TUBES: Desain dan Analisis Algoritma
(CSH2G3) genap 19/20
DOSEN: PHN
KELAS: IF-42-04

Penentuan Rute Wisata Menggunakan Greedy Algorithm

Rizal Muhammad Fauzan (NIM:1301180150)

Rachmat Dwi Putra (NIM:1301180201)

Delvanita Sri Wahyuni (NIM:1301184014)

Informatics, School of Computing,
Telkom University, Indonesia.

Email: (rizalmf154@gmail.com , dprach21@gmail.com, delvasw06@gmail.com)

Abstract. Penggunaan algoritma pada pencarian jarak terdekat pada suatu tujuan wisata merupakan hal yang penting dilakukan oleh orang-orang terlebih lagi kepada para wisatawan yang ingin berkunjung kepada suatu tempat wisata karena pada dasarnya orang-orang akan mencari tempat yang menarik untuk dikunjungi dan tentu saja hal yang pertama dipikirkan oleh para wisatawan yaitu mencari sebuah rute jalan yang dekat atau pendek ke tempat wisata. Selain efektif, pencarian jarak terdekat juga sangat dibutuhkan agar terhindar nya waktu yang cukup lama untuk sampai ke tempat tujuan. Setiap orang yang melakukan perjalanan pasti akan memilih jalur alternatif atau jalur lain dengan jarak waktu yang relatif pendek. Untuk mewujudkan tujuan tersebut dapat dilakukan menggunakan Algoritma Greedy untuk membantu proses optimasi terhadap pencarian sebuah jalur rute terdekat dengan waktu yang relatif sangat singkat.

1. Pendahuluan

Setiap orang dapat melakukan perjalanan dari satu tempat ke tempat lain dengan pertimbangkan setiap waktu, biaya dan bahan bakar. Biasanya orang-orang akan cenderung malas untuk pergi berwisata dikarenakan jarak dan waktu yang ditempuh sangat panjang. maka dari itu, diperlukannya penetapan dalam menentukan jalur pendek atau jalan alternatif yang dekat untuk mencapai tempat wisata yang diinginkan.

Masalah yang dihadapi pada paper ini yaitu bagaimana caranya agar mendapatkan rute wisata pada graph jarak jauh dengan mencari rute jalan yang pendek, sehingga solusi penyelesaian yang efektif yaitu dengan menentukan rute jalan pendek pada graph agar estimasi waktu yang dapat ditempuh oleh wisatawan relatif lebih cepat. Hal tersebut dapat dicapai dengan graph, yaitu suatu lintasan dari titik awal ke titik akhir.

Penggunaan jalur terdekat merupakan bagian dari teori sebuah graph. Jika diberikan suatu graph yang berbobot, permasalah jalur terdekat yaitu bagaimana caranya agar mendapatkan suatu jalur pada graph dengan nilai bobot atau jalur yang dibuatnya.

Algoritma yang digunakan yaitu, Algoritma Greedy sebagai yang menentukan jalur dimana dipilihnya jalur yang akan diambil terlebih dahulu atau dapat disebut dengan jalur optimum lokal sehingga sampai seluruh jalur diambil di akhir perjalanan dan menciptakan sebuah rute perjalanan yang terpendek atau yang biasa disebut dengan optimum global sehingga dapat pula menyelesaikan suatu penentuan rute.

2. **Analisis Algoritma**

2.1. Algoritma Greedy

2.1.1. Pemaparan Algoritma Greedy

Algoritma Greedy adalah sebuah metode mencari atau memecahkan masalah untuk menemukan solusi optimum dalam permasalahan optimasi langkah demi langkah. Algoritma Greedy adalah algoritma yang populer digunakan untuk memecahkan permasalahan optimasi. Permasalahan optimasi sendiri dapat berupa maximization dan minimization. Greedy secara bahasa artinya tamak atau rakus.

Dalam algoritma Greedy sebagian masalah tidak selalu berhasil memberikan solusi yang benar-benar optimum. Tetapi, algoritma Greedy pasti memberikan solusi yang mendekati (approximation) nilai optimum (Hayati & Yohanes, 2014).

Untuk mencari lintasan pendek pada Algoritma Greedy dapat menggunakan rumus :

$$D(i) = L1 + \text{bobot berikutnya}$$

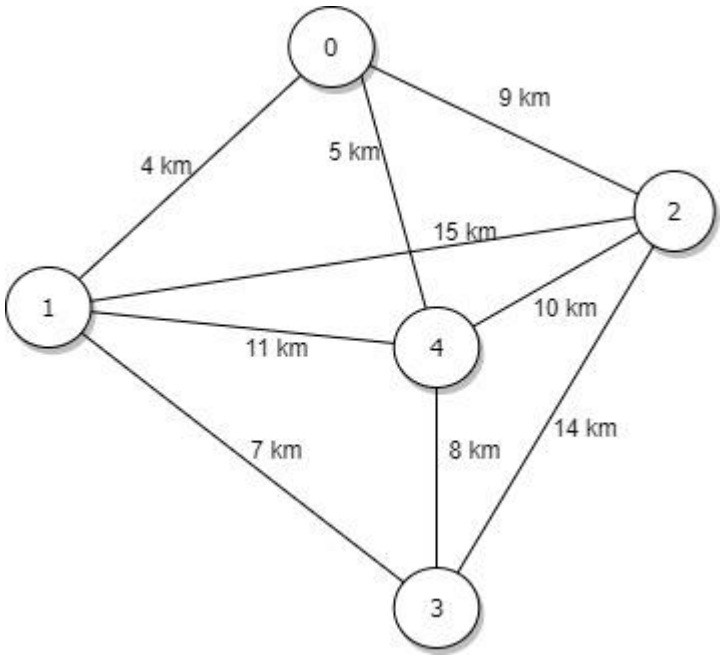
keterangan :

D(i) : sebagai inisial rute jarak terkecil
L1 : rute terpendek pertama

3. **Experimental Results**

3.1 Study Case / Problem Detail

Pada kasus ini, terdapat beberapa node dengan jarak yang berbeda-beda. Berikut ini adalah gambar dari seluruh node dan jarak nya pada rute wisata:



Gambar 1. Peta Sederhana Tempat Wisata antar Kota

Pada graf tersebut ditunjukkan beberapa kota wisata yang saling terhubung, terdapat beberapa kota yang dapat dikunjungi oleh parawisata, yaitu: Bandung, Jakarta, Banten, Surabaya, Yogyakarta. Dari masing-masing kota tersebut mempunyai jarak yang berbeda-beda seperti dari Bandung-Jakarta yaitu 4 KM, Bandung-Banten yaitu 9 KM, Banten-Surabaya 10 KM, Surabaya-Yogyakarta yaitu 8 KM dan seterusnya.

3.2 Computer Specifications

Berikut adalah spesifikasi program yang digunakan:

Komponen	Detail
Operating System	Windows 10 Home Single Language
System Manufacturer	Lenovo
System Model	81NB
BIOS	LENOVO AMCN27WW(V1.10)03/01/2020
Processsor	AMD Ryzen 3 3200U with Radeon Vega Mobile Gfx..
Memory	256 SSD
Direcx Version	10.0.18363 Build 18363

Gambar 2. Spesifikasi Program yang digunakan

3.3 Experimental Results

```
Vertex  Distance from Source: 0
 0                0       Path: 0
 1                4       Path: 0-->1
 2                9       Path: 0-->2
 3                11      Path: 0-->1 -->3
 4                5       Path: 0-->4

Time Execution: 113ms
```

Gambar 3. Vertex 0

```
Vertex  Distance from Source: 1
 0                4       Path: 1-->0
 1                0       Path: 1
 2                13      Path: 1-->0 -->2
 3                7       Path: 1-->3
 4                9       Path: 1-->0 -->4

Time Execution: 165ms
```

Gambar 4. Vertex 1

```
Vertex  Distance from Source: 2
 0                9       Path: 2-->0
 1                13      Path: 2-->0 -->1
 2                0       Path: 2
 3                14      Path: 2-->3
 4                10      Path: 2-->4

Time Execution: 217ms
```

Gambar 5. Vertex 2

```
Vertex  Distance from Source: 3
 0                11      Path: 3-->1 -->0
 1                7       Path: 3-->1
 2                14      Path: 3-->2
 3                0       Path: 3
 4                9       Path: 3-->4

Time Execution: 250ms
```

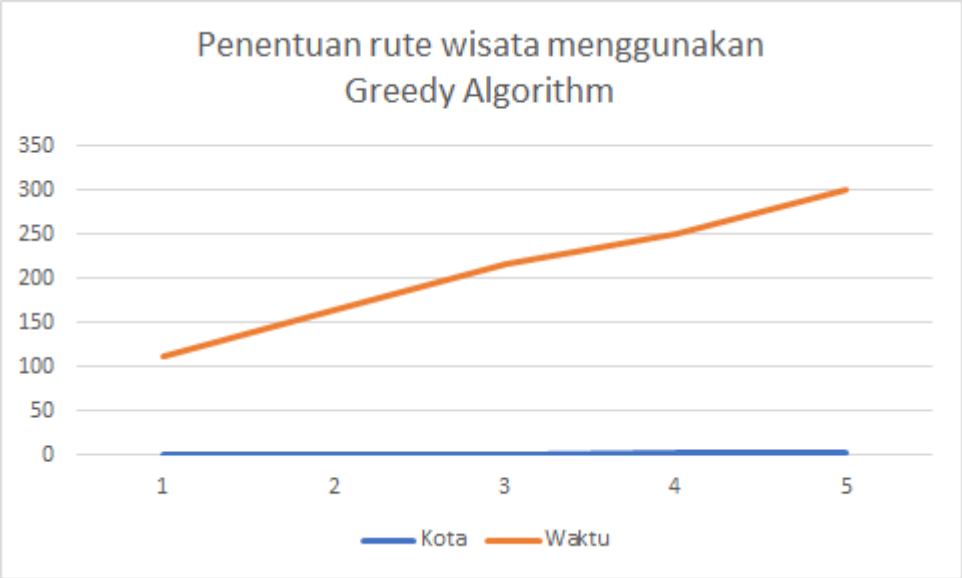
Gambar 6. Vertex 3

```
Vertex  Distance from Source: 4
 0                5       Path: 4-->0
 1                9       Path: 4-->0 -->1
 2                10      Path: 4-->2
 3                8       Path: 4-->3
 4                0       Path: 4

Time Execution: 301ms
```

Gambar 7. Vertex 7

Dari hasil diatas mulai dari vertex 0, 1, 2, 3, 4, dan yang paling kiri itu vertex tujuannya, yang tengah itu total jarak minimum, dan yang kanan jalan yang harus ditempuh agar bisa menghasilkan jarak minimum.

Penenentuan rute wisata menggunakan Algoritma Greedy :



Gambar 8. Perbandingan jarak dan waktu

4. Kesimpulan

Kesimpulan yang didapatkan adalah algoritma Greedy dapat mengeksekusi program jauh lebih cepat dari algoritma Brute Force. Jika dibandingkan dengan algoritma brute force (untuk mendapat solusi jarak optimal), rute yang dihasilkan tidak terlalu jauh, serta dapat menentukan rute yang melalui lebih banyak wisata. Dengan demikian, algoritma Greedy menghasilkan waktu paling optimal dalam mengeksekusi penentuan rute wisata.

References

[1] Hayati, E. N., & Yohanes, A. (2014). Pencarian Rute Terpendek Menggunakan Algoritma Greedy.

[2] Docplayer.info (2013, 20 Desember). Penggunaan Algoritma Greedy Dalam Penentuan Rute Wisata. Diakses pada 03 Mei 2020, dari https://docplayer.info/44961542-Penggunaan-algoritma-greedy-dalam-penentuan-rute-wisata.html

[3] Lubis, H. S. (2009). Perbandingan Algoritma Greedy dan Dijkstra untuk menentukan lintasan terpendek.

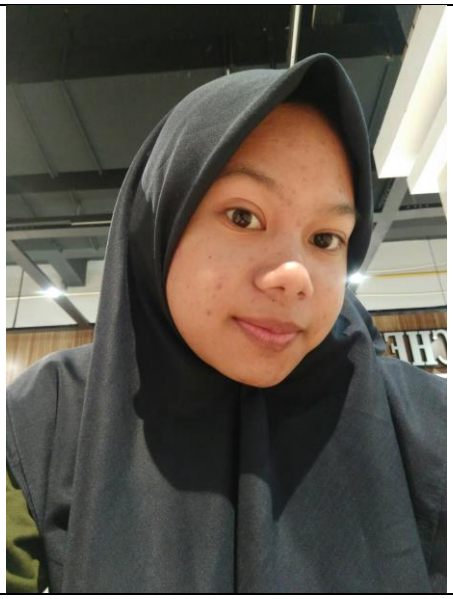
Lampiran

1. Program Code

```
#include <stdio.h>
#include <limits.h>
#include <iomanip>
#include <iostream>
#include <algorithm>
#include <chrono>

using namespace std;
using namespace std::chrono;

#define V 5

int minDistance(int dist[], bool sptSet[])
{
	int min = INT_MAX, min_index;
	for (int v = 0; v < V; v++)
		if (sptSet[v] == false && dist[v] <= min)
			min = dist[v], min_index = v;

	return min_index;
}


void printPath(int parent[], int j)
{

	if (parent[j] == -1)
		return;
	printPath(parent, parent[j]);

	printf("-->%d ", j);
}


void printSolution(int dist[], int n, int parent[], int src)
{

	for (int i = 0; i < V; i++)
	{
		printf("\n %d \t\t %d\tPath: %d", i, dist[i], src);
		printPath(parent, i);
	}
}


void greedyAlgo(int graph[V][V], int src)
```

```
{
	int dist[V];
	bool sptSet[V];
	int parent[V];
	for (int i = 0; i < V; i++)
	{
		parent[src] = -1;
		dist[i] = INT_MAX;
		sptSet[i] = false;
	}

	dist[src] = 0;

	for (int count = 0; count < V - 1; count++)
	{

		int u = minDistance(dist, sptSet);
		sptSet[u] = true;
		for (int v = 0; v < V; v++)
		{


			if (!sptSet[v] && graph[u][v] &&
				dist[u] + graph[u][v] < dist[v])
			{
				parent[v] = u;
				dist[v] = dist[u] + graph[u][v];
			}
		}
	}


	printSolution(dist, V, parent, src);
	cout << endl;
}

int main()
{
	int graph[V][V] = { { 0, 4, 9, 0, 5},
					    { 4, 0, 15, 7, 11},
					    { 9, 15, 0, 14, 10},
					    { 0, 7, 14, 0, 9},
					    { 5, 11, 10, 8, 0},
					    };
	auto start = high_resolution_clock::now();
	for (int i = 0; i < V; i++)
	{
		cout << "Vertex\tDistance from Source: " << i;
		greedyAlgo(graph, i);
		cout << endl;
	    auto stop = high_resolution_clock::now();
	    auto duration = duration_cast<microseconds>(stop - start);

	    cout << "Time Execution: " << duration.count() << "ms";
	    cout << endl;
	}

	return 0;
}
```

2. Foto Mahasiswa

<table>
<tr><td colspan="2">  </td></tr>
<tr><td colspan="2">Rizal Muhammad Fauzan (NIM:1301180150)</td></tr>
<tr><td>  </td><td>  </td></tr>
<tr><td>Rachmat Dwi Putra (NIM:1301180201)</td><td>Delvanita Sri Wahyuni (NIM:1301184014)</td></tr>
</table>