PEMILIHAN RUTE TERPENDEK PASIEN UNTUK PENANGANAN COVID-19 DI JAKARTA MENGGUNAKAN ALGORITMA STEEPEST ASCENT HILL CLIMBING

SHORTEST ROUTE SELECTION OF PATIENTS FOR HANDLING COVID - 19 IN JAKARTA USING STEEPEST ASCENT HILL CLIMBING ALGORITHM

Muhammad Dendi Agus Santoso[1], Purba Daru Kusuma[2], Casi Setia Ningsih[3]

[1,2,3] Universitas Telkom, Bandung
dendisantoso@student.telkomuniversity.ac.id[1], **purbodaru@telkomuniversity.ac.id**[2], **setiacasie@telkomuniversity.ac.id**[3]

Abstrak

Kasus Covid-19 di DKI Jakarta sampai saat ini masih terus meningkat, dimana setiap harinya selalu ada penambahan kasus positif Covid-19. Sehingga diperlukan penanganan yang cepat dan responsif. Untuk mendukung hal tersebut diperlukan informasi mengenai fasilitas kesehatan mana saja yang siap melayani pasien Covid-19 di DKI Jakarta.

Aplikasi ini akan menampilkan fasilitas mana saja yang siap melayani pasien Covid-19 serta menentukan rute yang harus dilewati untuk sampai ke lokasi tujuan menggunakan algoritma steepest ascent hill climbing. Dengan adanya aplikasi ini diharapkan dapat dimanfaatkan untuk melakukan penanganan Covid-19 menjadi lebih cepat karena pasien dapat menemukan fasilitas kesehatan yang sesuai dengan kebutuhan pasien sehingga dapat mempersingkat waktu untuk mendapatkan penanganan.

Dari hasil pengujian aplikasi penentuan rute terdekat dari lokasi pasien menuju ke fasilitas terkait covid-19 dengan penerapan algoritma steepest ascent hill climbing dari 10 kali pengujian didapatkan hasil semua tujuan dapat dirutekan dengan baik jika, komposisi SAW 80% jarak dan 20% kemacetan. Rata – rata memory yang diperlukan untuk menentukan rute menggunakan algoritma steepest ascent hill climbing sebesar 84,88 MB dan waktu yang diperlukan untuk menentukan rute adalah 00:01.648 (1648 ms).

Kata kunci : Covid-19, jarak, aplikasi, android, steepest ascent hill climbing.

Abstract

Covid-19 cases in DKI Jakarta are still increasing, where every day there are always additional positive cases of Covid-19. So it requires fast and responsive handling. To support this, information is needed about which health facilities are ready to serve Covid-19 patients in DKI Jakarta.

This application will display which facilities are ready to serve Covid-19 patients and determine the route that must be passed to get to the destination location using the steepest ascent hill climbing algorithm. With this application, it is hoped that it can be used to handle Covid-19 faster because patients can find health facilities that suit the patient's needs so that it can shorten the time to get treatment.

From the test results of the application to determine the closest route from the patient's location to the Covid-19-related facility with the application of the steepest ascent hill climbing algorithm, from 30 tests, it was found that all destinations can be routed with a SAW if composition 80% distance and 20% congestion. The average memory required to determine the route using the steepest ascent hill climbing algorithm is 84.88 MB and the time required to determine the route is 00: 01,648 (1648 ms).

Keywords: Covid-19, distance, aplication, android, steepest ascent hill climbing.

1. Pendahuluan

DKI Jakarta merupakan provinsi dengan jumlah kasus Covid-19 tertinggi di Indonesia dengan jumlah kasus mencapai lebih dari 100.000 kasus dan menyumbang 25% dari seluruh kasus Covid-

19 di Indonesia. Untuk mendukung penanganan Covid-19, Pemprov DKI Jakarta telah melakukan banyak hal terutama dalam hal penyampaian informasi kepada masyarakat dengan membuat website resmi corona.jakarta.go.id yang memuat berbagai informasi dan perkembangan Covid-19 di DKI Jakarta. Website tersebut juga memuat daftar rumah sakit rujukan Covid-19 yang dapat diakses oleh masyarakat.

Namun pasien tidak mengetahui fasilitas umum mana yang dapat melayani kebutuhannya, misalnya pasien hanya ingin melakukan rapid test, swap test atau hanya ingin melakukan isolasi mandiri. Selain itu, pasien juga tidak dapat langsung menentukan tempat mana yang paling dekat dengan lokasinya, sehingga diperlukan aplikasi pihak ketiga seperti google maps untuk memperkirakan fasilitas mana yang terdekat dan mana rute yang tercepat. Metode ini memerlukan beberapa langkah yang perlu dilakukan diantaranya perlunya mencari informasi fasilitas mana saja yang memberikan pelayanan sesuai keinginan pasien, kemudian memasukkan beberapa nama atau alamat dari fasilitas umum tersebut untuk dapat membandingkan mana yang lebih dekat. Hal ini tentu memakan banyak waktu.

Maka untuk mempermudah atau mempersingkat proses pemilihan fasilitas terkait Covid-19 sesuai keinginan pasien, pada tugas akhir ini dibuat aplikasi yang dapat menampilkan fasilitas umum mana saja yang dapat melayani Covid-19. Pasien juga akan mendapatkan rute perjalanan terdekat dengan fasilitas Covid-19 yang telah dipilih oleh aplikasi. Sehingga dengan adanya aplikasi ini diharapkan dapat mempercepat penanganan pasien Covid-19 di DKI Jakarta.

2. Dasar Teori

2.1 Jalur Terpendek

Salah satu masalah dalam graf adalah masalah pencarian jalur terpendek. Jalur terpendek mengacu pada jalur yang membutuhkan waktu paling sedikit untuk mencapai node tujuan dari node awal dalam satuan waktu [2]. Untuk mengatasi masalah ini digunakan graf berbobot, yaitu graf yang setiap sisinya memiliki nilai atau bobot. Nilai atau bobot ini dapat mewakili jarak antar kota, waktu tempuh, biaya perjalanan, atau lainnya [3].

2.2 Pencarian Jalur Terpendek

Untuk mencari jalur terpendek dapat diselesaikan dengan dua metode, Metode Konvensional merupakan algoritma dengan perhitungan matematis biasa. Pada metode konvensional terdapat beberapa algoritma untuk menentukan jalur terpendek, diantaranya adalah algoritma Djikstra, Floyd-Warshall, dan Belman-ford [4]. Metode Heuristik dalam konteks yang luas dapat diartikan sebagai suatu nilai informasi yang "dianggap" mendekati nilai pemecahan suatu masalah. Nilai heuristik bisa sangat relatif berdasarkan penilaian masing-masing pengambil keputusan [5].

2.3 Haversine Formula

Harversine formula merupakan persamaan yang penting dalam suatu navigasi, haversine memberikan jarak antara dua titik pada bola dari garis bujur dan garis lintang [6].

$$D = 2 \times sin^{-1}\left(\sqrt{\sin\left(\frac{lat2 - lat1}{2}\right)^2 + \sin\left(\frac{lon2 - lon1}{2}\right)^2 \times \cos lat1 \times \cos lat2}\right) \times \mu E \qquad (1)$$

Dimana:

lat1 = Latitude dari lokasi pengguna
lat2 = Latitude dari Node
lon1 = Longitude dari lokasi pengguna
lon2 = Longitude dari Node
μE = mean R radius bumi (6,371km)
D = Jarak antar lokasi pengguna dan Node

2.4 Simple Additive Weighting (SAW)

Metode SAW sering juga dikenal dengan metode penjumlahan berbobot. Konsep dasar dari metode SAW adalah mencari penjumlahan terbobot dari penilaian kinerja pada setiap alternatif pada suatu atribut [7].

$$V_i = \sum_{j=1}^{n} W_j R_{ij}$$

Nilai Vi yang lebih besar mengindikasikan bahwa alternatif Ai lebih terpilih.
Dimana :

Vi = Nilai akhir dari alternatif
Wi = Bobot yang telah ditentukan
Rij = Normalisasi matriks

2.5 Algoritma Hill Climbing

Algoritma hill climbing adalah algoritma yang menggunakan fungsi heuristik untuk melakukan optimasi dari sebuah permasalah. Cara kerja algoritma hill climbing dilakukan dengan cara membandingkan nilai sekarang dengan nilai selanjutnya. Apabila nilai selanjutnya lebih besar maka jalur tidak akan berubah sampai menemukan nilai optimal [8]. Algoritma hill climbing membandingkan semua tetangga dari state sekarang sebelum memilih gerakan selanjutnya [9].

Algoritma Hill Climbing mempunyai dua kelemahan:

1. Plateau, kondisi ketika ada dua atau lebih evaluation state yang mempunyai nilai heuristic sama besar dan juga merupakan nilai terbaik.
2. Local Maxima, yaitu solusi lokal yang ditemukan dengan fungsi evaluasi. Tetapi solusi ini bukan kondisigoal yang diharapkan, dan jika dilakukan evaluasi secara terus menerus, maka akan kembali lagi ke kondisi solusi lokal itu sendiri [5].

2.6 Simple Hill Climbing (SHC)

simple hill climbing dilakukan dengan cara melakukan evaluasi state sekarang, jika state awal merupakan tujuan maka proses berhenti tetapi apabila state sekarang buka tujuan pencarian akan dilanjutkan dengan mencari sebuah operator yang belum pernah digunakan dalam state sekarang. Gunakan operator tersebut sebagai state baru, jika nilai state baru lebih kecil dari nilai state sekarang gunakan state baru sebagai state sekarang [10]. simple hill climbing dapat dijalankan dalam tiga langkah. Pertama set state yang telah ditentukan sebelumnya, kedua jika state tetangga lebih baik dari state sekarang, state tetangga akan menjadi state baru saat ini, ketiga ulangi langkah kedua hingga mencapai state tujuan [11].

2.7 Steepest Ascent Hill Climbing (SAHC)

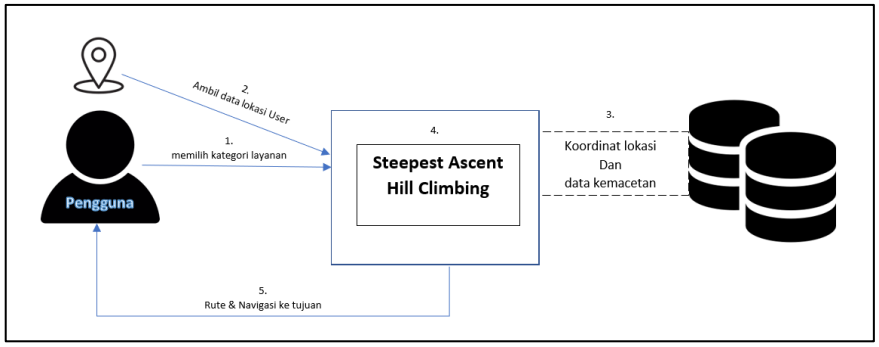
Steepest Ascent Hill Climbing merupakan pengembangan dari simple hill climbing. Perbedaannya jika pada simple hill climbing hanya membandingkan current state dengan state tetangga dan memilih state mana yang bernilai lebih kecil. Sedangkan pada Steepest Ascent Hill Climbing memperhitungkan nilai heuristik [11].

Steepest Ascent Hill Climbing diawali dari keadaan awal (initial state) yang telah di tentukan. Kemudian dilakukan pengujian dengan menghitung total jarak yang ditempuh. Pengujian tersebut dilakukan untuk mendapatkan nilai heuristik sebagai pembanding antar keadaan awal dengan tujuan berikutnya. Setelah di tentukan initial state sebagai current state, langkah berikutnya melakukan penukaran node hingga mencapai kondisi goal. Kondisi goal diartikan sebagai solusi optimal dimana : Nilai Next State < Nilai Curent State [12].

3. Perancangan dan Pengujian

3.1. Gambaran Umum Sistem

Aplikasi ini dapat digunakan pasien Covid-19 untuk mencari fasilitas umum sesuai kriteria yang di inginkan pasien beberapa kriteria yang tersedia diantaranya fasilitas kesehatan yang menyediakan tes Covid-19, rumah sakit rujukan covid-19, atau fasilitas isolasi mandiri. Berikut adalah gambaran umum dari sistem yang akan dibuat.

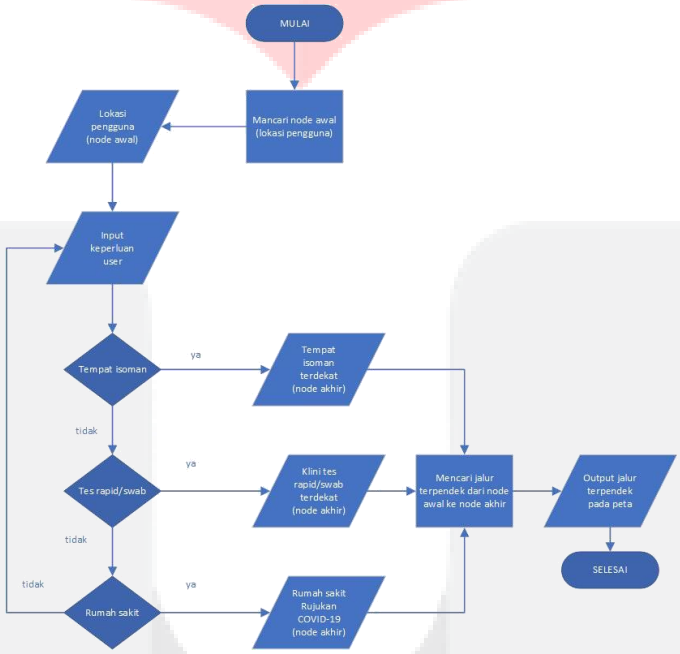


Gambar 1 Gambaran Umum Sistem

Dari gambar 1 dapat dilihat alur sistem yang terbagi menjadi 5 langkah dengan penjelasan sebagai berikut:

1. Pertama pengguna akan masuk ke menu pilih kategori yang berisi layanan tes Covid-19, Tempat isolasi mandiri, dan rumah sakit rujukan Covid-19.
2. Kemudian koordinat lokasi user akan dikirim ke sistem yang kemudian akan digunakan sebagai titik awal.
3. sistem mengambil data koordinat lokasi fasilitas sesuai kriteria yang dipilih pengguna serta mengambil data kemcetan dari database.
4. Selanjutnya akan dilakukan penentuan rute dengan menggunakan algoritma Steepest ascent hill climbing.
5. hasil dari proses algoritma Steepest ascent hill climbing akan ditampilkan kepada pasien dalam bentuk peta rute menuju lokasi tujuan.

3.2. Flow chart Sistem



Gambar 2 Flow Chart Sistem

Pada gambar 2 menjelaskan gambaran bagaimana sistem akan bekerja. Sistem akan memulai berjalan dengan mencari node awal berupa lokasi pengguna yang diambil dari data gps yang ada pada smarthphone pengguna. Selanjutnya pengguna akan diminta untuk melakukan input berupa pilihan fasilitas yang akan dituju atau yang akan dilakukan pencarian rutenya. Dalam aplikasi ini pilihan dibagi menjadi tiga, yaitu : tempat yang menyediakan fasilitas isolasi mandiri, fasilitas kesehatan yang dapat melakukan Tes Covid-19, dan Rumah sakit Rujukan Covid-19 di DKI Jakarta. Setelah penggunaa memilih fasilitas yang diinginkan sistem akan menjalankan algoritma *Steepest ascent hill climbing* untuk melakukan pencarian rute terdekat dengan memperhatikan variable berupa jarak pasien dengan tempat tujuan. kemudian hasil dari algoritma *hill climbing* akan ditampilan dalam bentuk *map* yang dapat dilihat pasien pada aplikasi android. Setelah rute ditampilkan, pasien juga dapat melakukan navigasi menuju tujuan yang telah dipilih oleh pengguna.

3.3. Implementasi Sistem

Berdasarkan Rancangan bab sebelumnya, telah dilakukan implementasi sebagai berikut :

A. Splash Screen

Pada awal memulai aplikasi akan di tampilkan *splash screen* selama dua detik sebelum aplikasi masuk ke menu utama.

Gambar 3 menampilkan [illegible]kasi. Splash screen berisi logo dari aplikasi SICO yang tampil selaman du[illegible]gsung masuk ke halaman menu utama.

Gambar 3 Splash screen

B. Menu Utama

Menu utama pada aplikasi ini adalah tampilan dengan yang berisi dengan tiga menu yaitu menu Tempat Tes, Tempat Isoman, dan RS Rujukan.

Gambar 4Tampilan Menu utama

Gambar 4 menunjukkan tampilan menu setelah splash screen selesai. Pada menu ini berisi tiga menu yang dapat dipilih oleh pengguna yaitu Tempat Tes Covid-19 yang akan manempilkan daftar dari lokasi tempat tes Covid-19, menu Tempat Isolasi Mandiri akan menampilan daftar hotel yang menyediakan fasilitas isolasi mandiri, sedangkan menu RS Rujukan Covid -19 akan menampilkan daftar rumah sakit rujukan yang ada di DKI Jakarta.

C. Daftar Tempat

Daftar tempat ditampilkan dengan menggunakan recyclerview yang dimana data berupa nama lokasi didapatkan dari database yang telah di hubungkan dengan aplikasi

Gambar 5 menunjukkan tampilan dari daftar tempat sesuai menu yang dipilih pengguna pada menu utama. Pada halaman ini daftar juga diurutkan mulai dari fasilitas yang terdekat dari posisi pengguna aplikasi berada.

D. Konfirmasi Tujuan

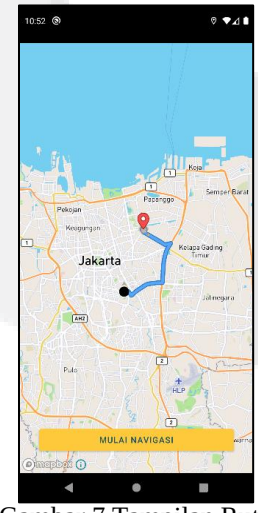
Setelah memilih tujuan dari halaman sebelumnya, pengguna akan masuk ke halaman konfirmasi tujuan. Dihalaman ini pengguna akan ditampilkan tujuan yang telah dipilih

Gambar 6 Konfirmasi tujuan

Gambar 6 Menampilkan Halaman Konfirmasi tujuan yang berisi tujuan yang telah dipilih pada halaman sebelumnya.

E. Rute map

Tampilan map akan muncul Ketika pengguna telah memilih tujuan dan mengkonfirmasi tujuan. Tampilan map pada aplikasi ini menggunakan MapBox api yang sekaligus juga menampilkan rute yang akan dilewati pengguna.



Gambar 7 Tampilan Rute

Gambar 7 Menampilkan rute yang akan dilalui pengguna mulai dari awal lokasi pengguna berada menuju lokasi tujuan yang telah di pilih oleh pengguna.

3.4. Pengujian Aplikasi

Pengujian Aplikasi dilakukan untuk mengetahui berapa resource yang dibutuhkan aplikasi untuk mencari rute berdasarkan algoritma yang digunakan. Dari pengujian ini dapat mengetahui minimum memory yang dibutuhkan untuk menjalankan aplikasi ini pada sebuah smart phone android. Berikut adalah hasil pengujian Aplikasi.

Table 1 Pengujian Aplikasi

Aspek Pengujian	Percobaan ke-					Rata -Rata
	1	2	3	4	5	
Memory (MB)	80,1	76,9	82,1	103	82,3	84,88
Waktu	00:01.813	00:02.186	00:01.464	00:01.301	00:01.417	00:01.648

Dari 5 kali percobaan pencarian rute, didapatkan hasil seperti pada table 1. memory rata – rata memory yang digunakan untuk proses penentuan rute menggunakan algoritma s*teepest ascent hill climbing* sebesar 84,88 MB. Sedangkan waktu yang dibutuhkan untuk memproses rute terpilih rata - rata adalah 00:01.648.

3.5. Pengujian SAW

Pengujian SAW dilakukan untuk mengetahui perbandingan bobot terbaik antara jarak dan kemacetan. Pengujian ini menguji apakah bobot pada jarak dan kemacetan berpengaruh terhadapa pemilihan rute dengan algoritma steepest ascent hill climbing. Dalam pengujian ini dilakukan pada waktu hari rabu dengan lokasi pengguna berada di Casa Jardin Residence, Jalan Daan Mogot KM. 11, RT.1/RW.4, Kedaung Kali Angke, Kota Jakarta Barat, Daerah Khusus Ibukota Jakarta (-6.152943989022467, 106.75162512557554) Dan tujuan ke RS Pondok Indah Puri Indah. Dengan hasil pengujian sebagai berikut :

Table 2 Pengujian SAW

No.	Jam	Nilai SAW		
		50% jarak + 50% kemacetan	70% jarak + 30% kemacetan	80% jarak + 20% kemacetan
1	08:00	Gagal	Gagal	Berhasil
2	12:00	Gagal	Gagal	Berhasil
3	18:00	Gagal	Gagal	Berhasil
4	21:00	Gagal	Gagal	Berhasil

Berdasarkan percobaan pada tabel 2 dengan bobot jarak dan kemacetan yang berbeda dan waktu yang berbeda di dapatkan hasil bahwa algoritma steepest ascent hill climbing berhasil mencari rute ketika bobot jarak 80 % dan bobot kemacetan 20% untuk menghitung nilai SAW. Kegagalan menentukan rute diakibatkan ketika di awal pencarian rute current state memiliki nilai terkecil dibandingkan nilai succsessornya .

3.6. Pengujian Algoritma

Pada pengujian Algoritma dilakukan beberapa pengujian dengan lokasi pengguna yang berbeda, lokasi tujuan yang berbeda, dan waktu yang berbeda untuk melihat perbedaan setiap rutenya.

Table 3 Pengujian Algoritma

No.	Lokasi User	Tujuan	Waktu	Rute Terpilih	Ket.
1	SMK Cengkareng	RS Hermina Daan Mogot	12:00	B28 - B29 – B58 – B34 - B35	Berhasil

2	SMK Cengkareng	RS Mitra Keluarga Kalideres	12:00	B28 – B27 – B26 – B25 – B54	Berhasil
3	SMK Cengkareng	Lab Klinik Bioprima	12:00	B28 – B29 – B30	Berhasil
4	Casa Jardin Residen	Rs Pondok Indah - Puri Indah	12:00	B167 - B163 - B149 - B87 - B147	Berhasil
5	Casa Jardin Residen	Lab Rs Cendana	12:00	B167 – B163 – B149 – B98 – B99 – B145	Berhasil
6	SMK Cengkareng	RSUD Kali Deres	21:00	B28 – B19 – B18 – B14 – B12 – B11 – B10	Berhasil
7	SMK Cengkareng	Rs Ciputra	21:00	B28 – B19 – B18 – B14 – B12 – B11 – B10 – B7	Berhasil
8	Casa Jardin Residen	RS Bina Sehat Mandiri	21:00	B167 – B163 – B149 – B98 – B99 – B146 – B145 – B151	Berhasil
9	Casa Jardin Residen	Rsud Cengkareng	21:00	B167 – B60 – B52 – B44 -B42	Berhasil
10	Casa Jardin Residen	Rsud Kalideres	21:00	B167 – B60 – B52 – B44 – B47 – B46 – B20 -B10	Berhasil

Dari hasil pengujian pada tabel 3 sebanyak 10 pengungujian pencarian rute. Dapat disimpulkan bahwa algoritma steepest ascent hill climbing dapat digunakan untuk melakukan pencarian rute terpendek dengan bobot SAW adalah 80% jarak dan 20% kemacetan

4. Kesimpulan

Berdasarkan hasil pengujian dan analisis yang telah dilakukan pada bab sebelumnya, maka dapat ditarik kesimpulan bahwa Aplikasi penentuan rute terdekat dari lokasi pasien menuju ke fasilitas Covid-19 dengan penerapan algoritma steepest ascent hill climbing setelah dilakukan pengujian didapatkan hasil semua tujuan dapat dirutekan dengan baik apabila bobot SAW 80% jarak dan 20% kemacetan. Rata – rata memory yang diperlukan untuk menentukan rute menggunakan algoritma steepest ascent hill climbing sebesar 84,88 MB dan waktu yang diperlukan untuk menentukan rute adalah 00:01.648 (1648 ms).

REFERENSI

[1] Covid19.go.id,"Peta Sebaran", 23 November 2020. Available: https://covid19.go.id/peta-sebaran. acses on 23 November 2020, 09:00WIB]

[2] ZHANG, Dongbo; XIONG, Lu. The research of dynamic shortest path based on cloud computing. In: 2016 12th International Conference on Computational Intelligence and Security (CIS). IEEE, 2016. p. 452-455.

[3] R. Munir, Matematika Diskrit, Bandung: Informatika Bandung, 2010.

[4] M. Ling, S. Fajar, H. Nur, W. Romi, "Pemanfaatan Metode Heuristik Dalam Pencarian Jalur Terpendek Dengan Algoritma Semut Dan Algoritma Genetika". Presented at Seminar Nasional Aplikasi Teknologi Informasi 2007 (SNATI 2007), Yogyakarta, 2007.

[5] A. David, W.P Indra, A.N Rangga, A. Moch, Hanif, "Penyelesaian Masalah 8-Puzzle dengan Algoritma Steepest-Ascent Hill Climbing". SETRUM - volume 4, No. 1, Cilegon, Indonesia, 2015.

[6] CHOPDE, Nitin R.; NICHAT, Mangesh. Landmark based shortest path detection by using A* and Haversine formula. International Journal of Innovative Research in Computer and Communication Engineering, 2013, 1.2: 298-302.

[7] Frieyadie, "Penerapan Metode Simple Additive Weight (Saw)Dalam Sistem Pendukung Keputusan Promosi Kenaikan Jabatan". Jurnal Pilar Nusa Mandiri Vol.XII, No. 1, Jakarta, 2016

[8] Borghoff, Julia, Lars R. Knudsen, and Krystian Matusiewicz. "Hill climbing algorithms and trivium." International Workshop on Selected Areas in Cryptography. Springer, Berlin, Heidelberg, 2010.

[9] ZhOU, Peng, et al. "An improved hill climbing search algorithm for rosa coupling." 2018 2nd IEEE Advanced Information Management, Communicates, Electronic and Automation Control Conference (IMCEC). IEEE, 2018.

[10] Thiang, Handry Khoswanto, Felix Pasila, and Hendra Thelly. "Aplikasi Metode Hill Climbing Pada Standalone Robot Mobil Untuk Mencari Rute Terpendek."Jurnal Informatif, Jurusan Teknik Elektro, Universitas Kristen Petra 3, 2009.

[11] DOZIER, Gerry, et al. Artificial potential field based robot navigation, dynamic constrained optimization and simple genetic hill-climbing. In: 1998 IEEE International Conference on Evolutionary Computation Proceedings. IEEE World Congress on Computational Intelligence (Cat. No. 98TH8360). IEEE, 1998. p. 189-194.

[12] Irfan, Muhammad. "Penyelesaian Travelling Salesman Problem (TSP) Menggunakan Algoritma Hill Climbing dan MATLAB." Matematika 17.1 ,2018.