IMPLEMENTASI ALGORITMA *UNIFORM COST SEARCH* (UCS) UNTUK MENETUKAN RUTE TERPENDEK

SKRIPSI

Oleh:
DIAZ RIZQI APRILLIANDO
NIM. 18650122

PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS SAINS DAN TEKNOLOGI
UNIVERSITAS ISLAM NEGERI MAULANA MALIK IBRAHIM
MALANG
2023

IMPLEMENTASI ALGORITMA *UNIFORM COST SEARCH* (UCS) UNTUK MENENTUKAN RUTE TERPENDEK

SKRIPSI

Oleh:
DIAZ RIZQI APRILLIANDO
NIM. 18650122

Diajukan kepada:
Universitas Islam Negeri (UIN) Maulana Malik Ibrahim Malang
Untuk Memenuhi Salah Satu Persyaratan Dalam
Memperoleh Gelar Sarjana Komputer (S.Kom)

PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS SAINS DAN TEKNOLOGI
UNIVERSITAS ISLAM NEGERI MAULANA MALIK IBRAHIM MALANG
2023

HALAMAN PERSETUJUAN

IMPLEMENTASI ALGORITMA *UNIFORM COST SEARCH* (UCS) UNTUK MENETUKAN RUTE TERPENDEK

SKRIPSI

Oleh:
DIAZ RIZQI APRILLIANDO
NIM. 18650122

Telah Diperiksa dan Disetujui untuk Diuji
Tanggal: 1 Desember 2023

Pembimbing I,	Pembimbing II,
Supriyono, M. Kom	Dr. Fresy Nugroho, M. T
NIP. 19841010 201903 1 012	NIP. 19710722 201101 1 001

Mengetahui,
Ketua Program Studi Teknik Informatika
Fakultas Sains dan Teknologi
Universitas Islam Negeri Maulana Malik Ibrahim Malang

Dr. Cahyo Crysdian, M.MT, IPM
NIP. 19771020 200912 1 001

HALAMAN PENGESAHAN

IMPLEMENTASI ALGORITMA *UNIFORM COST SEARCH* (UCS) UNTUK MENETUKAN RUTE TERPENDEK

SKRIPSI

Oleh:
DIAZ RIZQI APRILLIANDO
NIM. 18650122

Telah Dipertahankan di Depan Dewan Penguji Skripsi
dan Dinyatakan Diterima Sebagai Salah Satu Persyaratan
Untuk Memperoleh Gelar Sarjana Komputer (S.Kom)
Tanggal: 12 Desember 2023

Susunan Dewan Penguji

Ketua Penguji	: Dr. Totok Chamidy, M.Kom NIP. 19691222 200604 1 001	( )
Anggota Penguji I	: Agung Teguh Wibowo Almais, M. T NIDT. 19860301 20180201 1 235	( )
Anggota Penguji II	: Supriyono, M. Kom NIP. 19841010 201903 1 012	( )
Anggota Penguji III	: Dr. Fresy Nugroho, M. T NIP. 19710722 201101 1 001	( )

Mengetahui,
Ketua Program Studi Teknik Informatika
Fakultas Sains dan Teknologi
Universitas Islam Negeri Maulana Malik Ibrahim Malang

Dr. Fachrul Kurniawan, M.MT, IPM
NIP. 19771020 200912 1 001

PERNYATAAN KEASLIAN TULISAN

Saya yang bertanda tangan di bawah ini:

Nama	:	Diaz Rizqi Aprilliando
NIM	:	18650122
Fakultas	:	Sains dan Teknologi
Jurusan	:	Teknik Informatika
Judul Skripsi	:	Implementasi Algoritma Uniform Cost Search (UCS) Untuk Menetukan Rute Terpendek

Menyatakan dengan sebenarnya bahwa Skripsi yang saya tulis ini benar-benar merupakan hasil karya saya sendiri, bukan merupakan pengambil alihan data, tulisan, atau pikiran orang lain yang saya akui sebagai hasil tulisan atau pikiran saya sendiri, kecuali dengan mencantumkan sumber cuplikan pada daftar pustaka.

Apabila dikemudian hari terbukti atau dapat dibuktikan skripsi ini merupakan hasil jiplakan, maka saya bersedia menerima sanksi atas perbuatan tersebut.

Malang, 12 Desember 2023

Yang Membuat pernyataan,

Diaz Rizqi Aprilliando

NIM.18650122

MOTTO

”Always forget what you give
Never forget to forgive’’

HALAMAN PERSEMBAHAN

اَلْحَمْدُ لِلهِ رَبِّ الْعَالَمِيْ

Puji syukur kehadirat Allah SWT, shalawat dan salam bagi Rasul-Nya
Penulis mempersembahkan sebuah karya skripsi ini kepada
kedua orang tua, keluarga, dosen, teman-teman seperjuangan, dan
semua pihak yang sudah memberikan dukungan, motivasi, semangat, serta
doa sehingga lancar dalam menyelesaikan penulisan skripsi. Semoga Allah SWT
membalas kebaikan mereka.

KATA PENGANTAR

Assalamu alaikum, Wr. Wb

Puji dan syukur kehadirat Allah Swt atas berkat rahmat kesehatan dan hidayahnya-Nya, penulis diberikan kemudahan dalam menyelesaikan skripsi ini. Sholawat serta salam semoga terlimpah curahkan kepada nabi besar Muhammad SAW, yang telah membawa umat manusia dari zaman kegelapan menuju zaman yang terang benderang dengan agama Islam yang penuh dengan rahmat dan hidayah.

Penyusunan skripsi ini bertujuan untuk memenuhi syarat kelulusan mahasiswa Teknik Informatika Universitas Islam Negeri Maulana Malik Ibrahim Malang. Bagi saya selaku penulis, penyusunan skripsi dengan judul "Implementasi Algoritma *Uniform Cost Search* (UCS) Untuk Menentukan Rute Terpendek" merupakan tugas yang tidak mudah. Saya menyadari banyak sekali hambatan dalam proses penyusunan skripsi ini. Ketika pada akhirnya karya ini dapat terselesaikan, keberhasilan penulisan skripsi ini tidak lepas dari dukungan, motivasi, semangat serta doa dari banyak pihak. Oleh karenanya, dalam kesempatan ini penulis mengucapkan terima kasih yang sebesar-besarnya kepada:

1. Prof. Dr. M. Zainuddin, M.A., Selaku rektor Universitas Islam Negeri Maulana Malik Ibrahim Malang
2. Dr. Sri Hariani, M.Si., selaku dekan Fakultas Sains dan Teknologi
3. Dr. Fachrul Kurniawan ST., M.MT., IPM., selaku Ketua Program Studi Teknik Informatika
4. Bapak Supriyono, M.Kom selaku Dosen Pembimbing I yang telah dengan sabar

membimbing penulis, memberikan saran dan arahan sehingga penulis dapat menyelesaikan skripsi dengan baik.

5. Bapak Dr. Fresy Nugroho, M.T, selaku Dosen Pembimbing II yang telah membimbing dan memberi arahan kepada penulis.
6. Bapak Dr. Totok Chamidy, M.Kom selaku Dosen Penguji I dan Bapak Agung Teguh Wibowo Almais, M.T selaku Dosen Penguji II yang telah meluangkan waktu memberikan arahan untuk skripsi ini.
7. Orang tua saya yaitu Bapak M. Ilzam Marzuk dan Ibu Dyah Kusuma Wardhani, beserta keluarga besar yang selalu memberikan dukungan moral maupun spiritual sehingga penulis diberi kemudahan dalam menyelesaikan skripsi ini.
8. Pasangan saya Diah Rahmadhita Islami yang selalu memberikan dukungan, bantuan, dan memberikan arahan sehinga penulis bisa menyelesaikan skripsi ini.
9. Seluruh dosen dan staf Program Studi Teknik Informatika Universitas Islam Negeri Maulana Malik Ibrahim Malang dan segenap guru dari TK hingga SMA yang telah memberikan ilmu dan pengelaman berharga.
10. Teman-teman grup Ruwet Info Loker dan Unity of Informatics Force (UFO) 2018 terutama Danu, Ibram, Riswan, Noer, Adhit, Mas Kevin, Dwiki, Kipli yang telah senantiasa menemani penulis untuk menyelesaikan tugas dan skripsi.

Penulis menyadari bahwa dalam penyusunan skripsi ini masih jauh dari kata sempurna. Maka dari itu penulis akan menerima saran dan kritik yang membangun. Terlepas dari itu semua, penulis berharap semoga skripsi ini dapat memberikan manfaat kepada para pembaca.

Wassalamualaikum Wr. Wb.

Malang, 14 Desember 2023

Diaz Rizqi Aprilliando

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

ABSTRAK

Aprilliando, Diaz Rizqi. 2023. **"Implementasi Algoritma Uniform Cost Search Untuk Menentukan Rute Terpendek"**. Skripsi. Jurusan Teknik Informatika Fakultas Sains dan Teknologi Universitas Islam Negeri Maulana Malik Ibrahim Malang. Pembimbing: (I) Supriyono, M. Kom, (II) Dr. Fresy Nugroho, M.T.

Kata Kunci: *Algoritma Uniform Cost Search, Rute Terpendek*

Pada saat ini pencarian rute terpendek menjadi kunci utama bagi mobilitas efisien, terutama di tengah kepadatan populasi dan pertumbuhan perkotaan yang pesat. Kebutuhan mendesak untuk menemukan jalur tercepat tidak hanya sekadar pemilihan jalur terpendek, melainkan juga merupakan strategi untuk mengatasi kemacetan, mengoptimalkan waktu tempuh, dan meminimalisir dampak lingkungan. Pencarian rute terpendek melibatkan aspek-aspek terkait seperti menghindari kemacetan lalu lintas, mengatur pengiriman barang, dan memastikan aksesibilitas serta keamanan dalam situasi darurat. Meskipun demikian, kendala seperti ketidakakuratan data lokasi, ketidakpastian kondisi jalan, dinamika lalu lintas yang tidak dapat diprediksi, dan ketersediaan transportasi umum yang terbatas menjadi tantangan yang perlu diatasi. Solusi untuk mengatasi kendala tersebut melibatkan peningkatan akurasi data lokasi, informasi lalu lintas waktu nyata, dan pengembangan algoritma yang mempertimbangkan preferensi moda transportasi. Kerjasama antara penyedia transportasi umum dan platform pencarian rute terpendek juga diakui dapat meningkatkan informasi yang tersedia. Salah satu algoritma yang relevan dalam pencarian rute terpendek adalah Uniform Cost Search (UCS), yang bekerja dengan mempertimbangkan biaya setiap simpul dalam graf jaringan jalan atau peta. Validasi dari pengujian kedua menunjukkan bahwa hasil aplikasi dengan rute Kecamatan Rungkut → Kecamatan Sukolilo → Kecamatan Gubeng memiliki jarak 7.600 Meter, yang kurang lebih sama dengan hasil yang diberikan oleh Google Maps dengan rute serupa dan jarak 7.700 Meter. Implementasi algoritma UCS pada pencarian rute terpendek di kota tersebut sangat efektif dan dapat meminimalkan biaya operasional. Algoritma ini berhasil mengatasi permasalahan pada pencarian rute yang awalnya memakan waktu lama dan memiliki biaya operasional tinggi, menjadi lebih efisien dengan waktu yang lebih cepat dan biaya operasional yang rendah. Hasil dari pencarian rute terpendek ini membuktikan efisiensinya dalam mengurangi biaya operasional dan mencapai tujuan dengan lebih cepat. Validasi dari hasil aplikasi juga menunjukkan kesesuaian dengan rute yang diberikan oleh Google Maps, mengukuhkan kehandalan implementasi algoritma UCS dalam konteks pencarian rute terpendek di lingkungan perkotaan.

ABSTRACT

Aprilliando, Diaz Rizqi. 2023. **"Implementation of the Uniform Cost Search Algorithm to Determine the Shortest Route"**. Thesis. Department of Informatics Engineering, Faculty of Science and Technology. Maulana Malik Ibrahim State Islamic University Malang. Supervisor: (I) Supriyono, M.Kom, (II) Dr. Fresy Nugroho, M.T.

At this time, the search for the shortest route is the main key to efficient mobility, especially in the midst of population density and rapid urban growth. The urgent need to find the fastest path is not just about choosing the shortest path, but also a strategy to overcome congestion, optimize travel time, and minimize environmental impact. The search for the shortest route involves related aspects such as avoiding traffic jams, organizing the delivery of goods, and ensuring accessibility and safety in emergency situations. However, obstacles such as inaccurate location data, uncertainty of road conditions, unpredictable traffic dynamics, and limited availability of public transportation are challenges that need to be overcome. Solutions to overcome such constraints involve improving the accuracy of location data, real-time traffic information, and developing algorithms that consider transportation mode preferences. Cooperation between public transport providers and shortest route finding platforms is also recognized to improve the information available. One of the relevant algorithms in shortest route search is Uniform Cost Search (UCS), which works by considering the cost of each node in a road network graph or map. Validation from the second test showed that the results of the application with the route of Rungkut District→Sukolilo District→Gubeng District has a distance of 7,600 meters, which is approximately the same as the results provided by Google Maps with a similar route and a distance of 7,700 meters. Based on the results of research on the implementation of the UCS algorithm for the shortest route search in the city of Surabaya, it can be concluded that the implementation of the UCS algorithm on the shortest route search in the city is very effective and can minimize operational costs. This algorithm successfully overcomes the problem of route search which initially takes a long time and has high operational costs, becomes more efficient with faster time and low operational costs. The results of this shortest route search prove its efficiency in reducing operational costs and reaching destinations faster. Validation of the application results also demonstrates conformity with the routes provided by Google Maps, confirming the reliability of the UCS algorithm implementation in the context of finding the shortest route in urban environments.

Keywords: *Uniform Cost Search Algorithm, Shortest Route*

الوخلص

أبرليندو ، دياز رزقي. ٢٠٢٣."**تنفيذ خوارزمية البحث عن التكلفة الموحدة لتحديد أقصر طريق** ". فرضية. برنامج دراسة هندسة المعلوماتية ،كلية العلوم والتكنولوجيا ، جامعة الولاية الإسلامية مولانا مالك إبراهيم المستشارون : (١) سوبريونو ،الماجستير (٢)دكتور. فريسي نوجروهو، ماجستير في الهندسة

الكلماتالرئيسية :خوارزمية بحث التكلفة الموحدة ، أقصر طريق

في هذا الوقت ، يعد البحث عن أقصر طريق هو المفتاح الرئيسي للتنقل الفعال ، خاصة في خضم الكثافة السكانية والنمو الحضري السريع .إن الحاجة الملحة للعثور على أسرع مسار لا تتعلق فقط باختيار أقصر مسار ، ولكن أيضا استراتيجية للتغلب على الازدحام ، وتحسين وقت السفر ، وتقليل التأثير البيئي .يتضمن البحث عن أقصر طريق جوانب ذات صلة مثل تجنب الاختناقات المرورية ، وتنظيم تسليم البضائع ، وضمان إمكانية الوصول والسلامة في حالات الطوارئ .ومع ذلك ، فإن العقبات مثل بيانات الموقع غير الدقيقة ، وعدم اليقين في ظروف الطرق ، وديناميكيات حركة المرور التي لا يمكن التنبؤ بها ، والتوافر المحدود لوسائل النقل العام هي تحديات يجب التغلب عليها .تتضمن الحلول للتغلب على هذه القيود تحسين دقة بيانات الموقع ومعلومات حركة المرور في الوقت الفعلي وتطوير خوارزميات تأخذ في الاعتبار تفضيلات وضع النقل .ومن المسلم به أيضا التعاون بين مقدمي خدمات النقل العام وأقصر منصات البحث عن الطرق لتحسين المعلومات المتاحة .واحدة من الخوارزميات ذات الصلة في أقصر بحث عن الطريق هي .والتي تعمل من خلال النظر في تكلفة كل عقدة في الرسم البياني لشبكة الطرق أو الخريطة ، (UCS) البحث عن التكلفة الموحدة لها مسافة Gubeng منطقة Sukolilo منطقة Rungkut أظهر التحقق من الاختبار الثاني أن نتائج التطبيق مع مسار منطقة سبعة آلاف وستة مائة متر ، وهي تقريبا نفس النتائج التي قدمتها خرائط جوجل مع مسار مماثل ومسافة سبعة آلاف وسبعة مائة لأقصر بحث عن طريق في مدينة سورابايا ، يمكن الاستنتاج أن تنفيذ UCS متر .استنادا إلى نتائج البحث حول تنفيذ خوارزمية على أقصر بحث عن الطرق في المدينة فعال للغاية ويمكن أن يقلل من تكاليف التشغيل .تتغلب هذه الخوارزمية UCS خوارزمية بنجاح على مشكلة البحث عن المسار التي تستغرق وقتا طويلا في البداية ولها تكاليف تشغيلية عالية ، وتصبح أكثر كفاءة مع وقت أسرع وتكاليف تشغيل منخفضة .تثبت نتائج هذا البحث عن أقصر الطرق كفاءته في تقليل التكاليف التشغيلية والوصول إلى الوجهات بشكل أسرع .يوضح التحقق من صحة نتائج التطبيق أيضا التوافق مع المسارات التي توفرها خرائط جوجل ، مما يؤكد .في سياق العثور على أقصر طريق في البيئات الحضرية UCS موثوقية تنفيذ خوارزمية

BAB I
PENDAHULUAN

1.1 Latar Belakang

Pada era mobilitas modern yang penuh tantangan, pencarian rute terpendek telah menjadi pijakan utama bagi mobilitas efisien. Di tengah kepadatan populasi dan pertumbuhan perkotaan yang pesat, kebutuhan akan menemukan jalur tercepat dan paling efisien menjadi semakin mendesak. Rute terpendek bukan hanya sekadar pemilihan jalur terpendek, melainkan juga merupakan strategi untuk mengatasi kemacetan, mengoptimalkan waktu tempuh, dan meminimalisir dampak lingkungan (Faisal, 2017).

Pencarian rute terpendek melibatkan berbagai aspek yang saling terkait. Di satu sisi, mobilitas perkotaan yang semakin kompleks memicu perlunya mencari solusi dalam menghindari kemacetan lalu lintas yang kerap melibatkan waktu perjalanan yang signifikan. Dalam skenario bisnis dan logistik, pencarian rute terpendek memainkan peran penting dalam mengatur pengiriman barang dan menghemat biaya (Nugroho et al., 2017). Di sisi lain, kebutuhan akan aksesibilitas dan keamanan dalam situasi darurat mendorong perlunya mempercepat pencarian rute terpendek.

Namun, pencarian rute terpendek juga menghadapi beberapa kendala yang perlu diatasi. Ketidakakuratan data lokasi dan ketidakpastian kondisi jalan seringkali menghasilkan rekomendasi rute yang kurang optimal. Selain itu, dinamika lalu lintas yang tidak bisa diprediksi dapat menyebabkan pemilihan rute terpendek yang masih terjebak dalam kemacetan. Ketersediaan moda transportasi

umum yang terbatas dan aksesibilitas yang buruk bagi kelompok tertentu juga menjadi hambatan dalam pencarian rute terdekat (Nafiah, 2020).

Solusi seperti peningkatan akurasi data lokasi dan informasi lalu lintas waktu nyata menjadi esensial dalam memberikan rekomendasi rute yang lebih baik. Pengembangan algoritma yang mempertimbangkan faktor-faktor seperti preferensi moda transportasi dan ketersediaan jalan akan membantu menghasilkan rekomendasi rute yang lebih akurat dan sesuai (Sabilla & Taufiq, 2022). Kerjasama antara penyedia transportasi umum dengan platform pencarian rute terpendek dapat meningkatkan informasi yang tersedia dan mengoptimalkan penggunaan transportasi umum.

Salah satu algoritma yang digunakan dalam pencarian rute terpendek adalah *Uniform Cost Search* (UCS). UCS adalah algoritma yang bekerja dengan mempertimbangkan biaya dari setiap simpul dalam graf yang merepresentasikan jaringan jalan atau peta (Sun et al., 2017). Algoritma ini fokus pada simpul dengan biaya terendah, sehingga dapat menemukan rute dengan jarak minimum.

Penerapan algoritma UCS dalam pencarian rute terpendek sangat relevan karena algoritma ini mampu mencari solusi dengan jarak terpendek. Dalam konteks pencarian rute terpendek, biaya sering kali berkaitan dengan faktor-faktor seperti jarak, waktu tempuh, atau kriteria lain yang ingin dioptimalkan. Kemampuan UCS untuk menghitung biaya terendah secara efisien menjadikannya alat yang kuat dalam menangani permasalahan pencarian rute terpendek (Ramadhan, 2022).

Penelitian yang dilakukan oleh William Manuel Kurniawan menerapkan algoritma USC untuk menentukan rute terbaik menuju ITB. Penelitian ini bertujuan

untuk memanfaatkan Algoritma *Uniform Cost Search* (UCS) dalam mencari rute terpendek dan optimal untuk mencapai Institut Teknologi Bandung (ITB). Metode UCS digunakan untuk menemukan jalur dengan biaya terendah dalam perjalanan menuju ITB. Dalam penelitian ini, penerapan algoritma UCS dilakukan dengan memodelkan jaringan jalan dan jarak antarlokasi menuju ITB. Hasil penelitian menunjukkan bahwa penerapan algoritma UCS berhasil menghasilkan rute terpendek dan efisien menuju ITB berdasarkan biaya perjalanan. Algoritma ini mampu mengatasi tantangan kompleksitas jalan dan variasi biaya yang mungkin terjadi dalam perjalanan. Dengan penerapan UCS, pengguna dapat mengidentifikasi jalur terbaik berdasarkan biaya dan jarak yang dioptimalkan, sehingga membantu dalam pengambilan keputusan perjalanan yang lebih efisien. Penelitian ini memiliki potensi aplikasi dalam navigasi dan perencanaan perjalanan sehari-hari, memberikan kontribusi dalam meningkatkan efisiensi mobilitas dan waktu perjalanan (Kurniawan, 2022).

Ayat Al-Quran secara langsung tidak membahas tentang pencarian rute terpendek menggunakan teknologi navigasi modern seperti yang dikenal saat ini. Namun, prinsip-prinsip yang ada dalam Al-Quran dapat diterapkan dalam penggunaan teknologi ini. Salah satu ayat yang dapat dihubungkan dengan menolong sesame manusia dalam mencari rute terpendek adalah:

وَتَعَاوَنُوْا عَلَى الْبِرِّ وَالتَّقْوٰىۖ وَلَا تَعَاوَنُوْا عَلَى الْاِثْمِ وَالْعُدْوَانِۖ وَاتَّقُوا اللّٰهَۗ اِنَّ اللّٰهَ شَدِيْدُ الْعِقَابِ

"Dan tolong-menolonglah kamu dalam mengerjakan kebajikan dan takwa, dan jangan tolong-menolong dalam perbuatan dosa dan permusuhan. Bertakwalah kepada Allah, sesungguhnya Allah sangat berat siksaan-Nya."(QS. Al-Maidah:02).

Tafsir Al-Jalalain menjelaskan bahwa pada surat Al-Maidah Ayat 2, Allah SWT. menyuruh umat-Nya untuk saling tolong-menolong dalam kebaikan, ketaatan, dan ketakwaan. Mereka diingatkan agar tidak bersama-sama dalam perbuatan dosa dan permusuhan. Prinsip ini menekankan pentingnya kolaborasi dalam memperjuangkan kebaikan dan menghindari kerjasama dalam tindakan yang bertentangan dengan nilai-nilai agama dan norma moral. Pada bagian "Bertakwalah kepada Allah SWT., sesungguhnya Allah SWT. sangat berat siksaan-Nya" menjelaskan bahwa sikap hati yang penuh kesadaran dan ketaatan kepada perintah-perintah Allah SWT serta menjauhi larangan-Nya. Peringatan tentang beratnya siksaan Allah SWT bertujuan untuk memberikan insentif kepada umat-Nya agar selalu menjaga ketakwaan dan menghindari perbuatan dosa.

Ayat 02 surat Al-Maidah dalam Al-Quran menekankan etika muamalah dalam islam. Menitikberatkan pada prinsip-prinsip etika dan moral yang mendasari interaksi sosial. Umat Islam diajak untuk saling tolong-menolong dalam mengerjakan kebajikan dan memelihara takwa, menggarisbawahi pentingnya kebaikan dan kesalehan sebagai dasar muamalah. Larangan untuk tidak terlibat dalam perbuatan dosa dan permusuhan memandu umat agar menjauhi perilaku negatif dan merawat keharmonisan hubungan antarindividu. Puncaknya, muamalah Islam diberkahi dengan nasihat untuk bertakwa kepada Allah SWT., mengingatkan setiap tindakan dan interaksi dalam kerangka ketaatan kepada-Nya. Kesadaran akan beratnya siksaan Allah SWT. memberikan dimensi moral yang kuat, mendorong individu untuk membangun muamalah yang adil, penuh kasih, dan berlandaskan pada nilai-nilai keislaman.

1.2 Pernyataan Masalah

Berdasarkan dengan identifikasi masalah yang ada, maka pernyataan masalah pada penelitian ini adalah bagaimana mengimplementasikan algoritma *Uniform Cost Search* (UCS) pada pencarian rute terpendek?

1.3 Tujuan Penelitian

Pada penelitian yang dilakukan memiliki tujuan untuk mengimplementasikan algoritma *Uniform Cost Search* (UCS) pada pencarian rute terpendek.

1.4 Manfaat Penelitian

Pada penelitian yang dilakukan diharapkan dapat memberikan manfaat sebagai berikut:

1. Peningkatan Akurasi Navigasi: Implementasi metode *Uniform Cost Search* (UCS) pada pencarian rute terpendek dapat meningkatkan akurasi rekomendasi rute.
2. Optimalitas: UCS menjamin solusi optimal. Artinya, algoritma ini akan menemukan jalur dengan biaya minimum dari titik awal ke tujuan. Ini membuatnya sangat berguna dalam skenario di mana Anda ingin mencari solusi terbaik dalam hal biaya atau waktu.
3. Penerapan pada Graf dengan Bobot: Jika masalah yang Anda hadapi melibatkan graf dengan bobot pada setiap sisi (edge), UCS adalah pilihan yang baik. Algoritma ini menghitung biaya total dari titik awal ke setiap simpul dan secara sistematis memilih jalur dengan biaya minimum.

1.5 Batasan Masalah

Pada penelitian yang dilakukan memiliki batasan masalah sebagai berikut ini: Program menggunakan jalan di Kota Surabaya.

1.6 Sistematika Penulisan

BAB I PENDAHULUAN

Bab ini berisi latar belakang, pernyataan masalah, tujuan penelitian, hipotesis, manfaat penelitian, batasan masalah, dan sistematika penulisan laporan skripsi.

BAB II STUDI PUSTAKA

Bab ini berisi tentang materi yang mendukung rute optimal dalam pengiriman barang menggunakan algoritma *Uniform Cost Search* (UCS).

BAB III DESAIN PENELITIAN

Bab ini berisi tentang desain dan implementasi yang menjelaskan tentang pencarian rute optimal pada proses pengiriman barang dengan mengimplementasikan algoritma *Uniform Cost Search* (UCS).

BAB IV HASIL DAN PEMBAHASAN

Bab ini berisi tentang perhitungan dengan implementasi algoritma *Uniform Cost Search* (UCS) pada excel dan menjelaskan tentang pengkodean program menggunakan bahasa *python* dengan implementasi algortima *Uniform Cost Search* (UCS).

BAB V KESIMPULAN DAN SARAN

Bab ini berisi tentang kesimpulan berdasarkan hasi penelitian yang telah dilakukan. Pada bab ini juga berisi saran yang ditujukan kepada penulis untuk digunakan sebagai bahan pertimbangan bagi peneliti setelahnya yang akan mengembangkan penelitian serupa.

BAB II
STUDI PUSTAKA

2.1 Penelitian Terkait

Penelitian (Asmara et al., 2021) yaitu “Penerapan Algoritma *Uniform Cost Search* (UCS) untuk Rekomendasi Pembangunan Rumah” bertujuan untuk memberikan solusi dalam pemilihan rekomendasi lokasi dan urutan pembangunan rumah yang optimal. Dalam menghadapi tantangan tersebut, penelitian ini menerapkan algoritma *Uniform Cost Search* (UCS) yang dapat mengidentifikasi jalur dengan biaya terendah dalam ruang pencarian. Melalui pendekatan ini, penelitian berhasil menghasilkan rekomendasi yang efisien berdasarkan faktor-faktor biaya yang seragam, mengarah pada pemilihan urutan dan lokasi pembangunan rumah yang memberikan hasil optimal secara ekonomis.

Penelitian (Kurniawan, 2022) yaitu Penenrapan Algoritma USC Untuk Menentukan Rute Terbaik Menuju ITB. Penelitian ini bertujuan untuk memanfaatkan Algoritma *Uniform Cost Search* (UCS) dalam mencari rute terpendek dan optimal untuk mencapai Institut Teknologi Bandung (ITB). Metode UCS digunakan untuk menemukan jalur dengan biaya terendah dalam perjalanan menuju ITB. Dalam penelitian ini, penerapan algoritma UCS dilakukan dengan memodelkan jaringan jalan dan jarak antarlokasi menuju ITB. Hasil penelitian menunjukkan bahwa penerapan algoritma UCS berhasil menghasilkan rute terpendek dan efisien menuju ITB berdasarkan biaya perjalanan. Algoritma ini mampu mengatasi tantangan kompleksitas jalan dan variasi biaya yang mungkin terjadi dalam perjalanan. Dengan penerapan UCS, pengguna dapat

mengidentifikasi jalur terbaik berdasarkan biaya dan jarak yang dioptimalkan, sehingga membantu dalam pengambilan keputusan perjalanan yang lebih efisien. Penelitian ini memiliki potensi aplikasi dalam navigasi dan perencanaan perjalanan sehari-hari, memberikan kontribusi dalam meningkatkan efisiensi mobilitas dan waktu perjalanan.

Penelitian (Galih & Putra, 2021) yaitu Penerapan Algoritma UCS untuk Pencarian Rute Terpendek Pengiriman Surat di Kota Probolinggo. Penelitian ini menerapkan Algoritma *Uniform Cost Search* (UCS) dalam menemukan rute terpendek untuk pengiriman surat di Kota Probolinggo. Metode UCS digunakan untuk menemukan jalur dengan biaya terendah dalam pengiriman surat di jaringan jalan kota tersebut. Melalui penerapan algoritma UCS, penelitian ini berhasil mencapai hasil yang signifikan. Hasilnya menunjukkan bahwa algoritma UCS secara efektif dapat mengidentifikasi rute terpendek untuk pengiriman surat di Kota Probolinggo, dengan mempertimbangkan biaya dan jarak tempuh. Implementasi ini dapat membantu layanan pengiriman dalam meningkatkan efisiensi pengantaran surat dengan memilih rute yang paling efektif. Dengan demikian, penelitian ini memiliki implikasi praktis dalam bidang logistik dan pengiriman, serta memberikan panduan yang lebih baik bagi pengiriman surat di wilayah kota yang padat lalu lintas seperti Probolinggo.

2.2 Landasan Teori

Pada landasan teori membahas beberapa cakupan diantaranya definisi dan metode yang dipakai dalam proses penelitian. Oleh sebab itu dalam sebuah

penelitian peran dari landasan teori sangatlah penting untuk menjadi hipotesis sementara pada penelitian.

2.2.1 Geospasial

Geospatial, atau data geospasial, merujuk pada informasi yang terkait dengan lokasi atau wilayah geografis di Bumi. Pemahaman mendalam tentang data geospasial sangat penting dalam berbagai konteks penelitian, mulai dari ilmu geografi, ilmu lingkungan, hingga aplikasi teknologi informasi. Salah satu alat utama untuk bekerja dengan data geospasial adalah Sistem Informasi Geografis (GIS) (Pahleviannur, 2019) (Saputra, 2011), yang memungkinkan pengumpulan, penyimpanan, analisis, dan visualisasi data berbasis lokasi. Melibatkan koordinat geografis, batas wilayah, dan atribut terkait lokasi, geospatial menjadi dasar bagi banyak disiplin ilmu yang bergantung pada pemahaman hubungan spasial (Endrayanto & Muttaqin, 2021).

Penggunaan data geospasial berkembang pesat dalam penelitian lintas disiplin ilmu. Pemetaan dan analisis spasial memungkinkan para peneliti untuk mengeksplorasi pola dan tren yang terkait dengan lokasi tertentu. Dalam ilmu lingkungan, data geospasial memfasilitasi pemantauan dan pemahaman terhadap perubahan lingkungan seperti deforestasi, perubahan iklim, dan keanekaragaman hayati (Pahleviannur, 2019). Penelitian geospasial juga penting dalam konteks kesehatan masyarakat, dengan memetakan persebaran penyakit, sumber air bersih, atau infrastruktur kesehatan.

Penerapan teknologi geospatial juga memberikan kontribusi signifikan dalam perencanaan perkotaan dan pengembangan wilayah (Abidin, 2022).

Keputusan terkait dengan lokasi infrastruktur, zonasi penggunaan lahan, dan mitigasi risiko bencana semuanya dapat didukung oleh analisis data geospasial. Seiring dengan kemajuan teknologi seperti pemantauan satelit dan sensor GPS, informasi geospasial menjadi semakin akurat dan dapat diakses (Fawzi & Husna, 2021). Dengan demikian, geospatial tidak hanya menjadi alat penelitian, tetapi juga fondasi untuk pengambilan keputusan yang cerdas dan berkelanjutan dalam berbagai aspek kehidupan manusia.

Pengembangan geospasial pada penelitian ini memiliki peran yang sangat penting, dan salah satu alat yang digunakan adalah pustaka Python yang disebut Folium (Kennedy et al., 2018). Dalam konteks penelitian, geospasial mencakup analisis dan visualisasi data yang memiliki komponen lokasional atau spasial. Penggunaan library Folium memberikan keuntungan signifikan karena memfasilitasi pembuatan peta interaktif dengan mudah tanpa memerlukan keahlian mendalam dalam pengembangan web atau JavaScript.

Folium adalah sebuah pustaka (library) dalam bahasa pemrograman Python yang digunakan untuk membuat peta interaktif secara intuitif dan efisien (Wu, 2021). Berdasarkan pustaka JavaScript Leaflet.js, Folium memungkinkan pengguna untuk menghasilkan visualisasi geospasial yang menarik dengan mudah melalui antarmuka Python (Kennedy et al., 2018). Dengan menggunakan Folium, pengguna dapat membuat peta web interaktif, menambahkan elemen-elemen seperti marker, polyline, dan polygon, serta menyertakan informasi tambahan yang dapat diakses saat berinteraksi dengan peta tersebut. Pustaka ini menjadi alat populer dalam analisis data geospasial dan memungkinkan pengembang dan analis

data untuk mengintegrasikan elemen geografis dalam proyek mereka tanpa memerlukan keahlian khusus dalam pengembangan web

2.2.2 Teori Graf

Graf dapat merepresentasikan berbagai struktur dan memberikan solusi untuk berbagai masalah. Sebagai struktur data yang paling umum, graf G dapat didefinisikan sebagai pasangan himpunan (V, E), dengan notasi $G = (V, E)$. Di sini, V adalah himpunan simpul yang tidak boleh kosong, sedangkan E adalah himpunan sisi yang menghubungkan sepasang simpul.

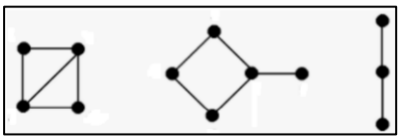
Pada konteks berarti V tidak boleh kosong, namun E bisa kosong. Oleh karena itu, sebuah graf memungkinkan untuk tidak memiliki satu sisi pun, tetapi minimal harus memiliki satu simpul. Graf yang hanya memiliki satu simpul tanpa sisi disebut sebagai graf trivial (Nggufron & Mashuri, 2019).

Graf, sebagai representasi visual dari keterhubungan antara simpul-simpul, memiliki kemungkinan pengelompokan yang beragam tergantung pada beberapa kriteria tertentu. Dalam memahami keragaman struktur graf, pendekatan pengelompokan menjadi suatu aspek penting. Salah satu metode pengelompokan yang dapat diterapkan adalah dengan mempertimbangkan keberadaan sisi ganda, di mana suatu graf mungkin termasuk jenis graf tak-sederhana atau multigraf, memungkinkan adanya lebih dari satu tepi yang menghubungkan sepasang simpul tertentu. Selain itu, pengelompokan dapat dilakukan berdasarkan jumlah simpul yang terdapat dalam graf, memberikan gambaran tentang kompleksitas dan ukuran jaringan. Pendekatan lainnya adalah dengan mempertimbangkan orientasi sisi, yang berkaitan dengan arah hubungan antar simpul. Konsep pengelompokan graf ini,

seperti yang dijelaskan oleh (Pahleviannur, 2019). Membuka pemahaman yang lebih mendalam terkait dengan karakteristik dan struktur graf dalam konteksnya yang beragam, sehingga mampu memberikan pandangan yang lebih holistik terhadap keterkaitan dan pola dalam suatu sistem atau jaringan. Berdasarkan ada tidaknya sisi ganda atau ada tidaknya loop, graf dapat dikelompokkan menjadi dua jenis, yaitu:

1. Graf Sederhana

Graf sederhana, atau yang dikenal sebagai simple graph, adalah jenis graf yang tidak mengandung gelang atau sisi ganda. Dalam graf sederhana, setiap tepi menghubungkan dua simpul tanpa adanya sisi yang menghubungkan simpul tersebut kembali atau sisi rangkap. Dengan kata lain, setiap pasangan simpul hanya dihubungkan oleh tepi yang tunggal, menjadikan struktur graf lebih sederhana dan mudah dipahami (Rahmadi & Sandariria, 2023). Konsep graf sederhana memiliki relevansi yang signifikan dalam berbagai konteks, termasuk representasi jaringan, analisis relasi antar-entitas, dan pemodelan hubungan dalam berbagai sistem. Berikut ini gambar graf sederhana. Berikut ini gambar graf sederhana

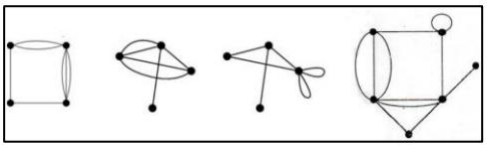


Gambar 2. 1 Graf Sederhana

2. Graf Tak-Sederhana

Graf tak-sederhana, yang sering juga disebut sebagai multigraf, adalah suatu jenis struktur graf yang menampilkan kekayaan hubungan yang lebih kompleks antara simpul-simpulnya. Dalam graf tak-sederhana, entitas atau simpul tidak hanya dihubungkan oleh satu sisi, tetapi mungkin memiliki lebih dari satu tepi yang

menghubungkan pasangan simpul tertentu, membentuk apa yang dikenal sebagai sisi ganda. Di samping itu, kemungkinan terjadinya gelang atau sirkuit semakin terbuka lebar, karena sisi ganda memungkinkan adanya jalur yang kembali ke simpul awal, menciptakan struktur yang lebih beragam dan kompleks. Konsep ini menjadi sangat penting dalam memodelkan situasi di mana interaksi atau hubungan antar-entitas bersifat lebih fleksibel dan dapat terjadi melalui jalur-jalur yang berbeda, memberikan gambaran yang lebih akurat dan lengkap tentang keterkaitan antar unsur-unsur dalam suatu sistem atau jaringan. Berikut ini merupakan gambar graf tak sederhana:

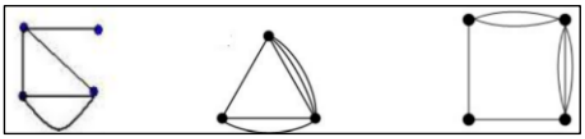


Gambar 2. 2 Graf Tak-Sederhana

Pada graf tak-sederhana dibagi menjadi 2 jenis yaitu:

1. Graf Ganda (*Multi Graph*)

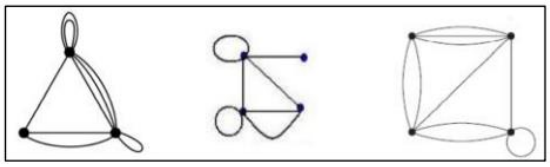
Graf ganda adalah jenis graf yang memiliki sisi ganda, di mana terdapat lebih dari satu tepi yang menghubungkan sepasang simpul tertentu. Konsep ini menandai adanya kemungkinan koneksi ganda antara simpul-simpul dalam struktur graf, membentuk suatu hubungan yang lebih kompleks dan variatif. Berikut ini adalah gambar graf ganda:



Gambar 2. 3 Graf Ganda

2. Graf Semu (Pseudo Graph)

Graf semu adalah jenis graf yang mencakup sisi gelang, yaitu suatu jalur tertutup dalam graf yang terdiri dari urutan simpul yang sama. Dengan adanya sisi gelang, graf semu memungkinkan terbentuknya pola hubungan yang lebih kompleks antara simpul-simpulnya, menciptakan struktur yang lebih kaya dan beragam. Konsep ini memberikan pemahaman yang lebih mendalam tentang kemungkinan jalur kembali atau sirkuit dalam konteks hubungan antar-entitas dalam suatu sistem atau jaringan. Berikut ini adalah gambar graf semu:

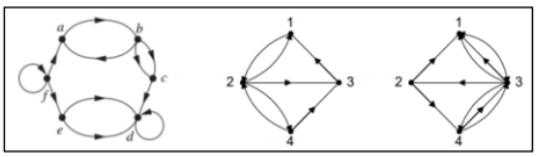


Gambar 2. 4 Graf Semu

Berdasarkan orientasi arah pada sisi, graf dapat diklasifikasikan menjadi dua jenis yaitu:

1. Graf Berarah (*Directed Graph*)

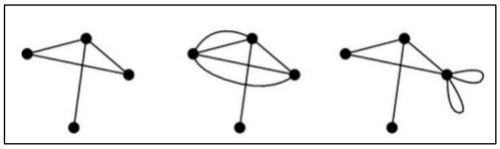
Graf berarah adalah jenis graf yang ditandai dengan adanya orientasi arah pada setiap sisi. Dalam struktur ini, setiap tepi memiliki arah tertentu yang menunjukkan hubungan satu arah antara dua simpul. Graf berarah memberikan representasi yang lebih kaya tentang alur atau arah pergerakan dalam suatu jaringan, membantu pemahaman yang lebih mendalam tentang hubungan antar-entitas. Berikut ini adalah gambar graf berarah:



Gambar 2. 5 Graf Berarah

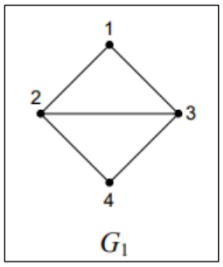
2. Graf Tak-Berarah (*Indirected Graph*)

Graf tak-berarah adalah jenis graf yang ditandai dengan ketiadaan orientasi arah pada setiap sisinya. Dalam graf ini, setiap tepi tidak memiliki arah khusus, yang berarti hubungan antara dua simpul bersifat dua arah atau saling ketergantungan. Konsep graf tak-berarah memberikan fleksibilitas dalam merepresentasikan hubungan non-direksional antar-entitas, di mana tidak ada arah yang ditetapkan pada setiap sisi. Berikut ini adalah gambar graf tak-berarah:



Gambar 2. 6 Graf Tak-Berarah

Pada konteks struktur graf, terdapat sejumlah istilah atau terminologi yang digunakan untuk merinci dan menggambarkan berbagai elemen dan hubungan di dalamnya. Beberapa di antaranya melibatkan konsep simpul, tepi, bobot, dan sebagainya, yang secara kolektif membentuk dasar pemahaman dalam analisis dan representasi graf.



Gambar 2. 7 Graf GI

Dalam konteks graf, konsep ketetanggaan merujuk pada hubungan antara dua simpul yang dikatakan bertetangga jika keduanya terhubung langsung oleh sebuah sisi. Ilustratifnya, pada Gambar 2.7, dapat dilihat bahwa simpul 1 memiliki tetangga langsung yaitu simpul 2 dan simpul 3, menggambarkan bagaimana ketetanggaan membangun jaringan keterhubungan dalam graf.

Istilah bersisian berkaitan dengan keberadaan sebuah sisi e yang dikatakan bersisian dengan simpul vi dan simpul vj. Artinya, sisi ini menghubungkan kedua simpul tersebut. Pada Gambar 2.7, sisi 1 bersisian dengan simpul 2 dan simpul 3, memberikan gambaran visual tentang cara bersisian mengindikasikan keterkaitan antar-simpul.

Derajat suatu simpul mencerminkan jumlah sisi yang bersisian dengannya atau jumlah simpul lain yang terhubung dengannya. Notasi d(v) digunakan untuk graf tidak berarah, sementara pada graf berarah terdapat din(v) dan dout(v) untuk menyatakan jumlah sisi yang bersisian dengan arah masuk dan keluar. Sebagai contoh, simpul 1 pada Gambar 2.7 memiliki derajat d(1) sebesar 2, menunjukkan keterhubungannya dengan dua simpul lainnya.

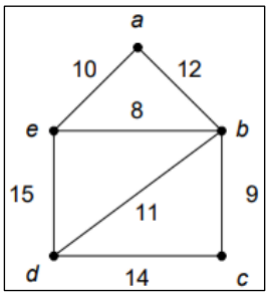
Lintasan adalah barisan simpul dan sisi yang membentuk jalur dari simpul awal v0 ke simpul tujuan vn dalam graf G. Misalnya, lintasan v0, e1, v1, e2, v2, ..., vn-1, en, vn membangun jalur yang melibatkan simpul-simpul dan sisi-sisi yang berurutan pada graf. Konsep ini memberikan pemahaman lebih mendalam tentang perjalanan dari satu simpul ke simpul lainnya.

Sirkuit adalah lintasan yang membentuk jalur berawal dan berakhir pada simpul yang sama. Sirkuit menciptakan suatu siklus tertutup dalam graf, menyoroti potensi perulangan lintasan dari dan kembali ke simpul awal.

Keterhubungan antar-simpul, seperti v1 dan v2, terjadi ketika terdapat lintasan dari v1 ke v2. Graf G disebut graf terhubung (connected graph) jika setiap pasang simpul vi dan vj dalam himpunan V memiliki lintasan yang menghubungkan

keduanya. Konsep ini menekankan keterkaitan dan aksesibilitas antar-simpul dalam suatu graf.

Graf berbobot adalah jenis graf di mana setiap sisi diberi sebuah nilai atau bobot tertentu. Bobot ini memberikan informasi tambahan tentang sisi-sisi dalam graf, menggambarkan nilai atau kriteria tertentu yang terkait dengan setiap hubungan. Konsep ini memberikan dimensi nilai tambahan dalam menganalisis dan memahami graf berbobot.

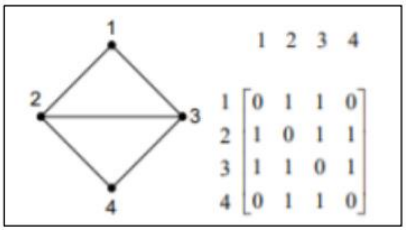


Gambar 2. 8 Graf Berbobot

Pada praktiknya di dunia pemrograman, representasi graf dapat diartikan melalui tiga bentuk utama yaitu:

1. Matriks Ketetanggaan (Adjacency Matrix)

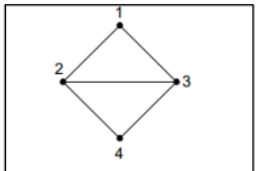
Pada konsep ini, setiap simpul dalam graf direpresentasikan oleh baris dan kolom dalam matriks. Dalam konteks graf tak berbobot, elemen-elemen matriks ini dapat berisi nilai 1 atau 0. Dalam matriks ketetanggaan, elemen aij dari matriks A, di mana i dan j adalah indeks baris dan kolom, mewakili hubungan antara simpul i dan j. Jika simpul i dan j bertetangga, nilai aij akan diatur sebagai 1; sebaliknya, jika simpul i dan j tidak bertetangga, nilai aij akan diatur sebagai 0. Dengan pendekatan ini, matriks ketetanggaan memberikan representasi yang jelas dan terstruktur mengenai keterhubungan antar-simpul dalam graf tak berbobot.



Gambar 2. 9 Representasi dengan Matrix Ketetanggan

2. *Adjacency List*

Dalam representasi graf, terdapat pula metode Adjacency List, di mana keterhubungan antar simpul diartikan dengan menggunakan linked list. Pada pendekatan ini, setiap simpul dalam graf memiliki suatu daftar (list) yang berisikan simpul-simpul yang terhubung secara langsung dengan simpul tersebut. Konsep ini memanfaatkan struktur data linked list untuk menyajikan keterhubungan antar simpul, memungkinkan pengelolaan dinamis dan efisien terhadap hubungan graf yang bersifat berubah-ubah. Dengan menggunakan Adjacency List, informasi mengenai tetangga-tetangga setiap simpul dapat diakses secara efisien melalui struktur data linked list yang terkait.



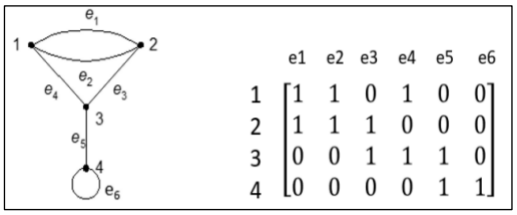
Simpul	Simpul Tetangga
1	2, 3
2	1, 3, 4
3	1, 2, 4
4	2, 3

Gambar 2. 10 Representasi dengan Adjacency List

3. Matriks Bersisian (*Incidency Matrix*)

Dalam matriks bersisian, setiap baris dan kolom matriks menunjukkan representasi hubungan antar simpul dalam graf tak berbobot. Matriks ini, dinyatakan sebagai A = [aij], mengandung elemen-elemen yang bernilai 1 atau 0,

sesuai dengan keterhubungan simpul i dan j. Jika simpul i dan j bertetangga, maka nilai aij akan diatur sebagai 1, mencerminkan adanya hubungan antara simpul-simpul tersebut. Di sisi lain, jika simpul i tidak bersisian dengan sisi j, nilai aij akan diatur sebagai 0, mengindikasikan ketiadaan hubungan antara simpul i dan sisi j. Dengan demikian, matriks bersisian memberikan representasi matematis yang jelas dan terstruktur mengenai keterhubungan antar simpul dalam konteks graf tak berbobot.



Gambar 2. 11 Representasi dengan Matriks Bersisian

2.2.3 Rute Terpendek

Rute terpendek mengacu pada jalur atau lintasan yang menawarkan jarak, biaya, atau waktu perjalanan minimal antara dua titik tertentu suatu jaringan atau sistem. Pentingnya konsep ini meluas ke berbagai bidang seperti navigasi, transportasi, logistik, dan telekomunikasi, di mana efisiensi pergerakan dan penggunaan sumber daya menjadi fokus utama.

Secara umum, rute terpendek mencerminkan usaha untuk mencapai tujuan dengan meminimalkan penggunaan sumber daya yang tersedia. Dalam hal transportasi dan navigasi, rute terpendek melibatkan identifikasi jalan atau lintasan dengan jarak paling singkat antara titik awal dan tujuan, dengan tujuan meminimalkan waktu dan bahan bakar yang dibutuhkan (Nafiah, 2020). Sementara dalam logistik dan pengiriman, perencanaan rute terpendek melibatkan strategi

untuk mengirim barang atau layanan dengan cara yang paling efisien, mengoptimalkan biaya dan waktu yang diperlukan. Menurut Edsger W. Dijkstra, rute terpendek didefinisikan sebagai jalur dengan jarak minimum di antara semua kemungkinan jalur antara dua titik dalam sebuah graf. Perspektif ini menunjukkan penekanan pada optimasi bobot atau biaya dalam pemilihan rute. Sedangkan menurut Ahmed K. Noor, rute terpendek mengacu pada jalur dengan jarak minimal yang memungkinkan perpindahan dari satu titik ke titik lain dalam lingkungan fisik atau jaringan. Lawrence R. Ford menambahkan bahwa rute terpendek adalah jalur yang meminimalkan fungsi biaya atau waktu dalam pergerakan dari satu titik ke titik lain dalam suatu konteks yang diberikan.

Keseluruhan konsep rute terpendek mencerminkan usaha lintas disiplin untuk mencapai tujuan dengan mempertimbangkan faktor-faktor seperti biaya, waktu, atau jarak secara optimal. Konsep ini telah mendapatkan perhatian luas dalam berbagai disiplin ilmu dan diterapkan dalam berbagai aplikasi praktis, memberikan kontribusi signifikan dalam pengembangan dan peningkatan efisiensi sistem dan jaringan

2.2.4 Uniform Cost Search (UCS)

Algoritma *Uniform Cost Search* (UCS) adalah algoritma pencarian graf yang digunakan untuk menemukan jalur dengan biaya terendah dari satu simpul awal ke simpul tujuan dalam graf berbobot. Algoritma ini berfokus pada mengeksplorasi jalur dengan biaya terkecil terlebih dahulu Menurut Thomas Cormen, Charles Leiserson, Ronald Rivest, dan Clifford Stein dalam buku "Introduction to Algorithms," UCS dijelaskan sebagai metode pencarian dengan

mengembangkan simpul-simpul berdasarkan urutan biaya terkecil. Algoritma ini menggunakan antrian prioritas sebagai struktur data utama, di mana simpul dengan biaya paling rendah ditempatkan di depan antrian. Hal ini memastikan bahwa pencarian dilakukan dengan mempertimbangkan jalur dengan biaya terkecil terlebih dahulu (H. Cormen et al., 2010). Berikut adalah langkah-langkah detail dari algoritma UCS:

1. Inisialisasi:
    a. Buat simpul awal sebagai simpul saat ini dengan biaya awal 0.
    b. Buat antrian prioritas (biasanya menggunakan priority queue) untuk menyimpan simpul yang akan dikelola. Prioritas diukur berdasarkan biaya total.
2. Loop Utama:

    Selama antrian prioritas tidak kosong, lakukan langkah-langkah berikut:

    a. Ambil simpul dengan biaya terendah dari antrian prioritas.
    b. Periksa apakah simpul tersebut adalah simpul tujuan. Jika ya, rekonstruksi jalur dan selesai.
    c. Jika bukan simpul tujuan, lakukan langkah berikut.
3. Eksplorasi Simpul Tetangga:
    a. Untuk setiap simpul tetangga yang terhubung dengan simpul saat ini, lakukan langkah-langkah berikut:
        1) Hitung biaya total untuk mencapai simpul tetangga ini melalui jalur saat ini.

2) Jika simpul tetangga belum pernah dieksplorasi sebelumnya atau jalur baru ini memiliki biaya lebih rendah daripada jalur sebelumnya, lakukan langkah berikut.

4. Pembaruan Antrian Prioritas:
   a. Tambahkan atau perbarui simpul tetangga dalam antrian prioritas dengan biaya total baru sebagai prioritas. Ini memastikan simpul dengan biaya terendah akan dikelola lebih dulu.
5. Pengulangan:
   a. Kembali ke langkah 2 dan terus lakukan iterasi sampai simpul tujuan ditemukan atau antrian prioritas kosong.
6. Rekonstruksi Jalur:

   Jika jalur terpendek ditemukan, rekonstruksi jalur dari simpul tujuan hingga simpul awal dengan melihat informasi langkah-langkah sebelumnya (misalnya, dengan menyimpan informasi parent dari setiap simpul).

Rumus perhitungan dalam algoritma Uniform Cost Search (UCS) digunakan untuk menghitung biaya total dari jalur yang sedang dieksplorasi. Biaya ini mencakup biaya dari simpul awal hingga simpul saat ini. Rumus ini memungkinkan algoritma UCS untuk memilih jalur dengan biaya terendah pada setiap langkah pencarian. Berikut adalah rumus perhitungan dalam algoritma UCS:

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

Keterangan:

F(n) = cost total dari jalur yang telah ditempuh hingga simpul n
g(n) = cost dari simpul awal hingga simpul n
h(n) = estimasi cost dari simpul n hingga simpul tujuan

Algoritma UCS memastikan bahwa jalur yang ditemukan adalah jalur dengan biaya terendah dari semua jalur yang dieksplorasi. Ini membuatnya optimal untuk pencarian rute terpendek dengan biaya seragam, tetapi bisa memerlukan sumber daya yang signifikan terutama dalam graf yang besar.

2.2.5 Global Positioning System (GPS)

Global Positioning System (GPS) adalah suatu teknologi yang berguna untuk menunjukkan posisi seseorang atau lokasi suatu tempat yang bisa digunakan kapan saja dan dimana saja, teknologi ini dirancang dengan memanfaatkan sebuah satelit yang ada di luar angkasa untuk mengirimkan sinyal yang dapat mendeteksi keberadaan posisi seseorang atau lokasi tempat tertentu. Penentuan posisi dengan menggunakan GPS dilakukan berdasarkan pengukuran jarak yang dilakukan ke beberapa satelit sekaligus.

GPS dibagi menjadi 2 metode yaitu:

1. Metode absolute atau yang disebut posisi titik, adalah metode pemosisian sederhana hanya pada satu penerima. Akurasi posisi di beberapa meter (tidak terlalu akurat) dan umumnya hanya dimaksudkan untuk tujuan navigasi.
2. Metode relatif atau dikenal sebagai metode differential positioning adalah memposisikan dengan lebih dari satu penerima. Satu GPS terpasang pada lokasi dimuka bumi dan terus menerus menerima sinyal satelit dalam jangka waktu tertentu dan dijadikan referensi seperti yang lain. Posisi dari metode ini memiliki akurasi yang tinggi (umum kurang dari satu meter) dan digunakan untuk tujuan survei ataupun pemetaan yang membutuhkan ketelitian tinggi (Siringoringo, 2016).

2.2.6 Google Maps API

Google Maps API adalah API yang disediakan oleh google dan dapat menampilkan secara gratis informasi-informasi geografis dalam bentuk map. Google maps API sangat membantu dalam menerjemahkan informasi geografis, contohnya seperti mentransformasikan titik-titik koordinat menjadi bentuk grafis yaitu map. Pada Google Maps API terdapat 4 jenis pemilihan model peta yaitu:

1. Roadmap, berfungsi untuk menampilkan peta 2 dimensi.
2. Satellite, berfungsi untuk menampilkan foto satelit.
3. Terrain, berfungsi untuk menampilkan relief fisik permukaan bumi, dan menampilkan tinggi suatu lokasi.
4. Hybrid, berfungsi untuk menunjukkan foro satelit dan diatasnya tergambar juga seperti yang tampil pada roadmap (jalan dan nama kota).

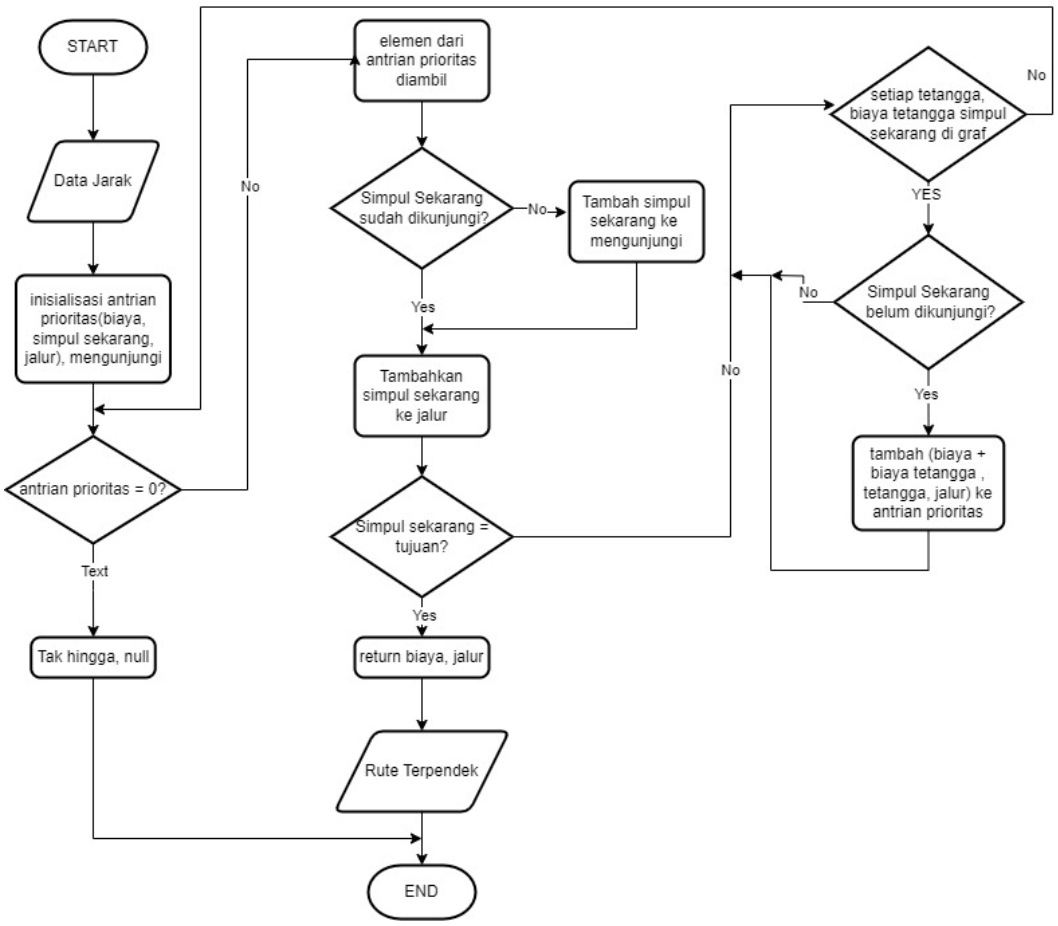
Penerapan Google Maps API pada aplikasi android yaitu dengan langkah berikut ini:

1. Melakukan registrasi di halaman web developer google maps untuk memperoleh key supaya bisa menggunakan layanan tersebut.
2. Menambahkan required permission ke dalam manifest aplikasi.
3. Menambahkan Google Play service ke dalam manifest aplikasi.
4. Membuat objek untuk menampilkan map di dalam class aplikasi (Siringoringo, 2016).

BAB III
DESAIN PENELITIAN

3.1 Desain Sistem

Desain sistem merupakan penerapan sistem yang digunakan untuk pengembangan aplikasi. Pada tahap ini memperlihatkan sistem yang akan dikembangkan dan dijelaskan dalam bentuk flowchart seperti gambar dibawah ini:



Gambar 3.1 *Flowchart* Desain Sistem

Gambar 3.1 menggambarkan diagram alir dari implementasi algoritma UCS untuk menentukan rute terpendek. Langkah awal melibatkan input data berupa jarak. Selanjutnya, proses beralih ke tahap algoritma UCS yang dimulai dengan

inisialisasi antrian prioritas. Antrian prioritas ini mencakup informasi biaya, simpul saat ini, dan jalur yang telah dilalui.

Langkah inisialisasi kedua adalah untuk menyimpan data yang telah dikunjungi dalam variabel bernama kunjung. Setelah langkah inisialisasi, algoritma memasuki tahap perulangan antrian prioritas. Jika kondisi antrian prioritas adalah 0, output yang dihasilkan adalah tak hingga (infinite), null, dan proses akan berakhir. Namun, jika kondisi tidak terpenuhi, elemen dari antrian prioritas diambil.

Selanjutnya, algoritma mengecek apakah simpul saat ini sudah dikunjungi. Jika sudah, perulangan tetap dilanjutkan; jika belum, simpul saat ini ditambahkan ke dalam variabel kunjung dan dimasukkan ke dalam jalur. Jika simpul saat ini adalah tujuan, algoritma akan mengembalikan nilai biaya dan jalur. Jika tidak, proses melanjutkan ke langkah berikutnya.

Algoritma kemudian memasuki perulangan untuk setiap tetangga. Biaya dari tetangga tersebut sudah terdapat dalam graf. Jika tetangga belum dikunjungi, biaya ditambahkan ke biaya dari tetangga, dan tetangga bersama jalurnya dimasukkan ke dalam antrian prioritas. Proses ini terus berlanjut dengan kembali ke perulangan awal hingga menemukan simpul tujuan. Berikut ini adalah pseudocode algoritma *uniform cost search*:

```
fungsi uniform_cost_search(graf, awal, tujuan):
    buat antrian prioritas kosong
    masukkan (0, awal, []) ke dalam antrian prioritas
# (biaya, simpul_sekarang, jalur)
    buat himpunan kosong untuk menyimpan simpul yang
telah dikunjungi
```

```
    selama antrian prioritas tidak kosong:
        biaya, simpul_sekarang, jalur = keluarkan
elemen pertama dari antrian prioritas
        jika simpul_sekarang sudah dikunjungi:
            lanjutkan
        tambahkan simpul_sekarang ke himpunan simpul
yang telah dikunjungi
        tambahkan simpul_sekarang ke jalur
        jika simpul_sekarang adalah tujuan:
            kembalikan biaya, jalur
        untuk setiap tetangga, biaya_tetangga di
tetangga simpul_sekarang di graf:
            jika tetangga belum dikunjungi:
                masukkan (biaya + biaya_tetangga,
tetangga, jalur) ke dalam antrian prioritas
    kembalikan tak hingga, null  # Jika tidak
ditemukan rute
inisialisasi jarak ke dalam array
tentukan titik awal dan titik akhir
biaya, jalur = uniform_cost_search(graf, simpul_awal,
simpul_tujuan)
jika jalur bukan null:
    cetak "Rute terpendek dari", simpul_awal, "ke",
simpul_tujuan, ":", jalur
    cetak "Total biaya:", biaya
jika tidak:
    cetak "Tidak ada rute yang memungkinkan dari",
simpul awal, "ke", simpul tujuan
```

Gambar 3.2 *Pseudo Code* Algoritma UCS

Penjelasan dari pseudo code diatas adalah:

1. Fungsi uniform_cost_search (graf, awal, tujuan): Ini adalah fungsi yang menerima graf, simpul awal, dan simpul tujuan sebagai parameter dan melakukan pencarian rute terpendek menggunakan algoritma UCS.
    a. Pertama, sebuah antrian prioritas kosong dibuat.
    b. (0, awal, []) dimasukkan ke dalam antrian prioritas. Ini adalah tuple yang berisi biaya saat ini (diawali dengan 0), simpul saat ini, dan jalur yang masih kosong.

c. Sebuah himpunan kosong dibuat untuk menyimpan simpul yang telah dikunjungi.
d. Selama antrian prioritas tidak kosong:
e. Elemen pertama dari antrian prioritas diambil (biaya, simpul_sekarang, jalur).
f. Jika simpul_sekarang sudah dikunjungi, maka perulangan dilanjutkan.
g. Simpul_sekarang ditambahkan ke himpunan simpul yang telah dikunjungi.
h. Simpul_sekarang ditambahkan ke jalur.
i. Jika simpul_sekarang adalah tujuan, maka fungsi mengembalikan biaya dan jalur.
j. Untuk setiap tetangga, biaya_tetangga di tetangga simpul_sekarang di graf:
k. Jika tetangga belum dikunjungi, maka (biaya + biaya_tetangga, tetangga, jalur) dimasukkan ke dalam antrian prioritas.

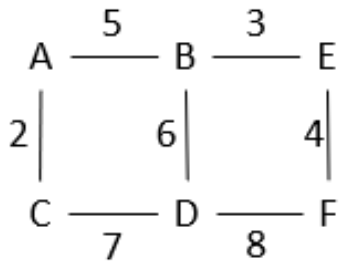
2. Inisialisasi jarak ke dalam array: Ini adalah langkah yang tidak dijelaskan dalam kode. Mungkin ini adalah langkah untuk menginisialisasi array jarak antara simpul-simpul di dalam graf.
3. Tentukan titik awal dan titik akhir: Di sini, Anda harus mengatur simpul awal dan simpul tujuan untuk pencarian rute.
4. Biaya, jalur = uniform_cost_search (graf, simpul_awal, simpul_tujuan): Anda memanggil fungsi uniform_cost_search dengan graf, simpul awal, dan simpul tujuan sebagai parameter, dan menyimpan biaya dan jalur hasil pencarian.
5. Jika jalur bukan null, maka jalur rute terpendek ditemukan:

a. Cetak pesan yang menyatakan rute terpendek dari simpul_awal ke simpul_tujuan, serta jalur yang ditempuh.
b. Cetak total biaya dari jalur.

6. Jika tidak ada rute yang ditemukan (jalur adalah null), cetak pesan bahwa tidak ada rute yang memungkinkan dari simpul_awal ke simpul_tujuan.

3.2 Implementasi Algoritma *Uniform Cost Search*

Penerapan Algoritma *Uniform Cost Search* (UCS) dalam menetukan rute terpendek telah meraih popularitas yang luas dalam berbagai konteks. Dalam tahapan perhitungan manual dari algoritma UCS, pada penenlitian terdahulu telah dilihat bagaimana algoritma USC secara cermat memilih jalur dengan biaya terendah untuk mencapai tujuan yang ditetapkan. Berikut ini perhitungan manual dari algoritma *uniform cost search*:



Gambar 3.3 Rute Setiap Simpul

Pada gambar 3.2, setiap huruf mewakili simpul (node), dan angka di antara paratheses () adalah bobot dari setiap lintasan yang menghubungkan dua simpul. Misalkan akan dicari rute terpendek dari simpul A ke simpul F menggunakan algoritma *Uniform Cost Search*, maka langkah-langkah perhitungannya adalah sebagai berikut:

Tabel 3. 1 Tabel prioritas

Iterasi	Simpul	Simpul Hidup

1	A	$C_{A=2}$ $B_{A=5}$
2	$C_{A=2}$	$B_{A=5}$ $D_{AC=9}$
3	$B_{A=5}$	$E_{AB=8}$ $D_{AC=9}$ $D_{AB=11}$
4	$E_{AB=8}$	$D_{AC=9}$ $D_{AB=11}$ $F_{ABE=12}$
5	$D_{AC=9}$	$D_{AB=11}$ $F_{ABE=12}$ $F_{ACD=17}$
6	$D_{AB=11}$	$F_{ABE=12}$ $F_{ACD=17}$ $F_{ABD=19}$
7	$F_{ABE=12}$	Solusi telah ditemukan

Inisialisasi:

Simpul awal: A.

Antrian prioritas awal: A (biaya = 0).

1. Iterasi 1

Keluarkan simpul A dari antrian. Perluas simpul A dan tambahkan tetangga-tetangganya ke dalam antrian prioritas:

a. Simpul C (biaya dari A ke C = 2),

b. Simpul B (biaya dari A ke B = 5),

c. Antrian prioritas sekarang: C_A (biaya = 2), B_A (biaya = 5).

2. Iterasi 2:

Keluarkan simpul C_A dari antrian. Perluas simpul C_A dan tambahkan tetangga-tetangganya ke dalam antrian prioritas:

a. Simpul A (sudah dieksplorasi sebelumnya, biaya dari A ke C = 2).

b. Simpul D (biaya dari A ke D via C= 9).

c. Antrian prioritas sekarang: B_A (biaya = 5), D_{AC} (biaya = 9).

3. Iterasi 3:

Keluarkan simpul B_A dari antrian. Perluas simpul B_A dan tambahkan tetangga-tetangganya ke dalam antrian prioritas:

a. Simpul A (sudah dieksplorasi sebelumnya, biaya dari A ke B = 5).

b. Simpul E (biaya dari A ke E via B = 8).

c. Simpul D (biaya dari A ke D via B = 11)

d. Antrian prioritas sekarang: E_{AB} (biaya = 8), D_{AC} (biaya = 9), D_{AB} (biaya = 11).

4. Iterasi 4:

Keluarkan simpul E_{AB} dari antrian. Perluas simpul E_{AB} dan tambahkan tetangga-tetangganya ke dalam antrian prioritas:

a. Simpul B (sudah dieksplorasi sebelumnya, biaya dari A ke E via B = 8).

b. Simpul F (biaya dari A ke F via B dan E = 12).

c. Antrian prioritas sekarang: D_{AC} (biaya = 9), D_{AB} (biaya = 11), F_{ABE} (biaya = 12).

5. Iterasi 5:

Keluarkan simpul D_{AC} dari antrian. Perluas simpul D_{AC} dan tambahkan tetangga-tetangganya ke dalam antrian prioritas:

a. Simpul C (sudah dieksplorasi sebelumnya, biaya dari A ke D via C = 9).

b. Simpul B (sudah dieksplorasi sebelumnya, biaya dari A ke D via B = 11).

c. Simpul F (biaya dari A ke F via C dan D = 17).

d. Antrian prioritas sekarang: D_{AB} (biaya = 11), F_{ABE} (biaya = 12), F_{ACD} (biaya = 17).

6. Iterasi 6:

Keluarkan simpul D_{AB} dari antrian. Perluas simpul D_{AB} dan tambahkan tetangga-tetangganya ke dalam antrian prioritas:

a. Simpul B (sudah dieksplorasi sebelumnya, biaya dari A ke D via B = 11).

b. Simpul F (biaya dari A ke F via B dan D = 19).

c. Antrian prioritas sekarang: F_{ABE} (biaya = 12), F_{ACD} (biaya = 17), F_{ABD} (biaya = 19).

7. Iterasi 7:

Keluarkan simpul F_{ABE} dari antrian. Karena simpul F_{ABE} adalah simpul tujuan, pencarian selesai. Jalur terpendek dari A ke F adalah: A -> B -> E -> F, dengan biaya total 8.

Dalam iterasi terakhir, simpul F berhasil dieksplorasi dan menjadi simpul tujuan, menandakan bahwa pencarian rute terpendek telah selesai. Jalur terpendek yang ditemukan adalah A -> B -> E -> F.

BAB IV
HASIL DAN PEMBAHASAN

4.1 Implementasi Algoritma *Uniform Cost Search* (UCS)

Pada tahap ini, pengujian dilaksanakan dengan cermat melalui penerapan algoritma *Uniform Cost Search* (UCS), yang diikuti oleh pemanfaatan data riil guna memberikan evaluasi yang akurat terhadap hasil yang dicapai. Penelitian ini memanfaatkan sebanyak 31 data kecamatan dan 52 data rute yang berlokasi di Kota Surabaya, mengambil sumber data dari platform Google Maps. Prosedur pengambilan data dilakukan dengan teliti untuk memastikan keakuratan dan kualitasnya sebelum digunakan dalam pengujian. Data riil ini menjadi pijakan yang kuat dalam mengukur kinerja algoritma UCS dalam konteks pengujian yang sesungguhnya. Melalui pendekatan ini, penelitian akan memberikan gambaran komprehensif mengenai kemampuan algoritma UCS dalam mengatasi tantangan rute kota nyata dan sejauh mana kesesuaian hasil implementasi dengan situasi lapangan yang ada. Berikut ini merupakan table data wilayah kota Surabaya.

Tabel 4. 1 Daftar wilayah yang diukur

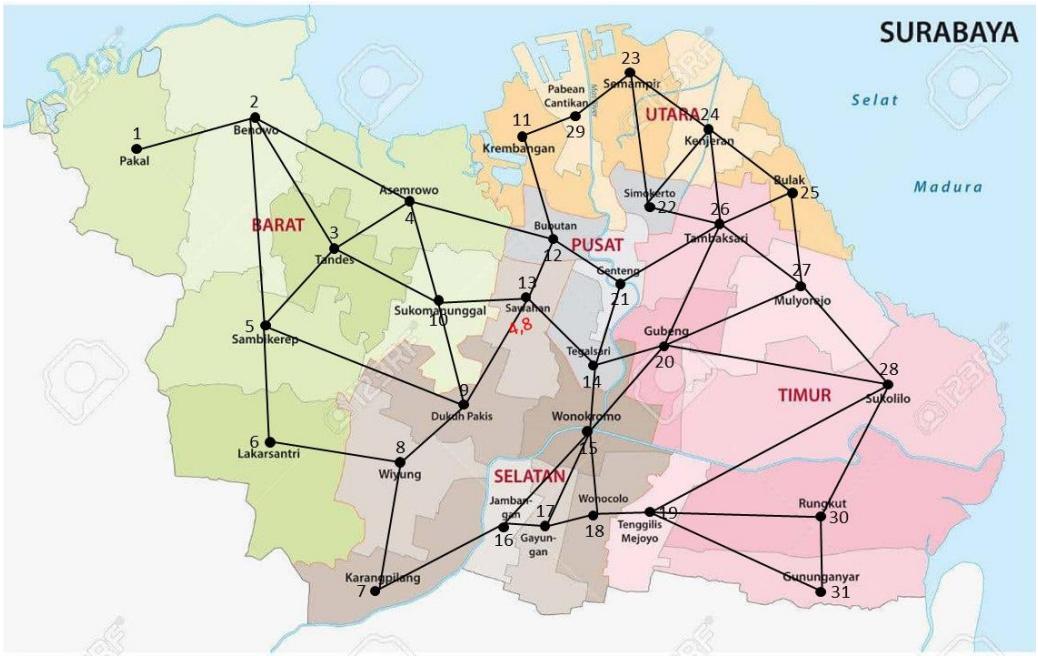
No.	Wilayah
1	Tegalsari
2	Simokerto
3	Genteng
4	Bubutan
5	Gubeng
6	Gunung Anyar
7	Sukolilo
8	Tambaksari
9	Mulyorejo
10	Rungkut
11	Tenggilis Mejoyo
12	Benowo
13	Pakal

No.	Wilayah
14	Asemrowo
15	Sukomanunggal
16	Tandes
17	Sambikerep
18	Lakarsantri
19	Bulak
20	Kenjeran
21	Semampir
22	Pabean Cantian
23	Krembangan
24	Wonokromo
25	Wonocolo
26	Wiyung
27	Karang Pilang
28	Jambangan
29	Gayungan
30	Dukuh Pakis
31	Sawahan

Proses penentuan simpul yang perlu dibuat melibatkan analisis jalur yang dapat diikuti dalam peta, dengan tujuan mencapai daerah yang dituju. Selain mengidentifikasi jalur, orientasi arah jalan juga menjadi aspek penting untuk memastikan bahwa kendaraan tidak mengambil jalur dengan arah yang tidak sesuai. Dari rangkaian jalur yang ada, langkah berikutnya adalah membuat simpul di tempat-tempat di mana jalan bercabang, biasanya terletak di perempatan atau tikungan jalan. Pemilihan lokasi simpul didasarkan pada pertimbangan apakah jalan pada simpul tersebut memiliki manfaat signifikan dalam mencapai tujuan akhir.

Proses penilaian skor untuk rute antar simpul dilakukan dengan menghitung jarak antara dua simpul menggunakan fitur pengukuran jarak dalam aplikasi Google Maps. Hasil dari pertimbangan dan perhitungan tersebut membentuk gambaran keseluruhan tentang rute terpendek.

Pada proses pendekatan di atas, tahap penentuan simpul dan perhitungan skor rute menjadi lebih terdefinisikan dan terukur. Keseluruhan langkah-langkah ini berkontribusi pada hasil akhir berupa gambaran visual yang merepresentasikan jalur terbaik berdasarkan analisis dan pertimbangan yang telah dilakukan sebelumnya. Dari penjelasan tersebut, maka dihasilkan gambar sebagai berikut:



Gambar 4.1 Peta Simpul dan Rute Kota Surabaya

Berikut ini merupakan tabel daftar data setiap jarak dari kecamatan 1 menuju kecamatan 2:

Tabel 4. 2 Daftar data setiap jarak

Kecamatan 1	Kecamatan 2	Jarak(m)
Pakal	Benowo	4300
Benowo	Asemrowo	7700
Benowo	Tandes	3400
Benowo	Sambikerep	4500
Asemrowo	Tandes	4300
Asemrowo	Sukomanunggal	2800
Asemrowo	Bubutan	4700
Tandes	Sambikerep	3400
Tandes	Sukomanunggal	3800

Kecamatan 1	Kecamatan 2	Jarak(m)
Sambikerep	Dukuh Pakis	4900
Sambikerep	Lakarsantri	5500
Lakarsantri	Wiyung	3000
Wiyung	Dukuh Pakis	5500
Wiyung	Karangpilang	5100
Karangpilang	Jambangan	6200
Sukomanunggal	Dukuh Pakis	2000
Sukomanunggal	Sawahan	3900
Sawahan	Dukuh Pakis	6900
Sawahan	Bubutan	3900
Sawahan	TegalSari	2300
Bubutan	Krembangan	2600
Bubutan	Genteng	5000
Krembangan	Pabean Cantikan	2700
Pabean Cantikan	Semampir	4400
Semampir	Simokerto	3400
Semampir	Kenjeran	2900
Kenjeran	Simokerto	4900
Kenjeran	Bulak	5000
Kenjeran	Tambaksari	4900
Bulak	Tambaksari	5400
Bulak	Mulyorejo	3900
Tambaksari	Mulyorejo	3400
Tambaksari	Gubeng	3600
Tambaksari	Genteng	3200
Simokerto	Tambaksari	2800
Mulyorejo	Gubeng	4800
Mulyorejo	Sukolilo	4300
Gubeng	Sukolilo	5000
Gubeng	Wonokromo	5000
Genteng	TegalSari	5000
Tegalsari	Wonokromo	4300
Wonokromo	Jambangan	6200
Wonokromo	Gayungan	7000
Wonokromo	Wonocolo	2300
Jambangan	Gayungan	1400
Gayungan	Wonocolo	5100
Wonocolo	Tenggilis Mejoyo	2700
Tenggilis Mejoyo	Sukolilo	5200
Tenggilis Mejoyo	Rungkut	5000

Kecamatan 1	Kecamatan 2	Jarak(m)
Tenggilis Mejoyo	Gunung Anyar	3900
Sukolilo	Rungkut	2600
Rungkut	Gunung Anyar	3300

Berikut ini merupakan kode implementasi algoritma *Uniform Cost Search* (UCS) dalam konteks pencarian jalur terpendek dalam graf berbobot. Algoritma UCS adalah salah satu metode yang efektif digunakan untuk menemukan jalur terpendek dalam graf berbobot, dengan mempertimbangkan biaya terkecil untuk mencapai setiap simpul. Kode dibawah ini adalah langkah awal dalam penelitian yang menggunakan Python untuk mengimpor modul csv yang digunakan untuk mengelola file CSV (Comma-Separated Values) dan modul PriorityQueue dari pustaka standar Python yang berguna untuk mengelola data.

```
import csv
from queue import PriorityQueue
```

Gambar 4.2 Source code import modul

Fungsi kode diatas adalah mengimpor dua modul yaitu modul csv untuk memanipulasi data dalam bentuk csv dan priority queue untuk mengimplementasikan antrian prioritas.

```
class KecamatanTidakDitemukan(Exception):
    def _init_(self, kecamatan):
        print("sesuaikan dengan data dari excel" % kecamatan)
```

Gambar 4.3 Source code cetak pesan

Fungsi kode pada gambar 4.3 adalah definisi dari class pengecualian yang diturunkan dari class exception, berfungsi untuk mencetak pesan jika data kecamatan tidak ditemukan.

```
def build_graph(path):

    file = open(path,'r')
    rute = {}
    next(file)
    for row in file:

        row = row.split(',')
        rute.setdefault(row[0], []).append((row[1],row[2]))
        rute.setdefault(row[1], []).append((row[0],row[2]))

    file.close()

    return rute
```

Gambar 4.4 Source code membangun graf antar node

Fungsi kode diatas untuk membaca data dari file dan membangun graf berdasarkan hubungan antar node yang terdapat dalam data tersebut. Graf digunakan untuk mewakili hubungan antara entitas dan bobotnya. Proses tersebut memungkinkan analisis jaringan dan masalah terkait graf dalam konteks ilmiah atau pemrograman.

```
def uniform_cost_search(graph, awal_kecamatan,
tujuan_kecamatan):

    mengunjungi = set()
    rute = []
    priority_queue = PriorityQueue()
    priority_queue.put((0, [awal_kecamatan]))

    while priority_queue:

        if priority_queue.empty():
            print ('jarak: tak terbatas \nrute: \nnone')
            break
```

```
        jarak, rute = priority_queue.get()
        kecamatan = rute[len(rute)-1]

        if kecamatan not in mengunjungi:
            mengunjungi.add(kecamatan)

            if kecamatan == tujuan_kecamatan:
                rute.append(jarak)
                display_rute(graph,rute)
                return rute

        childs = graph[kecamatan]
        tetangga=[i[0] for i in childs]

        for i in tetangga:
            if i not in mengunjungi:

                totaljarak = jarak +
int(kecamatan_terpendek(graph, kecamatan, i))
                temp = rute[:]
                temp.append(i)
                priority_queue.put((totaljarak, temp))

    return priority_queue
```

Gambar 4.5 Source code implementasi algoritma UCS

Fungsi uniform_cost_search yang diberikan di atas adalah implementasi praktis dari algoritma pencarian dengan biaya seragam (*Uniform Cost Search*/UCS). Tujuan utama algoritma UCS adalah menemukan rute terpendek antara dua kecamatan dalam sebuah graf. Fungsi ini memerlukan tiga parameter utama: graf (yang mewakili struktur data graf yang akan dicari), awal_kecamatan (kecamatan asal), dan tujuan_kecamatan (kecamatan tujuan).

Proses dimulai dengan inisialisasi berbagai variabel dan struktur data yang akan digunakan dalam pencarian. Ada tiga komponen utama yang digunakan untuk melacak informasi saat pencarian berlangsung:

1. Mengunjungi: Ini adalah himpunan (set) yang digunakan untuk mencatat kecamatan yang telah dikunjungi selama pencarian berlangsung. Hal ini penting untuk menghindari penjelajahan berulang ke kecamatan yang sama.
2. Rute: Variabel ini adalah daftar (list) yang akan menyimpan rute saat ini yang sedang dianalisis. Rute ini berubah seiring berjalannya pencarian.
3. Priority Queue: Digunakan untuk menyimpan rute-rute yang akan dianalisis selama pencarian. Antrian prioritas ini mengurutkan rute-rute berdasarkan bobot terkecil, sehingga rute dengan biaya terendah akan dieksplorasi terlebih dahulu.

Proses pencarian dimulai dengan menambahkan rute awal ke dalam antrian prioritas. Rute awal ini adalah rute yang dimulai dari kecamatan awal dan memiliki jarak awal 0 (karena dimulai dari dirinya sendiri).

Selanjutnya, algoritma UCS akan mengambil rute dengan biaya terendah dari antrian prioritas. Variabel "jarak" akan menyimpan biaya rute tersebut, "rute" akan berisi daftar kecamatan yang telah dikunjungi, dan "kecamatan" akan berisi kecamatan terakhir dalam rute. Kemudian, algoritma akan mendapatkan daftar kecamatan anak dari kecamatan saat ini. Daftar ini berisi pasangan kecamatan dan jaraknya yang terhubung dengan kecamatan saat ini. Algoritma akan mengeksplorasi kecamatan-kecamatan ini untuk mencari rute terpendek.

Seluruh proses ini berjalan hingga algoritma menemukan rute terpendek dari kecamatan awal ke kecamatan tujuan dalam graf yang diberikan. Dengan demikian, fungsi uniform_cost_search ini menjadi implementasi yang berguna untuk menyelesaikan masalah pencarian rute terpendek dalam konteks graf.

```
def kecamatan_terpendek(graph, current, tetangga):
    index = [i[0] for i in graph[current]].index(tetangga)
    return graph[current][index][1]
```

Gambar 4.6 Source code untuk mengambil jarak

Fungsi kecamatan_terpendek digunakan untuk mengambil jarak atau bobot dari current ke tetangga dalam graf yang diberikan. Ini bekerja dengan mencari tetangga dalam daftar koneksi dari current dan mengambil jaraknya.

```
def display_rute(graph,rute):
    length = len(rute)
    jarak = rute[-1]
    print()
    print('Jarak setiap kecamatan: %s m'%(jarak))
    print()
    print('Rute terpendek: ')
    count = 0
    while count < (length-2):
        m = kecamatan_terpendek(graph, rute[count],
rute[count+1])
        print('dari %s menuju %s dengan jarak %s meter'
%(rute[count],rute[count+1],m))
        count+=1
    return
```

Gambar 4.7 Source code menampilkan informasi rute

Fungsi display_rute digunakan untuk menampilkan informasi tentang rute terpendek dari graf. Ini mencetak jarak total dari rute dan informasi rute dari setiap kecamatan ke kecamatan berikutnya beserta jaraknya.

```
if _name_ == "_main_":

    while True:
        try:
            inputFile = input("Masukkan Excel: ")
            test = open(inputFile, 'r').readlines()
        except FileNotFoundError:
            print("Excel tidak ditemukan")
        else:
            break
```

```
    graph = build_graph(inputFile)

    while True:
        try:
            awal_kecamatan = input("Masukkan kecamatan awal: ")
            if awal_kecamatan not in graph:
                raise KecamatanTidakDitemukan(awal_kecamatan)
            break
        except KecamatanTidakDitemukan:
            print("Kecamatan tidak ada didalam peta")

    while True:
        try:
            tujuan_kecamatan = input("Masukkan kecamatan
tujuan: ")
            if tujuan_kecamatan not in graph:
                raise KecamatanTidakDitemukan(tujuan_kecamatan)
            break
        except KecamatanTidakDitemukan:
            print("Kecamatan tidak ada didalam peta")


    uniform_cost_search(graph, awal_kecamatan,
tujuan_kecamatan)

    pass
```

Gambar 4.8 Source code main

Fungsi kode di atas untuk mengambil input berupa nama file Excel, kecamatan awal, dan kecamatan tujuan. Selanjutnya, fungsi ini akan mencari rute terpendek antara dua kecamatan dalam graf yang dibangun berdasarkan data Excel. Jika data kecamatan tidak ditemukan dalam peta, maka fungsi akan menghasilkan output "kecamatan tidak ditemukan dalam peta."

4.2 Pembahasan

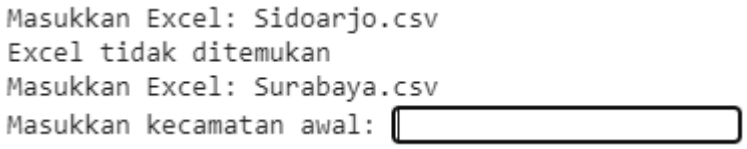
Proses pengujian suatu sistem melibatkan dua tahapan utama yaitu uji coba dan validasi. Tahap uji coba memperkenalkan sistem dilakukan dengan mengidentifikasi potensi masalah yang timbul. Ini mencakup skenario penggunaan

dan uji kasus untuk menilai sejauh mana kinerja optimalnya. Setelah uji coba, maka akan dilakukan tahap validasi sistem.

4.2.1 Uji Coba

a. Uji Coba Pertana

Pada tahap uji coba, langkah awal dilakukan dengan memasukkan sumber data yang diperlukan. Sumber data yang digunakan berasal dari *google maps* dan kemudian dimanipulasi melalui *excel*. Di dalam lembar kerja *excel* ini, disusun tabel yang memuat informasi mengenai jarak antar kecamatan yang bertetangga. Proses ini dilakukan untuk mendapatkan data yang relevan dan dibutuhkan dalam penelitian. Sebagai tindak lanjut, sistem akan memberikan pesan output yang menyatakan "*Excel* tidak ditemukan" jika perangkat lunak *excel* tidak terdeteksi. Namun, jika *excel* terdeteksi, penelitian akan melanjutkan ke langkah-langkah berikutnya.



Gambar 4.9 Memasukkan excel

Langkah kedua adalah data data dari *excel* akan dijadikan sebagai sebuah *graph* dengan memanggil fungsi build_graph pada gambar 4.4.

Langkah ketiga adalah dengan memasukkan kecamatan awal dan kecamatan tujuan. Jika nama kecamatan tidak ada, maka akan memanggil fungsi KecamatanTidakDitemukan pada gambar 4.3 dan akan mengeluarkan output “”

```
Masukkan kecamatan awal: Waru
kecamatan Waru tidak ada dalam excel
Tolong isi kembali kecamatan yang ada di excel
Masukkan kecamatan awal: Pakal
Masukkan kecamatan tujuan: Tandes
```

Gambar 4.10 Memasukkan nama kecamatan awal dan tujuan percobaan pertama

Langkah keempat memanggil fungsi uniform_cost_search untuk menentukan rute terpendek. Memasuki iterasi pertama, kecamatan awal yang telah ditentukan akan menjadi simpul awal yaitu kecamatan Pakal. Masuk ke perulangan priority_queue untuk mencari urutan manakah jarak yang terkecil. Karena kecamatan Pakal bukan tujuan, maka algoritma UCS akan melanjutkan kodinga selanjutnya. Algoritma UCS akan menghitung biaya jalur dari titik awal ke setiap tetangga dari kecamatan Pakal, yaitu hanya kecamatan Benowo. Biaya jalur dari kecamatan Pakal menuju kecamatan Benowo adalah 4.300 Meter. Algoritma UCS kemudian akan menambahkan kecamatan Benowo ke antrian.

Tabel 4. 3 Tabel Uji Coba Pertama Iterasi Pertama

Iterasi	Simpul	Simpul Hidup
1	Pakal	$Benowo_{pakal=4300}$

Pada iterasi kedua, algoritma UCS akan mengambil kecamatan Benowo dari antrian. Kecamatan Benowo memiliki biaya 4.300. Karena Kecamatan Benowo bukan tujuan, maka algoritma UCS akan melanjutkan ke kodingan selanjutnya. Algoritma UCS akan menghitung biaya jalur dari titik awal ke setiap tetangga kecamatan Benowo, yaitu Kecamatan Asemrowo, Kecamatan Tandes dan Kecamatan Sambikerep. Biaya jalur dari Kecamatan Benowo ke Kecamatan Asemrowo adalah 7.700 meter, biaya jalur dari Kecamatan Benowo ke Kecamatan Tandes 3400 Meter dan biaya jalur dari Kecamatan Benowo ke Kecamatan Sambikerep adalah 4500 meter. Karena biaya jalur dari Kecamatan Benowo ke

Kecamatan Tandes lebih rendah dari biaya tetangga lainnya, maka algoritma UCS akan memperbarui biaya tetangga Kecamatan Tandes menjadi 3400. Algoritma UCS kemudian akan menambahkan Kecamatan Tandes ke antrian.

Tabel 4. 4 Tabel Uji Coba Pertama Iterasi kedua

Iterasi	Simpul	Simpul Hidup
2	$Benowo_{pakal=4300}$	$Tandes_{pakal \rightarrow Benowo=7700}$ $Sambikerep_{pakal \rightarrow Benowo=8800}$ $Asemrowo_{pakal \rightarrow asemrowo=12000}$

Pada iterasi ketiga, algoritma UCS akan mengambil Kecamatan Tandes dari antrian. Kecamatan Tandes memiliki biaya 3400. Karena Kecamatan Tandes adalah tujuan, maka algoritma UCS akan mengembalikan Kecamatan Tandes. Jadi, jalur terpendek dari Kecamatan Pakal ke Kecamatan Tandes adalah Pakal-Benowo-Tandes, dengan total biaya 7.700 Meter.

Tabel 4. 5 Tabel Uji Coba Pertama Iterasi ketiga

Iterasi	Simpul	Simpul Hidup
3	$Tandes_{pakal \rightarrow Benowo=7700}$	*Solusi Ditemukan*

b. Uji Coba Kedua

Langkah pengujian seperti pada uji coba pertama. Namun, untuk pengujian kedua kecamatan awal adalah Kecamatan Rungkut dan kecamatan tujuan adalah Kecamatan Gubeng.

```
Masukkan kecamatan awal: Rungkut
Masukkan kecamatan tujuan: Gubeng
```

Gambar 4.11 Memasukkan nama kecamatan awal dan tujuan percobaan kedua

Langkah selanjutnya sama seperti sebelumnya yaitu memanggil fungsi uniform_cost_search untuk menentukan rute terpendek.

Memasuki iterasi pertama, kecamatan awal yang telah ditentukan akan menjadi simpul awal yaitu kecamatan Rungkut. Masuk ke perulangan priority_queue untuk mencari urutan manakah jarak yang terkecil. Karena kecamatan Rungkut bukan termasuk kecamatan tujuan maka perulangan akan dilanjutkan. Algoritma UCS akan menghitung biaya jalur dari titik awal ke setiap tetangga dari kecamatan Rungkut, yaitu kecamatan Sukolilo, kecamatan Tenggilis Mejoyo, dan kecamatan Gununganyar. Biaya jalur dari kecamatan Rungkut menuju kecamatan Sukolilo adalah 2.600 Meter, biaya jalur dari kecamatan Rungkut menuju Tenggilis Mejoyo adalah 5.000 Meter dan biaya jalur dari kecamatan Rungkut menuju kecamatan Gununganyar adalah 3.300 Meter. Algoritma UCS kemudian akan menambahkan kecamatan Sukolilo ke antrian.

Tabel 4. 6 Tabel Uji Coba Kedua Iterasi Pertama

Iterasi	Simpul	Simpul Hidup
1	Rungkut	$Sukolilo_{rungkut=2600}$ $Gununganyar_{rungkut=3300}$ $Tenggilis\ Mejoyo_{rungkut=5000}$

Pada iterasi kedua, algoritma UCS akan mengambil kecamatan Sukolilo dari antrian. Kecamatan Sukolilo memiliki biaya 2.600 Meter. Karena Kecamatan Sukolilo bukan tujuan, maka algoritma UCS akan melanjutkan ke kodingan selanjutnya. Algoritma UCS akan menghitung biaya jalur dari titik awal ke setiap tetangga kecamatan Sukolilo, yaitu Kecamatan Mulyorejo dan Kecamatan Gubeng. Biaya jalur dari Kecamatan Sukolilo ke Kecamatan Mulyorejo adalah 4.300 meter, dan biaya jalur dari Kecamatan Sukolilo ke Kecamatan Gubeng 5.000 Meter. Algoritma UCS akan memperbarui biaya tetangga – tetangga Kecamatan Sukolilo yaitu kecamatan Mulyorejo menjadi 6.900 Meter dan kecamatan Gubeng menjadi

7.6 Meter. Algoritma UCS kemudian akan menambahkan Kecamatan Gubeng dan kecamatan Mulyorejo ke antrian.

Tabel 4. 7 Tabel Uji Coba Kedua Iterasi Kedua

Iteriasi	Simpul	Simpul Hidup
2	$Sukolilo_{rungkut=2600}$	$Gununganyar_{rungkut=3300}$ $Tenggilis\ Mejoyo_{rungkut=5000}$ $Mulyorejo_{rungkut\rightarrow sukolilo=6900}$ $Gubeng_{rungkut\rightarrow sukolilo=7600}$

Pada iterasi ketiga, algoritma UCS akan mengambil Kecamatan Gununganyar dari antrian. Kecamatan Tandes memiliki biaya 3.300 Meter. Karena Kecamatan Gunnganyar bukan tujuan, maka algoritma UCS akan melanjutkan ke kodingan selanjutnya. Algoritma UCS akan menghitung biaya jalur dari titik awal ke setiap tetangga kecamatan Gununganyar, yaitu Kecamatan Mulyorejo dan Kecamatan Gubeng. Karena tetangga dari kecamatan Gununganyar sudah diketahui maka nilai akan ditambahkan di antrian dari kecamatan Rungkut menuju kecamatan Tenggilis Mejoyo via kecamatan Gununganyar 7.200 Meter.

Tabel 4. 8 Tabel Uji Coba Kedua Iterasi Ketiga

Iterasi	Simpul	Simpul Hidup
3	$Gununganyar_{rungkut=3300}$	$Tenggilis\ Mejoyo_{rungkut=5000}$ $Mulyorejo_{rungkut\rightarrow sukolilo=6900}$ $Tenggilis\ Mejoyo_{rungkut\rightarrow GA=7200}$ $Gubeng_{rungkut\rightarrow sukolilo=7600}$

Pada iterasi ke 4, algoritma UCS akan mengambil kecamatan Tenggilis Mejoyo dari antrian. Kecamatan Tenggilis Mejoyo memiliki biaya 3.900 Meter. Karena Kecamatan Tenggilis Mejoyo bukan tujuan, maka algoritma UCS akan melanjutkan ke kodingan selanjutnya. Algoritma UCS akan menghitung biaya jalur dari titik awal ke setiap tetangga kecamatan Tenggilis Mejoyo, yaitu Kecamatan

Wonocolo. Biaya jalur dari Kecamatan Tenggilis Mejoyo ke Kecamatan Wonocolo adalah 2.700 meter, dan biaya jalur dari Algoritma UCS akan memperbarui biaya tetangga – tetangga Kecamatan Tenggilis Mejoyo yaitu kecamatan Wonocolo menjadi 6.900 Meter. Algoritma UCS kemudian akan menambahkan Kecamatan Wonocolo ke antrian.

Tabel 4. 9 Tabel Uji Coba Kedua Iterasi Keempat

Iterasi	Simpul	Simpul Hidup
4	$Tenggilis\ Mejoyo_{rungkut=5000}$	$Mulyorejo_{rungkut \rightarrow sukolilo=6900}$ $Tenggilis\ Mejoyo_{rungkut \rightarrow GA=7200}$ $Gubeng_{rungkut \rightarrow sukolilo=7600}$ $Wonocolo_{rungkut \rightarrow TM=7700}$

Pada interasi ke lima, algoritma UCS akan mengambil kecamatan Mulyorejo dari antrian. Kecamatan Mulyorejo memiliki biaya 4.300 Meter. Karena Kecamatan Mulyorejo bukan tujuan, maka algoritma UCS akan melanjutkan ke kodingan selanjutnya. Algoritma UCS akan menghitung biaya jalur dari titik awal ke setiap tetangga kecamatan Sukolilo, yaitu Kecamatan Mulyorejo dan Kecamatan Gubeng. Biaya jalur dari Kecamatan Sukolilo ke Kecamatan Mulyorejo adalah 4.300 meter, dan biaya jalur dari Kecamatan Sukolilo ke Kecamatan Gubeng 5.000 Meter. Algoritma UCS akan memperbarui biaya tetangga – tetangga Kecamatan Sukolilo yaitu kecamatan Mulyorejo menjadi 6.900 Meter dan kecamatan Gubeng menjadi 7.6 Meter. Algoritma UCS kemudian akan menambahkan Kecamatan Gubeng dan kecamatan Mulyorejo ke antrian.

Tabel 4. 10 Tabel Uji Coba Kedua Iterasi Kelima

Iterasi	Simpul	Simpul Hidup
5	$Mulyorejo_{rungkut \rightarrow sukolilo=690}$	$Gubeng_{rungkut \rightarrow sukolilo=7600}$

		$Tenggilis\ Mejoyo_{rungkut \rightarrow GA=7200}$ $Wonocolo_{rungkut \rightarrow TM=7700}$ $Bulak_{rungkut \rightarrow sukolilo \rightarrow MR=10800}$ $Tambaksari_{rungkut \rightarrow sukolilo \rightarrow MR=1030}$ $Gubeng_{rungkut \rightarrow sukolilo \rightarrow MR=11700}$

Pada iterasi ke enam, algoritma UCS akan mengambil kecamatan Tenggilis Mejoyo dari antrian. Kecamatan Sukolilo memiliki biaya 2.600 Meter. Karena Kecamatan Sukolilo bukan tujuan, maka algoritma UCS akan melanjutkan ke kodingan selanjutnya. Algoritma UCS akan menghitung biaya jalur dari titik awal ke setiap tetangga kecamatan Sukolilo, yaitu Kecamatan Mulyorejo dan Kecamatan Gubeng. Biaya jalur dari Kecamatan Sukolilo ke Kecamatan Mulyorejo adalah 4.300 meter, dan biaya jalur dari Kecamatan Sukolilo ke Kecamatan Gubeng 5.000 Meter. Algoritma UCS akan memperbarui biaya tetangga – tetangga Kecamatan Sukolilo yaitu kecamatan Mulyorejo menjadi 6.900 Meter dan kecamatan Gubeng menjadi 7.6 Meter. Algoritma UCS kemudian akan menambahkan Kecamatan Gubeng dan kecamatan Mulyorejo ke antrian.

Tabel 4. 11 Tabel Uji Coba Kedua Iterasi Keenam

Iterasi	Simpul	Simpul Hidup
6	$Tenggilis\ Mejoyo_{rungkut \rightarrow GA=}$	$Gubeng_{rungkut \rightarrow sukolilo=7600}$ $Wonocolo_{rungkut \rightarrow TM=7700}$ $Bulak_{rungkut \rightarrow sukolilo \rightarrow MR=10800}$ $Tambaksari_{rungkut \rightarrow sukolilo \rightarrow MR=1030}$ $Gubeng_{rungkut \rightarrow sukolilo \rightarrow MR=11700}$

Pada iterasi ke tujuh, algoritma UCS akan mengambil Kecamatan Gubeng dari antrian. Kecamatan Gubeng memiliki biaya 7.600 Meter. Karena Kecamatan Gubeng adalah tujuan, maka algoritma UCS akan mengembalikan Kecamatan

Gubeng. Jadi, jalur terpendek dari Kecamatan Rungkut ke Kecamatan Gubeng adalah Kecamatan Rungkut → Kecamatan Sukolilo → Kecamatan Gubeng, dengan total biaya 7.600 Meter.

Tabel 4. 12 Tabel Uji Coba Kedua Iterasi Ketujuh

Iterasi	Simpul	Simpul Hidup
7	$Gubeng_{rungkut \rightarrow sukolilo=7600}$	Solusi telah ditemukan

4.2.2 Validasi

Pada tahap uji coba menghasilkan output seperti gambar 4.12, kemudian hasil uji coba akan dilakukan proses validasi untuk mengetahui apakah proses sudah berhasil atau belum, berikut ini merupakan tahapannya:

Tabel 4. 13 Tabel Uji Coba Pertama

Iterasi	Simpul	Simpul Hidup
1	Pakal	$Benowo_{pakal=4300}$
2	$Benowo_{pakal=4300}$	$Tandes_{pakal \rightarrow Benowo=7700}$ $Sambikerep_{pakal \rightarrow Benowo} = 8800$ $Asemrowo_{pakal \rightarrow asemrowo} = 12000$
3	$Tandes_{pakal \rightarrow Benowo=7700}$	$Solusi\ Ditemukan$

Pada tabel 4. Menunjukkan bahwa uji coba pertama memiliki 3 iterasi untuk mencari rute terpendek dari kecamatan awal, yaitu kecamatan Pakal menuju ke kecamatan tujuan, yaitu kecamatan Tandes.

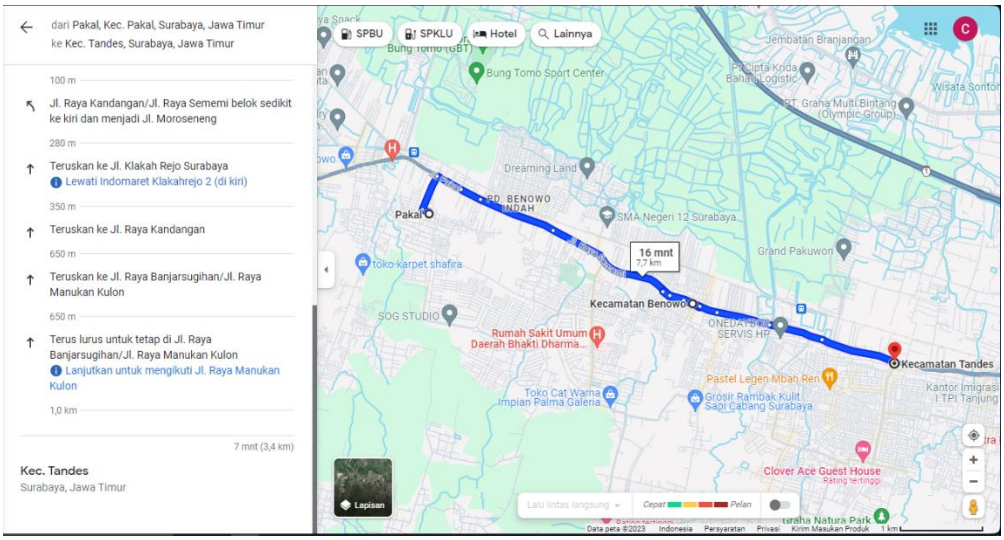
```
Kecamatan Saat Ini adalah Tandes

Jarak total kecamatan: 7700 m

Rute terdekat:
dari Pakal menuju Benowo dengan jarak 4300
 meter
dari Benowo menuju Tandes dengan jarak 3400
 meter
```

Gambar 4.12 Hasil dari uji coba

Hasil pada gambar 4.12 akan divalidasikan apakah uji coba tersebut berhasil atau tidak. hasil tersebut akan divalidasikan dengan menggunakan *google maps*.



Gambar 4.13 Validasi menggunakan google maps

Berikut ini merupakan rincian perjalanan dari Kecamatan Pakal, Surabaya, Jawa Timur yang di jelaskan menggunakan *Google Maps*:

a. Ambil Jl. Pakal AMD ke Jl. Pakal 2 mnt (550 m), kemudian ambil Jl. Raya Babat Jerawat dan Jl. Raya Sememi ke Jl. Klakah Rejo Surabaya di Sememi 8 mnt (3,8 km) total memakan biaya 10 mnt (4,3 km) Kec. Benowo, Surabaya, Jawa Timur
b. Ambil arah barat laut di Jl. Klakah Rejo Surabaya menuju Jl. Bandarejo Sawah I sepanjang 110 m, kemudian teruskan ke Jl. Moroseneng/Jl. Raya Sememi, lalu lanjutkan untuk mengikuti Jl. Raya Sememi sepanjang 280 m, lalu putar balik sepanjang 100 m Jl. Raya Kandangan/Jl. Raya Sememi belok sedikit ke kiri dan menjadi Jl. Moroseneng sepanjang 280 m, lalu teruskan ke Jl. Klakah Rejo Surabaya Lewati Indomaret Klakahrejo 2 (di

kiri) sepanjang 350 m, lalu teruskan ke Jl. Raya Kandangan sepanjang 650 m, lalu teruskan ke Jl. Raya Banjarsugihan/Jl. Raya Manukan Kulon sepanjang 650 m lalu terus lurus untuk tetap di Jl. Raya Banjarsugihan/Jl. Raya Manukan Kulon. Lanjutkan untuk mengikuti Jl. Raya Manukan Kulon sepanjang 1,0 km. dan total keseluruhannya 7 mnt (3,4 km). Lalu, sampailah ke Kec. Tandes, Surabaya, Jawa Timur.

c. Dari penjelasan tersebut maka dapat disimpulkan dari hasil output program dari peneliti dan google maps sama, yaiut rute Kecamatan Pakal → Kecamatan Benowo → Kecamatan Tandes dengan total jarak 7.700 meter.

Pada pegujian kedua, kecamatan awal adalah Rungkut dan kecamatan tujuan adalah Gubeng.

Mengetahui nilai google maps tersebut, nilai dari hasil eksperimen bisa ditentukan dengan menggunakan selisih kesalahan dalam bentuk persen atau biasa disebut *percentage error*. Rumus dari *percentage error* adalah:

$$\%\ error = \frac{|nilai\ perkiraan - nilai\ asli|}{nilai\ asli} \text{x}100\%$$

$$\%\ error = \frac{|jarak\ google\ maps - jarak\ ujicoba|}{jarak\ ujicoba} \text{x}100\%$$

$$\%\ error = \frac{|7700 - 7700|}{7700} \text{x}100\%$$

$$\%\ error = \frac{0}{7700} \text{x}100\%$$

$$\%\ error = 0\%$$

Hasil yang didapat dari *percentage error* adalah 0%. Nilai tersebut menjelaskan bahwa nilai perkiraan dan nilai asli sama.

Tabel 4. 14 Tabel Uji Coba Kedua

I	Simpul	Simpul Hidup
1	Rungkut	$Sukolilo_{rungkut=2600}$ $Gununganyar_{rungkut=3300}$ $Tenggilis\ Mejoyo_{rungkut=5000}$
2	$Sukolilo_{rungkut=2600}$	$Gununganyar_{rungkut=3300}$ $Tenggilis\ Mejoyo_{rungkut=5000}$ $Mulyorejo_{rungkut\rightarrow sukolilo=6900}$ $Gubeng_{rungkut\rightarrow sukolilo=7600}$
3	$Gununganyar_{rungkut=3300}$	$Tenggilis\ Mejoyo_{rungkut=5000}$ $Mulyorejo_{rungkut\rightarrow sukolilo=6900}$ $Tenggilis\ Mejoyo_{rungkut\rightarrow GA=7200}$ $Gubeng_{rungkut\rightarrow sukolilo=7600}$
4	$Tenggilis\ Mejoyo_{rungkut=5000}$	$Mulyorejo_{rungkut\rightarrow sukolilo=6900}$ $Tenggilis\ Mejoyo_{rungkut\rightarrow GA=7200}$ $Gubeng_{rungkut\rightarrow sukolilo=7600}$ $Wonocolo_{rungkut\rightarrow TM=7700}$
5	$Mulyorejo_{rungkut\rightarrow sukolilo=6900}$	$Gubeng_{rungkut\rightarrow sukolilo=7600}$ $Tenggilis\ Mejoyo_{rungkut\rightarrow GA=7200}$ $Wonocolo_{rungkut\rightarrow TM=7700}$ $Bulak_{rungkut\rightarrow sukolilo\rightarrow MR=10800}$ $Tambaksari_{rungkut\rightarrow sukolilo\rightarrow MR=10300}$ $Gubeng_{rungkut\rightarrow sukolilo\rightarrow MR=11700}$
6	$Tenggilis\ Mejoyo_{rungkut\rightarrow GA=}$	$Gubeng_{rungkut\rightarrow sukolilo=7600}$ $Wonocolo_{rungkut\rightarrow TM=7700}$ $Bulak_{rungkut\rightarrow sukolilo\rightarrow MR=10800}$ $Tambaksari_{rungkut\rightarrow sukolilo\rightarrow MR=10300}$ $Gubeng_{rungkut\rightarrow sukolilo\rightarrow MR=11700}$
7	$Gubeng_{rungkut\rightarrow sukolilo=7600}$	Solusi telah ditemukan

Berikut adalah Hasil pengujian tahap kedua:

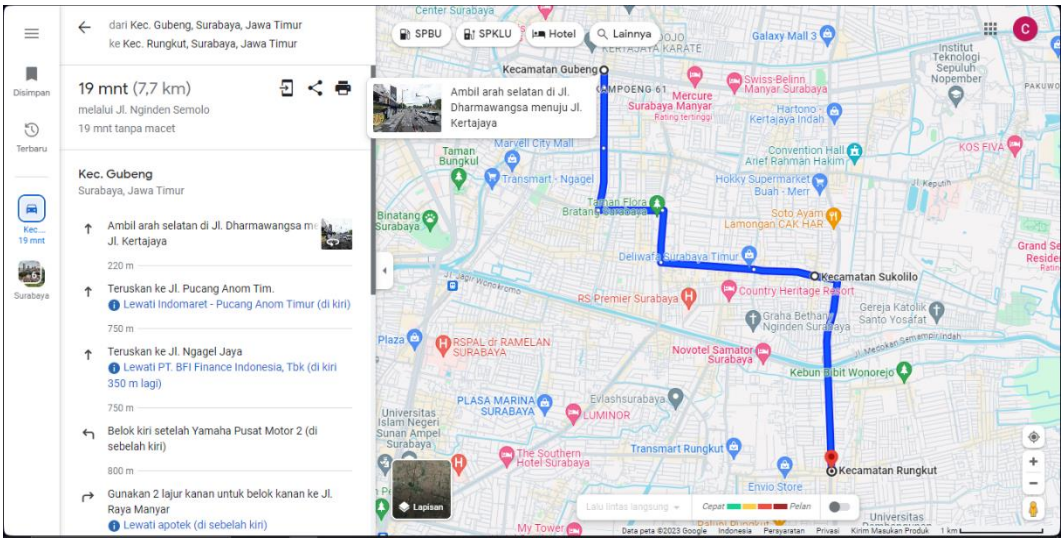
```
Kecamatan Saat Ini adalah Gubeng

Jarak total kecamatan: 7600 m

Rute terdekat:
dari Rungkut menuju Sukolilo dengan jarak 2600
 meter
dari Sukolilo menuju Gubeng dengan jarak 5000
 meter
```

Gambar 4.13 Hasil dari uji coba

Hasil pada gambar 4.13 akan divalidasikan dengan menggunakan google maps.



Gambar 4.14 Validasi menggunakan google maps

Validasi dari pengujian kedua adalah hasil dari aplikasi dengan rute Kecamatan Rungkut → Kecamatan Sukolilo → Kecamatan Gubeng dengan jarak 7.600 Meter sama dengan hasil yang ada di google maps dengan rute yang sama dan jarak 7.600 Meter.

Mengetahui nilai google maps tersebut, nilai dari hasil eksperimen bisa ditentukan dengan menggunakan selisih kesalahan dalam bentuk persen atau biasa disebut *percentage error*. Rumus dari *percentage error* adalah:

$$\%\ error = \frac{|nilai\ perkiraan - nilai\ asli|}{nilai\ asli} \text{x}100\%$$

$$\%\ error = \frac{|jarak\ google\ maps - jarak\ ujicoba|}{jarak\ ujicoba} \text{x}100\%$$

$$\%\ error = \frac{|7700 - 7600|}{7600} \text{x}100\%$$

$$\%\ error = \frac{100}{7700} \text{x} 100\%$$

% error = 1,32

Hasil yang didapat dari *percentage error* adalah 1%.

4.3 Kajian Integrasi Islam dan Sains

Dalam Al-Quran telah menjelaskan tentang pentingnya tolong menolong dalam semua hal kepada sesama seperti mencari rute terdekat untuk memudahkan dalam perjalanan dan meminimumkan biaya yang dikeluarkan, permasalahan tersebut dijelaskan dalam surat Al-Maidah ayat 02 yang berbunyi:

وَتَعَاوَنُوْا عَلَى الْبِرِّ وَالتَّقْوٰىۖ وَلَا تَعَاوَنُوْا عَلَى الْاِثْمِ وَالْعُدْوَانِۖ وَاتَّقُوا اللّٰهَۗ اِنَّ اللّٰهَ شَدِيْدُ الْعِقَابِ

"Dan tolong-menolonglah kamu dalam mengerjakan kebajikan dan takwa, dan jangan tolong-menolong dalam perbuatan dosa dan permusuhan. Bertakwalah kepada Allah, sesungguhnya Allah sangat berat siksaan-Nya."(QS. Al-Maidah:02).

Tafsir Al-Jalalain menjelaskan bahwa pada surat Al-Maidah Ayat 2, Allah menyuruh umat-Nya untuk saling tolong-menolong dalam kebaikan, ketaatan, dan ketakwaan. Mereka diingatkan agar tidak bersama-sama dalam perbuatan dosa dan permusuhan. Prinsip ini menekankan pentingnya kolaborasi dalam memperjuangkan kebaikan dan menghindari kerjasama dalam tindakan yang bertentangan dengan nilai-nilai agama dan norma moral. Pada bagian "Bertakwalah kepada Allah, sesungguhnya Allah sangat berat siksaan-Nya" menjelaskan bahwa sikap hati yang penuh kesadaran dan ketaatan kepada perintah-perintah Allah serta menjauhi larangan-Nya. Peringatan tentang beratnya siksaan Allah bertujuan untuk memberikan insentif kepada umat-Nya agar selalu menjaga ketakwaan dan menghindari perbuatan dosa.

Ayat 02 surat Al-Maidah dalam Al-Quran menekankan etika muamalah dalam islam. Menitikberatkan pada prinsip-prinsip etika dan moral yang mendasari interaksi sosial. Umat Islam diajak untuk saling tolong-menolong dalam mengerjakan kebajikan dan memelihara takwa, menggarisbawahi pentingnya kebaikan dan kesalehan sebagai dasar muamalah. Larangan untuk tidak terlibat dalam perbuatan dosa dan permusuhan memandu umat agar menjauhi perilaku negatif dan merawat keharmonisan hubungan antarindividu. Puncaknya, muamalah Islam diberkahi dengan nasihat untuk bertakwa kepada Allah, mengingatkan setiap tindakan dan interaksi dalam kerangka ketaatan kepada-Nya. Kesadaran akan beratnya siksaan Allah memberikan dimensi moral yang kuat, mendorong individu untuk membangun muamalah yang adil, penuh kasih, dan berlandaskan pada nilai-nilai keislaman. Dengan demikian, ayat-ayat ini membentuk landasan etika sosial dalam tolong menolong sesame umat manusia. Penelitian ini selain bertujuan mencari rute terpendek juga berusaha untuk meminumkan biaya yang dikeluarkan hal tersebut bisa untuk tolong menolong kepada sesama.

BAB V
KESIMPULAN DAN SARAN

5.1 Kesimpulan

Berdasarkan hasil penelitian mengenai implementasi algoritma *uniform cost search* (UCS) untuk pencarian rute terpendek di Kota Surabaya, dapat ditarik kesimpulan bahwa implementasi algoritma *uniform cost search* pada pencarian rute terpendek di Kota Surabaya dapat meminimalisir jarak. Implementasi algoritma ini bisa mengatasi permasalahan pada pencarian rute yang awalnya memakan waktu lama dan biaya operasional tinggi menjadi bisa memakan waktu yang lebih cepat dan biaya operasional rendah. Hasil dari pencarian rute terpendek sangat efisien mengurangi biaya operasional dan bisa lebih cepat sampai ke tujuan dan Hasil dari *percentage error* bisa disimpulkan bahwa data yang memiliki tingkat kesalahan sebesar 0% dan 1%.

5.2 Saran

Berdasarkan hasil pengujian yang diperoleh, penulis memahami bahwa masih diperlukan beberapa perbaikan dalam penelitian ini untuk meningkatkan kinerja sistem. Oleh karena itu, penulis menyarankan implementasi algoritma *uniform cost search* (UCS) lebih kompleks untuk pencarian rute terpendek.

DAFTAR PUSTAKA

Abidin, H. Z. (2022). *Peran Geodesi dan Geomatika Dalam Pembangunan Nasional (Aspek Data dan Informasi Geospasial)*. *2*(February), 1–44. https://doi.org/10.13140/RG.2.2.26194.02246

Asmara, R. D. Z., Yusuf, R., & Muharni, S. (2021). Penerapan Algoritma Uniform Cost Search (Ucs) Untuk Rekomendasi Pembangunan Rumah. *Journal Computer Science and Informatic Systems : J-Cosys*, *1*(1), 32–29. https://doi.org/10.53514/jc.v1i1.37

Endrayanto, R. K., & Muttaqin, A. (2021). Penerapan Machine Learning Berbasis Data Geospasial Untuk Optimalisasi Lahan Pertanian Pada Masa Pandemi Dan Pasca Pandemi. *Seminar Nasional Geomatika*, 161. https://doi.org/10.24895/sng.2020.0-0.1131

Faisal. (2017). Aplikasi Hasil Pencarian Dan Rute Pengiriman Barang Dari Solusi Masalah Transportasi Bikriteria Dengan Metode Logika Fuzzy. *Jurnal Instek*, *2*(2), 150–158.

Fawzi, N. I., & Husna, V. N. (2021). Pemanfaatan Informasi Geospasial Untuk Ketahanan Pangan Saat Pandemi Covid-19. *Seminar Nasional Geomatika*, *April*, 1. https://doi.org/10.24895/sng.2020.0-0.1114

Galih, B., & Putra, M. (2021). *Penerapan Algoritma UCS untuk Pencarian Rute Terpendek Pengiriman Surat di Kota Probolinggo*.

H. Cormen, T., E. Leiserson, C., L. Rivest, R., & Stein, C. (2010). Introduction to Algorithm, Second Edition. In *The Cambridge Companion to: The Roman Republic, Second Edition* (Vol. 7). https://doi.org/10.1017/CCO9781139424783.002

Kennedy, R. E., Yang, Z., Gorelick, N., Braaten, J., Cavalcante, L., Cohen, W. B., & Healey, S. (2018). Implementation of the LandTrendr algorithm on Google Earth Engine. *Remote Sensing*, *10*(5), 2019–2021. https://doi.org/10.3390/rs10050691

Kurniawan, W. M. (2022). *Penerapan Algoritma UCS Untuk Menentukan Rute Terbaik Menuju ITB*.

Nafiah, A. N. A. F. (2020). Perancangan Aplikasi Pencarian Rute Terdekat Jasa Binatu Online Berbasis Android Dengan Menggunakan Metode Dijkstra. *Ubiquitous: Computers and Its Applications Journal*, *3*, 99–106. https://doi.org/10.51804/ucaiaj.v3i2.99-106

Nggufron, N., & Mashuri, R. (2019). *Pencarian Ruter Terbaik Pemadam*

Kebakaran Kota Semarang Menggunakan Algoritma Dijkstra Dengan Logika Fuzzy Sebagai Penentu Bobot Pada Graf. *8*(1), 40–49.

Nugroho, T. P., Rohadi, E., & Prasetyo, A. (2017). Aplikasi Pencarian Jalur Terpendek Untuk Menemukan Lokasi Atm Di Kota Malang. *Jurnal Informatika Polinema*, *3*(4), 43. https://doi.org/10.33795/jip.v3i4.42

Pahleviannur, M. R. (2019). Pemanfaatan Informasi Geospasial Melalui Interpretasi Citra Digital Penginderaan Jauh untuk Monitoring Perubahan Penggunaan Lahan. *JPIG (Jurnal Pendidikan Dan Ilmu Geografi)*, *4*(2), 18–26. https://doi.org/10.21067/jpig.v4i2.3267

Rahmadi, D., & Sandariria, H. (2023). Penerapan Minimum Spanning Tree dalam Menentukan Rute Terpendek Distribusi Naskah Soal USBN di SMA Negeri se- Sleman. *Basis*, *2*(1), 66–71.

Ramadhan, M. F. (2022). *Penerapan Uniform Cost Search dalam Menentukan Rute Tercepat Memasuki Hard Mode pada Permainan Terraria*.

Sabilla, A. D., & Taufiq, A. (2022). *Journal of Information System and Computer PENERAPAN ALGORITMA A* PADA WEBGIS PENCARIAN RUTE TERPENDEK*. *2*(2), 32–35. https://journal.unisnu.ac.id/JISTER/

Saputra, R. (2011). *Sistem Informasi Geografis Pencarian Rute Optimum Obyek Wisata Kota Yogyakarta Dengan Algoritma Floyd-Warshall*.

Siringoringo, A. (2016). *Rancang Bangun Aplikasi Monitoring Kurir Berbasis Android*. Institut Teknologi Sepuluh Nopember.

Sun, Y., Wang, J., He, L., & Song, J. (2017). Iterative uniform-cost search of active antenna group selection for generalised spatial modulation. *IEEE International Conference on Communications*, 1–6. https://doi.org/10.1109/ICC.2017.7997093

Wu, Q. (2021). A Python package for interactive mapping and geospatial analysis with minimal coding in a Jupyter environment. *Journal of Open Source Software*, *6*(63), 3414. https://doi.org/10.21105/joss.03414