MODIFIKASI ALGORITMA *A-STAR* PADA OPTIMALISASI RUTE TERPENDEK

SKRIPSI

OLEH:
ACHMAD FAIZ SUKRONI
NIM. 200601110022

PROGRAM STUDI MATEMATIKA
FAKULTAS SAINS DAN TEKNOLOGI
UNIVERSITAS ISLAM NEGERI MAULANA MALIK IBRAHIM
MALANG
2024

MODIFIKASI ALGORITMA *A-STAR* PADA OPTIMALISASI RUTE TERPENDEK

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
Achmad Faiz Sukroni
NIM. 200601110022

PROGRAM STUDI MATEMATIKA
FAKULTAS SAINS DAN TEKNOLOGI
UNIVERSITAS ISLAM NEGERI MAULANA MALIK IBRAHIM MALANG
2024

MODIFIKASI ALGORITMA *A-STAR* PADA OPTIMALISASI RUTE TERPENDEK

SKRIPSI

Oleh
Achmad Faiz Sukroni
NIM. 200601110022

Telah Disetujui untuk Diuji

Malang, 12 Desember 2024

Dosen Pembimbing I	Dosen Pembimbing II
Juhari, M.Si.	Ach. Nasichuddin, M.A
NIPPPK. 19840209 202321 1 010	NIP. 19730705 200003 1 002

Mengetahui,
Ketua Program Studi Matematika

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

MODIFIKASI ALGORITMA *A-STAR* PADA OPTIMALISASI RUTE TERPENDEK

SKRIPSI

Oleh
Achmad Faiz Sukroni
NIM. 200601110022

Telah Dipertahankan di Depan Dewan Penguji Skripsi
dan Dinyatakan Diterima sebagai Salah Satu Persyaratan
untuk Memperoleh Gelar Sarjana Matematika (S.Mat)
Tanggal 26 Desember 2024

Ketua Penguji	**: Dr. Usman Pagalay, M.Si.**	.....
Penguji 1	**: Mohammad Nafie Jauhari, M.Si.**	.....
Penguji 2	**: Juhari, M.Si.**	.....
Penguji 3	**: Ach. Nasichuddin, M.A.**	.....

Mengetahui,
Ketua Program Studi Matematika

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

PERNYATAAN KEASLIAN TULISAN

Saya yang bertanda tangan di bawah ini:

Nama	:	Achmad Faiz Sukroni
NIM	:	200601110022
Program Studi	:	Matematika
Fakultas	:	Sains dan Teknologi
Judul Skripsi	:	Modifikasi Algoritma *A-Star* pada Optimalisasi Rute Terpendek

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 dan pikiran saya, kecuali dengan mencantumkan sumber cuplikan pada daftar pustaka. Apabila dikemudian hari terbukti atau dapat dibuktikan skripsi ini hasil jiplakan, maka saya bersedia menerima sanksi atas perilaku tersebut.

Malang, 26 Desember 2024
Yang membuat pernyataan,

Achmad Faiz Sukroni
NIM. 200601110022

MOTO

“Jaga dan Tata Sholatmu Insyaallah Yang Membuat Seluruh Alam Semesta Beserta Isinya akan Menata dan Menjaga Hidupmu”

PERSEMBAHAN

Dan skripsi ini penulis persembahkan untuk:
Alm. Ibunda saya Hanik Zubaidah, mohon maaf nggih apabila putramu ini masih belum bisa membanggakan dan membahagiakan hati panjenengan, mohon maaf bila putra kecilmu yang lemah ini terdapat banyak salah dan kekurangan nggih, semoga panjenengan diberikan ketenangan, ayem serta tempat ingkang sae nggih buk. Panjenengan tiyang sae, saeee sangett, sedanten remenn kalih panjenengan. Sepindah malih ngapunten lan maturmbahnuwun sanget nggih. Mugi mbenjang ketika pun wancine mugi kulo saget nemui lan sareng kalih panjenengan malih nggih.

Ayahanda Agus Mustofa yang dengan seluruh tenaga mendidik serta merawat putra panjenengan.
Reta Wanda Mardaningrum yang tak henti-hentinya memberikan kasih sayang, membantu dan men-suport dari awal sampai skripsi ini selesei, menarik penulis sehingga bisa menyeleseikan skripsi berikut. Terimakasih karena telah memberi warna kepada penulis.
Bapak Dhur Rachim dan Ibu Sukartini yang selalu memberikan suport serta memberikan kasih sayang.

Dosen-dosen yang telah membantu menyelesaikan skripsi penulis, serta memberikan ilmu. Serta diri saya sendiri yang mau diajak untuk memaksakan diri dan bersusah payah dalam menyelesaikan segala tantangan dan tekanan selama menempuh pendidikan strata satu.

KATA PENGANTAR

Assalamu'alaikum Warahmatullahi Wabarakatuh

Alhamdulillah, segala puji bagi Allah SWT yang Maha Pengasih lagi Maha Penyayang yang mana telah memberikan kita nikmat iman, islam, ihsan, serta nikmat sehat sehingga pada kesempatan kali ini peneliti dapat menyusun skripsi dengan judul "Modifikasi Algoritma *A-Star* pada Optimalisasi Rute Terpendek". Sholawat serta salam semoga tercurahkan pada Baginda nabi agung Muhammad SAW yang mana telah menuntun kita dari zaman kegelapan jahiliyah menuju zaman terang benderang yaitu *Ad-dinul Islam*.

Peneliti menyadari apabila tanpa adanya bantuan dari beberapa pihak, penelitian ni tidak dapat terselesaikan. Maka dari itu peneliti ingin menyampaikan terima kasih yang sebesar-besarnya kepada :

1. Prof. Dr. H. M. Zainuddin, M.A., selaku rektor Universitas Islam Negeri Maulana Malik Ibrahim Malang.
2. Prof. Dr. Hj. Sri Harini, M.Si., selaku dekan Fakultas Sains dan Teknologi Universitas Islam Negeri Maulana Malik Ibrahim Malang.
3. Dr. Elly Susanti, M.Sc, selaku ketua Program Studi Matematika, Universitas Islam Negeri Maulana Malik Ibrahim Malang.
4. Juhari, M.Si, selaku dosen pembimbing 1 yang mana telah memberikan dorongan, nasehat, dan motivasi kepada peneliti.
5. Ach. Nasichuddin, M.A. selaku dosen pembimbing 2 yang telah memberikan masukan, arahan, dan motivasi kepada peneliti.
6. Dr. Usman Pagalay, M.Si. selaku dosen penguji seminar proposal, seminar hasil, dan sidang skripsi yang telah memberikan saran dan kritikan yang bermanfaat bagi penulis.
7. Mohammad Nafie Jauhari, M.Si. selaku dosen penguji seminar proposal, seminar hasil, dan sidang skripsi yang telah memberikan saran serta arahan yang bermanfaat bagi penulis.
8. Seluruh dosen Program Studi Matematika dan seluruh guru yang telah mengajar dan memberikan ilmunya kepada peneliti.
9. Seluruh keluarga dan khususnya kepada Ayahanda Agus Mustofa dan almh. Ibuns Hanik Zubaidah yang tak luput dari doa dan usaha beliau.

10. Reta Wanda Mardaningrum yang selalu memberikan suport, dorongan, bantuan, serta kasih sayang kepada penulis.
11. Seluruh teman angkatan program studi matematika tahun 2020 yang sedang berjuang bersama senantiasa berbagi pengalaman, memberi bantuan, dan dorongan selama menuntut ilmu di masa perkuliahan.
12. Seluruh pihak yang tidak bisa peneliti sebutkan satu persatu, terima kasih atas do'a dan motivasinya dalam penyusunan skripsi ini..

Wassalamu'alaikum Warahmatullahi Wabarakatuh

Malang, 26 Desember 2024

Peneliti

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

DAFTAR LAMPIRAN

ABSTRAK

Sukroni, Achmad Faiz. 2024. **Modifikasi Algoritma *A-Star* pada Optimalisasi Rute Terpendek.** Skripsi. Program Studi Matematika, Fakultas Sains dan Teknologi, Universitas Islam Negeri Maulana Malik Ibrahim Malang. Pembimbing: (I) Juhari, M.Si. (II) Ach Nasichuddin, M.A.

Kata Kunci: *A-Star*, Algoritma Pencarian, Rute Terpendek, Modifikasi Algoritma, Optimisasi.

Algoritma *A-Star* adalah salah satu algoritma pencarian rute terpendek yang paling populer karena keefektifannya dalam menemukan solusi optimal. Penelitian ini berfokus pada modifikasi algoritma *A-Star* untuk meningkatkan performanya dalam mengoptimalkan rute terpendek. Studi dilakukan dengan membandingkan algoritma *A-Star* standar dan versi modifikasinya berdasarkan jumlah iterasi yang diperlukan untuk mencapai solusi optimal. Penilaian dilakukan secara manual dan melalui program komputer dengan menggunakan data sampel lokasi wisata di Blitar sebagai studi kasus. Hasil analisis dalam 20 percobaan peerbandingan menunjukkan bahwa algoritma *A-Star* yang telah dimodifikasi mampu meminimalkan jumlah iterasi hingga 32% dibandingkan algoritma *A-Star* standar. Temuan ini mengindikasikan bahwa modifikasi yang dilakukan pada algoritma *A-Star* dapat meningkatkan efisiensi tanpa mengurangi akurasi hasil. Penelitian ini memberikan kontribusi penting dalam pengembangan algoritma pencarian rute terpendek, khususnya dalam penerapannya pada sektor pariwisata dan sistem navigasi.

ABSTRACT

Sukroni, Achmad Faiz. 2024. **Modification of A-Star Algorithm in Shortest Route Optimization**. Thesis. Mathematics Study Program, Faculty of Science and Technology, State Islamic University of Maulana Malik Ibrahim Malang. Advisor: (I) Juhari, M.Sc. (II) Ach Nasichuddin, M.A.

Keywords: A-Star, Search Algorithm, Shortest Route, Algorithm Modification, Optimization.

The A-Star algorithm is one of the most popular shortest route search algorithms due to its effectiveness in finding optimal solutions. This study focuses on modifying the A-Star algorithm to improve its performance in optimizing the shortest route. The study was conducted by comparing the standard A-Star algorithm and its modified version based on the number of iterations required to reach the optimal solution. The assessment was carried out manually and through a computer program using sample data from tourist locations in Blitar as a case study. The results of the analysis in 20 comparative experiments showed that the modified A-Star algorithm was able to minimize the number of iterations by up to 32% compared to the standard A-Star algorithm. This finding indicates that the modifications made to the A-Star algorithm can increase efficiency without reducing the accuracy of the results. This study provides an important contribution to the development of shortest route search algorithms, especially in their application to the tourism sector and navigation systems.

مستخلص البحث

شكرني، أحمد فائز. ٢٠٢٤. تعديل خوارزمية *A-Star* لتحسين المسار الأقصر. رسالة جامعية. برنامج دراسة الرياضيات، كلية العلوم والتكنولوجيا، جامعة مولانا مالك إبراهيم الإسلامية الحكومية مالانغ. المسرف: (١) جوهري، ماجستير. (٢) احمد ناصح الدين، ماجستير.

الكلمات *المفتاحية* : *A-Star* ، خوارزمية البحث، المسار الأقصر، تعديل الخوارزمية، التحسين.

خوارزمية *A-Star* هي من احدى أكثر خوارزميات البحث عن المسار الأقصر شيوعًا بسبب فعاليتها في إيجاد الحل الأمثل. ركّذت هذه الدراسة على تعديل خوارزمية *A-Star* لتحسين أدائها في تحسين المسار الأقصر. أُجريت الدراسة بمقارنة الخوارزمية القياسية *A-Star* مع نسختها المعدلة بناءً على عدد التكرارات المطلوبة للوصول إلى الحل الأمثل. تم التقييم يدويًا ومن خلال برنامج حاسوبي باستخدام بيانا تاالعينةمن المواقع سياحية في بليتار كدراسة حالة. أظهرت نتائج التحليل في ٢٠ تجربة مقارنة أن الخوارزمية المعدلة *A-Star*تمكنت من تقليل عدد التكرارات بنسبة تصل إلى ٣٢٪ مقارنة بالخوارزمية القياسية *A-Star*. تشير هذه النتائج إلى أن التعديلات التي أُجريت على خوارزمية *A-Star* يمكن أن تحسن الكفاءة دون تقليل دقة النتائج. تقدمت هذه الدراسة مساهمة مهمة في تطوير خوارزميات البحث عن المسار الأقصر، لا سيما في تطبيقاتها في قطاع السياحة وأنظمة الملاحة.

BAB I
PENDAHULUAN

1.1 Latar Belakang

Analisis graf (*graph analysis*) merupakan salah satu cabang dari sekian banyak ilmu matematika dan ilmu komputer yang mempelajari sifat-sifat dan struktur dari graf, serta mengembangkan algoritma dan metode untuk menganalisis dan memecahkan masalah yang melibatkan graf. Analisis graf terbagi pada beberapa bidang seperti analisis konektivitas (bertujuan menentukan apakah titik-titik dalam graf terhubung atau tidak), analisis sentralitas (digunakan untuk mengidentifikasi titik-titik yang paling penting atau berpengaruh dalam suatu graf), analisis klaster (digunakan untuk mengidentifikasi kelompok-kelompok atau klaster titik yang memiliki kemiripan atau kedekatan dalam suatu graf), analisis jalur (analisis yang melibatkan jalur terpendek, jalur terpanjang, atau jalur dengan kriteria tertentu antara dua titik dalam graf), dan beberapa bidang analisis yang lainya. Pada penelitian ini menerapkan analisis jalur (*path analysis*) yang mana dalam konteks optimasi rute analisis ini digunakan untuk menemukan jalur terpendek, jalur terpanjang, atau jalur dengan kriteria tertentu antara dua titik dalam graf. Salah satu metode dalam analisis jalur untuk optimasi rute yaitu menggunakan algoritma *A-Star*.

Algoritma *A-Star* adalah algoritma pencarian jalur (*path finding*) yang dikembangkan pada tahun 1968 oleh Peter Hart, Nils Nilsson, dan Bertram Raphael. Algoritma *A-Star* merupakan pengembangan dari algoritma *Dijkstra* dengan penambahan fungsi heuristik untuk mengoptimalkan proses pencarian. Jika *Dijkstra* menjamin penemuan jalur terpendek namun dengan cara mengeksplorasi

semua kemungkinan node hanya menggunakan $g(n)$ biaya aktual saja, *A-Star* menggunakan estimasi jarak ke tujuan untuk memprioritaskan arah pencarian yang lebih menjanjikan. Pendekatan ini membuat *A-Star* lebih efisien dalam hal waktu komputasi dan penggunaan memori, terutama pada kasus pencarian jalur dalam peta yang luas dan kompleks. Kelebihan ini menjadi dasar pemilihan algoritma *A-Star* sebagai metode pencarian jalur dalam penelitian ini. Algoritma ini didasarkan pada konsep teori graf, di mana masalah pencarian rute dapat direpresentasikan sebagai graf $G = (V, E)$ dengan V sebagai himpunan titik (*node*) dan E sebagai himpunan sisi (*edge*) yang menghubungkan titik-titik tersebut. Setiap sisi memiliki bobot atau biaya yang terkait. Algoritma ini menggunakan fungsi evaluasi $f(n) = g(n) + h(n)$, di mana $g(n)$ adalah biaya aktual dari titik awal ke titik n, dan $h(n)$ adalah perkiraan heuristik biaya termurah dari titik n ke tujuan akhir. Algoritma ini dapat diterapkan pada perencanaan rute navigasi, permainan berbasis peta, robotika, kecerdasan buatan AI, jaringan transportasi, dan sebagainya.

Implementasi algoritma *A-Star* telah dilakukan pada penlitian sebelumnya. Penelitian (Maula et al., 2024) dengan tema "Implementasi Penggunaan Algortima A* pada Penentuan Jarak Terpendek dari Cilacap ke Yogyakarta" berhasil menemukan rute tercepat dari Kota Cilacap ke Kota Yogyakarta dengan jarak 4.371,3 km, dan peneltian ini algoritma *A-Star* lebih efektif dibanding dengan metode yang lainya, dikarenakan telah diuji dengan meggunakan algoritma *searching* seperti *Greedy Best First Search* dengan titik awal dan tujuan yang sama menghasilkan jarak tercepat 4.574,3 km, dengan penghematan jarak oleh algoritma *A-Star* sejauh 203 km. Penelitian selanjutnya (Sumantri & Hidayattullah, 2023) dengan tema "Penerapan Algoritma A*Star Untuk Mencari Rute Terpendek Dari

Kemayoran Ke Destinasi Monumen Nasional (MONAS)" mendapat hasil jarak optimal yakni 15,070 km. Penelitian selanjutnya (Ropiqoh & Lubis, 2023) dengan tema "Implementation Of *A-Star* Algorithm In Finfing The Shortest Route Of Cooking Oil Distribution In Karo Regency Using Graph" dalam pencarian rute terpendek pada distribusi minyak goreng algoritma *A-star* dapat menemukan rute optimal yang mana dibagi menjadi dua rute dengan rute pertama sejauh $11.29\ km$ dan rute kedua sejauh $30.95\ km$. Sedangkan rute perusahaan adalah $30.95\ km$, sehingga presentase penghematan jarak dengan algoritma *A-star* 62.16 %.

Penelitian selanjutnya (Aulia et al., 2023) dengan tema "Perbandingan Algoritma *Dijkstra* dan Algoritma *A-Star* dalam Penentuan Lintasan Terpendek dari Dinas Pendidikan Provinsi Lampung ke Beberapa Sekolah Menengah Atas (SMA) Negeri di Provinsi Lampung" pada 30 titik tujuan didapatkan hasil yang sama dalam penentuan rute terpendek. Walau demikian dari perbandingan waktu proses dengan menggunakan bahasa pemrograman *Python* dari kedua algoritma tersebut, didapat rata-rata *running time* algoritma *Dijkstra* adalah 78, 575 ms dan hasil rata-rata *running time* algoritme *A-Star* adalah 47,505 ms. Dari nilai tersebut, dapat disimpulkan bahwa rata-rata running time algoritma *A-Star* lebih cepat 31,7 ms dibandingkan dengan algoritma *Djikstra.*

Pada penelitian ini peneliti ingin mengembangkan atau memodifikasi algoritma *A-Star* dengan harapan modifikasi algoritma *A-Star* dapat lebih optimal dari algoritma *A-Star* standar pada rute destinasi wisata. Permasalahan pencarian rute terpendek menjadi aspek krusial dalam optimalisasi pengalaman wisatawan, mengingat faktor waktu dan efisiensi perjalanan sangat mempengaruhi kepuasan pengunjung. Dalam konteks ini, penerapan teknologi navigasi yang tepat menjadi

semakin penting, terutama dalam menghadapi dinamika perubahan kondisi lalu lintas dan situasi di lapangan.

Algoritma *A-Star* telah terbukti efektif dalam menyelesaikan permasalahan pencarian rute terpendek karena kemampuannya menggabungkan metode pencarian terarah (heuristik) dengan perhitungan jarak sebenarnya. Namun peneliti melihat celah akan algoritma ini masih bisa dikembangkan dan lebih optimal dengan memodifikasi algoritma *A-Star* dengan mengembangkan fungsi evaluasi pada algoritma *A-Star* tradisional.

Penerapan optimasi rute terpendek merupakan bentuk ikhtiar manusia yang mencerminkan prinsip berhemat yang bisa dikaitkan dengan ajaran dalam agama Islam. Konsep berhemat dengan pemilihan rute terpendek ini mencakup pertimbangan terhadap jarak, energi, dan biaya yang diperlukan. Allah SWT menganjurkan hambanya tentang perencanaan dan pengelolaan suatu hal supaya terwujudnya efisiensi dan optimal, sebagaimana disampaikan dalam Surah Yusuf ayat 47-49 (*Lajnah Pentashihan Mushaf Al-Qur'an*, n.d.):

قَالَ تَزْرَعُوْنَ سَبْعَ سِنِيْنَ دَاَبًا فَمَا حَصَدْتُّمْ فَذَرُوْهُ فِيْ سُنْۢبُلِهٖٓ اِلَّا قَلِيْلًا مِّمَّا تَاْكُلُوْنَ ﴿٤٧﴾ ثُمَّ يَاْتِيْ مِنْۢ بَعْدِ ذٰلِكَ سَبْعٌ شِدَادٌ يَّاْكُلْنَ مَا قَدَّمْتُمْ لَهُنَّ اِلَّا قَلِيْلًا مِّمَّا تُحْصِنُوْنَ ﴿٤٨﴾ ثُمَّ يَاْتِيْ مِنْۢ بَعْدِ ذٰلِكَ عَامٌ فِيْهِ يُغَاثُ النَّاسُ وَفِيْهِ يَعْصِرُوْنَ ﴿٤٩﴾

Artinya: *"Yusuf berkata: Supaya kamu bertanam tujuh tahun (lamanya) sebagaimana biasa; maka apa yang kamu tuai hendaklah kamu biarkan di bulirnya kecuali sedikit untuk kamu makan. Kemudian sesudah itu akan datang tujuh tahun yang sangat sulit, yang menghabiskan apa yang kamu simpan untuk menghadapinya (tahun sulit), kecuali sedikit dari apa yang kamu simpan. Kemudian setelah itu akan datang tahun yang padanya manusia diberi hujan (dengan cukup) dan di masa itu mereka memeras anggur."* (QS. Yusuf [12]: 47-49)

Dalam Tafsir Ibnu Katsir, ayat ini menerangkan tentang pentingnya perencanaan dan pengelolaan sumber daya dengan bijaksana, seperti yang dicontohkan oleh Nabi Yusuf AS dalam mengelola cadangan pangan untuk

menghadapi masa paceklik (Al-Syaikh, 2004). Hal ini menunjukkan penerapan ilmu untuk berhemat dan mengoptimalkan sumber daya yang tepat dapat menghadapi perubahan masa sehingga menjadi hamba yang selamat. Dari pemaparan tersebut dapat disimpulkan bahwa Allah menganjurkan hambanya untuk merencanakan dan mengelola sesuatu dengan efisien dan optimal.

Penelitian ini menerapkan ilmu dari konsep matematika teori graf dan algoritma pencarian jalur yang diaplikasikan pada masalah nyata di bidang pencarian rute terpendek. Dengan mengombinasikan ilmu matematika dan teknologi informasi, bertujuan untuk mengoptimalkan rute terpendek sehingga dapat meningkatkan efisiensi perjalanan dan mengurangi biaya. Dengan memodifikasi algoritma *A-Star* diharapkan dapat diperoleh algoritma *A-Star* yang lebih optimal, dengan membadingkan kedua algoritma tersebut pada penentuan rute terpendek destinasi wisata Blitar.

1.2 Rumusan Masalah

Berdasarkan latar belakang yang telah diuraikan di atas, maka rumusan permasalahan pada penelitian ini yakni:

1. Bagaimana langkah-langkah modifikasi algoritma *A-Star*
2. Bagaimana hasil efektivitas penerapan modifikasi algoritma *A-Star* dalam optimalisasi penentuan rute terpendek?

1.3 Tujuan Penelitian

Berdasarkan rumusan masalah yang telah diuraikan di atas, maka tujuan pada penelitian ini yakni:

1. Menentukan modifikasi algoritma *A-Star*.
2. Menentukan efektifitas penerapan modifikasi algoritma *A-Star* dalam penentuan rute terpendek dengan membandingkan hasil daeri optimalisasi rute.

1.4 Manfaat Penelitian

Diharapkan penelitian ini dapat memberikan konstribusi pengembangan algoritma pencarian rute adaptif, memperkaya literatur tentang penerapan algoritma *A-Star* dalam penentuan rute terpendek, memberikan wawasan baru bagi pembaca bahwasanya Modifikasi algoritma *A-Star* dapat lebih mengoptimalkan dalam pencarian rute, serta menambah khasanah ilmu pengetahuan dalam bidang matematika terapan pada jurusan matematika.

1.5 Batasan Masalah

Dalam penelitian ini, terdapat beberapa Batasan masalah yang harus dipertimbangkan agar ruang lingkup penelitian menjadi jelas dan fokus. Batasan-batasan tersebut antara lain:

1. Lokasi yang menjadi data penelitian adalah destinasi wisata Blitar, yang mana data tersebut sebagai sampel uji perbandingan algoritma *A-Star* tradisional dengan modifikasi algoritma *A-Star*.
2. Pengambilan rute ditentukan pada rute jalan utama menuju titik-titik lokasi destinasi wisata Blitar.
3. Perhitungan dari penelitian ini hanya mengambil bobot jarak, namun untuk waktu diabaikan karena keadaaan kemacetan jalan yang tidak bisa diprediksi.

4. Perhitungan manual diambil tiga rute dengan tiga titik awal dan tiga titik akhir sebagai langkah manual, dan perhitungan program diambil dua puluh .
5. Menggunakan graf berarah, yang mana pada masing masing data memungkinkan adanya jalan khusus satu arah sehingga bobot (jarak) berbeda apabila dari arah titik yang berbeda menuju titik tersebut.
6. Data penelitian yang diolah didapatkan lewat Google Maps dengan mengambil titik koordniat, titik (*node*), dan mengambil bobot (jarak dalam satuan km) pada tiap jalan yang menghubungkan antar titik sampel untuk diuji.

1.6 Definisi Istilah

Graf	:	Struktur data yang dapat digunakan untuk merepresentasikan hubungan antara objek-objek.
Algoritma *A-Star*	:	Algoritma pencarian jalur (*path-finding)* yang digunakan dalam kecerdasan buatan dan ilmu komputer untuk menemukan jalur terpendek atau jalur dengan biaya paling optimal antara dua titik dalam suatu graf.
Node	:	Simpul yang membentuk sebuah graf.
Edge	:	Sisi garis penghubung antara dua titik dalam sebuah graf.
Nilai heuristik	:	Suatu perkiraan atau estimasi yang digunakan untuk memperkirakan biaya atau jarak termurah dalam sebuah titik menuju ke titik tujuan.

Vehicle Routing Problem (VRP)	:	Sebuah permasalahan dalam bidang optimasi yang berkaitan dengan penentuan rute optimal untuk sekelompok kendaraan yang harus menuju lokasi tertentu.
Vertices	:	Titik-titik sudut yang didalam suatu bangun geometri seperti segitiga, persegi, atau polygon lainnya.
Graf *trivial*	:	Graf yang terdiri dari satu titik (*vertices)* tanpa sisi (*edges)* atau dengan satu titik dan sisi yang menghubungkannya sendiri.
Distance	:	Jarak antara dua titik dalam ruang.
Fungsi *distance-plus-cost*	:	Fungsi yang terdiri dari gabungan antara jarak (*distance)* dan biaya (*cost)* yang diperlukan untuk mencapai suatu simpul dalam algoritma *A-Star*.
Google Maps	:	Pelayanan pemetaan *online* yang disediakan oleh *google* untuk pengguna dalam mencari lokasi, petunjuk arah, bisnis dan layanan lokal, tampilan peta, tampilan satelit, dan tampilan jalan.
Greedy Best First Search	:	Merupakan algoritma pencarian dengan menggunakan fungsi heuristik untuk memilih titik berikutnya yang akan dieksplorasi, selalu memilih titik yang tampak paling menjanjikan untuk mencapai tujuan.

Sisi gelang atau sisi ganda	:	Sisi yang menghubungkan suatu simpul dengan dirinya sendiri, dengan kata lain sisi yang berawal dan berakhir pada simpul yang sama.
Fungsi nilai heuristik	:	Fungsi yang memperkirakan biaya atau jarak dari suatu titik ke titik tujuan dalam masalah pencarian.
Antrian Prioritas (*Priority queue*)	:	Struktur data yang menyimpan elemen-elemen dengan prioritas terkait, di mana elemen dengan prioritas tertinggi (atau terendah, tergantung implementasi) selalu diakses terlebih dahulu.
Cost	:	Harga (bisa meliputi jarak).
Parent node	:	Titik yang berada satu tingkat di atas titik lain dan terhubung langsung. Titik ini yang menjadi acuan atau induk bagi titik-titik di bawahnya.
Path finding	:	proses pencarian jalur optimal atau efisien dari satu titik awal ke titik tujuan di dalam suatu ruang atau graf.
Path	:	Dalam konteks ilmu komputer dan teori graf adalah serangkaian simpul (node) yang terhubung oleh sisi (edge) yang menunjukkan hubungan atau koneksi antara simpul-simpul tersebut. Path merepresentasikan jalur yang diambil dari satu simpul ke simpul lainnya dalam sebuah graf, baik itu graf berbobot maupun graf tak berbobot.

Staring point	:	*node* awal dimulainya rute pencarian.
Goal point	:	*node* tujuan dalam pencarian rute terpendek.
Current node	:	Dalam konteks algoritma pencarian jalur (*pathfinding*), seperti *A-star* atau *Dijkstra*, merujuk pada titik yang sedang diproses pada iterasi saat ini oleh algoritma. Ini adalah titik yang saat ini dievaluasi untuk menemukan tetangga-tetangga (neighbor) atau titik-titik yang terhubung, guna memperluas pencarian jalur terbaik ke tujuan
Backtracking	:	Teknik algoritmik untuk memecahkan masalah secara rekursif dengan mencoba membangun solusi secara bertahap.

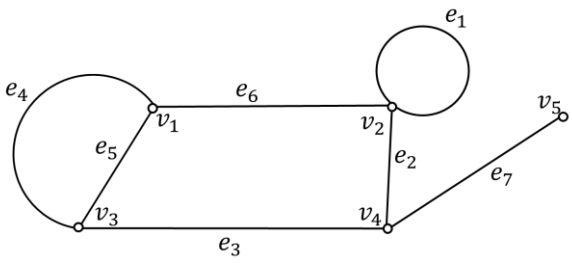
BAB II
KAJIAN TEORI

2.1 Definisi Graf

Sebuah graf linier (atau secara sederhana disebut graf) $G = (V, E)$ adalah suatu sistem yang terdiri atas suatu himpunan objek $V = \{v_1, v_2, \dots\}$ yang disebut himpunan titik, dan $E = \{e_1, e_2, \dots\}$ yang merupakan himpunan sisi sedemikian hingga tiap sisi e_k dikaitkan dengan suatu pasangan tak-terurut (v_i, v_j). Titik v_i, v_j yang berkaitan dengan e_k, disebut titik-titik ujung sisi e_k.

V tidak boleh kosong, sedangkan E boleh kosong. Jadi sebuah graf dimungkinkan tidak mempunyai sisi satu buah pun, tetapi simpulnya harus ada, minimal satu. Graf yang hanya mempunyai satu buah simpul tanpa sebuah sisi dinamakan graf trivial.

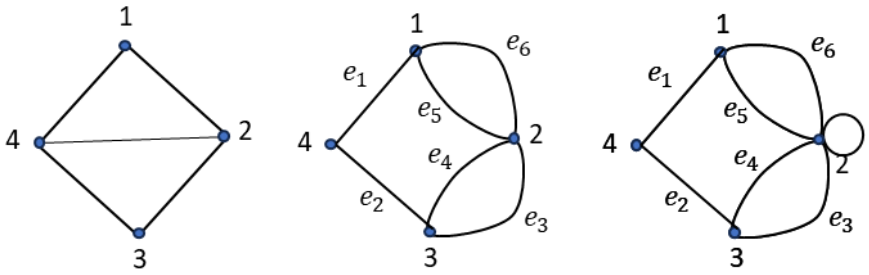
Cara merepresentasikan sebuah graf yang paling umum adalah dengan diagram. Dalam diagram tersebut, titik-titik dinyatakan sebagai noktah dan tiap sisi dinyatakan sebagai kurva sederhana yang menghubungkan tiap dua titik. Untuk lebih jelasnya perhatikan contoh graf pada gambar di bawah ini.



Gambar 2.1 Contoh Graf

Dalam sebuah graf, seperti terlihat pada contoh diatas, dimungkinkan adanya suatu sisi yang dikaitkan dengan pasangan (v_2, v_2). Sisi yang dua titik ujungnya

sama disebut *loop*/gelang. Dalam graf pada gambar e_1, merupakan sebuaa $loop$. Dari contoh diatas juga perlu dicatat bahwa dalam graf dimungkinkna adanya lebih dari satu sisi yang dikaitkan dengan sepasang titik. Sebagai contoh e_4, dan e_5 pada graf di atas dikaitkan dengan pasangan titik (v_1, v_3). Pasangan sisi semacam ini disebut sisi-sisi paralel/ sejajar atau sisi rangkap.



Gambar 2.2 $G_1, G_2,$ dan G_3

Gambar di atas memperlihatkan tiga buah graf G_1, G_2, dan G_3. G_1 adalah graf dengan himpunan simpul V dan himpunan sisi E adalah:

$$V = \{1,2,3,4\}$$

$$E = \{(1,2),(1,3),(2,3),(2,4),(3,4)\}$$

G_2 adalah graf dengan himpunan simpul V dan himpunan sisi E adalah:

$$V = \{1,2,3,4)$$

$E = \{(1,2),(2,3),(1,3),(1,3),(2,4),(3,4),(3,4)\}$ himpunan ganda

$$= \{e_1, e_2, e_3, e_4, e_5, e_6, e_7\}$$

G_3 adalah graf dengan himpunan simpul V dan himpunan sisi E adalah:

$$V = (1,2,3,4)$$

$E = \{(1,2),(2,3),(1,3),(1,3),(2,4),(3,4),(3,4),(3,3)\}$ himpunan ganda

$$= \{e_1, e_2, e_3, e_4, e_5, e_6, e_7, e_8\}$$

Pada G_2, sisi $e_6 = (1,6)$ dan sisi $e_5 = (1,6)$ dinamakan sisi ganda (multiple *edges* atau paralel *edges*) karena kedua sisi ini menghubungi dua buah simpul yang sama, yaitu simpul 1 dan simpul 3. Pada G_3, sisi $e_8 = (2,2)$ dinamakan gelang atau kalang (*loop*) karena ia berawal dan berakhir pada simpul yang sama (Juhari, 2021).

2.2 Jenis-jenis Graf

Graf dapat dikelompokan menjadi beberapa kategori (jenis) bergantung pada sudut pandang pengelompokan graf dapat dipandang berdasarkan ada tidaknya sisi ganda atau sisi kalang, berdasarkan jumlah simpul, atau berdasarkan orientasi arah pada sisi. Berdasarkan ada tidaknya gelang atau sisi ganda pada suatu graf, maka secara umum graf dapat digolongkan menjadi dua jenis (Juhari, 2021):

2.2.1 Graf Sederhana

Graf yang tak mempunyai gelang atau sisi ganda disebut sebagai graf sederhana. Dalam graf sederhana, setiap sisi direpresentasikan sebagai sebuah pasangan yang tak terurut. Dengan kata lain, sisi (u, v) sama dengan sisi (v, u). Kita bisa mendefinisikan suatu graf sederhana $G = (V, E)$ sebagai kumpulan titik yang tak kosong dan kumpulan sisi-sisi yang berbeda, yang masing-masing merupakan pasangan tak terurut.

2.2.2 Graf Tak Sederhana

Graf yang mengandung sisi ganda atau gelang dinamakan graf tak sederhana (*unsimple graph*). Ada dua macam graf tak sederhana, yaitu graf ganda (*multigraph*) dan graf semu (*pseudograph*). Graf ganda adalah graf yang mengandung sisi ganda. Sisi ganda yang menghubungkan sepasang simpul bisa

lebih dari dua buah. Sisi ganda dapat diasosisiasikan sebagai pasangan tak-terurut yang sama. Kita dapat juga mendefinisikan graf ganda $G = (V, E)$ terdiri dari himpunan tidak kosong simpul-simpul dan E adalah himpunan ganda (*multiset*) yang mengandung sisi ganda.

Graf semu adalah graf yang mengandung gelang (*loop*). Graf semu lebih umum daripada graf ganda, karena sisi pada graf semu dapat terhubung ke dirinya sendiri. Jumlah simpul pada graf kita sebut sebagai kardinalitas graf, dan dinyatakan dengan $n = |V|$, dan jumlah sisi kita nyatakan dengan $m = |E|$. Sisi pada graf dapat mempunyai orientasi arah. Berdasarkan orientasi arah pada sisi, maka secara umum graf dibedakan atas 2 jenis:

a. Graf Tak Berarah (*Undirected Graph*)

Graf yang sisinya tidak mempunyai orientasi arah disebut graf tak-berarah. Pada graf tak-berarah, urutan pasangan simpul yang dihubungkan oleh sisi tidak diperhatikan. Jadi $(u, v) = (v, u)$ adalah sisi yang sama.

b. Graf Berarah (*Directed Graph* atau *Digraph*)

Graf yang setiap sisinya diberikan orientasi arah disebut sebagai graf berarah. Kita lebih suka menyebut sisi berarah dengan sebutan busur (*arc*). Pada graf berarah (u, v) dan (v, u) menyatakan dua buah busur yang berbeda, dengan kata lain$(u, v) \neq (v, u)$. Untuk busur (u, v) simpul u dinamakan simpul asal dan simpul v dinamakan simpul terminal. Graf berarah sering dipakai untuk menggambarkan aliran proses, peta lalu lintas suatu kota (jalan searah atau dua arah), dan sebagainya. Pada graf berarah, gelang diperbolehkan, tetapi sisi ganda tidak.

Definisi graf dapat diperluas sehingga mencakup graf ganda berarah (*directed multigraph*). Pada graf ganda berarah, gelang dan sisi ganda diperbolehkan ada. Tabel di bawah akan meringkas perluasan definisi graf.

Tabel 2.1 Ringkasan Definisi Graf

Jenis	Sisi	Sisi ganda diperbolehkan?	Sisi gelang diperbolehkan?
Graf sederhana	Tak berarah	Tidak	Tidak
Graf ganda	Tak berarah	Ya	Tidak
Graf semu	Tak berarah	Ya	Ya
Graf berarah	Berarah	Tidak	Ya
Graf tak berarah	Tak berarah	Ya	Ya

2.3 Algoritma

Algoritma adalah serangkaian langkah-langkah logis dan terstruktur yang dirancang untuk menyelesaikan suatu masalah atau mencapai tujuan tertentu dalam waktu yang terbatas. Menurut (Cormen et al., 2022) algoritma didefinisikan sebagai "urutan langkah-langkah yang terdefinisi dengan baik untuk menyelesaikan suatu masalah komputasi." Sebuah algoritma harus memiliki lima karakteristik utama: *finiteness* (berakhir dalam langkah tertentu), *definiteness* (setiap langkah jelas), *input, output*, dan *effectiveness* (dapat dijalankan secara praktis).

Dalam dunia komputasi, algoritma sering digunakan untuk memproses data, melakukan perhitungan, dan menyelesaikan masalah yang kompleks secara efisien. Algoritma dapat diterapkan pada berbagai bidang, termasuk pencarian, pengurutan, pengoptimalan, dan perencanaan. Algoritma dapat diklasifikasikan berdasarkan pendekatan atau metode yang digunakan. Berikut adalah jenis-jenis algoritma beserta penjelasannya:

1. Algoritma Pencarian (*Search Algorithms*). Dirancang untuk menemukan elemen tertentu dalam kumpulan data atau menemukan jalur optimal dalam graf. Contoh: *Binary Search*, *Depth First Search* (DFS), *Breadth First Search* (BFS), *A-Star*.
2. Algoritma Pengurutan (*Sorting Algorithms*). Digunakan untuk mengatur elemen dalam urutan tertentu, seperti *ascending* atau *descending*. Contoh: *Bubble Sort, Merge Sort, Quick Sort, Heap Sort.*
3. Algoritma *Divide* and *Conquer*. Memecah masalah menjadi submasalah yang lebih kecil, menyelesaikannya secara terpisah, dan menggabungkan hasilnya. Contoh: *Merge Sort, Binary Search, Quick Sort*.
4. Algoritma Dinamika (*Dynamic Programming*). Menyelesaikan masalah dengan memecahnya menjadi submasalah yang saling tumpang tindih dan menyimpan hasil submasalah untuk menghindari perhitungan ulang. Contoh: *Knapsack Problem, Floyd-Warshall Algorithm, Longest Common Subsequence.*
5. Algoritma *Greedy.* Memilih langkah terbaik pada setiap tahap dengan harapan mendapatkan solusi optimal secara keseluruhan. Contoh: *Prim's Algorithm, Kruskal's Algorithm, Huffman Coding*.
6. Algoritma Heuristik. Menggunakan pendekatan yang mendekati solusi terbaik dengan waktu yang lebih singkat, biasanya untuk masalah kompleks atau NP-*Hard*. Contoh: *Simulated Annealing, Genetic Algorithm, A-Star*.
7. Algoritma *Backtracking*. Menghasilkan semua kemungkinan solusi secara rekursif dengan menghapus solusi yang tidak memenuhi syarat. Contoh: *N-Queens Problem, Sudoku Solver, Traveling Salesman Problem* (TSP).

8. Algoritma *Brute Force*. Mencoba semua kemungkinan solusi hingga menemukan solusi yang benar. Contoh: *Exhaustive Search, String Matching (Naive Approach)*.

Algoritma *A-Star* adalah algoritma pencarian yang menggunakan pendekatan heuristik untuk menemukan jalur terpendek dalam graf berbobot. Berdasarkan jenis-jenis algoritma, *A-Star* dapat diklasifikasikan sebagai berikut:

1. Algoritma Pencarian Jalur Terpendek (Shortest Path Search Algorithm). *A-Star* dirancang untuk mencari jalur dengan total biaya minimum antara simpul awal ($node_{start}$) dan simpul tujuan ($node_{goal}$).
2. Algoritma Heuristik. *A*-Star menggunakan fungsi heuristik $h(n)$ untuk memperkirakan biaya dari simpul saat ini ke tujuan. Kombinasi heuristik dengan biaya aktual $g(n)$ membuatnya efisien dalam pencarian solusi.
3. Algoritma Informasi Lengkap (*Informed Search Algorithm*). Karena *A-Star* menggunakan informasi tambahan berupa fungsi heuristik, algoritma ini termasuk kategori pencarian dengan informasi lengkap.
4. Algoritma Optimal dengan *Admissible Heuristic*. A-Star dijamin memberikan solusi optimal jika heuristik $h(n)$ memenuhi sifat:
   a. **Admissible**: Tidak pernah melebihi biaya sebenarnya ke tujuan ($h(n) \leq h'(n)$)
   b. **Consistent**: Nilai heuristik antar simpul tidak mengalami lompatan negatif.

2.4 Lintasan Terpendek

Mencari jalur terpendek dalam sebuah graf adalah salah satu persoalan optimasi yang umum. Graf yang digunakan untuk mencari jalur terpendek biasanya berbobot, artinya setiap sisi graf memiliki nilai atau bobot tertentu. Bobot tersebut dapat mewakili jarak antar kota, waktu pengiriman barang, biaya pembangunan, dan faktor lainnya.

Penting untuk dicatat bahwa istilah terpendek tidak selalu dapat diinterpretasikan dengan jelas sebagai jarak minimum, karena hal itu dapat bervariasi tergantung pada konteks persoalan yang sedang dibahas. Namun, secara umum, "terpendek" berarti meminimalkan total bobot atau nilai pada suatu jalur dalam graf (Munir, 2023). Terdapat dua jenis dalam pencarian jalur yakni *blind search* (*uniform search*) dan *heuristik search* (*informed search*) (Russel & Norvig, 2010).

2.5 Informed Search

Merupakan algoritma pencarian dimana terdapat informasi tambahan dengan menggunakan fungsi heuristik untuk mencapai goal state yang tentunya akan sangat membantu dalam proses pencarian secara efisien. Contoh algoritma yakni yaitu *Best-First Search* (BFS). BFS Merupakan sebuah algoritma yang menggunakan fungsi evaluasi untuk memilih titik yang terbaik (tercepat) untuk dieksplorasi selanjutnya. Terdapat dua algoritma dalam BFS yakni *Greedy Best-First Search* dan algoritma *A-Star*.

Greedy Best-First Search merupakan algoritma yang selalu memilih titik yang tampaknya paling dekat dengan tujuan berdasarkan fungsi heuristik. Disini

Greedy BFS hanya memakai fungsi $h(n)$ saja dan dapat menemukan jalur tercepat akan tetapi dapat memakan proses yang lebih lama dari *A-Star* (Russel & Norvig, 2010)

2.6 Prinsip Dasar Algoritma *A-Star* Tradisional

Pada tahun 1968, Peter Hart, Nils Nilsson, dan Bertram Raphael memperkenalkan algoritma A* Algoritma yang biasa dikenal sebagai "*A-star*" adalah salah satu algoritma pencarian graf yang paling unggul yang dapat menemukan jalur dengan biaya terendah dari suatu titik awal ke titik tujuan yang diinginkan. Algoritma *A-star*, sebagai algoritma perencanaan jalur yang umum digunakan, telah banyak digunakan diterapkan di berbagai bidang seperti navigasi robot dan pencarian rute terpendek. Algoritma *A-star* adalah algoritma heuristik yang sangat meningkatkan efisiensi pencarian jalur dengan memasukkan informasi tentang titik target sambil mempertahankan pencarian jalur optimal. Fungsi evaluasi adalah sebagai berikut (Yudha et al., 2022):

$$f(n) = g(n) + h(n) \quad (2.1)$$

Keterangan:

$f(n)$: fungsi evaluasi titik n pada setiap simpul

$g(n)$: Akumulatif biaya aktual dari titik awal hingga titik n

$h(n)$: perkiraan biaya dari titik n ke titik tujuan, juga dikenal sebagai fungsi heuristik.

$h(n)$ ditentukan dengan melakukan perhitungan dengan rumus *Heuristic Euclidean Distance* (HED):

$$h(n) = \sqrt{\left(x_{goal} - x_n\right)^2 + \left(y_{goal} - y_n\right)^2} \quad (2.2)$$

Keterangan:

x_n : Nilai bujur/ longitude titik n

x_{goal} : Nilai bujur/ longitude titik tujuan

y_n : Nilai lintang/ latitude titik n

y_{goal} : Nilai lintang/ latitude titik tujuan

Untuk fungsi $g(n)$ sebagai berikut:

$$g(n) = e(node_{start}, node_A) + e(node_A, node_n) \quad (2.3)$$

Keterangan:

$node_{start}$: Titik awal

$node_A$: Titik yang sedang dijalankan

$node_n$: Titik n

$e(node_{start}, node_A)$: Jarak aktual dari titik awal menuju titik yang sedang dijalankan (dieksplorasi)

$e(node_A, node_n)$: Jarak aktual dari titik yang sedang dijalankan menuju titik n.

Beberapa terminologi dasar yang terdapat pada algoritma *A-Star* adalah:

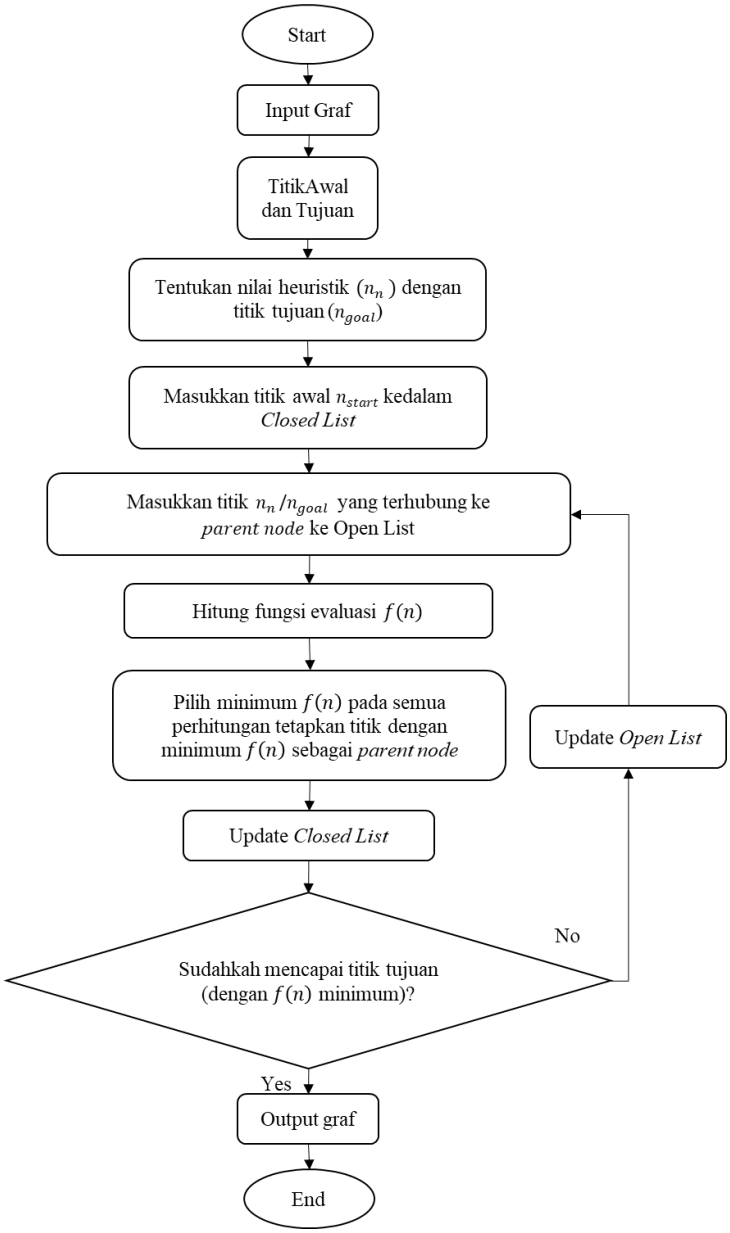
1. Starting point adalah sebuah terminologi untuk posisi awal sebuah titik.
2. A adalah simpul yang sedang dijalankan dalam algortima pencarian rute terpendek.
3. Titik (*node*) adalah petak-petak kecil sebagai simbol dari daerah pathfinding.

4. *Parent node* adalah titik yang berada satu tingkat di atas titik lain dan terhubung langsung. Titik ini yang menjadi acuan atau induk bagi titik-titik di bawahnya
5. *Open list* adalah tempat penyimpanan data titik yang dapat diakses dari titik awal maupun titik yang sedang dijalankan. Contoh: *Open_list* = $\{n_1, n_2, n_3\}$
6. *Closed list* adalah tempat penyimpanan data titik ($node$) sampai titik A yang merupakan bagian dari jalur terpendek. Contoh:

   Closed_list $== \{n_1, n_2, n_3, n_A\}$

   Keterangan :
7. Harga (F) adalah nilai yang diperoleh dari proses penjumlahan, nilai g merupakan jumlah nilai tiap titik dalam jalur terpendek dari titik awal ke A, dan H adalah jumlah perkiraan nilai dari sebuah simpul ke simpul tujuan.
8. Titik tujuan yaitu simpul yang menjadi tempat akhir perjalanan.

   Berikut flowchart Algoritma *A-star*:



Gambar 2.3 Flowchart Algoritma *A-star*

Langkah-langkah algoritma *A-star* adalah sebagai berikut:

1. Mulai pencarian dari titik awal n_{start}, dan tambahkan n_{start} ke "*open_list*" sebagai titik (*node*) yang akan dicari.
2. Cari semua titik n yang terhubung dengan titik awal n_{start}, dan tambahkan ke "*open_list*". Tetapkan titik awal n_{start} sebagai *parent node* dari titik-titik ini.

3. Hapus titik awal dari "*open_list*", dan tambahkan ke "*closed_list*".
4. Hitung fungsi evaluasi $f(n)$ untuk setiap titik dalam "$open\_list$" saat ini, kemudian pilih titik n_1 dengan nilai $f(n)$ terkecil. Hapus n_1 dari "*open_list*", tambahkan n_1 ke "*closed_list*", dan kosongkan "*open_list*".
5. Cari semua titik yang terhubung pada titik n_1 dan masukkan ke "$open\_list$". Jika titik-titik tersebut merupakan hambatan atau titik dalam "*closed_list*", mereka tidak dipertimbangkan. Tetapkan n_1 sebagai *parent node* dari titik-titik yang terdapat pada "*open_list*", hitung fungsi evaluasi $f(n)$, kemudian pilih titik n_2 dengan nilai fungsi evaluasi terkecil. Jika pada perhitungan pertama dari titik awal atau dari perhitungan sebelum-sebelumnya terdapat nilai fungi evaluasi yang lebih kecil (n_3) maka:
   a. Kita pilih nilai fungsi evaluasi pada titik n_3.
   b. Kita *update* atau kita kosongkan "*closed_list*".
   c. mengisi "*closed_list*" dengan titik n_3 (titik n dengan fungsi evaluasi terkecil dari perhitungan yang lain) dan titik-titik terpilih sebelum titik n_3.
   d. Tetapkan titik n_3 sebagai *parent node*.
   e. Cari semua titik yang terhubung dengan titik n_3, kita masukkan ke "*open_list*", kita hitung nilai fungsi evaluasi pada titik-titik yang terdapat pada "*open_list*", kita pilih titik n dengan fungsi evaluasi terkecil, dan kita masukkan titik n dengan fungsi evaluasi terkecil pada "*closed_list*".
6. Lanjutkan perhitungan dengan memperhatikan nilai fungsi evaluasi $f(n)$ pada setiap perhitungan selanjutnya, jika terdapat nilai fungsi evaluasi lebih minimal pada perhitungan sebelumnya, maka kita *update* dan melanjutkan perhitungan pada titik dengan nilai fungsi evaluasi terkecil. Perhitungan

berhenti apabila titik terpilih dengan nilai fungsi $f(n)$ terkecil yakni titik tujuan (n_{goal}).

2.7 *Jupyter Notebook*

Jupyter Notebook adalah sebuah lingkungan interaktif berbasis web yang digunakan untuk menulis dan menjalankan kode program, membuat visualisasi data, serta mendokumentasikan proses analisis secara terpadu. Platform ini mendukung berbagai bahasa pemrograman melalui kernel, dengan *Python* sebagai bahasa utama yang paling populer (Kluyver et al., 2016). Nama "*Jupyter*" berasal dari kombinasi *Julia*, *Python*, dan *R*, yaitu tiga bahasa pemrograman yang didukungnya pada awal pengembangan. Saat ini, *Jupyter Notebook* telah menjadi alat utama dalam pengolahan data dan pengembangan berbasis penelitian ilmiah serta pendidikan.

Jupyter Notebook memiliki fitur utama yang mendukung aktivitas ilmiah, seperti antarmuka interaktif, kemampuan visualisasi data, dan dukungan untuk berbagai format dokumentasi. *Notebook* ini memungkinkan pengguna menulis kode, menjalankan program, dan melihat hasil secara langsung dalam satu platform. Selain itu, fitur dokumentasi yang mendukung *Markdown* dan *LaTeX* memberikan fleksibilitas untuk membuat laporan yang terstruktur (Shen, 2014). Kemampuan untuk mengekspor dokumen ke berbagai format, seperti HTML dan PDF, semakin meningkatkan fungsionalitasnya.

Secara teknis, *Jupyter Notebook* terdiri dari tiga komponen utama: *notebook server*, kernel, dan antarmuka pengguna. *Notebook server* berfungsi sebagai *backend* untuk mengelola file notebook dan menangani komunikasi antara kernel

dan antarmuka. Kernel adalah mesin eksekusi yang menjalankan kode, dengan IPython sebagai kernel yang paling umum digunakan. Sementara itu, antarmuka pengguna berbasis web memungkinkan pengguna untuk menulis dan menjalankan kode, serta melihat visualisasi hasil analisis (Pérez & Granger, 2007).

Dalam dunia pendidikan, *Jupyter Notebook* sering digunakan untuk mengajarkan konsep pemrograman, analisis data, dan pembelajaran mesin. Selain itu, notebook ini juga populer di kalangan peneliti yang membutuhkan platform untuk dokumentasi eksperimen yang dapat direproduksi. Keunggulan ini menjadikan Jupyter sebagai alat penting untuk kolaborasi dalam tim riset maupun komunitas terbuka (Kluyver et al., 2016). Namun, Jupyter Notebook memiliki beberapa keterbatasan, seperti kurangnya dukungan untuk pengelolaan versi secara langsung dan kinerja yang kurang optimal untuk aplikasi berskala besar.

2.8 Perilaku Hemat dalam Al-Qur’an dan Hadits

Islam sebagai agama yang mengajarkan rahmat bagi seluruh alam (*rahmatan lil alamin*), memberikan pedoman tentang perilaku yang baik dalam kehidupan manusia, termasuk dalam hal berhemat. Al-Qur'an dan Hadits merupakan sumber pedoman utama dalam menjalani kehidupan sesuai dengan ajaran Islam. Meneguhkan komitmen terhadap Al-Qur'an dan Hadis adalah suatu kewajiban yang memiliki dampak besar dalam perilaku dan penyelesaian masalah kita. Al-Quran Surah Al-Furqan ayat 67 (Kemenag, 2023):

وَالَّذِينَ إِذَا أَنفَقُوا لَمْ يُسْرِفُوا وَلَمْ يَقْتُرُوا وَكَانَ بَيْنَ ذَٰلِكَ قَوَامًا

Artinya: "*Dan orang-orang yang apabila membelanjakan (harta), mereka tidak berlebihan, dan tidak (pula) kikir, dan adalah (pembelanjaan itu) di tengah-tengah antara yang demikian*". (QS. Al-Furqan [25]: 67)

Ayat ini menekankan prinsip keseimbangan dalam menggunakan sumber daya, tidak berlebihan dan tidak pula terlalu kikir. Dengan menerapkan ilmu pengetahuan, kita dapat mengelola sumber daya secara optimal sehingga tidak terjadi pemborosan maupun kekurangan (Tarigan & M.Ag, 2012).

Dari Jabir bin Abdullah radhiallahu 'anhu, Rasulullah shallallahu 'alaihi wa sallam bersabda:

عَنْ عَائِشَةَ رَضِيَ اللهُ عَنْهَا قَالَتْ: قَالَ رَسُوْلُ اللهِ صَلَّى اللهُ عَلَيْهِ وَسَلَّمَ: إِنّ اللَّهَ تَعَالى يُحِبّ إِذَا عَمِلَ أَحَدُكُمْ عَمَلاً أَنْ يُتْقِنَهُ (رواه الطبرني والبيهقي)

Artinya*:"Dari Aisyah ra, sesungguhnya Rasulullah SAW bersabda: Sesungguhnya Allah menyukai jika salah seorang di antara kamu mengerjakan suatu pekerjaan, ia melakukannya dengan itqan (tepat, terarah, jelas, dan tuntas)". (HR. Thabrani)*

Dalam hadis ini, Rasulullah SAW menjelaskan bahwa Allah SWT sangat mencintai orang yang melaksanakan pekerjaannya dengan itqan. Istilah "*itqan*" di sini berarti melakukan pekerjaan dengan kesungguhan, ketekunan, dan usaha maksimal, atau bisa disebut sebagai sikap profesional. Seseorang yang bekerja dengan *itqan* adalah mereka yang tidak sekadar menyelesaikan tugas, tetapi juga berkomitmen untuk memberikan hasil yang terbaik dan mencapai standar kualitas yang diharapkan (KEMENAG, 2015).

Hadits Riwayat Muslim:

إِنَّ اللَّهَ كَتَبَ الْإِحْسَانَ عَلَى كُلِّ شَيْءٍ فَإِذَا قَتَلْتُمْ فَأَحْسِنُوا الْقِتْلَةَ وَإِذَا ذَبَحْتُمْ فَأَحْسِنُوا الذِّبْحَةَ وَلْيُحِدَّ أَحَدُكُمْ شَفْرَتَهُ وَلْيُرِحْ ذَبِيحَتَهُ

Artinya: *"Sesungguhnya Allah mewajibkan untuk berbuat baik dalam segala hal. Apabila kamu membunuh, maka bunuhlah dengan cara yang baik. Apabila kamu menyembelih, maka sembelihlah dengan cara yang baik. Hendaklah seseorang dari kamu mengasah pisaunya dan mempermudah penyembelihan binatangnya." (HR. Muslim)*

Syarah Arba'in menguraikan beberapa hikmah dari hadits ini yaitu Allah Yang Maha Pemurah memerintahkan hamba-Nya untuk berbuat baik dalam segala hal.

Perintah Allah ini menjadikan ihsan sebagai kewajiban yang mencakup seluruh aspek kehidupan. Bahkan dalam hal-hal yang diizinkan untuk diakhiri, seperti menyembelih hewan, kita harus melakukannya dengan cara terbaik dan paling manusiawi. Otoritas tertinggi dalam segala perintah dan hukum ada pada Allah. Konsep ihsan bersifat *universal*, berlaku untuk semua tindakan. Metode pengajaran Nabi yang menggunakan contoh konkret memudahkan pemahaman umatnya. Khusus dalam konteks penyembelihan, kita diwajibkan melakukannya dengan penuh kebaikan dan sesuai syariat. Penyembelihan sebaiknya dilakukan oleh orang yang ahli, menggunakan alat yang tajam untuk meminimalkan penderitaan hewan. Hadits ini mengajarkan bahwa hewan yang akan disembelih harus diperlakukan dengan lembut, seolah-olah sedang diistirahatkan, bukan disakiti (Al-Utsaimin, 2013).

Kesimpulan yang dapat diambil dari ayat Alquran dan Hadist diatas, sebagai muslim yang baik kita dianjurkan untuk memilih metode yang baik dalam hal aspek apapun dan anjuran melakukan suatu pekerjaan dengan *itqan* (profesional). Seperti menyembelih hewan dengan cara yang baik yang dapat diambil garis lurus menunjukkan pentingnya memakai metode yang baik sehingga menghasilkan suatu hasil yang optimal (hemat). Dengan penerapan *Improved* algoritma *A-Star* dalam destinasi wisata diharapkan mendapatkan hasil yang optimal, sehingga dapat memenuhi perintah dan anjuran tentang sikap *itqan*, pemilihan metode yang baik, dan berhemat.

2.9 Kajian Penelitian dengan Teori Pendukung

Pada penelitian ini untuk memperkuat landasan, akan dipaparkan beberapa kajian penelitian yang berkaitan dengan topik yang bertujuan untuk

mengoptimalkan rute pengiriman di Kantor Pos pemeriksa Kabupaten Blitar. Pada penelitian (Anwar & Manuharawati, 2021) "Implementasi *A-Star* Dalam Penentuan Rute Terpendek Desa Srikandi Rembang Menuju Kampus Universitas Billfath" dengan hasil dapat disimpulkan bahwa penggunaan algoritma *A-star* dapat menemukan rute terpendek dengan enam jrute alternatif dan satu rute terdekat dengan jarak $293.4\ km$. Penelitian selanjutnya berkatan dengan metode yang digunakan pada penelitian ini yaitu "Penerapan Metode A* pada Pencarian Rute Tercepat Menuju Destinasi Wisata Cagar Budaya Menes Pandeglang" yang memiliki hasil penggunaan algoritma *A-Star* menemukan rute terpendek menuju tempat wisata cagar budaya di Kecamatan Menes Kabupaten Pandeglang melalui Kota Serang – Persimpangan Maja – Persimpangan Cipacung – Persimpangan Mengger – Persimpangan Batu Bantar – Desa Alas Wangi – Menes dengan jarak tempuh 29,400 KM (Susilawati et al., 2020).

Penelitian selanjutnya (Wayahdi et al., 2021) dengan tema "*Greedy*, *A-Star*, and *Djikstra's* Algorithms in Finding Shortest path" peneliti membandingkan keefektifan ketiga algoritma tersebut dalam pencarian rute terpendek pada suatu lintasan atau grafik. Dari hasil penelitian tersebut peneliti dapat menyimpulkan bahwa ketiga algoritma ini dapat solusi dalam menentukan lintasan terpendek dengan hasil yang berbeda-beda. Algoritma *Greedy* cepat dalam menemukan solusi tetapi cenderung tidak menemukan solusi yang optimal, algoritma *Djiksta's* dapat menemukan solusi rute optimal akan tetapi algoritma ini cenderung lebih lambat dalam mencari solusi karena harus membandingkan biaya satu lintasan dengan biaya lintasan lain, kemudian algoritma *A-Star* cenderung lebih baik dari *Greedy*

dalam penentuan rute optimal tetapi lintasan atau grafik tersebut harus memiliki data yang lebih kompleks.

BAB III
METODE PENELITIAN

3.1 Jenis Penelitian

Penelitian ini menggunakan pendekatan evaluatif kuantitatif dan pengembangan metode dengan memanfaatkan data lapangan yang diperoleh melalui *Google Maps*. Pendekatan evaluatif kuantitatif diterapkan untuk menganalisis efektivitas algoritma saat diimplementasikan. Pengumpulan data dilakukan melalui observasi lapangan dengan mengakumulasi informasi dari objek penelitian yang ditargetkan. Data yang terkumpul kemudian diolah dan dianalisis untuk menghasilkan kesimpulan kuantitatif berdasarkan hasil pengujian yang telah dilaksanakan.

3.2 Data dan Sumber Data

Penelitian ini menggunakan sumber data sekunder yang diperoleh dari *google map* dengan mengambil titik (*node*), titik koordinat, serta jarak antar titik dengan satuan kilometer. Dengan mengambil titik (*node*) sampel pada destinasi wisata Blitar. Dalam titik-titik sampel tersebut terdiri dari titik awal, titik-titik yang akan dipilih untuk dicari rute optimal, dan titik tujuan. Dari data tersebut akan diambil tiga titik awal dan tiga titik akhir sebagai sampel penelitian untuk perbandingan antara algoritma *A-Star* tradisional dengan modifikasi algoritma *A-Star* dengan manual, dan dua puluh titik awal dan dua puluh titik akhir menggunakan program.

3.3 Tahapan Penelitian

Berikut adalah tahapan penelitian dengan judul “Modifikasi Algoritma *A-Star* dalam Optimalisasi rute terpendek”:

1. Menentukan Data dari Google Maps Berupa Capture Jalur yang Akan Dilewati. Tahap pertama yang harus dilakukan adalah mengamati dan mengumpulkan data obyek yang akan dijadikan sebagai bahan penelitian. Data yang dihimpun berupa data titik (*node*), titik koordinat, serta jarak antar titik dengan satuan kilometer yang diperoleh dari data spasial Google Maps.

   Pada penelitian ini, ada beberapa hal penting yang akan dibahas, mulai dari penangkapan gambar (*capturing*) yang berupa screenshoot titik-titik destinasi budaya, titik-titik lokasi yang sudah didapatkan akan dibuat menjadi *node* sebuah graf, *edge* pada graf diambil sekaligus dari rute perjalanan, menganalisa apakah *edge* tersebut berbentuk dua arah atau satu arah dan untuk bobot *edge* (jarak jalan yang menghubungkan antar titik) diambil dari *google maps* dengan kendaraan mobil yang mana apabila pada proses pencarian jarak antar titik tersebut melalui jalan yang melewati titik lain maka *edge* tersebut tidak diambil.
2. Menentukan dan Mengumpulkan Koordinat dari Titik-Titik Sampel Destinasi Wisata Blitar. Koordinat setiap titik ditetapkan dengan mengambil data *longitude* dan *latitude* lokasi-lokasi serta persimpangan jalan melalui *Google Maps*. Koordinat yang telah diperoleh ini kemudian dimanfaatkan sebagai dasar perhitungan nilai heuristik $h(n)$ dalam implementasi algoritma *A-Star* tradisional dan modifikasi algoritma *A-Star* sebagai perbandingan dalam menemukan rute terpendek.
3. Modifikasi Algoritma *A-Star* guna mendapatkan metode yang lebih optimal. Untuk cara memodifikasi algoritma *A-Star* tradisional yaitu dengan memodifikasi fungsi evaluasi pada algoritma tersebut $f(n) = g(n) + h(n)$,

$g(n)$ mewakili jarak pada peta (km) pada tiap antar titik dan $h(n)$ mewakili jarak perkiraan dengan menggunakan metode *Euclidan*. Untuk meningkatkan efisiensi pencarian algoritma *A-star* ditambahkan variabel normalisasi $\alpha(n)$ pada fungsi evaluasi, yang mana variabel $\alpha(n)$ terdiri dari *average distance* (rata-rata jarak) dan *current distance* (jarak saat ini ke jarak yang terhubung).

Penambahan variabel $\alpha(n)$ berfungsi untuk meningkatkan efisiensi pencarian pada titik (*node*) yang ekstrem, maksud dari ekstrem yakni perbedaan jarak sisi (*edge*) yang terlalu tinggi dengan sisi yang lain dapat menyebabkan banyak update pencarian, sehingga kegunaan penambahan vaariabel $\alpha(n)$ adalah untuk menemukan jalur optimal dengan minimum iterasi.

4. Menghitung Nilai Heuristik. Dalam proses pencarian dan perhitungan nilai heuristik, diperlukan metode untuk mengevaluasi data dari kondisi tertentu guna menentukan sejauh mana data tersebut dapat membantu menemukan solusi yang efisien. Nilai heuristik bervariasi bergantung pada titik tujuan yang diinginkan dengan cara menentukan titik tujuan dan menghitung pada masing masing titik n, sehingga perannya menjadi faktor penting dalam penyelesaian masalah. Fungsi ini diterapkan untuk menghitung jarak antara dua titik.

   Perhitungan fungsi heuristik dalam suatu graf dengan menggunakan algoritma *A-Star* tradisional yakni menghitung nilai $h(n)$ pada tiap titik n menuju titik tujuan ($node_{goal}$), yang mana nilai heuristik ini sebagai bahan untuk fungsi evaluasi $f(n)$. Untuk cara perhitungan nilai heuristik $h(n)$ pada modifikasi algoritma *A-Star* masih sama yakni dengan menggunakan metode

Euclidan (2.2) dan merubah satuan dari $h(n)$ dari derajat menjadi km dengan mengalikan 1 *degree earth* $\left(111.332\,\frac{km}{\circ}\right)$. 1 derajat bumi didapatkan dari total keliling bumi yaitu $40{,}075\ km$ (empat puluh ribu tujuh puluh lima kilo meter) dibagi 360 derajat sehingga menghasilkan $\left(111.332\,\frac{km}{\circ}\right)$ (GANNETT, 1916).

5. Melakukan Eksperimen Pencarian Rute Terpendek untuk Mendapatkan Hasil Perbandingan Algoritma. Setelah data sampel dikumpulkan langkah berikutnya adalah melakukan eksperimen pencarian rute terpendek untuk membandingkan efisiensi algoritma *A-Star* tradisional dengan modifikasi algoritma *A-Star*. Dalam pengujian ini, akan ditentukan titik awal dan akhir rute pada titik-titik sampel destinasi wisata Blitar. Proses pengujian dilakukan dengan mengambil tiga sampel uji berupa tiga titik awal dan tiga titik akhir pada sampel data destinasi wisata Blitar untuk perbandingan proses manual, dan dua puluh untuk program. Penerapan dengan modifikasi algoritma *A-Star* masih sama dengan algoritma *A-Star* tradisional.
6. Menampilkan Program Visualisasi Hasil Perbandingan algoritma *A-Star* tradisional dengan Modifikasi algoritma *A-Star* menggunakan *Jupyter notebook.*

BAB IV
HASIL DAN PEMBAHASAN

4.1 Deskripsi data

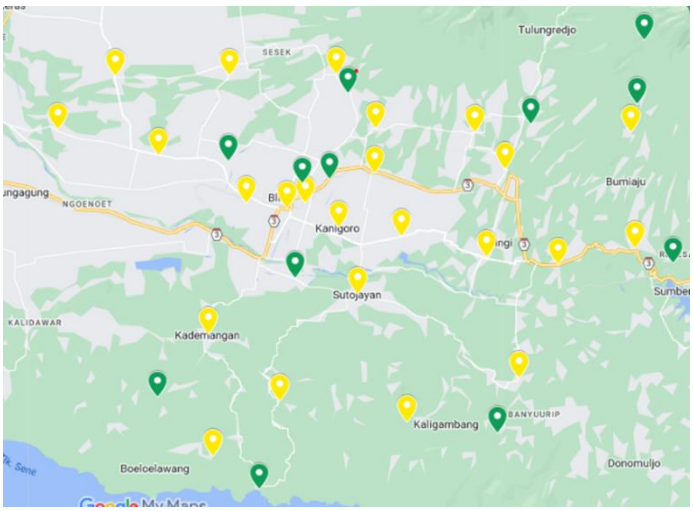
Data yang digunakan dalam penelitian ini berasal dari *Google Maps* dengan mengambil titik-titik lokasi pada tiap kecamatan, titik-titik destinasi wisata, titik koordinat pada tiap titik (*node*), serta jarak antar titik (*node*). Sebagaimana telah diringkas pada tabel dibawah ini:

Tabel 4.1 Data Titik Sampel Kecamatan dan Wisata Blitar

No	Titik (*node*)	Simbol	Titik Koordinat
1	Kebun Teh Sirah Kencong	$W2$	$(-7.9761°, 112.43014°)$
2	Gumuk Sapu Angin	$W3$	$(-8.02476°, 112.4243°)$
3	Rambut Monte	$W5$	$(-8.01237°, 112.039°)$
4	Doko	$K1$	$(-8.04568°, 112.41986°)$
5	Bendungan Lahor	$W6$	$(-8.14501°, 112.45102°)$
6	Selorejo	$K2$	$(-8.13352°, 112.42316°)$
7	Kesamben	$K3$	$(-8.14563°, 112.36573°)$
8	Selopuro	$K4$	$(-8.14034°, 112.31279°)$
9	Wlingi	$K5$	$(-8.07364°, 112.3264°)$
10	Gandusari	$K6$	$(-8.04564°, 112.30376°)$
11	Garum	$K7$	$(-8.04307°, 112.23021°)$
12	Candi penataran	$W7$	$(-8.01642°, 112.20974°)$
13	Nglegok	$K8$	$(-8.1241°, 112.2495°)$
14	Talun	$K9$	$(-8.1241°, 112.2495°)$
15	Kanigoro	$K10$	$(-8.11798°, 112.20272°)$
16	Sutojayan	$K11$	$(-8.16805°, 112.21737°)$
17	Kampung Coklat	$W8$	$(-8.15605°, 112.1702°)$
18	Blitar Park	$W9$	$(-8.08099°, 112.19602°)$
19	Makam Bung Karno	$W10$	$(-8.08461°, 112.17608°)$
20	Sanan Wetan	$K12$	$(-8.09918°, 112.1781°)$
21	Kepanjen Kidul	$K13$	$(-8.10288°, 112.1639°)$
22	Sanan Kulon	$K14$	$(-8.09945°, 112.13405°)$
23	Penangkaran Rusa Maliran	$W11$	$(-8.0673°, 112.12124°)$
24	Srengat	$K15$	$(-8.0629°, 112.06911°)$
25	Ponggok	$K16$	$(-8.00251°, 112.12145°)$
26	Udanawu	$K17$	$(-8.00351°, 112.03683°)$
27	Wonodadi	$K18$	$(-8.0437°, 111.99413°)$
28	Kademangan	$K19$	$(-8.19811°, 112.10594°)$
29	Goa Luweng	$W12$	$(-8.24644°, 112.06807°)$
30	Bakung	$K20$	$(-8.29064°, 112.1092°)$

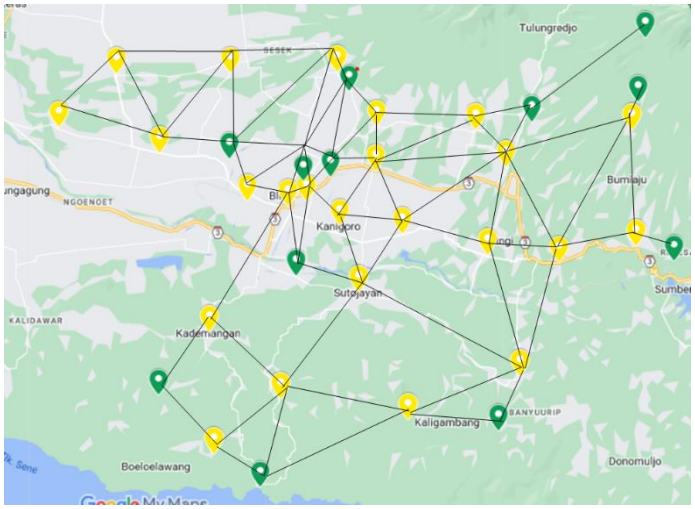
No	Titik (*node*)	Simbol	Titik Koordinat
31	Tambak	$W13$	$(-8.31541°, 112.14345°)$
32	Wonotirto	$K21$	$(-8.24858°, 112.15893°)$
33	Panggung Rejo	$K22$	$(-8.26417°, 112.25334°)$
34	Air Terjun Jurug Bening	$W14$	$(-8.27248°, 112.32052°)$
35	Binangun	$K23$	$(-8.23152°, 112.33673°)$
36	SPBU Garum	$K24$	$(-8.07625°, 112.22918°)$

Simbol W merupakan titik wisata dan K adalah jalan besar pada tiap kecamatan. Hal ini dilakukan sebagai pembeda. Untuk detail gambarnya akan dilampirkan sebagai berikut:



Gambar 4.1 Data Titik Sebaran Jalan Beserta Wisata

Pada gambar diatas titik kuning mewakili titik jalan Kecamatan dan titik hijau mewakili wisata. Pada masing-masing antar titik tersebut memiliki jalan penghubung. Data jarak antar titik kami lampirkan pada lampiran (1). Data jarak antar titik tersebut kita ambil berdasarkan jarak sebenarnya pada *Google Maps* dengan satuan kilometer dengan menggunakan kendaraan roda empat. Berikut visualisasi graf sebaran wisata beserta jalan pada tiap kecamatan:



Gambar 4.2 Graf Sebaran Titik pada Maps

Untuk graf dengan tambahan bobot pada setiap sisi (jalan yang menghubungkan antar titik) dan simbol titik (jalan pada tiap kecamatan dan wisata) kita lampirkan pada Lampiran (2).

4.2 Modifikasi Algoritma *A-Star*

Modifikasi algoritma *A-Star* dilakukan guna memperoleh metode yang lebih optimal dari algoritma *A-Star* tradisional. Modifikasi ini dilakukan pada fungsi evaluasi $f(n)$ dengan menambah faktor normalisasi $\alpha(n)$ pada variabel nilai heuristik $h(n)$. Penambahan variabel $\alpha(n)$ variabel normalisasi berfungsi untuk meningkatkan efisiensi pencarian pada titik (*node*) yang ekstrem, maksud dari ekstrem yakni perbedaan jarak sisi (*edge*) yang terlalu tinggi dengan sisi yang lain dapat menyebabkan banyak iterasi dalam proses pencarian, sehingga penambahan variabel $\alpha(n)$ berfungsi untuk menemukan jalur optimal dengan iterasi yang lebih minimum. Sehingga fungsi evaluasi termodifikasi sebagai berikut :

$$f(n) = g(n) + h(n) * \alpha(n) \qquad (4.1)$$

Keterangan:

$f(n)$: fungsi evaluasi titik n pada setiap simpul

$g(n)$: Akumulatif biaya aktual dari titik awal hingga titik n

$h(n)$: perkiraan biaya dari titik n ke titik tujuan, juga dikenal sebagai fungsi heuristik

$\alpha(n)$: Faktor normalisasi

Faktor normalisasi sebagai berikut:

$$\alpha(n) = \frac{\mu_e}{d(n)} \tag{4.2}$$

Keterangan:

μ_e : rata- rata dari semua jarak antar titik

$d(n)$: jarak dari titik saat ini (A) ke titik yang terhubung

4.3 Pengujian Pencarian Rute Terpendek

Pengujian rute terpendek akan dilakukan sesuai dengan algoritma tahapan penelitian yang telah dijelaskan pada bab 3. Berikut merupakan alur penelitian pengujian rute:

1. Implementasi Algoritma *A-Star* Tradisional

   Langkah awal implementasi algoritma *A-Star* yakni kita tentukan titik awal beserta titik tujuan. Sebagaimana dalam tabel berikut :

Tabel 4.2 Titik Awal dan Titik Tujuan Pengujian

Pengujian ke-	Titik Awal	Titik Tujuan
1	$K6$	$W8$
2	$K2$	$W10$
3	$K22$	$W5$

a. Penentuan Rute $K6 \rightarrow W8$

Kita cari nilai heuristik $h(n)$ pada tiap titik n_n (titik selain titik awal K6 dan titik tujuan W8) menuju titik tujuan W8, hasil pencarian kita lampirkan pada Lampiran (3). K6 merupakan titik awal sehingga K6 kita masukkan kedalam *Closed_List* dengan jarak tempuh $g(K6) = 0$ km*:*

Closed_list $= \{K6\}$

Kemudian kita cari titik yang terhubung pada K6 dan kita masukkan ke dalam *Open_List* dan kita hitung nilai evaluasi titik tiitk tersebut:

Open_List = $\{K5, K7, K24\}$

$$\begin{aligned} f(K5) &= g(K5) + h(K5) = 5.8 + 19.6 = 25.4 \\ f(K7) &= g(K7) + h(K7) = 8.5 + 14.2 = 22.7 \\ f(K24) &= g(K24) + h(K24) = 12.3 + 10.6 = 22.9 \end{aligned} \quad (1)$$

Dari hasil perhitungan pertama kita peroleh bahwa K7 memiliki nilai evaluasi yang minimum sehingga K7 kita masukkan ke dalam *Closed_List* dengan jarak tempuh $g(\mathrm{K7}) = 8.5$ km*:*

Closed_list $= \{K6, K7\}$

Kemudian kita cari titik yang terhubung dengan titik K7, kita update *Open_List* dengan memasukkan titik-titik tersebut, dan kita hitung nilai evaluasi pada masing-masing titik:

Open_List = $\{K24, W9, W7\}$

$$\begin{aligned} f(K24) &= g(K24) + h(K24) = 8.5 + 5 + 10.6 = 24.1 \\ f(W9) &= g(W9) + h(W9) = 8.5 + 6.7 + 8.7 = 23.9 \\ f(W7) &= g(W7) + h(W7) = 8.5 + 7.2 + 15.6 = 31.3 \end{aligned} \quad (2)$$

Karena pada perhitungan pertama masih terdapat nilai evaluasi yang lebih minimum maka kita update pencarian rute pada titik dengan nilai evaluasi

minimum yaitu pada ($K6 \rightarrow K24$) dengkemudian kita update *Closed_List* dengan jarak tempuh $g(K24) = 12.3$ km:

Closed_list= $\{K6, K24\}$

Kemudian kita cari titik yang terhubung dengan K24, kita update *Open_List* dengan memasukkan titik-titik tersebut, dan kita hitung nilai evaluasi pada masing-masing titik:

Open_List = $\{K7, W9, K9\}$

$$\begin{aligned} f(K7) &= g(K7) + h(K7) = 12.3 + 5 + 14.2 = 31.5 \\ f(W9) &= g(W9) + h(W9) = 12.3 + 4.2 + 8.7 = 25.2 \\ f(K9) &= g(K9) + h(K9) = 12.3 + 7.5 + 9.4 = 29.2 \end{aligned} \quad (3)$$

Karena pada perhitungan kedua terdapat nilai evaluasi yang lebih minimum maka kita update pencarian rute pada titik dengan nilai evaluasi minimum yaitu pada ($K6 \rightarrow K7 \rightarrow W9$) kemudian kita update *Closed_List* dengan jarak tempuh $g(W9) = 8.5$ km*:*

Closed_list= $\{K6, K7, W9\}$

Kemudian kita cari titik yang terhubung dengan W9, kita update *Open_List* dengan memasukkan titik-titik tersebut, dan kita hitung nilai evaluasi pada masing-masing titik:

Open_List = $\{W10, K12, K10, K12\}$

$$\begin{aligned} f(W10) &= g(W10) + h(W10) = 15.2 + 3.3 + 7.45 = 25.95 \\ f(K12) &= g(K12) + h(K12) = 15.2 + 3.1 + 5.9 = 24.2 \\ f(K10) &= g(K10) + h(K10) = 15.2 + 6.1 + 5.4 = 26.7 \\ f(K24) &= g(K24) + h(K24) = 15.2 + 4.2 + 10.6 = 30 \end{aligned} \quad (4)$$

Karena pada perhitungan ke-dua masih terdapat nilai evaluasi yang lebih minimum maka kita update pencarian rute pada titik dengan nilai evaluasi

minimum yaitu pada ($K6 \rightarrow K7 \rightarrow K24$) kemudian kita update *Closed_List* dengan jarak tempuh $g(K24) = 13.5$ km:

Closed_list= $\{K6, K7, K24\}$

Kemudian kita cari titik yang terhubung dengan K24, kita update *Open_List* dengan memasukkan titik-titik tersebut, dan kita hitung nilai evaluasi pada masing-masing titik:

Open_List = $\{K9, W9, K5\}$

$$\begin{aligned} f(K9) &= g(K9) + h(K9) = 13.5 + 7.5 + 9.4 = 30.4 \\ f(W9) &= g(W9) + h(W9) = 13.5 + 4.2 + 8.6 = 26.3 \\ f(K5) &= g(K5) + h(K5) = 13.5 + 12.8 + 19.6 = 45.9 \end{aligned} \quad (5)$$

Karena pada perhitungan ke-empat terdapat nilai evaluasi yang lebih minimum maka kita update pencarian rute pada titik dengan nilai evaluasi minimum yaitu pada ($K6 \rightarrow K7 \rightarrow W9 \rightarrow K12$) kemudian kita update *Closed_List* dengan jarak tempuh $g(K12) = 18.3$ km:

Closed_list= $\{K6, K7, W9, K12\}$

Kemudian kita cari titik yang terhubung dengan K12, kita update *Open_List* dengan memasukkan titik-titik tersebut, dan kita hitung nilai evaluasi pada masing-masing titik:

Open_List = $\{W10, K13, K10\}$

$$\begin{aligned} f(W10) &= g(W10) + h(W10) = 18.3 + 2.4 + 7.5 = 28.1 \\ f(K13) &= g(K13) + h(K13) = 18.3 + 2.2 + 5.8 = 26.3 \\ f(K10) &= g(K10) + h(K10) = 18.3 + 4.5 + 5.4 = 28.1 \end{aligned} \quad (6)$$

Karena pada perhitungan ke-tiga terdapat nilai evaluasi yang lebih minimum maka kita update pencarian rute pada titik dengan nilai evaluasi minimum yaitu pada ($K6 \rightarrow K24 \rightarrow W9$) kemudian kita update *Closed_List* dengan jarak tempuh $g(W9) = 16.5$ km:

Closed_list= $\{K6, K24, W9\}$

Kemudian kita cari titik yang terhubung dengan W9, kita update *Open_List* dengan memasukkan titik-titik tersebut, dan kita hitung nilai evaluasi pada masing-masing titik:

Open_List = $\{K10, K12, W10, K7\}$

$$\begin{aligned} f(K10) &= g(K10) + h(K10) = 16.5 + 6.1 + 5.4 = 28 \\ f(K12) &= g(K12) + h(K12) = 16.5 + 3.1 + 5.9 = 25.5 \\ f(W10) &= g(W10) + h(W10) = 16.5 + 3.3 + 7.5 = 27.3 \\ f(K7) &= g(K7) + h(K7) = 16.5 + 6.7 + 14.2 = 37.4 \end{aligned} \quad (7)$$

Karena pada perhitungan pertama masih terdapat nilai evaluasi yang lebih minimum maka kita update pencarian rute pada titik dengan nilai evaluasi minimum yaitu pada $(K6 \rightarrow K5)$ kemudian kita update *Closed_List* dengan jarak tempuh $g(K5) = 5.8$ km:

Closed_list= $\{K6, K5\}$

Kemudian kita cari titik yang terhubung dengan titik K5, kita update *Open_List* dengan memasukkan titik-titik tersebut, dan kita hitung nilai evaluasi pada masing-masing titik:

Open_List = $\{K24, K4, K9\}$

$$\begin{aligned} f(K24) &= g(K24) + h(K24) = 5.8 + 12{,}8 + 10.6 = 29.2 \\ f(K4) &= g(K4) + h(K4) = 5.8 + 8.4 + 15.9 = 30.1 \\ f(K9) &= g(K9) + h(K9) = 5.8 + 13.1 + 9.4 = 28.3 \end{aligned} \quad (8)$$

Karena pada perhitungan ke-tujuh terdapat nilai evaluasi yang lebih minimum maka kita update pencarian rute pada titik dengan nilai evaluasi minimum yaitu pada $(K6 \rightarrow K24 \rightarrow W9 \rightarrow K12)$ kemudian kita update *Closed_List* dengan jarak tempuh $g(K12) = 19.6$ km:

Closed_list= $\{K6, K24, W9, K12\}$

Kemudian kita cari titik yang terhubung dengan K12, kita update *Open_List* dengan memasukkan titik-titik tersebut, dan kita hitung nilai evaluasi pada masing-masing titik:

Open_List = $\{K10, K13, W10\}$

$$\begin{aligned} f(K10) &= g(K10) + h(K10) = 19.6 + 4.5 + 5.4 = 29.5 \\ f(K13) &= g(K13) + h(K13) = 19.6 + 2.2 + 5.8 = 27.6 \\ f(W10) &= g(W10) + h(W10) = 19.6 + 2.4 + 7.5 = 29.5 \end{aligned} \quad (9)$$

Karena pada perhitungan ke-empat masih terdapat nilai evaluasi yang lebih minimum maka kita update pencarian rute pada titik dengan nilai evaluasi minimum yaitu pada $(K6 \rightarrow K7 \rightarrow W9 \rightarrow W10)$ kemudian kita update *Closed_List* dengan jarak tempuh $g(W10) = 18.5$ km:

Closed_list= $\{K6, K7, W9, W10\}$

Kemudian kita cari titik yang terhubung dengan W10, kita update *Open_List* dengan memasukkan titik-titik tersebut, dan kita hitung nilai evaluasi pada masing-masing titik:

Open_List = $\{K12, K13, K14\}$

$$\begin{aligned} f(K12) &= g(K12) + h(K12) = 18.5 + 2.4 + 5.9 = 26.8 \\ f(K13) &= g(K13) + h(K13) = 18.5 + 3.6 + 5.8 = 27.8 \\ f(K14) &= g(K14) + h(K14) = 18.5 + 7.2 + 7.5 = 33.2 \end{aligned} \quad (10)$$

Karena pada perhitungan ke-enam terdapat nilai evaluasi yang lebih minimum maka kita update pencarian rute pada titik dengan nilai evaluasi minimum yaitu pada $(K6 \rightarrow K7 \rightarrow W9 \rightarrow K12 \rightarrow K13)$ kemudian kita update *Closed_List* dengan jarak tempuh $g(K13) = 20.5$ km:

Closed_list= $\{K6, K7, W9, K12, K13\}$

Kemudian kita cari titik yang terhubung dengan K13, kita update *Open_List* dengan memasukkan titik-titik tersebut, dan kita hitung nilai evaluasi pada masing-masing titik:

Open_List = $\{K14, K19, K10, W8\}$

$$\begin{aligned} f(K14) &= g(K14) + h(K14) = 20.5 + 3.9 + 7.5 = 31.9 \\ f(K19) &= g(K19) + h(K19) = 20.5 + 16.5 + 8.7 = 45.7 \\ f(K10) &= g(K10) + h(K10) = 20.5 + 5.8 + 5.4 = 31.7 \\ f(W8) &= g(W8) + h(W8) = 20.5 + 9.5 + 0 = 30 \end{aligned} \quad (11)$$

Karena pada perhitungan ke-lima masih terdapat nilai evaluasi yang lebih minimum maka kita update pencarian rute pada titik dengan nilai evaluasi minimum yaitu pada $(K6 \rightarrow K7 \rightarrow K24 \rightarrow W9)$ kemudian kita update *Closed_List* dengan jarak tempuh $g(W9) = 17.7$ km:

Closed_list= $\{K6, K7, K24, W9\}$

Kemudian kita cari titik yang terhubung dengan W9, kita update *Open_List* dengan memasukkan titik-titik tersebut, dan kita hitung nilai evaluasi pada masing-masing titik:

Open_List = $\{W10, K12, K10\}$

$$\begin{aligned} f(W10) &= g(W10) + h(W10) = 17.7 + 3.3 + 7.5 = 28.5 \\ f(K12) &= g(K12) + h(K12) = 17.7 + 3.1 + 5.9 = 26.7 \\ f(K10) &= g(K10) + h(K10) = 17.7 + 6.1 + 5.4 = 29.2 \end{aligned} \quad (12)$$

Karena pada perhitungan ke-empat terdapat nilai evaluasi yang lebih minimum maka kita update pencarian rute pada titik dengan nilai evaluasi minimum yaitu pada $(K6 \rightarrow K7 \rightarrow W9 \rightarrow K10)$ kemudian kita update *Closed_List* dengan jarak tempuh $g(K10) = 21.3$ km:

Closed_list= $\{K6, K7, W9, K10\}$

Kemudian kita cari titik yang terhubung dengan K10, kita update *Open_List* dengan memasukkan titik-titik tersebut, dan kita hitung nilai evaluasi pada masing-masing titik:

Open_List = $\{K9, K11, W8, K13\}$

$$\begin{aligned} f(K9) &= g(K9) + h(K9) = 21.3 + 6.6 + 9.4 = 37.3 \\ f(K11) &= g(K11) + h(K11) = 21.3 + 7.1 + 5.5 = 33.9 \\ f(W8) &= g(W8) + h(W8) = 21.3 + 7.9 + 0 = 29.2 \\ f(K13) &= g(K13) + h(K13) = 21.3 + 5.8 + 5.8 = 32.9 \end{aligned} \quad (13)$$

Karena pada perhitungan ke-duabelas terdapat nilai evaluasi yang lebih minimum maka kita update pencarian rute pada titik dengan nilai evaluasi minimum yaitu pada $(K6 \rightarrow K7 \rightarrow K24 \rightarrow W9 \rightarrow K12)$ kemudian kita update *Closed_List* dengan jarak tempuh $g(K12) = 20.8$ km:

Closed_list= $\{K6, K7, K24, W9, K12\}$

Kemudian kita cari titik yang terhubung dengan titik K12, kita update *Open_List* dengan memasukkan titik-titik tersebut, dan kita hitung nilai evaluasi pada masing-masing titik:

Open_List = $\{W10, K13, K10\}$

$$\begin{aligned} f(W10) &= g(W10) + h(W10) = 20.8 + 2.4 + 7.5 = 30.7 \\ f(K13) &= g(K13) + h(K13) = 20.8 + 2.2 + 5.8 = 28.8 \\ f(K10) &= g(K10) + h(K10) = 20.8 + 4.5 + 5.44 = 30.74 \end{aligned} \quad (14)$$

Karena pada perhitungan ke-sepuluh masih terdapat nilai evaluasi yang lebih minimum maka kita update pencarian rute pada titik dengan nilai evaluasi minimum yaitu pada $(K6 \rightarrow K7 \rightarrow W9 \rightarrow W10 \rightarrow K12)$ kemudian kita update *Closed_List* dengan jarak tempuh $g(K12) = 20.9$ km:

Closed_list= $\{K6, K7, W9, W10, K12\}$

Kemudian kita cari titik yang terhubung dengan W10, kita update *Open_List* dengan memasukkan titik-titik tersebut, dan kita hitung nilai evaluasi pada masing-masing titik:

Open_List = $\{K13, K10\}$

$$f(K10) = g(K10) + h(K10) = 20.9 + 4.5 + 5.4 = 30.8$$
$$f(K13) = g(K13) + h(K13) = 20.9 + 2.2 + 5.8 = 28.9 \quad (15)$$

Karena pada perhitungan ke-tujuh terdapat nilai evaluasi yang lebih minimum maka kita update pencarian rute pada titik dengan nilai evaluasi minimum yaitu pada $(K6 \rightarrow K24 \rightarrow W9 \rightarrow W10)$ kemudian kita update *Closed_List* dengan jarak tempuh $g(W10) = 19.8$ km:

Closed_list= $\{K6, K24, W9, W10\}$

Kemudian kita cari titik yang terhubung dengan W10, kita update *Open_List* dengan memasukkan titik-titik tersebut, dan kita hitung nilai evaluasi pada masing-masing titik:

Open_List = $\{K12, K13, K14, W11, K8\}$

$$f(K12) = g(K12) + h(K12) = 19.8 + 2.4 + 5.94 = 28.14$$
$$f(K13) = g(K13) + h(K13) = 19.8 + 3.6 + 5.8 = 29.2$$
$$f(K14) = g(K14) + h(K14) = 19.8 + 7.2 + 7.5 = 34.5$$
$$f(W11) = g(W11) + h(W11) = 19.8 + 11.6 + 11.1 = 42.5 \quad (16)$$
$$f(K8) = g(K8) + h(K8) = 19.8 + 13.4 + 17.2 = 50.4$$

Karena pada perhitungan ke-sembilan terdapat nilai evaluasi yang lebih minimum maka kita update pencarian rute pada titik dengan nilai evaluasi minimum yaitu pada $(K6 \rightarrow K24 \rightarrow W9 \rightarrow K12 \rightarrow K13)$ kemudian kita update *Closed_List* dengan jarak tempuh $g(W10) = 21.8$ km:

Closed_list= $\{K6, K24, W9, K12, K13\}$

Kemudian kita cari titik yang terhubung dengan K13, kita update *Open_List* dengan memasukkan titik-titik tersebut, dan kita hitung nilai evaluasi pada masing-masing titik:

Open_List = $\{K14, K19, K10, W8\}$

$$\begin{aligned} f(K14) &= g(K14) + h(K14) = 21.8 + 3.9 + 7.5 = 32.2 \\ f(K19) &= g(K19) + h(K19) = 21.8 + 16.5 + 8.7 = 47 \\ f(K10) &= g(K10) + h(K10) = 21.8 + 45.8 + 5.44 = 33 \\ f(W8) &= g(W8) + h(W8) = 21.8 + 9.5 + 0 = 31.3 \end{aligned} \quad (17)$$

Karena pada perhitungan ke-sepuluh masih terdapat nilai evaluasi yang lebih minimum maka kita update pencarian rute pada titik dengan nilai evaluasi minimum yaitu pada ($K6 \rightarrow K7 \rightarrow W9 \rightarrow W10 \rightarrow K13$) kemudian kita update *Closed_List* dengan jarak tempuh $g(K13) = 22.1$ km:

Closed_list= $\{K6, K7, W9, W10, K13\}$

Kemudian kita cari titik yang terhubung dengan W10, kita update *Open_List* dengan memasukkan titik-titik tersebut, dan kita hitung nilai evaluasi pada masing-masing titik:

Open_List = $\{K14, K19, K10, W8\}$

$$\begin{aligned} f(K14) &= g(K14) + h(K14) = 22.1 + 3.9 + 7.53 = 33.53 \\ f(K19) &= g(K19) + h(K19) = 22.1 + 16.5 + 8.7 = 47.3 \\ f(K10) &= g(K10) + h(K10) = 22.1 + 5.8 + 5.44 = 33.3 \\ f(W8) &= g(W8) + h(W8) = 22.1 + 9.5 + 0 = 31.6 \end{aligned} \quad (18)$$

Karena pada perhitungan ke-tujuh terdapat nilai evaluasi yang lebih minimum maka kita update pencarian rute pada titik dengan nilai evaluasi minimum yaitu pada ($K6 \rightarrow K24 \rightarrow W9 \rightarrow K10$) kemudian kita update *Closed_List* dengan jarak tempuh $g(K10) = 22.6$ km:

Closed_list= $\{K6, K24, W9, K10\}$

Kemudian kita cari titik yang terhubung dengan K10, kita update *Open_List* dengan memasukkan titik-titik tersebut, dan kita hitung nilai evaluasi pada masing-masing titik:

Open_List = $\{K9, K11, W8, K13\}$

$$\begin{aligned} f(K9) &= g(K9) + h(K9) = 22.6 + 6.6 + 9.4 = 38.6 \\ f(K11) &= g(K11) + h(K11) = 22.6 + 7.1 + 5.5 = 35.2 \\ f(W8) &= g(W8) + h(W8) = 22.6 + 7.9 + 0 = 30.5 \\ f(K13) &= g(K13) + h(K13) = 22.6 + 5.8 + 5.8 = 34.2 \end{aligned} \quad (19)$$

Karena pada perhitungan ke-enam belas terdapat nilai evaluasi yang lebih minimum maka kita update pencarian rute pada titik dengan nilai evaluasi minimum yaitu pada $(K6 \rightarrow K24 \rightarrow W9 \rightarrow W10 \rightarrow K12)$ kemudian kita update *Closed_List* dengan jarak tempuh $g(K12) = 22.2$ km:

Closed_list= $\{K6, K24, W9, W10, K12\}$

Kemudian kita cari titik yang terhubung dengan K12, kita update *Open_List* dengan memasukkan titik-titik tersebut, dan kita hitung nilai evaluasi pada masing-masing titik:

Open_List = $\{K13, K10\}$

$$\begin{aligned} f(K13) &= g(K13) + h(K13) = 22.2 + 2.2 + 5.8 = 30.2 \\ f(K10) &= g(K10) + h(K10) = 22.2 + 4.5 + 5.4 = 32.1 \end{aligned} \quad (20)$$

Karena pada perhitungan ke-enam terdapat nilai evaluasi yang lebih minimum maka kita update pencarian rute pada titik dengan nilai evaluasi minimum yaitu pada $(K6 \rightarrow K7 \rightarrow W9 \rightarrow K12 \rightarrow W10)$ kemudian kita update *Closed_List* dengan jarak tempuh $g(W10) = 20.7$ km:

Closed_list= $\{K6, K7, W9, K12, W10\}$

Kemudian kita cari titik yang terhubung dengan W10, kita update *Open_List* dengan memasukkan titik-titik tersebut, dan kita hitung nilai evaluasi pada masing-masing titik:

Open_List = $\{K12, K13, K14, W11, K8\}$

$$\begin{aligned} f(K13) &= g(K13) + h(K13) = 20.7 + 3.6 + 5.8 = 30.1 \\ f(K14) &= g(K14) + h(K14) = 20.7 + 7.2 + 7.5 = 35.4 \\ f(W11) &= g(W11) + h(W11) = 20.7 + 11.6 + 11.1 \\ &= 43.4 \\ f(K8) &= g(K8) + h(K8) = 20.7 + 13.4 + 17.2 = 51.3 \end{aligned} \quad (21)$$

Karena pada perhitungan ke-enam terdapat nilai evaluasi yang lebih minimum maka kita update pencarian rute pada titik dengan nilai evaluasi minimum yaitu pada $(K6 \rightarrow K7 \rightarrow W9 \rightarrow K12 \rightarrow K10)$ kemudian kita update *Closed_List* dengan jarak tempuh $g(K10) = 22.8$ km:

Closed_list= $\{K6, K7, W9, K12, K10\}$

Kemudian kita cari titik yang terhubung dengan K10, kita update *Open_List* dengan memasukkan titik-titik tersebut, dan kita hitung nilai evaluasi pada masing-masing titik:

Open_List = $\{K9, K11, W8, K13\}$

$$\begin{aligned} f(K9) &= g(K9) + h(K9) = 22.6 + 6.6 + 9.4 = 38.6 \\ f(K11) &= g(K11) + h(K11) = 22.6 + 7.1 + 11.2 = 40.9 \\ f(W8) &= g(W8) + h(W8) = 22.6 + 7.9 + 0 = 30.5 \\ f(K13) &= g(K13) + h(K13) = 22.6 + 5.8 + 5.8 = 34.2 \end{aligned} \quad (22)$$

Karena pada perhitungan ke-delapan masih terdapat nilai evaluasi yang lebih minimum maka kita update pencarian rute pada titik dengan nilai evaluasi minimum yaitu pada $(K6 \rightarrow K5 \rightarrow K9)$ kemudian kita update *Closed_List* dengan jarak tempuh $g(K13) = 18.9$ km:

Closed_list= $\{K6, K5, K9\}$

Kemudian kita cari titik yang terhubung dengan titik K9, kita update *Open_List* dengan memasukkan titik-titik tersebut, dan kita hitung nilai evaluasi pada masing-masing titik:

Open_List = $\{K24, K10, K11\}$

$$\begin{aligned} f(K24) &= g(K24) + h(K24) = 18.9 + 7.5 + 10.6 = 37 \\ f(K10) &= g(K10) + h(K10) = 18.9 + 6.6 + 5.4 = 30.9 \\ f(K11) &= g(K11) + h(K11) = 18.9 + 7.9 + 5.5 = 32.3 \end{aligned} \quad (23)$$

Karena pada perhitungan ke-dua belas masih terdapat nilai evaluasi yang lebih minimum maka kita update pencarian rute pada titik dengan nilai evaluasi minimum yaitu pada $(K6 \rightarrow K7 \rightarrow K24 \rightarrow W9 \rightarrow W10)$ kemudian kita update *Closed_List* dengan jarak tempuh $g(W10) = 21.$ km:

Closed_list= $\{K6, K7, K24, W10\}$

Kemudian kita cari titik yang terhubung dengan W10, kita update *Open_List* dengan memasukkan titik-titik tersebut, dan kita hitung nilai evaluasi pada masing-masing titik:

Open_List = $\{K12, K13, K14, W11, K8\}$

$$\begin{aligned} f(K13) &= g(K13) + h(K13) = 21 + 3.6 + 5.8 = 30.4 \\ f(K14) &= g(K14) + h(K14) = 21 + 7.2 + 7.5 = 35.7 \\ f(W11) &= g(W11) + h(W11) = 21 + 11.6 + 11.1 = 43.7 \\ f(K8) &= g(K8) + h(K8) = 21 + 13.4 + 17.2 = 51.6 \end{aligned} \quad (24)$$

Karena pada perhitungan ke-enam belas terdapat nilai evaluasi yang lebih minimum maka kita update pencarian rute pada titik dengan nilai evaluasi minimum yaitu pada $(K6 \rightarrow K7 \rightarrow K24 \rightarrow W9 \rightarrow K12 \rightarrow K13)$ kemudian kita update *Closed_List* dengan jarak tempuh $g(WK1310) = 23$ km:

Closed_list= $\{K6, K7, K24, W9, K12, K13\}$

Kemudian kita cari titik yang terhubung dengan titik K13, kita update *Open_List* dengan memasukkan titik-titik tersebut, dan kita hitung nilai evaluasi pada masing-masing titik:

Open_List = $\{K14, K19, K10, W8\}$

$$\begin{aligned} f(K14) &= g(K14) + h(K14) = 23 + 3.9 + 7.5 = 34.4 \\ f(K19) &= g(K19) + h(K19) = 23 + 16.5 + 8.7 = 48.2 \\ f(K10) &= g(K10) + h(K10) = 23 + 5.8 + 5.4 = 32.9 \\ f(W8) &= g(W8) + h(W8) = 23 + 9.5 + 0 = 32.5 \end{aligned} \quad (25)$$

Karena pada perhitungan ke-lima belas masih terdapat nilai evaluasi yang lebih minimum maka kita update pencarian rute pada titik dengan nilai evaluasi minimum yaitu pada $(K6 \rightarrow K7 \rightarrow W9 \rightarrow W10 \rightarrow K12 \rightarrow K13)$ kemudian kita update *Closed_List* dengan jarak tempuh $g(K13) = 23.1$ km:

Closed_list= $\{K6, K7, K24, W9, K12, K13\}$

Kemudian kita cari titik yang terhubung dengan K13, kita update *Open_List* dengan memasukkan titik-titik tersebut, dan kita hitung nilai evaluasi pada masing-masing titik:

Open_List = $\{K14, K19, K10, W8\}$

$$\begin{aligned} f(K14) &= g(K14) + h(K14) = 23.1 + 3.9 + 7.5 = 34.5 \\ f(K19) &= g(K19) + h(K19) = 23.1 + 16.5 + 8.7 = 48.3 \\ f(K10) &= g(K10) + h(K10) = 23.1 + 5.8 + 5.4 = 34.3 \\ f(W8) &= g(W8) + h(W8) = 23.1 + 9.5 + 0 = 32.6 \end{aligned} \quad (26)$$

Karena pada perhitungan ke-tiga masih terdapat nilai evaluasi yang lebih minimum maka kita update pencarian rute pada titik dengan nilai evaluasi minimum yaitu pada $(K6 \rightarrow K24 \rightarrow K9)$ kemudian kita update *Closed_List* dengan jarak tempuh $g(K13) = 19.8$ km:

Closed_list= $\{K6, K24, K9\}$

Kemudian kita cari titik yang terhubung dengan K9, kita update *Open_List* dengan memasukkan titik-titik tersebut, dan kita hitung nilai evaluasi pada masing-masing titik:

Open_List = $\{K7, W9, K9\}$

$$\begin{aligned} f(K4) &= g(K4) + h(K4) = 19.8 + 9 + 15.9 = 44.7 \\ f(K10) &= g(K10) + h(K10) = 19.8 + 6.6 + 5.4 = 31.8 \\ f(K11) &= g(K8) + h(K8) = 19.8 + 7.9 + 5.5 = 33.2 \end{aligned} \quad (27)$$

Karena pada perhitungan ke-delapan belas masih terdapat nilai evaluasi yang lebih minimum maka kita update pencarian rute pada titik dengan nilai evaluasi minimum yaitu pada ($K6 \rightarrow K24 \rightarrow W9 \rightarrow W10 \rightarrow K13$) kemudian kita update *Closed_List* dengan jarak tempuh $g(K13) = 23.4$ km:

Closed_list= $\{K6, K24, W9, W10, K13\}$

Kemudian kita cari titik yang terhubung dengan W10, kita update *Open_List* dengan memasukkan titik-titik tersebut, dan kita hitung nilai evaluasi pada masing-masing titik:

Open_List = $\{K14, K19, K10, W8\}$

$$\begin{aligned} f(K14) &= g(K14) + h(K14) = 23.4 + 3.9 + 7.5 = 34.8 \\ f(K19) &= g(K19) + h(K19) = 23.4 + 16.5 + 8.7 = 48.6 \\ f(K10) &= g(K10) + h(K10) = 23.4 + 5.8 + 5.44 = 34.6 \\ f(W8) &= g(W8) + h(W8) = 23.4 + 9.5 + 0 = 32.9 \end{aligned} \quad (28)$$

Karena pada perhitungan ke-tujuh belas terdapat nilai evaluasi yang lebih minimum dan mencapai titik tujuan maka rute terpendek telah ditemukan ($K6 \rightarrow K7 \rightarrow W9 \rightarrow K10 \rightarrow W8$) dengan total update perubahan rute pencarian atau *Closed_List* sebanyak dua puluh lima kali, total iterasi 28 kali, dengan rute sepanjang 29.2 km

Closed_List = $\{K6, K7, W9, K10, W8\}$

b. Penentuan Rute $K2 \rightarrow W10$

Kita cari nilai heuristik $h(n)$ pada tiap titik n_n (titik selain titik awal K2 dan titik tujuan W10) menuju titik tujuan W8, hasil pencarian kita lampirkan pada Lampiran (4). K2 merupakan titik awal sehingga K2 kita masukkan kedalam *Closed_List* dengan jarak tempuh $g(K2) = 0$ km:

Closed_list$= \{K2\}$

Kemudian kita cari titik yang terhubung pada K2 dan kita masukkan ke dalam *Open_List* dan kita hitung nilai evaluasi titik tiitk tersebut:

Open_List = $\{K1, K3\}$

$$\begin{aligned} f(K1) &= g(K1) + h(K1) = 13.4 + 39.3 = 52.7 \\ f(K3) &= g(K3) + h(K3) = 8.3 + 21.9 = 30.2 \end{aligned} \quad (1)$$

Dari hasil perhitungan pertama kita peroleh bahwa K3 memiliki nilai evaluasi yang minimum sehingga K3 kita masukkan ke dalam *Closed_List* dengan jarak tempuh $g(K3) = 8.3$ km:

Closed_list$= \{K2, K3\}$

Kemudian kita cari titik yang terhubung dengan titik K3, kita update *Open_List* dengan memasukkan titik-titik tersebut, dan kita hitung nilai evaluasi pada masing-masing titik:

Open_List = $\{K1, K5, K4, K23\}$

$$\begin{aligned} f(K1) &= g(K1) + h(K1) = 8.3 + 13.55 + 27.3 = 49.1 \\ f(K5) &= g(K5) + h(K5) = 8.3 + 11.7 + 16.6 = 36.6 \\ f(K4) &= g(K4) + h(K4) = 8.3 + 6.8 + 31.2 = 31.2 \\ f(K23) &= g(K23) + h(K23) = 8.3 + 12.8 + 24 = 43.1 \end{aligned} \quad (2)$$

Dari hasil perhitungan ke-dua kita peroleh bahwa K4 memiliki nilai evaluasi yang minimum sehingga K4 kita masukkan ke dalam *Closed_List* dengan jarak tempuh $g(K4) = 15.1$ km:

Closed_list= $\{K2, K3, K4\}$

Kemudian kita cari titik yang terhubung dengan titik K4, kita update *Open_List* dengan memasukkan titik-titik tersebut, dan kita hitung nilai evaluasi pada masing-masing titik:

Open_List = $\{K5, K23, K9\}$

$$\begin{aligned} f(K5) &= g(K5) + h(K5) = 15.1 + 8.4 + 16.6 = 40.1 \\ f(K23) &= g(K23) + h(K23) = 15.1 + 14.8 + 24 = 53.9 \\ f(K9) &= g(K9) + h(K9) = 15.1 + 9 + 9.07 = 33.17 \end{aligned} \quad (3)$$

Dari hasil perhitungan ke-tiga kita peroleh bahwa K9 memiliki nilai evaluasi yang minimum sehingga K4 kita masukkan ke dalam *Closed_List* dengan jarak tempuh $g(K9) = 24.1$ km:

Closed_list= $\{K2, K3, K4, K9\}$

Kemudian kita cari titik yang terhubung dengan titik K9, kita update *Open_List* dengan memasukkan titik-titik tersebut, dan kita hitung nilai evaluasi pada masing-masing titik:

Open_List = $\{K24, K5, K10, K11\}$

$$\begin{aligned} f(K24) &= g(K24) + h(K24) = 24.1 + 7.5 + 5.81 = 37.41 \\ f(K5) &= g(K5) + h(K5) = 24.1 + 13.1 + 16.6 = 53.8 \\ f(K10) &= g(K10) + h(K10) = 24.1 + 6.6 + 4.08 = 34.78 \\ f(K11) &= g(K11) + h(K11) = 24.1 + 7.9 + 9.8 = 41.8 \end{aligned} \quad (4)$$

Dari hasil perhitungan ke-empat kita peroleh bahwa K10 memiliki nilai evaluasi yang minimum sehingga K10 kita masukkan ke dalam *Closed_List* dengan jarak tempuh $g(K10) = 30.7$ km:

Closed_list= $\{K2, K3, K4, K9, K10\}$

Kemudian kita cari titik yang terhubung dengan titik K10, kita update *Open_List* dengan memasukkan titik-titik tersebut, dan kita hitung nilai evaluasi pada masing-masing titik:

Open_List = $\{W9, K12, K13, K11\}$

$$\begin{aligned} f(W9) &= g(W9) + h(W9) = 30.7 + 6.1 + 2.1 = 38.9 \\ f(K12) &= g(K12) + h(K12) = 30.7 + 4.5 + 1.05 = 36.25 \\ f(K13) &= g(K13) + h(K13) = 30.7 + 5.8 + 1.92 = 38.42 \\ f(K11) &= g(K11) + h(K11) = 30.7 + 7.1 + 9.8 = 47.6 \end{aligned} \quad (5)$$

Dari hasil perhitungan ke-lima kita peroleh bahwa K12 memiliki nilai evaluasi yang minimum sehingga K12 kita masukkan ke dalam *Closed_List* dengan jarak tempuh $g(K12) = 35.2$ km:

Closed_list= $\{K2, K3, K4, K9, K10, K12\}$

Kemudian kita cari titik yang terhubung dengan titik K12, kita update *Open_List* dengan memasukkan titik-titik tersebut, dan kita hitung nilai evaluasi pada masing-masing titik:

Open_List = $\{W9, W10, K13\}$

$$\begin{aligned} f(W9) &= g(W9) + h(W9) = 35.2 + 3.1 + 2.1 = 41.7 \\ f(W10) &= g(W10) + h(W10) = 35.2 + 2.4 + 0 = 37.6 \\ f(K13) &= g(K13) + h(K13) = 35.2 + 2.2 + 1.92 = 40.62 \end{aligned} \quad (6)$$

Karena pada perhitungan ke-dua masih terdapat nilai evaluasi yang lebih minimum maka kita update pencarian rute pada titik dengan nilai evaluasi minimum yaitu pada $(K2 \rightarrow K3 \rightarrow K5)$ kemudian kita update *Closed_List* dengan jarak tempuh $g(K13) = 20$ km:

Closed_list= $\{K2, K3, K5\}$

Kemudian kita cari titik yang terhubung dengan K5, kita update *Open_List* dengan memasukkan titik-titik tersebut, dan kita hitung nilai evaluasi pada masing-masing titik:

Open_List = $\{W4, K6, K24, K9, K4\}$

$$f(W4) = g(W4) + h(W4) = 20 + 4.8 + 19.3 = 44.1$$
$$f(K6) = g(K6) + h(K6) = 20 + 5.8 + 14.6 = 40.4$$
$$f(K24) = g(K24) + h(K24) = 20 + 12.8 + 5.81 = 38.61 \quad (7)$$
$$f(K9) = g(K9) + h(K9) = 20 + 13.1 + 9.07 = 42.17$$
$$f(K4) = g(K4) + h(K4) = 20 + 8.4 + 16.1 = 44.5$$

Karena pada perhitungan ke-empat masih terdapat nilai evaluasi yang lebih minimum maka kita update pencarian rute pada titik dengan nilai evaluasi minimum yaitu pada $(K2 \rightarrow K3 \rightarrow K4 \rightarrow K5 \rightarrow K24)$ kemudian kita update *Closed_List* dengan jarak tempuh $g(K24) = 31.6$km:

Closed_list= $\{K2, K3, K4, K9, K24\}$

Kemudian kita cari titik yang terhubung dengan K24, kita update *Open_List* dengan memasukkan titik-titik tersebut, dan kita hitung nilai evaluasi pada masing-masing titik:

Open_List = $\{K7, W9, K6\}$

$$f(K7) = g(K7) + h(K7) = 31.6 + 5 + 7.4 = 44$$
$$f(W9) = g(W9) + h(W9) = 31.6 + 6.7 + 2.15 = 40.45 \quad (8)$$
$$f(K6) = g(K6) + h(K6) = 31.6 + 12.3 + 14.6 = 58.8$$

Karena pada perhitungan ke-enam terdapat nilai evaluasi yang lebih minimum dan mencapai titik tujuan maka rute terpendek telah ditemukan $(K2 \rightarrow K3 \rightarrow K4 \rightarrow K9 \rightarrow K10 \rightarrow K12 \rightarrow W10)$ dengan total update perubahan rute pencarian atau *Closed_List* sebanyak dua kali, total perhitungan delapan kali, dengan rute sepanjang 37.6 km

Closed_List $= \{K2, K3, K4, K9, K10, K12, W10\}$

c. Penentuan Rute $K22 \rightarrow W5$

Kita cari nilai heuristik $h(n)$ pada tiap titik $node_n$ (titik selain titik awal K22 dan titik tujuan W5) menuju titik tujuan W5, hasil pencarian kita lampirkan pada Lampiran 5. K22 merupakan titik awal sehingga K22 kita masukkan kedalam *Closed_List* dengan jarak tempuh $g(K22) = 0$ km:

Closed_list$= \{K22\}$

Kemudian kita cari titik yang terhubung pada K22 dan kita masukkan ke dalam *Open_List* dan kita hitung nilai evaluasi titik tiitk tersebut:

Open_List = $\{K21, K11, W14\}$

$$\begin{aligned} f(K21) &= g(K21) + h(K21) = 20 + 30.9 = 50.9 \\ f(K11) &= g(K11) + h(K11) = 17.7 + 19.8 = 37.5 \\ f(W14) &= g(W14) + h(W14) = 10.7 + 25.4 = 36.1 \end{aligned} \quad (1)$$

Dari hasil perhitungan pertama kita peroleh bahwa W14 memiliki nilai evaluasi yang minimum sehingga W14 kita masukkan ke dalam *Closed_List* dengan jarak tempuh $g(W14) = 10.7$ km:

Closed_list$= \{K22, W14\}$

Kemudian kita cari titik yang terhubung dengan titik W14, kita update *Open_List* dengan memasukkan titik-titik tersebut, dan kita hitung nilai evaluasi pada masing-masing titik:

Open_List = $\{K23\}$

$$f(K23) = g(K23) + h(K23) = 10.7 + 6 + 20.6 = 37.3 \quad (2)$$

Dari hasil perhitungan ke-dua kita peroleh bahwa K23 memiliki nilai evaluasi yang minimum sehingga K23 kita masukkan ke dalam *Closed_List* dengan jarak tempuh $g(K23) = 16.7$ km:

Closed_list= $\{K22, W14, K23\}$

Kemudian kita cari titik yang terhubung dengan titik W14, kita update *Open_List* dengan memasukkan titik-titik tersebut, dan kita hitung nilai evaluasi pada masing-masing titik:

Open_List = $\{K11, K4, K3\}$

$$f(K11) = g(K11) + h(K11) = 16.7 + 17.8 + 19.8 = 54.3$$
$$f(K4) = g(K4) + h(K4) = 16.7 + 14.8 + 11.2 = 42.7 \quad (3)$$
$$f(K3) = g(K3) + h(K3) = 16.7 + 12.8 + 11.3 = 40.8$$

Karena pada perhitungan pertama masih terdapat nilai evaluasi yang lebih minimum maka kita update pencarian rute pada titik dengan nilai evaluasi minimum yaitu pada $(K22 \rightarrow K11)$ kemudian kita update *Closed_List* dengan jarak tempuh $g(K13) = 17.7$ km:

Closed_list= $\{K22, K11\}$

Kemudian kita cari titik yang terhubung dengan K11, kita update *Open_List* dengan memasukkan titik-titik tersebut, dan kita hitung nilai evaluasi pada masing-masing titik:

Open_List = $\{K21, W8, K10, K9, K23\}$

$$f(K21) = g(K21) + h(K21) = 17.7 + 16.3 + 30.9 = 64.9$$
$$f(W8) = g(W8) + h(W8) = 17.7 + 5.8 + 22.9 = 46.4$$
$$f(K10) = g(K10) + h(K10) = 17.7 + 7.1 + 17.8 = 42.6 \quad (4)$$
$$f(K9) = g(K9) + h(K9) = 17.7 + 7.9 + 13.7 = 39.3$$
$$f(K23) = g(K23) + h(K23) = 17.7 + 17.8 + 20.6 = 56.1$$

Dari hasil perhitungan ke-empat kita peroleh bahwa K9 memiliki nilai evaluasi yang minimum sehingga K9 kita masukkan ke dalam *Closed_List* dengan jarak tempuh $g(K9) = 25.6$ km:

Closed_list= $\{K22, K11, K9\}$

Kemudian kita cari titik yang terhubung dengan titik K9, kita update *Open_List* dengan memasukkan titik-titik tersebut, dan kita hitung nilai evaluasi pada masing-masing titik:

Open_List = $\{K10, K24, K5, K4\}$

$$\begin{aligned} f(K10) &= g(K10) + h(K10) = 25.6 + 6.6 + 17.8 = 50 \\ f(K24) &= g(K24) + h(K24) = 25.6 + 7.5 + 13.4 = 46.5 \\ f(K5) &= g(K5) + h(K5) = 25.6 + 13.1 + 3.83 = 42.5 \\ f(K4) &= g(K4) + h(K4) = 25.6 + 9 + 11.2 = 45.8 \end{aligned} \quad (5)$$

Karena pada perhitungan ke-tiga masih terdapat nilai evaluasi yang lebih minimum maka kita update pencarian rute pada titik dengan nilai evaluasi minimum yaitu pada $(K22 \rightarrow W14 \rightarrow K23 \rightarrow K3)$ kemudian kita update *Closed_List* dengan jarak tempuh $g(K3) = 29.5$ km:

Closed_list= $\{K22, W14, K23, K3\}$

Kemudian kita cari titik yang terhubung dengan titik K3, kita update *Open_List* dengan memasukkan titik-titik tersebut, dan kita hitung nilai evaluasi pada masing-masing titik:

Open_List = $\{K2, K1, K4, K5\}$

$$\begin{aligned} f(K2) &= g(K11) + h(K11) = 29.5 + 8.3 + 13.1 = 50.9 \\ f(K1) &= g(K4) + h(K4) = 29.5 + 13.5 + 8.18 = 51.2 \\ f(K4) &= g(K4) + h(K4) = 29.5 + 6.8 + 11.2 = 47.5 \\ f(K5) &= g(K5) + h(K5) = 29.5 + 11.7 + 3.83 = 45 \end{aligned} \quad (6)$$

Karena pada perhitungan ke-lima masih terdapat nilai evaluasi yang lebih minimum maka kita update pencarian rute pada titik dengan nilai evaluasi minimum yaitu pada $(K22 \rightarrow K11 \rightarrow K9 \rightarrow K5)$ kemudian kita update *Closed_List* dengan jarak tempuh $g(K13) = 38.7$ km:

Closed_list= $\{K22, K11, K9, K5\}$

Kemudian kita cari titik yang terhubung dengan titik K5, kita update *Open_List* dengan memasukkan titik-titik tersebut, dan kita hitung nilai evaluasi pada masing-masing titik:

Open_List = $\{K24, K6, W5, K1, K3, K4\}$

$$\begin{aligned} f(K24) &= g(K24) + h(K24) = 38.7 + 12.8 + 13.4 = 64.9 \\ f(K6) &= g(K6) + h(K6) = 38.7 + 5.8 + 4.57 = 49.1 \\ f(W4) &= g(W4) + h(W5) = 38.7 + 4.8 + 0 = 43.5 \\ f(K1) &= g(K1) + h(K1) = 38.7 + 16.9 + 8.18 = 63.8 \\ f(K3) &= g(K3) + h(K3) = 38.7 + 11.7 + 11.3 = 61.7 \\ f(K4) &= g(K4) + h(K4) = 38.7 + 8.4 + 11.2 = 58.3 \end{aligned} \quad (7)$$

Karena pada perhitungan ke-empat masih terdapat nilai evaluasi yang lebih minimum maka kita update pencarian rute pada titik dengan nilai evaluasi minimum yaitu pada $(K22 \rightarrow K11 \rightarrow K10)$ kemudian kita update *Closed_List* dengan jarak tempuh $g(K10) = 24.8$ km:

Closed_list= $\{K22, K11, K10\}$

Kemudian kita cari titik yang terhubung dengan K10, kita update *Open_List* dengan memasukkan titik-titik tersebut, dan kita hitung nilai evaluasi pada masing-masing titik:

Open_List = $\{W9, K12, K13, W8\}$

$$\begin{aligned} f(W9) &= g(W9) + h(W9) = 24.8 + 6.1 + 30.9 = 47.7 \\ f(K12) &= g(K12) + h(K12) = 24.8 + 4.5 + 22.9 = 48.7 \\ f(K13) &= g(K13) + h(K13) = 24.8 + 5.8 + 17.8 = 51.6 \\ f(W8) &= g(W8) + h(W8) = 24.8 + 7.9 + 13.7 = 55.6 \end{aligned} \quad (8)$$

Karena pada perhitungan ke-tiga masih terdapat nilai evaluasi yang lebih minimum maka kita update pencarian rute pada titik dengan nilai evaluasi minimum yaitu pada $(K22 \rightarrow W14 \rightarrow K23 \rightarrow K4)$ kemudian kita update *Closed_List* dengan jarak tempuh $g(K4) = 31.5$ km:

Closed_list= $\{K22, W14, K23, K4\}$

Kemudian kita cari titik yang terhubung dengan titik K4, kita update *Open_List* dengan memasukkan titik-titik tersebut, dan kita hitung nilai evaluasi pada masing-masing titik:

Open_List = $\{K3, K5, K9\}$

$$f(K3) = g(K3) + h(K3) = 31.5 + 6.8 + 11.3 = 49.6$$
$$f(K5) = g(K5) + h(K5) = 31.5 + 8.4 + 3.83 = 43.7 \quad (9)$$
$$f(K9) = g(K9) + h(K9) = 31.5 + 9 + 13.7 = 54.2$$

Karena pada perhitungan ke-tujuh terdapat nilai evaluasi yang lebih minimum dan mencapai titik tujuan maka rute terpendek telah ditemukan ($K22 \rightarrow K11 \rightarrow K9 \rightarrow K5 \rightarrow W5$) dengan total update perubahan rute pencarian atau *Closed_List* sebanyak lima kali, total perhitungan sembilan kali, dengan rute sepanjang 43.5 km.

Closed_List = $\{K22, K11, K9, K5, W5\}$

2. Implementasi Modifikasi Algoritma *A-Star*

Implementasi modifikasi algoritma *A-star* untuk tahapan penerapan tetap menggunakan langkah-langkah algoritma *A-Star* Tradisional, yang membedakan hanya pada fungsi evaluasi. Adapun untuk fungsi evaluasi modifikasi Algoritma *A-Star* Sebagai berikut:

$$f(n) = g(n) + h(n) * \alpha(n); \quad \alpha(n) = \frac{\mu_e}{d(n)}$$

Sehingga kita cari terlebih dahulu nilai rata-rata semua bobot *edge* :

$$\mu_e = \frac{\sum e_i}{|E|} = \frac{jumlah\ total\ bobot\ edge}{jumlah\ total\ edge}$$

=(8+8.9+8.6+10.3+4.8+11.8+8.5+17.3+11.6+5.9+3.9+7.2+2.3+13.4+3.3+10.9+3.6+2.4+2.2+5.8+9.5+16.5+12.7+11.5+14+8.9+11.4+12.4+13.2+22.7+20+13.4+16.3+7.9+5.8+10.7+17.7+6+7.1+7.9+17.8+14.+12.8+6.6+6.1+4.2+6.7+9.2+7.2+8.5+5+12.3+12.8+7.5+13.1+9+8.4+6.8+8.3+13.5+11.7+16.9+4.8+5.8+8.9+15.7+10.8+13.+4.2+15.4+2.8+3.1+4.5+3.1)/ 74=9.57027027

Untuk data titik awal dan titik akhir masih sama seperti halnya data implementasi algoritma *A-Star* tradisional, pada tabel (4.2).

a. Penentuan Rute $K6 \rightarrow W8$

Untuk nilai $h(n)$ sama dengan pencarian rute pada algoritma *A-Star* Tradisional pada pencarian rute terpendek $K6 \rightarrow W8$. $K6$ sebagai titik awal dan $W8$ sebagai titik tujuan, maka K6 kita masukkan kedalam *Closed_List* dengan jarak tempuh $g(K6) = 0$ km:

Closed_list= $\{K6\}$

Kemudian kita cari titik yang terhubung dengan titik K6, kita update *Open_List* dengan memasukkan titik-titik tersebut, dan kita hitung nilai evaluasi pada masing-masing titik:

Open_List = $\{K7, K24, K5\}$

$$f(K7) = g(K7) + h(K7) * \alpha(K7) = 8.5 + 14.2 * \frac{9.57}{8.5}$$
$$= 24.487$$
$$f(K24) = g(K24) + h(K24) * \alpha(K24) = 12.3 + 10.6 * \frac{9.57}{12.3} \quad (1)$$
$$= 20.547$$

$$f(K5) = g(K5) + h(K5) * \alpha(K5) = 5.8 + 19.6 * \frac{9.57}{5.8}$$
$$= 38.14$$

Dari hasil perhitungan pertama kita peroleh bahwa K24 memiliki nilai evaluasi yang minimum sehingga K24 kita masukkan ke dalam *Closed_List* dengan jarak tempuh $g(K5) = 12.3$ km:

Closed_list= $\{K6, K5\}$

Kemudian kita cari titik yang terhubung dengan titik K24, kita update *Open_List* dengan memasukkan titik-titik tersebut, dan kita hitung nilai evaluasi pada masing-masing titik:

Open_List = $\{K7, W9, K9\}$

$$f(K7) = g(K7) + h(K7) * \alpha(K7) = 12.3 + 5 + 14.2 * \frac{9.57}{5}$$
$$= 44.479$$
$$f(W9) = g(W9) + h(W9) * \alpha(W9)$$
$$= 12.3 + 4.2 + 8.7 * \frac{9.57}{4.2} = 36.324 \quad (2)$$
$$f(K9) = g(K9) + h(K9) * \alpha(K9)$$
$$= 12.3 + 7.5 + 9.36 * \frac{9.57}{7.5} = 31.743$$

Karena pada perhitungan pertama terdapat nilai evaluasi yang lebih minimum maka kita update pencarian rute pada titik dengan nilai evaluasi minimum yaitu pada $(K6 \rightarrow K7)$ kemudian kita update *Closed_List* dengan jarak tempuh $g(K7) = 8.5$ km:

Closed_list= $\{K6, K7\}$

Kemudian kita cari titik yang terhubung dengan titik K7, kita update *Open_List* dengan memasukkan titik-titik tersebut, dan kita hitung nilai evaluasi pada masing-masing titik:

Open_List = $\{W7, K24, W9\}$

$$
\begin{aligned}
f(W7) &= g(W7) + h(W7) * \alpha(W7) \\
&= 8.5 + 7.2 + 15.6 * \frac{9.57}{7.25} = 36.435 \\
f(K24) &= g(K24) + h(K24) * \alpha(K24) \\
&= 8.5 + 5 + 9.36 * \frac{9.57}{5} = 33.789 \\
f(W9) &= g(W9) + h(W9) * \alpha(W9) = 8.5 + 6.7 + 8.7 * \frac{9.57}{6.7} \\
&= 27.627
\end{aligned} \tag{3}
$$

Dari hasil perhitungan ke-tiga kita peroleh bahwa W9 memiliki nilai evaluasi yang minimum sehingga W9 kita masukkan ke dalam *Closed_List* dengan jarak tempuh $g(W9) = 15.2$ km:

Closed_list= $\{K6, K7, W9\}$

Kemudian kita cari titik yang terhubung dengan titik W9, kita update *Open_List* dengan memasukkan titik-titik tersebut, dan kita hitung nilai evaluasi pada masing-masing titik:

Open_List = $\{K12, K10, K24, W10\}$

$$
\begin{aligned}
f(K12) &= g(K12) + h(K12) * \alpha(K12) \\
&= 15.2 + 3.1 + 5.94 * \frac{9.57}{3.1} = 36.638 \\
f(K10) &= g(K10) + h(K10) * \alpha(K10) \\
&= 15.2 + 6.1 + 5.44 * \frac{9.57}{6.1} = 29.834 \\
f(K24) &= g(K24) + h(K24) * \alpha(K24) \\
&= 15.2 + 4.2 + 10.6 * \frac{9.57}{4.2} = 43.554 \\
f(W10) &= g(W10) + h(W10) * \alpha(W10) \\
&= 15.2 + 3.3 + 7.45 * \frac{9.57}{3.3} = 40.106
\end{aligned} \tag{4}
$$

Dari hasil perhitungan ke-empat kita peroleh bahwa K10 memiliki nilai evaluasi yang minimum sehingga K10 kita masukkan ke dalam *Closed_List* dengan jarak tempuh $g(K10) = 21.3$ km:

Closed_list= $\{K6, K7, W9, K10\}$

Kemudian kita cari titik yang terhubung dengan titik K10, kita update *Open_List* dengan memasukkan titik-titik tersebut, dan kita hitung nilai evaluasi pada masing-masing titik:

Open_List = $\{K12, K13, W8, K11\}$

$$
\begin{aligned}
f(K12) &= g(K12) + h(K12) * \alpha(K12) \\
&= 21.3 + 4.5 + 5.94 * \frac{9.57}{4.5} = 38.432 \\
f(K13) &= g(K13) + h(K13) * \alpha(K13) \\
&= 21.3 + 5.8 + 5.78 * \frac{9.57}{5.8} = 36.637 \\
f(W8) &= g(W8) + h(W8) * \alpha(W8) = 21.3 + 7.9 + 0 * \frac{9.57}{7.9} \\
&= 29.2 \\
f(K11) &= g(K11) + h(K11) * \alpha(K11) \\
&= 21.3 + 7.1 + 5.28 * \frac{9.57}{7.1} = 35.517
\end{aligned}
\tag{5}
$$

Karena pada perhitungan ke-lima terdapat nilai evaluasi yang lebih minimum dan mencapai titik tujuan maka rute terpendek telah ditemukan ($K6 \rightarrow K7 \rightarrow W9 \rightarrow K10 \rightarrow W8$) dengan total update perubahan rute pencarian atau *Closed_List* sebanyak satu kali, total perhitungan lima kali, dengan rute sepanjang 29.2 km.

Closed_List = $\{K6, K7, W9, K10, W8\}$

b. Penentuan Rute $K2 \rightarrow W10$

Untuk nilai $h(n)$ sama dengan pencarian rute pada algoritma *A-Star* Tradisional pada pencarian rute terpendek $K2 \rightarrow W10$. $K2$ sebagai titik awal dan $W10$ sebagai titik tujuan, maka K2 kita masukkan kedalam *Closed_List* dengan jarak tempuh $g(K2) = 0$ km:

Closed_list$= \{K2\}$

Kemudian kita cari titik yang terhubung dengan titik K6, kita update *Open_List* dengan memasukkan titik-titik tersebut, dan kita hitung nilai evaluasi pada masing-masing titik:

Open_List $= \{K1, K2\}$

$$\begin{aligned} f(K1) &= g(K1) + h(K1) * \alpha(K1) = 15.4 + 39.3 * \frac{9.57}{15.4} \\ &= 39.822 \\ f(K3) &= g(K3) + h(K3) * \alpha(K3) = 8.3 + 21.9 * \frac{9.57}{8.3} \\ &= 33.551 \end{aligned} \quad (1)$$

Dari hasil perhitungan pertama kita peroleh bahwa K3 memiliki nilai evaluasi yang minimum sehingga K3 kita masukkan ke dalam *Closed_List* dengan jarak tempuh $g(K3) = 8.3$ km:

Closed_list$= \{K2, K3\}$

Kemudian kita cari titik yang terhubung dengan titik K3, kita update *Open_List* dengan memasukkan titik-titik tersebut, dan kita hitung nilai evaluasi pada masing-masing titik:

Open_List $= \{K5, K4, K3\}$

$$\begin{aligned} f(K5) &= g(K5) + h(K5) * \alpha(K5) \\ &= 8.3 + 11.7 + 16.6 * \frac{9.57}{11.7} = 33.57 \end{aligned} \quad (2)$$

$$f(K4) = g(K4) + h(K4) * \alpha(K4) = 8.3 + 6.8 + 16.1 * \frac{9.57}{6.8}$$
$$= 37.759$$
$$f(K3) = g(K3) + h(K3) * \alpha(K3) = 8.3 + 12.8 + 24 * \frac{9.57}{12.8}$$
$$= 39.04$$

Dari hasil perhitungan ke-dua kita peroleh bahwa K5 memiliki nilai evaluasi yang minimum sehingga K5 kita masukkan ke dalam *Closed_List* dengan jarak tempuh $g(K5) = 20$ km:

Closed_list= $\{K2, K3, K5\}$

Kemudian kita cari titik yang terhubung dengan titik K5, kita update *Open_List* dengan memasukkan titik-titik tersebut, dan kita hitung nilai evaluasi pada masing-masing titik:

Open_List = $\{K6, K24, K9, W5\}$

$$f(K6) = g(K6) + h(K6) * \alpha(K6) = 20 + 5.8 + 14.6 * \frac{9.57}{5.8}$$
$$= 49.89$$
$$f(K24) = g(K24) + h(K24) * \alpha(K24)$$
$$= 20 + 12.8 + 5.81 * \frac{9.57}{12.8} = 37.144$$
$$f(K9) = g(K9) + h(K9) * \alpha(K9) = 20 + 13.1 + 9.07 * \frac{9.57}{13.1} \quad (3)$$
$$= 39.726$$
$$f(W5) = g(W5) + h(W5) * \alpha(W5)$$
$$= 20 + 4.8 + 19.3 * \frac{9.57}{4.8} = 63.288$$

Dari hasil perhitungan ke-tiga kita peroleh bahwa K24 memiliki nilai evaluasi yang minimum sehingga K24 kita masukkan ke dalam *Closed_List* dengan jarak tempuh $g(K24) = 32.8$ km:

Closed_list= $\{K2, K3, K5, K24\}$

Kemudian kita cari titik yang terhubung dengan titik K24, kita update *Open_List* dengan memasukkan titik-titik tersebut, dan kita hitung nilai evaluasi pada masing-masing titik:

Open_List = $\{K7, W9, K9, K6\}$

$$
\begin{aligned}
f(K7) &= g(K7) + h(K7) * \alpha(K7) = 32.8 + 5 + 7.4 * \frac{9.57}{5} \\
&= 51.964 \\
f(W9) &= g(W9) + h(W9) * \alpha(W9) \\
&= 32.8 + 4.2 + 2.15 * \frac{9.57}{4.2} = 41.889 \\
f(K9) &= g(K9) + h(K9) * \alpha(K9) \\
&= 32.8 + 7.5 + 9.07 * \frac{9.57}{7.5} = 51.87 \\
f(K6) &= g(K6) + h(K6) * \alpha(K6) \\
&= 32.8 + 12.3 + 14.6 * \frac{9.57}{12.3} = 56.45
\end{aligned}
\tag{4}
$$

Karena pada perhitungan ke-dua terdapat nilai evaluasi yang lebih minimum maka kita update pencarian rute pada titik dengan nilai evaluasi minimum yaitu pada $(K2 \rightarrow K3 \rightarrow K4)$ kemudian kita update *Closed_List* dengan jarak tempuh $g(K4) = 15.1$ km:

Closed_list= $\{K2, K3, K4\}$

Kemudian kita cari titik yang terhubung dengan titik K4, kita update *Open_List* dengan memasukkan titik-titik tersebut, dan kita hitung nilai evaluasi pada masing-masing titik:

Open_List = $\{K5, K9, K23\}$

$$
\begin{aligned}
f(K5) &= g(K5) + h(K5) * \alpha(K5) \\
&= 15.1 + 8.4 + 16.6 * \frac{9.57}{8.4} = 42.412
\end{aligned}
\tag{5}
$$

$$f(K9) = g(K9) + h(K9) * \alpha(K9) = 15.1 + 9 + 9.07 * \frac{9.57}{9}$$
$$= 33.744$$
$$f(K23) = g(K23) + h(K23) * \alpha(K23)$$
$$= 15.1 + 14.8 + 24 * \frac{9.57}{14.8} = 45.419$$

Dari hasil perhitungan ke-lima kita peroleh bahwa K9 memiliki nilai evaluasi yang minimum sehingga K9 kita masukkan ke dalam *Closed_List* dengan jarak tempuh $g(K9) = 24.1$ km:

Closed_list= $\{K2, K3, K4, K9\}$

Kemudian kita cari titik yang terhubung dengan titik K9, kita update *Open_List* dengan memasukkan titik-titik tersebut, dan kita hitung nilai evaluasi pada masing-masing titik:

Open_List = $\{K10, K24, K11\}$

$$f(K10) = g(K10) + h(K10) * \alpha(K10)$$
$$= 24.1 + 6.6 + 4.08 * \frac{9.57}{6.6} = 36.616$$
$$f(K24) = g(K24) + h(K24) * \alpha(K24)$$
$$= 24.1 + 7.5 + 5.81 * \frac{9.57}{7.5} = 39.013 \quad (6)$$
$$f(K11) = g(K11) + h(K11) * \alpha(K11)$$
$$= 24.1 + 7.9 + 2.15 * \frac{9.57}{7.9} = 43.871$$

Dari hasil perhitungan ke-enam kita peroleh bahwa K10 memiliki nilai evaluasi yang minimum sehingga K10 kita masukkan ke dalam *Closed_List* dengan jarak tempuh $g(K10) = 30.7$ km:

Closed_list= $\{K2, K3, K4, K9, K10\}$

Kemudian kita cari titik yang terhubung dengan titik K10, kita update *Open_List* dengan memasukkan titik-titik tersebut, dan kita hitung nilai evaluasi pada masing-masing titik:

Open_List = $\{W9, K12, K13\}$

$$
\begin{aligned}
f(W9) &= g(W9) + h(W9) * \alpha(W9) \\
&= 30.7 + 6.1 + 2.15 * \frac{9.57}{30.7} = 40.173 \\
f(K12) &= g(K12) + h(K12) * \alpha(K12) \\
&= 30.7 + 4.5 + 1.05 * \frac{9.57}{4.5} = 37.433 \\
f(K13) &= g(K13) + h(K13) * \alpha(K13) \\
&= 30.7 + 5.8 + 1.92 * \frac{9.57}{5.8} = 39.66
\end{aligned} \tag{7}
$$

Dari hasil perhitungan ke-tujuh kita peroleh bahwa K12 memiliki nilai evaluasi yang minimum sehingga K12 kita masukkan ke dalam *Closed_List* dengan jarak tempuh $g(K12) = 35.2$ km:

Closed_list= $\{K2, K3, K4, K9, K10, K12\}$

Kemudian kita cari titik yang terhubung dengan titik K10, kita update *Open_List* dengan memasukkan titik-titik tersebut, dan kita hitung nilai evaluasi pada masing-masing titik:

Open_List = $\{W9, W10, K13\}$

$$
\begin{aligned}
f(W9) &= g(W9) + h(W9) * \alpha(W9) \\
&= 35.2 + 3.1 + 2.15 * \frac{9.57}{3.1} = 44.937 \\
f(W10) &= g(W10) + h(W10) * \alpha(W10) \\
&= 35.2 + 2.4 + 0 * \frac{9.57}{2.4} = 37.6 \\
f(K13) &= g(K13) + h(K13) * \alpha(K13) \\
&= 35.2 + 2.2 + 1.92 * \frac{9.57}{2.2} = 45.752
\end{aligned} \tag{8}
$$

Karena pada perhitungan ke-delapan terdapat nilai evaluasi yang lebih minimum dan mencapai titik tujuan maka rute terpendek telah ditemukan ($K2 \rightarrow K3 \rightarrow K4 \rightarrow K9 \rightarrow K10 \rightarrow K12 \rightarrow W10$) dengan total update perubahan rute pencarian atau *Closed_List* sebanyak satu kali, total perhitungan delapan kali, dengan rute sepanjang 37.6 km.

Closed_List $= \{K2, K3, K4, K9, K10, K12, W10\}$

c. Penentuan Rute $K22 \rightarrow W5$

Untuk nilai $h(n)$ sama dengan pencarian rute pada algoritma *A-Star* tradisional pada pencarian rute terpendek $K22 \rightarrow W5$. $K22$ sebagai titik awal dan $W4$ sebagai titik tujuan, maka K22 kita masukkan kedalam *Closed_List* dengan jarak tempuh $g(K22) = 0$ km:

Closed_list$= \{K22\}$

Kemudian kita cari titik yang terhubung dengan titik K22, kita update *Open_List* dengan memasukkan titik-titik tersebut, dan kita hitung nilai evaluasi pada masing-masing titik:

Open_List = $\{K21, K11, W14\}$

$$
\begin{aligned}
f(K21) &= g(K21) + h(K21) * \alpha(K21) = 20 + 30.9 * \frac{9.57}{20} \\
&= 34.828 \\
f(K11) &= g(K11) + h(K11) * \alpha(K11) = 17.7 + 19.8 * \frac{9.57}{17.7} \\
&= 28.435 \\
f(W14) &= g(W14) + h(W14) * \alpha(W14) \\
&= 10.7 + 25.4 * \frac{9.57}{10.7} = 33.482
\end{aligned}
\qquad (1)
$$

Dari hasil perhitungan pertama kita peroleh bahwa K11 memiliki nilai evaluasi yang minimum sehingga K11 kita masukkan ke dalam *Closed_List* dengan jarak tempuh $g(K11) = 17.7$ km:

Closed_list$= \{K22, K11\}$

Kemudian kita cari titik yang terhubung dengan titik K11, kita update *Open_List* dengan memasukkan titik-titik tersebut, dan kita hitung nilai evaluasi pada masing-masing titik:

Open_List = $\{K21, W8, K10, K9, K23\}$

$$
\begin{aligned}
f(K21) &= g(K21) + h(K21) * \alpha(K21) \\
&= 17.7 + 16.3 + 30.9 * \frac{9.57}{16.3} = 52.193 \\
f(W8) &= g(W8) + h(W8) * \alpha(W8) \\
&= 17.7 + 5.8 + 22.9 * \frac{9.57}{5.8} = 61.392 \\
f(K10) &= g(K10) + h(K10) * \alpha(K10) \\
&= 17.7 + 7.1 + 17.8 * \frac{9.57}{7.1} = 48.86 \\
f(K9) &= g(K9) + h(K9) * \alpha(K9) \\
&= 17.7 + 7.9 + 13.7 * \frac{9.57}{7.9} = 42.243 \\
f(K23) &= g(K23) + h(K23) * \alpha(K23) \\
&= 17.7 + 17.8 + 20.6 * \frac{9.57}{17.8} = 46.606
\end{aligned}
\tag{2}
$$

Karena pada perhitungan pertama terdapat nilai evaluasi yang lebih minimum maka kita update pencarian rute pada titik dengan nilai evaluasi minimum yaitu pada $(K22 \rightarrow W14)$ kemudian kita update *Closed_List* dengan jarak tempuh $g(W14) = 10.7$ km:

Closed_list$= \{K22, W14\}$

Kemudian kita cari titik yang terhubung dengan titik W14, kita update *Open_List* dengan memasukkan titik-titik tersebut, dan kita hitung nilai evaluasi pada masing-masing titik:

Open_List = $\{K23\}$

$$f(K23) = g(K23) + h(K23) * \alpha(K23) \\ = 10.7 + 6 + 20.6 * \frac{9.57}{6} = 49.65 \quad (3)$$

Karena pada perhitungan pertama masih terdapat nilai evaluasi yang lebih minimum maka kita update pencarian rute pada titik dengan nilai evaluasi minimum yaitu pada $(K22 \rightarrow K21)$ kemudian kita update *Closed_List* dengan jarak tempuh $g(W10) = 20.7$ km:

Closed_list= $\{K22, K21\}$

Kemudian kita cari titik yang terhubung dengan titik K21, kita update *Open_List* dengan memasukkan titik-titik tersebut, dan kita hitung nilai evaluasi pada masing-masing titik:

Open_List = $\{K19, W8, K11\}$

$$f(K19) = g(K19) + h(K19) * \alpha(K19) \\ = 20.7 + 11.5 + 31.6 * \frac{9.57}{11.5} = 57.871$$

$$f(W8) = g(W8) + h(W8) * \alpha(W8) \\ = 20 + 15.4 + 22.9 * \frac{9.57}{15.4} = 49.671 \quad (4)$$

$$f(K11) = g(K11) + h(K11) * \alpha(K11) = 20 + 16.3 * \frac{9.57}{16.3} \\ = 47.958$$

Karena pada perhitungan ke-dua masih terdapat nilai evaluasi yang lebih minimum maka kita update pencarian rute pada titik dengan nilai evaluasi

minimum yaitu pada ($K22 \rightarrow K11 \rightarrow K9$) kemudian kita update *Closed_List* dengan jarak tempuh $g(K9) = 25.6$ km:

Closed_list= $\{K22, K11, K9\}$

Kemudian kita cari titik yang terhubung dengan titik K9, kita update *Open_List* dengan memasukkan titik-titik tersebut, dan kita hitung nilai evaluasi pada masing-masing titik:

Open_List = $\{K10, K24, K5, K4\}$

$$
\begin{aligned}
f(K10) &= g(K10) + h(K10) * \alpha(K10) \\
&= 25.6 + 6.1 + 17.8 * \frac{9.57}{7.1} = 58.083 \\
f(K24) &= g(K24) + h(K24) * \alpha(K24) \\
&= 25.6 + 7.5 + 13.4 * \frac{9.57}{7.5} = 50.247 \\
f(K5) &= g(K5) + h(K5) * \alpha(K5) \\
&= 25.6 + 13.1 + 3.83 * \frac{9.57}{17.7} = 41.505 \\
f(K4) &= g(K4) + h(K4) * \alpha(K4) = 25.6 + 9 + 11.2 * \frac{9.57}{9} \\
&= 46.543
\end{aligned}
\tag{5}
$$

Dari hasil perhitungan ke-lima kita peroleh bahwa K5 memiliki nilai evaluasi yang minimum sehingga K5 kita masukkan ke dalam *Closed_List* dengan jarak tempuh $g(K5) = 38.7$ km:

Closed_list= $\{K22, K11, K9, K5\}$

Kemudian kita cari titik yang terhubung dengan titik K9, kita update *Open_List* dengan memasukkan titik-titik tersebut, dan kita hitung nilai evaluasi pada masing-masing titik:

Open_List = $\{K24, K6, W5, K1, K3, K4\}$

$$
\begin{aligned}
f(K24) &= g(K24) + h(K24) * \alpha(K24) \\
&= 38.7 + 12.8 + 13.4 * \frac{9.57}{12.8} = 61.547 \\
f(K6) &= g(K6) + h(K6) * \alpha(K6) \\
&= 38.7 + 5.8 + 4.57 * \frac{9.57}{5.8} = 52.062 \\
f(W4) &= g(W4) + h(W4) * \propto (W4) = 38.7 + 4.8 + 0 * \frac{9.57}{4.8} \\
&= 43.5 \\
f(K1) &= g(K1) + h(K1) * \alpha(K1) \\
&= 38.7 + 16.9 + 3.83 * \frac{9.57}{16.9} = 60.245 \\
f(K3) &= g(K3) + h(K3) * \alpha(K3) \\
&= 38.7 + 11.7 + 8.18 * \frac{9.57}{11.7} = 58.678 \\
f(K4) &= g(K4) + h(K4) * \alpha(K4) \\
&= 38.7 + 8.4 + 11.3 * \frac{9.57}{8.4} = 59.896
\end{aligned}
\tag{6}
$$

Karena pada perhitungan ke-delapan terdapat nilai evaluasi yang lebih minimum dan mencapai titik tujuan maka rute terpendek telah ditemukan $(K22 \rightarrow K11 \rightarrow K9 \rightarrow K5 \rightarrow W4)$ dengan total update perubahan rute pencarian atau *Closed_List* sebanyak tiga kali, total perhitungan enam kali, dengan rute sepanjang 43.5 km

Closed_List $= \{K22, K11, K9, K5, W4\}$

4.3 Analisis Hasil Pengujian

Berikut hasil pengujian pencarian rute terpendek pada algoritma *A-Star* tradisional dan modifikasi algoritma *A-Star* dengan melakukan tiga kali pengujian pada masing-masing algoritma dengan rute yang sama:

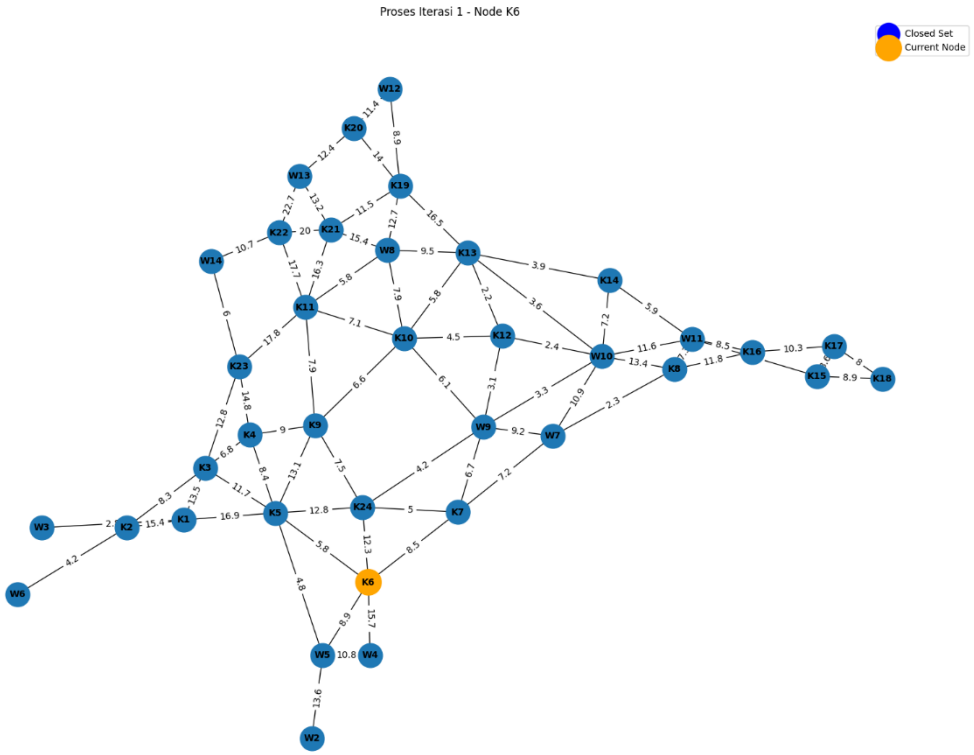
Tabel 4.3 Hasil Pengujian Algoritma *A-Star* Tradisional Manual

Rute	Update Pencarian	Rute Terpendek	Jarak	Jumlah perhitungan
$K6 \rightarrow W8$	25	$K6 \rightarrow K7 \rightarrow W9 \rightarrow K10 \rightarrow W8$	29.2 km	28
$K2 \rightarrow W10$	2	$K2 \rightarrow K3 \rightarrow K4 \rightarrow K9 \rightarrow K10 \rightarrow K12 \rightarrow W10$	37.6 km	8
$K22 \rightarrow W4$	5	$K22 \rightarrow K11 \rightarrow K9 \rightarrow K5 \rightarrow W4$	43.5 km	9

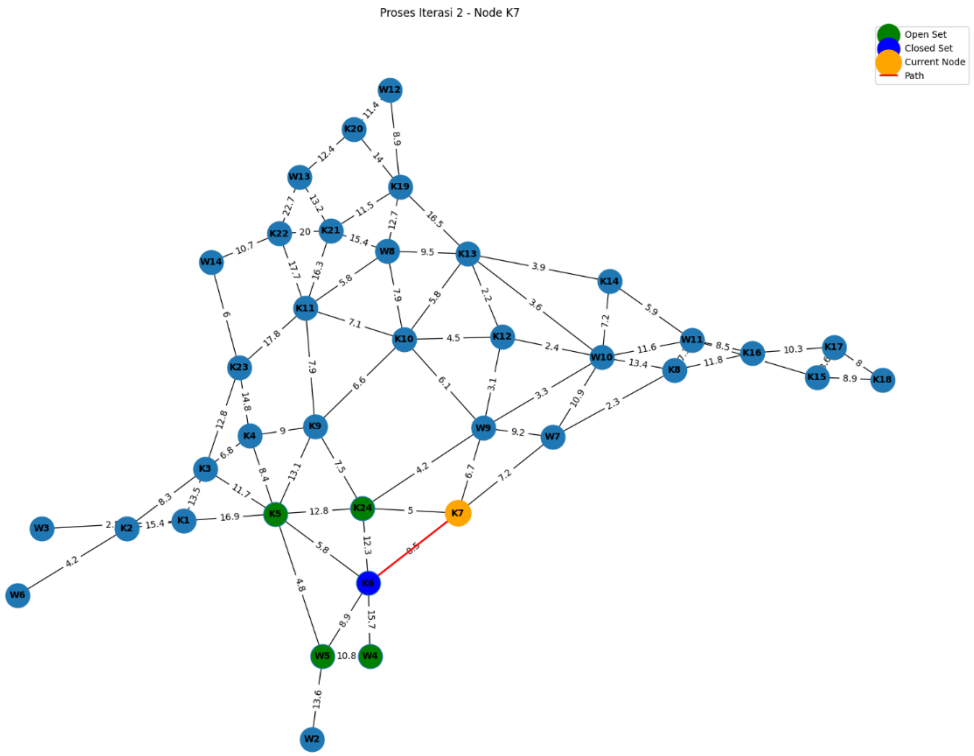
Tabel 4.4 Hasil Pengujian Modifikasi Algoritma *A-Star* Manual

Rute	Update Pencarian	Rute Terpendek	Jarak	Jumlah perhitungan
$K6 \rightarrow W8$	1	$K6 \rightarrow K7 \rightarrow W9 \rightarrow K10 \rightarrow W8$	29.2 km	5
$K2 \rightarrow W10$	1	$K2 \rightarrow K3 \rightarrow K4 \rightarrow K9 \rightarrow K10 \rightarrow K12 \rightarrow W10$	37.6 km	8
$K22 \rightarrow W4$	3	$K22 \rightarrow K11 \rightarrow K9 \rightarrow K5 \rightarrow W4$	43.5 km	6

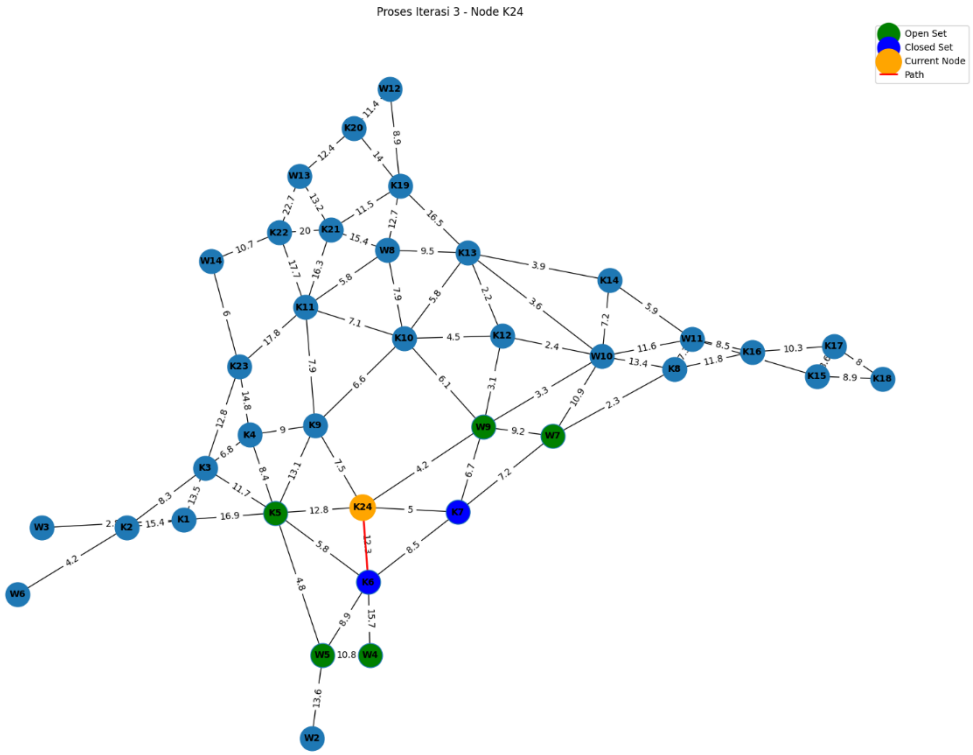
Untuk banyak update pencarian pada tiap pencarian rute kita dapatkan pada jumlah warna hijau pada kata update, dan untuk banyak jumlah perhitungan pada tiap pencarian rute kita dapatkan dari angka terakhir sebelah kanan perhitungan pada perhitungan terakhir. Berikut visualisasi perbandingan algoritma *A-Star* tradisional dengan modifikasi algoritma *A-Star*: Untuk algoritma *A-Star* standar sebagai berikut:



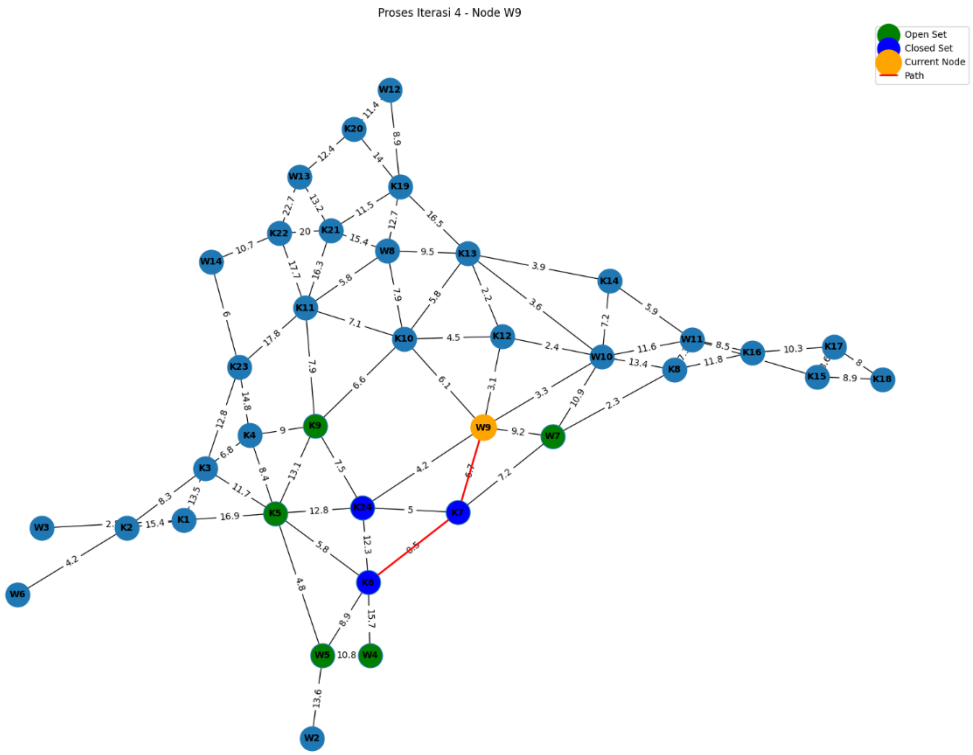
Gambar 4.3 Iterasi 1 Algoritma *A-Star* Standar Rute $\boldsymbol{K6 \rightarrow W8}$



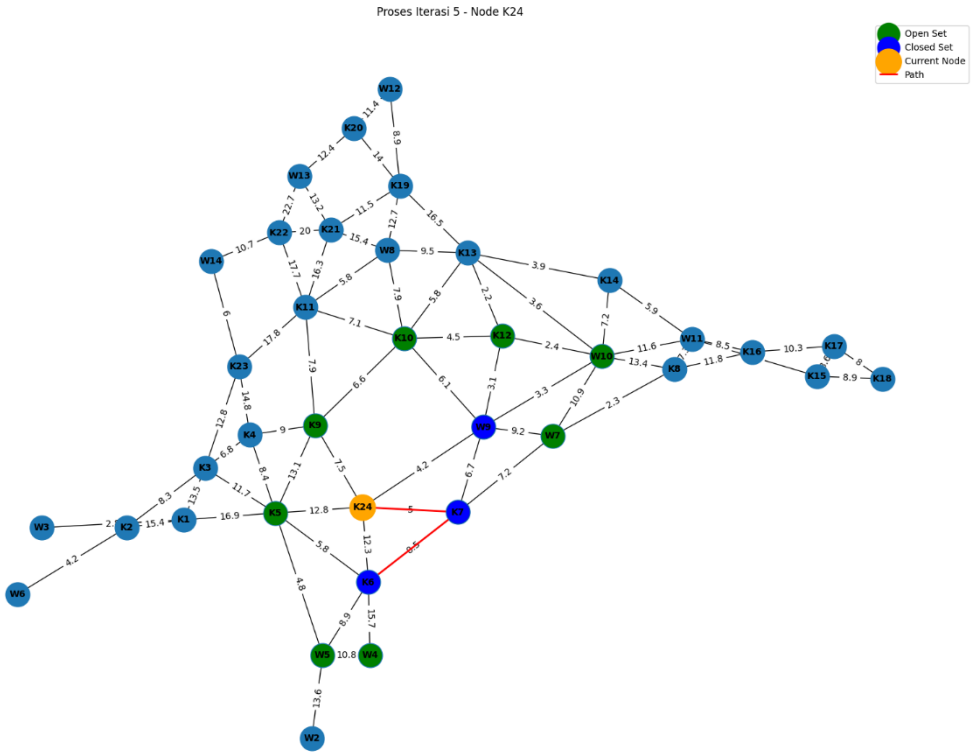
Gambar 4.4 Iterasi 2 Algoritma *A-Star* Standar Rute $\boldsymbol{K6 \rightarrow W8}$



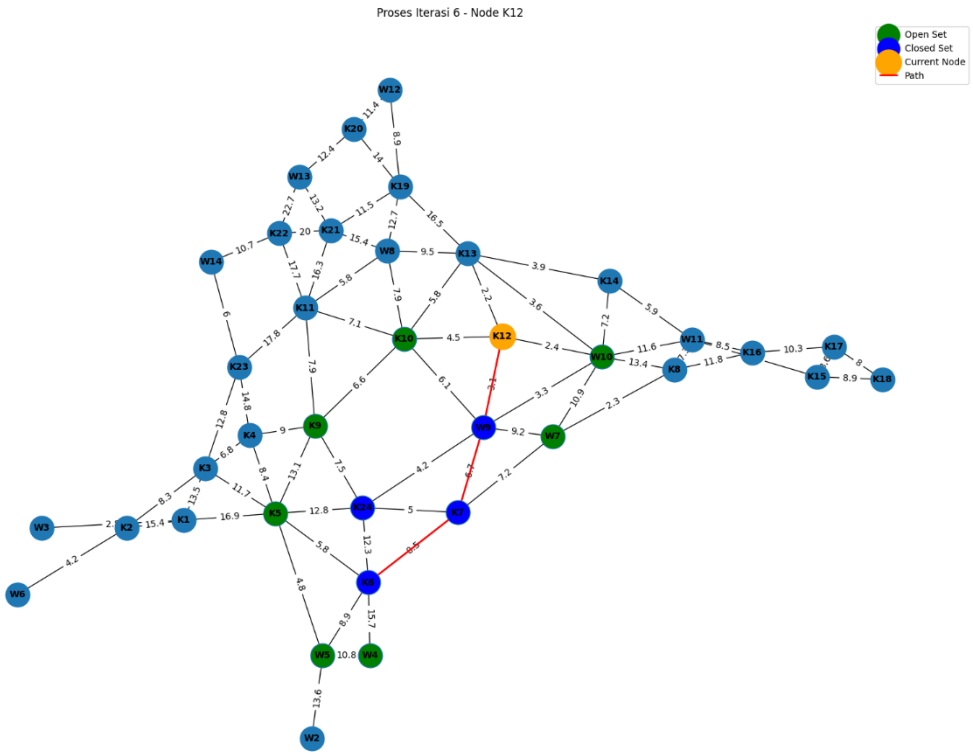
Gambar 4.5 Iterasi 3 Algoritma *A-Star* Standar Rute $\boldsymbol{K6 \rightarrow W8}$



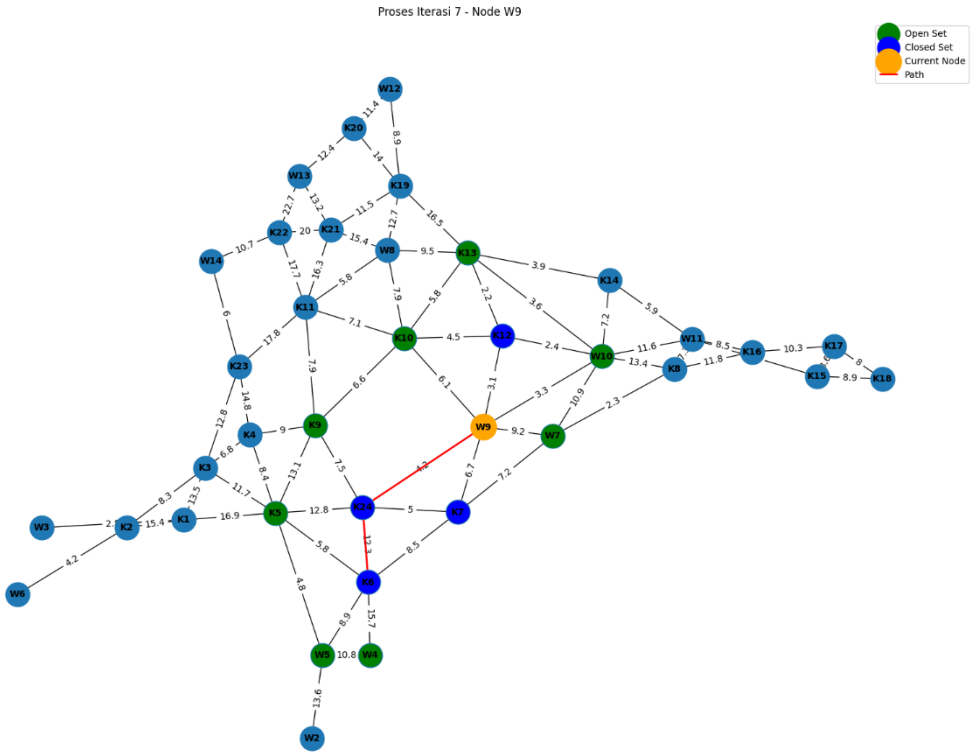
Gambar 4.6 Iterasi 4 Algoritma *A-Star* Standar Rute $\boldsymbol{K6 \rightarrow W8}$



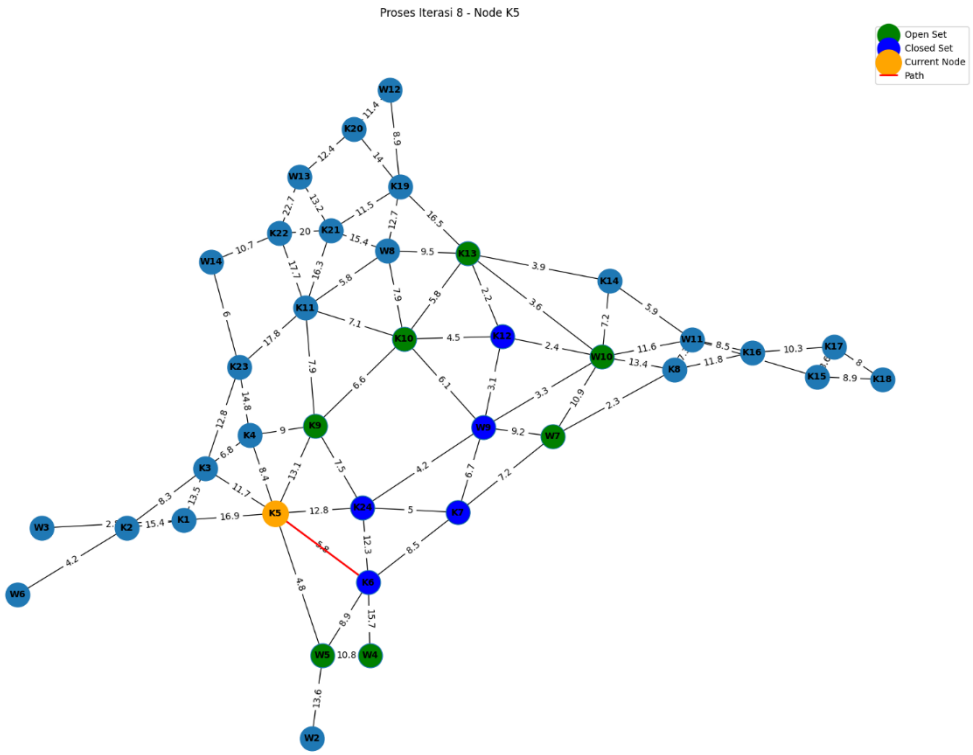
Gambar 4.7 Iterasi 5 Algoritma *A-Star* Standar Rute $\boldsymbol{K6 \rightarrow W8}$



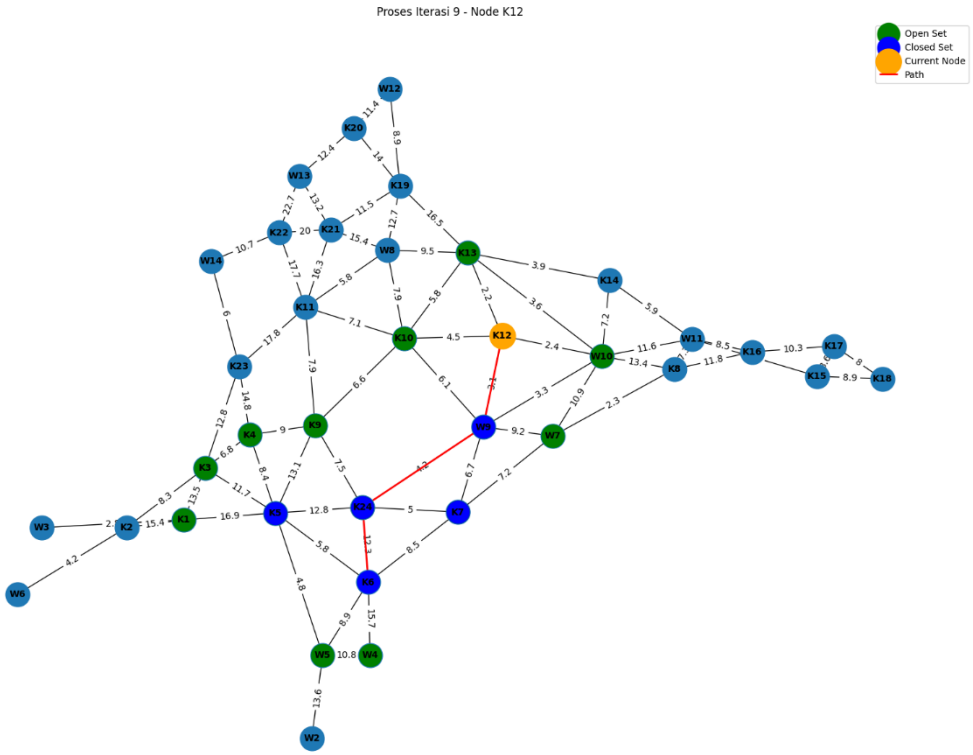
Gambar 4.8 Iterasi 6 Algoritma *A-Star* Standar Rute $\boldsymbol{K6 \rightarrow W8}$



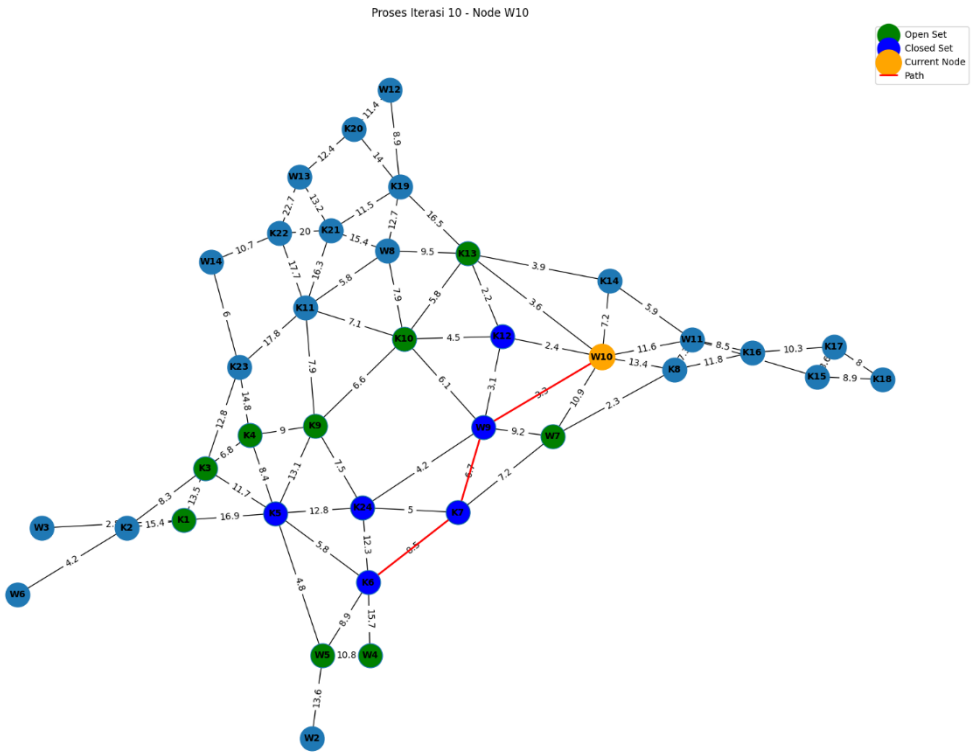
Gambar 4.9 Iterasi 7 Algoritma *A-Star* Standar Rute $\boldsymbol{K6 \rightarrow W8}$



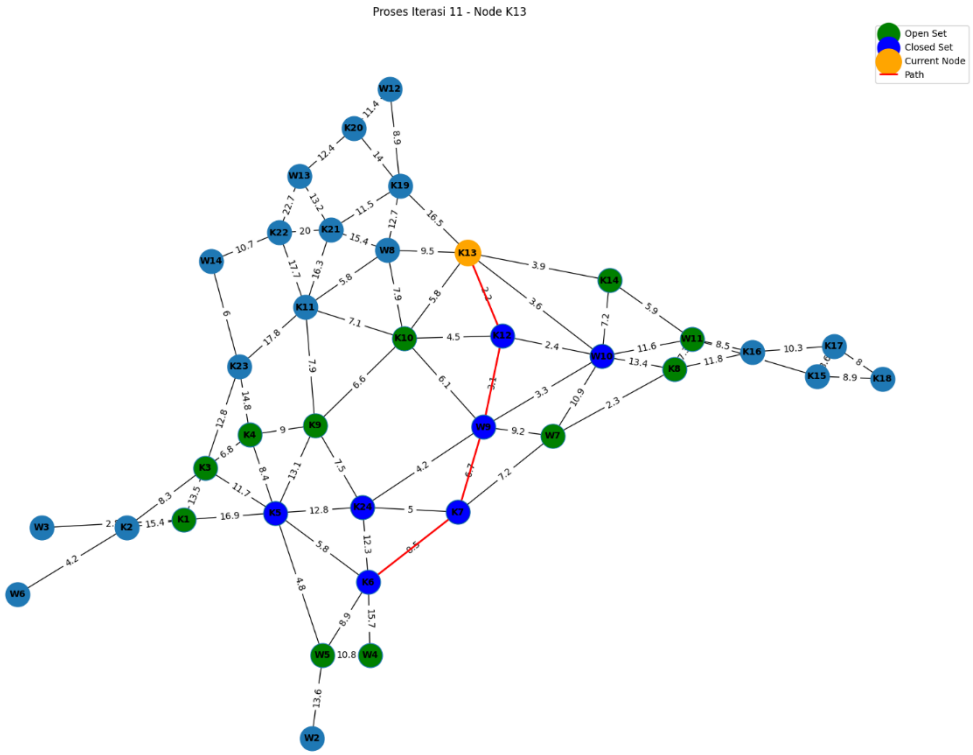
Gambar 4.10 Iterasi 8 Algoritma *A-Star* Standar Rute $\boldsymbol{K6 \rightarrow W8}$



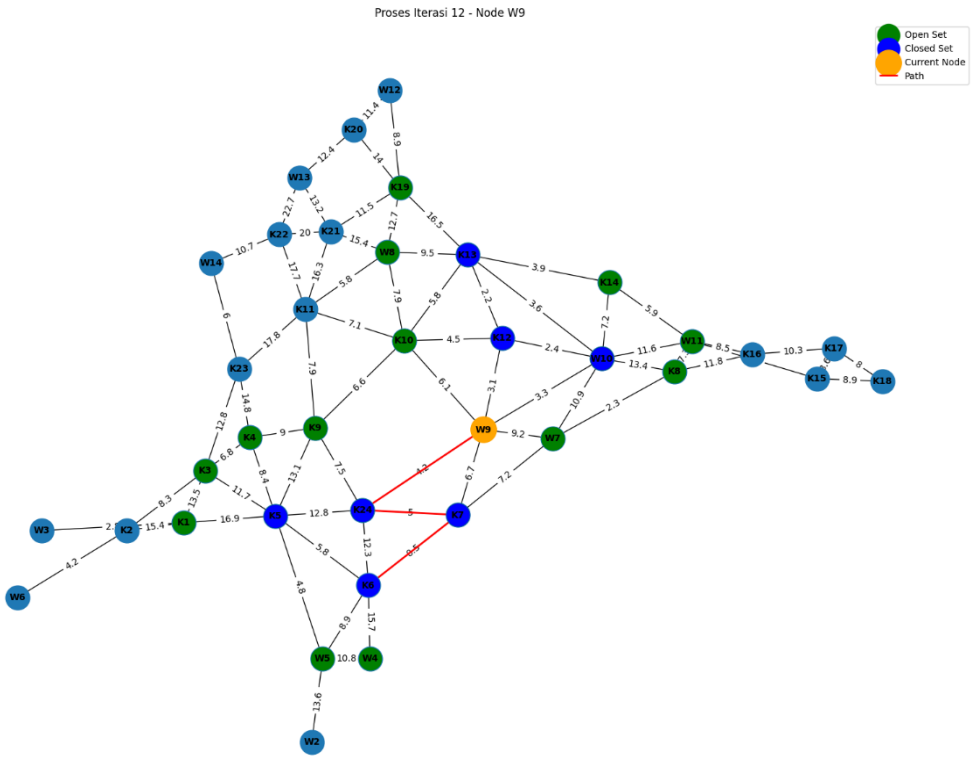
Gambar 4.11 Iterasi 9 Algoritma *A-Star* Standar Rute $\boldsymbol{K6 \rightarrow W8}$



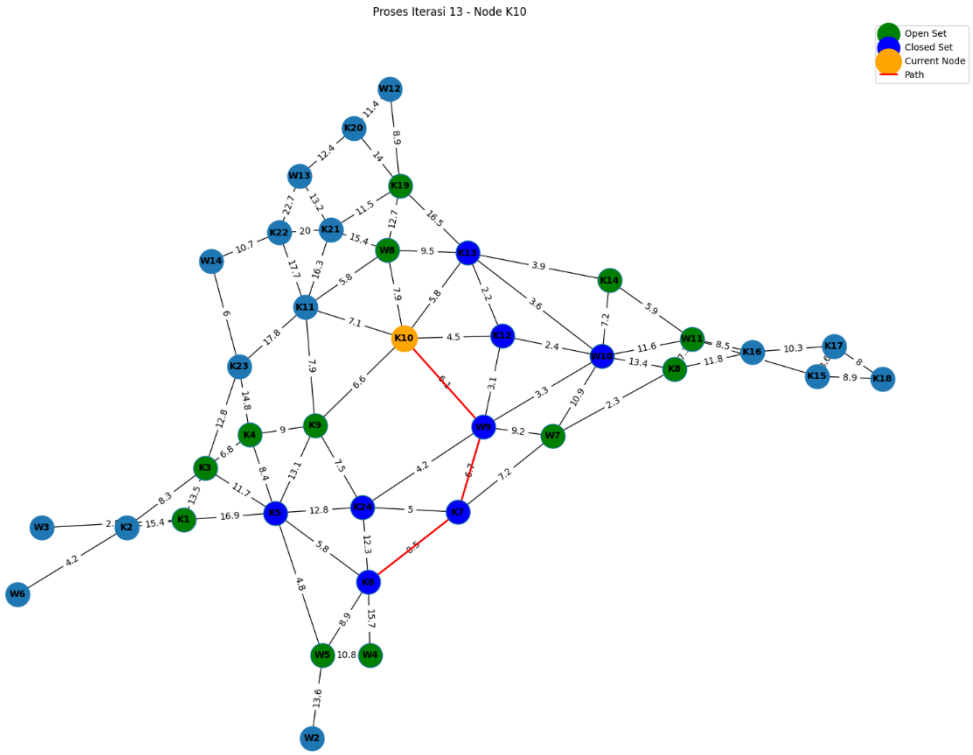
Gambar 4.12 Iterasi 10 Algoritma *A-Star* Standar Rute $\boldsymbol{K6 \rightarrow W8}$



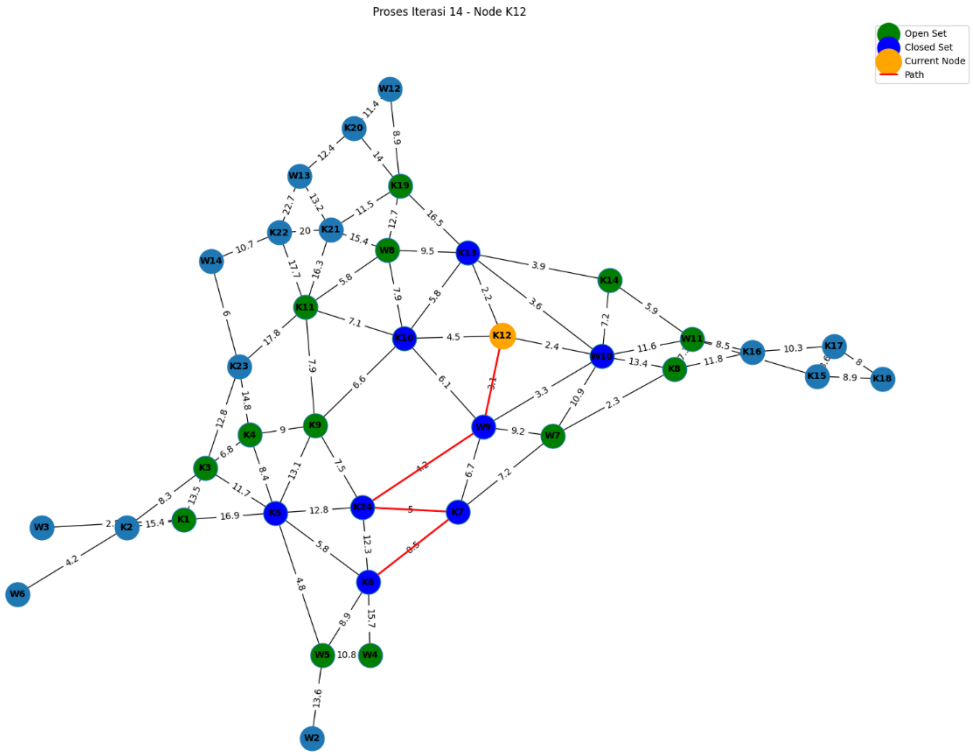
Gambar 4.13 Iterasi 11 Algoritma *A-Star* Standar Rute $\boldsymbol{K6 \rightarrow W8}$



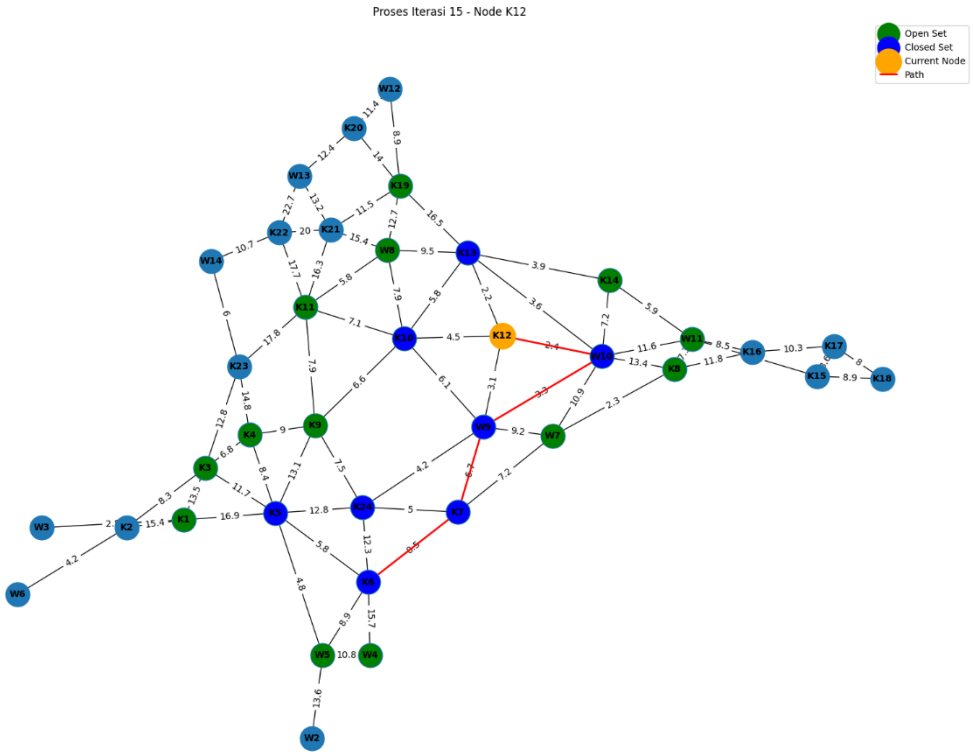
Gambar 4.14 Iterasi 12 Algoritma *A-Star* Standar Rute $\boldsymbol{K6 \rightarrow W8}$



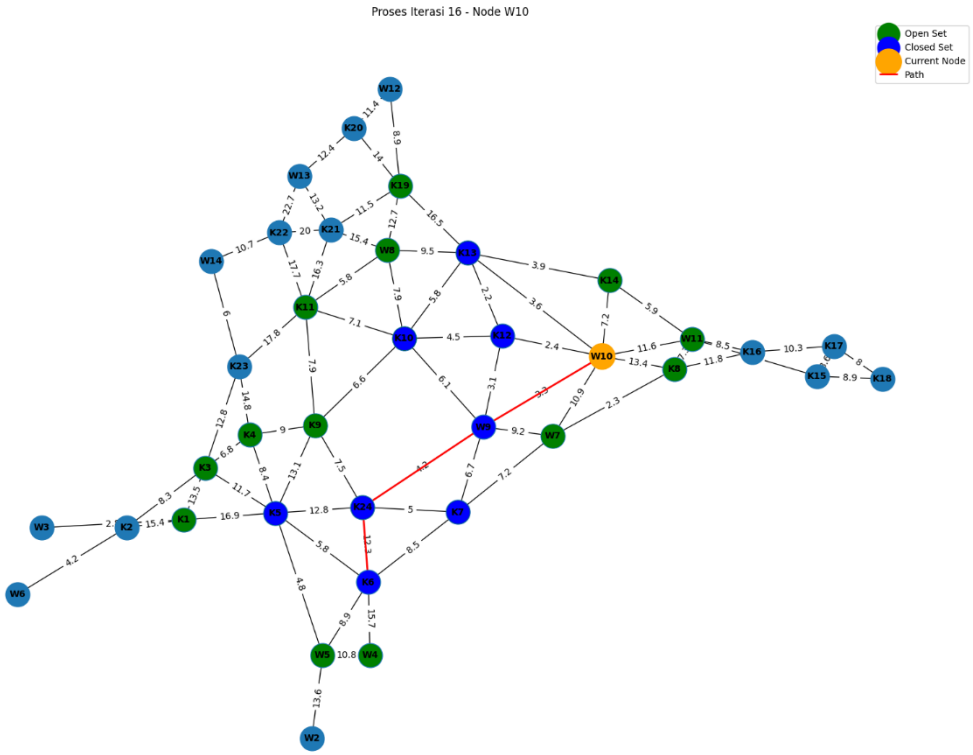
Gambar 4.15 Iterasi 13 Algoritma *A-Star* Standar Rute $\boldsymbol{K6 \rightarrow W8}$



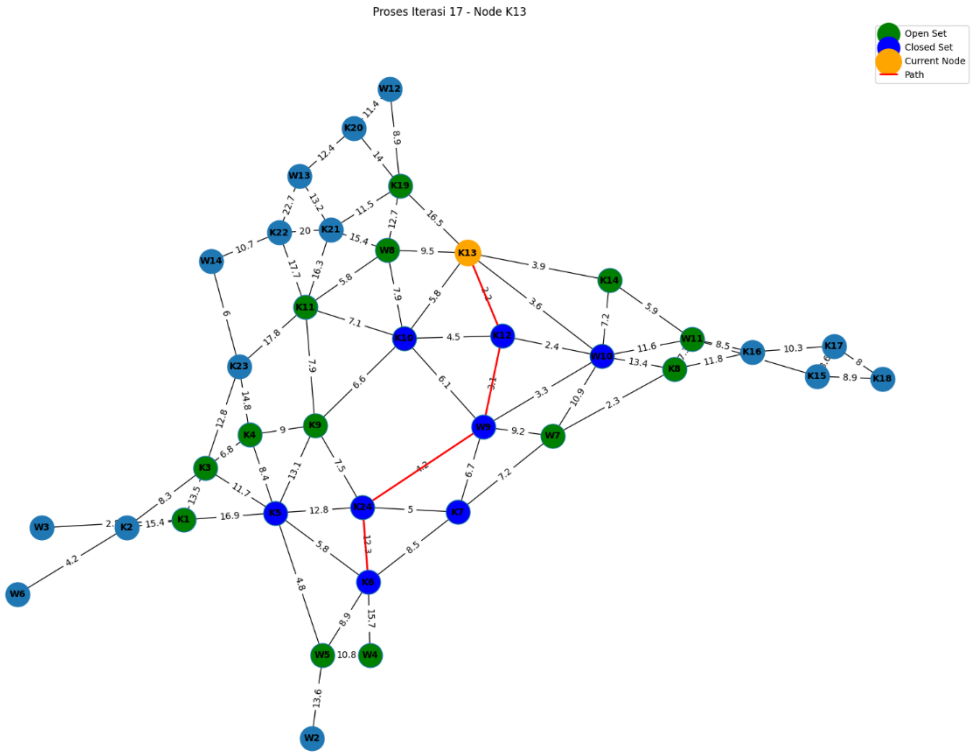
Gambar 4.16 Iterasi 14 Algoritma *A-Star* Standar Rute $\boldsymbol{K6 \rightarrow W8}$



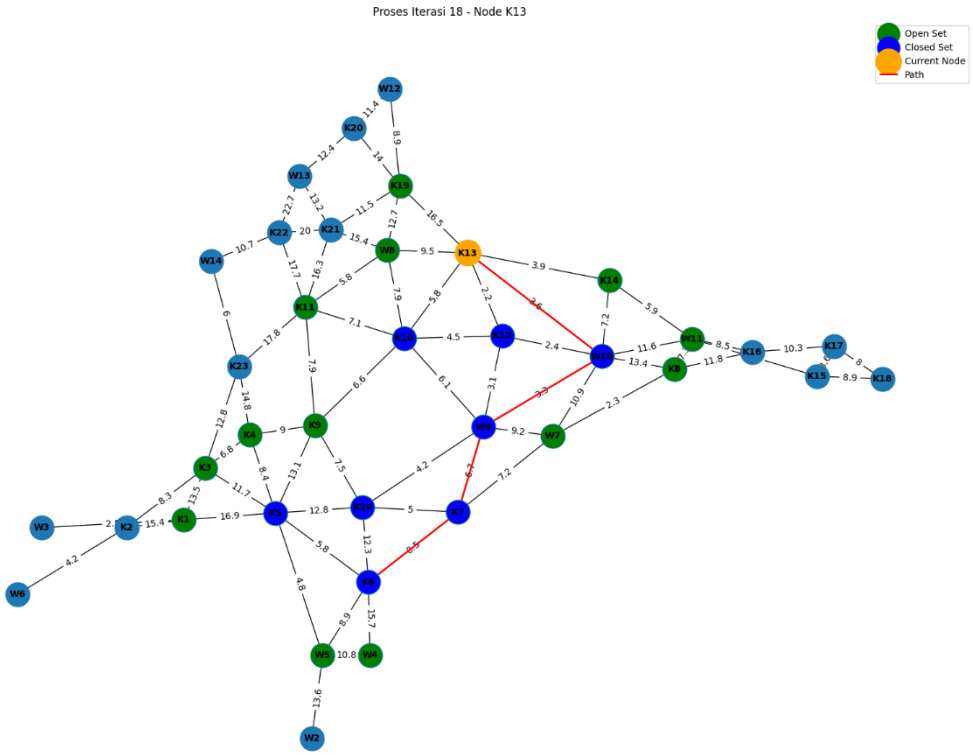
Gambar 4.17 Iterasi 15 Algoritma *A-Star* Standar Rute $\boldsymbol{K6 \rightarrow W8}$



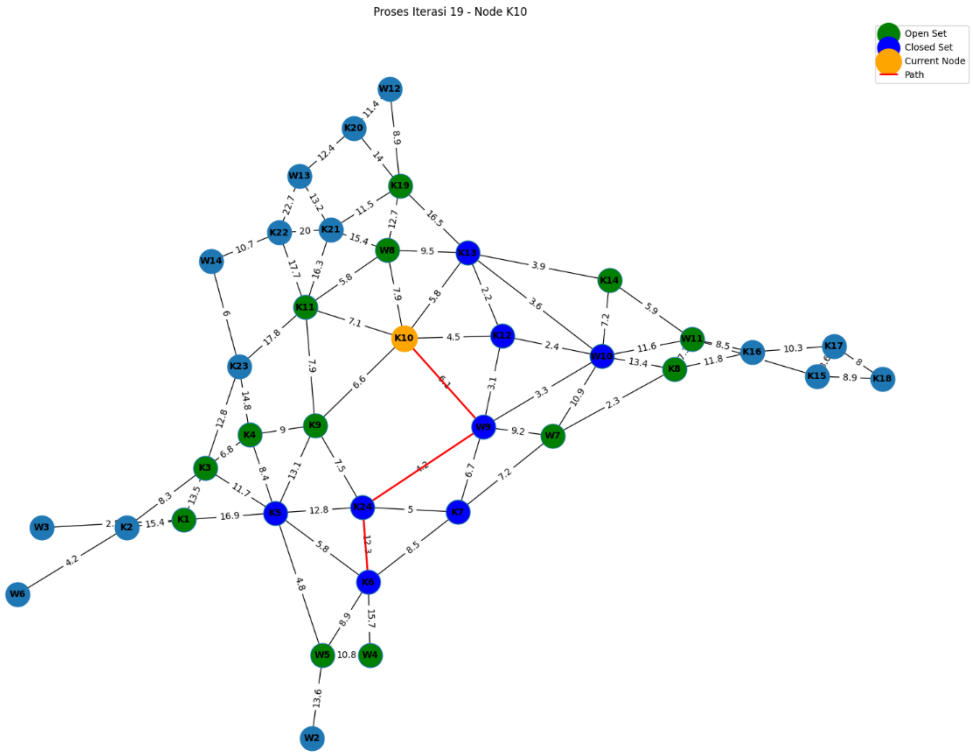
Gambar 4.18 Iterasi 16 Algoritma *A-Star* Standar Rute $\boldsymbol{K6 \rightarrow W8}$



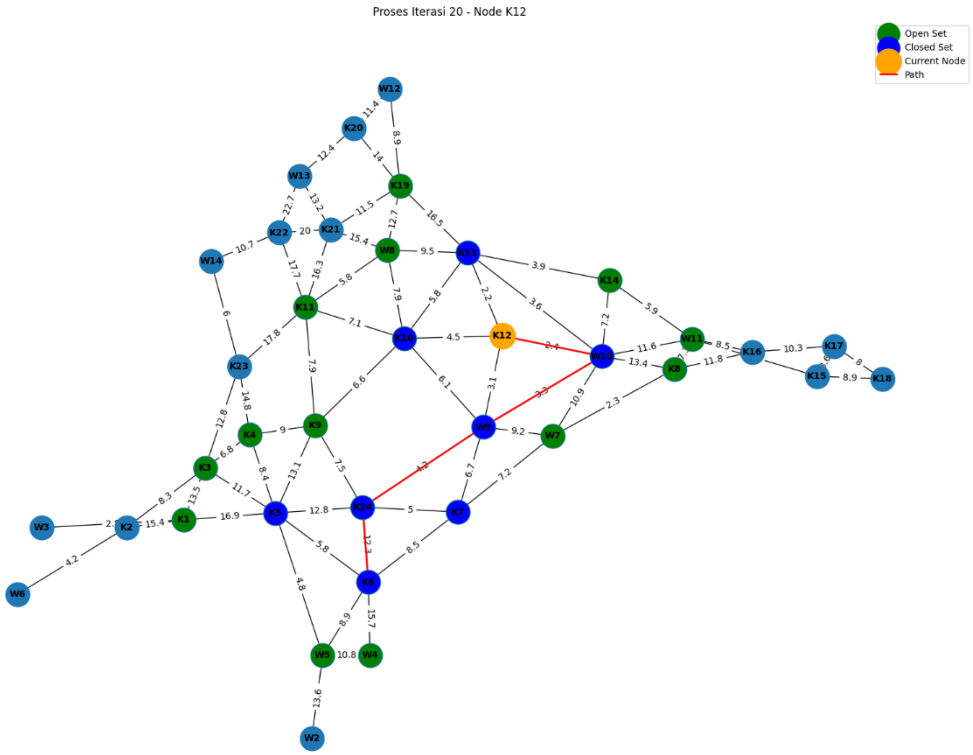
Gambar 4.19 Iterasi 17 Algoritma *A-Star* Standar Rute $\boldsymbol{K6 \rightarrow W8}$



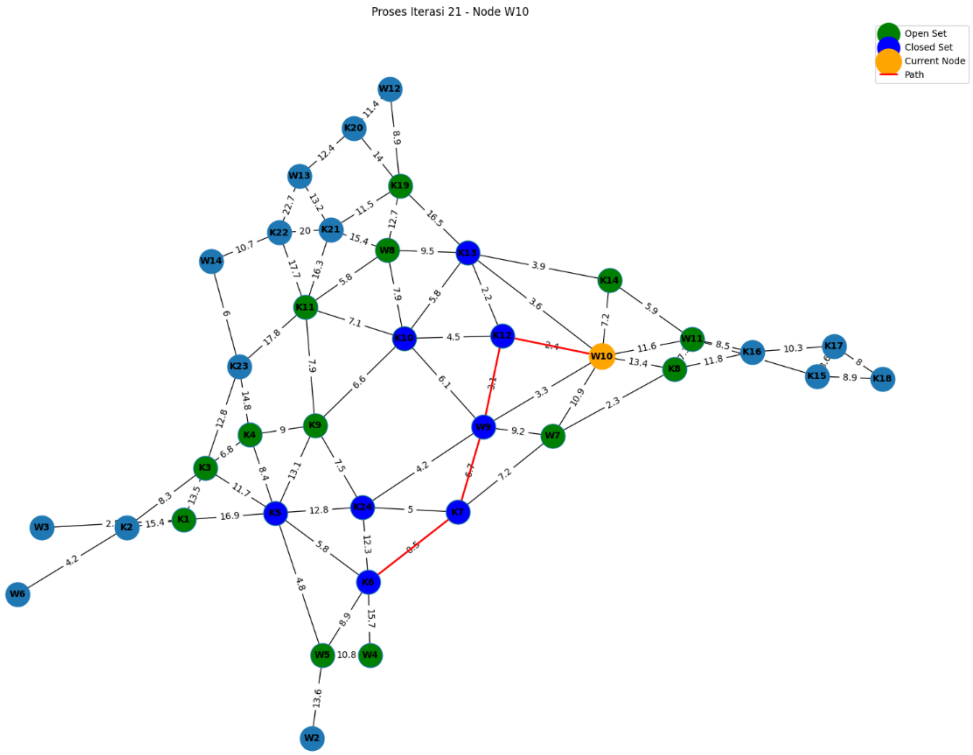
Gambar 4.20 Iterasi 18 Algoritma *A-Star* Standar Rute $\boldsymbol{K6 \rightarrow W8}$



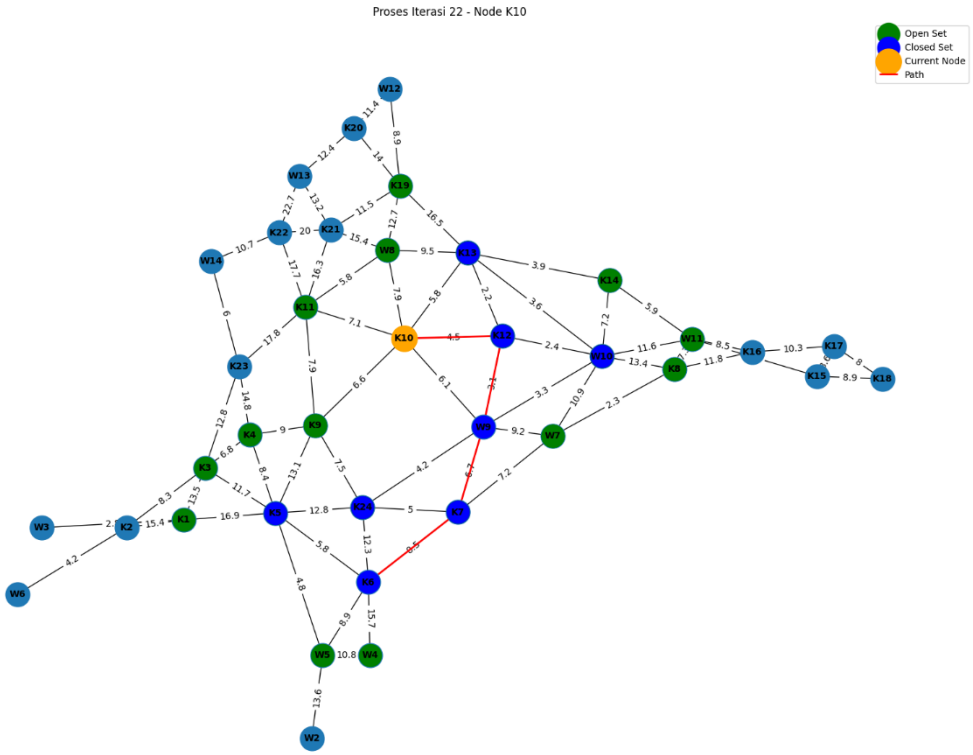
Gambar 4.21 Iterasi 19 Algoritma *A-Star* Standar Rute $\boldsymbol{K6 \rightarrow W8}$



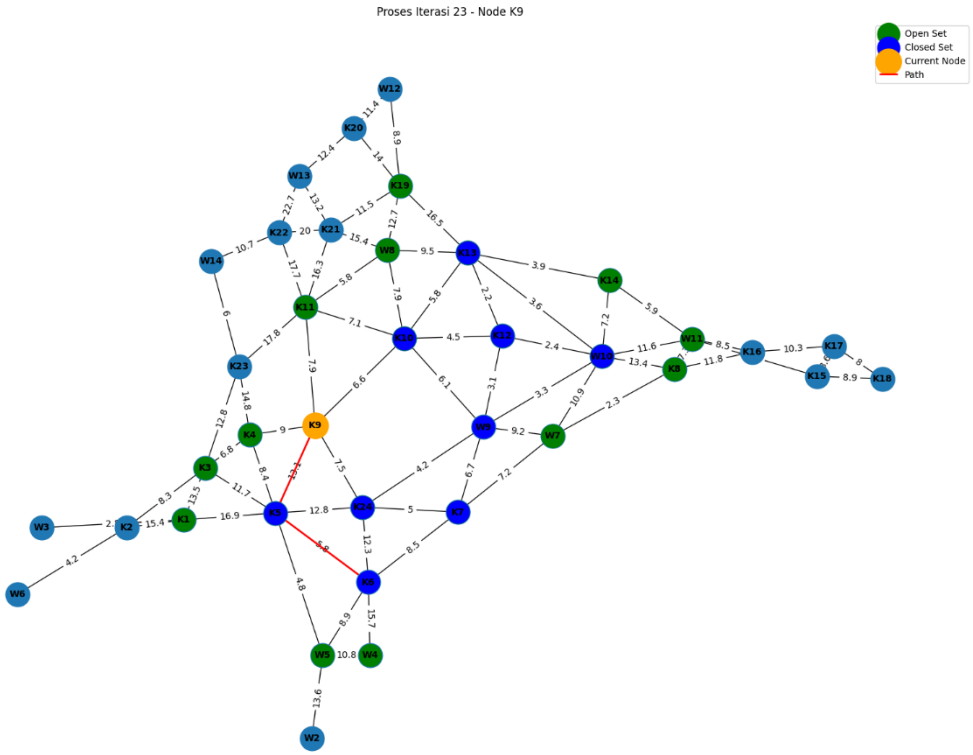
Gambar 4.22 Iterasi 20 Algoritma *A-Star* Standar Rute $\boldsymbol{K6 \rightarrow W8}$



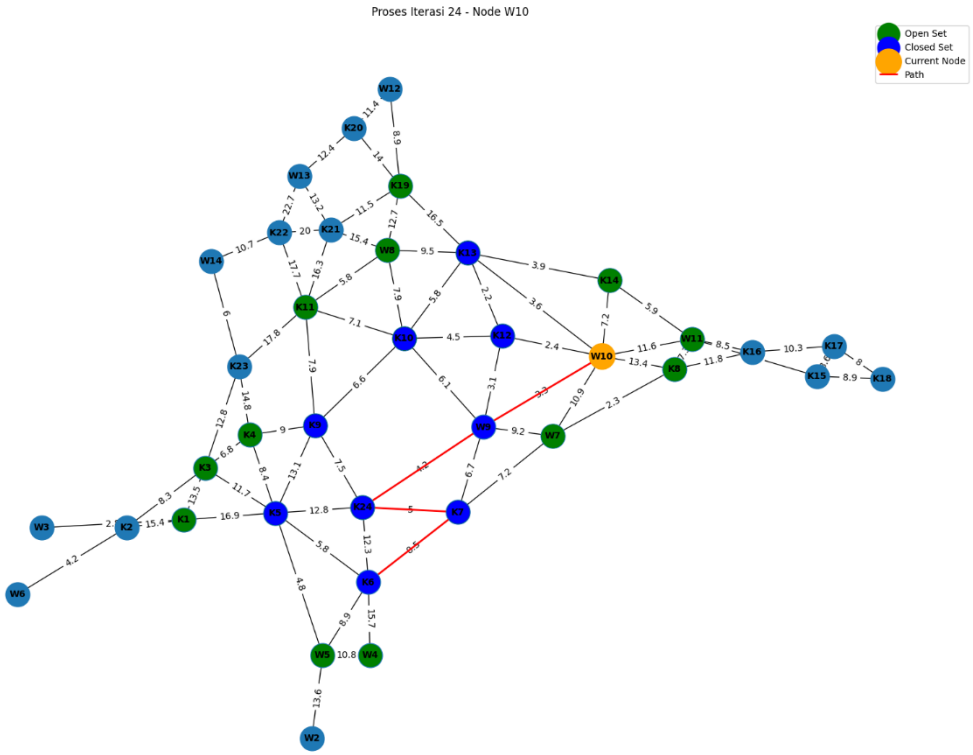
Gambar 4.23 Iterasi 21 Algoritma *A-Star* Standar Rute $\boldsymbol{K6 \rightarrow W8}$



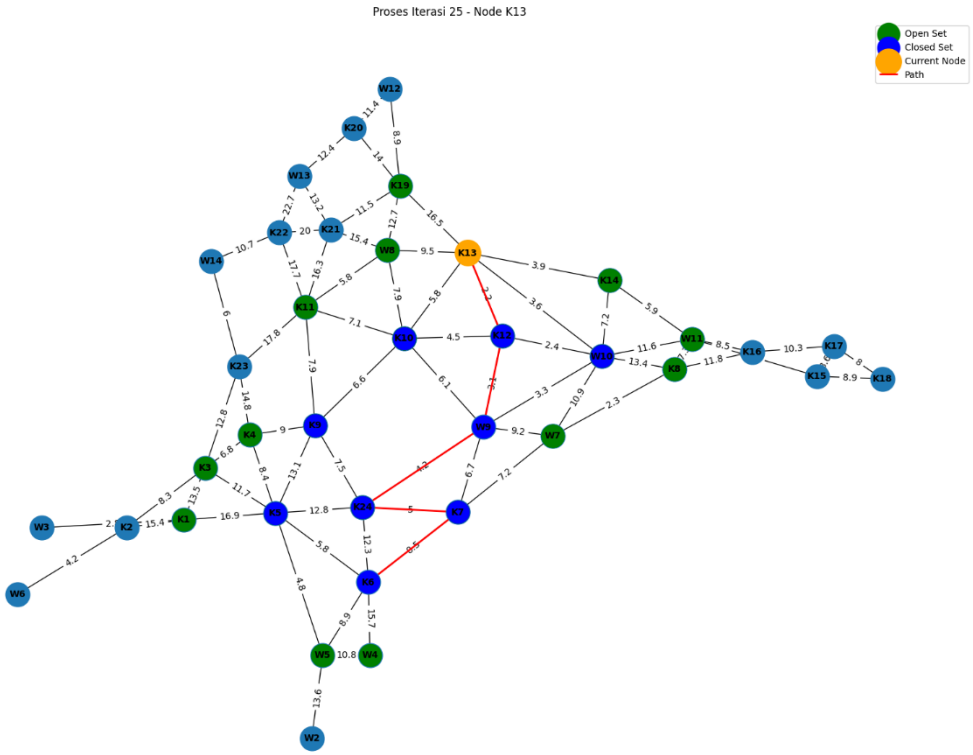
Gambar 4.24 Iterasi 22 Algoritma *A-Star* Standar Rute $\boldsymbol{K6 \rightarrow W8}$



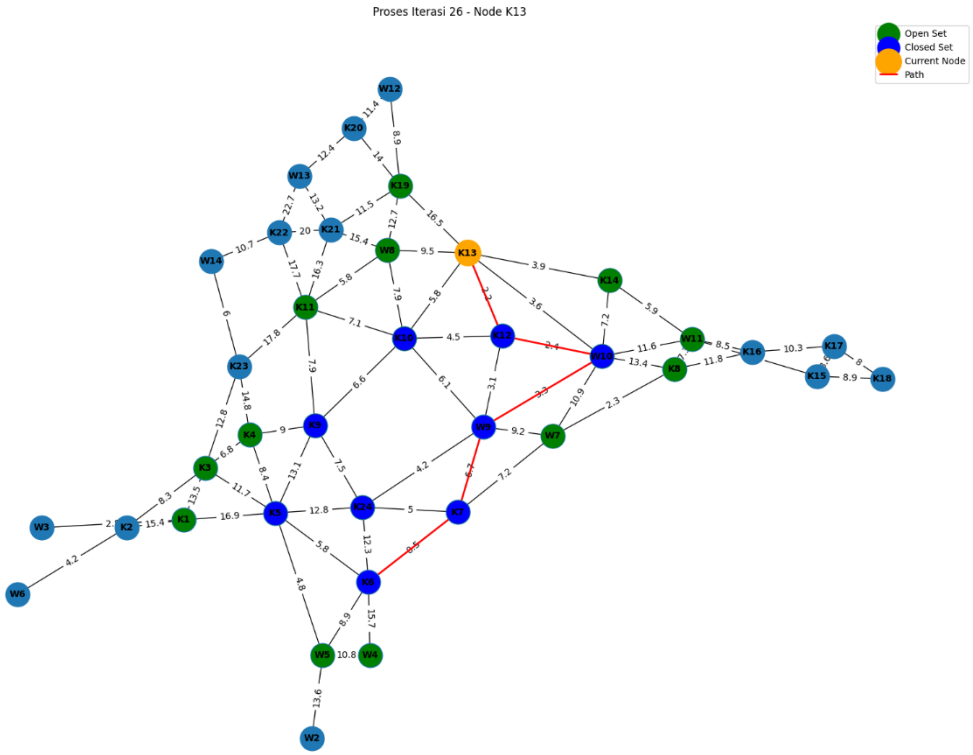
Gambar 4.25 Iterasi 23 Algoritma *A-Star* Standar Rute $\boldsymbol{K6 \rightarrow W8}$



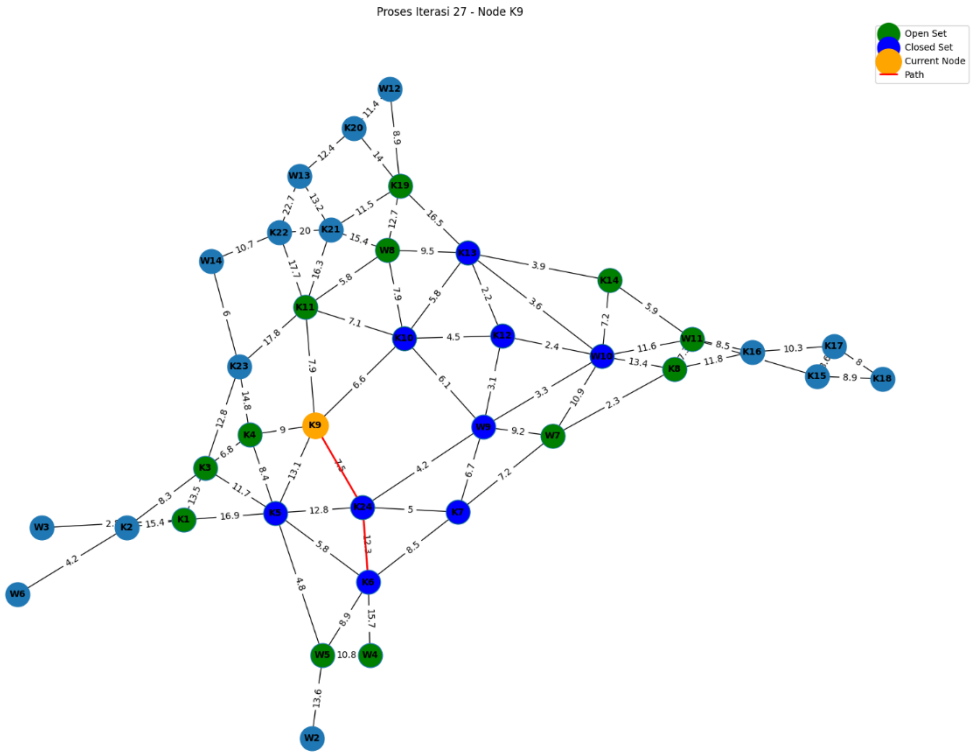
Gambar 4.26 Iterasi 24 Algoritma *A-Star* Standar Rute $\boldsymbol{K6 \rightarrow W8}$



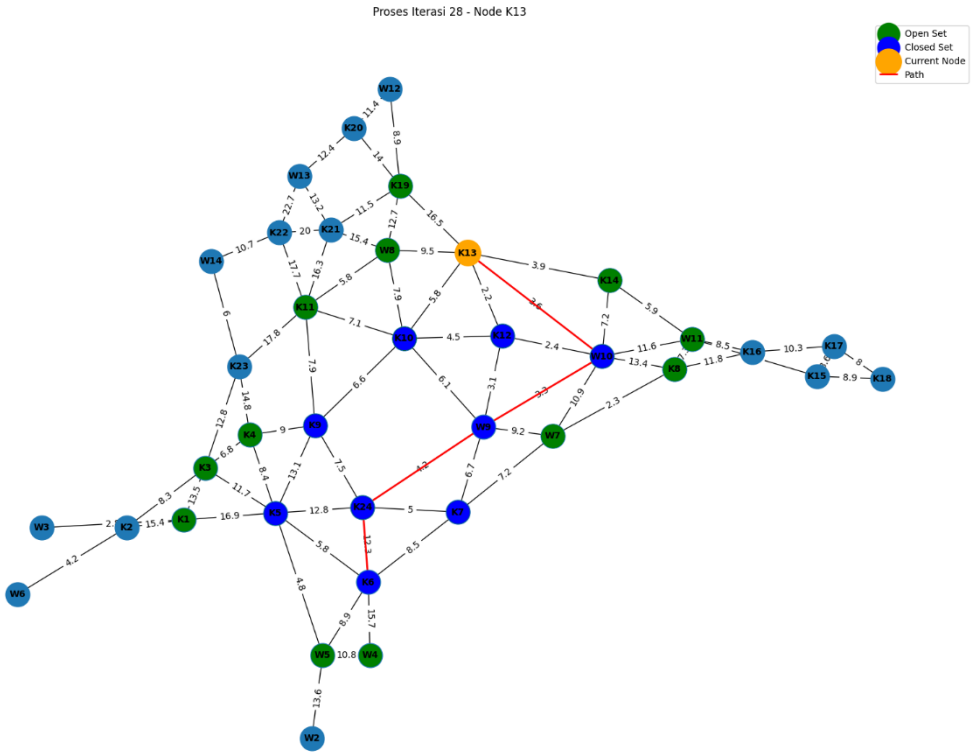
Gambar 4.27 Iterasi 25 Algoritma *A-Star* Standar Rute $\boldsymbol{K6 \rightarrow W8}$



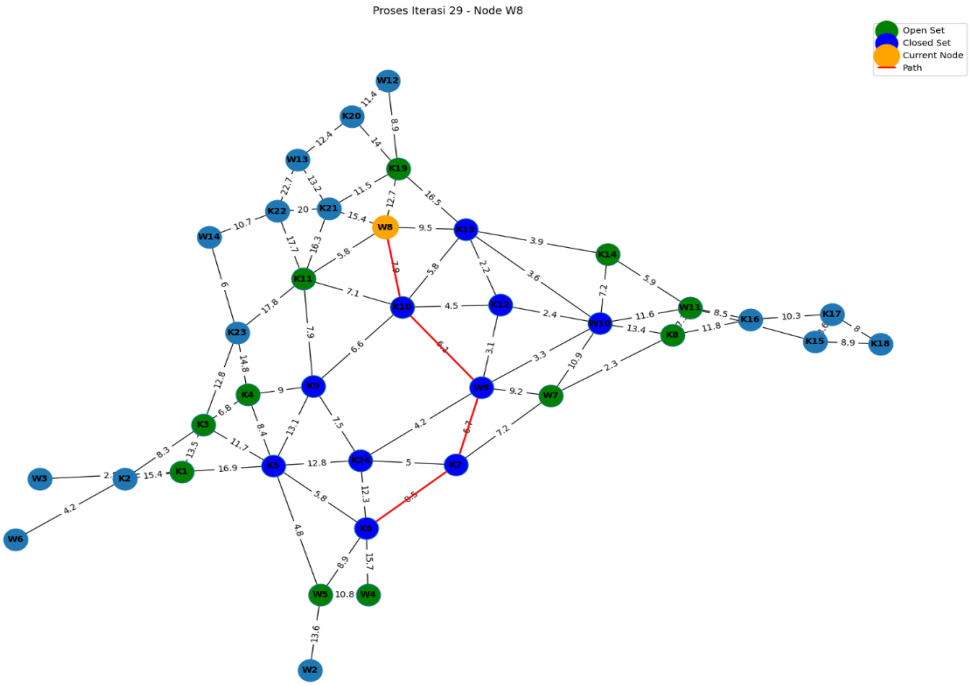
Gambar 4.28 Iterasi 26 Algoritma *A-Star* Standar Rute $\boldsymbol{K6 \rightarrow W8}$



Gambar 4.29 Iterasi 27 Algoritma *A-Star* Standar Rute $\boldsymbol{K6 \rightarrow W8}$

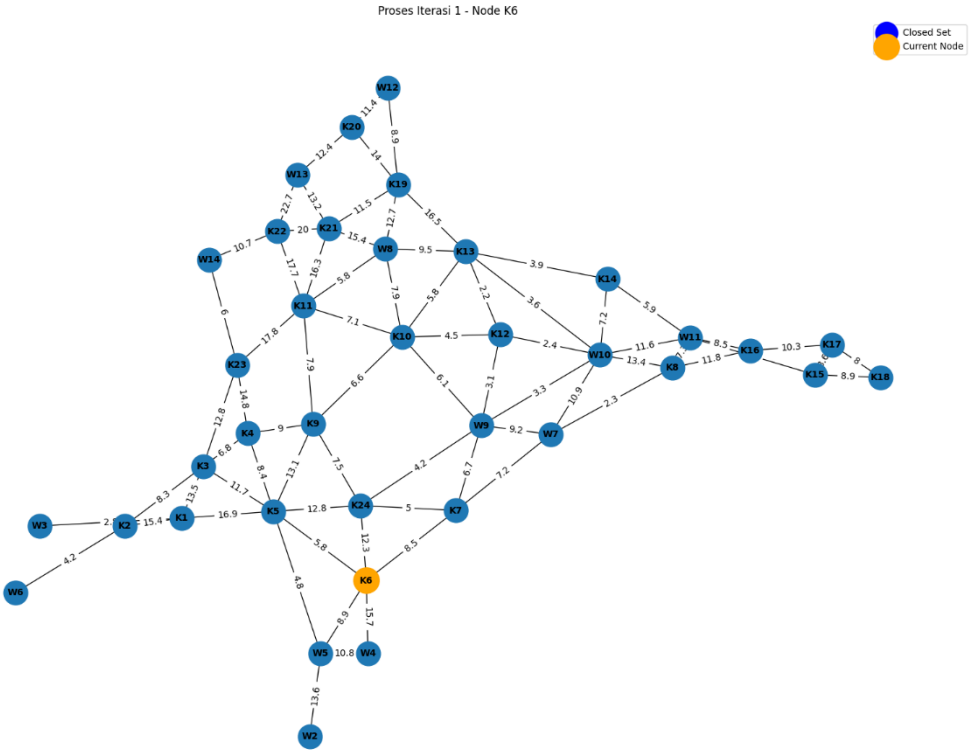


Gambar 4.30 Iterasi 28 Algoritma *A-Star* Standar Rute $\boldsymbol{K6 \rightarrow W8}$

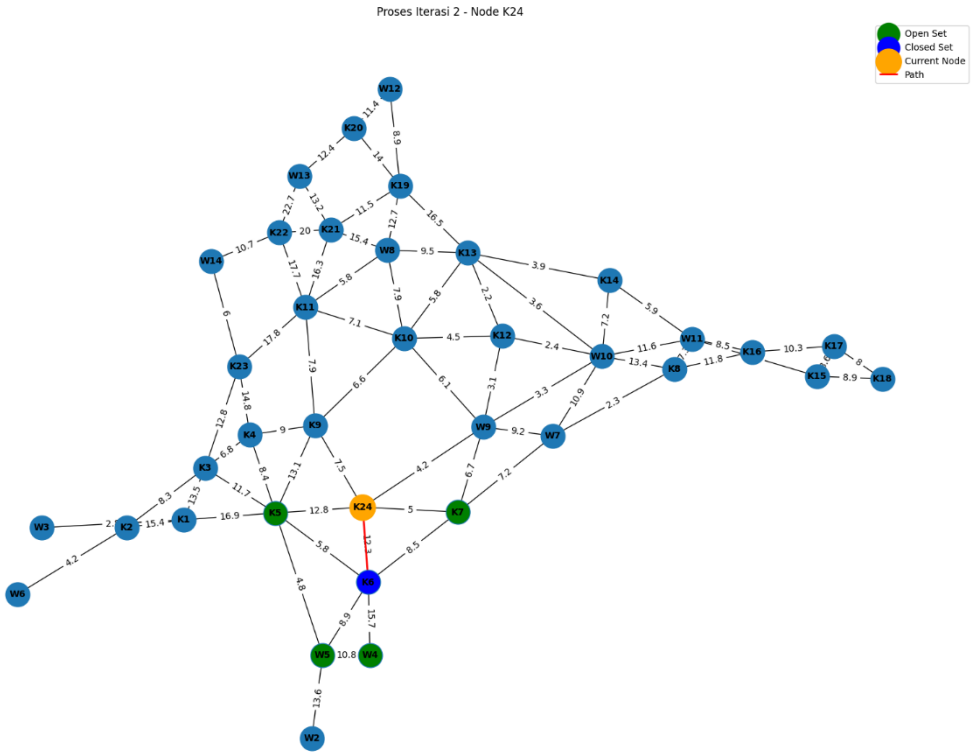


Gambar 4.31 Rute $\boldsymbol{K6 \rightarrow W8}$ Algoritma *A-Star* Standar ditemukan

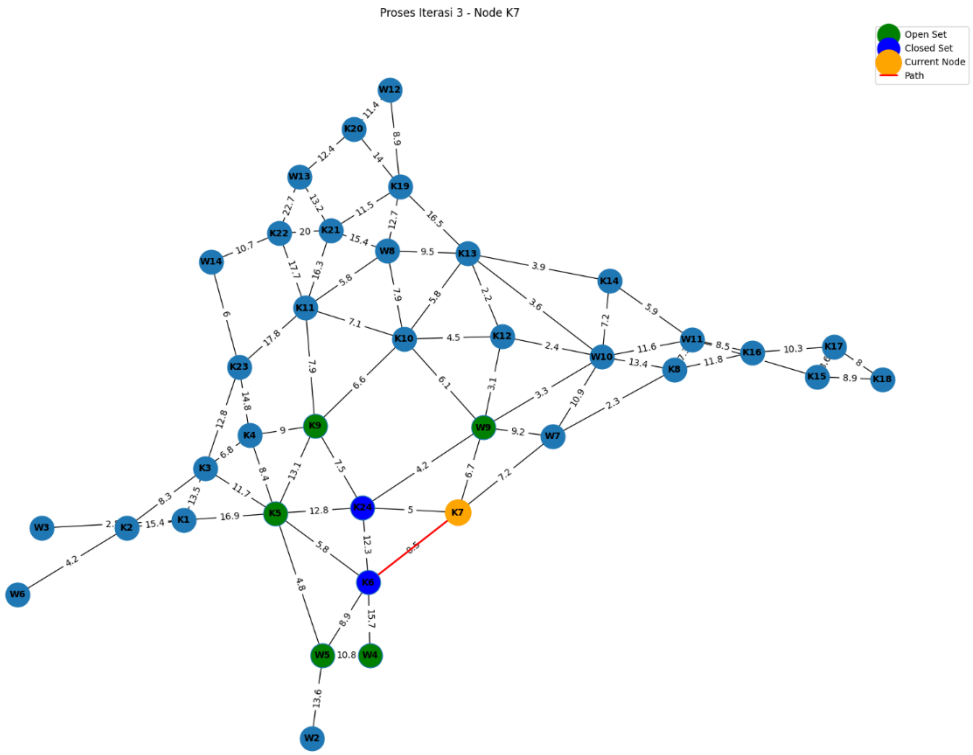
Untuk visualisasi modifikasi algoritma *A-Star* sebagai berikut:



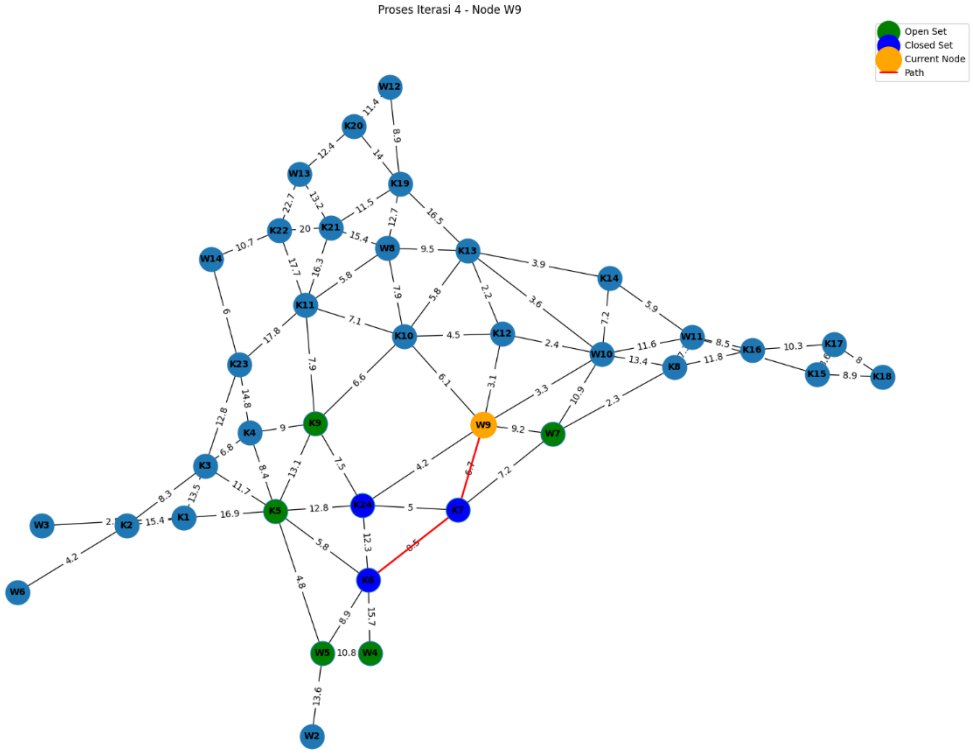
Gambar 4.32 Iterasi 1 Modifikasi Algoritma *A-Star* Rute $\boldsymbol{K6 \rightarrow W8}$



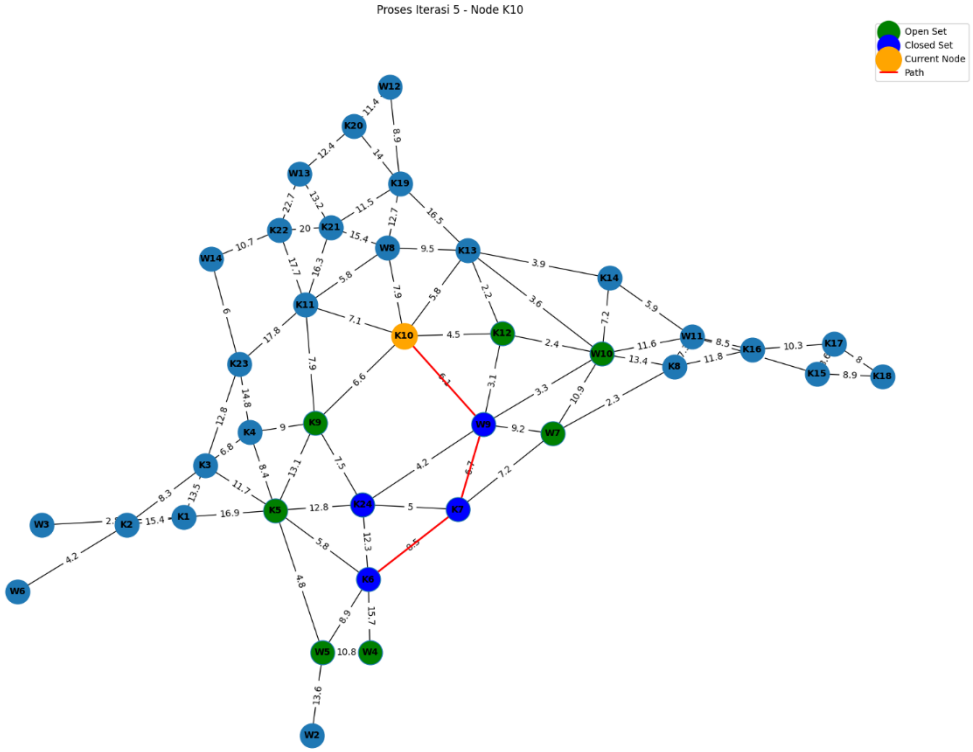
Gambar 4.33 Iterasi 2 Modifikasi Algoritma *A-Star* Rute $\boldsymbol{K6 \rightarrow W8}$



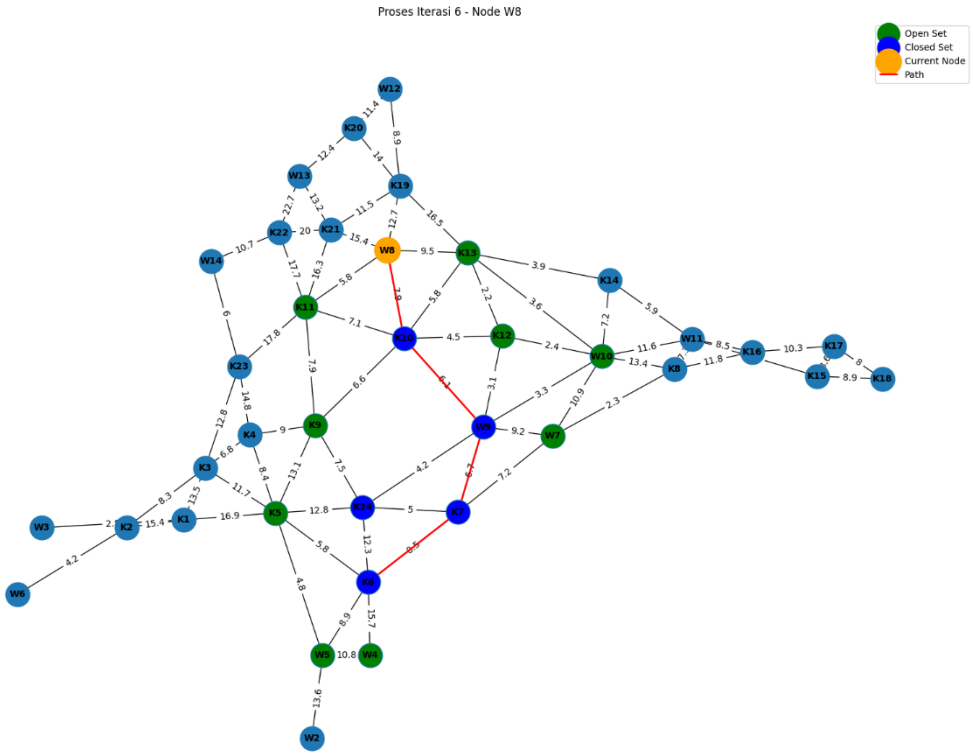
Gambar 4.34 Iterasi 3 Modifikasi Algoritma *A-Star* Rute $\boldsymbol{K6 \rightarrow W8}$



Gambar 4.35 Iterasi 4 Modifikasi Algoritma *A-Star* Rute $\boldsymbol{K6 \rightarrow W8}$



Gambar 4.36 Iterasi 5 Modifikasi Algoritma *A-Star* Rute $\boldsymbol{K6 \rightarrow W8}$



Gambar 4.37 Modifikasi Algoritma *A-Star* Rute $\boldsymbol{K6 \rightarrow W8}$ ditemukan

Berikut hasil pencarian rute terpendek algoritma *A-Star* tradisional dan modifikasi algoritma *A-Star* menggunakan program *Jupyter Notebook*:

Tabel 4.5 Hasil Pencarian Rute Terpendek Algoritma *A-Star* Tradisional

Rute	Update Pencarian	Rute Terpendek	Jarak	Jumlah Iterasi
$W10 \rightarrow W8$	6	W10→K13→W8	29.2 km	8
$K18 \rightarrow W8$	5	K18→K15→W11→K14→K13→ W8	37.6 km	10
$W7 \rightarrow W8$	19	W7→W9→K10→W8	43.5 km	23
$W4 \rightarrow W8$	27	W4→K6→K7→W9→K10→W8	38.1 km	30
$K6 \rightarrow W8$	5	K6→K7→W9→K10→W8	29.2 km	28
$K15 \rightarrow W8$	7	K15→W11→K14→K13→W8	27.3 km	6
$K7 \rightarrow W8$	13	K7→W9→K10→W8	20.7 km	16
$K20 \rightarrow W10$	2	K20→K19→K13→W10	34.1 km	5

Rute	Update Pencarian	Rute Terpendek	Jarak	Jumlah Iterasi
$K2 \rightarrow W10$	3	K2→K3→K4→K9→K10→K12→ W10	37.6 km	9
$W4 \rightarrow W10$	5	W4→K6→K7→W7→W10	35.5 km	8
$K4 \rightarrow W10$	0	K4→K9→K10→K12→W10	22.5 km	4
$K17 \rightarrow W10$	2	K17→K15→W11→W10	28.5 km	6
$K11 \rightarrow W10$	0	K11→K10→K12→W10	14 km	3
$K6 \rightarrow W10$	5	K6→K7→W7→W10	26.6 km	7
$K5 \rightarrow W10$	6	K5→K9→K10→K12→W10	26.6 km	10
$K18 \rightarrow W10$	1	K18→K15→W11→W10	28.5 km	4
$K22 \rightarrow W5$	5	K22→K11→K9→K5→W5	43.5 km	9
$K2 \rightarrow W5$	3	K2→K3→K5→W5	24.8 km	5
$K4 \rightarrow W5$	0	K4→K5→W5	13.2 km	2
$K11 \rightarrow W5$	2	K11→K9→K5→W5	25.8 km	4

Tabel 4.6 Hasil Pencarian Rute Terpendek Modifikasi Algoritma *A-Star*

Rute	Update Pencarian	Rute Terpendek	Jarak	Jumlah Iterasi
$W10 \rightarrow W8$	1	W10→K13→W8	29.2 km	3
$K18 \rightarrow W8$	6	K18→K15→W11→K14→K13→ W8	37.6 km	10
$W7 \rightarrow W8$	1	W7→W9→K10→W8	43.5 km	4
$W4 \rightarrow W8$	1	W4→K6→K7→W9→K10→W8	38.1 km	6
$K6 \rightarrow W8$	1	K6→K7→W9→K10→W8	29.2 km	5
$K15 \rightarrow W8$	2	K15→W11→K14→K13→W8	27.3 km	6
$K7 \rightarrow W8$	0	K7→W9→K10→W8	20.7 km	2
$K20 \rightarrow W10$	2	K20→K19→K13→W10	34.1 km	5

Rute	Update Pencarian	Rute Terpendek	Jarak	Jumlah Iterasi
$K2 \rightarrow W10$	1	K2→K3→K4→K9→K10→K12→ W10	37.6 km	8
$W4 \rightarrow W10$	2	W4→K6→K7→W7→W10	35.5 km	5
$K4 \rightarrow W10$	0	K4→K9→K10→K12→W10	22.5 km	4
$K17 \rightarrow W10$	3	K17→K15→W11→W10	28.5 km	5
$K11 \rightarrow W10$	0	K11→K10→K12→W10	14 km	3
$K6 \rightarrow W10$	2	K6→K7→W7→W10	26.6 km	4
$K5 \rightarrow W10$	1	K5→K9→K10→K12→W10	26.6 km	5
$K18 \rightarrow W10$	0	K18→K15→W11→W10	28.5 km	3
$K22 \rightarrow W5$	4	K22→K11→K9→K5→W5	43.5 km	7
$K2 \rightarrow W5$	1	K2→K3→K5→W5	24.8 km	4
$K4 \rightarrow W5$	0	K4→K5→W5	13.2 km	2
$K11 \rightarrow W5$	0	K11→K9→K5→W5	25.8 km	3

Dari hasil pengujian pencarian rute terpendek sebanyak dua puluh kali didapatkan hasil bahwa modifikasi algoritma *A-Star* lebih cepat dalam menentukan rute terpendek kita lihat dalam banyak iterasi pada masing-masing algoritma. Presentase minimum iterasi/ efektifitas pencarian rute didapatkan melalui:

$$nilai\ efektifitas = \frac{ut - um}{ut} \times 100\%$$

Dimana variabel ut mewakili jumlah iterasi pencarian rute terpendek algoritma *A-Star* tradisional dan um mewakili jumlah iterasi pencarian rute terpendek modifikasi algoritma *A-Star*. Berikut hasil perhitungan presentase nilai efektifitas:

Tabel 4.7 Presentase Hasil Efektifitas Minimum Iterasi

Pengujian ke-	**Perhitungan**	**Hasil**
1	$\frac{8-3}{8} \times 100\%$	62.5%
2	$\frac{10-10}{10} \times 100\%$	0%
3	$\frac{5-3}{5} \times 100\%$	82.6%
4	$\frac{30-6}{30} \times 100\%$	80%
5	$\frac{28-5}{28} \times 100\%$	82.1%
6	$\frac{6-6}{6} \times 100\%$	0%
7	$\frac{16-2}{16} \times 100\%$	87.5%
8	$\frac{5-5}{5} \times 100\%$	0%
9	$\frac{9-8}{9} \times 100\%$	11.1%
10	$\frac{8-5}{8} \times 100\%$	37.5%
11	$\frac{4-4}{4} \times 100\%$	0%
12	$\frac{6-5}{6} \times 100\%$	16.7%
13	$\frac{3-3}{3} \times 100\%$	0%
14	$\frac{7-4}{7} \times 100\%$	42.9%
15	$\frac{10-5}{10} \times 100\%$	50%
16	$\frac{4-3}{4} \times 100\%$	25%
17	$\frac{9-7}{9} \times 100\%$	22.2%
18	$\frac{5-4}{5} \times 100\%$	20%
19	$\frac{2-2}{2} \times 100\%$	0%
20	$\frac{4-3}{4} \times 100\%$	25%
Rata-rata		32.3%

Algoritma *A-Star* memiliki keunggulan hampir selalu dapat menemukan rute terpendek karena melakukan proses update pencarian apabila terdapat nilai evaluasi $f(n)$ yang lebih minimum. Akan tetapi peneliti menemukan sedikit celah pada algoritma *A-Star* ini, dimana terkadang terlalu banyaknya proses pencarian rute atau iterasi tersebut, sehingga peneliti memodifikasi dengan tujuan menghemat proses iterasi. Dari hasil presentase hasil efektifitas modifikasi algoritma *A-Star* didapat bahwa algoritma ini lebih hemat/ cepat 32.3% dalam proses pencarian atau iterasi dalam penentuan rute terpendek dari algoritma *A-Star* tradisional.

BAB V
PENUTUP

5.1 Kesimpulan

Berdasarkan penelitian yang telah dilakukan dapat disimpulkan bahwa modifikasi algoritma *A-Star* memungkinkan menggantikan algoritma *A-Star* tradisional. Algoritma *A-Star* memiliki keunggulan dalam pencarian rute terpendek hampir selalu dapat menemukan jarak minimum. Tetapi dalam prosesnya terkadang terlalu banyak proses iterasi sehingga peneliti memodifikasi algoritma *A-Star* dengan data sampel titik lokasi wisata Blitar dengan menggunakan manual sebagai langkah-langkah penerapan algoritma, dan menggunakan program sebagai uji perbandingan kedua algoritma tersebut. Dengan melakukan sebanyak 20 kali pengujian pencarian rute terpendek didapatkan hasil rute dan jarak yang sama akan tetapi modifikasi algoritma *A-Star* lebih hemat 32.3% dalam proses iterasi pencarian dibanding algoritma *A-Star* tradisional.

5.2 Saran

Penelitian ini telah menghasilkan modifikasi baru mengenai modifikasi algoritma *A-Star*. Dengan penelitian ini diharapkan dapat menjadi pondasi untuk penelitian selanjutnya dalam pencarian rute terpendek. Selain itu diharapkan dalam penelitian ini dapat menjadi pondasi terkait pencarian formula *A-Star* atau algoritma lain dalam memodifikasi sehingga menjadikan algoritma yang lebih optimal seperti membandingan banyak modifikasi fungsi evaluasi $f(n)$ sehingga didapatkan kesimpulan seperti menemukan modifikasi baru yang lebih optimal, atau semisal klaim bahwa modifikasi *A-Star* dengan fungsi evaluasi

$f(n) = g(n) + h(n) * \alpha(n)$ sudah paling optimal pada lingkup algoritma *A-Star.*

DAFTAR PUSTAKA

Al-Syaikh, A. bin M. bin A. bin I. (2004). *Tafsir Ibnu Katsir 4.4.pdf* (pp. 522–523). JpnMuslim. https://archive.org/details/Tafsir_Ibnu_Katsir_Lengkap_114Juz/Tafsir Ibnu Katsir 4.4/page/n31/mode/2up?view=theater

Al-Utsaimin, M. S. bin. (2013). *Syarah Al-Arba'in An-Nawawiyah* (p. 249). UMMUL QURA. https://ia600501.us.archive.org/11/items/indun/indu45.pdf

Anwar, K., & Manuharawati. (2021). Math Unesa. *Jurnal Ilmiah Matematika*, *9*(2), 437–446. https://media.neliti.com/media/publications/249234-model-infeksi-hiv-dengan-pengaruh-percob-b7e3cd43.pdf

Aulia, S. R., Wamiliana, W., Asmiati, A., & Notiragayu, N. (2023). Perbandingan Algoritme Dijkstra dan Algoritme A* (A-Star) dalam Penentuan Lintasan Terpendek dari Dinas Pendidikan Provinsi Lampung ke Beberapa Sekolah Menengah Atas (SMA) Negeri di Provinsi Lampung. *Jurnal Pepadun*, *4*(2), 183–190. https://doi.org/10.23960/pepadun.v4i2.177

Cormen, T. H., Leiserson, C. E., Rivest, R. L., & Stein, C. (2022). Introduction to algorithms, 4 Edition. In *The MIT Press Cambridge, Massachusetts London, England*. http://lccn.loc.gov/2021037260

GANNETT, S. S. (1916). Department of the Interior United States Geological Survey. *Pubs.Usgs.Gov*, *19*, 388. http://pubs.usgs.gov/wri/1985/4075/report.pdf

Juhari, M. S. (2021). Matematika Diskrit. *Pena Persada*, 1–154. http://repository.uin-malang.ac.id/9541/

Kemenag. (2023). *Qur'an Kemenag*.

KEMENAG. (2015). *Anjuran Islam tentang Etos Kerja dan Profesionalisme Sumber: https://islam.nu.or.id/khutbah/anjuran-islam-tentang-etos-kerja-dan-profesionalisme-5ElUf ___ Download NU Online Super App, aplikasi keislaman terlengkap! https://nu.or.id/superapp (Android/iOS)*. https://islam.nu.or.id/khutbah/anjuran-islam-tentang-etos-kerja-dan-profesionalisme-5ElUf

Kluyver, T., Ragan-Kelley, B., Pérez, F., Granger, B., Bussonnier, M., Frederic, J., Kelley, K., Hamrick, J., Grout, J., Corlay, S., Ivanov, P., Avila, D., Abdalla,

S., Willing, C., & Jupyter Development Team. (2016). Jupyter Notebooks—a publishing format for reproducible computational workflows. *Positioning and Power in Academic Publishing: Players, Agents and Agendas - Proceedings of the 20th International Conference on Electronic Publishing, ELPUB 2016*, 87–90. https://doi.org/10.3233/978-1-61499-649-1-87

Lajnah Pentashihan mushaf Al-Qur'an. (n.d.). https://quran.kemenag.go.id/quran/per-ayat/surah/12?from=47&to=49

Maula, A. S. R., Tundo, Adrianto, S., Kastum, & Sutisna, N. (2024). Implementasi Penggunaan Algoritma A* pada Penentuan Jarak Terpendek dari Cilacap ke Yogyakarta. *Jurnal Ilmiah Informatika Komputer*, *29*(1), 73–82. https://doi.org/https://doi.org/10.35760/ik.2024.v29i1.10661

Munir, R. (2023). Graf. *Revisi Buku Matematika Diskrit 2012, Update*.

Pérez, F., & Granger, B. E. (2007). IPython: A system for interactive scientific computing. *Computing in Science and Engineering*, *9*(3), 21–29. https://doi.org/10.1109/MCSE.2007.53

Ropiqoh, & Lubis, R. S. (2023). Implementation of a-Star Algorithm in Finding the Shortest Route of Cooking Oil Distribution in Karo Regency Using Graph. *Mathline*, *8*(2), 725–738. http://doi.org/10.31943/mathline.v8i2.442

Russel, S. J., & Norvig, P. (2010). Artificial intelligence : Moder Approach. In M. J. Horton (Ed.), *2010 The 2nd International Conference on Computer and Automation Engineering, ICCAE 2010* (third edit, Vol. 4). Pearson Education, Inc., Upper Saddle River, New Jersey. https://doi.org/10.1109/ICCAE.2010.5451578

Shen. (2014). Effects of nutritional factors on the development of ethanol induced fatty liver in KK and KK A(y) mice. *Journal of Nutrition*, *105*(10), 1263–1268. https://doi.org/10.1093/jn/105.10.1263

Sumantri, E., & Hidayattullah, S. (2023). Penerapan Algoritma A*Star untuk Mencari Rute Terpendek dari Kemayoran ke Destinasi Monumen Nasional (MONAS). *Jurnal Sains Dan Teknologi*, *5*(2), 673–680. https://doi.org/10.55338/saintek.v5i2.1432

Tarigan, D. H. A. A., & M.Ag. (2012). *Tafsir Ayat Ekonomi* (M. Y. Nasution & A. Grafika (Eds.)). Citapustaka Media Perintis.

Wayahdi, M. R., Ginting, S. H. N., & Syahputra, D. (2021). Greedy, A-Star, and Dijkstra's Algorithms in Finding Shortest Path. *International Journal of Advances in Data and Information Systems*, *2*(1), 45–52. https://doi.org/10.25008/ijadis.v2i1.1206

Yudha, M. H. P., Supian, S., & Napitupulu, H. (2022). Optimalization Route to Tourism Places in West Java Using A-STAR Algorithm. *CAUCHY: Jurnal Matematika Murni Dan Aplikasi*, *7*(3), 464–473. https://doi.org/10.18860/ca.v7i3.17032

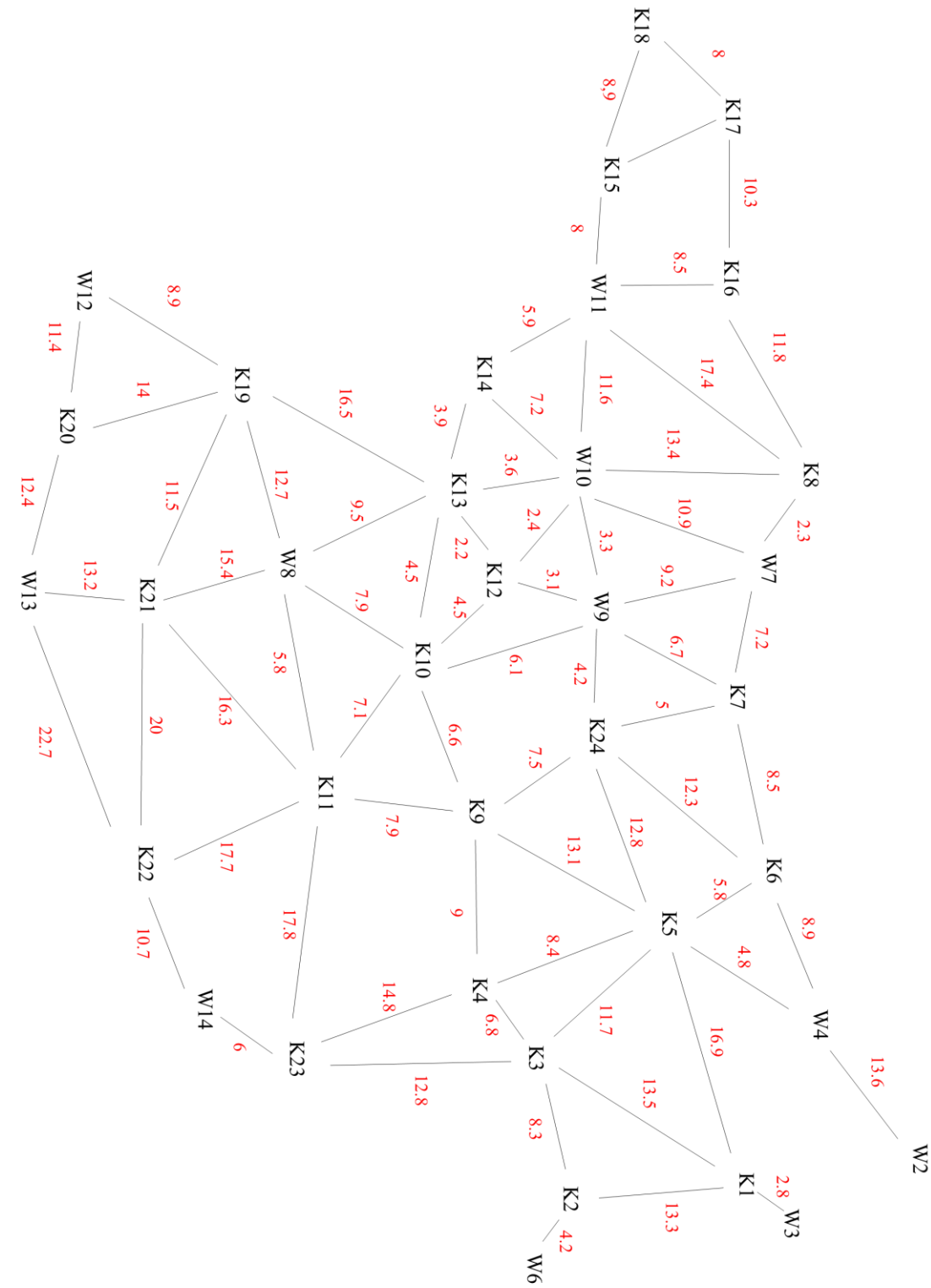
LAMPIRAN

Lampiran 1. Data Jarak Antar Titik

No	Titik		Jarak	No	Titik		Jarak
1	K1	W3	2.8	39	K18	K17	8
2	K2	W6	4.2	40		K15	8.9
3		K1	15.4	41	K19	W8	12.7
4	K3	K2	8.3	42		K21	11.5
5		K1	13.5	43		K20	14
6		K5	11.7	44		W12	8.9
7	K4	K5	8.4	45	K20	W13	12.4
8		K3	6.8	46	K21	K22	20
9	K5	K1	16.9	47		W8	15.4
10		W5	4.8	48		K11	16.3
11		K6	5.8	49	K22	W14	10.7
12	K6	W5	8.9	50		K11	17.7
13		W4	15.7	51	K23	K4	14.8
14	K7	W7	7.2	52		K3	12.8
15		K6	8.5	53	K24	K6	12.3
16		K24	5	54		K5	12.8
17	K8	W7	2.3	55		K9	7.5
18		W10	13.4	56	W5	W4	10.8
19	K9	K5	13.1	57		W2	13.6
20		K4	9	58	W8	K10	7.9
21	K10	K9	6.6	59		K11	5.8
22		W9	6.1	60	W9	K24	4.2
23	K11	K10	7.1	61		K7	6.7

No	Titik		Jarak	No	Titik		Jarak
24		K9	7.9	62		W7	9.2
25		K23	17.8	63		K12	3.1
26	K12	W9	3.1	64	W10	W9	3.3
27		K10	4.5	65		W7	10.9
28	K13	K12	2.2	66		K13	3.6
29		K10	5.8	67		K12	2.4
30		W8	9.5	68	W11	K8	17.3
31		K19	16.5	69		W10	11.6
32	K14	K13	3.9	70		K14	5.9
33		W10	7.2	71	W12	K20	11.4
34	K15	W11	4.8	72	W13	K21	13.2
35	K16	K8	11.8	73		K22	22.7
36		W11	8.5	74	W14	K23	6
37	K17	K15	8.6				
38		K16	10.3				

Lampiran 2. Graf Data



Lampiran 3. Hasil Perhitungan Nilai Heuristik $h(n)$ dengan titik tujuan $W8$

No	Titik	$\boldsymbol{h(n) = \sqrt{(x_{goal} - x_n)^2 + (y_{goal} - y_n)^2} \times 111.132}$	Hasil
1	W2	$h(n) = \sqrt{(-8.15605 - (-7.9761))^2 + (112.1702 - 112.43014)^2} \times 111.132$	35
2	W3	$h(n) = \sqrt{(-8.15605 - (-8.02476))^2 + (112.1702 - 112.4243)^2} \times 111.132$	31.8
3	W5	$h(n) = \sqrt{(-8.15605 - (-8.03954))^2 + (112.1702 - 112.34545)^2} \times 111.132$	23.4
4	K1	$h(n) = \sqrt{(-8.15605 - (-8.04568))^2 + (112.1702 - 112.41986)^2} \times 111.132$	30.2
5	W6	$h(n) = \sqrt{(-8.15605 - -8.14501)^2 + (112.1702 - 112.45102)^2} \times 111.132$	31
6	K2	$h(n) = \sqrt{(-8.15605 - -8.13352)^2 + (112.1702 - 112.42316)^2} \times 111.132$	28.1
7	K3	$h(n) = \sqrt{(-8.15605 - 8.14563)^2 + (112.1702 - 112.36573)^2} \times 111.132$	21.7
8	K4	$h(n) = \sqrt{(-8.15605 - 8.14034)^2 + (112.1702 - 112.31279)^2} \times 111.132$	15.9
9	K5	$h(n) = \sqrt{(-8.15605 - 8.07364)^2 + (112.1702 - 112.3264)^2} \times 111.132$	19.6
10	K6	$h(n) = \sqrt{(-8.15605 - 8.04564)^2 + (112.1702 - 112.30376)^2} \times 111.132$	18.9
11	K7	$h(n) = \sqrt{(-8.15605 - 8.04307)^2 + (112.1702 - 112.23021)^2} \times 111.132$	14.2
12	W7	$h(n) = \sqrt{(-8.15605 - 8.01642)^2 + (112.1702 - 112.20974)^2} \times 111.132$	15.6
13	K8	$h(n) = \sqrt{(-8.15605 - 8.00154)^2 + (112.1702 - 112.2007)^2} \times 111.132$	17.2
14	K9	$h(n) = \sqrt{(-8.15605 - 8.1241)^2 + (112.1702 - 112.2495)^2} \times 111.132$	9.4

No	Titik	$h(n) = \sqrt{(x_{goal} - x_n)^2 + (y_{goal} - y_n)^2} \times 111.132$	Hasil
15	K10	$h(n) = \sqrt{(-8.15605 - 8.11798)^2 + (112.1702 - 112.20272)^2} \times 111.132$	5.4
16	K11	$h(n) = \sqrt{(-8.15605 - 8.16805)^2 + (112.1702 - 112.21737)^2} \times 111.132$	5.5
17	W8	$h(n) = \sqrt{(-8.15605 - 8.15605)^2 + (112.1702 - 112.1702)^2} \times 111.132$	0
18	W9	$h(n) = \sqrt{(-8.15605 - 8.08099)^2 + (112.1702 - 112.19602)^2} \times 111.132$	8.6
19	W10	$h(n) = \sqrt{(-8.15605 - 8.0673)^2 + (112.1702 - 112.12124)^2} \times 111.132$	7.5
20	K12	$h(n) = \sqrt{(-8.15605 - 8.09918)^2 + (112.1702 - 112.1781)^2} \times 111.132$	6
21	K13	$h(n) = \sqrt{(-8.15605 - 8.10288)^2 + (112.1702 - 112.1639)^2} \times 111.132$	5.8
22	K14	$h(n) = \sqrt{(-8.15605 - 8.09945)^2 + (112.1702 - 112.13405)^2} \times 111.132$	7.5
23	W11	$h(n) = \sqrt{(-8.15605 - 8.0673)^2 + (112.1702 - 112.12124)^2} \times 111.132$	11.1
24	K15	$h(n) = \sqrt{(-8.15605 - 8.0629)^2 + (112.1702 - 112.06911)^2} \times 111.132$	15.2
25	K16	$h(n) = \sqrt{(-8.15605 - 8.00251)^2 + (112.1702 - 112.12145)^2} \times 111.132$	17.6
26	K17	$h(n) = \sqrt{(-8.15605 - 8.00351)^2 + (112.1702 - 112.03683)^2} \times 111.132$	22.4
27	K18	$h(n) = \sqrt{(-8.15605 - 8.0437)^2 + (112.1702 - 111.99413)^2} \times 111.132$	23.5
28	K19	$h(n) = \sqrt{(-8.15605 - 8.19811)^2 + (112.1702 - 112.10594)^2} \times 111.132$	8.7

No	Titik	$\boldsymbol{h(n) = \sqrt{(x_{goal} - x_n)^2 + (y_{goal} - y_n)^2} \times 111.132}$	Hasil
29	W12	$h(n) = \sqrt{(-8.15605 - 8.24644)^2 + (112.1702 - 112.06807)^2} \times 111.132$	14.3
30	K20	$h(n) = \sqrt{(-8.15605 - 8.29064)^2 + (112.1702 - 112.1092)^2} \times 111.132$	16
31	W13	$h(n) = \sqrt{(-8.15605 - 8.31541)^2 + (112.1702 - 112.14345)^2} \times 111.132$	17.6
32	K21	$h(n) = \sqrt{(-8.15605 - 8.24858)^2 + (112.1702 - 112.15893)^2} \times 111.132$	10.8
33	K22	$h(n) = \sqrt{(-8.15605 - 8.26417)^2 + (112.1702 - 112.25334)^2} \times 111.132$	15.1
34	W14	$h(n) = \sqrt{(-8.15605 - 8.27248)^2 + (112.1702 - 112.32052)^2} \times 111.132$	21
35	K23	$h(n) = \sqrt{(-8.15605 - 8.23152)^2 + (112.1702 - 112.33673)^2} \times 111.132$	20.1
36	K24	$h(n) = \sqrt{(-8.15605 - 8.07625)^2 + (112.1702 - 112.22918)^2} \times 111.132$	10.6

Lampiran 4. Hasil Perhitungan Nilai Heuristik $h(n)$ dengan titik tujuan $W10$

No	Titik	$\boldsymbol{h(n) = \sqrt{(x_{goal} - x_n)^2 + (y_{goal} - y_n)^2} \times 111.132}$	Hasil
1	W2	$h(n) = \sqrt{(-8.15605 - (-7.9761))^2 + (112.1702 - 112.43014)^2} \times 111.132$	31.1
2	W3	$h(n) = \sqrt{(-8.15605 - (-8.02476))^2 + (112.1702 - 112.4243)^2} \times 111.132$	28.6
3	W5	$h(n) = \sqrt{(-8.15605 - (-8.03954))^2 + (112.1702 - 112.34545)^2} \times 111.132$	19.3
4	K1	$h(n) = \sqrt{(-8.15605 - (-8.04568))^2 + (112.1702 - 112.41986)^2} \times 111.132$	39.3

No	Titik	$h(n) = \sqrt{(x_{goal} - x_n)^2 + (y_{goal} - y_n)^2} \times 111.132$	Hasil
5	W6	$h(n) = \sqrt{(-8.15605 - -8.14501)^2 + (112.1702 - 112.45102)^2} \times 111.132$	31.2
6	K2	$h(n) = \sqrt{(-8.15605 - -8.13352)^2 + (112.1702 - 112.42316)^2} \times 111.132$	27.9
7	K3	$h(n) = \sqrt{(-8.15605 - 8.14563)^2 + (112.1702 - 112.36573)^2} \times 111.132$	21.9
8	K4	$h(n) = \sqrt{(-8.15605 - 8.14034)^2 + (112.1702 - 112.31279)^2} \times 111.132$	31.2
9	K5	$h(n) = \sqrt{(-8.15605 - 8.07364)^2 + (112.1702 - 112.3264)^2} \times 111.132$	16.6
10	K6	$h(n) = \sqrt{(-8.15605 - 8.04564)^2 + (112.1702 - 112.30376)^2} \times 111.132$	14.6
11	K7	$h(n) = \sqrt{(-8.15605 - 8.04307)^2 + (112.1702 - 112.23021)^2} \times 111.132$	7.4
12	W7	$h(n) = \sqrt{(-8.15605 - 8.01642)^2 + (112.1702 - 112.20974)^2} \times 111.132$	8.8
13	K8	$h(n) = \sqrt{(-8.15605 - 8.00154)^2 + (112.1702 - 112.2007)^2} \times 111.132$	10.5
14	K9	$h(n) = \sqrt{(-8.15605 - 8.1241)^2 + (112.1702 - 112.2495)^2} \times 111.132$	9.1
15	K10	$h(n) = \sqrt{(-8.15605 - 8.11798)^2 + (112.1702 - 112.20272)^2} \times 111.132$	4.1
16	K11	$h(n) = \sqrt{(-8.15605 - 8.16805)^2 + (112.1702 - 112.21737)^2} \times 111.132$	9.8
17	W8	$h(n) = \sqrt{(-8.15605 - 8.15605)^2 + (112.1702 - 112.1702)^2} \times 111.132$	24.6
18	W9	$h(n) = \sqrt{(-8.15605 - 8.08099)^2 + (112.1702 - 112.19602)^2} \times 111.132$	2.2
19	W10	$h(n) = \sqrt{(-8.15605 - 8.0673)^2 + (112.1702 - 112.12124)^2} \times 111.132$	0

No	Titik	$h(n) = \sqrt{(x_{goal} - x_n)^2 + (y_{goal} - y_n)^2} \times 111.132$	Hasil
20	K12	$h(n) = \sqrt{(-8.15605 - 8.09918)^2 + (112.1702 - 112.1781)^2} \times 111.132$	1.1
21	K13	$h(n) = \sqrt{(-8.15605 - 8.10288)^2 + (112.1702 - 112.1639)^2} \times 111.132$	2
22	K14	$h(n) = \sqrt{(-8.15605 - 8.09945)^2 + (112.1702 - 112.13405)^2} \times 111.132$	4.8
23	W11	$h(n) = \sqrt{(-8.15605 - 8.0673)^2 + (112.1702 - 112.12124)^2} \times 111.132$	6.5
24	K15	$h(n) = \sqrt{(-8.15605 - 8.0629)^2 + (112.1702 - 112.06911)^2} \times 111.132$	12.1
25	K16	$h(n) = \sqrt{(-8.15605 - 8.00251)^2 + (112.1702 - 112.12145)^2} \times 111.132$	11.3
26	K17	$h(n) = \sqrt{(-8.15605 - 8.00351)^2 + (112.1702 - 112.03683)^2} \times 111.132$	18
27	K18	$h(n) = \sqrt{(-8.15605 - 8.0437)^2 + (112.1702 - 111.99413)^2} \times 111.132$	20.9
28	K19	$h(n) = \sqrt{(-8.15605 - 8.19811)^2 + (112.1702 - 112.10594)^2} \times 111.132$	15
29	W12	$h(n) = \sqrt{(-8.15605 - 8.24644)^2 + (112.1702 - 112.06807)^2} \times 111.132$	21.7
30	K20	$h(n) = \sqrt{(-8.15605 - 8.29064)^2 + (112.1702 - 112.1092)^2} \times 111.132$	24.1
31	W13	$h(n) = \sqrt{(-8.15605 - 8.31541)^2 + (112.1702 - 112.14345)^2} \times 111.132$	25.2
32	K21	$h(n) = \sqrt{(-8.15605 - 8.24858)^2 + (112.1702 - 112.15893)^2} \times 111.132$	18.2
33	K22	$h(n) = \sqrt{(-8.15605 - 8.26417)^2 + (112.1702 - 112.25334)^2} \times 111.132$	21.9
34	W14	$h(n) = \sqrt{(-8.15605 - 8.27248)^2 + (112.1702 - 112.32052)^2} \times 111.132$	26.3

No	Titik	$h(n) = \sqrt{(x_{goal} - x_n)^2 + (y_{goal} - y_n)^2} \times 111.132$	Hasil
35	K23	$h(n) = \sqrt{(-8.15605 - 8.23152)^2 + (112.1702 - 112.33673)^2} \times 111.132$	43.1
36	K24	$h(n) = \sqrt{(-8.15605 - 8.07625)^2 + (112.1702 - 112.22918)^2} \times 111.132$	5.81

Lampiran 5. Hasil Perhitungan Nilai Heuristik $h(n)$ dengan titik tujuan $W5$

No	Titik	$h(n) = \sqrt{(x_{goal} - x_n)^2 + (y_{goal} - y_n)^2} \times 111.132$	Hasil
1	W2	$h(n) = \sqrt{(-8.15605 - (-7.9761))^2 + (112.1702 - 112.43014)^2} \times 111.132$	12.3
2	W3	$h(n) = \sqrt{(-8.15605 - (-8.02476))^2 + (112.1702 - 112.4243)^2} \times 111.132$	9.1
3	W5	$h(n) = \sqrt{(-8.15605 - (-8.03954))^2 + (112.1702 - 112.34545)^2} \times 111.132$	0
4	K1	$h(n) = \sqrt{(-8.15605 - (-8.04568))^2 + (112.1702 - 112.41986)^2} \times 111.132$	8.2
5	W6	$h(n) = \sqrt{(-8.15605 - -8.14501)^2 + (112.1702 - 112.45102)^2} \times 111.132$	16
6	K2	$h(n) = \sqrt{(-8.15605 - -8.13352)^2 + (112.1702 - 112.42316)^2} \times 111.132$	13.1
7	K3	$h(n) = \sqrt{(-8.15605 - 8.14563)^2 + (112.1702 - 112.36573)^2} \times 111.132$	11.3
8	K4	$h(n) = \sqrt{(-8.15605 - 8.14034)^2 + (112.1702 - 112.31279)^2} \times 111.132$	11.2
9	K5	$h(n) = \sqrt{(-8.15605 - 8.07364)^2 + (112.1702 - 112.3264)^2} \times 111.132$	11.2
10	K6	$h(n) = \sqrt{(-8.15605 - 8.04564)^2 + (112.1702 - 112.30376)^2} \times 111.132$	4.6
11	K7	$h(n) = \sqrt{(-8.15605 - 8.04307)^2 + (112.1702 - 112.23021)^2} \times 111.132$	12.7

No	Titik	$h(n) = \sqrt{(x_{goal} - x_n)^2 + (y_{goal} - y_n)^2} \times 111.132$	Hasil
12	W7	$h(n) = \sqrt{(-8.15605 - 8.01642)^2 + (112.1702 - 112.20974)^2} \times 111.132$	15
13	K8	$h(n) = \sqrt{(-8.15605 - 8.00154)^2 + (112.1702 - 112.2007)^2} \times 111.132$	16.2
14	K9	$h(n) = \sqrt{(-8.15605 - 8.1241)^2 + (112.1702 - 112.2495)^2} \times 111.132$	13.7
15	K10	$h(n) = \sqrt{(-8.15605 - 8.11798)^2 + (112.1702 - 112.20272)^2} \times 111.132$	17.8
16	K11	$h(n) = \sqrt{(-8.15605 - 8.16805)^2 + (112.1702 - 112.21737)^2} \times 111.132$	19.8
17	W8	$h(n) = \sqrt{(-8.15605 - 8.15605)^2 + (112.1702 - 112.1702)^2} \times 111.132$	22.9
18	W9	$h(n) = \sqrt{(-8.15605 - 8.08099)^2 + (112.1702 - 112.19602)^2} \times 111.132$	30.9
19	W10	$h(n) = \sqrt{(-8.15605 - 8.0673)^2 + (112.1702 - 112.12124)^2} \times 111.132$	19.4
20	K12	$h(n) = \sqrt{(-8.15605 - 8.09918)^2 + (112.1702 - 112.1781)^2} \times 111.132$	19.6
21	K13	$h(n) = \sqrt{(-8.15605 - 8.10288)^2 + (112.1702 - 112.1639)^2} \times 111.132$	21.3
22	K14	$h(n) = \sqrt{(-8.15605 - 8.09945)^2 + (112.1702 - 112.13405)^2} \times 111.132$	24.2
23	W11	$h(n) = \sqrt{(-8.15605 - 8.0673)^2 + (112.1702 - 112.12124)^2} \times 111.132$	24.9
24	K15	$h(n) = \sqrt{(-8.15605 - 8.0629)^2 + (112.1702 - 112.06911)^2} \times 111.132$	30.7
25	K16	$h(n) = \sqrt{(-8.15605 - 8.00251)^2 + (112.1702 - 112.12145)^2} \times 111.132$	25.1
26	K17	$h(n) = \sqrt{(-8.15605 - 8.00351)^2 + (112.1702 - 112.03683)^2} \times 111.132$	34.6

No	Titik	$h(n) = \sqrt{(x_{goal} - x_n)^2 + (y_{goal} - y_n)^2} \times 111.132$	Hasil
27	K18	$h(n) = \sqrt{(-8.15605 - 8.0437)^2 + (112.1702 - 111.99413)^2} \times 111.132$	38.8
28	K19	$h(n) = \sqrt{(-8.15605 - 8.19811)^2 + (112.1702 - 112.10594)^2} \times 111.132$	32.1
29	W12	$h(n) = \sqrt{(-8.15605 - 8.24644)^2 + (112.1702 - 112.06807)^2} \times 111.132$	38.2
30	K20	$h(n) = \sqrt{(-8.15605 - 8.29064)^2 + (112.1702 - 112.1092)^2} \times 111.132$	38.8
31	W13	$h(n) = \sqrt{(-8.15605 - 8.31541)^2 + (112.1702 - 112.14345)^2} \times 111.132$	38.2
32	K21	$h(n) = \sqrt{(-8.15605 - 8.24858)^2 + (112.1702 - 112.15893)^2} \times 111.132$	30.9
33	K22	$h(n) = \sqrt{(-8.15605 - 8.26417)^2 + (112.1702 - 112.25334)^2} \times 111.132$	27.6
34	W14	$h(n) = \sqrt{(-8.15605 - 8.27248)^2 + (112.1702 - 112.32052)^2} \times 111.132$	25.4
35	K23	$h(n) = \sqrt{(-8.15605 - 8.23152)^2 + (112.1702 - 112.33673)^2} \times 111.132$	20.6
36	K24	$h(n) = \sqrt{(-8.15605 - 8.07625)^2 + (112.1702 - 112.22918)^2} \times 111.132$	13.4

Lampiran 6. Program Visualisasi Algoritma *A-Star* Tradisional

```
import networkx as nx
import matplotlib.pyplot as plt
from heapq import heappop, heappush
import time

def heuristic(node, h):
    """Mengambil nilai heuristik dari node berdasarkan h(n)."""
    return h.get(node, float('inf'))
```

```
def visualize_graph_step(graph, current, open_set, closed_set, step,
path=None):
    """Visualisasi graf pada setiap langkah pencarian."""
    pos = nx.spring_layout(graph, seed=42)  # Posisi node tetap
    plt.figure(figsize=(16, 12))
    nx.draw(graph, pos, with_labels=True, node_size=700, font_size=10,
font_weight="bold")
    edge_labels = nx.get_edge_attributes(graph, 'weight')
    nx.draw_networkx_edge_labels(graph, pos, edge_labels=edge_labels)

 # Gambar Open Set, Closed Set, dan Current Node
    open_nodes = [item[1][-1] for item in open_set]
    nx.draw_networkx_nodes(graph, pos, nodelist=open_nodes,
node_color="green", node_size=600, label="Open Set")
    nx.draw_networkx_nodes(graph, pos, nodelist=closed_set,
node_color="blue", node_size=600, label="Closed Set")
    nx.draw_networkx_nodes(graph, pos, nodelist=[current],
node_color="orange", node_size=800, label="Current Node")

 # Visualisasi path jika ada
    if path:
        edges = list(zip(path, path[1:]))
        nx.draw_networkx_edges(graph, pos, edgelist=edges, edge_color="red",
width=2, label="Path")

    plt.title(f"Proses Iterasi {step} - Node {current}")
    plt.legend()
    plt.show()
    time.sleep(0.5)
    f_score = {node: float('inf') for node in graph.nodes}
    f_score[start] = heuristic(start, h)
def a_star(graph, start, goal, h):
    """Algoritma A* dengan f(n) = g(n) + h(n)."""
    open_set = []  # Heap untuk prioritas berdasarkan f(n)
    heappush(open_set, (0, [start]))  # (f(n), path lengkap)
    closed_set = set()
    step = 0

    while open_set:
        step += 1
        f_current, current_path = heappop(open_set)
        current_node = current_path[-1]
```

```
        # Tambahkan node ke Closed Set
        if current_node not in closed_set:
            closed_set.add(current_node)

        # Visualisasi langkah
        visualize_graph_step(graph, current_node, open_set, closed_set, step,
path=current_path)

        # Jika mencapai tujuan
        if current_node == goal:
            print(f"Tujuan ditemukan pada iterasi ke-{step}.")
            return current_path

        # Hitung g(n) dari jalur lengkap
        g_current = sum(graph.edges[current_path[i], current_path[i +
1]]['weight']
                        for i in range(len(current_path) - 1))

        # Proses semua tetangga
        for neighbor in graph.neighbors(current_node):
            if neighbor in current_path:  # Hindari siklus
                continue

            edge_weight = graph[current_node][neighbor]['weight']
            g_tentative = g_current + edge_weight
            f_tentative = g_tentative + heuristic(neighbor, h)
            new_path = current_path + [neighbor]

            # Log evaluasi
            print(f"Evaluating neighbor {neighbor}: g(n)={g_tentative:.1f},
h(n)={heuristic(neighbor, h):.1f}, f(n)={f_tentative:.1f}")

            # Tambahkan ke Open Set
            heappush(open_set, (f_tentative, new_path))

    print("Tidak ditemukan jalur ke tujuan.")
    return None
# Data graf
graph = nx.Graph()

# Menambahkan edges dengan bobot
edges = [
    ('K1', 'W3', 2.8), ('K2', 'W6', 4.2), ('K2', 'K1', 15.4), ('K3', 'K2', 8.3), ('K3',
'K1', 13.5),
```

```
    ('K3', 'K5', 11.7), ('K4', 'K5', 8.4), ('K4', 'K3', 6.8), ('K5', 'K1', 16.9), ('K5',
'W5', 4.8),
    ('K5', 'K6', 5.8), ('K6', 'W5', 8.9), ('K6', 'W4', 15.7), ('K7', 'W7', 7.2), ('K7',
'K6', 8.5),
    ('K7', 'K24', 5), ('K8', 'W7', 2.3), ('K8', 'W10', 13.4), ('K9', 'K5', 13.1), ('K9',
'K4', 9),
    ('K10', 'K9', 6.6), ('K10', 'W9', 6.1), ('K11', 'K10', 7.1), ('K11', 'K9', 7.9),
('K11', 'K23', 17.8),
    ('K12', 'W9', 3.1), ('K12', 'K10', 4.5), ('K13', 'K12', 2.2), ('K13', 'K10', 5.8),
('K13', 'W8', 9.5),
    ('K13', 'K19', 16.5), ('K14', 'K13', 3.9), ('K14', 'W10', 7.2), ('K15', 'W11',
4.8), ('K16', 'K8', 11.8),
    ('K16', 'W11', 8.5), ('K17', 'K15', 8.6), ('K17', 'K16', 10.3), ('K18', 'K17', 8),
('K18', 'K15', 8.9),
    ('K19', 'W8', 12.7), ('K19', 'K21', 11.5), ('K19', 'K20', 14), ('K19', 'W12',
8.9), ('K20', 'W13', 12.4),
    ('K21', 'K22', 20), ('K21', 'W8', 15.4), ('K21', 'K11', 16.3), ('K22', 'W14',
10.7), ('K22', 'K11', 17.7),
    ('K23', 'K4', 14.8), ('K23', 'K3', 12.8), ('K24', 'K6', 12.3), ('K24', 'K5', 12.8),
('K24', 'K9', 7.5),
    ('W5', 'W4', 10.8), ('W5', 'W2', 13.6), ('W8', 'K10', 7.9), ('W8', 'K11', 5.8),
('W9', 'K24', 4.2),
    ('W9', 'K7', 6.7), ('W9', 'W7', 9.2), ('W9', 'K12', 3.1), ('W10', 'W9', 3.3),
('W10', 'W7', 10.9),
    ('W10', 'K13', 3.6), ('W10', 'K12', 2.4), ('W11', 'K8', 17.3), ('W11', 'W10',
11.6), ('W11', 'K14', 5.9),
    ('W12', 'K20', 11.4), ('W13', 'K21', 13.2), ('W13', 'K22', 22.7), ('W14', 'K23',
6)
]

graph.add_weighted_edges_from(edges)

# Data heuristik
h = {
    'W2': 35, 'W3': 31.8, 'W4': 23.4, 'K1': 30.2, 'W6': 31, 'K2': 28.1, 'K3': 21.7,
'K4': 15.9,
    'K5': 19.6, 'K6': 18.9,  'K7': 14.2, 'W7': 15.6, 'K8': 17.2, 'K9': 9.36, 'K10':
5.44, 'K11': 5.5,
    'W8': 0, 'W9': 8.67, 'W10': 7.45, 'K12': 5.94, 'K13': 5.78, 'K14': 7.53, 'W11':
11.1, 'K15': 15.2,
    'K16': 17.6, 'K17': 22.4, 'K18': 23.5, 'K19': 8.7, 'W12': 14.3, 'K20': 16, 'W13':
17.6, 'K21': 10.8,
    'K22': 15.1, 'W14': 21, 'K23': 20.1, 'K24': 10.6
}
```

```
# Jalankan A*
start_node = 'K6'
goal_node = 'W8'
path = a_star(graph, start_node, goal_node, h)

print("Jalur Terpendek:", path)
```

Lampiran 7. Program Visualisasi Modifikasi Algoritma *A-Star*

```
import networkx as nx
import matplotlib.pyplot as plt
from heapq import heappop, heappush
import time
import numpy as np

def heuristic(node, h):
    """Mengambil nilai heuristik dari node berdasarkan h(n)."""
    return h.get(node, float('inf'))

def calculate_average_edge_weight(graph):
    """Menghitung rata-rata semua edge dalam graf sebagai π."""
    weights = nx.get_edge_attributes(graph, 'weight').values()
    if weights:
        return np.mean(list(weights))
    return 1  # fallback jika tidak ada edge

def visualize_graph_step(graph, current, open_set, closed_set, step,
path=None):
    """Visualisasi graf pada setiap langkah pencarian."""
    pos = nx.spring_layout(graph, seed=42)  # Posisi node tetap
    plt.figure(figsize=(16, 12))

    nx.draw(graph, pos, with_labels=True, node_size=700, font_size=10,
font_weight="bold")
    edge_labels = nx.get_edge_attributes(graph, 'weight')
    nx.draw_networkx_edge_labels(graph, pos, edge_labels=edge_labels)
 # Gambar Open Set, Closed Set, dan Current Node
    open_nodes = [item[1][-1] for item in open_set]
    nx.draw_networkx_nodes(graph, pos, nodelist=open_nodes,
node_color="green", node_size=600, label="Open Set")
```

```
    nx.draw_networkx_nodes(graph, pos, nodelist=closed_set,
node_color="blue", node_size=600, label="Closed Set")
    nx.draw_networkx_nodes(graph, pos, nodelist=[current],
node_color="orange", node_size=800, label="Current Node")

    # Visualisasi path jika ada
    if path:
        edges = list(zip(path, path[1:]))
        nx.draw_networkx_edges(graph, pos, edgelist=edges, edge_color="red",
width=2, label="Path")

    plt.title(f"Proses Iterasi {step} - Node {current}")
    plt.legend()
    plt.show()
    time.sleep(0.5)

def a_star(graph, start, goal, h):
    """Algoritma A* dengan f(n) = g(n) + h(n) * (π / c(n))."""
    open_set = []  # Heap untuk prioritas berdasarkan f(n)
    heappush(open_set, (0, [start]))  # (f(n), path lengkap)
    closed_set = set()
    step = 0

    # Hitung rata-rata bobot edge dalam graf sebagai π
    pi_value = calculate_average_edge_weight(graph)

    while open_set:
        step += 1
        f_current, current_path = heappop(open_set)
        current_node = current_path[-1]

        # Tambahkan node ke Closed Set
        if current_node not in closed_set:
            closed_set.add(current_node)
  # Visualisasi langkah
        visualize_graph_step(graph, current_node, open_set, closed_set, step,
path=current_path)

        # Jika mencapai tujuan
        if current_node == goal:
            print(f"Tujuan ditemukan pada iterasi ke-{step}.")
            return current_path
```

```
        # Hitung g(n) dari jalur lengkap
        g_current = sum(graph.edges[current_path[i], current_path[i +
1]]['weight']
                        for i in range(len(current_path) - 1))

        # Proses semua tetangga
        for neighbor in graph.neighbors(current_node):
            if neighbor in current_path:  # Hindari siklus
                continue

            edge_weight = graph[current_node][neighbor]['weight']
            g_tentative = g_current + edge_weight
            alpha_tentative = pi_value / edge_weight
            f_tentative = g_tentative + heuristic(neighbor, h) * alpha_tentative
            new_path = current_path + [neighbor]

            # Log evaluasi
            print(f"Evaluating neighbor {neighbor}: g(n)={g_tentative:.1f},
h(n)={heuristic(neighbor, h):.1f}, alpha(n)={alpha_tentative:.2f},
f(n)={f_tentative:.1f}")

            # Tambahkan ke Open Set
            heappush(open_set, (f_tentative, new_path))

    print("Tidak ditemukan jalur ke tujuan.")
    return None

# Data graf
graph = nx.Graph()
# Menambahkan edges dengan bobot
edges = [
    ('K1', 'W3', 2.8), ('K2', 'W6', 4.2), ('K2', 'K1', 15.4), ('K3', 'K2', 8.3), ('K3',
'K1', 13.5),
    ('K3', 'K5', 11.7), ('K4', 'K5', 8.4), ('K4', 'K3', 6.8), ('K5', 'K1', 16.9), ('K5',
'W5', 4.8),
    ('K5', 'K6', 5.8), ('K6', 'W5', 8.9), ('K6', 'W4', 15.7), ('K7', 'W7', 7.2), ('K7',
'K6', 8.5),
    ('K7', 'K24', 5), ('K8', 'W7', 2.3), ('K8', 'W10', 13.4), ('K9', 'K5', 13.1), ('K9',
'K4', 9),
    ('K10', 'K9', 6.6), ('K10', 'W9', 6.1), ('K11', 'K10', 7.1), ('K11', 'K9', 7.9),
('K11', 'K23', 17.8),
    ('K12', 'W9', 3.1), ('K12', 'K10', 4.5), ('K13', 'K12', 2.2), ('K13', 'K10', 5.8),
('K13', 'W8', 9.5),
    ('K13', 'K19', 16.5), ('K14', 'K13', 3.9), ('K14', 'W10', 7.2), ('K15', 'W11',
```

```
 4.8), ('K16', 'K8', 11.8),
    ('K16', 'W11', 8.5), ('K17', 'K15', 8.6), ('K17', 'K16', 10.3), ('K18', 'K17', 8),
('K18', 'K15', 8.9),
    ('K19', 'W8', 12.7), ('K19', 'K21', 11.5), ('K19', 'K20', 14), ('K19', 'W12',
8.9), ('K20', 'W13', 12.4),
    ('K21', 'K22', 20), ('K21', 'W8', 15.4), ('K21', 'K11', 16.3), ('K22', 'W14',
10.7), ('K22', 'K11', 17.7),
    ('K23', 'K4', 14.8), ('K23', 'K3', 12.8), ('K24', 'K6', 12.3), ('K24', 'K5', 12.8),
('K24', 'K9', 7.5),
    ('W5', 'W4', 10.8), ('W5', 'W2', 13.6), ('W8', 'K10', 7.9), ('W8', 'K11', 5.8),
('W9', 'K24', 4.2),
    ('W9', 'K7', 6.7), ('W9', 'W7', 9.2), ('W9', 'K12', 3.1), ('W10', 'W9', 3.3),
('W10', 'W7', 10.9),
    ('W10', 'K13', 3.6), ('W10', 'K12', 2.4), ('W11', 'K8', 17.3), ('W11', 'W10',
11.6), ('W11', 'K14', 5.9),
    ('W12', 'K20', 11.4), ('W13', 'K21', 13.2), ('W13', 'K22', 22.7), ('W14', 'K23',
6)
]

graph.add_weighted_edges_from(edges)

# Data heuristik
h = {
    'W2': 35, 'W3': 31.8, 'W4': 23.4, 'K1': 30.2, 'W6': 31, 'K2': 28.1, 'K3': 21.7,
'K4': 15.9,
    'K5': 19.6, 'K6': 18.9, 'K7': 14.2, 'W7': 15.6, 'K8': 17.2, 'K9': 9.36, 'K10':
5.44, 'K11': 5.5,
    'W8': 0, 'W9': 8.67, 'W10': 7.45, 'K12': 5.94, 'K13': 5.78, 'K14': 7.53, 'W11':
11.1, 'K15': 15.2,
    'K16': 17.6, 'K17': 22.4, 'K18': 23.5, 'K19': 8.7, 'W12': 14.3, 'K20': 16, 'W13':
17.6, 'K21': 10.8,
    'K22': 15.1, 'W14': 21, 'K23': 20.1, 'K24': 10.6
}

# Jalankan A*
start_node = 'K6'
goal_node = 'W8'
path = a_star(graph, start_node, goal_node, h)

print("Jalur Terpendek:", path)
```

RIWAYAT HIDUP

Penulis, Achmad Faiz Sukroni, lahir di Blitar pada tanggal 27 Maret 2002. Penulis lahir dari pasangan Agus Mustofa dan Ibu Hanik Zubaidah sebagai anak pertama dari 2 bersaudara. Pendidikan pertama penulis, ditempuh di MI PLUS AL-AZHAR Blitar, kemudian melanjutkan pendidikan menengah pertama Mts MA'ARIF NU 2 Sutojayan dan lulus pada tahun 2017. Setelah itu, penulis menyelesaikan pendidikan menengah atas di MA MA'ARIF NU Kota Blitar dan lulus pada tahun 2020. Pada tahun yang sama, penulis melanjutkan studi di Universitas Islam Negeri (UIN) Maulana Malik Ibrahim Malang mengambil program studi Matematika di Fakultas Sains dan Teknologi. Selama masa kuliah, selain menyelesaikan tugasnya sebagai mahasiswa, penulis juga merupakan santri pada Pondok Pesantren Gasek Sabilurrosyad yang diasuh oleh Abah Kyai Marzuki Mustamar.

KEMENTERIAN 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 : Achmad Faiz Sukroni
NIM : 200601110022
Fakultas / Jurusan : Sains dan Teknologi / Matematika
Judul Skripsi : Modifikasi Algoritma *A-Star* pada Optimalisasi Rute Terpendek
Pembimbing I : Juhari, M.Si.
Pembimbing II : Ach Nasichuddin, M.A.

No	Tanggal	Hal	Tanda Tangan
1.	9 November 2023	Konsultasi Topik	1.
2.	25 April 2024	Konsultasi Bab I, II, dan III	2.
3.	1 Juli 2024	Konsultasi Bab I, II, dan III	3.
4.	5 Juli 2024	Konsultasi Bab I, II, dan III	4.
5.	9 Juli 2024	ACC Bab I, II, dan III	5.
6.	25 Juni 2024	ACC Bab I, II, dan III	6.
7.	10 Juli 2024	Konsultasi Kajian Agama Bab I dan II	7.
8.	18 Juli 2024	Konsultasi Kajian Agama Bab I dan II	8.
9.	19 Juli 2024	ACC Kajian Agama Bab I dan II	9.
10.	15 Agustus 2024	ACC Seminar Proposal	10.
11.	27 September 2024	Konsultasi Revisi Seminar Proposal	11.
12.	23 Oktober 2024	Konsultasi Bab IV dan V	12.
13.	25 Oktober 2024	ACC Bab IV dan V	13.
14.	29 Oktober 2024	Konsultasi Kajian Agama Bab IV	14.
15.	29 Oktober 2024	ACC Kajian Agama Bab IV	15.

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

16.	7 November 2024	ACC Seminar Hasil	16.
17.	21 November 2024	Konsultasi Revisi Seminar Hasil	17.
18.	25 November 2024	Konsultasi Revisi Seminar Hasil	18.
19.	10 Desember 2024	ACC Sidang Skripsi	19.

Malang, 26 Desember 2024
Mengetahui,
Ketua Program Studi Matematika

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