Penerapan Algoritma Greedy dalam Penyelesaian Masalah Minimum Spanning Tree untuk Mengoptimalisasi Jaringan Komunikasi

Mohammad Andhika Fadillah - 13522128
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jalan Ganesha 10 Bandung
E-mail (gmail): : 13522128@std.stei.itb.ac.id

***Abstract*— Jaringan komunikasi yang efisien sangat penting untuk memastikan transmisi data yang cepat dan handal, terutama dalam dunia yang semakin terhubung secara digital. Salah satu pendekatan untuk mengoptimalkan jaringan komunikasi adalah dengan membangun Minimum Spanning Tree (MST), yang menghubungkan semua simpul dalam jaringan dengan biaya total minimum dan tanpa membentuk siklus. Penerapan MST dalam jaringan memungkinkan penurunan biaya pemasangan dan pemeliharaan infrastruktur jaringan secara signifikan. Algoritma Greedy, khususnya Algoritma Prim dan Kruskal, sangat efektif dalam menyelesaikan masalah ini karena keduanya dapat menemukan solusi optimal dengan cepat dan efisien. Studi kasus pada jaringan komunikasi skala besar, seperti pada kampus, akan menunjukkan penerapan praktis algoritma ini dan hasil yang dicapai, termasuk pengurangan panjang total kabel dan peningkatan efisiensi jaringan yang signifikan, yang akhirnya dapat berkontribusi pada penurunan biaya operasional dan peningkatan kualitas layanan jaringan.**

Keywords— Jaringan Komunikasi, Transmisi Data, Minimum Spanning Tree (MST), Algoritma Greedy, Algoritma Prim, Algoritma Kruskal, Optimasi Jaringan

I. PENDAHULUAN

1.1 Latar Belakang

Dalam era digitalisasi yang semakin berkembang pesat, jaringan komunikasi yang efisien dan andal menjadi tulang punggung bagi beragam aktivitas dan layanan. Jaringan komunikasi memungkinkan pertukaran data yang cepat dan handal, baik dalam skala lokal, nasional, maupun internasional. Keberadaan jaringan yang efisien mendukung berbagai sektor, mulai dari bisnis, pemerintahan, pendidikan, hingga kehidupan sehari-hari individu. Ketika jaringan mengalami gangguan atau efisiensinya menurun, dampaknya dapat dirasakan secara luas, mulai dari penurunan produktivitas, gangguan layanan, hingga kerugian finansial yang signifikan.

Seiring dengan meningkatnya permintaan akan layanan data yang cepat dan andal, tantangan dalam merancang dan mengelola jaringan komunikasi juga semakin kompleks. Salah satu tantangan utama adalah bagaimana membangun jaringan yang tidak hanya handal dan cepat, tetapi juga hemat biaya, baik dalam hal instalasi maupun pemeliharaan. Di sinilah konsep Minimum Spanning Tree (MST) memainkan peran penting. MST adalah sebuah struktur graf yang menghubungkan semua simpul dalam jaringan dengan biaya total minimum dan tanpa membentuk siklus. Dalam konteks jaringan komunikasi, MST membantu mengurangi panjang total kabel atau biaya transmisi antar simpul, yang pada akhirnya berkontribusi pada penghematan biaya operasional dan peningkatan efisiensi.

1.2 Masalah Penelitian

Masalah utama dalam optimasi jaringan komunikasi adalah bagaimana memastikan bahwa semua titik dalam jaringan terhubung dengan biaya minimum. Dalam banyak kasus, jaringan yang tidak dioptimalkan dapat mengakibatkan penggunaan sumber daya yang berlebihan, seperti panjang kabel yang tidak diperlukan atau pemakaian bandwidth yang tidak efisien. Untuk mengatasi masalah ini, algoritma berbasis Greedy seperti Algoritma Prim dan Kruskal sering digunakan karena kemampuan mereka untuk menemukan solusi optimal dengan cepat.

1.3 Ruang Lingkup

Ruang lingkup penelitian ini mencakup beberapa aspek penting dari penerapan Algoritma Greedy dalam konteks optimasi jaringan komunikasi. Pertama, penelitian akan mengkaji karakteristik jaringan komunikasi yang membutuhkan optimasi, termasuk faktor-faktor seperti topologi jaringan, jumlah simpul, dan distribusi beban lalu lintas. Kedua, penelitian akan membahas implementasi Algoritma Prim dan Kruskal menggunakan bahasa pemrograman Python, dengan fokus pada efisiensi algoritma dalam hal waktu dan ruang. Ketiga, penelitian akan melakukan pengujian kinerja kedua algoritma pada berbagai ukuran dan tipe jaringan, termasuk jaringan acak, jaringan grid, dan jaringan dengan topologi tertentu yang umum digunakan dalam jaringan komunikasi.

Selain itu, penelitian ini juga akan melakukan studi kasus pada jaringan komunikasi kampus untuk mengilustrasikan penerapan praktis dari MST dan hasil yang dicapai dalam konteks dunia nyata. Studi kasus ini akan melibatkan analisis jaringan yang ada, penerapan Algoritma Prim dan Kruskal, dan evaluasi hasil optimasi

dalam hal pengurangan panjang total kabel, biaya operasional, dan peningkatan efisiensi jaringan.

1.4 Metode Penelitian

Metode penelitian yang digunakan meliputi beberapa tahapan kunci. Pertama, representasi jaringan komunikasi sebagai graf berbobot akan dilakukan, di mana simpul merepresentasikan titik-titik koneksi seperti gedung atau perangkat jaringan, dan tepi merepresentasikan jalur kabel atau saluran komunikasi. Selanjutnya, Algoritma Prim dan Kruskal akan diimplementasikan dalam Python, menggunakan struktur data yang efisien seperti heap untuk Prim dan union-find untuk Kruskal. Pengujian kinerja akan dilakukan dengan mengukur waktu eksekusi dan penggunaan memori dari kedua algoritma pada berbagai skenario jaringan. Studi kasus pada jaringan komunikasi kampus akan digunakan untuk mengilustrasikan penerapan praktis dari MST, dengan fokus pada pengurangan biaya dan peningkatan efisiensi jaringan.

1.5 Manfaat Penelitian

Penelitian ini diharapkan dapat memberikan kontribusi signifikan dalam bidang optimasi jaringan komunikasi, dengan hasil yang dapat diterapkan secara praktis oleh perancang dan manajer jaringan. Dengan pemahaman yang lebih mendalam tentang kinerja Algoritma Prim dan Kruskal, diharapkan para praktisi dapat memilih metode yang paling sesuai untuk membangun MST dalam konteks yang berbeda-beda, sehingga dapat mencapai optimasi jaringan yang efektif dan efisien. Selain itu, hasil studi kasus diharapkan dapat memberikan bukti empiris tentang bagaimana penerapan MST dapat mengurangi panjang total kabel dan biaya operasional, serta meningkatkan kualitas layanan jaringan secara keseluruhan. Penelitian ini juga diharapkan dapat membuka peluang untuk pengembangan lebih lanjut dalam optimasi jaringan komunikasi melalui pendekatan algoritma lainnya yang mungkin lebih cocok untuk skenario tertentu.

II. LANDASAN TEORI

A. *Algoritma Greedy*

Algoritma Greedy merupakan salah satu metode dalam mencari solusi pada masalah optimasi, di mana keputusan diambil berdasarkan pilihan terbaik yang tersedia pada setiap langkahnya. Pendekatan ini tidak mempertimbangkan implikasi jangka panjang dan hanya fokus pada pencapaian keuntungan segera. Prinsip dasar dari Algoritma Greedy adalah mengambil langkah terbaik yang ada saat ini tanpa memperhitungkan hasil di masa depan.

Skema umum algoritma greedy :

```
function greedy(C : himpunan_kandidat) → himpunan_solusi
{ Mengembalikan solusi dari persoalan optimasi dengan algoritma greedy }
Deklarasi
  x : kandidat
  S : himpunan_solusi

Algoritma:
  S ← {}      { inisialisasi S dengan kosong }
  while (not SOLUSI(S)) and (C ≠ {} ) do
      x ← SELEKSI(C)     { pilih sebuah kandidat dari C}
      C ← C – {x}        { buang x dari C karena sudah dipilih }
      if LAYAK(S ∪ {x}) then      { x memenuhi kelayakan untuk dimasukkan ke dalam himpunan solusi }
          S ← S ∪ {x}      { masukkan x ke dalam himpunan solusi }
      endif
  endwhile
  {SOLUSI(S) or C = {} }

  if SOLUSI(S) then     { solusi sudah lengkap }
     return S
  else
    write('tidak ada solusi')
  endif
```

Gambar 2.1 Skema umum algoritma greedy

Elemen-elemen Algoritma Greedy :

1. Himpunan Kandidat
   Himpunan kandidat adalah himpunan yang berisi solusi yang mungkin akan dipilih pada setiap langkah.

2. Himpunan Solusi
   Himpunan solusi berisi solusi yang sudah terpilih.

3. Fungsi Seleksi
   Fungsi untuk memilih kandidat berdasarkan strategi greedy tertentu.

4. Fungsi Solusi
   Fungsi untuk menentukan apakah kandidat yang dipilih sudah memberikan solusi.

5. Fungsi Kelayakan
   Fungsi yang digunakan untuk memeriksa apakah kandidata yang dipilih dapat dimasukkan ke dalam hipunan solusi.

6. Fungsi Obyektif.
   Fungsi obyektif merupakan fungsi untuk memaksimumkan atau meminimumkan (optimasi).

B. *Minimum Spanning Tree*

Minimum Spanning Tree (MST) adalah konsep penting dalam teori graf yang berkaitan dengan pemodelan hubungan antar entitas dalam sebuah jaringan. MST adalah subgraf dari graf tertentu yang menghubungkan semua simpul (node) dalam graf tersebut dengan tepat satu jalur, tanpa membentuk siklus, dan memiliki berat total yang minimal.

C. *Algoritma Prim*

Algoritma Prim adalah salah satu algoritma greedy yang digunakan untuk mencari Minimum Spanning Tree (MST) dalam sebuah graf berbobot.

Berikut adalah Langkah – Langkah terkait Algoritma Prim:

- Langkah 1: ambil sisi dari graf G yang berbobot minimum, masukkan ke dalam T.
- Langkah 2: pilih sisi (u, v) yang mempunyai bobot minimum dan bersisian dengan simpul di T, tetapi (u, v) tidak membentuk sirkuit di T. Masukkan (u, v) ke dalam T.
- Langkah 3: ulangi langkah 2 sebanyak n – 2 kali.

Skema umum Algoritma Prim:

```
procedure Prim(input G : graf, output T : pohon)
{ Membentuk pohon merentang minimum T dari graf berbobot G.
Masukan: graf-berbobot terhubung G = (V, E), dengan |V| = n
Luaran: pohon rentang minimum T = (V, E') }

Deklarasi
  i, p, q, u, v : integer

Algoritma
  Cari sisi (p, q) dari E yang berbobot terkecil
  T ← {(p, q)}
  for i←1 to n – 2 do
    Pilih sisi (u, v) dari E yang bobotnya terkecil namun bersisian dengan simpul di T
    T ← T ∪ {(u, v)}
  endfor
```

Gambar 2.2 Skema umum algoritma prim

Komplesitas algoritma prim: O(n2)

D. Algoritma Kruskal

Algoritma Kruskal adalah algoritma greedy yang digunakan dalam teori graf untuk mencari Minimum Spanning Tree (MST), yang merupakan subgraf dari graf berbobot yang menghubungkan semua simpul dengan total bobot minimal.

Berikut adalah Langkah – Langkah terkait algoritma kruskal :

- Langkah 0: sisi-sisi dari graf sudah diurut menaik berdasarkan bobotnya – dari bobot kecil ke bobot besar)
- Langkah 1: T masih kosong
- Langkah 2: pilih sisi (u, v) dengan bobot minimum yang tidak membentuk sirkuit di T. Tambahkan (u, v) ke dalam T.
- Langkah 3: ulangi langkah 2 sebanyak n – 1 kali.

Skema umum Algoritma Prim:

```
procedure Kruskal(input G : graf, output T : pohon)
{ Membentuk pohon merentang minimum T dari graf berbobot G.
Masukan: graf-berbobot terhubung G = (V, E), dengan |V| = n
Luaran: pohon rentang minimum T = (V, E') }

Deklarasi
  i, u, v : integer

Algoritma
  { Asumsi: sisi-sisi dari graf sudah diurut menaik berdasarkan bobotnya dari kecil ke besar}
  T ← {}
  while jumlah sisi di dalam T < n – 1 do
    Pilih sisi (u, v) dari E yang bobotnya terkecil
    if (u, v) tidak membentuk sirkuit di T then
      T ← T ∪ {(u, v)}
    endif
  endfor
```

Gambar 2.3 Skema umum algoritma kruskal

III. Aplikasi Algoritma Greedy

Identifikasi Elemen Algoritma Greedy:

1. Himpunan Kandidat
   Himpunan kandidat adalah kumpulan elemen yang tersedia dan dapat dipilih sebagai bagian dari solusi pada setiap langkah iterasi algoritma greedy.

   - Himpunan Kandidat (Algoritma Prim): Semua simpul dalam graf yang belum menjadi bagian dari MST.

   - Himpunan Kandidat (Algoritma Kruskal): Semua sisi dalam graf yang dapat ditambahkan ke MST tanpa membentuk siklus.

   Untuk Algoritma Prim, himpunan kandidat adalah simpul yang berdekatan dengan simpul yang sudah menjadi bagian dari MST tetapi belum ditambahkan ke MST. Untuk Algoritma Kruskal, himpunan kandidat adalah semua sisi yang belum dipertimbangkan untuk ditambahkan ke MST, diurutkan berdasarkan bobotnya.

2. Himpunan Solusi
   Himpunan solusi adalah kumpulan elemen yang membentuk solusi akhir yang optimal.

   - Himpunan solusi (Algoritma Prim): Sekumpulan simpul dan sisi yang membentuk MST pada setiap tahap iterasi.
   - Himpunan solusi (Algoritma Kruskal): Sekumpulan sisi yang dipilih yang membentuk MST yang menghubungkan semua simpul tanpa siklus.

   Himpunan solusi dalam konteks MST adalah jaringan minimum yang menghubungkan semua simpul tanpa membentuk siklus, dan dengan total bobot minimum.

3. Fungsi Seleksi
   Fungsi seleksi adalah aturan atau kriteria yang digunakan untuk memilih elemen terbaik dari himpunan kandidat pada setiap langkah iterasi.

   - Fungsi Seleksi (Algoritma Prim): Memilih simpul yang berdekatan dengan MST yang ada dan memiliki bobot sisi terkecil.
   - Fungsi Seleksi (Algoritma Kruskal): Memilih sisi dengan bobot terkecil dari himpunan kandidat yang belum dipilih.

   Untuk Algoritma Prim, fungsi seleksi memilih simpul terdekat (dengan bobot sisi terkecil) dari simpul yang sudah menjadi bagian dari MST. Sedangkan untuk

Algoritma Kruskal, fungsi seleksi memilih sisi dengan bobot terkecil yang tidak membentuk siklus jika ditambahkan ke MST.

4. Fungsi Solusi
   Fungsi solusi adalah kriteria yang digunakan untuk menentukan apakah suatu himpunan elemen sudah membentuk solusi yang valid atau final.

   - Fungsi Solusi (Algoritma Prim): MST terbentuk jika semua simpul sudah ditambahkan ke MST dan tidak ada lagi simpul yang dapat dipilih.
   - Fungsi Solusi (Algoritma Kruskal): MST terbentuk jika semua simpul sudah terhubung dan jumlah sisi yang dipilih adalah n−1 (dimana n adalah jumlah simpul).

   MST dianggap selesai ketika semua simpul dalam graf terhubung tanpa siklus dan dengan total bobot minimum. Pada tahap ini, tidak ada lagi elemen yang perlu dipilih atau ditambahkan.

5. Fungsi Kelayakan
   Fungsi kelayakan adalah kriteria atau aturan yang digunakan untuk memeriksa apakah elemen yang dipilih dapat ditambahkan ke himpunan solusi tanpa melanggar aturan yang berlaku.

   - Fungsi Kelayakan (Algoritma Prim): Sisi yang dipilih harus menghubungkan simpul yang sudah dalam MST dengan simpul yang belum dalam MST.
   - Fungsi Kelayakan (Algoritma Kruskal): Sisi yang dipilih tidak boleh membentuk siklus jika ditambahkan ke himpunan solusi.

   Untuk Algoritma Prim, fungsi kelayakan memastikan bahwa simpul yang dipilih adalah simpul yang valid untuk ditambahkan ke MST. Untuk Algoritma Kruskal, fungsi kelayakan memastikan bahwa sisi yang dipilih tidak membentuk siklus dalam MST yang sedang dibangun.

6. Fungsi Obyektif
   Fungsi obyektif adalah kriteria yang digunakan untuk mengukur kualitas solusi dan menentukan tujuan dari algoritma tersebut.

   - Fungsi Obyektif: Meminimalkan total bobot atau panjang dari semua sisi dalam MST yang menghubungkan semua simpul.

   Fungsi obyektif dalam konteks MST adalah meminimalkan total biaya atau panjang kabel yang diperlukan untuk menghubungkan semua titik dalam jaringan tanpa membentuk siklus. Solusi yang optimal adalah yang memiliki total bobot terkecil.

IV. IMPLEMENTASI

```
class DisjointSet:
    def __init__(self, vertices):
        self.parent = {v: v for v in vertices}
        self.rank = {v: 0 for v in vertices}

    def find(self, vertex):
        if self.parent[vertex] != vertex:
            self.parent[vertex] = self.find(self.parent[vertex])
        return self.parent[vertex]

    def union(self, u, v):
        root_u = self.find(u)
        root_v = self.find(v)

        if root_u != root_v:
            if self.rank[root_u] > self.rank[root_v]:
                self.parent[root_v] = root_u
            elif self.rank[root_u] < self.rank[root_v]:
                self.parent[root_u] = root_v
            else:
                self.parent[root_v] = root_u
                self.rank[root_u] += 1
```

```
def kruskal_mst(graph):
    edges = [(weight, u, v) for u in graph for v, weight in graph[u]]
    edges.sort()
    mst = []
    ds = DisjointSet(graph.keys())

    for weight, u, v in edges:
        if ds.find(u) != ds.find(v):
            ds.union(u, v)
            mst.append((u, v, weight))

    return mst
```

```
def visualize_graph_and_mst(graph, mst):
    # Buat graph NetworkX dari input graph
    G = nx.Graph()
    for u in graph:
        for v, weight in graph[u]:
            G.add_edge(u, v, weight=weight)

    # Buat graph untuk MST
    MST = nx.Graph()
    for u, v, weight in mst:
        MST.add_edge(u, v, weight=weight)

    # Posisi node untuk kedua graph
    pos = nx.spring_layout(G)

    # Plot graph original
    plt.figure(figsize=(9, 4))
```

```
    plt.subplot(121)

    nx.draw(G, pos, with_labels=True, node_color='lightblue',
node_size=700, font_size=15, font_weight='bold')
    edge_labels = nx.get_edge_attributes(G, 'weight')
    nx.draw_networkx_edge_labels(G, pos,
edge_labels=edge_labels, font_size=15)
    plt.title("Graph Asli")

    # Plot MST
    plt.subplot(122)
    nx.draw(MST, pos, with_labels=True,
node_color='lightgreen', node_size=700, font_size=15,
font_weight='bold')
    mst_edge_labels = nx.get_edge_attributes(MST, 'weight')
    nx.draw_networkx_edge_labels(MST, pos,
edge_labels=mst_edge_labels, font_size=15)
    plt.title("Minimum Spanning Tree (MST)")

    plt.show()
```

```
def prim_mst(graph, start_node):
    mst = []
    visited = set()
    min_heap = [(0, start_node, None)]

    while min_heap:
        cost, current_node, previous_node = heapq.heappop(min_heap)
        if current_node not in visited:
            visited.add(current_node)
            if previous_node is not None:
                mst.append((previous_node, current_node, cost))

            for neighbor, weight in graph[current_node]:
                if neighbor not in visited:
                    heapq.heappush(min_heap, (weight, neighbor,
current_node))

    return mst
```

V. PENGUJIAN DAN ANALISIS

Studi kasus menggunakan wilayah ITB Jatinangor. Jarak didapatkan melalui pengukuran dari aplikasi google maps.

A. Pengujian

Titik Awal	Titik Akhir	Jarak
GKU 3	GKU 2	300
	GKU 1	200
	ASRAMA TB 5	280
	ASRAMA TB 1	150

Titik Awal	Titik Akhir	Jarak
GKU 2	GKU 3	300
	GKU 1	90
	ASRAMA TB 5	170
	ASRAMA TB 1	350

Titik Awal	Titik Akhir	Jarak
GKU 1	GKU 3	200
	GKU 2	90
	ASRAMA TB 5	200
	ASRAMA TB 1	260

Titik Awal	Titik Akhir	Jarak
Asrama TB5	GKU 3	280
	GKU 2	170
	GKU 1	200
	ASRAMA TB 1	200

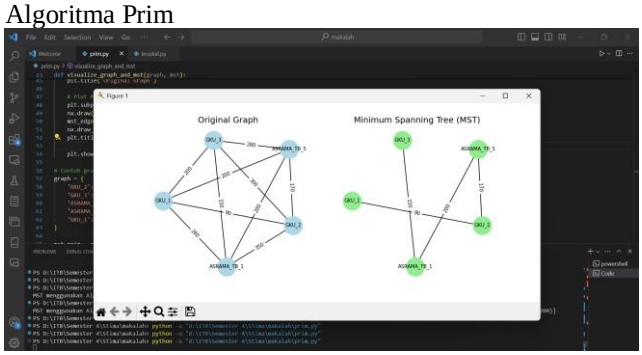
Titik Awal	Titik Akhir	Jarak
Asrama TB1	GKU 3	150
	GKU 2	350
	GKU 1	260
	ASRAMA TB 5	200

Berikut Implementasi graf yang berisi jarak dari titik awal ke titik akhir sesuai pada tabel :

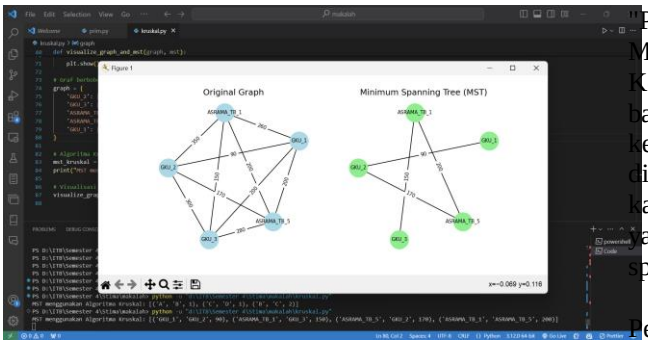
```
graph = {
    'GKU_2': [('GKU_3', 300), ('GKU_1', 90), ('ASRAMA_TB_5', 170),
('ASRAMA_TB_1', 350)],
    'GKU_3': [('GKU_2', 300), ('GKU_1', 200), ('ASRAMA_TB_5', 280),
('ASRAMA_TB_1', 150)],
    'ASRAMA_TB_5': [('GKU_3', 280), ('GKU_2', 170), ('GKU_1', 200),
('ASRAMA_TB_1', 200)],
    'ASRAMA_TB_1': [('GKU_3', 150), ('GKU_2', 350), ('GKU_1', 260),
('ASRAMA_TB_5', 200)],
    'GKU_1': [('GKU_3', 200), ('GKU_2', 90), ('ASRAMA_TB_5', 200),
('ASRAMA_TB_1', 260)]
}
```

Hasil yang didapatkan yaitu sebagai berikut :

1. Algoritma Prim



2. Algoritma Kruskal



B. Analisis

Algoritma Kruskal sangat efisien pada graf yang memiliki banyak simpul tetapi sedikit tepi karena fokus pada pemilihan tepi yang paling ringan dan memastikan tidak ada siklus. Hasil MST dari Kruskal memastikan total bobot minimum yang mungkin, menjadikannya solusi optimal untuk masalah MST.

Algoritma Prim lebih efisien untuk graf yang padat karena setiap simpul terhubung ke banyak simpul lainnya dan algoritma fokus pada simpul yang sudah ada dalam MST.

VI. KESIMPULAN

Sesuai dengan percobaan yang telah dilakukan, didapat Kesimpulan bahwa algoritma kruskal dan Algoritma Prim dalam persoalan minimum spanning tree menunjukkan pendekatan greedy yang efektif untuk membangun MST. Meskipun menggunakan strategi yang berbeda—Kruskal berfokus pada pemilihan tepi dengan bobot terkecil dari seluruh graf, sedangkan Prim berfokus pada penambahan tepi terdekat dari simpul yang sudah ada dalam MST—keduanya memastikan hasil akhir berupa graf yang menghubungkan semua simpul dengan bobot total minimum dan tanpa siklus. Pilihan antara kedua algoritma tergantung pada karakteristik graf yang sedang dianalisis serta kebutuhan spesifik dalam hal kinerja dan kompleksitas.

VIDEO LINK AT YOUTUBE

https://youtu.be/IaCPTbkB8GM

UCAPAN TERIMA KASIH

Penulis ingin mengucapkan rasa syukur kepada Tuhan yang Maha Esa karena atas rahmat dan anugerah-Nya, penulis dapat menyelesaikan makalah Mata Kuliah IF2211 yang berjudul "Penerapan Algoritma Greedy dalam Penyelesaian Masalah Minimum Spanning Tree untuk Mengoptimalisasi Jaringan Komunikasi". Dengan penuh kesyukuran, penulis menyadari bahwa tanpa petunjuk dan rahmat dari-Nya, segala usaha dan kerja keras ini tidak akan membuahkan hasil seperti yang diharapkan. Selain itu, penulis juga ingin mengucapkan terima kasih yang sebesar-besarnya kepada kedua orang tua tercinta yang telah memberikan dukungan moral, material, dan spiritual selama penulis menyusun makalah ini.

Penulis juga ingin menyampaikan penghargaan yang tinggi kepada dosen pengajar Mata Kuliah IF2211, beserta dosen pengampu lainnya yang telah memberikan bimbingan, ilmu, dan arahan yang sangat berharga selama masa perkuliahan.

REFERENSI

[1] Munir, Rinaldi. Algoritma Greedy (Bagian 1). Diakses melalui https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2020-2021/Algoritma-Greedy-(2021)-Bag1.pdf pada 10 Juni 2024 pukul 20.10 WIB

[2] Munir, Rinaldi. Algoritma Greedy (Bagian 2). Diakses melalui https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2020-2021/Algoritma-Greedy-(2021)-Bag2.pdf pada 10 Juni 2024 pukul 20.30 WIB

[3] Munir, Rinaldi. Algoritma Greedy (Bagian 3). Diakses melalui https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2021-2022/Algoritma-Greedy-(2022)-Bag3.pdf pada 10 Juni 2024 pukul 20.55 WIB

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, 12 Juni 2024

Mohammad Andhika Fadillah 13522128