', '150', '147', '149', 'BMW'], ['BMW', '149', '147', '150', '151', '152', '158', '206', '177', '160', '65', '24', '23', '21', '20', '125', '106', '19', '3', '1', 'CAI']] Total jarak tempuh: 40.669999999999995 Km
22	Masukkan titik awal: GWR Masukkan titik tujuan: CAI, RS, SSB Rute yang dilalui: [['GWR', '157', '162', '66', '69', '71', '72', '73', '85', '172', '87', '90', '175', '33', '34', '35', '36', '37', '38', 'RS'], ['RS', '38', '28', '27', '26', '25', 'RONAL', '14', '58', '12', '11', '29', '56', '57', '10', '9', '3', '1', 'CAI'], ['CAI', '1', '2', '4', '5', '6', '39', '40', '41', '42', '43', '44', '45', '46', '47', 'SSB']] Total jarak tempuh: 46.81999999999999 Km

23	Masukkan titik awal: BMW Masukkan titik tujuan: DENI, SSB, ARI Rute yang dilalui: [['BMW', '149', '147', '150', '151', '152', '158', '206', '177', '160', '65', '24', '50', '51', '84', '52', '169', '103', '105', '128', '129', '56', '57', '10', '11', '12', '58', 'ARI'], ['ARI', '58', '12', '13', '91', '92', '93', '94', '95', '96', '97', '98', '99', '100', '101', '102', 'DENI'], ['DENI', '18', '17', '16', '15', '192', '2', '4', '5', '6', '39', '40', '41', '42', '43', '44', '45', '46', '47', 'SSB']] Total jarak tempuh: 54.379999999999995 Km
24	Masukkan titik awal: PDKL Masukkan titik tujuan: ARI, NOVOTEL, VB Rute yang dilalui: [['PDKL', '130', '138', '144', '152', '158', '206', '177', '160', '65', '24', '63', '67', '163', 'NOVOTEL'], ['NOVOTEL', '68', '69', '71', '74', '76', '77', '78', '75', '81', '83', '84', '52', '169', '103', '105', '128', '129', '56', '57', '10', '11', '12', '58', 'ARI'], ['ARI', '58', '12', '13', '91', '92', '93', '94', '95', '194', '193', '192', '191', '190', '189', '184', 'VB']] Total jarak tempuh: 50.86 Km
25	Masukkan titik awal: ARI Masukkan titik tujuan: DENI, RONAL, RS Rute yang dilalui: [['ARI', '58', '14', 'RONAL'], ['RONAL', '25', '26', '27', '28', '38', 'RS'], ['RS', '38', '28', '27', '26', '25', 'RONAL', '14', '13', '91', '92', '93', '94', '95', '96', '97', '98', '99', '100', '101', '102', 'DENI']] Total jarak tempuh: 18.784999999999997 Km
26	Masukkan titik awal: NOVOTEL Masukkan titik tujuan: BMW, GWR, PDKL Rute yang dilalui: [['NOVOTEL', '68', '69', '66', '157', '156', 'GWR'], ['GWR', '8', '7', '205', '206', '158', '152', '144', '138', '130', 'PDKL'], ['PDKL', '131', '132', '134', '135', 'BMW']] Total jarak tempuh: 21.97 Km

27	Masukkan titik awal: PDKL Masukkan titik tujuan: CAI, VB, BMW, RONAL Rute yang dilalui: [['PDKL', '131', '132', '134', '135', 'BMW'], ['BMW', '149', '147', '150', '153', '156', '157', '162', '66', '69', '71', '72', '73', '85', '172', '87', '90', '175', '33', '34', '27', '26', '25', 'RONAL'], ['RONAL', '14', '58', '12', '11', '29', '56', '57', '10', '9', '3', '1', 'CAI'], ['CAI', '1', '2', '4', '5', '188', '187', '186', '185', '180', '181', '182', '183', '184', 'VB']] Total jarak tempuh: 62.019999999999996 Km
28	Masukkan titik awal: BMW Masukkan titik tujuan: DENI, ARI, VB, NOVOTEL Rute yang dilalui: [['BMW', '149', '147', '150', '153', '156', '157', '162', '66', '67', '163', 'NOVOTEL'], ['NOVOTEL', '68', '69', '71', '74', '76', '77', '78', '75', '81', '83', '84', '52', '169', '103', '105', '128', '129', '56', '57', '10', '11', '12', '58', 'ARI'], ['ARI', '58', '12', '13', '91', '92', '93', '94', '95', '96', '97', '98', '99', '100', '101', '102', 'DENI'], ['DENI', '18', '17', '16', '15', '192', '191', '190', '189', '184', 'VB']] Total jarak tempuh: 54.42999999999999 Km
29	Masukkan titik awal: SSB Masukkan titik tujuan: PDKL, RS, BMW, ARI Rute yang dilalui: [['SSB', '47', '46', '48', '49', '62', '170', '116', '117', '113', '114', '115', '124', '130', 'PDKL'], ['PDKL', '131', '132', '134', '135', 'BMW'], ['BMW', '149', '147', '150', '153', '156', '157', '162', '66', '69', '71', '72', '73', '85', '172', '87', '90', '175', '33', '34', '35', '36', '37', '38', 'RS'], ['RS', '38', '28', '27', '26', '25', 'RONAL', '14', '58', 'ARI']] Total jarak tempuh: 56.535000000000004 Km
30	Masukkan titik awal: CAI Masukkan titik tujuan: NOVOTEL, SSB, DENI, GWR Rute yang dilalui: [['CAI', '1', '3', '9', '15', '16', '17', '18', 'DENI'], ['DENI', '18', '17', '16', '15', '9', '10', '11', '29', '61', '30', '31',

	'165', '32', '54', '53', '86', '172', '87', '88', '85', '73', '72', '70', '68', 'NOVOTEL'], ['NOVOTEL', '68', '69', '66', '157', '156', 'GWR'], ['GWR', '8', '7', '205', '204', '203', '202', '109', '110', '174', '171', '48', '46', '47', 'SSB']] Total jarak tempuh: 51.56 Km

RIWAYAT HIDUP

Bella Nafa Savitri atau akrab dipanggil Bella, lahir di Rembang pada tanggal 27 Januari 1999, tinggal di Kabupaten Tangerang, Banten. Anak pertama dari dua bersaudara dari pasangan Bapak Sungkono dan Ibu Afifah Hikmawati, serta merupakan kakak dari Ahmad Dhani Wafiy.

Pendidikan taman kanak-kanak ditempuh di TK Kusuma Bangsa dan lulus pada tahun 2005, dilanjutkan sekolah dasar di SDN Kuta Jaya 2 dan lulus pada tahun 2011, setelah itu melanjutkan ke SMP Permata Insani Islamic School dan lulus pada tahun 2014, kemudian melanjutkan ke jenjang SMA di SMAN 8 Kota Tangerang dan lulus pada tahun 2017, lalu melanjutkan ke jenjang perguruan tinggi di Universitas Islam Negeri Maulana Malik Ibrahim Malang. Selama menempuh pendidikan di Universitas Islam Negeri Maulana Malik Ibrahim Malang, penulis pernah menjadi asisten laboratorium praktikum Pengantar Ilmu Komputer. Selain itu, penulis juga aktif dalam kegiatan kepanitiaan dalam acara Studi Integratif Mahasiswa Matematika pada tahun 2018 dan Pengenalan Budaya dan Akademik Kemahasiswaan yang dilaksanakan oleh Fakultas Sains dan Teknologi pada tahun 2019.

KEMENTRIAN AGAMA RI
UNIVERSITAS ISLAM NEGERI
MAULANA MALIK IBRAHIM MALANG
FAKULTAS SAINS DAN TEKNOLOGI
Jl. Gajayana No. 50 Dinoyo Malang Telp./Fax.(0341)558933

BUKTI KONSULTASI SKRIPSI

Nama : Bella Nafa Savitri
NIM : 17610120
Fakultas/Jurusan : Sains dan Teknologi/Matematika
Judul Skripsi : Implementasi Algoritma *Floyd Warshall* dalam Pencarian Rute Terpendek Lokasi *Tower Base Transceiver Station* (BTS) pada PT Citra Akses Indonusa.
Pembimbing I : Mohammad Nafie Jauhari, M.Si
Pembimbing II : Evawati Alisah, M.Pd

No	Tanggal	Hal	Tanda Tangan
1	12 Februari 2021	Konfirmasi Bimbingan Proposal Skripsi	1
2	5 Mei 2021	Konsultasi Bab I, II, III	2
3	25 Mei 2021	Konsultasi Bab I, II, III	3
4	7 April 2021	Konsultasi Revisi Bab I, II, III	4
5	23 November 2021	Acc Pendaftaran Seminar Proposal	5
6	26 November 2021	Acc Pendaftaran Seminar Proposal	6
7	9 Juni 2022	Bimbingan Revisi Seminar Proposal	7
8	10 Juni 2022	Bimbingan Revisi dan Acc Seminar Hasil	8
9	21 Juni 2022	Bimbingan Revisi Seminar Hasil dan Acc Sidang	9
10	23 Juni 2022	ACC Keseluruhan	10

Malang, 27 Juni 2022
Mengetahui,
Ketua Program Studi Matematika

Dr. Elly Susanti, M.Sc
NIP.19741129 200012 2 005