

Analisis Pemilihan Rute Efektif untuk *Singapore MRT* Dengan *Uninformed Search*

Patrick Amadeus Irawan – 13520109

Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jalan Ganesha 10 Bandung
E-mail (gmail): 13520109@std.stei.itb.ac.id

Abstract—Pengambilan keputusan mengenai pilihan rute jalan efektif untuk mobilisasi baiknya dilakukan secara efektif untuk menghemat sumber daya dari berbagai segi. Latar belakang permasalahan ini dapat terjawab dengan menggunakan *route planning algorithm*. Berbagai jenis algoritma dengan jenis pendekatan berbeda apabila dipilih dengan baik dan benar akan menghasilkan suatu pilihan rute yang efektif. Implementasi dari metode terkait akan dibahas secara mendalam pada *paper* ini.

Keywords—BFS, DFS, UCS, Route Planning, Cost, Heuristics

I. PENDAHULUAN

Mobilisasi merupakan hal yang tidak terlepas dari hidup manusia, salah satu jenis fasilitas mobilisasi adalah fasilitas *Mass Rapid Transit* (MRT) atau *Light Rail Transit* (LRT) yang memungkinkan perpindahan dalam waktu yang relatif singkat dengan harga yang terjangkau. Sistem transportasi ini tentunya berkaitan erat dengan pemilihan kombinasi rute untuk berpindah ke tempat yang ingin dituju. Tentunya, setiap orang ingin mengoptimalkan pemakaian fasilitas ini dengan memilih rute yang terpendek sehingga sampai ke tujuan dengan lebih cepat (kecuali untuk kebutuhan khusus).

Keadaan pandemi yang bertransisi menjadi endemi akan berakibat pada meningkatnya volume mobilisasi untuk banyak orang di seluruh dunia kembali. Salah satunya adalah negara dengan fasilitas transportasi umum terbaik dengan volume pemanfaatan tertinggi di dunia, yakni Singapura.



Gambar 1.1 Ilustrasi negara Singapura

Tentunya publik telah mengenal ciri khas dari transportasi umum yang tersedia di Singapura, yakni *Mass Rapid Transit* (MRT) *Commuter* yang disediakan oleh pemerintah Singapura, terbuka untuk warga negara-nya sendiri, ataupun wisatawan.

MRT memungkinkan kita untuk melakukan mobilisasi ke berbagai rute di Singapura dalam waktu yang relatif singkat dengan biaya yang terjangkau. MRT sendiri dibagi menjadi beberapa jalur tergantung pada distrik regional yang berkaitan di Singapura. Tidak hanya itu, MRT juga terintegrasi dengan layanan fasilitas transportasi lainnya, seperti LRT untuk mendukung alternatif mobilisasi.



Gambar 1.2 Peta rute MRT Singapura

Meskipun dapat ditempuh dalam waktu relatif singkat dengan proporsi operasional yang cenderung tersedia setiap saat, tentu saja baik warga setempat dan wisatawan ingin memanfaatkan fasilitas ini dengan efektif, salah satunya dengan melakukan pemilihan rute tujuan terpendek, bahkan ketika harus melakukan transit antar stasiun.

Pengaplikasian metode pencarian rute dengan berbagai pendekatan algoritma, seperti algoritma BFS, DFS, DSL, IDS, dan UCS merupakan salah satu solusi *Uninformed Search* yang dapat digunakan untuk menjawab rute efektif yang dapat digunakan oleh tiap kemungkinan keberangkatan dan tujuan. Terdapat juga algoritma lainnya yakni *Informed Search* yang membutuhkan faktor domain heuristik tambahan untuk optimalisasi pencarian rute efektif. Pada implementasi kali ini, akan ditelaah jalur non-LRT yang terdapat pada Gambar 1.2 dengan pendekatan 3 algoritma berbeda untuk menguji dan membandingkan efektivitas rute tiap-tiap pendekatan. Perlu diperhatikan bahwa implementasi ini mengesampingkan terlebih dahulu waktu transit antar stasiun, anggapan yang digunakan adalah jalur dapat langsung dipakai tanpa perlu waktu menunggu yang berarti.

II. DASAR TEORI

A. Uninformed Search

Uninformed Search merupakan jenis algoritma pencarian rute yang beroperasi secara *brute-force* tanpa bergantung pada informasi heuristik tambahan. Istilah lain dari *Uninformed Search* adalah *Blind Search*, sesuai dengan namanya, pencarian dilakukan secara buta tanpa meninjau efektivitas dari faktor heuristik tambahan, hanya bergantung pada keberadaan posisi jalur pencarian saja.

Terdapat beberapa contoh algoritma berjenis *Uninformed Search*, di antara lain :

1. *Breadth-First Search* : pencarian prioritas secara melebar, keuntungannya adalah pencarian solusi dijamin didapat, kekurangannya adalah tidak efisien secara memori.
2. *Depth-First Search* : pencarian prioritas secara mendalam, keuntungannya adalah efisiensi memori, kekurangannya adalah pencarian dapat berujung pada *infinite loop* dan solusi tidak dijamin ditemukan.
3. *Depth-Limited Search* : pencarian dengan algoritma serupa DFS, tetapi dengan kedalaman yang diberikan batasan tertentu
4. *Iterative Deepening Depth-First Search* : pencarian yang mengkombinasikan algoritma BFS dan DFS sederhana dengan membatasi kedalaman tertentu, kemudian melanjutkan pencarian secara melebar terlebih dahulu. Keuntungannya adalah mengoptimalkan penggunaan memori dan efisiensi waktu pencarian, kekurangannya adalah repetisi implementasi algoritma.
5. *Uniform-Cost Search* : pencarian dengan pertimbangan pemilihan jalur berikutnya dengan *cost* kumulatif terkecil, keuntungannya adalah pencarian *cost* total dijamin memberikan solusi terbaik, kekurangannya adalah kemungkinan terjadinya *infinite loop* ada dikarenakan orientasinya berbasis *cost* jalur, dapat dioptimisasi dengan mencegah kembali ke jalur yang sama.
6. *Bidirectional Search* : pencarian dua arah dari *start node* dan juga dari *goal node*. Teknik yang digunakan dapat bervariasi dari jenis-jenis algoritma yang sudah dijelaskan sebelumnya. Keuntungannya adalah mempercepat proses pencarian dan efisiensi memori, kekurangannya adalah implementasi yang kompleks dan hanya bisa diimplementasi pada graf statis, dimana *goal node* telah diketahui.

B. Informed Search

Informed Search merupakan jenis algoritma pencarian rute yang beroperasi dengan bantuan fungsi heuristik. Fungsi heuristik diartikan sebagai suatu fungsi yang digunakan untuk mencari jalur yang menjanjikan dengan mempertimbangkan *state* sementara dan seberapa dekat *state* tersebut dengan *goal* yang ingin dituju. Pendekatan menggunakan heuristik melibatkan berbagai jenis pertimbangan, termasuk dari disiplin

ilmu yang lain. Nilai dari heuristik selalu bilangan positif. Rumus umum dari pencarian menggunakan metode *Informed Search* adalah $f(n) = g(n) + h(n)$, dimana $f(n)$ merupakan *cost* total, $g(n)$ merupakan *cost* bersih, dan $h(n)$ merupakan *cost* dari pertimbangan heuristik yang telah terdefinisi.

Terdapat beberapa contoh algoritma berjenis *Informed Search*, di antara lain :

1. *A* Search* : pencarian prioritas dengan rumus umum $f(n) = g(n) + h(n)$ sebagaimana didefinisikan sebelumnya. Bertujuan untuk menghindari jalur yang lebih mahal *cost* nya.
2. *Greedy Best-First Search* : pencarian prioritas dengan rumus umum $f(n) = h(n)$ saja. Bertujuan untuk mendapatkan jalur dengan fungsi heuristik minimum sehingga diharapkan mencapai tujuan dengan perpindahan seminimal mungkin.

Perlu dicatat bahwa pendekatan dengan metode heuristik tidak menjamin didapatnya solusi optimal. Namun, dengan konsiderasi pemilihan heuristik yang baik, algoritma *Informed Search* akan menghasilkan solusi yang baik dalam waktu singkat.

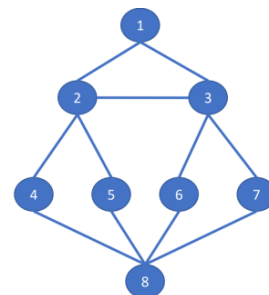
C. Breadth First Search Algorithm

BFS merupakan jenis algoritma pencarian rute yang berbasis *Uninformed Search* atau tanpa informasi heuristik tambahan. Algoritma traversal graf secara BFS memiliki sistematis traversal graf dengan mengunjungi simpul secara melebar terlebih dahulu, baru mendalam. Traversal graf menggunakan BFS dapat dipakai pada perwujudan graf statis maupun graf dinamis.

Algoritma penelusuran BFS secara lisan apabila dimulai dari simpul *v* adalah sebagai berikut :

1. Kunjungi simpul *v*
2. Kunjungi semua simpul yang bertetangga dengan simpul *v* terlebih dahulu.
3. Kunjungi simpul yang belum dikunjungi dan bertetangga dengan simpul -simpul yang tadi dikunjungi, demikian seterusnya.

Berikut penjelasan mengenai algoritma penelusuran BFS dengan bantuan tabel iterasi, misalkan terdapat graf terhubung seperti berikut.



Gambar 2.1 Contoh graf terhubung

Sumber : <https://informatika.stei.itb.ac.id/~rinaldi.munir/>

Dengan bantuan tabel yang iterasi pencarian, maka didapatkan proses pencarian dan hasil akhir sebagai berikut.

Iterasi	v	Q	dikunjungi							
			1	2	3	4	5	6	7	8
Inisialisasi	1	{1}	T	F	F	F	F	F	F	F
Iterasi 1	1	{2,3}	T	T	T	F	F	F	F	F
Iterasi 2	2	{3,4,5}	T	T	T	T	F	F	F	F
Iterasi 3	3	{4,5,6,7}	T	T	T	T	T	F	F	F
Iterasi 4	4	{5,6,7,8}	T	T	T	T	T	T	T	F
Iterasi 5	5	{6,7,8}	T	T	T	T	T	T	T	T
Iterasi 6	6	{7,8}	T	T	T	T	T	T	T	T
Iterasi 7	7	{8}	T	T	T	T	T	T	T	T
Iterasi 8	8	{}	T	T	T	T	T	T	T	T

Urutan simpul2 yang dikunjungi: 1, 2, 3, 4, 5, 6, 7, 8

Gambar 2.2 Tabel hasil penelusuran BFS

Sumber : <https://informatika.stei.itb.ac.id/~rinaldi.munir/>

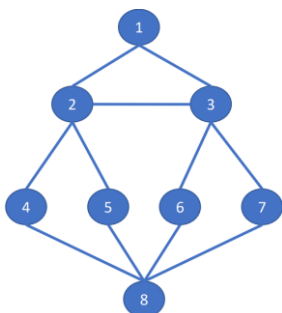
D. Depth First Search Algorithm

DFS merupakan jenis algoritma pencarian rute yang berbasis *Uninformed Search* atau tanpa informasi heuristik tambahan. Algoritma traversal graf secara DFS memiliki sistematis traversal graf dengan mengunjungi simpul secara melebar terlebih dahulu, baru mendalam. Traversal graf menggunakan DFS dapat dipakai pada perwujudan graf statis maupun graf dinamis.

Algoritma penelusuran DFS secara lisan apabila dimulai dari simpul v adalah sebagai berikut:

1. Kunjungi simpul v
2. Kunjungi simpul w yang bertetangga dengan simpul v.
3. Ulangi DFS mulai dari simpul w.
4. Ketika mencapai simpul u sedemikian sehingga semua simpul yang bertetangga dengannya telah dikunjungi, pencarian dirunut-balik (backtrack) ke simpul terakhir yang dikunjungi sebelumnya dan mempunyai simpul w yang belum dikunjungi.
5. Pencarian berakhir bila tidak ada lagi simpul yang belum dikunjungi yang dapat dicapai dari simpul yang telah dikunjungi.

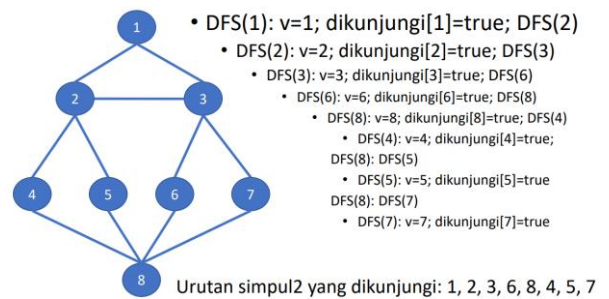
Berikut penjelasan mengenai algoritma penelusuran DFS dengan bantuan visualisasi lisan secara rekursif, misalkan terdapat graf terhubung seperti berikut.



Gambar 2.1 Contoh graf terhubung

Sumber : <https://informatika.stei.itb.ac.id/~rinaldi.munir/>

Dengan bantuan visualisasi lisan dari implementasi DFS secara rekursif, didapatkan proses pencarian dan hasil akhir sebagai berikut.



Gambar 2.3 Ilustrasi hasil penelusuran secara DFS

Sumber : <https://informatika.stei.itb.ac.id/~rinaldi.munir/>

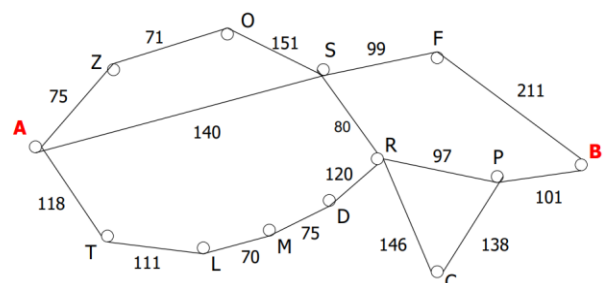
E. Uniform-Cost Search Algorithm

UCS merupakan jenis algoritma pencarian rute yang berbasis *Uninformed Search* atau tanpa informasi heuristik tambahan. Algoritma traversal graf secara UCS memiliki sistematis yang berbeda dengan BFS dan DFS, dimana algoritma UCS mempertimbangkan *cost* kumulatif sementara pada simpul-simpul hidup yang sedang dituju. Traversal graf menggunakan UCS dapat dipakai pada perwujudan graf statis maupun graf dinamis. Rumus umum dari UCS adalah $f(n) = g(n)$ dengan $g(n)$ sebagai *cost* bersih dari perpindahan 2 simpul berbeda.

Algoritma penelusuran UCS secara lisan apabila dimulai dari simpul v adalah sebagai berikut :

1. Siapkan *Priority Queue*, selanjutnya disebut PQ.
2. Masukkan v ke dalam PQ, beserta dengan *cost*-nya.
3. Ulangi hingga PQ tidak kosong.
4. Lanjutkan perjalanan pada simpul di PQ dengan *cost* kumulatif terendah.
5. Apabila simpul yang terpilih merupakan *goal node*, algoritma selesai (atau dapat melanjutkan untuk mencari alternatif solusi),
6. Apabila simpul yang terpilih bukan merupakan *goal node*, anggap simpul terpilih merupakan simpul v, kembali ke tahap 2.

Berikut penjelasan mengenai algoritma penelusuran UCS dengan bantuan tabel iterasi, misalkan terdapat graf terhubung seperti berikut.



Gambar 2.4 Contoh graf terhubung

Sumber : <https://informatika.stei.itb.ac.id/~rinaldi.munir/>

Dengan bantuan tabel yang iterasi pencarian, maka didapatkan proses pencarian dan hasil akhir sebagai berikut.

Simpul-E	Simpul Hidup
A	$Z_{A-75}, T_{A-118}, S_{A-140}$
Z_{A-75}	$T_{A-118}, S_{A-140}, O_{A2-146}$
T_{A-118}	$S_{A-140}, O_{A2-146}, L_{AT-229}$
S_{A-140}	$O_{A2-146}, R_{AS-220}, L_{AT-229}, F_{AS-239}, O_{AS-291}$
O_{A2-146}	$R_{AS-220}, L_{AT-229}, F_{AS-239}, O_{AS-291}$
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 iterasi algoritma UCS
Sumber : <https://informatika.stei.itb.ac.id/~rinaldi.munir/>

III. IMPLEMENTASI DAN EKSPERIMEN

Implementasi berbagai algoritma *path planning* dibagi menjadi beberapa tahap, yakni *mapping* jarak antara korespondensi stasiun berurutan pada peta.

Alasan utama implementasi *Uninformed Search* dikarenakan tidak tersedianya sumber informasi mengenai pendekatan heuristik untuk kasus jalur MRT Singapore sebagai faktor optimasi tambahan, sebagaimana dijelaskan pada teknik *Informed Search*. Apabila pendekatan heuristik dilakukan tanpa pertimbangan yang jelas, dapat menghasilkan hasil *Informed Search* yang cenderung kurang akurat bahkan bila dibandingkan dengan *Uninformed Search*.

A. Mapping Jarak Antar Stasiun Berurutan

Teknik yang digunakan untuk estimasi jarak antar 2 stasiun berurutan adalah dengan menggunakan *Google Maps*. Apabila ditemukan kasus dimana jarak riil antara dua stasiun tidak tersedia, digunakan estimasi sederhana berupa pencarian lokasi terdekat yang berkorespondensi dengan lokasi yang berkaitan. Estimasi jarak yang dilakukan adalah pembulatan bilangan bulat terdekat.



Gambar 3.1 Peta MRT dengan jarak antar stasiun
Sumber : Dokumen Penulis

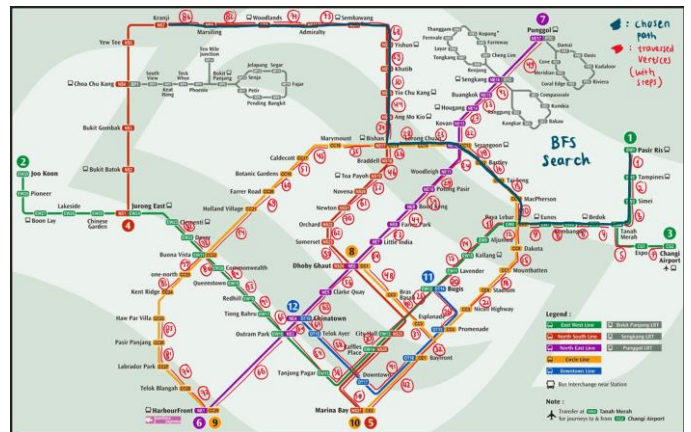
Pada eksperimen kali ini, akan dilakukan pengujian untuk perjalanan dari stasiun Pasir Ris menuju stasiun Kranji. Akan dilakukan pengujian untuk sejumlah algoritma *path planning*, diantaranya algoritma BFS, DFS, dan juga UCS yang merupakan beberapa dari banyak jenis algoritma pencarian rute dengan kategori *Uninformed Searching*. Perlu diperhatikan untuk jalur LRT (berwarna abu-abu) tidak dipertimbangkan, implementasi kali ini fokus untuk jalur MRT saja.

Berikut disediakan tabel yang akan mempermudah *mapping* jumlah stasiun untuk setiap jalur (*line*) MRT yang berkorespondensi.

MRT Line	Jumlah Stasiun
East West Line (EW)	31
North South Line (NS)	25
North East Line (NE)	25
Circle Line (CC)	27
Downtown Line (DT)	6
LRT Line	37
TOTAL	151

B. BFS Route Planning

Sesuai dengan namanya, algoritma ini mengedepankan prioritas traversal pencarian rute secara melebar, dengan asumsi graf sampel yang diuji merupakan graf yang terhubung. Pada gambar dibawah ini dipaparkan representasi urutan pencarian secara *Breadth First*.



Gambar 3.2 Implementasi BFS pada peta MRT
Sumber : Dokumen Penulis

Penjelasan singkat implementasi adalah sebagai berikut, dimulai dari stasiun Pasir Ris dimana belum ada kemungkinan traversal berbasis *Breadth First* sehingga turun ke arah Tampines dan Simei. Kemudian apabila ditemui percabangan, akan terbentuk busur baru dan traversal akan mendahului busur baru yang terbentuk, baru orientasi secara *Depth* kembali. Implementasi serupa berlangsung secara rekursif untuk penemuan simpul-simpul lainnya dengan mempertahankan prioritas pencarian melebar, apabila semua cabang sudah terbentuk orientasi *Depth* akan dijalankan untuk

traversal hingga mencapai kedalaman maksimal simpul tertentu.

Total traversal yang dibutuhkan atau dengan kata lain, pembentukan busur untuk implementasi algoritma BFS pada rute Pasir Ris ke Kranji adalah 86 busur. Algoritma BFS menghasilkan *cost* perpindahan sebanyak 62 kilometer. Anotasi jalur optimal akhir yang didapat dari algoritma ini ditandai dengan warna biru tua.

C. DFS Route Planning

Algoritma ini mengedepankan prioritas traversal pencarian rute secara mendalam, dengan asumsi graf sampel yang diuji merupakan graf yang terhubung. Pada gambar dibawah ini dipaparkan representasi urutan pencarian secara *Depth First*



Gambar 3.2 Implementasi BFS pada peta MRT
Sumber : Dokumen Penulis

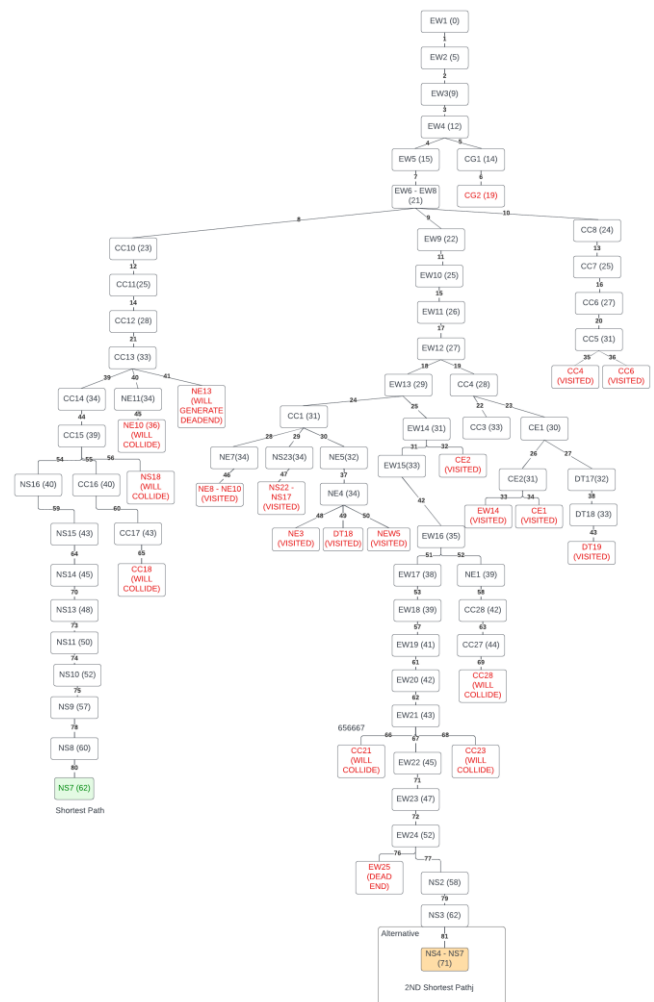
Penjelasan singkat implementasi DFS adalah sebagai berikut, dimulai dari stasiun Pasir Ris dimana belum ada kemungkinan traversal berbasis *Depth First* langsung muncul sehingga turun ke arah Tampines dan Simei. Kemudian apabila ditemui percabangan, akan tetap diprioritaskan pencarian secara mendalam. Implementasi DFS pada makalah ini menerapkan konvensi jalur kanan terlebih dahulu, sedikit berbeda dengan traversal graf pada umumnya yang mendahulukan sisi kiri. Oleh karena itu, prioritas penelusuran selanjutnya adalah ke arah stasiun Bedok sebagai persimpangan arah kanan dari stasiun Tanah Merah. Setiap kali ditemukan percabangan, orientasi traversal akan tetap memprioritaskan kedalaman cabang kanan terlebih dahulu, hingga menyentuh kedalaman maksimal baru *backtrack* ke percabangan terakhir. Algoritma ini diterapkan secara rekursif untuk simpul lainnya dalam membentuk busur sebagai penunjuk arah ke stasiun Kranji.

Total traversal yang dibutuhkan atau dengan kata lain, pembentukan busur untuk implementasi algoritma DFS pada rute Pasir Ris ke Kranji adalah 27 busur, jauh lebih sedikit dibandingkan 2 algoritma lainnya, sedangkan *cost* perpindahan yang didapatkan adalah 62 kilometer. Anotasi jalur optimal dari algoritma ini ditandai dengan garis berwarna biru tua.

D. UCS Route Planning

Berbeda dengan dua pendekatan lainnya, *Uniform Cost Search* merupakan teknik pencarian yang mengoptimalkan *cost*, yakni dalam kasus *route planning* MRT ini memiliki objektif untuk meminimalkan jarak total perjalanan. Implementasi secara algoritmik nya adalah dengan menggunakan struktur data *Priority Queue*. Produk akhir yang dihasilkan adalah jumlah kumulatif *cost* dengan nilai terkecil, sesuai objektifnya yaitu meminimumkan *route planning cost*.

Pada gambar di bawah ini, dilampirkan graf tak berarah sebagai representasi dari pencarian dengan algoritma UCS, dilengkapi dengan urutan pencariannya.



Gambar 3.2 Graf Implementasi UCS pada peta MRT
Sumber : Dokumen Penulis

Penjelasan dari algoritma UCS ini adalah sebagai berikut, implementasinya mirip dengan konsep *branch and bound* yakni optimalisasi *cost* dengan memilih simpul dengan *cost* kumulatif sementara terendah sebagai jalur selanjutnya. Dilakukan perulangan hingga menemukan *goal node* yang ingin dituju, dalam kasus ini, stasiun Kranji.

Struktur singkat dari elemen yang terdapat pada graf implementasi sebagai berikut :

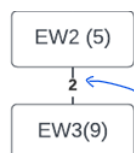
a. Simpul



Gambar 3.3 Simpul Graf implementasi UCS
Sumber : Dokumen Penulis

Representasi informasi stasiun berupa kodenya (tinjau tabel informasi mengenai korespondensi singkatan kode stasiun dengan *commuter line* MRT Singapura) , dilengkapi dengan informasi *cost* kumulatif sementara. Terdapat jenis simpul hidup, simpul mati beserta alasannya, dan simpul *goal*.

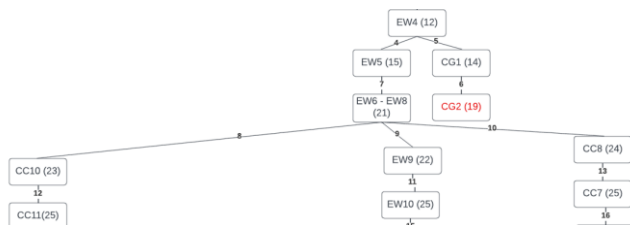
b. Busur



Gambar 3.4 Busur Graf Implementasi UCS
Sumber : Dokumen Penulis

Sederhananya, merupakan representasi hubungan antar dua simpul berbeda, dilengkapi dengan angka yang merepresentasikan urutan proses perpindahan simpul yang lebih tinggi ke lebih rendah.

Kita tinjau representasi bagian graf berikut untuk memberikan penjelasan singkat mengenai implementasi algoritma UCS pada graf yang telah dibuat.



Gambar 3.5 Protongan Graf Implementasi UCS pada peta MRT
Sumber : Dokumen Penulis

Tahapan dari penerapan algoritma UCS sesuai dengan representasi urutannya adalah sebagai berikut :

1. Dimulai dari EW4 dengan *cost* kumulatif 12, terdapat percabangan ke arah EW5 dan CG1 (tinjau Gambar 3.1 untuk representasi *cost* tiap jalur) dimulai dari kiri ke kanan , berkorespondensi pada urutan 4 dan 5.
2. Kemudian, algoritma diprioritaskan pada simpul CG1 karena *cost* kumulatif yang lebih rendah, yakni 14.
3. Karena CG2 merupakan ujung dari peta, maka simpul diberi warna merah atau dinamai *dead node*, penyebab khusus simpul ini adalah CG2 akan terpaksa kembali ke CG1 apabila dipaksa bergerak sehingga melanggar prinsip UCS yakni tidak boleh mengunjungi simpul yang sama lebih dari sekali.
4. Traversal dilanjutkan pada simpul EW15 ,dimana urutan 7 menghasilkan lanjutan kumulatif dari EW6 –

EW8 (untuk mempersingkat penulisan graf). Perlu dicatat bahwa traversal sekaligus seperti ini tidak boleh diterapkan untuk semua kasus jalur lurus, hanya boleh apabila tidak ada simpul saingan (pada kasus EW6 – EW8 , tidak terdapat simpul saingan).

5. EW8 bercabang ke 3 simpul baru, yakni CC10, EW9, CC8 dengan korespondensi *cost* kumulatif hasil penjumlahan sesuai gambar.
6. Prioritas didahulukan kepada EW10 sebagai simpul dengan *cost* kumulatif terendah sementara, yakni 25.
7. Algoritma serupa diteruskan secara iteratif, hingga mencapai *goal node*.

Goal node yang ingin dicapai adalah stasiun Krenji, atau secara kode NS7. NS7 dicapai setelah membangkitkan 80 traversal berbeda untuk solusi optimal, atau 81 simpul berbeda untuk solusi alternatif kedua. *Cost* yang dibutuhkan serupa dengan algoritma lainnya, yakni 62 kilometer.

E. Simpulan Hasil Pencarian Rute Berbagai Algoritma

Eksperimen menunjukkan bahwa ketiga algoritma menghasilkan *final cost* yang serupa, dimana secara teoretis untuk penelusuran rute pada graf, UCS dipastikan menghasilkan solusi optimal. Simpulan yang dapat diambil untuk implementasi pada pencarian rute optimal dari stasiun Pasir Ris ke stasiun Kranji adalah dikarenakan rutanya yang sepadan dengan konsep DFS, memungkinkan untuk algoritma DFS unggul dalam segi jumlah traversal yang diperlukan untuk mencapai solusi optimal. Bukan berarti untuk rute lainnya, konsep penggunaan DFS dan BFS dipastikan memberi solusi yang optimal, sehingga penulis menyarankan untuk tetap menggunakan UCS apabila tujuan yang ingin dicapai adalah mendapatkan pemilihan rute optimal dengan *cost* minimum, bukan efisiensi traversal yang diperlukan.

Kemungkinan pengembangan yang dapat diterapkan adalah penggunaan *Informed Search* di lain kesempatan, terutama apabila tersedia sumber pertimbangan heuristik yang faktual dimana dengan penelusuran tersebut, terdapat kemungkinan rute dengan *cost* yang lebih optimal.

VIDEO LINK AT YOUTUBE

<https://youtu.be/G4Z-7eDvTEI>

IV. UCAPAN TERIMA KASIH

Pertama-tama, penulis ingin menyampaikan rasa syukur terbesarnya kepada Tuhan Yang Maha Esa karena atas rahmat-Nya, makalah berjudul “Analisis Pemilihan Rute Efektif untuk *Singapore* MRT Dengan *Route Path Planning*” dapat terselesaikan dengan baik dan tepat waktu. Penulis juga ingin menyampaikan rasa terima kasih kepada dosen mata kuliah IF2211 Strategi Algoritma, Dr. Ir. Rinaldi Munir, M. T., Dra. Masayu Leylia Khodra, S.T, M.T, dan Dr. Nur Ulfa Maulidevi, S. T, M. Sc. atas bimbingannya selama menjalani perkuliahan Strategi Algoritma sebagai bahan dalam pembuatan makalah ini. Tidak lupa penulis ingin berterimakasih kepada pihak dan sumber yang dijadikan referensi dalam pembuatan makalah ini.

REFERENCES

- [1] <https://informatika.stei.itb.ac.id/~rinaldi.munir/> Diakses pada 20 Mei 2022 , 21.55 WIB.
- [2] <https://www.travelsingapura.com/peta-mrt-singapore/> Diakses pada 20 Mei 2022 , 20.13 WIB
- [3] <https://www.javatpoint.com/ai-informed-search-algorithms> Diakses pada 21 Mei 2022 , 13.15 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, 21 Mei 2022



Patrick Amadeus Irawan - 13520109