

Proyek Akhir Mata Kuliah Struktur Data

Topik: *Implementasi dan Visualisasi Algoritma Shortest Path pada Peta Kampus di Yogyakarta*

1. Tujuan Proyek

Proyek akhir ini dirancang agar mahasiswa:

- a. Memahami prinsip kerja berbagai algoritma *shortest path* pada graf.
- b. Mampu mengimplementasikan algoritma tersebut secara efisien menggunakan bahasa pemrograman yang telah ditentukan.
- c. Mampu menganalisis kelebihan, kekurangan, dan kompleksitas masing-masing algoritma.
- d. Menampilkan hasil proses dan visualisasi jalur terpendek antar kampus di Yogyakarta.

2. Studi Kasus: Peta Kampus di Yogyakarta

Gunakan graf berbobot yang merepresentasikan jarak antar kampus berikut (dalam km):

Kode Kampus

- | | |
|---|---|
| 0 | Universitas Gadjah Mada (UGM) |
| 1 | Universitas Negeri Yogyakarta (UNY) |
| 2 | Universitas Pembangunan Nasional “Veteran” Yogyakarta (UPN) |
| 3 | Universitas Islam Indonesia (UII) |
| 4 | Universitas Ahmad Dahlan (UAD) |
| 5 | Universitas Sanata Dharma (USD) |
| 6 | Universitas Muhammadiyah Yogyakarta (UMY) |
| 7 | Universitas Atma Jaya Yogyakarta (UAJY) |
| 8 | Institut Seni Indonesia (ISI) |
| 9 | Universitas Islam Negeri (UIN) Sunan Kalijaga |

Setiap edge mewakili jarak antar kampus (dapat berupa perkiraan realistik), misalnya: UGM–UNY (2 km), UNY–UPN (4 km), UGM–UIN (3 km), dan seterusnya.

3. Daftar Algoritma Shortest Path

Setiap mahasiswa akan mendapatkan satu algoritma berbeda dan bahasa pemrograman berbeda (C, C++, R, Python), untuk menghindari duplikasi proyek.

Algoritma <i>Shortest Path</i>	Deskripsi Singkat
Dijkstra's Algorithm	Menentukan jalur terpendek dari satu simpul ke semua simpul lain pada graf berbobot positif.
Bellman-Ford Algorithm	Menentukan jalur terpendek dari satu simpul, mendukung bobot negatif.
Floyd-Warshall Algorithm	Menentukan jarak terpendek antar semua pasangan simpul.
Johnson's Algorithm	Kombinasi Bellman-Ford & Dijkstra untuk <i>all-pairs shortest path</i> dengan efisiensi tinggi.
A (A-Star) Algorithm*	Jalur terpendek berbasis <i>heuristic</i> (estimasi jarak ke tujuan).
Bidirectional Dijkstra	Menjalankan Dijkstra dari dua arah (awal dan tujuan) agar lebih cepat.
Yen's K-Shortest Algorithm	Paths Menemukan lebih dari satu jalur terpendek (<i>k shortest paths</i>).
CH (Contraction Hierarchies)	Optimasi Dijkstra untuk graf besar (seperti peta jalan nyata).
BFS Shortest Path (Unweighted Graph)	Jalur terpendek pada graf tak berbobot (menggunakan BFS).
DAG Shortest (Topological)	Path Jalur terpendek pada graf berarah tanpa siklus (Directed Acyclic Graph).

4. Spesifikasi Teknis Proyek

a. Input Data:

- Node: daftar kampus di Yogyakarta.
- Edge: jarak antar kampus (bobot).
- Simpul asal dan tujuan (misal: UGM ke UMY).

b. Proses:

- Implementasikan algoritma shortest path sesuai kelompok.
 - Tampilkan proses langkah demi langkah (misalnya: pemilihan simpul, pembaruan jarak, dan jalur sementara).
 - Visualisasikan hasil akhir dalam bentuk teks atau graf (menggunakan ASCII, Graphviz, atau pustaka visualisasi).
- c. Output:
- Jalur terpendek (misalnya: UGM → UNY → UPN → UAD).
 - Total jarak minimum.
 - Log proses per iterasi (contoh: simpul yang sedang diproses dan jarak sementara).
 - Gambar atau diagram graf hasil jalur akhir.

5. Sistematika Laporan Proyek

Laporan dibuat dalam format PDF dengan struktur sebagai berikut:

BAB I – PENDAHULUAN

- Latar belakang dan tujuan proyek
- Rumusan masalah dan batasan masalah
- Manfaat proyek

BAB II – DASAR TEORI

- Konsep dasar graf dan shortest path
- Penjelasan algoritma yang diimplementasikan
- Analisis kompleksitas waktu dan ruang

BAB III – PERANCANGAN PROYEK

- Struktur data graf (array, list, adjacency matrix/list)
- Flowchart dan pseudocode algoritma
- Representasi data kampus (node dan bobot)

BAB IV – IMPLEMENTASI DAN HASIL

- Potongan kode utama
- Output hasil eksekusi berupa jalur terpendek
- Penjelasan hasil per iterasi
- Visualisasi graf dan jalur terpendek

BAB V – PENUTUP

- Kesimpulan
- Saran

Lampiran

- Kode program lengkap
- Dataset jarak antar kampus

6. Kriteria Penilaian

Kriteria	Bobot
Pemahaman konsep algoritma	20%
Implementasi algoritma (<i>code correctness</i>)	30%
Persentasi	15%
Laporan tertulis dan analisis	35%

7. Contoh Kasus (Ilustrasi Sederhana)

Misalnya, mahasiswa (Dijkstra's Algorithm) membuat graf seperti ini:

Kampus:

- 0: UGM
- 1: UNY
- 2: UPN
- 3: UAD
- 4: UMY

a. Langkah Input Program

Mahasiswa menjalankan program, lalu program menanyakan jumlah simpul dan nama kampus, kemudian meminta jarak antar setiap pasangan kampus satu per satu seperti ini:

b. Contoh Jalannya Program:

- Masukkan jumlah kampus: 5

- Masukkan nama kampus ke-0: UGM
- Masukkan nama kampus ke-1: UNY
- Masukkan nama kampus ke-2: UPN

- Masukkan nama kampus ke-3: UAD
 - Masukkan nama kampus ke-4: UMY
- Masukkan jarak antar kampus (km):
- Jika tidak terhubung, masukkan 0.
- Jarak dari UGM ke UNY: 2
 - Jarak dari UGM ke UPN: 6
 - Jarak dari UGM ke UAD: 0
 - Jarak dari UGM ke UMY: 0
- Jarak dari UNY ke UPN: 4
 - Jarak dari UNY ke UAD: 0
 - Jarak dari UNY ke UMY: 0
- Jarak dari UPN ke UAD: 3
 - Jarak dari UPN ke UMY: 0
- Jarak dari UAD ke UMY: 5
- Karena graf tidak berarah, maka jarak UGM–UNY otomatis sama dengan UNY–UGM. Program akan membuat matriks jarak secara simetris berdasarkan input ini.
- c. Input Node Asal dan Tujuan
- Setelah data selesai dimasukkan, program menanyakan:
- Masukkan kampus asal: UGM
 - Masukkan kampus tujuan: UMY
- d. Proses di Balik Layar (Langkah Dijkstra):
- Program kemudian:
- Membentuk matriks adjacency berdasarkan input.
 - Menjalankan algoritma Dijkstra dari UGM (asal).
 - Mengupdate jarak ke setiap simpul.
 - Menentukan jalur terpendek ke UMY (tujuan).

e. Hasil Output

Hasil Algoritma Dijkstra

Kampus asal : UGM

Kampus tujuan : UMY

Jalur terpendek: UGM -> UNY -> UPN -> UAD -> UMY

Total jarak: 14 km
