Pemanfaatan Algoritma A-Star untuk Navigasi Peta Digital

Eka Sunandika, 13517130
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
13517130@std.stei.itb.ac.id

***Abstract*—Peta memiliki kegunaan untuk mencari suatu tempat dan untuk mengetahui jalur untuk menuju tempat yang dicari. Terdapat peta digital yang mudah untuk digunakan. Keberadaan teknologi saat ini dapat memungkinkan kita untuk mendapatkan jalur secara otomatis menggunakan algoritma yang tersedia. Salah satunya adalah menggunakan algoritma A* untuk melakukan navigasi tersebut.**

Keywords— A*, Algoritma, jalur, peta

I. PENDAHULUAN

Peta merupakan gambaran permukaan bumi pada bidang datar. Tiap peta memiliki skala dan sistem proyeksi dalam pembuatannya. Permukaan bumi yang luas tersebut dapat digambarkan pada peta dengan skala tertentu sehingga dengan ukuran tersebut bisa merepresentasikan bentuk aslinya. Sebuah peta merupakan bentuk dua dimensi dari suatu ruang tiga dimensi yang dibuat pada bidatang datar.

Terdapat dua cara dalam menyajikan peta, yaitu peta secara fisik yang dibuat dengan media seperti kertas dan menyajikan peta secara digital dengan media digital yang dapat ditampilkan atau dicetak di layar digital (komputer, dll). Peta fisik memiliki kekurangan pada ukurannya yang akan semakin besar apabila menggambarkan wilayah yang luas dengan detail. Maka dari itu, peta digital memiliki keunggulan kemudahan dalam penggunaan dan ukurannya yang relatif kecil karena kita dapat gunakan pada komputer atau bahkan *smarphone* yang mudah dibawa kemana – mana.

Kita dapat mengakses peta digital dengan mudah. Peta digital yang umum digunakan orang – orang adalah *Google Maps*. Aplikasi tersebut mudah digunakan dan dapat diakses menggunakan *smarphone*. Aplikasi tersebut sangat berguna untuk melakukan navigasi pada peta.

Tujuan dari pembuatan peta ini dapat membantu kita dalam beberapa kegiatan sehari – hari. Contoh sederhananya adalah membantu kita untuk mencari suatu tempat dan mencari jalur ketempat tersebut. Dengan menggunakan *Google Maps*, jalur terpedek pada dua titik(dua tempat) yang berbeda dapat ditentukan. Kegunaan aplikasi tersebut sangat membantu apabila jalur yang dicari dari dua tempat yang berbeda itu memiliki jarak yang cukup jauh. Hal ini yang membedakan dengan peta fisik yang mudah jika jarak yang dicari tidak terlalu jauh.

Terdapat beberapa algoritma dalam menentukan jalur dua tempat yang berbeda. Salah satunya adalah dengan algoritma A*. Teori graf digunakan dalam algoritma tersebut untuk memetakan titik dan jalur yang dapat terbentuk dalam menentukan jalur terpendek yang dapat ditempuh.

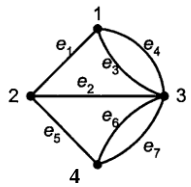
II. DASAR TEORI

A. Graf

a. Definisi Graf

Secara matematis, dapat didefinisikan bahwa suatu graf G merupakan pasangan himpunan (*V, E)*, dapat ditulis dengan notasi G = (V,E). *V* adalah simbol dari himpunan simpul (*vertices* atau *node*) yang tidak boleh kosong dan E adalah simbol dari himpunan sisi (*edgesI atau* arcs) yang boleh kosong dan menghubungkan sepasang simpul.

Simpul pada graf dapat diberi nama dengan huruf (a, b, c, d, …), angka (1, 2, 3, …), atau gabungan keduanya. Simpul *u* dan simpul *v* yang dihubungkan dengan suatu sisi (*u, v*) dapat dinyatakan dengan lambang e_1, e_2, ….



Gambar 1: Ilustrasi graf (Sumber: Munir, Rinaldi. 2006. Diktat Kuliah IF2120 Matematika Diskrit. Bandung : Institut Teknolgi Bandung)

Gambar diatas merupakan contoh graf yang miliki himpunan simpul sebagai berikut :

$V = \{1, 2, 3, 4\}$

$E = \{e_1, e_2, e_3, e_4, e_5, e_6, e_7\}$

$= \{(1, 2), (2, 3), (1, 3), (1, 3), (2, 4), (3, 4), (3, 4)\}$

b. Jenis Graf

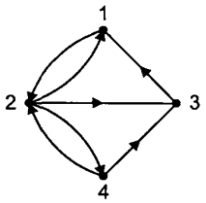
Berdasarkan ada tidaknya sisi ganda atau kalang :

1. Graf Sederhana (*simple graph*)
   Merupakan graf yang tidak mengandung gelang maupun sisi-ganda. Sisi (*u, v*) dapat dulis juga dengan (*v, u*).
2. Graf tak-sederhana (*unsimple-graph*)

Merupakan graf yang mengandung sisi ganda atau gelang. Memiliki dua macam, yaitu graf ganda (*multigraph*) yang mengandung sisi ganda dan graf semu (*pseudograph*) yang mengandung gelang (*loop*).

Berdasarkan orientasi arah :

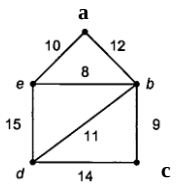
1. Graf tak-berarah (*undirected graph*)
   Merupakan graf yang sisinya tidak memiliki orientasi arah. Pasangan simpul yang dihubungkan sisi tidak diperhatikan urutannya, maka $(u, v) = (v, u)$.
2. Graf berarah (*directed graph* atau *digraph*)
   Merupakan graf yang setiap sisinya diberikan orientasi arah. Pada busur (u , v), simpul u adalah simpul asal dan simpul v adalah simpul terminal.



Gambar 2: Ilustrasi graf berarah (Sumber: Munir, Rinaldi. 2006. Diktat Kuliah IF2120 Matematika Diskrit. Bandung: Institut Teknolgi Bandung)

c. Terminologi Dasar

1. Bertetangga (*Adjacent*)
   Dua buah simpul pada graf tak-berarah yang terhubung oleh sebuah sisi dikatakan bertetangga. Simpul u dan simpul v dikatakan bertetangga jika terdapat sisi (u, v) pada suatu graf.
2. Bersisian (*Incident*)
   Sisi e dikatakan bersisian dengan simpul u dan simpul v jika terdapat sisi $e = (u, v)$ pada suatu graf.
3. Simpul Terpencil (*Isolated Vertex*)
   Simpul yang tidak mempunyai sisi yang bersisian dengannya dan tidak bertetangga dengan simpul lainnya disebut dengan simpul terpencil.
4. Graf Kosong (*Null Graph*)
   Graf kosong merupakan graf yang himpunan sisinya adalah himpunan kosong.
5. Degree (*Degree*)
   Jumlah sisi yang bersisian dengan suatu simpul disebut sebagai derajat simpul tersebut. Memiliki notasi $d(v)$. Pada graf berarah, derajat simpul v dinyatakan dengan $d_{in}(v)$ yang merupakan derajat masuk ke simpul v dan $d_{out}(v)$ yang merupakan derajat keluar dari simpul v.
6. Lintasan
   Lintasan memiliki definisi apabila sebuah lintasan yang panjangnya n dari simpul awal v_0 ke simpul tujuan v_n di dalam graf G adalah barisan berselang – seling simpul – simpul dan sisi – sisi yang memiliki bentuk $v_0, e_1, v_1, e_2, v_3, \dots , v_{n-1}, e_m, v_n$ sedemikian sehingga $e_1 = (v_0, v_1)$, $e_2 = (v_1, v_2),\dots ,$ $e_n = (v_{n-1}, v_n)$ adalah sisi – sisi dari graf G. Apabila graf tersebut adalah graf sederhana, maka dalam menyatakan lintasannya cukup dengan menuliskan lintasan sebagai barisan simpul-simpulnya. Pada graf ganda, penulisan lintasan tersebut berupa simpul dan sisi dengan berselang – seling. Lintasan yang berawal dan berakhiran di simpul yang sama disebut lintasan tertutup (*closed path*), sedangkan yang tidak seperti itu disebut lintasan terbuka (*open path*).
7. Siklus atau sirkuit (*Cycle* atau *Cirkuit*)
   Lintasan tertutup atau yang memiliki simpul awal dan akhir yang sama disebut sebagai siklus atau sirkuit. Sirkuit sederhana (*simple circuit*) adalah sirkuit dengan setiap sisi yang dilaluinya berbeda.
8. Terhubung (*Connected*)
   Jika terdapat lintasari dari simpul u ke simpul v, maka kedua simpul tersebut dikatakan terhubung. Dua simpul pada graf berarah dikatakan terhubung kuat (*strongly connected*) apabila terdapat lintasan dari u ke v dan sebaliknya. Dua buah simpul yang tidak terhubung kuat tetapi terhubung pada graf tak-berarahnya dikatakan terhubung lemah (*weakly connected*).
9. Upagraf (*Subgraph*) dan Komplemen Upagraf
   Sebuah graf G_1 disebut upagraf dari G jika terdapat V_1 yang merupakan himpunan bagian dari V dan terdapat E_1 yang merupakan himpunan bagian dari E. Komplemen dari upagrag G_1 terhadap graf G adalah graf $G_2 = (V_2, E_2)$ sedemikian sehingga $E_2 = E – E_1$ dan V_2 adalah himpunan simpul yang anggota-anggota E_2 bersisian dengannya.
10. Upagraf Merentang (*Spanning Subgraph*)
    Upagraf G_1 dikatakan upagraf merentang jika pada G_1 terdapat semua simpul yang ada di graf G.
11. *Cut-set*
    Cut-set dari graf terhubung G adalah himpunan sisi yang bila dibuang dari G menyebabkan G tidak terhubung.
12. Graf berbobot (*weighted graph*)
    Apabila terdapat graf yang setiap sisinya diberi harga (bobot), maka graf tersebut disebut graf berbobot. Bobot yang terdapat pada sisi graf memiliki makna yang berbeda – beda bergantung pada masalah yang ada. Contoh – contoh nya adalah bobot yang menyatakan jarak antara dua kota, biaya perjalanan antara dua kota, ongkos produksi, dan jarak yang ditempuh player atau *AI* dalam suatu permainan,
    Pada *pathfinding*, graf yang digunakan adalah graf berbobot. Graf ini yang akan memodelkan jarak yang ingin ditempuh antara dua titik. Bobot pada sisi tersebut menyatakan jarak di peta *game* tersebut.



Gambar 3: Ilustrasi graf berbobot (Sumber: Munir, Rinaldi. 2006. Diktat Kuliah IF2120 Matematika Diskrit. Bandung: Institut Teknolgi Bandung)

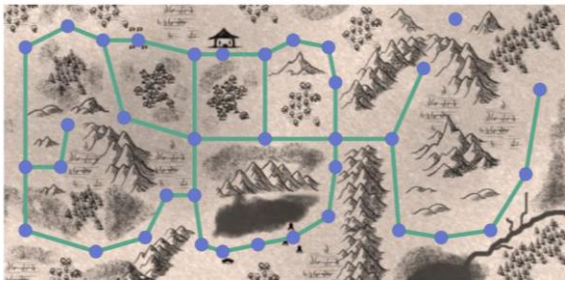
*B. Algoritma A**

Algoritma ini digunakan untuk mencari jalur optimal antara dua titik. Dalam navigasi peta, algoritma A* akan memeriksa dari simpul awal yang belum diperiksa dan akan berhenti jika simpul yang dituju sudah diperiksa. Jarak yang didapat merupakan jarak yang optimal ketika menggunakan algoritma ini.

Terdapat terminologi dalam algoritma ini, $g(n)$ merepresentasikan biaya persis yang diperlukan suatu lintasan dari simpul n ke semua simpul, dan $h(n)$ yang merepresentasikan biaya heuristic dari simpul n ke simpul tujuan.. Heuristic dapat digunakan untuk mengontrol algoritma A*. Jika $h(n) = 0$, maka hanya $g(n)$ yang mempengaruhi algoritma. Semakin kecil nilai dari $h(n)$, semakin banyak simpul yang diperluas dan menyebabkan algoritma jadi melambat. Apabila $h(n)$ memiliki besar yang sama dengan biaya yang diperlukan simpul n ke tujuan, maka algoritma akan berjalan dengan cepat dan memiliki jalur terbaik. Jika $h(n)$ memiliki nilai lebih besar dari biaya yang diperlukan dari simpul n ke tujuan, ada kemungkinan jalur minimum tidak dapat ditemukan. Kemampuan algortima ini bisa bervariasi dan berguna dalam navigasi peta.

Algoritma A* akan memproses $f(n) = g(n) + h(n)$. Dua nilai tersebut harus memiliki skala yang sama untuk dihitung. Jika skalanya berbeda, maka algoritma ini akan berjalan lambat dan tidak akan memperoleh lintasan yang baik. Jika nilai heuristic memiliki nilai yang sama dengan jarak pada lintasan yang optimal, maka algoritma A* hanya memperluas beberapa simpul saja.

Cara untuk belajar algoritma ini adalah dengan memahami data yang di input dan di output. Graf menjadi inputan pada algoritma ini. Graf merepresentasikan sekumpulan tempat(simpul) dan jalur(sisi).



Gambar 4: Ilustrasi graf pada peta (Sumber: https://www.redblobgames.com/pathfinding/a-star/introduction.html)

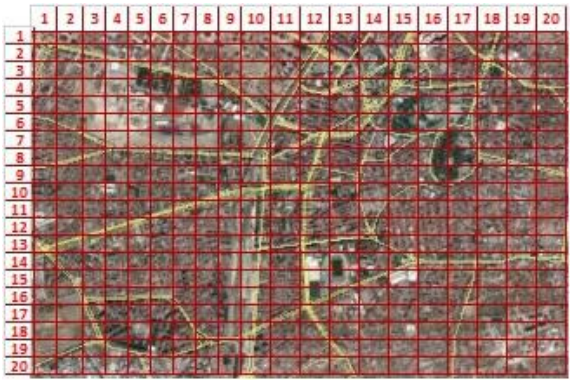
A* hanya melihat graf. Algoritma ini tidak melihat objek lain pada peta. Jalur terpendek dicari berdasarkan graf. Output dari algoritma berupa jalur yang harus diambil berdasarkan simpul dan sisi pada graf. A* akan lebih cepat bekerja apabila simpul yang ada tidak terlalu banyak.

III. Implementasi Algoritma

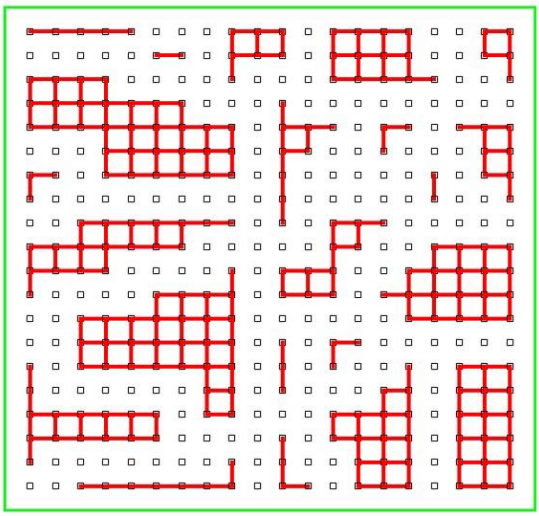
Navigasi pada peta direpresentasikan dengan graf. Peta biasanya dibentuk mejadi *grid*(kisi) untuk memudahkan representasi graf. Hal ini dilakukan karena merupakan langkah yang paling mudah untuk membentuk simpul dan sisi pada graf yang akan terbentuk.

Gambar 5: Ilustrasi peta (Sumber: E.Dere, A.Durdu, Usage of the A* Algorithm to Find the Shortest Path in Transportation Systems.)

Semua jalur pada peta diubah menjadi simpul dan sisi yang bisa dilalui. Objek lain merupakan simpul yang tidak bisa dilalui. Contoh gambar dibawah adalah peta yang dibuat menjadi *grid* 20x20.



Gambar 6: Ilustrasi peta dengan grid (Sumber: E.Dere, A.Durdu, Usage of the A* Algorithm to Find the Shortest Path in Transportation Systems.)



Gambar 7: Ilustrasi peta dengan grid (Sumber: E.Dere, A.Durdu, Usage of the A* Algorithm to Find the Shortest Path in Transportation Systems.)

Peta yang sudah dubah menjadi bentuk graf, dapat dilakukan pencarian jalur terpendek menggunakan algoritma A*. Tiap simpul terhubung dengan simpul lain dengan suatu sisi. Simpul pada peta memiliki beberapa keterangan seperti nama tempat/jalan tersebut. Tempat awal menjadi titik awal dan Tempat tujuan menjadi titik akhir. Langkah pencarian algoritma A* ini adalah dengan membandingkan tiap langkah yang selanjutnya dipilih berdasarkan fungsi biaya terkecil. Biaya yang harus ditempuh dalam menentukan jalur terpendek didasarkan oleh panjang jalanan, banyaknya persimpangan, dan lain – lain. Heuristic algoritma ini ditentukan berdasarkan hal tersebut. Hasil yang didapat bisa dipastikan bahwa jalur tersebut adalah yang paling terpendek dan efektif jika menggunakan algoritma A* ini.

Setelah dilakukan pencarian jalur terpendek akan ada navigasi jalur yang dapat ditempuh untuk sampai pada lokasi tersebut tujuan.

Gambar 8: Ilustrasi peta (Sumber: E.Dere, A.Durdu, Usage of the A* Algorithm to Find the Shortest Path in Transportation Systems.)

Biaya yang harus ditempuh dalam menentukan jalur terpendek didasarkan oleh panjang jalanan, banyaknya persimpangan, dan lain – lain. Heuristic algoritma ini ditentukan berdasarkan hal tersebut. Hasil yang didapat bisa dipastikan bahwa jalur tersebut adalah yang paling terpendek dan efektif jika menggunakan algoritma A* ini.

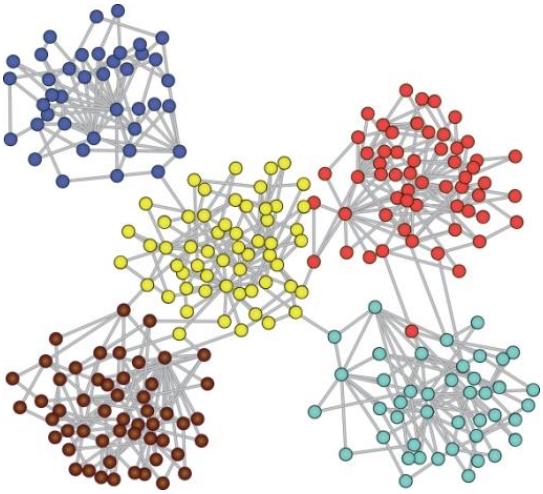
Terdapat beberapa algoritma lain dalam mencari solusi pada permasalahan jalur terpendek, seperti algoritma DFS, BFS, Dijkstra, dll. Algoritma A* memiliki kemiripan dengan algoritma Dijkstra(pada Djikstra nilai heuristic nya nol). Algoritma A* memiliki hasil yang cukup baik dalam menentukan jalur, dikarenakan fungsi biaya dalam menentukan jalurnya memiliki heuristic yang ditentukan sedemikian untuk menghasilkan hasil yang paling optimal dan terbaik.

Aplikasi *Google Maps* tidak hanya menampilkan jalur terpendek yang dapat ditempuh, tetapi juga bisa memberikan waktu estimasi perjalanan. Tidak hanya algoritma untuk pencarian jalur terpendek saja yang digunakan. Ada beberapa hal lain yang diperhitungkan untuk mendapat jalur seefektif mungkin. Informasi lalu lintas juga bisa diketahui pada aplikasi tersebut. Jadi, jalur yang dipilih biasa tidak hanya yang terdekat, tetapi juga yang memiliki lalulintas yang cenderung lancer sehingga waktu yang ditempuh dalam perjalan tersebut semakin cepat.

Untuk mencari jalur pada jarak yang tidak terlalu dekat memang mudah, tetapi untuk jarak yang jauh maka graf yang terbentuk juga semakin kompleks dan besar. Lebih membutuhkan banyak waktu dalam menentukan jalur antara dua titik pada kondisi tersebut.

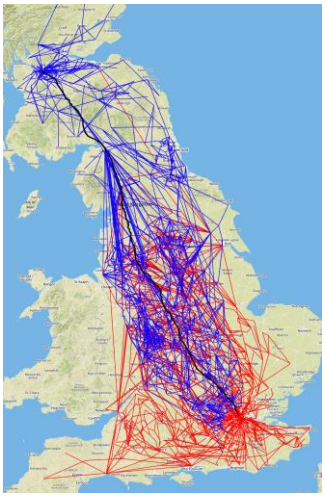
Seperti contohnya apabila ingin dicari jalur dari ITB ke Alun – Alun Bandung, maka peta yang diproses untuk menentukan jalur yang dapat ditempuh tersebut dapat memproses peta kedalam simpul dan sisi dengan jumlah yang tidak terlalu banyak. Jika jalur yang ingin dicari adalah antara kota Bandung ke Bekasi, maka peta akan memproses graf dalam ukuran yang sangat besar dan memiliki banyak simpul dan sisi yang tidak perlu dicek untuk dilalui.

Permasalahan tersebut dapat dilakukan dengan melakukan generalisasi terhadap simpul yang akan dibentuk. Simpul yang dibentuk tidak perlu pertempat, tetapi bisa dibuat perkota. Teknik pengelompokan graf dilakukan untuk generalisasi tersebut(*Graph Clustering*). Hal ini dilakukan untuk mempermudah pencarian.



Gambar 9: Ilustrasi pengelompokan graf (Sumber: https://i.redd.it/cncm2s4itxv11.png)

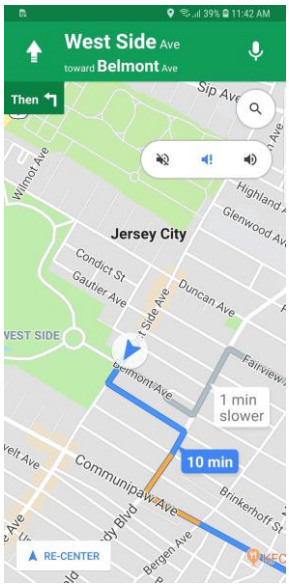
Pengelompokan graf akan melakukan pengelompokan graf yang memiliki kemiripan atau keterhubungan terhadap suatu hal yang sudah ditentukan. Apabila ingin mengelompokan kota, maka daerah kota tersebut menjadi sebuah kesatuan graf dan terhubung dengan graf – graf lainnya (kota lainnya). Terdapat algoritma sperti algoritma *fuzzy clustering* untuk melakukan hal tersebut.



Gambar 10: Ilustrasi pengelompokan graf (Sumber: https://www.mjt.me.uk/assets/images/contraction-hierarchies/london-to-edinburgh.png)

Tiap graf akan diproses untuk membentuk suatu sekumpulan graf yang lebih besar. Metode ini akan menjadi efektif dibandingkan apabila melakukan penelusuran ke semua jalur yang terbentuk. Dengan cara ini, maka jalur yang dicari adalah jalur menuju kota – kota (kelompok graf) dan akan lebih efektif dan tidak memakan memori yang banyak akibat simpul dan sisi yang dilalui akibat penelusuran ke jalur yang tidak dituju.

Pencarian jalur akan dilakukan lagi pada bagian kota yang dituju ke tempat tujuan. Seperti contoh sebelumnya, yaitu kota Bekasi sebagai kota tujuan, maka akan dicari jalur mana yang harus dituju dari pintu masuk kota tersebut ke tempat tujuan. Pencarian dilakukan lebih efektif sehingga tidak perlu mengecek jalur lain sehingga waktu yang diperlukan dalam pencarian lebih singkat. Kemudahan tersebut yang disediakan oleh peta digital dalam menentukan navigasi pada peta. Kita tidak perlu melakukan pengecekan sendiri secara manual. Sudah terdapat algoritma yang digunakan untuk menentukan jalur terpendek untuk sampai ke suatu tempat tersebut.



Gambar 11: Ilustrasi pengelompokan graf (Sumber: https://img.gadgethacks.com/img/62/61/63661279469947/0/google-maps-101-tweak-voice-navigation-prompts-android-iphone-for-clearer-spoken-directions.w1456.jpg)

Tiap aplikasi peta digital pastinya memiliki algoritma dan caranya masing – masing untuk menentukan jalur untuk menempuh dua titik yang berbeda. Jalur yang ditampilkan pada peta digital dipastikan merupakan yang paling efektif berdasarkan pertimbangan – pertimbangan yang ada. Peta digital seperti *Google Maps* biasanya juga memiliki fitur navigasi yang menuntun pengguna menuju tempat tujuannya. Jalur yang dituju akan diberitahu hingga sampai tujuan dan akan selalu melakukan *update* terhadap posisi pengguna saat itu. Fitur ini juga bisa memberi peringatan untuk melakukan belok kepada pengguna. Apabila jalur yang ditempuh berbeda dengan yang ditampilkan dilayar, maka aplikasi akan melakukan rute ulang jalur tersebut. Fitur ini sangat berguna dan memudahkan perjalanan pengguna yang menggunakan aplikasi tersebut.

IV. KESIMPULAN

Peta digital memiliki kemudahan dibandingkan peta fisik. Informasi bisa didapat dengan mudah dan praktis menggunakan peta digital. Navigasi peta untuk menentukan jalur tempat yang dituju dapat dilakukan dengan otomatis menggunakan algoritma A*. Peta direpresentasikan sebagai graf yang memiliki simpul dan sisi. Tempat awal menjadi titik awal pencarian dan tempat tujuan menjadi titik akhir pencarian. Jalur yang dihasilkan merupakan jalur terpendek. Salah satu aplikasi peta digital yang populer adalah *Google Maps*. Pencarian jalur pada aplikasi tersebut juga memperhatikan hal lain seperti informasi lalu lintas untuk menghasilkan hasil yang lebih akurat dan optimal.

V. UCAPAN TERIMAKASIH

Puji syukur penulis ucapkan kepada Tuhan Yang Maha Esa atas nikmat-Nya yang sangat melimpah dan atas diberikannya kesempatan kepada saya untuk menyelesaikan makalah ini. Juga ucapan terima kasih sebesar – besarnya kepada dosen – dosen mata kuliah IF2210 Strategi Algoritma atas ilmu dan bimbingannya selama 1 semester ini sehingga makalah ini dapat terselesaikan. Saya juga berterima kasih kepada orang tua dan keluarga terdekat yang tiada hentinya selalu mendoakan agar saya mendapatkan segala sesuatu yang terbaik dan yang telah memberi dukungan atas segala hal yang saya lakukan. Dan terakhir, tidak lupa berterima kasih kepada teman – teman saya yang selalu membantu dalam menghadapi berbagai permasalahan dan selalu ada dikala senang maupun duka.

REFERENSI

[1] Munir, Rinaldi. 2006. Diktat Kuliah IF2120 Matematika Diskrit. Bandung: Institut Teknolgi Bandung.

[2] E.Dere, A.Durdu, Usage of the A* Algorithm to Find the Shortest Path in Transportation Systems, http://indexive.com/Paper/Download/93/usage-of-the-a-algorithm-to-find-the-shortest-path-in-transportation-systems, diakses pada tanggal 26 April 2019.

[3] W. Zeng, R. L. Church, Finding shortest paths on real road networks: the case for A*, https://dl.acm.org/citation.cfm?id=1552452, diakses pada 25 April 2019.

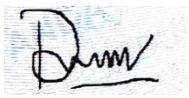
[4] Schaeffer , Satu Elisa, Survey - Graph clustering, http://www.leonidzhukov.net/hse/2016/networks/papers/GraphClustering_Schaeffer07.pdf, diakses pada 26 April 2019.

[5] Amit Patel, Introduction to A*, http://theory.stanford.edu/~amitp/GameProgramming/AStarComparison.html, diakses pada tanggal 25 April 2019.

PERYATAAN

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, 26 April 2019



Eka Sunandika, 13517130