Pencarian ATM Terdekat di ITB Ganesha dengan Algoritma *Uniform Cost Search* (UCS)

Wardatul Khoiroh - 13523001
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jalan Ganesha 10 Bandung
E-mail: wardatulkhoiroh11@gmail.com , 13523001@std.stei.itb.ac.id

Abstrak—penelitian ini berjudul Pencarian ATM Terdekat di Kampus ITB Ganesha Menggunakan Algoritma Uniform Cost Search (UCS) dengan tujuan mengkaji penerapan algoritma UCS dalam menentukan rute terdekat menuju fasilitas ATM dari berbagai bank di lingkungan kampus ITB. Harapannya, makalah ini dapat memberikan kontribusi dalam optimalisasi sistem navigasi kampus berbasis graf dan menjadi inspirasi bagi pengembangan aplikasi serupa di institusi pendidikan lainnya. Penelitian ini bertujuan untuk mengidentifikasi penerapan algoritma UCS dalam pencarian jalur dengan biaya minimum, membangun representasi graf dari lokasi-lokasi penting di kampus ITB, serta menganalisis efektivitas UCS dalam membantu pengguna menemukan ATM terdekat secara efisien. Metode yang digunakan dalam penelitian ini adalah pemodelan graf berbobot dari peta kampus ITB, di mana setiap simpul mewakili lokasi fisik seperti gedung atau fasilitas umum, dan setiap sisi memiliki bobot yang merepresentasikan estimasi jarak antar lokasi. Uniform Cost Search diterapkan dengan bantuan struktur data priority queue, yang memungkinkan algoritma menelusuri jalur paling ekonomis ke lokasi ATM yang dituju. Sistem dikembangkan menggunakan bahasa pemrograman Python dengan antarmuka grafis berbasis Tkinter. Hasil penelitian menunjukkan bahwa algoritma UCS efektif dalam mencari rute terdekat menuju ATM di berbagai titik kampus ITB, dengan memberikan hasil jalur yang optimal serta instruksi arah yang jelas. Pendekatan ini terbukti mampu meningkatkan efisiensi navigasi kampus, mengurangi waktu pencarian, serta memberikan pengalaman pengguna yang lebih baik melalui sistem pencarian berbasis logika graf dan algoritma pencarian jalur.

Kata Kunci—ATM, ITB Ganesha, pencarian jalur, Uniform Cost Search, navigasi kampus, Python, Tkinter, graf berbobot.

I. PENDAHULUAN

Kampus merupakan tempat yang memiliki beragam aktivitas akademik dan non-akademik, sehingga membutuhkan fasilitas pendukung yang memadai guna menunjang kebutuhan sivitas akademika. Salah satu fasilitas vital yang banyak dibutuhkan di lingkungan kampus adalah mesin Anjungan Tunai Mandiri (ATM) dari berbagai bank nasional. Keberadaan ATM memudahkan mahasiswa, dosen, dan staf dalam melakukan transaksi keuangan seperti penarikan tunai, transfer, pembayaran, dan pembelian tanpa harus meninggalkan area kampus. Namun, seiring luas dan kompleksnya tata letak gedung di kampus Institut Teknologi Bandung (ITB) Ganesha, banyak pengguna mengalami kesulitan dalam menemukan lokasi ATM terdekat dari tempat mereka berada.

Institut Teknologi Bandung (ITB) merupakan salah satu perguruan tinggi terbaik di Indonesia yang berlokasi di Jalan Ganesha No. 10, Bandung, Jawa Barat. Kampus ini memiliki tata ruang yang luas dan kompleks, dengan lebih dari 40 titik lokasi penting seperti gedung kuliah, laboratorium, fasilitas umum, serta ruang administrasi. Di antara titik-titik tersebut tersebar sejumlah ATM dari berbagai bank seperti BNI, BRI, Mandiri, BCA, Niaga, dan Bukopin. Namun, karena tidak semua pengguna mengenal letak pasti dari masing-masing ATM, maka dibutuhkan suatu sistem navigasi yang dapat secara otomatis memberikan rute tercepat dan termudah untuk mencapainya.

Untuk mengatasi permasalahan tersebut, dalam penelitian ini dibangun sebuah sistem pencari rute ke ATM terdekat berbasis algoritma Uniform Cost Search (UCS). UCS merupakan algoritma pencarian jalur dalam struktur graf yang memprioritaskan node dengan biaya akumulatif terkecil. Dengan menggunakan representasi graf berbobot dari lokasi-lokasi di ITB Ganesha, algoritma UCS akan menelusuri jalur termurah dari lokasi pengguna menuju titik ATM tujuan. Setiap lokasi dimodelkan sebagai simpul (node), dan jalur antar lokasi direpresentasikan sebagai sisi (edge) dengan bobot yang sama (asumsi jarak seragam). Implementasi sistem dilakukan dalam bahasa Python menggunakan pustaka Tkinter untuk tampilan antarmuka grafis, serta PIL untuk pengolahan gambar peta kampus.

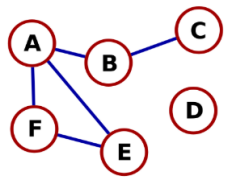
Dengan sistem ini, pengguna hanya perlu memasukkan nama lokasi awal dan nama bank (opsional), lalu sistem akan menampilkan jalur optimal ke ATM terdekat, termasuk instruksi arah langkah demi langkah. Diharapkan sistem ini dapat meningkatkan kenyamanan dan efisiensi aktivitas sivitas akademika ITB dalam mengakses layanan perbankan di lingkungan kampus.

II. DASAR TEORI

A. Graf Berbobot

Graf adalah suatu diagram yang memuat informasi tertentu jika diinterpretasikan secara tepat. Graf seringkali digunakan dalam kehidupan sehari-hari, biasanya digunakan untuk menggambarkan berbagai macam struktur yang ada sehingga

didapat suatu visualisasi objek-objeknya. Hal ini bertujuan untuk memudahkan dalam pemahaman. Beberapa contoh graf yang kita jumpai di kehidupan sehari-hari yaitu struktur organisasi, rangkaian Listrik, dan peta. Berikut salah satu contoh graf:

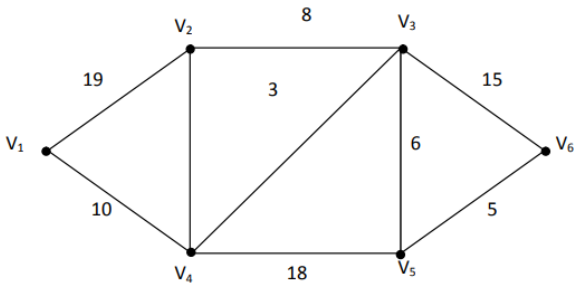


Gambar 2.1 Contoh Graf

Sumber: Wikipedia Graf

Graf terdiri dari beberapa simpul dan sisi, dimana dua simpul yang dihubungkan oleh sisi secara langsung disebut dengan bertetangga. Contohnya pada gambar 2.1, simpul A dan B adalah bertetangga, begitupun dengan simpul A-F dan A-E. Graf ada beberapa jenis pengelompokannya. Berdasarkan ada tidaknya gelang atau sisi ganda, graf digolongkan menjadi dua jenis yaitu graf sederhana dan graf tak sederhana. Graf sederhana adalah graf yang tidak mengandung gelang maupun sisi ganda. Sedangkan graf tak sederhana adalah graf yang mengandung sisi ganda atau gelang. Graf tak sederhana ini dibagi menjadi dua macam yaitu graf ganda dan graf semu. Suatu titik juga memiliki derajat yaitu jumlah sisi yang bersisian dengan titik/simpul tersebut. Contohnya pada gambar 2.1, simpul A memiliki derajat yaitu 3 diantaranya sisi A-B, sisi A-F, dan sisi A-E.

Graf juga dibedakan menjadi dua macam yaitu graf berbobot dan graf tidak berbobot. Graf berbobot adalah graf yang setiap sisinya diberikan sebuah nilai/harga/bobot dimana bobot pada tiap sisinya dapat berbeda bergantung pada masalah yang dimodelkan dengan graf tersebut. Bobot dapat dinyatakan jarak antara dua kota, biaya perjalanan antara dua kota, ataupun waktu tempuh antara dua kota.

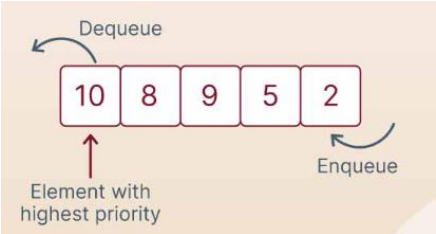


Gambar 2.2 Contoh Graf Berbobot

Sumber: PPT Rinaldi Munir

B. *Priority Queue*

Priority queue adalah salah satu struktur data dalm dunia komputer dan ilmu komputer. Struktur ini memiliki peran krusial dalam mengorganisir dan memanajemen data. *Priority queue* ini adalah struktur data khusus yang menyimpan elemen-elemen dengan nilai prioritas tertentu. Hal ini mirip dengan antrian pada umumnya, namun perbedaannya adalah tiap elemen memiliki nilai prioritas yang akan menentukan urutan elemen saat diambil sehingga elemen dengan nilai prioritas tertinggi akan diambil terlebih dahulu, disusul dengan elemen lainnya. Penggunaan struktur data ini biasanya ada dalam algoritma Dijkstra yaitu untuk menemukan jalur terpendek dalam graf berbobot positif. Selain itu, digunakan juga untuk algoritma A* untuk mencari jarak terpendek dengan heuristi, dan masih banyak penggunaan lainnya.



Gambar 2.3 *Priority Queue*

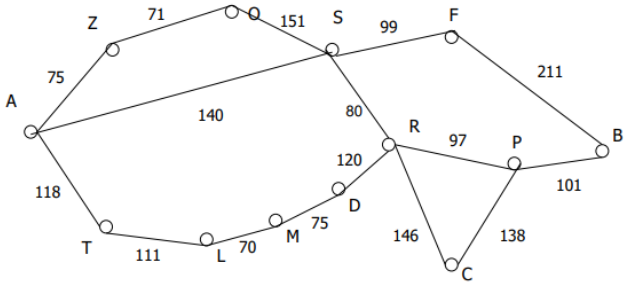
Sumber: shiksha online

Elemen pada gambar 2.3 tersusun rapi seperti halnya antrian. Elemen 10 memiliki prioritas tertinggi sehingga akan diambil terlebih dahulu dengan fungsi dequeue. Elemen yang terbaru akan dimasukkan dengan metode enqueue. Elemen selanjutnya yang akan diambil adalah elemen 9 karena memiiki prioritas kedua setelah elemen 10, bukan elemen 8 karena prioritasnya yang lebih kecil dan bukan berdasarkan urutan antrian masuk.

C. Algoritma *Uniform Cost Search* (UCS)

Uniform Cost Search (UCS) adalah algoritma pencarian jalur (pathfinding) yang digunakan untuk menemukan jalur dengan biaya total terendah dari titik awal ke titik tujuan pada sebuah graf berbobot. UCS bekerja dengan prinsip pencarian berbasis biaya kumulatif terendah. Artinya, algoritma selalu mengeksplorasi simpul (node) yang memiliki total biaya (cost) paling kecil dari simpul awal. Algoritma ini memanfaatkan struktur data Priority Queue untuk menjamin bahwa node dengan biaya terendah selalu dieksekusi terlebih dahulu. UCS merupakan varian dari algoritma Dijkstra, dan dijamin akan menemukan solusi optimal jika biaya edge (jarak antar node) bersifat non-negatif.

Langkah-langkah penyelesaian masalah dengan algoritma UCS yaitu dimulai dengan memasukkan simpul awal ke dalam priority queue dengan cost = 0. Selama queue tidak kosong, ambil simpul dengan cost terkecil dari queue (simpul yang sudah terbuka karena bertetangga dengan simpul sebelumnya yang sudah terbuka). Jika simpul tersebut adalah tujuan, maka pencariaan diselesaikan. Jika bukan tujuan, maka ekspansi simpil tersebut dengan menambahkan semua anak simpul ke dalam queue, biaya anak simpul = biaya simpul induk + jarak ke anak.



Gambar 2.4 Contoh Peta yang Diselesaikan

Sumber: PPT Rinaldi Munir

Dari gambar 2.4, sebelum menggunakan algoritma UCS, kita akan mengidentifikasikan simpul awal dan simpul yang dicari atau simpul tujuannya. Disini simpul awalnya yaitu simpul A dan simpul B sebagai simpul tujuan. Algoritma UCS ini nantinya akan memerika simpul yang bertetangga dengan simpul awal terlebih dahulu, lalu akan dibuat priority queue berdasarkan jarak terkecil (g(n)). Simpul selanjutnya yang diperiksa diambil dari elemen pertama pada priority queue dan dihapus dari priority queue tersebut setelah diambil. Skor yang didapat dari simpul tersebut adlah simpul total dari simpul awal hingga simpul tersebut, hal ini yang disebut dengan g(n). Simpul-simpul dalam priority queue akan diperiksa hingga mencapai simpul yang dicari. Ketika sudah ditemukan, maka priority queue akan dikosongkan. Berikut tabel proses pencarian algoritma UCS pada peta gambar 2.4 :

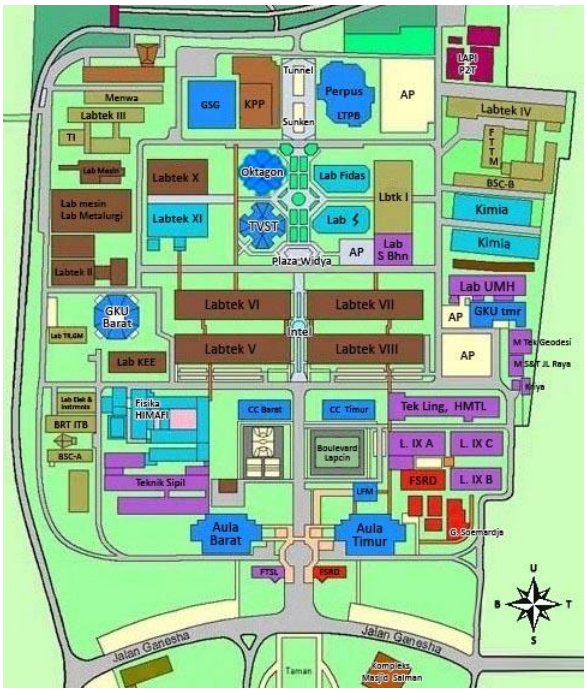
Simpul-E	Simpul Hidup
A	Z_{A-75}, T_{A-118}, S_{A-140}
Z_{A-75}	T_{A-118}, S_{A-140}, O_{AZ-146}
T_{A-118}	S_{A-140}, O_{AZ-146}, L_{AT-229}
S_{A-140}	O_{AZ-146}, R_{AS-220}, L_{AT-229}, F_{AS-239}, O_{AS-291}
O_{AZ-146}	R_{AS-220}, L_{AT-229}, F_{AS-239}, O_{AS-291}
R_{AS-220}	L_{AT-229}, F_{AS-239}, O_{AS-291}, $P_{ASR-317}$, $D_{ASR-340}$, $C_{ASR-366}$
L_{AT-229}	F_{AS-239}, O_{AS-291}, $M_{ATL-299}$, $P_{ASR-317}$, $D_{ASR-340}$, $C_{ASR-366}$
F_{AS-239}	O_{AS-291}, $M_{ATL-299}$, $P_{ASR-317}$, $D_{ASR-340}$, $C_{ASR-366}$, $B_{ASF-450}$
O_{AS-291}	$M_{ATL-299}$, $P_{ASR-317}$, $D_{ASR-340}$, $C_{ASR-366}$, $B_{ASF-450}$
$M_{ATL-299}$	$P_{ASR-317}$, $D_{ASR-340}$, $D_{ATLM-364}$, $C_{ASR-366}$, $B_{ASF-450}$
$P_{ASR-317}$	$D_{ASR-340}$, $D_{ATLM-364}$, $C_{ASR-366}$, $B_{ASRP-418}$, $C_{ASRP-455}$, $B_{ASF-450}$
$D_{ASR-340}$	$D_{ATLM-364}$, $C_{ASR-366}$, $B_{ASRP-418}$, $C_{ASRP-455}$, $B_{ASF-450}$
$D_{ATLM-364}$	$C_{ASR-366}$, $B_{ASRP-418}$, $C_{ASRP-455}$, $B_{ASF-450}$
$C_{ASR-366}$	$B_{ASRP-418}$, $C_{ASRP-455}$, $B_{ASF-450}$
$B_{ASRP-418}$	Solusi ketemu

Gambar 2.5 Tabel Proses Pencarian Algoritma UCS

Sumber: PPT Rinaldi Munir

D. Peta Institut Teknologi Bandung

Institut Teknologi Bandung (ITB) merupakan salah satu perguruan tinggi negeri terkemuka di Indonesia yang fokus pada pendidikan, penelitian, dan pengabdian masyarakat dalam bidang sains, teknologi, seni, dan humaniora. ITB didirikan pada 3 Juli 1920 dengan nama awal Technische Hoogeschool te Bandoeng (THB) dan menjadi kampus teknik pertama di Indonesia. Lokasi utama ITB berada di Jl. Ganesha No. 10, Bandung, Jawa Barat. Kawasan ini dikenal sebagai Kampus ITB Ganesha dan menjadi pusat kegiatan akademik dan administratif utama. Peta kampus merupakan representasi visual dari tata letak dan posisi berbagai gedung, laboratorium, jalur akses, serta fasilitas pendukung di lingkungan ITB Ganesha. Peta ini sangat penting untuk mahasiswa baru dalam mengenali lokasi kelas dan gedung tujuan. Selain itu, sangat penting untuk pengunjung agar tidak tersesat di lingkungan kampus yang komplek serta sebagai aplikasi navigasi misalnya untuk mencari fasilitas tertentu seperti ATM, kantin, ruang kuliah umum, atau kantor layanan akademik. Berikut merupakan peta ITB Ganesha dan keterangannya :



Gambar 2.6 Peta ITB Ganesha dan Keterangannya

Sumber: ITB Ganesha Map

III. PEMBAHASAN

Algoritma Uniform Cost Search (UCS) digunakan untuk mencari jalur terpendek dari suatu lokasi di dalam kampus ITB Ganesha menuju lokasi ATM terdekat. Kampus ITB Ganesha memiliki banyak titik lokasi penting seperti gedung perkuliahan, laboratorium, aula, hingga pusat layanan mahasiswa. Beberapa lokasi juga memiliki fasilitas ATM dari berbagai bank seperti BNI, BRI, Bukopin, BCA, Mandiri, dan Niaga. Setiap ATM yang ada di ITB memiliki fitur tambahan yaitu ATM Bersama sehingga bank apapun dapat menggunakan ATM tersebut dengan biaya admin. Untuk

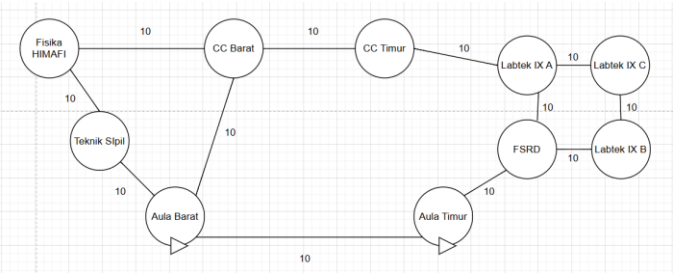
mencari jalur terdekat menuju ATM, lokasi awal pengguna dijadikan sebagai simpul awal. Sementara itu, lokasi-lokasi ATM dijadikan sebagai simpul tujuan (goal nodes). Proses pencarian akan menghitung jalur dengan total jarak (cost) terkecil dari lokasi pengguna ke ATM terdekat menggunakan struktur data priority queue untuk memprioritaskan simpul dengan cost terendah yang belum dikunjungi. Sebagai contoh, jika pengguna berada di Labtek V – PIKSI, Teknik Informatika, maka algoritma UCS akan menelusuri graf rute kampus hingga menemukan ATM terdekat, misalnya ATM BNI GKU Timur, ATM BRI East Hall, atau ATM Mandiri Gerbang Depan, tergantung dari letak dan keterhubungan antar simpul di graf.

A. Penentuan Simpul dan Rute

Simpul dalam graf ini dibuat berdasarkan lokasi strategis dan pertemuan jalur di area kampus ITB. Setiap simpul mewakili sebuah lokasi, seperti gedung atau area terbuka yang memiliki koneksi dengan simpul lain melalui jalan, lorong, atau jalur pedestrian. Sebagai contoh: "GKU Timur" terhubung ke "CC Timur", "East Hall", dan "ATM BNI GKU Timur". "Aula Barat" terhubung ke "ATM BNI Aula Barat" dan "Teknik Sipil". "Gerbang Depan" terhubung ke "ATM Mandiri" dan "ATM BCA". Untuk setiap pasangan simpul yang terhubung, diberikan bobot jarak yang diestimasi secara seragam (misalnya 10 meter), atau berdasarkan estimasi jarak sebenarnya jika tersedia. Arah jalan satu arah diperhatikan untuk menjaga validitas rute, seperti hanya memungkinkan jalan dari "East Hall" ke "ATM BRI", namun bukan sebaliknya.

B. Visualisasi Graf dan Jalur

Graf yang digunakan divisualisasikan dalam bentuk peta ITB Ganesha, di mana simpul-simpul ATM ditandai secara eksplisit dan jalur koneksi antar lokasi diilustrasikan dalam bentuk panah atau garis. Setiap node ATM ditandai sebagai node terminal. Jalur pencarian akan mengikuti struktur graf dan prioritas jarak (cost) terkecil hingga mencapai ATM terdekat. Berikut Sebagian visualisasi graf dari peta ITB Ganesha:



Gambar 3.1 Visualisasi Graf

Simpul menyatakan Lokasi Gedung/bangunan di ITB Ganesha ditandai dengan lingkaran dan ada namanya. Tiap sisi ada bobot dan diasumsikan memiliki bobot yang sama yaitu 10 (kedepannya bisa dikembangkan dengan jarak sesungguhnya). Tanda segitiga menandakan di Gedung tersebut ada ATM.

C. Proses Algoritma UCS

Dari gambar 3.1, didapat jalur dan titik atau simpul tujuan ada dua, dimana akan dipilih ATM mana yang terdekat dari posisi pengguna sekarang. Apabila pengguna berada di CC Barat, maka berikut alur algoritma UCS yang terjadi:

Simpul-E	Simpul Hidup
CC Barat	Fisika HIMAFI$_{\text{CC Barat-10}}$, Aula Barat$_{\text{CC Barat-10}}$, CC Timur$_{\text{CC Barat-10}}$
Fisika HIMAFI$_{\text{CC Barat-10}}$	Aula Barat$_{\text{CC Barat-10}}$, CC Timur$_{\text{CC Barat-10}}$, Teknik Sipil$_{\text{CC Barat Fisika HIMAFI-20}}$
Aula Barat$_{\text{CC Barat-10}}$	Solusi ketemu

Pada pencarian kali ini, Solusi ditemukan lebih cepat karena jarak tiap Gedung dianggap sama, sehingga pengurutan di awal simpul hidup juga menentukan pencarian. Bisa saja dikemudian hari, Solusi yang ditemukan meminta pengguna ke ATM yang tidak paling dekat karena jarak yang sama tersebut. Jadi, saat proses pencarian dan simpul-E sudah mencapai gedung yang berisi ATM/tempat ATM maka pencarian akan berhenti dan itulah solusi terdekatnya.Alur algoritma UCS yang dihasilkan dapat berbeda-beda, namun hasil akhirnya jalur dan ATM terdekat akan tetap sama. Berikut contoh alur algoritma UCS lainnya:

Simpul-E	Simpul Hidup
CC Barat	CC Timur$_{\text{CC Barat-10,}}$ Fisika HIMAFI$_{\text{CC Barat-10}}$, Aula Barat$_{\text{CC Barat-10}}$
CC Timur$_{\text{CC Barat-10}}$	Fisika HIMAFI$_{\text{CC Barat-10}}$, Aula Barat$_{\text{CC Barat-10}}$, Labtek IX A$_{\text{CC Barat CC Timur-20}}$
Fisika HIMAFI$_{\text{CC Barat-10}}$	Aula Barat$_{\text{CC Barat-10}}$, Labtek IX A$_{\text{CC Barat CC Timur-20}}$, Teknik Sipil$_{\text{CC Barat Fisika HIMAFI-20}}$
Aula Barat$_{\text{CC Barat-10}}$	Solusi ketemu

IV. IMPLEMENTASI PROGRAM

Program yang dikembangkan bertujuan untuk mempermudah pengguna dalam mencari lokasi ATM terdekat di lingkungan kampus ITB Ganesha, berdasarkan posisi awal pengguna serta bank yang diinginkan. Sistem ini dibangun menggunakan bahasa Python dengan antarmuka grafis yang dibuat menggunakan pustaka Tkinter, serta visualisasi peta menggunakan modul PIL (Python Imaging Library). Fungsi utama program adalah mengimplementasikan algoritma Uniform Cost Search (UCS) untuk menelusuri graf lokasi kampus dan mencari jalur terpendek ke node ATM terdekat.

A. Struktur Data

Struktur data pertama yang digunakan adalah dictionary nama_lokasi, yang memetakan nomor identitas lokasi dengan nama tempat sebenarnya di ITB. Walaupun struktur ini tidak digunakan langsung dalam pencarian, daftar ini dapat dimanfaatkan untuk validasi atau pengembangan input numerik di masa depan. Struktur data ini diambil berdasarkan peta ITB Ganesha yang saya ambil dari ITB Ganesha Map. Berikut struktur data dari nama_lokasi:

```
# ====================== DATA LOKASI RESMI ======================
nama_lokasi = {
    "1": "FTSP",
    "2": "Aula Barat",
    "3": "Teknik Sipil",
    "4": "Fisika",
    "5": "RSG",
    "6": "FSRD",
    "7": "Aula Timur",
    "8": "LFM/9009",
    "9": "Semi Murni, Desain",
    "10": "Labtek IXB - Teknik Arsitektur",        You, 7 hours ago • fea
    "11": "Labtek IXC - Teknik Geodesi, Teknik Lingkungan",
    "12": "Labtek IXA - Teknik Planologi",
    "13": "Teknik Lingkungan",
    "14": "Teknik Geodesi",
    "15": "GKU Timur",
    "16": "Labtek VIII - FIMPA, Teknik Elektro, UPT Bahasa",
    "17": "Labtek V - PIKSI, Teknik Informatika",
    "18": "Labtek VI - Farmasi, MKDU",
    "19": "Labtek VII - Teknik Fisika, Teknik Kelautan",
    "20": "CC Timur",
    "21": "CC Barat",
    "22": "Plaza Widya",
    "23": "Labtek I - Teknik Mesin",
    "24": "Labtek XI - Biologi, Geofisika, Meteorologi",
    "25": "Labtek X - Teknik Kimia, Teknik Mineral",
    "26": "Oktagon",
    "27": "TVST",
    "28": "TPB",
    "29": "Kerjasama PLN - ITB",
    "30": "Labtek I - Teknik Geofisika, Lab Struktur",
    "31": "Kimia",
    "32": "FTM, Teknik Geologi, Teknik Pertambangan",
    "33": "Pusat Penelitian Energi, LAPI",
    "34": "Perpustakaan Pusat",
    "35": "GSU",
    "36": "Labtek III",
    "37": "Labtek II",
    "38": "Labtek I FTI, Teknik Industri, Matematika, Astronomi, PPPPM",
    "39": "Pariwisata",
    "40": "ATM BNI GKU Timur",
    "41": "SARAGA"
}
```

Gambar 4.1 Struktur Data nama_lokasi
Sumber: Dokumen Penulis

Struktur data utama adalah graph, yaitu representasi graf dari peta kampus ITB dalam bentuk dictionary adjacency list. Setiap kunci adalah sebuah lokasi (node), dan nilainya adalah daftar lokasi yang dapat dijangkau langsung dari node tersebut. Secara default, hubungan dalam graf dibuat satu arah, sehingga fungsi make_bidirectional() digunakan untuk menambahkan koneksi dua arah agar rute pencarian dapat dilakukan secara fleksibel, mencerminkan kenyataan bahwa jalur pejalan kaki di kampus dapat ditempuh dua arah. Berikut struktur data utamanya :

```
# ====================== GRAF ======================
graph = {
    "FTSP": ["Aula Barat"],
    "Aula Barat": ["Teknik Sipil", "RSG", "ATM BNI Aula Barat"],
    "Teknik Sipil": ["Fisika"],
    "Fisika": ["RSG"],
    "RSG": ["FSRD", "Aula Timur"],
    "FSRD": ["Aula Timur"],
    "Aula Timur": ["LFM/9009"],
    "LFM/9009": ["Semi Murni, Desain"],
    "Semi Murni, Desain": ["Labtek IXB - Teknik Arsitektur"],
    "Labtek IXB - Teknik Arsitektur": ["Labtek IXC - Teknik Geodesi, Teknik Lingkungan"],
    "Labtek IXC - Teknik Geodesi, Teknik Lingkungan": ["Labtek IXA - Teknik Planologi"],
    "Labtek IXA - Teknik Planologi": ["Teknik Lingkungan"],
    "Teknik Lingkungan": ["Teknik Geodesi"],
    "Teknik Geodesi": ["GKU Timur"],
    "GKU Timur": ["ATM BNI GKU Timur", "East Hall", "CC Timur"],
    "East Hall": ["ATM BRI", "ATM Bukopin"],
    "Labtek VIII - FIMPA, Teknik Elektro, UPT Bahasa": ["Labtek VI - Farmasi, MKDU"],
    "Labtek V - PIKSI, Teknik Informatika": ["Labtek VI - Farmasi, MKDU"],
    "Labtek VI - Farmasi, MKDU": ["Labtek VII - Teknik Fisika, Teknik Kelautan"],
    "Labtek VII - Teknik Fisika, Teknik Kelautan": ["Plaza Widya"],
    "CC Timur": ["Plaza Widya"],
    "CC Barat": ["Plaza Widya"],
    "Plaza Widya": ["GKU Timur", "Labtek V - PIKSI, Teknik Informatika"],
    "Labtek I - Teknik Mesin": ["Labtek XI - Biologi, Geofisika, Meteorologi"],
    "Labtek XI - Biologi, Geofisika, Meteorologi": ["Labtek X - Teknik Kimia, Teknik Mineral"],
    "Labtek X - Teknik Kimia, Teknik Mineral": ["Oktagon"],
    "Oktagon": ["TVST"],
    "TVST": ["TPB"],
    "TPB": ["Kerjasama PLN - ITB"],
    "Kerjasama PLN - ITB": ["Labtek I - Teknik Geofisika, Lab Struktur"],
    "Labtek I - Teknik Geofisika, Lab Struktur": ["Kimia"],
    "Kimia": ["FTM, Teknik Geologi, Teknik Pertambangan"],
    "FTM, Teknik Geologi, Teknik Pertambangan": ["Pusat Penelitian Energi, LAPI"],
    "Pusat Penelitian Energi, LAPI": ["Perpustakaan Pusat"],
    "Perpustakaan Pusat": ["GSU"],
    "GSU": ["Labtek III"],
    "Labtek III": ["Labtek II"],
    "Labtek II": ["Labtek I FTI, Teknik Industri, Matematika, Astronomi, PPPPM"],
    "Labtek I FTI, Teknik Industri, Matematika, Astronomi, PPPPM": ["Pariwisata"],
    "Pariwisata": ["SARAGA"],
    "GKU Barat": ["ATM Niaga"],
    "Gerbang Depan": ["ATM Mandiri", "ATM BCA"],
    "ATM BNI Aula Barat": [],
    "ATM BNI GKU Timur": [],
    "ATM BRI": [],
    "ATM Bukopin": [],
    "ATM Niaga": [],
    "ATM Mandiri": [],
    "ATM BCA": [],
    "SARAGA": []
}
def make_bidirectional(graph):
    bidir_graph = {}
    for node in graph:
        for neighbor in graph[node]:
            bidir_graph.setdefault(node, []).append(neighbor)
            bidir_graph.setdefault(neighbor, []).append(node)
    return bidir_graph

graph = make_bidirectional(graph)
```

Gambar 4.2 Struktur Data Graf
Sumber: Dokumen Penulis

Selain graf, program juga menyimpan data lokasi ATM berdasarkan bank dalam dictionary atm_nodes_by_bank. Ini memudahkan pengguna yang hanya ingin mencari ATM dari bank tertentu seperti BNI, BCA, Mandiri, dan sebagainya. Jika pengguna tidak memilih bank, maka sistem akan menggunakan semua node ATM yang telah digabungkan dalam list all_atm_nodes. Berikut struktur data lokasi ATM:

```
# ===================== ATM NODES =====================
atm_nodes_by_bank = {
    "BNI": ["ATM BNI Aula Barat", "ATM BNI GKU Timur"],
    "NIAGA": ["ATM Niaga"],
    "BRI": ["ATM BRI"],
    "BUKOPIN": ["ATM Bukopin"],
    "MANDIRI": ["ATM Mandiri"],
    "BCA": ["ATM BCA"]
}
all_atm_nodes = [atm for lst in atm_nodes_by_bank.values() for atm in lst]
```

Gambar 4.3 Struktur Data Lokasi ATM
Sumber: Dokumen Penulis

B. Implementasi UCS

Fungsi ucs(graph, start, goals) merupakan implementasi dari algoritma Uniform Cost Search (UCS) yang digunakan untuk mencari jalur terpendek berdasarkan total biaya dari simpul awal menuju salah satu simpul tujuan yang ditentukan. Pada fungsi ini, pencarian dilakukan dengan pendekatan UCS berbasis biaya terkecil, menggunakan struktur data priority queue yang diimplementasikan melalui pustaka heapq. Priority queue diinisialisasi dengan satu elemen berupa tuple (cost, node, path) yang berarti simpul awal dimasukkan dengan biaya 0 dan jalur masih kosong. Selama queue belum kosong, algoritma akan mengambil elemen dengan total biaya terendah, kemudian memeriksa apakah simpul tersebut adalah tujuan. Jika ya, maka jalur tersebut langsung dikembalikan beserta total biayanya. Jika bukan, simpul tersebut akan ditandai sebagai telah dikunjungi dan diperluas ke semua tetangganya yang belum dikunjungi. Setiap tetangga kemudian dimasukkan ke dalam antrian dengan biaya baru yang merupakan penjumlahan dari biaya saat ini dengan cost antar node (dalam kasus ini, setiap edge dianggap memiliki cost tetap yaitu 10 satuan). Berikut *source code* dari fungsi ucs:

```
# ===================== UCS =====================
def ucs(graph, start, goals):
    queue = [(0, start, [])]
    visited = set()
    while queue:
        cost, node, path = heapq.heappop(queue)
        if node in visited:
            continue
        visited.add(node)
        path = path + [node]
        if node in goals:
            return path, cost
        for neighbor in graph.get(node, []):      You, 7 hours ago
            if neighbor not in visited:
                heapq.heappush(queue, (cost + 10, neighbor, path))
    return None, float('inf')
```

Gambar 4.4 Fungsi ucs

Sumber: Dokumen Penulis

C. Antarmuka Pengguna (GUI)

Pada bagian antarmuka grafis (GUI) dibangun menggunakan pustaka tkinter, seluruh elemen dirancang untuk memungkinkan pengguna mencari ATM terdekat di lingkungan ITB Ganesha secara interaktif. Setiap fungsi dan elemen antarmuka memiliki peran tersendiri yang saling terintegrasi untuk mendukung pencarian jalur menggunakan algoritma Uniform Cost Search (UCS).

Fungsi root = tk.Tk() merupakan titik awal pembuatan jendela utama aplikasi. Di sini, judul jendela ditentukan dengan root.title(...) dan ukuran jendela diset ke resolusi 1100x700 piksel melalui root.geometry(...). Jendela ini menjadi wadah utama dari seluruh komponen yang ditampilkan.

Bagian frame_left dan frame_right masing-masing bertindak sebagai wadah tata letak vertikal dan horizontal untuk mengelompokkan elemen GUI. frame_left digunakan untuk menampilkan peta kampus dalam bentuk gambar, sedangkan frame_right digunakan untuk menampilkan kolom input, hasil pencarian, dan tombol aksi. Berikut kode dari interface awalnya:

```
# ===================== GUI =====================
root = tk.Tk()
root.title("Pencarian ATM Terdekat ITB Ganesha")
root.geometry("1100x700")

frame_left = tk.Frame(root)
frame_left.pack(side=tk.LEFT, padx=10, pady=10)

frame_right = tk.Frame(root)
frame_right.pack(side=tk.TOP, anchor='n', padx=20, pady=20)
```

Gambar 4.5 Fungsi root dan frame
Sumber: Dokumen Penulis

Bagian pemrosesan gambar (img = Image.open(...)) berfungsi untuk memuat file gambar peta ITB (peta_itb.png) dan menampilkannya di sisi kiri jendela aplikasi. Jika file gambar tidak ditemukan, akan ditampilkan pesan teks peringatan agar pengguna menyimpan gambar dengan nama yang benar. Gambar ditampilkan menggunakan ImageTk.PhotoImage dari pustaka PIL (Pillow), dan ditempatkan ke dalam label label_img.

Komponen input terdiri dari dua bagian yaitu masukan lokasi dan masukan nama bank. Keduanya menggunakan tk.Entry sebagai kotak teks, dan diberi label menggunakan tk.Label. Lokasi merupakan simpul awal dalam graf pencarian, sedangkan bank adalah filter untuk memilih jenis ATM yang ingin dicari. Input bank bersifat opsional sehingga jika dikosongkan, pencarian akan dilakukan terhadap semua ATM yang tersedia.

Fungsi arahkan(path) bertanggung jawab untuk menghasilkan petunjuk langkah demi langkah dari hasil pencarian jalur. Fungsi ini mengiterasi isi daftar path, dan menyusun instruksi teks sederhana yang menggambarkan pergerakan antar simpul, misalnya: “Dari GKU Timur ke ATM BNI GKU Timur (ikuti papan arah/lurus/belok)”. Hasilnya akan ditampilkan dalam bagian hasil pencarian.

Fungsi cari() adalah fungsi utama yang dieksekusi saat tombol “Cari ATM Terdekat” ditekan. Fungsi ini mengambil input dari pengguna, baik lokasi maupun nama bank. Pertama-

tama, dilakukan validasi apakah lokasi yang dimasukkan benar-benar ada di dalam graf. Jika tidak ditemukan, akan muncul peringatan menggunakan messagebox.showerror. Setelah validasi, fungsi akan mencari jalur terpendek dari lokasi awal ke ATM tujuan menggunakan fungsi ucs(...). Hasil pencarian akan ditampilkan dalam format jalur, jarak tempuh (dalam satuan meter), dan petunjuk langkah. Jika jalur tidak ditemukan, akan ditampilkan pesan bahwa pencarian gagal. Berikut merupakan *source code* utama dari GUI:

```
# Input
tk.Label(frame_right, text="Masukkan Lokasi Anda:", font=("Helvetica", 12, "bold")).pack(anchor="w")
entry_lokasi = tk.Entry(frame_right, font=("Helvetica", 12))
entry_lokasi.pack(fill='x')

tk.Label(frame_right, text="Masukkan Bank (BNI, BRI, MANDIRI, BCA, BUKOPIN, NIAGA) atau kosong:", font=("Helvetica", 12,
entry_bank = tk.Entry(frame_right, font=("Helvetica", 12))
entry_bank.pack(fill='x')

frame_result = tk.LabelFrame(frame_right, text="Hasil Pencarian", font=("Helvetica", 12, "bold"), padx=10, pady=10)
frame_result.pack(fill='both', expand=True, pady=20)

output = tk.Label(frame_result, text="", justify="left", font=("Helvetica", 11), anchor="w", wraplength=450)
output.pack(anchor="w")

# Petunjuk langkah
def arahkan(path):
    instruksi = []
    for i in range(1, len(path)):
        instruksi.append(f"- Dari {path[i-1]} ke {path[i]} (ikuti papan arah/lurus/belok)")
    return "\n".join(instruksi)
```

Gambar 4.6 Fungsi Utama GUI

Sumber: Dokumen Penulis

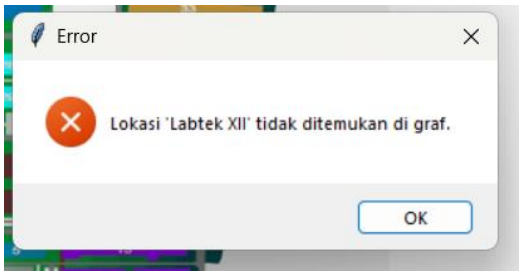
Akhirnya, tombol aksi tk.Button(...) dengan label “Cari ATM Terdekat” dipasang untuk menjalankan fungsi cari() saat ditekan. Proses pencarian ini merupakan pemicu utama yang menghubungkan seluruh alur input, proses, dan output aplikasi.

D. Cara Menjalankan Program dan *Testing*

Untuk menjalankan program pencarian ATM terdekat di Kampus ITB Ganesha, pengguna harus memastikan bahwa Python 3 telah terinstal di perangkatnya. Selain itu, library eksternal seperti Pillow perlu dipasang terlebih dahulu menggunakan perintah pip install pillow jika belum tersedia. Setelah itu, pengguna cukup menjalankan file program .py menggunakan perintah python3 pencarian-atm-itb.py melalui terminal atau IDE seperti VSCode, PyCharm, atau IDLE. Setelah program berjalan, akan muncul antarmuka grafis (GUI) yang menampilkan peta kampus di sisi kiri dan form input di sisi kanan. Pengguna hanya perlu mengisi lokasi awal (misalnya “GKU Timur”, “Aula Barat”, dll.) dan nama bank (opsional, seperti “BNI”, “BCA”, atau dikosongkan untuk mencari semua ATM), kemudian klik tombol “Cari ATM Terdekat”. Program akan menampilkan hasil berupa jalur tercepat menuju ATM, total jarak tempuh, serta instruksi langkah demi langkah untuk mencapai tujuan. Jika lokasi tidak valid, program akan menampilkan pesan kesalahan. Proses pencarian dilakukan secara otomatis menggunakan algoritma Uniform Cost Search dan hasilnya ditampilkan langsung dalam antarmuka. Berikut beberapa tampilan GUI *testing*:

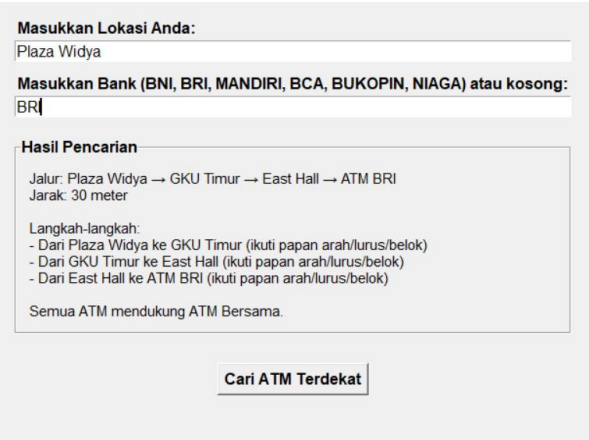
Gambar 4.7 Tampilan Awal

Sumber: Dokumen Penulis



Gambar 4.8 Tampilan Error Lokasi Tidak Valid

Sumber: Dokumen Penulis



Gambar 4.9 Tampilan Hasil

Sumber: Dokumen Penulis

V. KESIMPULAN

Penerapan algoritma Uniform Cost Search (UCS) dalam pencarian ATM terdekat di kawasan ITB Ganesha terbukti efektif untuk menentukan jalur dengan jarak terpendek dari lokasi pengguna menuju lokasi ATM tujuan. Dengan merepresentasikan area kampus dalam bentuk graf berbobot, algoritma UCS dapat melakukan eksplorasi node secara sistematis berdasarkan akumulasi cost terkecil, sehingga memastikan bahwa hasil pencarian merupakan rute paling efisien. Implementasi sistem menggunakan struktur data priority queue memungkinkan algoritma mengakses node

prioritas dengan optimal, sedangkan visualisasi berbasis antarmuka GUI (Tkinter) memberikan kemudahan bagi pengguna dalam menginput lokasi dan memilih bank tujuan. Berdasarkan hasil pengujian, sistem dapat menampilkan urutan jalur yang harus dilalui, estimasi jarak tempuh, serta instruksi navigasi yang informatif. Dengan demikian, dapat disimpulkan bahwa algoritma UCS sangat cocok untuk digunakan dalam sistem pencarian rute optimal, khususnya di lingkungan yang kompleks seperti kampus ITB. Sistem ini juga berpotensi dikembangkan lebih lanjut dengan integrasi peta dinamis atau data koordinat real-time guna meningkatkan akurasi dan fleksibilitas dalam berbagai skenario penggunaan di dunia nyata.

LINK VIDEO YOUTUBE

https://youtu.be/lD0UT2BqW_s?si=ze4reyS8aHZXr0Jm

LINK REPOSITORY GITHUB

https://github.com/wrdtlkhoir/Makalah_STIMA.git

UCAPAN TERIMA KASIH

Puji syukur saya panjatkan kepada Tuhan Yang Maha Esa atas segala rahmat dan karunia-Nya, sehingga saya dapat menyelesaikan makalah ini dengan baik. Saya juga ingin mengucapkan terima kasih kepada pihak-pihak yang telah memberikan dukungan, bantuan, dan motivasi selama proses penulisan makalah ini. Ucapan terima kasih saya sampaikan kepada:

1. Dr. Nur Ulfa Maulidevi, S.T, M.Sc. selaku dosen pengajar yang telah memberikan arahan dalam mengerjakan makalah ini serta masukan yang sangat berharga.
2. Muhammad Su'udi dan Siti Aminah selaku orang tua yang selalu memberikan doa, semangat, dan dukungan moral.
3. Teman-teman yang turut memberikan ide, diskusi, dan motivasi dalam menyelesaikan makalah ini.
4. Semua pihak yang tidak bisa saya sebutkan satu per satu, yang telah membantu baik secara langsung maupun tidak langsung.

Semoga makalah ini dapat memberikan manfaat dan kontribusi yang positif bagi pembaca. Saya menyadari bahwa makalah ini masih jauh dari sempurna, oleh karena itu saran dan kritik yang membangun sangat saya harapkan.

REFERENSI

[1] Munir, Rinaldi. (2025). Penentuan Rute (Bagian 1). 20 Juni 2024, dari https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2024-2025/21-Route-Planning-(2025)-Bagian1.pdf.

[2] Munir, Rinaldi. (2025). Penentuan Rute (Bagian 1). 20 Juni 2024, dari https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2024-2025/22-Route-Planning-(2025)-Bagian2.pdf.

[3] Dündar, P., Güner, M., & Çolakoğlu, Ö. (2012). An approach to the modular line balancing problem for an apparel product manufacturing with graph theory. Journal of Textile & Apparel / Tekstil ve Konfeksiyon, 22(4).

PERNYATAAN

Dengan ini saya menyatakan bahwa makalah yang saya tulis ini adalah tulisan saya sendiri, bukan saduran, atau terjemahan dari makalah orang lain, dan bukan plagiasi.

Bandung, 24 Juni 2025

Wardatul Khoiroh
13523001