Penentuan Rute Terbaik pada Permainan *Taxi Rider*

Perbandingan antara Algoritma Greedy dan Pemrograman Dinamis

Ezra Hizkia Nathanael - 13510076
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
ezra.hizkia@students.itb.ac.id

Makalah ini disusun sebagai salah satu prasyarat kelulusan mata kuliah Strategi Algoritma (IF 3051) bagi mahasiswa program studi Teknik Informatika, Sekolah Teknik Elektro dan Informatika, Institut Teknologi Bandung tahun ajaran 2012/2013. Tema utama dari makalah ini membahas seputar algoritma penentuan rute terpendek yang disesuaikan dengan permasalahan dan fitur yang terdapat pada game berjudul Taxi Rider. Beberapa macam algoritma yang akan digunakan di dalam makalah ini adalah algoritma Greedy dan metoda Pemrograman Dinamis. Makalah ini disusun berdasarkan urutan: Pendahuluan, Prinsip Utama, Dasar Teori, Pemecahan Masalah untuk Penentuan Rute Terpendek, Pemecahan Masalah untuk Penentuan Rute Terbaik, dan Kesimpulan.

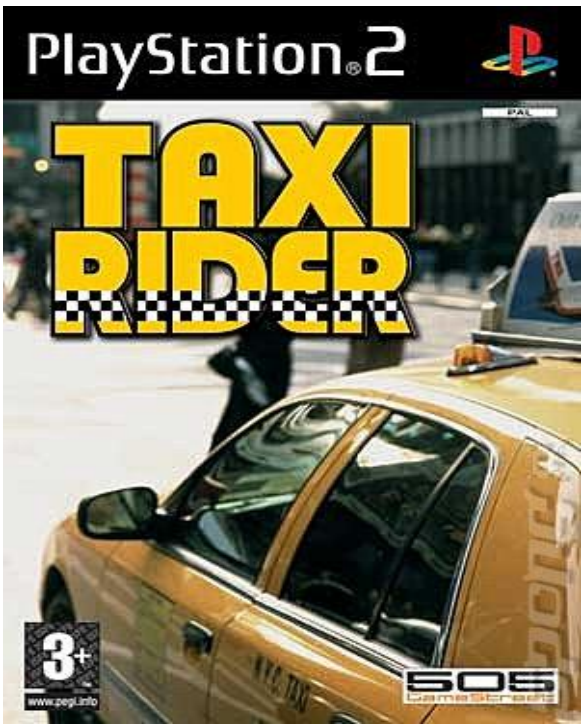
***Kata kunci*: algoritma, dynamic programming, greedy, taxi rider**

I. PENDAHULUAN

Pada bagian ini penulis akan menyampaikan beberapa pendahuluan, di antaranya latar belakang pemilihan topik dan judul, serta beberapa hal mendasar yang terkait dengan permainan Taxi Rider itu sendiri.

Latar belakang pemilihan topik berupa pemilihan rute terpendek tidak lain dikarenakan dalam kehidupan sehari-hari, hal ini cukup sering kita lakukan baik disadari maupun tidak. Dalam mata kuliah Strategi Algoritma pun seringkali permasalahan ini diangkat dan diselesaikan dengan berbagai algoritma yang ada. Beberapa algoritma yang dapat digunakan antara lain algoritma Brute Force, Greedy, Branch and Bound, dan Pemrograman Dinamis (Dynamic Programming). Namun, pada game ini, penulis memodifikasi beberapa fitur yang disediakan sehingga rute yang dipilih pada akhirnya tidak hanya sekadar rute yang terpendek, melainkan rute terbaik. Beberapa faktor yang menentukan rute yang terbaik antara lain *traffic density* (kepadatan lalu lintas), kondisi jalan, dan adanya suatu *event*/kegiatan yang dapat menyebabkan kemacetan lalu lintas atau bahkan penutupan jalan.

Sedikit penjelasan mengenai permainan Taxi Rider itu sendiri. Taxi Rider merupakan suatu permainan yang dijalankan pada konsol PlayStation2 dan bergenre simulasi. Permainan ini dikembangkan oleh perusahaan Tamsoft dan disiarkan oleh perusahaan 505 Game Street. Permainan ini dirilis untuk pasar Eropa pada tahun 2006 dan mulai masuk ke Indonesia sekitar tahun 2008.



Gambar 1: Halaman Sampul Depan Permainan Taxi Rider

diambil dari laman http://cdn3.spong.com/pack/t/a/taxirider188288l/_-Taxi-Rider-PS2-_.jpg

Permainan ini menceritakan kehidupan seorang pengemudi taksi yang dalam kesehariannya mencari penumpang dan mengantarkan para penumpang hingga tiba di tujuan. Ketika mengendarai kendaraan, belum ada navigasi yang dimunculkan, sehingga pengendara (pemain) bebas untuk menjelajahi kota dan mencari penumpang yang membutuhkan. Setelah menaikkan penumpang, pada bagian atas layar akan dimunculkan suatu anak panah yang menunjuk ke arah tujuan. Fungsinya sebagai navigasi arah sekaligus untuk memandu pengendara memilih jalan yang tercepat. Tidak lupa, karena ini merupakan suatu permainan, maka untuk mengantarkan penumpang dibatasi dalam suatu waktu tertentu yang disesuaikan dengan jarak tempuhnya.

Tujuan yang ingin ditempuh penumpang acak dan tidak ada pola tertentu. Waktu kerja dibatasi dalam suatu range tertentu, dan setiap harinya terdapat jumlah minimum yang harus dicapai (lihat pada bagian quota). Tempat yang ingin dituju oleh penumpang akan memiliki tulisan 'goal' besar di atasnya (seperti contohnya terlihat pada gambar).

Gambar 2 : Tampilan Permainan Taxi Rider

diambil dari laman http://cdn0.spong.com/screen-shot/t/a/taxirider188285l/_-Taxi-Rider-PS2-_.jpg

II. PRINSIP UTAMA

Yang akan menjadi fokus kita bersama sepanjang makalah ini adalah anak panah merah yang terdapat di bagian atas layar. Anak panah itu membantu menavigasi pengendara taksi untuk membawa penumpang menuju ke tempat tujuannya. Mengapa pada gambar tersebut terdapat dua anak panah? Hal itu dikarenakan tempat yang ingin dituju tidak hanya berjumlah satu di kota tersebut, sehingga keduanya ditunjukkan oleh anak panah, terlepas dari bagaimanapun jaraknya.

Untuk itulah, penulis sedikit memodifikasi bantuan anak panah pada game ini untuk dapat menunjukkan rute yang terbaik untuk ditempuh. Dengan demikian, cukup satu anak panah yang digunakan, di mana dengan strategi yang kita rancang pasti anak panah tersebut akan menavigasikan kita menuju tempat yang paling tepat bagi penumpang taksi kita.

Perlu diingat, untuk menentukan rute terbaik bukan hanya terbatas kepada pemilihan rute terpendek, karena akan ada dua faktor utama yang menentukan. Faktor utama jelas dari jarak di antara posisi saat ini dan tempat tujuan. Faktor keduanya adalah *traffic density* atau kepadatan jalan raya. Kepadatan jalan raya ini dapat dipengaruhi oleh banyak hal, misalnya pada saat hari perayaan, maka di sekitar kuil mungkin terjadi kemacetan. Atau juga bisa dalam bentuk penutupan jalan karena perbaikan sehingga walaupun jarak yang kita tempuh lebih dekat melalui jalan itu, kemungkinan melewatinya menjadi 0. Kepadatan ini dinyatakan dalam suatu bilangan di antara 0 dan 1. Angka 0 menunjukkan bahwa jalan tersebut tidak dapat dilewati sama sekali, sedangkan angka 1 menunjukkan bahwa jalan tersebut sangat sepi sehingga layak dilewati (misalnya saja melewati pemakaman pada saat malam hari). Angka kepadatan jalan ini kemudian digunakan untuk menghitung koefisien kelayakan jalan, yang dihitung dari jarak yang ditempuh dibagi dengan angka kepadatan jalan. Ambil contoh, terdapat dua rute pillihan jalan, yakni dari A ke B, dan dari A ke C. Jarak dari A ke B adalah 5 km, dan jarak dari A ke C adalah 3 km. Angka kepadatan jalan untuk rute A-B adalah 0,8 dan untuk rute A-C sebesar 0,2. Maka, ketika kita menghitung koefisien kelayakan jalan, untuk rute pertama (A-B) adalah sebesar:

$$\text{Koefisien A-B} = \frac{jarak\ A-B}{angka\ kepadatan\ jalan\ A-B} = \frac{5}{0.8} = 6.25$$

Sedangkan untuk menghitung rute kedua kita gunakan cara yang sama, yakni :

$$\text{Koefisien A-C} = \frac{jarak\ A-C}{angka\ kepadatan\ jalan\ A-C} = \frac{3}{0.2} = 15$$

Ketika sudah mendapatkan nilai dari kedua pilihan rute ini, maka seperti prinsip *minimum cost*, kita memilih rute dengan nilai koefisien yang paling kecil. Untuk contoh kasus ini maka kita akan memilih rute A-B, meskipun jarak tempuhnya lebih jauh namun waktu tempuh bisa lebih cepat. Dapat kita lihat pula di sini bahwa apabila suatu jalan memiliki angka kepadatan jalan 0, maka koefisien kelayakan jalan akan bernilai tidak berhingga. Penjelasan lebih detail akan ada pada bagian pembahasan dan contoh permasalahan di bagian IV.

III. DASAR TEORI

Sebelum melanjutkan ke pembahasan dan contoh permasalahan, ada baiknya apabila kita bersama-sama menyegarkan ingatan terkait dengan algoritma Greedy dan Pemrograman Dinamis itu sendiri.

A. Algoritma Greedy

Algoritma Greedy didasarkan kepada persoalan optimasi, yakni persoalan di mana solusi yang ditemukan diharapkan bukan sekadar solusi melainkan solusi yang terbaik. Solusi ini bisa yang maksimum (misalnya untuk mendapatkan laba kita menginginkan laba yang sebesar-besarnya), sedangkan solusi minimum kita inginkan ketika berhadapan dengan suatu permasalahan biaya, di mana biaya harus ditekan sekecil mungkin. Pada persoalan optimasi diberikan sekumpulan kendala dan fungsi optimasi. Solusi yang memenuhi semua kendala disebut solusi layak (feasible solution), sedangkan solusi layak yang mengoptimumkan fungsi optimasi disebut solusi optimum[1].

Algoritma Greedy dapat didefinisikan sebagai suatu algoritma yang memecahkan masalah secara *step-by-step* atau tahap demi tahap, di mana untuk setiap langkahnya kita akan mengambil pilihan terbaik yang kita jumpai saat itu tanpa memperhitungkan konsekuensinya ke depan (prinsip "take what you can get now!"). Sekali kita mengambil suatu solusi pada satu langkah maka kita tidak akan memperhitungkannya kembali pada pengambilan solusi di langkah selanjutnya. Hal ini berarti tidak ada proses *backtracking* atau runut-balik, namun algoritma ini tidak termasuk ke dalam cakupan makalah ini. Kembali ke algoritma Greedy, dengan mengambil suatu solusi yang maksimum/minimum lokal diharapkan maka pada langkah terakhir kita akan mendapatkan suatu solusi yang optimum global.

Algoritma Greedy tersusun oleh beberapa elemen yang membentuk skema dasarnya, yakni:

1. Himpunan Kandidat

   Himpunan ini berisi elemen-elemen pembentuk solusi. Untuk contoh pada makalah ini maka himpunan kandidat yang akan digunakan adalah jalan-jalan yang tersedia dan dapat ditempuh dalam membentuk rute terbaik untuk mengantar penumpang. Pada setiap langkah, satu buah kandidat akan diambil dari himpunannya.

2. Himpunan Solusi

   Berisi kandidat-kandidat yang terpilih sebagai solusi persoalan. Himpunan solusi merupakan himpunan bagian dari himpunan kandidat.

3. Fungsi Seleksi

   Dinyatakan melalui predikat SELEKSI, yakni fungsi yang pada setiap langkah memilih kandidat yang paling memungkinkan untuk mencapai solusi optimal. Kandidat yang sudah dipilih tidak akan dipertimbangkan lagi pada langkah selanjutnya. Fungsi seleksi ini biasanya memilih yang bernilai paling kecil atau paling besar (bergantung dari optimasinya) dari himpunan kandidat untuk selanjutnya diperiksa kelayakannya.

4. Fungsi Kelayakan

   Dinyatakan melalui predikat LAYAK, yang memeriksa apakah suatu kandidat yang telah dipilih dapat memberikan solusi yang layak, yakni kandidat tersebut bersama dengan himpunan solusi yang sudah terbentuk tidak melanggar constraints yang ada. Kandidat yang layak dimasukkan ke dalam himpunan solusi, sedangkan yang tidak layak dibuang dan tidak pernah dipertimbangkan lagi.

5. Fungsi Objektif

   Fungsi yang memaksimumkan atau meminumkan nilai solusi (misalnya panjang lintasan, keuntungan, dan lain-lain)

B. Pemrograman Dinamis

Pemrograman Dinamis/Program Dinamis (Dynamic Programming) adalah metode pemecahan masalah dengan cara menguraikan solusi menjadi sekumpulan tahap-tahap yang saling berkaitan. Pada metode ini terdapat sejumlah berhingga pilihan yang mungkin, solusi setiap tahap dibangun dari tahap sebelumnya, dan menggunakan prinsip optimasi dan kendala. Melalui definisi ini, dapat kita lihat keterkaitan antara algoritma Greedy dan metode Dynamic Programming ini, perbedaan yang paling mencolok adalah, ketika algoritma Greedy memilih suatu solusi pada satu tahap dan tidak mempertimbangkannya lagi, maka pada Dynamic Programming, setiap solusi tetap dicatat dan dijadikan sebagai landasan untuk membangun solusi pada tahap selanjutnya. Karena prinsip inilah metode Pemrograman Dinamis dibangun berdasarkan struktur rekursif. Karena prinsipnya serupa dengan algoritma Greedy tentu saja prinsip optimalisasi berlaku pada persoalan yang akan diselesaikan dengan metode pemrograman dinamis ini, namun dengan sedikit modifikasi. Prinsip optimalitas pada metode pemrograman dinamis berbunyi: "jika solusi total optimal, maka bagian solusi sampai tahap ke-k juga optimal". Dapat dirumuskan secara sederhana bahwa ongkos pada tahap k+1 = (ongkos yang dihasilkan pada tahap k) + (ongkos dari tahap k ke tahap k+1).

Maka dapat disimpulkan bahwa ada empat langkah yang perlu dilakukan dalam mengembangkan algoritma program dinamis, yaitu:

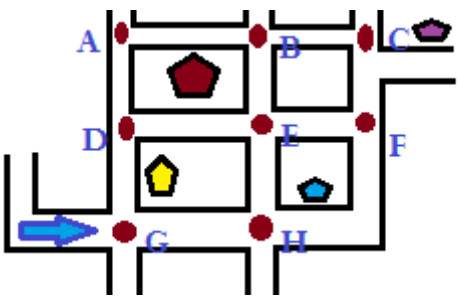
a) karakteristikkan struktur solusi optimal
b) definisikan secara rekursif nilai solusi optimal
c) hitung nilai solusi optimal secara maju dan mundur
d) konstruksi solusi optimal

Solusi optimal yang dihasilkan oleh program dinamis dapat lebih dari satu buah, dan dapat melalui metode program dinamis maju maupun mundur.

IV. Penentuan Rute Terpendek

A. Menggunakan Algortima Greedy

Untuk bagian pembahasan ini, kita akan menggunakan suatu peta seperti yang tergambar pada gambar 3 di bawah ini:



Gambar 3: Peta Jalan untuk Pembahasan

Dapat dilihat di sini bahwa node-node yang ada menyatakan percabangan jalan, di mana terdapat 8 percabangan jalan, baik yang berupa perempatan maupun pertigaan. Node ini dinamakan dengan alfabet A hingga H. Untuk yang berbentuk segilima menandakan suatu bangunan. Segilima berwarna kuning menandakan kuil (pintu masuknya ada di ruas jalan D-G), segilima merah menandakan kantor pemerintahan (pintu masuknya ada di ruas jalan A-B), segilima biru menandakan pemakaman (pintu masuknya ada di ruas jalan H-F), dan segilima ungu menandakan hotel. Tujuan kita pada contoh kasus ini adalah membawa penumpang yang berasal dari anak panah berwarna biru pada gambar untuk menuju hotel, di mana untuk setiap rute memiliki keadaannya masing-masing.

Untuk bagian pertama ini, kita akan mencari rute terbaik dalam keadaan ideal, artinya untuk angka kepadatan jalan semuanya bernilai 1, sehingga pada bagian pertama ini kita cukup menemukan rute terpendek untuk mencapai tujuan. Jarak antar node untuk permasalahan ini dituliskan di dalam tabel 1 di bawah ini:

Tabel 1: Matriks Jarak antar Node

Node	A	B	C	D	E	F	G	H
A	-1	3	-1	1	-1	-1	-1	-1
B	3	-1	2.5	-1	1	-1	-1	-1
C	-1	2.5	-1	-1	-1	1	-1	-1
D	1	-1	-1	-1	3	-1	2	-1
E	-1	1	-1	3	-1	2.5	-1	2
F	-1	-1	1	-1	2.5	-1	-1	4.5
G	-1	-1	-1	2	-1	-1	-1	3
H	-1	-1	-1	-1	2	4.5	3	-1

Tabel/matriks di atas menunjukkan jarak antar node dalam satuan kilometer. Nilai -1 menunjukkan bahwa tidak ada hubungan langsung antara dua node tersebut. Hal ini perlu kita perhatikan, yakni hubungan langsung. Inilah yang menjelaskan mengapa nilai A-G dan B-H adalah -1, karena untuk menuju G A harus melalui D (minimal) dan juga B harus melewati E (minimal) untuk mencapai H. Hal ini akan sangat penting ke depannya ketika kita sudah memperhitungkan angka kepadatan jalan untuk koefisien kelayakan jalan. Namun, untuk bagian pertama ini kita cukup memperhatikan jaraknya saja.

Karena kita akan menyelesaikan permasalahan ini dengan algoritma Greedy, maka kita akan mendaftarkan terlebih dahulu elemen-elemen dalam algoritma Greedy:

1. Himpunan Kandidat

Himpunan kandidat di sini adalah himpunan rute-rute yang dapat ditempuh untuk mencapai tujuan.

2. Himpunan Solusi

Himpunan solusi di sini berarti himpunan rute yang membentuk jalan untuk menuju tujuan yang memenuhi fungsi seleksi, layak, dan objektif.

3. Fungsi Seleksi

Fungsi seleksi pada permasalahan ini adalah memilih rute dengan jarak terkecil/terpendek.

4. Fungsi Kelayakan

Fungsi layak di sini adalah bahwa rute yang dipilih merupakan rute terpendek, menuju ke arah solusi serta tidak mengunjungi node yang sudah dikunjungi sebelumnya. Urutan prioritasnya adalah rute terpendek.

5. Fungsi Objektif

Fungsi objektifnya adalah memilih rute dari tempat asal menuju tujuan di mana jaraknya paling kecil/pendek.

Karena kita memulainya dari anak panah biru, maka node pertama yang kita kunjungi adalah node G dan akan berakhir di node C. Berikut ini merupakan tahapan pembentukan solusinya:

(i) Dari node G, kita dapat mengunjungi node D dan H. Nilai G-D adalah 2 dan nilai G-H adalah 3. Maka sesuai algoritma Greedy kita akan menuju node D. Rute yang sudah terbentuk sejauh ini adalah G-D, dengan bobot 2.

(ii) Dari node D, kita dapat mengunjungi node A dan E. Nilai D-A adalah 1 dan nilai D-E adalah 3. Maka sesuai algoritma Greedy kita akan menuju node A. Rute yang sudah terbentuk sejauh ini adalah G-D-A, dengan bobot 3.

(iii) Dari node A, kita hanya dapat mengunjungi node B. Nilai A-B adalah 1. Maka sesuai algoritma Greedy kita akan menuju node B. Rute yang sudah terbentuk sejauh ini adalah G-D-A-B, dengan bobot 4.

(iv) Dari node B, kita dapat mengunjungi node C dan E. Nilai B-C adalah 2,5 dan nilai B-E adalah 1. Maka sesuai algoritma Greedy kita akan menuju node E. Rute yang sudah terbentuk sejauh ini adalah G-D-A-B-E, dengan bobot 5.

(v) Dari node E, kita dapat mengunjungi node H dan F. Nilai E-H adalah 2 dan nilai E-F adalah 2,5. Maka sesuai algoritma Greedy kita akan menuju node H. Rute yang sudah terbentuk sejauh ini adalah G-D-A-B-E-H, dengan bobot 7.

(vi) Dari node H, kita hanya dapat mengunjungi node F karena node G sudah dikunjungi sebelumnya. Nilai H-F adalah 4,5. Maka sesuai algoritma Greedy kita akan menuju node F. Rute yang sudah terbentuk sejauh ini adalah G-D-A-B-E-H-F, dengan bobot 11,5.

(vii) Dari node F, kita hanya dapat mengunjungi node C karena node E sudah dikunjungi sebelumnya. Nilai F-C adalah 1. Maka sesuai algoritma Greedy kita akan menuju node C. Rute yang sudah terbentuk sejauh ini adalah G-D-A-B-E-H-F-C, dengan bobot 12,5.

Dengan demikian, dapat disimpulkan bahwa dengan menggunakan algoritma Greedy maka solusi yang diperoleh adalah lintasan G-D-A-B-E-H-F-C dengan total panjang lintasan 12,5. Dan menurut algoritma Greedy ini

merupakan solusi yang terbaik. Pada bagian selanjutnya kita akan membandingkan hasilnya dengan metode Pemrograman Dinamis.

B. Menggunakan Metode Pemrograman Dinamis

Pada bagian kedua ini kita akan menggunakan metode pemrograman dinamis. Seperti yang sudah tertulis pada bagian dasar teori, ada dua macam metode pemrograman dinamis, yakni maju dan mundur. Kita akan mencoba menggunakan metode maju di sini. Apabila kita menggunakan metode maju, maka tujuannya adalah mendapatkan nilai dari solusi pada tahap terakhir dengan membangunnya mulai dari tahap pertama hingga tahap terakhir-1. Di sini kita definisikan bahwa akan ada 4 tahap, di mana tahap pertama tentunya dimulai dari node G. Selanjutnya kita bangun relasi rekurensnya. Di sini tahapan akan dinyatakan dengan huruf k dan status dengan huruf s. Maka relasinya menjadi:

$f_1(s) = c_{x1}s$ (basis)
$f_k(s) = \min \{c_{xks} + f_{k-1}(xk)\}$ (rekurens)

Di mana variabel x_k menyatakan variabel peubah keputusan pada tahap k, c_{xks} menyatakan cost dari x_k ke s, $f_k(x_k,s)$ menyatakan total bobot lintasan dari x_k ke s, dan $f_k(s)$ adalah nilai minimum dari $f_k(x_k,s)$.

Pada tabel tahapan nantinya, akan ada notasi xk*, di mana x_k^* ini berari nilai x_k yang meminimumkan $f_k(x_k,s)$.

Selanjutnya kita akan bersama-sama melihat tahapan pembentukan solusinya. Dimulai dari tahap 1 hingga tahap 4:

Tahap 1:

$$f_1(s) = c_{x1s}$$

Tabel 2 : Tabel Pembentukan Solusi ProgDin Tahap 1

s	Solusi Optimum	
	$f_1(s)$	x_1^*
D	2	G
H	3	G

Tahap 2:

$$f_2(s) = \min\{c_{x2s} + f_1(x_2)\}$$

Tabel 3 : Tabel Pembentukan Solusi ProgDin Tahap 2

$s \setminus x_2$	$f_2(x_2,s) = c_{x2s} + f_1(x_2)$		Solusi Optimum	
	D	H	$f_2(s)$	x_2^*
A	3	-1	3	D
E	5	5	5	H atau D
F	-1	7.5	7.5	H

Tahap 3:

$$f_3(s) = \min\{c_{x3s} + f_2(x_3)\}$$

Tabel 4: Tabel Pembentukan Solusi ProgDin Tahap 3

$s \setminus x_3$	$f_3(x_3,s) = c_{x3s} + f_2(x_3)$			Solusi Optimum	
	A	E	F	$f_3(s)$	x_3^*
B	6	-1	-1	6	A
C	-1	-1	8.5	8.5	F

Keterangan: memang pada tahap ini kita sudah mencapai goal yakni untuk rute G-H-F-C, namun kita masih akan membandingkannya dengan tahap ke 4.

Tahap 4:

$$f_3(s) = \min\{c_{x3s} + f_2(x_3)\}$$

Tabel 5: Tabel Pembentukan Solusi ProgDin Tahap 4

$s \setminus x_{\setminus 4}$	$f_4(x_4,s) = c_{x4s} + f_3(x_4)$	Solusi Optimum	
	B	$f_4(s)$	x_4^*
C	8.5	8.5	B

Dengan demikian dapat kita simpulkan bahwa melalui pemrograman dinamis, kita menemukan solusi optimum pada dua rute, yakni G-H-F-C dan G-D-A-B-C dengan bobot yang sama yakni 8,5. Hasil ini pasti merupakan hasil yang paling optimum. Selanjutnya, ketika membandingkan dengan hasil yang diperoleh melalui algoritma Greedy, kita melihat bahwa hasil metode pemrograman dinamis jauh lebih baik.

Pada bagian selanjutnya kita akan menentukan rute terbaik dengan melibatkan faktor angka kepadatan jalan.

V. PENENTUAN RUTE TERBAIK

Bagian ini, kita akan memfokuskan diri untuk menentukan rute terbaik yang dapat dilewati seorang pengendara taksi menggunakan algoritma Greedy. Untuk menentukan rute terbaik, kita harus memperhitungkan angka kepadatan jalan, di mana nilai tersebut tertera pada matriks di bawah ini:

Tabel 6 : Matriks Angka Kepadatan Jalan setiap Rute

Node	A	B	C	D	E	F	G	H
A	0	0	0	0.8	0	0	0	0
B	0	0	0.75	0	0.8	0	0	0
C	0	0.75	0	0	0	0.9	0	0
D	0.8	0	0	0	0.75	0	0.05	0
E	0	0.8	0	0.75	0	0.8	0	0.1
F	0	0	0.9	0	0.8	0	0	1
G	0	0	0	0.05	0	0	0	0.85
H	0	0	0	0	0.1	1	0.85	0

Kembali lagi kita mengingat bahwa angka kepadatan jalan ini memiliki nilai terkecil 0 yang berarti jalan tersebut mustahil untuk dilewati, dan nilai terbesar adalah 1 yang berarti bahwa jalan tersebut sangat layak untuk dilewati karena *traffic* yang jarang atau bahkan tidak ada kendaraan lain yang melintas. Pada contoh kasus ini, kita dapati bahwa latar permainan saat ini adalah pada malam tahun baru, di mana kuil sedang mengadakan perayaan dan menimbulkan kemacetan di jalan depan kuil. Pada malam yang sama, gubernur mendatangi kantor pemerintahan di kota ini sehingga ruas jalan A-B ditutup total. Ada perbaikan jalan di ruas E-H sehingga mempersulit laju kendaraan, dan ruas jalan H-F selalu sepi setiap malam, tidak ada orang yang mau melewatinya karena ada beberapa takhayul yang beredar di kota ini bila pada malam tahun baru melewati kuburan maka tahun itu akan sial. Selebihnya, ruas jalan lain memiliki nilai di antara 0.7-0.9, yang menandakan bahwa ruas jalan tersebut cukup layak dilewati, namun masih ada kemungkinan untuk bertemu dengan traffic lain sepanjang perjalanan.

Sesuai dengan apa yang sudah tertulis pada prinsip utama, maka selanjutnya kita akan menghitung koefisien kelayakan jalan untuk setiap rute yang ada. Rumusnya adalah:

Koefisien kelayakan jalan X-Y = $jarak\ X-Y \,/\, angka\ kepadatan\ jalan\ X-Y$

Dengan demikian, kita peroleh matriks baru yang berisikan koefisien kelayakan jalan ini untuk setiap *node* dan rute:

Tabel 7 : Matriks Koefisien Kelayakan Jalan setiap Rute

Node	A	B	C	D	E	F	G	H
A	∞	∞	∞	1.25	∞	∞	∞	∞
B	∞	∞	3.33	∞	1.25	∞	∞	∞
C	∞	3.33	∞	∞	∞	1.11	∞	∞
D	1.25	∞	∞	∞	4	∞	40	∞
E	∞	1.25	∞	4	∞	3.125	∞	20
F	∞	∞	1.11	∞	3.125	∞	∞	4.5
G	∞	∞	∞	40	∞	∞	∞	3.53
H	∞	∞	∞	∞	20	4.5	3.53	∞

Dapat kita lihat di sini bahwa nilai tak hingga berarti jalan tersebut tidak ada atau memang tidak dapat dilewati sama sekali sehingga tidak akan dilibatkan dalam pembangunan solusi. Selanjutnya, kita akan menerapkan algoritma Greedy dengan terlebih dahulu mendaftarkan elemen-elemennya:

1. Himpunan Kandidat

Himpunan kandidat di sini adalah himpunan rute-rute yang dapat ditempuh untuk mencapai tujuan.

2. Himpunan Solusi

Himpunan solusi di sini berarti himpunan rute yang membentuk jalan untuk menuju tujuan yang memenuhi fungsi seleksi, layak, dan objektif.

3. Fungsi Seleksi

Fungsi seleksi pada permasalahan ini adalah memilih rute dengan koefisien kelayakan jalan paling kecil.

4. Fungsi Kelayakan

Fungsi layak di sini adalah bahwa rute yang dipilih merupakan rute dengan koefisien kelayakan jalan terkecil, menuju ke arah solusi serta tidak mengunjungi node yang sudah dikunjungi sebelumnya.

5. Fungsi Objektif

Fungsi objektifnya adalah memilih rute dari tempat asal menuju tujuan di mana nilai koefisien kelayakan jalan setiap rute paling kecil.

Pada persoalan ini kita akan menggunakan prinsip optimasi minimum, yakni mencari yang nilai/bobotnya paling kecil. Maka tahap demi tahap dalam pembangunan solusi menggunakan algoritma Greedy adalah:

(i) Dimulai dari titik G, kita memilih titik dengan koefisien kelayakan jalan terkecil. Maka kita memilih titik H dengan nilai 3,53 dan membentuk rute G-H.

(ii) Selanjutnya, dari titik H kita lihat titik lain dengan nilai terkecil. Nilai terkecil sebetulnya ada di titik G, atau dengan kata lain akan membentuk lintasan H-G, namun pilihan ini tidak memenuhi fungsi kelayakan karena tidak menuju solusi dan titik G sudah dikunjungi sebelumnya. Maka titik yang dipilih adalah titik F dengan nilai 4,5. Lintasan yang sudah terbentuk adalah lintasan G-H-F.

(iii) Pada titik F, yang memenuhi nilai terkecil adalah titik C dengan nilai 1.11, dan titik ini memenuhi fungsi kelayakan, maka akan dipilih dan membentuk lintasan G-H-F-C.

Dengan demikian, maka didapatkanlah rute terbaik pada contoh kasus ini adalah rute G-H-F-C, dengan mempertimbangkan panjangnya lintasan (jarak) dan juga angka kepadatan jalan. Algoritma Greedy pada contoh kasus ini dapat menemukan solusi yang paling baik.

VI. KESIMPULAN

Untuk pemilihan rute terpendek, pemrograman dinamis dapat menemukan solusi yang lebih baik dibandingkan dengan algoritma Greedy, karena algoritma Greedy hanya mencari nilai optimum lokal, sedangkan pada pemrograman dinamis nilai dari setiap tahap dicatat dan dipertimbangkan. Pada pemilihan rute terbaik, faktor yang mempengaruhi adalah panjang lintasan (jarak) dan angka kepadatan jalan yang akan menghasilkan koefisien kelayakan jalan. Di dalam contoh kasus yang diberikan, algoritma Greedy dapat menemukan solusi rute terbaik yang ada.

VII. UCAPAN TERIMA KASIH

Ezra Hizkia Nathanael selaku penulis mengucapkan rasa syukur yang tiada berhingga kepada Tuhan atas bimbingan dan tuntunannya sehingga makalah ini dapat disusun dan selesai tepat pada waktunya. Terima kasih pula kepada para dosen pengampu mata kuliah Strategi Algoritma (IF3051) tahun ajaran 2012/2013, Bapak Rinaldi Munir untuk penanggungjawab kelas 1 dan Ibu Masayu Leylia Khodra sebagai penanggungjawab kelas 2. Rasa terima kasih juga disampaikan kepada teman-teman anggota unit kegiatan mahasiswa Genshiken ITB yang telah memberikan dukungan baik teknis maupun non-teknis dalam penyusunan makalah ini, beserta dengan mereka yang namanya tidak dapat penulis sebutkan satu per satu. Semoga makalah ini bermanfaat bagi kita semua, salam. Tuhan memberkati.

REFERENSI

[1] Diktat Kuliah IF2251 Strategi Algoritmik
Disusun oleh Ir. Rinaldi Munir, M.T
Program Studi Teknik Informatika – STEI. 2006.
[2] http://www.gamefaqs.com/ps2/920963-taxi-rider
Tanggal Akses: 20 Desember 2012 ; 18.00

PERNYATAAN

Saya menyatakan bahwa makalah ini benar merupakan hasil karya saya pribadi dan bukan merupakan plagiarisme dari makalah orang lain maupun sumber lainnya.

Bandung,
21 Desember 2012

Ezra Hizkia Nathanael
13510076