SKRIPSI

PENDEKATAN *CELLULAR AUTOMATA* DALAM OPTIMALISASI ALGORITMA DIJKSTRA UNTUK PENCARIAN JALUR TERPENDEK

CELLULAR AUTOMATA APPROACH IN OPTIMIZING DIJKSTRA ALGORITHM FOR SHORTEST PATH SEARCH

Diajuhkan untuk memenuhi sebagian persyaratan

memperoleh gelar Sarjana Komputer

NURFADHILAH ACO

D0221117

PROGRAM STUDI INFORMATIKA

FAKULTAS TEKNIK

UNIVERSITAS SULAWESI BARAT

MAJENE

2025

LEMBAR PENGESAHAN

PENDEKATAN *CELLULAR AUTOMATA* DALAM OPTIMALISASI ALGORITMA DIJKSTRA UNTUK PENCARIAN JALUR TERPENDEK

SKRIPSI

Untuk memenuhi sebagian persyaratan
Memproleh gelar Sarjana Komputer

Nurfadhilah Aco
NIM:D0221117

Skripsi ini diuji dan dinyatakan lulus
Pada 08/05/2025
Telah diperiksa dan disetujui oleh:

Pembimbing I	Pembimbing II
Dr. Eng. Sulfayanti, S.Si.,M.T	**Wawan Firgiawan,S.T.M.Kom**
NIP:198903172020122011	NIDK:8948080023

Dekan Fakultas Teknik, Universitas Sulawesi Barat	Ketua Program Studi Informatika,
Dr. Ir. Hafsah Nirwana, M.T.	**Muh.Rafli Rasyid,Skom.M.T**
NIP: 196404051990032002	NIP. 198808182022031006

LEMBAR PERSETUJUAN

SKRIPSI

PENDEKATAN *CELLULAR AUTOMATA* DALAM OPTIMALISASI *ALGORITMA DIJKSTRA* UNTUK PENCARIAN JALUR TERPENDEK

Telah disiapkan dan disusun oleh

NURFADHILAH ACO
D0221117

Telah dipertahankan di depan tim penguji
Pada tanggal 08 Mei 2025

Susunan Tim Penguji

Pembimbing I

Dr. Eng.Sulfayanti, S.Si., M.T
NIP: 198903172020122011

Pembimbing II

Wawan Firgiawan, S.T,. M.Kom
NIDK:89480880023

Penguji I

Arnita Irianti,S.Si., M.Si
NIP: 198708062018032001

Penguji II

Farid Wajidi,S.Kom.,MT
NIP: 198904182019031018

Penguji III

A.Amirul Asnan Cirua,S.T.,M.Kom
NIP:199304022024061001

PERNYATAAN ORISINALITAS

Saya menyatakan dengan sebenar benarnya bahwa sepanjang pengetahuan saya, di dalam naskah skripsi ini tidak terdapat karya ilmiah yang pernah oleh orang lain untuk memperoleh gelar akademik di suatu perguruan tinggi, dan tidak terdapat karya atau pendapat yang pernah ditulis atau diterbitkan oleh orang lain kecuali yang secara tertulis di sitasi dalam naska ini dan disebutkan dalam daftar referensi.

Apabila ternyata dalam naskah skripsi ini dapat dibuktikan terdapat unsur-unsur plagiasi, saya bersedia skripsi ini digugurkan dan gelar akademik yang telah saya peroleh (sarjana) dibatalkan, serta diproses sesuai dengan perundang-undangan yang berlaku **(UU No.20 Tahun 2003, Pasal 25 ayat 2 Pasal 70)**

Majene, 19 Mei 2025

NURFADHILAH ACO

NIM: D0221117

Abstrak

Nurfadhilah Aco Usulan Pendekatan *Cellular Automata* Dalam Optimalisasi Algoritma Dijkstra Untuk Pencarian Jalur Terpendek (dibimbing oleh Ibu **Dr. Eng. Sulfayanti, S.Si.,M.T** dan Bapak **Wawan Firgiawan,S.T.M.Kom**)

Penentuan jalur terpendek merupakan aspek krusial dalam optimalisasi berbagai bidang, seperti transportasi, logistik, dan jaringan komunikasi. Pemilihan jalur yang optimal tidak hanya mempercepat waktu tempuh tetapi juga berkontribusi terhadap efisiensi biaya operasional, penggunaan bahan bakar, serta sumber daya lainnya. Salah satu algoritma yang banyak digunakan dalam pencarian jalur terpendek adalah Algoritma Dijkstra, yang bekerja berdasarkan prinsip *greedy* untuk menemukan lintasan dengan bobot minimum pada graf berbobot. Namun, meskipun efektif, algoritma ini memiliki keterbatasan dalam hal kompleksitas komputasi, terutama pada jaringan besar dan dinamis.

Untuk mengatasi tantangan tersebut, pendekatan *Cellular Automata* dapat diintegrasikan ke dalam Algoritma Dijkstra guna meningkatkan efisiensi dalam pencarian jalur terpendek. *Cellular Automata* mampu menangani perubahan kondisi lingkungan secara real-time, seperti kemacetan atau perubahan rute, sehingga memberikan solusi yang lebih adaptif dan optimal. Dengan mengombinasikan kedua pendekatan ini, pencarian jalur terpendek dapat dilakukan secara lebih akurat dan efisien, terutama dalam sistem transportasi dan distribusi yang kompleks. Penelitian ini bertujuan untuk mengembangkan simulasi berbasis Algoritma Dijkstra yang

dioptimalkan dengan *Cellular Automata* guna mendukung pengambilan keputusan dalam pencarian jalur terpendek pada berbagai kondisi dinamis.

Kata Kunci: Jalur Terpendek, Algoritma Dijkstra, *Cellular Automata*, Optimasi, Transportasi.

Abstract

Nurfadhilah Aco *cellular automata approach in optimizing* dijkstra algorithm *for shortest path search* (dibimbing oleh Ibu **Dr. Eng. Sulfayanti, S.Si.,M.T** dan Bapak **Wawan Firgiawan,S.T.M.Kom**)

The determination of the shortest path is a crucial aspect of optimization in various fields, such as transportation, logistics, and communication networks. Selecting an optimal route not only reduces travel time but also enhances operational cost efficiency and resource utilization. One of the most widely used algorithms for finding the shortest path is Dijkstra's Algorithm, which operates based on the *greedy* principle to determine the minimum-weight path in a weighted graph. However, this algorithm has computational complexity limitations, particularly in large-scale and dynamic networks.

To address these challenges, the *Cellular Automata* approach can be integrated into Dijkstra's Algorithm to enhance the efficiency of shortest path searches. *Cellular Automata* enables real-time modeling of environmental changes, such as traffic congestion or route alterations, providing a more adaptive and optimal solution. The combination of these two methods is expected to improve the accuracy and efficiency of shortest path searches, especially in complex transportation and distribution systems. This study aims to develop a simulation-based model of Dijkstra's Algorithm optimized with *Cellular Automata* to support decision-making in various dynamic conditions.

BAB I

PENDAHULUAN

1.1 Latar Belakang

Penentuan jalur terpendek adalah aspek penting dalam mengoptimalisasikan di berbagai bidang, seperti bidang transportasi, logistic, dan berbagai jaringan komunikasi. Optimalisasi dalam pemilihan jalur terpendek tidak hanya mempercepat waktu yang dapat ditempuh tetapi juga dapat mengurangi kontribusi pada penghematan biaya operasional, penggunaan bahan bakar serta sumber daya lainya. Dalam sektor industri transportasi dan distribusi, jalur yang lebih pendek diartikan sebagai lintasan yang mempunyai bobot yang minimal, yaitu penjumlahan bobot dari seluruh lintasan yang dilalui (Wiladi, 2023). Bobot ini dapat mewakili jarak, waktu, atau biaya yang terkait dengan penggunaan suatu lintasan.

Penentuan lintasan terpendek pada praktiknya sering menghadapi berbagai tantangan salah satunya yaitu banyaknya alternatif jalur yang harus dipertimbangkan terutama dalam jaringan jalan yang padat atau kompleks. Sering kali sulit memperkirakan jalur mana yang memberikan jarak tempuh paling optimal secara real-time, mengingat perubahan kondisi di lapangan seperti kemacetan, cuaca, atau kecelakaan lalu lintas Oleh karena itu, dibutuhkan sistem yang mampu mempresentasikan data secara efisien dan menyederhanakan proses pencarian jalur terpendek. Sistem ini harus dapat menganalisis berbagai faktor secara simultan dan memberikan rekomendasi jalur terbaik. Salah satu algoritma yang terbukti efektif dalam menyelesaikan masalah ini adalah Algoritma Dijkstra, yang dikenal sebagai bagian dari algoritma *greedy*.

Algoritma ini dirancang untuk mencari lintasan terpendek dari satu titik ke titik lain pada graf berbobot.(Napiah & Darma Astuti, 2022.)

Algoritma Dijkstra adalah salah satu algoritma paling populer dan banyak digunakan dalam menyelesaikan masalah jarak terpendek pada graf berarah *(directed graph)*. Masalah ini sangat relevan terutama dalam kehidupan modern di perkotaan yang dinamis, di mana penentuan jalur tercepat atau terpendek sangat penting untuk menghemat waktu dan energi. Selain itu, di era digitalisasi dan revolusi industri 4.0 berbagai aplikasi berbasis teknologi seperti sistem navigasi, logistik pintar, dan kendaraan otonom, sangat bergantung pada algoritma seperti dijkstra untuk memandu proses pengambilan keputusan dalam pencarian jalur terbaik. Algoritma dapat menentukan titik-titik mana yang harus dilalui untuk mencapai tujuan dengan jarak terpendek dan waktu tersingkat yang pada akhirnya meningkatkan efisiensi operasional (Safutra 2024).

Masalah pencarian jalur terpendek merupakan bagian dari tantangan optimasi yang lebih luas. Representasi graf nilai bobot pada setiap sisi graf diartikan sebagai jarak, waktu, atau biaya antara dua titik (misalnya kota atau node dalam jaringan). Tujuan dari pencarian jalur terpendek adalah menemukan jalur yang meminimalkan bobot total sepanjang lintasan tersebut. Dalam mendukung proses tersebut, diperlukan pendekatan berbasis simulasi yang mampu mengatasi berbagai kemungkinan dan memfasilitasi pencarian jalur secara optimal. Algoritma Dijkstra yang juga termasuk dalam kategori algoritma *greedy* sangat efisien dalam mencari jalur terpendek dengan memanfaatkan prinsip pemilihan keputusan terbaik disetiap langkahnya (Sholahuddin 2025).Namun, meskipun efektif Algoritma Dijkstra sebagai solusi pencarian jalur

Algoritma Dijkstra sendiri telah digunakan secara luas dalam berbagai aplikasi nyata. Di Palangkaraya, misalnya algoritma ini diterapkan untuk menentukan rute terdekat dalam pengiriman barang (Christian Rufus et al., 2024) yang membantu mengoptimalkan rute pengiriman di wilayah tersebut. Selain itu, algoritma ini juga digunakan dalam sektor pariwisata, seperti pada implementasi rute terpendek untuk destinasi wisata di Klaten yang bertujuan untuk meningkatkan pengalaman wisatawan dengan menawarkan rute paling efisien menuju lokasi wisata. Penerapan algoritma dijkstra di kampus Universitas Ibnu Sina untuk pencarian rute terpendek menuju gedung-gedung penting di kampus juga menunjukkan bahwa algoritma ini sangat fleksibel dan dapat diaplikasikan pada berbagai konteks (Rizqy Amalia et al., 2025).

Mengatasi keterbatasan tersebut, pendekatan *Cellular Automata* dapat diintegrasikan dalam optimasi algoritma dijkstra. *Cellular Automata* merupakan model komputasi yang terdiri dari sel-sel yang berinteraksi secara lokal tetapi dapat menghasilkan perilaku global yang kompleks. Pengadopsian *Cellular Automata* dalam Algoritma Dijkstra simulasi pencarian jalur terpendek dapat dilakukan lebih efisien karena *Cellular Automata* mampu menangani dinamika perubahan lingkungan secara lebih realistis. Pada simulasi lalu lintas, *Cellular Automata* dapat memperhitungkan perubahan kondisi jalan akibat kemacetan yang memungkinkan pencarian jalur terpendek secara *real-time* dengan memperhitungkan variabel-variabel dinamis tersebut (Alharbi et al., 2023).

Integrasi Algoritma Dijkstra dengan pendekatan *Cellular Automata* diharapkan dapat memberikan solusi yang lebih optimal dalam pencarian jalur terpendek terutama dalam lingkungan yang dinamis dan kompleks. Simulasi yang dihasilkan akan lebih

representatif terhadap situasi dunia nyata memungkinkan penentuan jalur yang lebih akurat dan efisien. Penelitian ini bertujuan untuk mengembangkan simulasi berbasis Algoritma Dijkstra yang dioptimalkan dengan *Cellular Automata* guna membantu pengambilan keputusan dalam pencarian jalur terpendek pada algoritma dijkstra dan *Cellular Automata* pada berbagai situasi dinamis (Awal et al., 2023).

1.2 Rumusan Masalah

Berdasarkan latar belakang di atas, rumusan masalah dalam penelitian ini adalah bagaimana implementasi Algoritma Dijkstra dalam menentukan jalur terpendek dengan pendekatan *Cellular Automata*?

1.3 Tujuan

Tujuan penelitian ini adalah untuk mengetahui kinerja Algoritma Dijkstra dalam menentukan jalur terpendek dengan memanfaatkan pendekatan *Cellular Automata*.

1.4 Manfaat Penelitian

1. **Manfaat Teoritis:** Penelitian ini memberikan kontribusi pada pengembangan algoritma pencarian jalur terpendek, khususnya dalam penerapan Algoritma Dijkstra yang dikombinasikan dengan pendekatan *Cellular Automata*. Temuan penelitian ini diharapkan dapat menambah literatur dan menjadi acuan bagi penelitian selanjutnya dalam bidang optimasi jalur dan pengambilan keputusan dinamis.

2. **Manfaat Praktis:** Hasil penelitian ini dapat diimplementasikan pada berbagai sistem yang membutuhkan efisiensi rute, seperti sistem navigasi, manajemen logistik, dan layanan transportasi. Pengoptimasian jalur yang lebih efisien, pengguna akan dapat menghemat waktu dan sumber daya, sehingga meningkatkan performa operasional dalam kegiatan sehari-hari maupun skala industri.
3. **Manfaat Teknologi:** Penelitian ini juga berpotensi mendorong pengembangan aplikasi berbasis simulasi jalur terpendek untuk mendukung perencanaan rute di lingkungan yang dinamis. Teknologi ini akan bermanfaat dalam simulasi lalu lintas dan aplikasi smart city untuk meningkatkan mobilitas dan efisiensi di perkotaan.

1.5 Batasan Penelitian

1. **Lingkup Algoritma:** Penelitian ini terbatas pada penggunaan Algoritma Dijkstra untuk pencarian jalur terpendek yang dikombinasikan dengan pendekatan *Cellular Automata*.
2. **Batasan Data:** Simulasi dan eksperimen dalam penelitian ini menggunakan data graf berbobot yang representatif untuk jaringan jalur atau jalan. Penelitian ini tidak mencakup dataset jaringan yang kompleks dan besar, seperti jaringan transportasi kota secara keseluruhan atau jaringan distribusi logistik yang menggunakan data yang besar dan kompleks.
3. **Parameter:** Dalam penerapan *Cellular Automata*, perubahan lingkungan yang diperhitungkan terbatas pada beberapa parameter yang dapat

direpresentasikan oleh perubahan kondisi jalur atau bobot, seperti jarak tempuh. Faktor eksternal lain, seperti cuaca atau kondisi jalan yang dinamis, tidak diperhitungkan dalam simulasi.

BAB II

TINJAUAN PUSTAKA

2.1 Teori Graf

Graf merupakan struktur yang terdiri dari dua himpunan. himpunan titik (simpul) yang mewakili objek, dan himpunan sisi yang menggambarkan hubungan antara pasangan objek tersebut. Dengan demikian, graf dapat digunakan untuk memodelkan berbagai jenis objek beserta relasi di antara mereka.

1. $V(G)$, ialah *vertex* (simpul) atau himpunan vertex untuk memberi notasi , dengan V saja yakni himpunan tak kosong dari simpul-simpul
2. $E(G)$, ialah *edge* (sisi), umumnya digunakan notasi dengan E saja yakni himpunan (mungkin kosong) dari sisi yang menghubungkan satu pasang simpul. Tiap elemen dalam $E(G)$ termasuk suatu pasangan tak mempunyai urutan dari simpul-simpul di $V(G)$.

Dalam teori graf, istilah simpul (juga dikenal sebagai *vertex*, titik, atau noktah) digunakan untuk merepresentasikan objek-objek dalam suatu sistem. Jika terdapat sebuah sisi e yang menghubungkan dua simpul u dan v, maka e disebut sebagai sisi yang menghubungkan u dan v. Secara formal, sebuah graf didefinisikan sebagai pasangan himpunan $G = (V, E)$, di mana:

- V adalah himpunan tak kosong dan berhingga dari simpul-simpul E adalah himpunan sisi-sisi yang menghubungkan pasangan simpul dalam V.

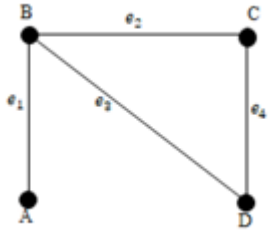
Dengan demikian, graf dapat digunakan untuk merepresentasikan berbagai model objek serta hubungan di antaranya.

2.1.1 Jenis Jenis Graf

Berdasarkan orientasi ada tindakan nya sisi ganda atau gelang,graf terbagi atas dua macam, yakni sebagai berikut.

a. Graf Sederhana (*Simple Graph*)

Graf sederhana adalah graf yang tidak mempunyai sisi ganda atau gelang.Dalam graf sederhana himpunan pasangan tak tertentu disebut dengan rusuk. Menurut, graf sederhana juga dapat diartikan selaku G = (V,E). Terbagi atas V yaitu *vertex* (simpul), himpunan tidak kosong, E yaitu *Edge* (sisi). Berikut adalah contoh graf sederhana



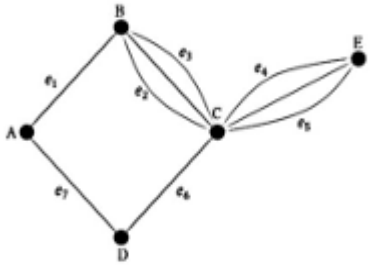
Gambar 2. 1 Graf Sederhana (*Simple Graph*)

(*Sumber* : okta khoirum ma'rifah, 2024)

Graf tak sederhana ialah graf yang mempunyai gelang atau sisi ganda.Graf tak sederhana dibagi atas dua macam, yakni:

2. Multigraph (Graf Ganda)

Graf dapat disebut ganda jika vertex (simpul) terhubung dengan vertex (simpul) lain dan hanya melalui satu edge (sisi) yang sama. Seperti contoh dalam gambar dibawah



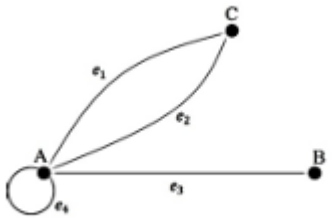
Gambar 2. 2 Graf Ganda

(*Sumber* : Okta Khoirum Ma'rifah, 2024)

Sehingga, tiap sisi sisi e yang dikaitkan dengan pasangan vertex (simpul) tak berurut. Lalu bila ada sisi yang mengaitkan dengan dua simpul V1 dan simpul v2 maka dapat dituliskan sebagai □1 = (□1,□2).

Graf Semu (*Pseudograph*)

Graf semu adalah graf dengan grafik berarah yang terkandung loop dan sebagian sisi (*edge*). Misalnya contoh dalam Gambar dibawah ini. Dalam gambar sisi (*edge*) *e*4 merupakan gelang, karena sisi dari *e*4 berhubungan dengan satu titik *v* pada gambar disimbolkan dengan A. Kemudian pada pada sisi (*edge*) *e*1 dan *e*2 sama-sama menghubungkan antara kedua titik simpul (*vertex*) yaitu A dan C.



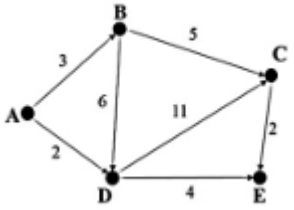
Gambar 2. 3 Graf Semu (*Pseudograph*)

(*Sumber* : Okta Khoirum Ma'rifah, 2024)

b. Graf Tak Berarah dan Berbobot

1. Graf Berarah dan Berbobot

dikatakan berarah dan berbobot jika tiap sisi yang memiliki orientasi arah juga memiliki bobot tiap sisinya. Pada Gambar memperlihatkan bahwa graf berbobot dan berarah memiliki lima titik yang terbagi atas titik A, B, C, D, E. Titik A memperlihatkan arahan pada titik B, titik B ke arah titik C, titik C ke arah titik E, titik D ke arah titik C dan selanjutnya.

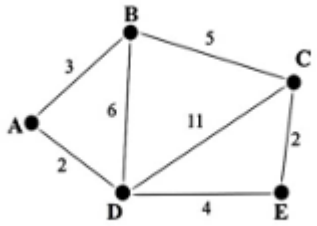


Gambar 2. 4 Graf Berarah dan Berbobot

(*Sumber* : Okta Khoirum Ma'rifah, 2024)

2. Graf Tak Berarah dan Berbobot

Graf tak berarah dan berbobot adalah graf yang setiap sisinya tidak diberikan orientasi arah tetapi memiliki bobot tiap sisinya. Pada Gambar 2.5, terdiri dari lima titik yakni titik A, B,C, D, E. Titik A tidak menuju ke titik B atau D, akan tetapi antara titik A dan B telah diketahui bobotnya. Begitu pula pada titik-titik yang lain.

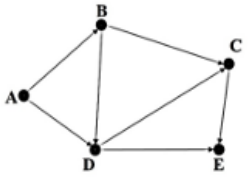


Gambar 2. 5 Graf Tak Berarah dan Berbobot

(*Sumber* : Okta Khoirum ma'rifah, 2024)

3. Graf Berarah dan Tak berbobot

Graf berarah dan tak berbobot adalah graf yang sisinya diberikan orientasi arah akan tetapi tidak berbobot tiap sisinya. Dalam Gambar dibawah memperlihatkan graf tak berbobot dan berarah.

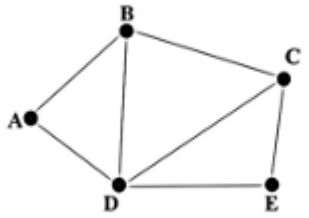


Gambar 2. 6 Graf Berarah dan Tak berbobot

(*Sumber* : okta khoirum ma'rifah, 2024)

4. Graf Tak Berarah dan Tak Berbobot

Graf tak berbobot dan tak berarah adalah graf yang sisinya tidak diberikan orientasi arah dan tidak memiliki bobot. Pada Gambar 2.7 mempunyai graf tak berarah dan tak berbobot.



Gambar 2. 7 Graf Tak Berarah dan Tak Berbobot

(*Sumber* : Okta Khoirum Ma'rifah, 2024)

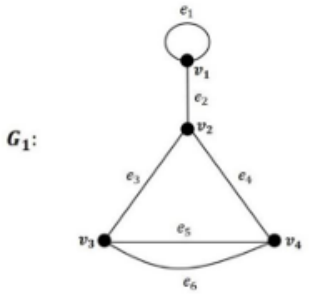
2.1.2 Representasi Graf

Jika graf akan diproses menggunakan program komputer, maka graf tersebut harus direpresentasikan di dalam memori. Terdapat beberapa representasi yang memungkinkan untuk graf, di antaranya adalah sebagai berikut.

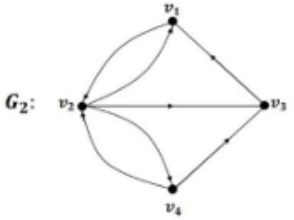
a. Matriks Ketetanggaan (*Adjacency Matrix*)

Jika graf G adalah graf dengan n simpul yaitu $v1$, $v2$,$v3$,…,vn maka matriks ketetanggaan dari graf G adalah suatu matriks $A(G) = (aij)$ berukuran $n \times n$ dimana aij adalah banyaknya sisi yang menghubungkan simpul vi dan vj. Matriks ketetanggaan untuk graf sederhana dan tidak berarah selalu simetri sedangkan untuk graf berarah matriks ketetanggaannya belum tentu simetri (akan simetri jika berupa graf berarah lengkap). Selain itu diagonal utamanya

selalu nol karena tidak terdapat sisi gelang (loop). Jumlah elemen matriks ketetanggaan untuk graf dengan n simpul adalah $n2$.



$$A(G_1)_{4\times4} = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 2 \\ 0 & 1 & 2 & 0 \end{bmatrix}$$



$$A(G_2)_{4\times4} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 \end{bmatrix}$$

Gambar 2. 8 Matriks Ketetanggaan (*Adjacency Matrix*)

(*Sumber* : Okta Khoirum Ma'rifah, 2024)

Keuntungan representasi graf dengan menggunakan matriks ketetanggaan adalah elemen matriksnya dapat diakses langsung melalui indeks. Selain itu juga dapat digunakan untuk menentukan secara langsung apakah simpul i dan j bertetangga.

Derajat setiap titik simpul i dapat dihitung dari matriks ketetanggaan. Untuk graf tak berarah cara perhitungannya sebagai berikut ini.

$$d(v_i) = \sum_{j=1}^{n} a_{ij} \tag{2.1}$$

Sedangkan untuk graf berarah adalah:

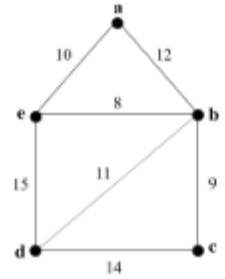
$$d_{in}(v_j) = \sum_{i=1}^{n} a_{ij}, \tag{2.2}$$

Dan

$$d_{out}(v_i) = \sum_{j=1}^{n} a_{ij} \tag{2.2}$$

Bagi graf berbobot *aij* yang menyatakan bobot untuk setiap sisinya yang menghubungkan simpul *i* dan simpul *j*. Jika tidak terdapat sisi yang menghubungkan antara simpul *i* dan simpul *j* atau dari simpul *i* ke simpul *i* sendiri maka *aij*=∞ (diberikan nilai tak berhingga).

G:



$$A(G)_{5\times5} = \begin{bmatrix} \infty & 12 & \infty & \infty & 10 \\ 12 & \infty & 9 & 11 & 8 \\ \infty & 9 & \infty & 14 & \infty \\ \infty & 11 & 14 & \infty & 15 \\ 10 & 8 & \infty & 15 & \infty \end{bmatrix}$$

Gambar 2. 9 Matriks Ketetanggaan (*Adjacency Matrix*)

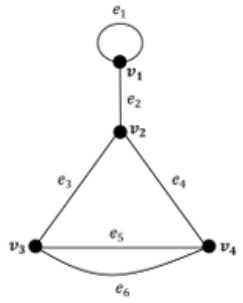
(*Sumber* : Okta Khoirum Ma'rifah, 2024)

b. Matriks Bersisian (*Incidency Matrix*)

Jika G adalah graf dengan n simpul yaitu $v1,v2,v3,\ldots,vn$ dan t sisi yaitu $e1,e2,e3,\ldots,et$ maka matriks bersisian pada graf adalah suatu matriks $M(G)=(mij)$ berukuran $n\times t$ dimana masukan mi,j diberikan oleh:

$$m_{i,j} = \begin{cases} 0. & \text{simpul } v_i \text{ tidak bersisian dengan sisi } e_j \\ 1. & \text{simpul } v_i \text{ bersisian dengan sisi yang bukan sisi gelang } (loop)\ e_j \\ 2. & \text{simpul } v_i \text{ bersisian dengan sisi gelang } (loop)\ e_j \end{cases}$$

G:



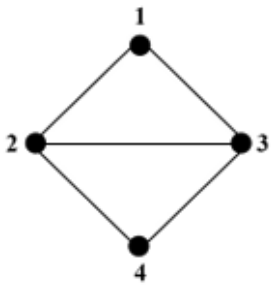
$$M(G)_{4\times 6}\begin{bmatrix} 2 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 \end{bmatrix}$$

Derajat simpul i dapat dihitung dengan menghitung jumlah seluruh elemen pada baris i (kecuali jika pada graf yang memuat gelang). Jumlah elemen matriks bersisian adalah nt.

c. Senarai Ketetanggaan

Kelemahan dari matriks ketetanggaan adalah bila graf yang memiliki jumlah sisi relatif sedikit matriksnya bersifat jarang (sparse) yaitu memuat banyak elemen nol sedangkan elemen yang bukan nol sedikit. Ditinjau dari implementasinya di dalam komputer, kebutuhan ruang memori untuk matriks jarang boros karena komputer menyimpan elemen yang tidak perlu. Pencegahan dari masalah ini adalah menggunakan representasi senarai

ketetanggaan. Senarai ketetanggaan mengenumerasi simpul-simpul yang bertetangga dengan setiap simpul dalam graf.



Simpul:	Simpul tetangga:
1	2, 3
2	1, 2, 3
3	1, 2, 4
4	2, 3

Gambar 2. 10 Senarai Ketetanggan

(*Sumber* : okta khoirum ma'rifah, 2024)

2.2. Cellular Automata

Pendekatan *Cellular Automata* adalah metode yang digunakan untuk memodelkan sistem kompleks dengan membagi sistem tersebut menjadi unit-unit kecil (sel) yang berinteraksi secara lokal. Setiap sel dalam model ini mengikuti aturan sederhana, namun melalui interaksi antar-sel, terbentuklah perilaku global yang jauh lebih kompleks. Pendekatan ini mampu merepresentasikan sistem dinamis di mana perubahan keadaan pada sel-sel tetangga turut memengaruhi kondisi keseluruhan sistem, sehingga menghasilkan pola perilaku yang lebih rumit dan realistis

Cellular Automata adalah model komputasi berbasis grid yang terdiri dari sekumpulan sel yang masing-masing memiliki satu atau beberapa keadaan. Keadaan sel ini berubah secara berkala sesuai aturan yang memperhitungkan kondisi sel itu

sendiri dan kondisi sel-sel tetangganya. Model ini memungkinkan simulasi dinamika sistem kompleks melalui interaksi sederhana antara sel, yang pada akhirnya membentuk pola perilaku global yang dapat beradaptasi dengan perubahan lingkungan. *cellular automata*, dengan aturan yang tampak sederhana, mampu menghasilkan representasi yang realistis untuk sistem yang kompleks.

Konsep *cellular automata* pertama kali diperkenalkan oleh matematikawan John von Neumann pada tahun 1940-an untuk memahami proses otomatisasi dan reproduksi diri di alam. Sejak itu, model ini dikembangkan lebih lanjut, terutama oleh Stephen

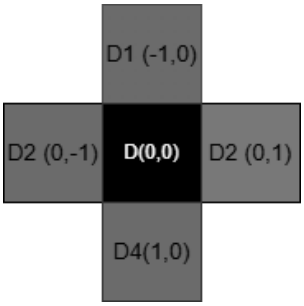
Wolfram pada 1980-an, yang mengeksplorasi pola-pola yang dapat muncul dari aturan cellular automata sederhana. *Cellular automata* kini diterapkan dalam berbagai bidang, seperti fisika, ekologi, biologi, dan ilmu komputer, untuk memodelkan fenomena seperti penyebaran api, pertumbuhan populasi, dan pola lalu lintas. Model ini juga berguna dalam optimasi dan komputasi, terutama dalam simulasi jalur dan pencarian rute, seperti kombinasi *cellular automata* dengan Algoritma Dijkstra untuk pencarian jalur terpendek pada lingkungan dinamis (Alharbi et al., 2023).

1.2.1 **Jenis Jenis *Cellular Automata***

a. *4-Direction Graf*

4-Directional Graph adalah representasi graf berbasis grid dua dimensi di mana setiap simpul (node) terhubung ke tetangganya dalam empat arah utama: atas, bawah, kiri, dan kanan. Model ini umum digunakan dalam berbagai aplikasi seperti algoritma pencarian jalur *(pathfinding)*, pemrosesan citra, dan simulasi berbasis grid.

Dalam graf 4 arah, setiap simpul pada grid memiliki maksimal empat tetangga, tergantung pada posisinya di dalam grid. Misalnya, simpul yang berada di tepi grid mungkin hanya memiliki dua atau tiga tetangga. Arah gerak biasanya direpresentasikan sebagai pasangan koordinat (dx, dy):



Gambar 2. 11 *4-Direction Graf*

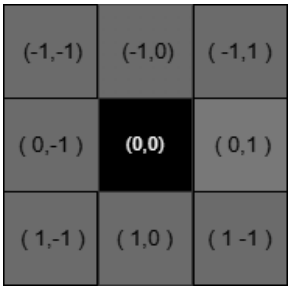
(Sumber : Putra Putra, 2024)

b. *8 - Directional Graph*

8-Directional Graph adalah representasi graf berbasis grid dua dimensi di mana setiap simpul (node) terhubung ke delapan tetangganya: empat arah utama (atas, bawah, kiri, kanan) dan empat arah diagonal (kiri atas, kanan atas, kiri bawah, kanan bawah). Model ini umum digunakan dalam berbagai aplikasi seperti algoritma pencarian jalur (*pathfinding*), pemrosesan citra, dan simulasi berbasis grid.

Dalam graf 8 arah, setiap simpul pada grid memiliki maksimal delapan tetangga, tergantung pada posisinya di dalam grid. Misalnya, simpul yang

berada di sudut grid mungkin hanya memiliki tiga tetangga. Arah gerak biasanya direpresentasikan sebagai pasangan koordinat (dx, dy):



Gambar 2. 12 *8-Directional Graph*

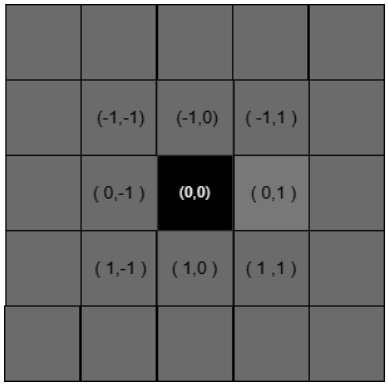
(Sumber : Putra Putra, 2024)

c. *16-Directional Graph*

16-Directional Graph adalah representasi graf berbasis grid dua dimensi di mana setiap simpul (node) terhubung ke enam belas tetangganya. Model ini umum digunakan dalam berbagai aplikasi seperti algoritma pencarian jalur (*pathfinding*), pemrosesan citra, dan simulasi berbasis grid.

Dalam graf 16 arah, setiap simpul pada grid memiliki maksimal enam belas tetangga, tergantung pada posisinya di dalam *grid.* Misalnya, simpul yang berada di sudut grid mungkin hanya memiliki beberapa tetangga. Arah gerak biasanya direpresentasikan sebagai pasangan koordinat (dx,dy) yang mencakup delapan arah utama dan delapan arah tambahan di antara arah utama tersebut.

Contoh pasangan koordinat untuk arah-arah tersebut meliputi:



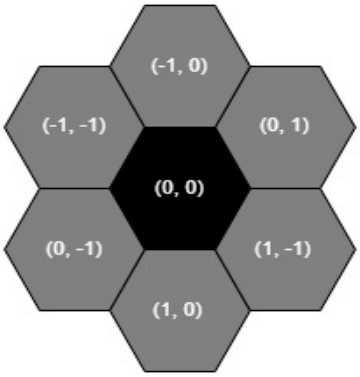
Gambar 2. 13 *16-Directional Graph*
(Sumber : Putra Putra, 2024)

d. *6-Directional Graph*

6-Directional Graph adalah representasi graf berbasis grid heksagonal dua dimensi di mana setiap simpul (node) terhubung ke enam tetangganya. Model ini umum digunakan dalam berbagai aplikasi seperti algoritma pencarian jalur (pathfinding), pemrosesan citra, dan simulasi berbasis grid.

Dalam graf 6 arah, setiap simpul pada grid heksagonal memiliki enam tetangga, tergantung pada posisinya di dalam *grid*. Misalnya, simpul yang berada di tepi grid mungkin hanya memiliki tiga atau empat tetangga. Arah gerak biasanya direpresentasikan sebagai pasangan koordinat (dx, dy) yang mencakup enam arah utama sesuai dengan orientasi grid heksagonal.

Contoh pasangan koordinat untuk arah-arah tersebut meliputi:

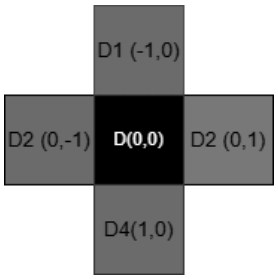


Gambar 2. 14 *6-Directional Graph*

(Sumber : Putra Putra, 2024)

1.2.2 Model *graph* yang diterapkan dalam simulasi

Penulis menerapkan model *graph 4-arah (4-Directional Graph),* yakni sebuah representasi graf berbasis grid dua dimensi di mana setiap simpul (node) memiliki keterhubungan dengan simpul lain dalam empat arah utama: atas, bawah, kiri, dan kanan. Model ini lazim digunakan dalam berbagai bidang seperti algoritma pencarian jalur (*pathfinding*), pengolahan citra, serta simulasi berbasis grid. Pada *graph* 4-arah ini, setiap simpul dapat memiliki hingga empat tetangga, bergantung pada posisinya di dalam grid.



(Sumber : Putra Putra, 2024)

Contoh Penerapan dalam *Cellular Automata* menggunakan Algoritma Dijkstra

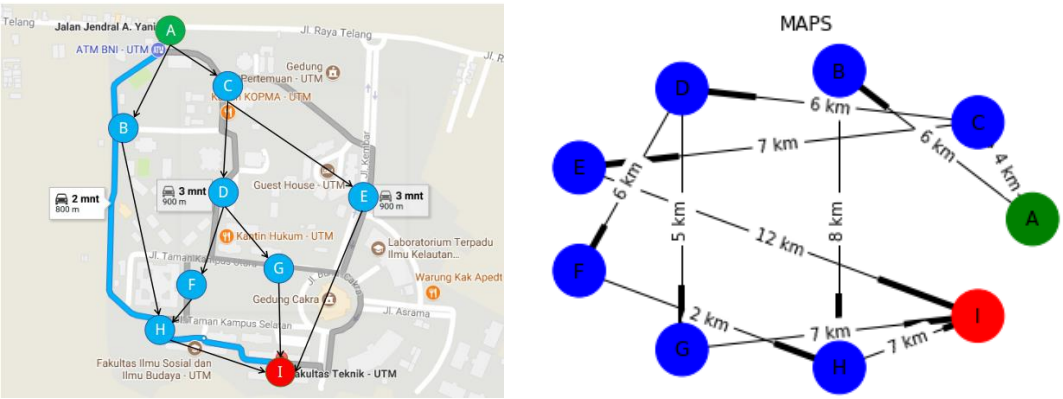
Tabel 3. 1 Penerapan *Cellular Automata*

0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0
0	1	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0	1	0
0	1	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0	1	0
0	1	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0	1	0
0	1	0	0	0	1	0	0	0	0	1	1	1	0	0	0	0	0	1	0
0	1	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	1	0
0	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	1	0
0	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	1	0
0	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	1	0
0	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	1	0
0	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	1	0
0	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	1	0
0	0	0	0	0	1	0	0	0	0	1	1	1	1	1	1	1	1	1	0
0	0	0	0	0	D1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	D4	1	D2	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	D3	0	0	0	0	0	0	0	0	0	0	0	0	0	0

2.3. Jalur Terpendek

Jalur terpendek adalah jalur dalam sebuah graf yang memiliki total bobot atau jarak terkecil antara dua simpul (node) tertentu. Dalam graf, setiap simpul dihubungkan oleh tepi (*edge*) yang memiliki bobot, yang dapat merepresentasikan berbagai parameter seperti jarak fisik, waktu tempuh, biaya, atau bahkan tingkat kesulitan. Bobot ini berfungsi sebagai indikator kuantitatif yang menentukan nilai dari setiap jalur yang mungkin ditempuh antara dua simpul. Oleh karena itu, jalur terpendek adalah hasil minimalisasi dari total bobot yang dilalui, memberikan

efisiensi maksimum dalam mencapai tujuan dari satu titik ke titik lainnya (Arif Sudibyo et al., 2021).



a) Ilustrasi Pengambilan Data b) Implementasi Data ke Graf berbobot

Gambar 2. 15 Proses Pengambilan Data untuk Penentuan Jalur Terpendek

Pencarian jalur terpendek penting dalam banyak aplikasi, terutama dalam bidang seperti transportasi, telekomunikasi, dan logistik. Di bidang transportasi, misalnya, pencarian jalur terpendek dapat membantu menentukan rute paling efisien untuk kendaraan, yang memungkinkan penghematan waktu dan bahan bakar. Di bidang jaringan komputer, jalur terpendek berguna untuk optimasi rute data, mengurangi latensi, dan meningkatkan kecepatan transmisi. Oleh karena itu, jalur terpendek menjadi salah satu konsep kunci dalam optimasi dan sering digunakan dalam analisis graf serta pengembangan algoritma.

Berbagai algoritma digunakan untuk menemukan jalur terpendek pada graf, di antaranya Algoritma Dijkstra, Bellman-Ford, dan A*. Algoritma-algoritma ini bekerja dengan mengevaluasi bobot setiap tepi untuk menemukan jalur dengan nilai

terkecil dari titik asal ke titik tujuan. Algoritma Dijkstra, misalnya, terkenal karena keefektifannya dalam menemukan jalur terpendek pada graf dengan bobot positif, dan sangat sesuai untuk jaringan yang membutuhkan pemrosesan jalur secara efisien.

2.4.Optimalisasi Jalur

Optimalisasi adalah proses atau tindakan yang bertujuan untuk mencapai hasil yang paling efisien, efektif, atau mendekati kondisi ideal dari suatu sistem atau proses. Dalam konteks umum, optimalisasi melibatkan usaha untuk memaksimalkan atau meminimalkan suatu fungsi, variabel, atau parameter kinerja, tergantung pada tujuan spesifik yang ingin dicapai. Tujuan optimalisasi dapat berupa pengurangan waktu, biaya, atau energi, serta peningkatan kualitas, produktivitas, atau efisiensi keseluruhan (Nirwana et al., 2024)

Dalam konteks sistem transportasi dan jaringan, optimalisasi jalur adalah upaya untuk menemukan rute atau jalur paling efisien dari satu titik ke titik lain. Hal ini sangat penting dalam manajemen logistik, pengiriman barang, dan navigasi lalu lintas, dimana jalur yang optimal dapat menghemat waktu, mengurangi konsumsi bahan bakar, serta menekan biaya operasional. Optimalisasi jalur juga memungkinkan penurunan kepadatan lalu lintas, yang pada gilirannya berkontribusi terhadap pengurangan polusi udara dan dampak negatif terhadap lingkungan. Oleh karena itu, proses optimalisasi jalur memiliki peran penting dalam mendukung keberlanjutan lingkungan dan efisiensi operasional dalam berbagai sektor.

Optimalisasi jalur sering dilakukan dengan menggunakan algoritma pencarian dan teknik komputasi yang dapat mengevaluasi berbagai kemungkinan rute berdasarkan kriteria tertentu, seperti jarak terpendek atau waktu tercepat. Algoritma Dijkstra, misalnya, merupakan salah satu algoritma yang paling umum digunakan untuk mencari jalur terpendek dalam graf berbobot. Dengan perkembangan teknologi dan metode komputasi yang semakin canggih, pendekatan optimalisasi jalur kini dapat dikombinasikan dengan model dinamis, seperti *Cellular Automata,* yang memungkinkan pengambilan keputusan real-time di tengah kondisi yang berubah-ubah. Hal ini menjadi penting dalam lingkungan dinamis, seperti pengaturan lalu lintas atau jaringan transportasi, di mana perubahan kondisi jalan atau permintaan lalu lintas membutuhkan respons cepat untuk mempertahankan efisiensi jalur yang optimal

2.5. Algoritma Dijkstra

Algoritma Dijkstra adalah algoritma yang digunakan untuk menentukan lintasan terpendek dari satu titik tertentu ke titik-titik lainnya dalam suatu graf berbobot. Algoritma ini dikembangkan oleh ilmuwan komputer Belanda, Edsger Wybe Dijkstra, pada tahun 1959, dan hingga saat ini masih menjadi salah satu algoritma paling efisien untuk pencarian jalur terpendek pada graf dengan bobot positif. Lintasan terpendek antara dua titik ditentukan melalui pohon pembangun yang memiliki nilai minimum, sehingga memungkinkan untuk mendapatkan jalur dengan total bobot terkecil dari titik asal ke titik tujuan (Arif Sudibyo et al., 2020).

Algoritma Dijkstra bekerja dengan menggunakan bobot setiap sisi pada graf untuk menghitung jarak terpendek antara dua titik. Algoritma ini melakukan eksplorasi

dari satu titik asal (*source*) ke titik lainnya dengan memperbarui jarak terpendek secara iteratif hingga mencapai semua simpul atau hingga mencapai simpul tujuan. Setiap simpul dalam graf diberikan label jarak sementara, yang akan diperbarui ketika algoritma menemukan jalur yang lebih pendek. Algoritma ini sangat efisien untuk graf terhubung dengan bobot positif karena meminimalkan jumlah pengulangan yang diperlukan untuk mencapai simpul tujuan. Berikut adalah langkah-langkah rinci dari algoritma Dijkstra.

Input:

- G: Graf sederhana yang terhubung dengan bobot positif pada setiap sisi
- ∞: Angka lebih besar dari jumlah total bobot seluruh sisi dalam graf
- w(u,v): Bobot sisi antara simpul u dan v
- a: Titik awal *(source*)
- z: Titik tujuan (*destination*)
- V: *evertex*
- E: *edge*
- F: himpunan yang digunakan untuk menyimpan simpul-simpul yang telah diproses atau yang sudah ditemukan jarak terpendek nya.
- D:adalah pendahulu atau simpul yang datang sebelum simpul u pada jalur terpendek

Langkah Algoritma:

1. Inisialisasi graf G dengan simpul a dan tanpa sisi. Misalkan V(T) adalah himpunan simpul dari T, dan E(T) adalah himpunan sisi dari T.
2. Atur L(a) = 0 untuk simpul awal a, dan untuk semua simpul lainnya dalam graf, tetapkan L(u) = ∞.
3. Tetapkan v sama dengan a dan inisialisasi F menjadi {a}.
4. Selama z belum termasuk dalam V(T), lakukan langkah berikut:

- Perbarui *F* dengan menambahkan simpul yang berdekatan dengan *v* dan tidak berada dalam *V(T)*.
- Untuk setiap simpul *u* yang berdekatan dengan *v* dan tidak berada dalam *V(T)*, jika $L(v) + w(v, u) < L(u)$, make perbaru $L(u) = L(v) + w(v, u)$ dan set $D(u) := v$.
- Temukan simpul *a* dalam *F* yang memiliki label terkecil
- Tambahkan simpul *x* ke dalam *V(T)* dan tambahkan sisi *{D(x), x}* ke dalam *E(T)*. Atur *v* sebagai *x*.

Output:

Algoritma Dijkstra sangat efisien untuk jaringan dengan graf berarah dan bobot positif, dan hingga kini, algoritma ini menjadi solusi dalam berbagai aplikasi seperti navigasi GPS, jaringan komputer, dan sistem informasi geografis. *L(z)* yang mewakili jarak terpendek dari titik awal *a* ke titik tujuan *z*.

1.3 Penelitian Terdahulu

Adapun penelitian terdahulu yang relevan dengan topik penelitian ini dapat dilihat pada tabel 3.2. Beberapa penelitian sebelumnya telah menggunakan algoritma dijkstra untuk pendekatan *cellular automata* dalam pencarian jalur terdekat. Namun Penelitian-penelitian tersebut umumnya berfokus pada objek analisis yang berbeda

Tabel 3. 2 Penelitian Sebelumnya

NO	Judul Penelitian	Penulis	Algoritma	Objek	Hasil	Perbedaan penelitian
1	Implementasi Algoritma Dijkstra Dalam Penentuan Jalur Terpendek Studi Kasus Jarak Tempat Kuliah Terdekat	Muhammad Muharram	Algoritma Dijkstra	Jarak Tempat Kuliah Terdekat	Menemukan rute tercepat dan rute terpendek dari suatu lokasi ke lokasi yang lain, dalam hal ini berupa suatu persimpangan ke persimpangan yang lain adalah dengan menggunakan algoritma Dijkstra dengan input berupa persimpangan asal dan tujuan serta sebuah graf yang merepresentasikan nod sebagai persimpangan dan simpul atau path sebagai jalur yang menghubungkannya.	Penelitian ini menggunakan pendekatan cellular automata dalam pencarian jalur terpendek dengan menggunakan algoritma Dijkstra
2	Implementasi Algoritma Dijkstra Dalam Pencarian Rute Terpendek Tempat Wisata	Rahmad Hidayat	Algoritma Dijkstra	Pencarian Rute Terpendek Tempat Wisata Di Kabupaten Klaten	Penelitian yang telah dilakukan menemukan dan menyimpulkan sebagai berikut. Algoritma Dijkstra dapat digunakan dalam pencarian rute terpendek tempat wisata di Kabupaten Klaten. Dari hasil	Penelitian ini menggunakan pendekatan cellular automata dalam pencarian jalur terpendek dengan

	Di Kabupaten Klaten				perhitungan yang secara manual dan yang menggunakan *Software Tora*, diperoleh hasil yang sama.	menggunakan algoritma Dijkstra
3	perencanaan jalur evakuasi kebakaran yang efisien untuk fasilitas perawatan rumah sakit dengan menggunakan algoritma dijkstra	Nur Ihwan Safutra Asrul Fole Alam Gunawan Muhammad Fachry Hafid Arfandi Ahmad Yan Herdianzah	Algoritma Dijkstra	Gedung perawatan RSUD I Lagaligo	Tujuan yang dicapai dalam penelitian ini adalah untuk mengembangkan jalur evakuasi kebakaran di RSUD I Lagaligo yang terletak di Kabupaten Luwu Timur dengan menggunakan Algoritma Dijkstra. Tujuan utamanya adalah untuk meningkatkan kualitas layanan sebagai bagian dari proses renovasi bangunan. Algoritma Dijkstra dipilih karena kemampuannya dalam menentukan jalur tercepat dan terpendek antara dua titik dalam suatu graf. Penelitian ini menggunakan teknik analisis data yang sistematis dan logis untuk menentukan rute evakuasi yang paling optimal. Hasil	Penelitian ini menggunakan pendekatan cellular automata dalam pencarian jalur terpendek dengan menggunaknalgoritma Dijkstra

					penelitian menunjukkan adanya pengurangan yang signifikan dalam jarak evakuasi setelah penerapan Algoritma Dijkstra, yang berdampak pada peningkatan efisiensi waktu tempuh. Selain itu, titik kumpul secara umum tetap konsisten, kecuali pada beberapa jalur tertentu yang mengalami perubahan titik kumpul. Temuan ini menunjukkan efektivitas Algoritma Dijkstra dalam meningkatkan perencanaan evakuasi kebakaran, sehingga berkontribusi terhadap peningkatan keselamatan dan efektivitas operasional RSUD I Lagaligo. Penelitian ini memberikan wawasan yang berharga bagi manajemen rumah sakit dan pemangku kepentingan yang terlibat dalam renovasi serta peningkatan sistem	

					keselamatan fasilitas kesehatan.	
4.	Perbandingan Algoritma Greedy dan Algoritma Dijkstra dalam Pencarian Rute Terpendek dari Kabupaten Tuban ke Kota Surabaya	Jonathan Steven Iskandar Yosefina Finsensia Riti.	Algoritma Dijkstra	Kabupaten Tuban ke Kota Surabaya	Pada peneliti terdahulu mengenai pencarian rute terpendek dari pendonor darah terdekat, menyimpulkan hasil pengujian pertumbuhan waktu kode program didapatkan bahwa nilai fungsi n dari algoritma Dijkstra lebih besar daripada algoritma *Greedy*, sehingga waktu eksekusi dari algoritma Dijkstra lebih cepat dibandingkan dengan algoritma *Greedy*.Namun pada penelitian ini, pengujian parameter pertama menghitung waktu eksekusi kode program yang dijalankan melalui bahasa Python ternyata membuktikan	Penelitian ini menggunakan pendekatan cellular automata dalam pencarian jalur terpendek dengan menggunakan algoritma Dijkstra

					bahwa algoritma *Greedy* lebih cepat daripada algoritma Dijkstra dengan pengujian 3 kali dimana sebesar 0.00023706666669524405 detik, dan pada algoritma Dijkstra 0.0006108666668422913 detik. Parameter kedua mengenai kompleksitas kode program bahasa Python yang didapatkan algoritma *Greedy* sebanyak 43 baris, dan algoritma Dijkstra sebanyak 69 baris. Parameter ketiga melakukan pengujian urutan atau sequence program yang dijalankan melalui program bahasa Python. Hasil algoritma *Greedy* pada urutannya hanya memperhatikan langkah panjang rute terpendek di depannya,akan tetapi algoritma Dijkstra pada urutannya selalu	

					memperhatikan keseluruhan rute yang dilewati atau dilalui,sehingga mendapatkan bobot paling kecil dari total keseluruhan. Sehingga urutan pengujian perbandingan ini lebih cepat algoritma *Greedy* daripada algoritma Dijkstra.Parameter keempat terkait hasil jarak terpendek, baik dalam pengujian secara dasar atau manual, maupun dilakukan oleh kode program bahasa Python. Hasil rute terpendek yang diperoleh ternyata algoritma Dijkstra sebesar 105 (km) lebih pendek daripada algoritma *Greedy* sebesar 129 (km). Rute-rute yang dilalui algoritma Dijkstra adalah Tuban - Widang - Babat - Pucuk - Lamongan - Duduk	

					Sampeyan - Ambeng Ambeng – Kebomas Surabaya . Dengan demikian, melalui perbandingan parameter algoritma yang diuji, dapat disimpulkan bahwa parameter waktu eksekusi, kompleksitas, dan urutan pengujian program lebih baik dilakukan oleh algoritma *Greedy*. Akan tetapi, pada pengujian parameter hasil luaran jarak terpendek yang ditempuh dalam hasil rute paling, ternyata lebih baik algoritma Dijkstra daripada algoritma *Greedy*. Melalui penelitian ini dapat direkomendasikan penggunaan algoritma *Greedy* dan algoritma Dijkstra dalam	
5.	Penerapan Algoritma Dijkstra Dalam	Edward Christian Rufus,	Algoritma Dijkstra	Menentukan Rute Terpendek	Penelitian ini menunjukkan bahwa algoritma	Penelitian ini menggunakan pendekatan cellular

	Menentukan Rute Terpendek Untuk Jasa Pengiriman Barang Di Palangka Raya	Raydamar Rizkyaka Riyadi, Dicky Nugraha Hasibuan, Efrans Christian		Untuk Jasa Pengiriman Barang Di Palangka Raya	Dijkstra mampu mengoptimalkan rute pengiriman barang untuk jasa kurir express di Palangka Raya dengan mencari rute terpendek dari peta jalur yang dimodelkan ke dalam bentuk graf. Meskipun demikian, algoritma Dijkstra memiliki keterbatasan yaitu hanya mempertimbangkan jarak sebagai faktor utama dan membutuhkan waktu komputasi yang lebih lama untuk graf dengan ukuran yang lebih besar karena jumlah iterasinya juga akan semakin banyak. Saran untuk penelitian selanjutnya adalah untuk mengembangkan algoritma yang dapat mempertimbangkan faktor-faktor selain jarak seperti kondisi jalan dan kemacetan lalu lintas, serta	automata dalam pencarian jalur terpendek dengan menggunaknalgoritma Dijkstra

					meningkatkan efisiensi waktu komputasi.Penerapan algoritma optimasi rute seperti ini dapat membantu perusahaan jasa pengiriman meningkatkan kepuasan pelanggan dan daya saing dengan menyediakan layanan yang lebih cepat dan efisien.	
6	Penerapan Algoritma Dijkstra Dan Algoritma Greedy Pada Optimasi Jalur Evakuasi Banjir	MuliyaWiladi Wasono Asmaidi3	Algoritma Dijkstra	Jalur Evakuasi Banji	Berdasarkan hasil dan pembahasan dari penelitian ini dapat disimpulkan bahwa penggabungan Algoritma Dijkstra dan Algoritma *Greedy* dapat digunakan pada kasus penentuan jalur evakuasi banjir di Jl. Terong dan Jl. Terong Pipit, Kelurahan Sempaja Timur, Kota Samarinda. Pada penelitian ini didapatkan 11 lintasan optimal dengan total bobot tidak lebih dari batas maksimum yaitu 10.	Penelitian ini menggunakan pendekatan cellular automata dalam pencarian jalur terpendek dengan menggunakan algoritma Dijkstra

					Lintasan-lintasan optimal tersebut merupakan urutan evakuasi banjir yang dapat	
7	Implementasi Algoritma Dijkstra Pada Pencarian Rute Terpendek Objek Wisata	Muliya Wiladi Perayoga,Rachman, Hendradi, Purwono, Setiawan, Agus	Algoritma Dijkstra	Pencarian Rute Terpendek Objek Wisata	Mengacu dari dari hasil implementasi yang sudah diulas sebelumnya maka bisa dikatakan bila pengembangan sistem informasi Dinas Pariwisata Kabupaten Temanggung berbasis website ini mempunyai informasi mengenai lokasi pariwisata yang ada di Kabupaten Temanggung, beserta bisa memberi rute terpendek dari titik awal menuju objek wisata tujuan maupun dari sebuah objek wisata ke objek wisata yang lain. Melalui aplikasi ini, wisatawan bisa mencari informasi seputar wisata yang ada di Kabupaten Temanggung, sebab aplikasi ini bisa memberi informasi wisata yang ada di	Penelitian ini menggunakan pendekatan cellular automata dalam pencarian jalur terpendek dengan menggunakan algoritma Dijkstra

					Kabupaten Temanggung saja. Melalui metode Rapid *Application Development* dalam sistem informasi berbasis website harapannya bisa membantu memberi kemudahan pengguna sebagai wisatawan guna menetapkan jalur menuju lokasi wisata khususnya yang ada di Kabupaten Temanggung. Untuk admin harapannya bisa mengatur secara mudah serta fleksibel. Melalui sistem informasi ini harapannya pula bisa turut mempromosikan objek wisata yang ada di Kabupaten Temanggung. Pencarian rute terpendek objek wisata di Kabupaten Temanggung ini memakai bahasa PHP, MySQL sebagai basis datanya serta mapbox bagi tampilan petanya. Adapun saran yang bisa	

					dikembangkan bagi sistem ini yakni harapannya bagi pengembang berikutnya, perlu pengembangan lagi agar lebih sempurna, melengkapi melaluipenambahan sistem navigasi dalam pencarian rute terpendeknya.	
8	Application Of The Dijkstra Algorithm To Determine The Shortest Route From City Center Surabaya To Historical Places	Pratiwi, Hanna	Algoritma Dijkstra	Menentukan Rute Terpendek Dari Pusat Kota Surabaya Ke Tempat Bersejarah	Dari penelitian dan hasil implementasi Algoritma Dijkstra berdasarkan lintasannya maka dapat diambil kesimpulan bahwa: 1. Didapatkan 5 rute terpendek yang dapat dilalui saat menuju ke tempat bersejarah dari awalnya adalah Stasiun Gubeng. 2. Permasalahan pencarian rute terpendek untuk mengunjungi tempat bersejarah dapat	Penelitian ini menggunakan pendekatan cellular automata dalam pencarian jalur terpendek dengan menggunakan algoritma Dijkstra

					diselesaikan dengan Algoritma Dijkstra. 3. Dengan menggunakan metode Dijkstra ini, dapat ditentukan jalur terpendek dari sebuah jalur perjalanan dengan menentukan vertex awal dan vertex tujuan serta membandingkan nilai dari masing-masing vertex.	
9	Implementasi Algoritma Dijkstra Dalam Pencarian Rute Terpendek Tempat Wisata Di Kabupaten Klaten	Nugroho Arif Sudibyo Permadi Eka Setyawan Yohana Putra Surya Rahmad Hidayat	Algoritma Dijkstra	Tempat Wisata Di Kabupaten Klaten	Penelitian yang telah dilakukan menemukan dan menyimpulkan sebagai berikut. Algoritma Dijkstra dapat digunakan dalam pencarian rute terpendek tempat wisata di Kabupaten Klaten. Dari hasil perhitungan yang secara manual dan yang menggunakan software Tora, diperoleh hasil yang sama.	Penelitian ini menggunakan pendekatan cellular automata dalam pencarian jalur terpendek dengan menggunakan algoritma Dijkstra

10	Perancangan Jalur Terpendek Evakuasi Bencana di Kawasan Boulevard Manado Menggunakan Algoritma Dijkstra	Aryando Giovani Rumondor Steven Ray Sentinuwo Alwin Melkie Sambul	Algoritma Dijkstra	Kawasan Boulevard Manado	Berdasarkan dari proses dan tahapan – tahapan yang dilakukan dalam penelitian ini, maka peneliti dapat menarik kesimpulan didapatkan jalur terpendek evakuasi yang akan dilalui pengunjung kawasan Boulevard Manado menuju zona aman evakuasi (*shelter),* serta total jarak dari lokasi rawan bencana menuju lokasi evakuasi, waktu tempuh dan petunjuk arah dengan menggunakan Algoritma Dijkstra.	Penelitian ini menggunakan pendekatan cellular automata dalam pencarian jalur terpendek dengan menggunakan algoritma Dijkstra

Dari berbagai penelitian sebelumnya yang menggunakan algoritma Dijkstra sebagai metode utama dalam pencarian jalur terpendek, terlihat bahwa algoritma ini telah diterapkan dalam berbagai kasus, seperti pencarian rute terpendek ke tempat wisata, kampus, jalur evakuasi bencana, hingga pengiriman barang. Hasil penelitian menunjukkan bahwa algoritma Dijkstra efektif dalam menentukan jalur optimal berdasarkan jarak terpendek, tetapi memiliki keterbatasan dalam mempertimbangkan faktor-faktor dinamis seperti perubahan kondisi jalan, kepadatan lalu lintas, atau hambatan lainnya.

Beberapa penelitian juga membandingkan algoritma Dijkstra dengan algoritma lain, seperti algoritma Greedy, untuk mengevaluasi efisiensi waktu komputasi dan hasil rute yang dihasilkan. Hasilnya menunjukkan bahwa meskipun algoritma Dijkstra memberikan jalur yang lebih optimal dalam hal jarak, algoritma lain kadang lebih efisien dalam hal waktu eksekusi.

Dari hasil analisis terhadap penelitian sebelumnya, saya mengusulkan pendekatan baru dengan mengintegrasikan Cellular Automata untuk mengoptimalkan algoritma Dijkstra dalam pencarian jalur terpendek. Cellular Automata, dengan kemampuannya dalam memodelkan perubahan lingkungan secara dinamis, dapat membantu algoritma Dijkstra dalam mempertimbangkan faktor tambahan seperti kemacetan, kondisi jalan, atau bahkan perubahan rute secara real-time.

Dengan pendekatan ini, penelitian saya bertujuan untuk meningkatkan efisiensi dan adaptabilitas algoritma Dijkstra, sehingga dapat digunakan dalam skenario yang lebih kompleks dan dinamis. Oleh karena itu, judul penelitian yang saya usulkan adalah:

“Pendekatan Cellular Automata dalam Optimalisasi Algoritma Dijkstra untuk Pencarian Jalur Terpendek”

.

BAB V

KESIMPULAN DAN SARAN

5.1.Kesimpulan

Berdasarkan hasil penelitian yang telah dilakukan, dapat disimpulkan bahwa Algoritma Dijkstra terbukti efektif dalam menentukan jalur terpendek pada jaringan jalan di Kabupaten Majene. Implementasi algoritma ini mampu secara optimal menghitung rute dengan bobot terkecil berdasarkan data yang diperoleh dari Google Maps. Selain itu, integrasi pendekatan *Cellular Automata* membantu dalam memvisualisasikan serta memahami pola jalur yang terbentuk secara lebih dinamis. Dengan membagi jaringan jalan menjadi sel-sel, *Cellular Automata* memungkinkan analisis lebih lanjut terhadap perubahan kondisi jalan secara real-time, seperti adanya hambatan atau perubahan lalu lintas. Namun, meskipun Algoritma Dijkstra bekerja secara optimal dalam kondisi ideal, algoritma ini memiliki keterbatasan dalam menghadapi perubahan jalur yang dinamis, seperti kemacetan atau kondisi darurat. Oleh karena itu, untuk situasi yang lebih kompleks, integrasi dengan algoritma lain atau metode tambahan diperlukan agar hasil lebih adaptif. Hasil pengujian juga menunjukkan bahwa jalur yang dihasilkan oleh Algoritma Dijkstra lebih efisien dibandingkan rute alternatif lainnya, di mana penggunaan bobot

berdasarkan jarak atau waktu tempuh memberikan hasil yang lebih akurat dalam menentukan jalur terbaik. Dari segi efisiensi waktu komputasi,

Algoritma Dijkstra mampu menyelesaikan perhitungan jalur terpendek dengan baik. Namun, dalam skenario dengan jumlah simpul yang sangat besar, waktu komputasi dapat meningkat sehingga diperlukan optimasi lebih lanjut untuk meningkatkan efisiensinya.

2.2.Saran

1. Pengembangan Algoritma untuk Kondisi Dinamis
   Untuk meningkatkan efisiensi dalam kondisi jalur yang berubah-ubah, penelitian selanjutnya dapat mengintegrasikan Algoritma Dijkstra dengan metode lain, seperti A* atau algoritma berbasis kecerdasan buatan, guna meningkatkan kemampuan adaptasi terhadap perubahan real-time.
2. Penggunaan Data Real-Time
   Implementasi sistem berbasis data real-time dari sensor lalu lintas atau GPS dapat meningkatkan akurasi pemilihan jalur. Dengan demikian, sistem dapat memberikan rute yang lebih optimal sesuai dengan kondisi jalan yang sebenarnya.
3. Pengujian pada Skala yang Lebih Luas
   Penelitian ini dapat diperluas dengan menerapkan metode yang sama pada jaringan jalan dengan cakupan wilayah yang lebih besar atau lingkungan dengan karakteristik berbeda, seperti perkotaan dengan tingkat kemacetan tinggi.
4. Optimasi Waktu Komputasi
   Untuk menangani skenario dengan jumlah simpul yang lebih besar, optimasi terhadap struktur data yang digunakan, seperti memanfaatkan Fibonacci Heap untuk priority queue dalam Algoritma Dijkstra, dapat diterapkan guna meningkatkan efisiensi waktu komputasi.

5. Integrasi dengan Sistem Navigasi dan Transportasi Cerdas.

   Hasil penelitian ini dapat diterapkan dalam sistem navigasi dan transportasi berbasis IoT dan kecerdasan buatan untuk meningkatkan efisiensi sistem transportasi, khususnya dalam manajemen jalur evakuasi atau optimasi rute logistik.

   Dengan adanya penelitian ini, diharapkan dapat memberikan kontribusi terhadap pengembangan sistem navigasi berbasis algoritma yang lebih adaptif dan efisien, serta membuka peluang penelitian lebih lanjut dalam bidang pemetaan jaringan jalan dan optimasi jalur

   .

DAFTAR PUSTAKA

Napiah, M., Astuti, R. D., & Mustofa, M. (2022). Implementasi algoritma Dijkstra menentukan jarak terdekat Jelambar–Kampus STMIK Nusa Mandiri Cengkareng. *Akrab Juara: Jurnal Ilmu-Ilmu Sosial*, 7(1), 80. https://doi.org/10.58487/akrabjuara.v7i1.1757

Nurfadila, R. A., Saputri, M. D., & Komarullah, H. (2024). Pencarian jalur evakuasi terdekat siaga tsunami Pantai Selatan Jember menggunakan algoritma Dijkstra. *As-Sunniyah: Jurnal Ilmiah Mahasiswa*, 4(2), 102–108. https://doi.org/10.62097/au.v6i01

Aos, A. N. A., & Putri, N. (2023). Dinamika vegetasi dan suhu permukaan lahan berbasis remote sensing di Waduk Jatigede Provinsi Jawa Barat: Studi pendahuluan. *Jurnal Geosains dan Remote Sensing*, 4(2), 67–76. https://doi.org/10.23960/jgrs.ft.unila.112

Safutra, N., Fole, A., & Herdianzah, Y. (2024). Perencanaan jalur evakuasi kebakaran yang efisien untuk fasilitas perawatan rumah sakit dengan menggunakan algoritma Dijkstra. Jurnal Rekayasa Sistem Industri, 9(2), 44–58.

Alharbi, M., Edwards, G., & Stocker, R. (2023). Reversible Quantum-Dot Cellular Automata-Based Arithmetic Logic Unit. *Nanomaterials*, *13*(17). https://doi.org/10.3390/nano13172445

Awal, E. E., Sitanggang, I. S., & Syaufina, L. (2023). Model Prediksi Perubahan Tutupan Lahan Pada Area Kebakaran Lahan Gambut Menggunakan Model Cellular Automata Markov. *Jurnal Informatika Dan Teknologi Informasi*, *1*(2). https://doi.org/10.56854/jt.v1i2.141

Christian Rufus, E., Rizkyaka Riyadi, R., Nugraha Hasibuan, D., Christian, E., Handrianus Pranatawijaya, V., & Nyaho Jl Yos Sudarso Palangka Raya, T. (2024). PENERAPAN ALGORITMA DIJKSTRA DALAM MENENTUKAN RUTE TERPENDEK UNTUK JASA PENGIRIMAN BARANG DI PALANGKA RAYA. In *Jurnal Mahasiswa Teknik Informatika* (Vol. 8, Issue 3).

Jalur, P., Evakuasi, T., Di, B., Boulevard, K., Menggunakan, M., Dijkstra, A., Giovani Rumondor, A., Sentinuwo, S. R., & Sambul, A. M. (n.d.). Aryando Giovani Rumondor. *Jurnal Teknik Informatika*, *14*(2).

Nirwana, H., Firgiawan, W., Cokrowibowo, S., & Zainuddin, Z. (2024). com Nirwana et al.: An Intelligent Optimazitation Method for Evacuation Route Planning in the Occurrence … An Intelligent Optimazitation Method f or Evacuation Route Planning in the Occurrence of Natural Disasters. *Technology & Applied Science Research*, *14*(6), 17696–17703. https://doi.org/10.48084/etasr.8538

Owusu, P. A., Leonenko, V. N., Mamchik, N. A., & Skorb, E. V. (2019). Modeling the growth of dendritic electroless silver colonies using hexagonal cellular automata. *Procedia Computer Science*, *156*, 43–48. https://doi.org/10.1016/j.procs.2019.08.128

Wiladi, M. (2023). *Penerapan Algoritma Dijkstra dan Algoritma Greedy Pada Optimasi Jalur Evakuasi Banjir*. *2*(1), 25–38. http://jurnal.fmipa.unmul.ac.id/index.php/Basis

Putra. (2024). Prediksi perubahan area pertambangan batubara di Kabupaten Muara Enim dan sekitarnya menggunakan cellular automata. *Jurnal Ilmiah Teknik dan Sains (JITS)*, 2(2), 52–56. https://doi.org/10.62278/jits.v2i2.42