APLIKASI PENCARIAN JALUR TERPENDEK UNTUK MENEMUKAN LOKASI ATM DI KOTA MALANG

Tommy Purwantoro Nugroho[1], Erfan Rohadi[2], Arief Prasetyo[3]

Program Studi Teknik Informatika, Jurusan Teknologi Informasi, Politeknik Negeri Malang
[1] tommy.purwantoro@gmail.com, [2] erfanr@polinema.ac.id, [3] arief.prasetyo@polinema.ac.id

Abstrak

Pendatang dan wisatawan di Kota Malang seringkali kesulitan dalam menemukan lokasi ATM terdekat untuk melakukan transaksi keuangan termasuk mengambil uang cash. Mereka belum mengetahui rute jalur terpendek untuk mencapai lokasi ATM. Skripsi ini, mengembangkan sebuah aplikasi yang bisa memberikan informasi mengenai ATM yang akan dituju dan bisa mengoptimalkan pencarian rute terpendek dengan menggunakan algoritma A* yang menerapkan heuristic search. Pencarian tersebut mempunyai informasi tentang cost/biaya untuk mencapai tujuan dari lokasi awal. Aplikasi ini juga telah diuji coba dengan cara menyebarkan kuesioner kepada 20 responden. Hasilnya 90 % responden menyatakan bahwa aplikasi ini friendly user, memiliki proses yang cepat dalam pencarian rute dan membantu pengguna menemukan ATM. Secara umum 80% responden merasakan manfaat aplikasi yang dikembangkan.

Kata Kunci : rute terpendek, ATM, Algoritma A*, *Heuristic search*

1. Pendahuluan

Malang merupakan kota pendidikan dan kota pariwisata. Banyak tempat pendidikan seperti perguruan tinggi dan objek wisata yang berpotensial untuk menarik pendatang. Seiring dengan perkembangan Kota Malang, semakin banyak pula pendatang dan wisatawan di Kota Malang. Namun, bagi pendatang atau wisatawan tersebut, mereka sering tidak membawa uang cash dan membutuhkan informasi mengenai ATM di Kota Malang.

Beberapa pendatang atau wisatawan di Kota Malang sering menggunakan mobil atau kendaraan pribadi untuk berkendara. Kendaraan pribadi dianggap lebih efisien karena bisa berhenti dan pergi ke rute manapun sesuai keinginan pengguna. Salah satunya yaitu untuk mengambil uang di ATM. Di tengah perjalanan mereka akan membutuhkan ATM untuk mengambil uangnya. Namun permasalahan yang datang adalah untuk orang yang baru pertama kali ke Malang seringkali mengalami kesulitan menemukan lokasi ATM terdekat untuk melakukan transaksi keuangan termasuk mengambil uang cash. Mereka belum mengetahui rute jalur terpendek untuk mencapai lokasi ATM. Pendatang masih kesulitan dalam mencari jalur alternatif tersebut. Mereka akan kehabisan banyak waktu dan tenaga di perjalanan jika belum benar dalam menentukan rutenya menuju lokasi ATM. Masalah lain yaitu mereka juga tidak mengetahui nominal pecahan uang yang ada di ATM yang dituju. Sebagian besar orang mempertimbangkan hal tersebut dalam memilih ATM mana yang akan dituju. Maka informasi tentang rute terpendek untuk menuju ke ATM terdekat menjadi suatu kebutuhan.

Makalah ini akan membahas bagaimana mengembangkan sebuah aplikasi yang mampu menampilkan rute terpendek untuk mencapai lokasi ATM yang sesuai dengan keinginan *user*.

2. Landasan Teori

2.1 Pencarian Rute Terpendek

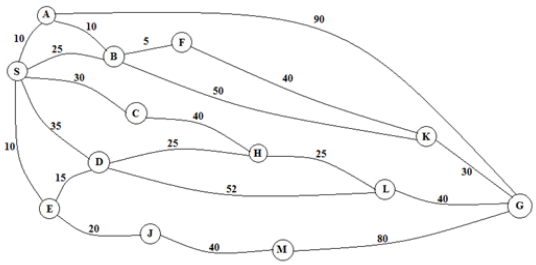
Dalam Kehidupan sehari-hari kebanyakan orang sering melakukan perjalanan dari suatu tepat ke tempat lain yang akan dituju dengan menggunakan jalan yang sama. Untuk mengoptimalkan jarak perjalanan maka sangat diperlukan penentuan jalur terpendek dari tempat asal ke tempat tujuan. Pencarian jalur terpendek merupakan permasalahan untuk menemukan jalur diantara dua node yaitu node awal dan node tujuan. Node akan merepresentasikan lokasi pada peta sedangkan bobotnya akan merepresentasikan jarak yang dibutuhkan untuk melakukan perjalanan antara dua tempat tersebut. Banyak metode atau algoritma yang digunakan untuk mencari rute terpendek. Salah satunya adalah Algoritma A*.

2.2 Algoritma A*

Algoritma ini merupakan algoritma Best First Search yang menggabungkan Uniform Cost Search dan Greedy Best-First Search. Jarak yang diperhitungkan didapat dari jarak sebenarnya (actual cost) ditambah dengan jarak perkiraan (estimated cost) (Suyanto, 2014). Dalam notasi matematika dituliskan sebagai:

$$f(n) = g(n) + h(n) \quad \text{.....(1)}$$

Dengan perhitungan jarak seperti ini, algoritma A* adalah complete dan optimal. Misalkan terdapat 13 kota yang dinyatakan oleh simpul-simpul dalam suatu graph dua arah. Setiap angka pada busur menyatakan jarak sebenarnya (actual cost) antara satu kota dengan yang lainnya. Nilai h(n) adalah fungsi heuristik, yaitu jarak garis lurus dari simpul n menuju simpul G dalam satuan kilometer.



Gambar 1. Contoh kasus algoritma A*

Berikut ini adalah tabel nilai h(n) yang menyatakan jarak perkiraan dari setiap simpul-simpul menuju simpul G :

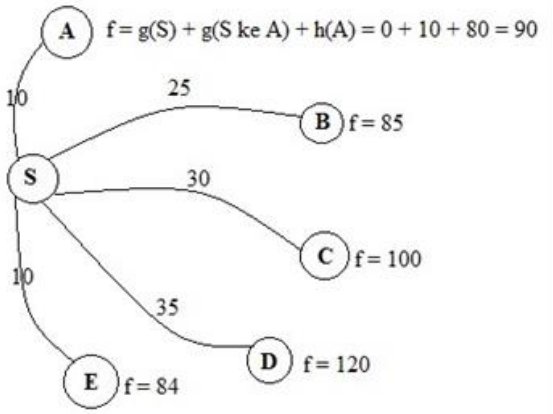
Tabel 1. Nilai h(n) contoh kasus algoritma A*

N	S	A	B	C	D	E	F	G	H
h(n)	80	80	60	70	85	74	70	0	40

J	K	L	M
100	30	20	70

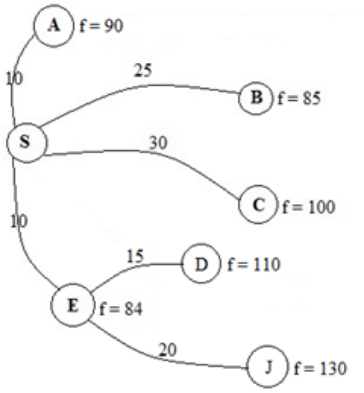
Langkah-langkah pencarian rute berdasarkan algoritma tersebut adalah sebagai berikut:

1. Langkah pertama, karena di OPEN hanya terdapat satu simpul (S), maka S terpilih sebagai BestNode dan dipindahkan ke CLOSED. Kemudian bangkitkan semua suksesor S, yaitu: A, B, C, D, E. Karena kelima suksesor tidak ada di OPEN maupun CLOSED, maka kelimanya dimasukkan ke OPEN. Menghasilkan OPEN = [A, B, C, E] dan CLOSED = [S].



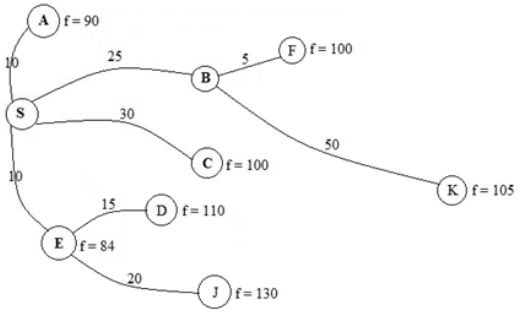
Gambar 2. Langkah pertama contoh kasus algoritma A*

2. Langkah kedua, E dengan biaya terkecil (yaitu 84) terpilih sebagai BestNode dan dipindahkan ke CLOSED. Selanjutnya, semua suksesor E dibangkitka, yaitu: D dan J. Karena belum pernah ada di OPEN maupun CLOSED, maka J dimasukkan ke OPEN. Sedangkan simpul D sudah ada di OPEN, maka harus dicek apakah parent dari D perlu diganti atau tidak, Ternyata, jarak dari S ke D melalui E (yaitu 10 + 15 = 25) lenih kecil daripada jarak dari S ke D (yaitu 35). Oleh karena itu, parent dari D harus diubah, yang semula S menjadi E. Dengan perubahan parent ini, maka nilai g dan f pada D juga diperbarui (nilai g yang semula 25 menjadi 35, dan nilai f dari 120 menjadi 110). Langkah kedua ini menghasilkan OPEN = [A, B, C, D, J] dan CLOSED = [S, E].



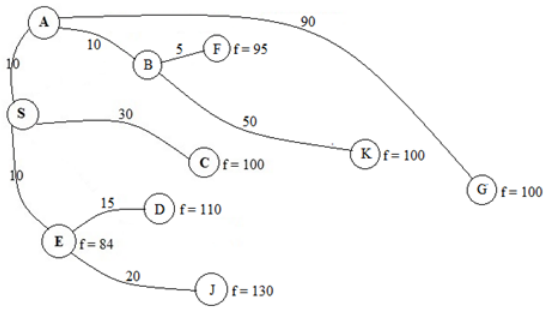
Gambar 3. Langkah kedua contoh kasus algoritma A*

3. Langkah ketiga, B dengan jarak terkecil (yaitu 85) terpilih sebagai BestNode dan dipindahkan ke CLOSED. Selanjutnya, semua suksesor B dibangkitkan, yaitu: A, F dan K. Karena belum pernah ada di OPEN maupun CLOSED, maka F dan K dimasukkan ke OPEN. Sedangkan simpul A sudah ada di OPEN, maka harus dicek apakah parent dari A perlu diganti atau tidak. Ternyata, jarak dari S ke A melalui B (yaitu 25 + 10 = 35) lebih besar daripada jarak dari S ke A (yaitu 10). Oleh karena itu, parent dari A tidak perlu diubah (tetap S), Akhir dari langkah ketiga ini menghasilkan OPEN = [A, C, D, F, J, K] dan CLOSED = [S, E, B].



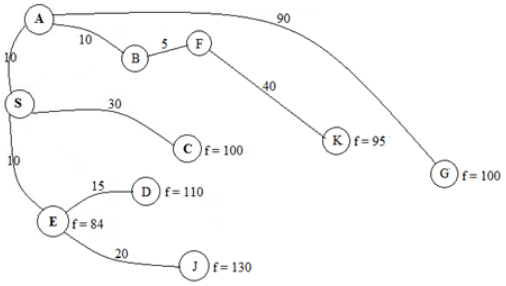
Gambar 4. Langkah ketiga contoh kasus algoritma A*

4. Langkah keempat, A dengan jarak terkecil (yaitu 90) terpilih sebagai BestNode dan dipindahkan ke CLOSED. Selanjutnya, semua suksesor A dibangkitkan, yaitu: B dan G. Karena belum pernah ada di OPEN maupun CLOSED, maka G dimasukkan ke OPEN. Sedangkan simpul B sudah ada di CLOSED, maka harus dicek apakah parent dari B perlu diganti atau tidak. Ternyata, jarak dari S ke B melalui A (yaitu 10 + 10 = 20) lebih kecil daripada jarak dari S ke B (yaitu 25). Oleh karena itu, parent dari B harus diubah, yang semula S menjadi A, nilai g dan f pada B juga harus diperbarui (nilai g yang semula 25 menjadi 20, dan nilai f dari 85 menjadi 80). Dalam kasus ini, B hanya mempunyai dua jalur, yaitu F dan K. Nilai g(F) yang semula 30 diubah menjadi 25, dan nilai f(F) dari 100 menjadi 95. Nilai g(K) yang semula 75 diubah menjadi 70, dan nilai f(K) dari 105 menjadi 100. Akhirnya, OPEN = [C, D, F, G, J, K] dan CLOSED = [S, E, B, A].



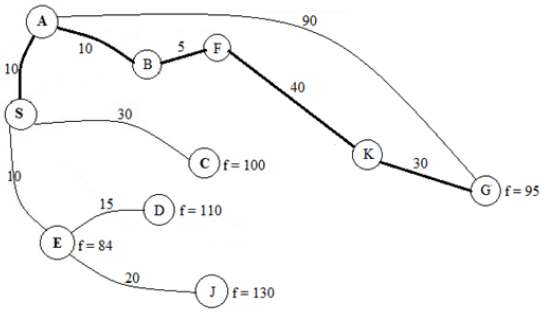
Gambar 5. Langkah keempat contoh kasus algoritma A*

5.Langkah kelima, F dengan jarak terkecil (yaitu 95) terpilih sebagai BestNode dan dipindahkan ke CLOSED. Selanjutnya, semua suksesor F dibangkitkan, yaitu: K. Karena K sudah ada di OPEN, maka harus dicek apakah parent dari K perlu diganti atau tidak. Jarak dari S ke K melalui F ternyata lebih kecil daripada jarak dari S ke K melalui parent lama (B). Oleh karena itu, parent dari K harus diubah, yang semula B menjadi F. Selanjutnya, nilai g(K) yang semula 70 diubah menjadi 65, dan nilai f(K) dari 100 menjadi 95. Akhirnya, OPEN = [C, D, F, G, J, K] dan CLOSED = [S, E, B, A, F].



Gambar 6. Langkah kelima contoh kasus algoritma A*

6. Langkah keenam, K dengan jarak terkecil (yaitu 95) terpilih sebagai BestNode dan dipindahkan ke CLOSED. Selanjutnya, semua suksesor K dibangkitkan, yaitu: G. Karena D sudah ada di OPEN, maka harus dicek apakah parent dari G perlu diganti atau tidak. Jarak dari S ke G melalui K ternyata lebih kecil daripada jarak S ke G melalui parent lama (A). Oleh karena itu, parent dari G harus diubah, yang semula A menjadi K. Selanjutnya, nilai g(G) yang semula 100 diubah menjadi 95, dan nilai f(G) dari 100 menjadi 95. Pada akhir langkah keenam ini, OPEN = [C, D, G, J] dan CLOSED = [S, E, B, A, F, K].



Gambar 7. Langkah keenam contoh kasus algoritma A*

7. Selanjutnya, G dengan jarak terkecil (yaitu 95) terpilih sebagai BestNode. Karena BestNode sama dengan goal, berarti solusi sudah ditemukan. Rute dan total jarak bisa ditelusuri balik dari G menuju S karena setiap simpul hanya memiliki satu parent dan setiap simpul memiliki informasi jarak sebenarnya (g). Penelusuran menghasilkan rute S-A-B-F-K-G dengan total jarak sama dengan 95.

2.3 Android

Android merupakan sebuah sistem operasi untuk perangkat mobile berbasis linux yang mencakup sistem operasi, middleware dan aplikasi. Android menyediakan platform terbuka bagi para pengembang untuk menciptakan aplikasi mereka. Android dipuji sebagai "platform mobile pertama yang Lengkap, Terbuka, dan Bebas".

- Lengkap (Complete Platform):

Android merupakan sistem operasi yang ama dan memiliki banyak tools untuk membangun sebuah perangkat lunak dan memungkinan untuk mengembangkan aplikasi.

- Terbuka (Open Source Platform):

Platform Android disediakan melalui lisensi open source. Pengembang dapat mengembangkan aplikasi dengan bebas.

- Bebas (Free Platform):

Android dapat dikembangkan secara bebas dan gratis. Tidak terdapat lisensi atau biaya royalti untuk dikembangkan dalam platform android, tidak ada biaya keanggotaan, tidak memerlukan biaya pengujian, dan tidak ada kontrak yang diperlukan.

2.4 Google Maps API

Google Maps API adalah kumpulan API yang memungkinkan untuk menghamparkan data di Google Map yang disesuaikan. Dengan menggunakan Google Maps dapat membuat aplikasi web dan seluler yang menarik dengan platform pemetaan canggih dari Google, termasuk data citra satelit, Street View, profil ketinggian, petunjuk arah mengemudi, peta bergaya, analisis, dan basis data tempat yang luas.

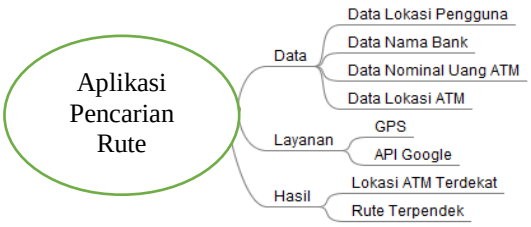
3 Pembahasan

3.1 Analisa sistem

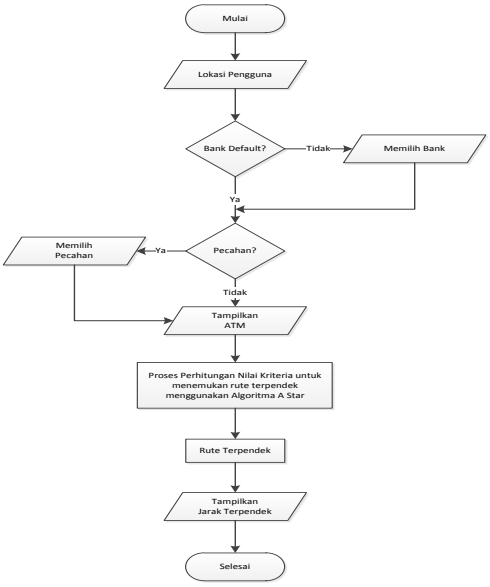
Analisa sistem merupakan tahapan yang akan menentukan benar atau tidaknya langkah selanjutnya dalam pembuatan aplikasi. Tujuan dari analisa sistem antara lain adalah untuk mempelajari aktivitas sistem untuk mendapatkan gambaran yang menyeluruh tentang sistem yang sedang berjalan dan permasalahan yang terjadi serta analisa dari algoritma yang digunakan.

3.2 Perancangan Sistem

Penulis melakukan 4 proses analisis yaitu, analisis data yang digunakan untuk penelitian berupa data lokasi pengguna, data nama Bank, data nominal uang ATM serta data lokasi ATM, analisis Database yang sesuai dengan data yang ada, analisis interface, dan analisis sistem yang keduanya mengacu pada WBS dan *flowchart*.
WBS dan *Flowchart* yang penulis gunakan yaitu sebagai berikut:



Gambar 8. Work Breakdown Structure



Gambar 9. Flowchart Sistem

4 Implementasi

4.1 Implementasi Algoritma A*

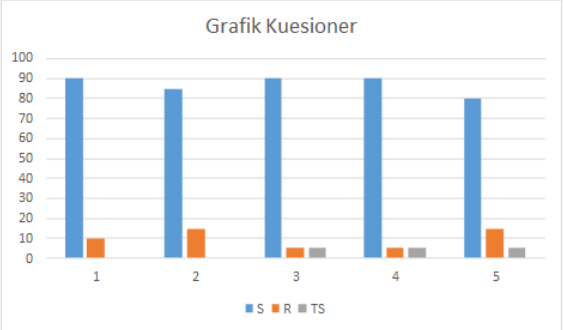
Algoritma A* merupakan perbaikan dari metode BFS dengan memodifikasi fungsi heuristiknya. A* (A Star) akan meminimumkan total biaya lintasan. Pada kondisi yang tepat, A* akan memberikan solusi yang terbaik dalam waktu yang optimal.

Algoritma A* bekerja dengan prinsip yang hampir sama dengan BFS, kecuali dengan dua perbedaan, yaitu :

1. Simpul-simpul di list “terbuka” diurutkan oleh cost keseluruhan dari simpul awal ke simpul tujuan, dari cost terkecil sampai cost terbesar. Dengan kata lain, menggunakan priority queue (antrian prioritas). Cost keseluruhan dihitung dari cost dari simpul awal ke simpul sekarang (current node) ditambah cost perkiraan menuju simpul tujuan.
2. Simpul di list “tertutup” bisa dimasukkan ke list “terbuka” bila jalan terpendek (cost lebih kecil) menuju simpul tersebut ditemukan.

4.2 Hasil Uji Coba dengan Responden

Untuk menguji sistem dari segi tampilan, *friendly user*, kehandalan atau keakurasian serta manfaat dari aplikasi yang dikembangkan, telah dilakukan penyebaran kuesioner kepada 20 responden. Responden yang dimaksud adalah para pengguna jasa layanan ATM bank, diantaranya ialah Bank BRI, Bank BNI, dan Bank Mandiri di Kota Malang. Pendapat dari responden tersebut ditampilkan pada Gambar



Gambar 10. Grafik presentase Kuesioner

5 Kesimpulan dan Saran

5.1 Kesimpulan

Berdasarkan hasil dari perancangan Aplikasi Pencarian Jalur Terpendek Untuk Menemukan Lokasi ATM (Anjungan Tunai Mandiri) Terdekat di Kota Malang Berbasis Android, dapat di ambil beberapa kesimpulan, yaitu: dengan menggunakan data yang dihasilkan dari uji coba dan dari hasil kuesioner yang disebarkan kepada 20 responden, aplikasi mampu berjalan dengan baik secara fungsional untuk menampilkan rute terdekat ATM di Kota Malang menggunakan algoritma A*(A star), dapat membantu pendatang dan wisatawan yang datang ke Kota Malang untuk menuju lokasi ATM berdasarkan kriteria yang diinginkan. 90 % responden menyatakan bahwa aplikasi ini *friendly user*, memiliki proses yang cepat dalam pencarian rute dan membantu pengguna menemukan ATM. Secara umum 80% responden merasakan manfaat aplikasi yang dikembangkan.

5.2 Saran

Dalam menguji aplikasi ini dapat digunakan metode lain seperti metode genetika dan metode *ant colony* untuk dijadikan perbandingan keakurasian dalam penelitian lebih lanjut.

Daftar Pustaka:

Sormin, M., (2014): *Perancangan Aplikasi Pencarian Jalur Terpendek Menggunakan Algoritma A**, Pelita Informatika Budi Darma, Volume: VI, Nomor: 3, April 2014, ISSN: 2301-9425, Jurnal Mahasiswa Program Studi Teknik Informatika, STMIK Budidarma Medan.

Ilham, R., dkk., (2011): *Pengembangan Aplikasi Pencarian Rute Terpendek Dengan Metode Algoritma A* Berbasis Web*, Jurnal Elektro ELTEK Vol. 2, No. 2, Oktober 2011. ISSN: 2086-8944, Jurnal Mahasiswa Jurusan Teknik Elektro, Institut Teknologi Nasional Malang.

Mutiana, V., dkk, (2013): *Optimasi Pencarian Jalur dengan Metode A-Star. Studi Kasus: Area Gading Serpong, Tangerang*, ISSN 2085-4552, Jurnal Mahasiswa Program Studi Teknik Informatika, Universitas Multimedia Nusantara, Tangerang, Indonesia.

Zaki, A., (2008):*Seri Penuntun Praktis AJAX untuk Pemula*,.Jakarta,Elex Media Computindo.

Prasetyo, D., (2007): *150 Rahasia Pemrograman Java*, Jakarta, Elex Media Computindo.

Swansen, R. C., (1995): *The Quality Improvement Handbook: Team Guide to Tools and Techniques*, Washington, D. C., CRC PRESS

Juran, J. M., (1999): J*uran's Quality Handbook (5th Edition)*, McGraw-Hill.

Suyanto, (2010): *Artificial Intellegence*, Bandung, Informatika.

Mufti, Y., (2015): *Panduan Mudah Pengembangan Google Map Android*, Yogyakarta, ANDI OFFSET.