

PENERAPAN ALGORITMA A* (STAR) MENGGUNAKAN GRAPH UNTUK MENGHITUNG JARAK TERPENDEK

Ida Bagus Gede Wahyu Antara Dalem

Program Studi Ilmu Komputer, Pasca Sarjana Universitas Pendidikan Ganesha, Singaraja-Bali

e-mail: wahyuantara88@gmail.com

Received : April, 2018

Accepted : April, 2018

Published : April, 2018

Abstrak

Algoritma A* (Astar) merupakan salah satu algoritma yang termasuk dalam kategori metode pencarian yang memiliki informasi (*informed search method*). Algoritma ini sangat baik sebagai solusi proses *path finding* (pencari jalan). Algoritma ini mencari jarak rute terpendek yang akan ditempuh suatu point awal (*starting point*) sampai ke objek tujuan. Teknik pencarian yang digunakan dalam simulasi ini adalah menggunakan Algoritma A* dengan fungsi heuristik. Tujuan utama penelitian ini mempelajari cara kerja algoritma A* dalam mencari jarak tercepat, yang disimulasikan seperti kondisi ketika seorang mencari rute dalam keadaan jalanan macet. Simulasi ini memberikan gambaran yang lebih realistis terhadap perilaku algoritma A* dalam pencarian jarak rute terpendek.

Kata Kunci: Algoritma A* (Astar), Path Finding, Rute terpendek

Abstract

Algorithm A* (Astar) is one of the algorithm that belongs to category of searching method that has information (*informed searched method*). This algorithm is very good as process solution *path finding* (way seeker). This algorithm looks for the shortest route which will be taken a starting point to the next object. Collection data technique used in this simulation is by using Algorithm A* with heuristic function. The main purpose of this research is to learn the way of work Algorithm A* in looking for the fastest, which is simulated like a condition when someone looks for route in a traffic way. This simulation gives a realistic pictures for Algorithm A* behavior in looking for shortest route.

Keywords: Algorithm A* (Astar), Path Finding, Short Route

1. PENDAHULUAN

Didalam kehidupan sehari-hari perpindahan dari suatu tempat ke tempat lain sangat mempertimbangkan efisiensi waktu dan biaya, maka diperlukan pengetahuan untuk menentukan suatu jalur terpendek antar tempat yang ingin dituju. Algoritma pencarian jalur terpendek atau lebih dikenal sebagai *shortest-path* dipakai dalam menentukan rute dalam sebuah *graph* [1].

Graph adalah pasangan himpunan *vertex*/simpul dan *edges*/sisi, dimana setiap sisi berhubungan dengan satu atau dua buah simpul[2]. Dikatakan juga *Graf* adalah salah satu pokok bahasan Matematika Diskrit yang telah lama dikenal dan banyak diaplikasikan pada berbagai bidang[3]. Penerapan konektifitas *graph* dalam kehidupan nyata biasa ditemukan di berbagai bidang diantaranya penerapan di bidang telekomunikasi, bidang jaringan komputer,

bidang transportasi, jaringan listrik dan saluran air [4].

Teknik pencarian yang sering digunakan untuk menentukan jalur terpendek yaitu pencarian buta (*blind search*) dan pencarian *heuristik* (*heuristic search*). Pencarian buta cenderung lebih mudah dipahami dibandingkan pencarian heuristik, tetapi hasil pencarian yang diperoleh pencarian *heuristic* lebih variatif dan waktu pencarian solusi lebih cepat. Ada beberapa cara mencari optimasi jalur terpendek untuk menghubungkan satu tempat dengan tempat yang lain diantaranya adalah algoritma Dijkstra, algoritma Semut atau *Ant Colony*, algoritma Floyd Warshall, algoritma Bellman Ford, algoritma Distance Vector, algoritma Ford-Fulkerson dan algoritma A-Star. Algoritma tersebut dibuat untuk mencari cara yang paling efisien dan efektif untuk menganalisa pemilihan jalur tercepat dengan berbagai penambahan variable [5]. Salah satu metode pencarian jalur terpendek yang termasuk dalam pencarian heuristik adalah Algoritma A* [6]. Pada pembahasan makalah ini Algoritma A-Star akan menjadi fokus.

Algoritma ini pertama kali dideskripsikan pada tahun 1968 oleh Peter Hart, Nils Nilsson, dan Bertram Raphael. Dalam makalah mereka, disebut dengan Algoritma A. Lalu dengan optimasi heuristik, disebut dengan A Star (A *). Algoritma A-Star merupakan salah satu algoritma pencarian rute yang optimal dan komplit. Optimal berarti rute yang dihasilkan adalah rute yang paling baik dan komplit berarti algoritma tersebut dapat mencapai tujuan yang di harapkan [7]. Dalam penerapannya, Algoritma A-Star menggunakan jarak sebagai proses kalkulasi nilai terbaik. Dikatakan juga kecepatan pencarian rute terpendek menggunakan algoritma Dijkstra berbeda dengan algoritma Astar dimana Astar lebih cepat untuk proses pencarian rute terpendek dengan selisih waktu rata-rata 40 ms[8]. Algoritma A Star (A*) adalah algoritma pencarian terbaik dalam mencari jalur terpendek dengan perhitungan terkecil pada jalur dengan simpul awal menuju simpul akhir [9].

2. PEMBAHASAN

2.1 Graph

Definisi sebuah graf itu sendiri adalah sebagai pasangan himpunan (V, E) yang dalam hal ini:

V = himpunan tidak kosong dari vertex-vertex (vertices atau node), dan

E = himpunan sisi (*edges* atau *arcs*) yang menghubungkan sepasang *vertex*[14].

Graph adalah kumpulan simpul (*nodes*) yang dihubungkan satu sama lain melalui sisi/busur (*edges*)[11].

Suatu *Graf* G terdiri dari dua himpunan yaitu himpunan V dan himpunan E Terdapat beberapa istilah penting yang berkaitan dengan *graf*. Berikut ini didefinisikan beberapa terminologi yang sering digunakan: [10]

1. *Graf* tak berarah didefinisikan sebagai suatu pasangan berurutan (V, E) , $G = (V, E)$, yang mempunyai sifat-sifat sebagai berikut :
 - Komponen pertama, V adalah terbatas (finite), himpunan yang tidak kosong. Elemen V ini disebut verteks dari G.
 - Komponen kedua, E, adalah himpunan terbatas dari suatu himpunan. Setiap elemen E, adalah suatu himpunan yang terdiri dari tepat dua verteks yang berbeda. Elemen E disebut *edge* dari G
2. *Graf* G dikatakan berbobot (*weighted*) bila terdapat bilangan riil yang berasosiasi dengan masing-masing *edge* G. Bobot-bobot itu sendiri biasanya berfungsi pada saat dibuat suatu lintasan antar *verteks-verteks* pada *graf* tersebut.

Menurut [13] Teori *graf* merupakan sebuah topik dasar dalam dalam matematika diskrit dan ilmu komputer. Teori ini sudah berusia tua namun memiliki banyak terapan sampai saat ini. *Graf* digunakan untuk merepresentasikan objek-objek diskrit dan hubungan antar objek-objek tersebut. Representasi visual dari sebuah graf adalah dengan menyatakan objek dengan sebuah noktah, bulatan, atau titik, sedangkan hubungan antar objek dinyatakan dengan garis. Mencari rute yang paling dekat dari posisi awal hingga akhir dengan beban paling ringan atau sedikit dibandingkan dengan seluruh rute yang ada. Terdapat banyak algoritma yang dapat digunakan dalam pencarian rute terpendek. Salah satunya adalah algoritma A*[15].

2.2 Algoritma A* (A-Star)

2.2.1 Konsep Algoritma A* (A-Star)

(A Star) Dalam sains komputer, A* (dibaca "A star") adalah algoritma komputer yang digunakan secara luas dalam mencari jalur (*path finding*) dan grafik melintang (*graph traversal*), proses *plotting* sebuah jalur melintang secara efisien antara titik-titik, disebut *node*. Terkenal karena penampilannya dan akurasi, algoritma ini diperluas untuk berbagai bidang. A* mencapai penampilan yang lebih baik dengan menggunakan *heuristik*. A* menggunakan *Best First Search* (BFS) dan menemukan jalur dengan biaya terkecil (*least-cost path*) dari *node* awal (*initial node*) yang diberikan ke *node* tujuan (*goal node*). Algoritma ini menggunakan fungsi heuristik jarak ditambah biaya (biasa dinotasikan dengan $f(x)$) untuk menentukan urutan di mana *search*-nya melalui *node*-*node* yang ada di pohon (*tree*).

Notasi yang dipakai oleh Algoritma A* adalah sebagai berikut:

$$f(n) = g(n) + h(n)$$

$f(n)$ = biaya estimasi terendah

$g(n)$ = biaya dari *node* awal ke *node* n

$h(n)$ = perkiraan biaya dari *node* n ke *node* akhir.

Dalam penerapannya, Algoritma A* memiliki beberapa terminologi dasar diantaranya *starting point*, simpul (*nodes*), A, *open list*, *closed list*, harga (*cost*), halangan (*unwalkable*).

1. *Starting point* adalah sebuah terminologi untuk posisi awal sebuah benda.
2. A adalah simpul yang sedang dijalankan dalam algoritma pencarian jalan terpendek.
3. Simpul adalah petak-petak kecil sebagai representasi dari area pathfinding. Bentuknya dapat berupa persegi, lingkaran, maupun segitiga.
4. *Open list* adalah tempat menyimpan data simpul yang mungkin diakses dari *starting point* maupun simpul yang sedang dijalankan.
5. *Closed list* adalah tempat menyimpan data simpul sebelum A yang juga merupakan bagian dari jalur terpendek yang telah berhasil didapatkan.
6. Harga adalah nilai yang diperoleh dari penjumlahan, jumlah nilai tiap simpul dalam jalur terpendek dari *starting point* ke A,

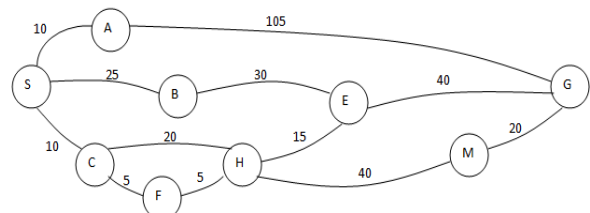
dan jumlah nilai perkiraan dari sebuah simpul ke simpul tujuan.

7. Simpul tujuan yaitu simpul yang dituju.
8. Halangan adalah sebuah atribut yang menyatakan bahwa sebuah simpul tidak dapat dilalui oleh A.

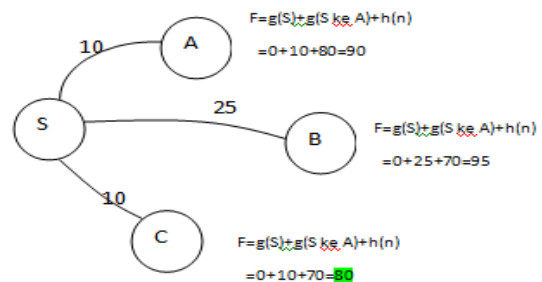
Menurut [12] Algoritma A* (A star) dikenal sebagai salah satu algoritma yang paling sering digunakan untuk pencarian jalur (*path finding*) dan penerusan grafis (*graph traversal*), yaitu proses *plotting* jalur yang paling efisien antar titik, yang disebut dengan *nodes*.

Menurut [13] Algoritma A star menggunakan dua senarai yaitu *OPEN* dan *CLOSED*. *OPEN* adalah senarai (*list*) yang digunakan untuk menyimpan simpul-simpul yang pernah dibangkitkan dan nilai heuristiknya telah dihitung tetapi belum terpilih sebagai simpul terbaik (*best node*) dengan kata lain, *OPEN* berisi simpul-simpul masih memiliki peluang untuk terpilih sebagai simpul terbaik, sedangkan *CLOSED* adalah senarai untuk menyimpan simpul-simpul yang sudah pernah dibangkitkan dan sudah pernah terpilih sebagai simpul terbaik (peluang untuk terpilih sudah tertutup).

2.2.2 Simulasi Algoritma A* (A-Star)



n	S	A	B	C	E	F	G	H	M
$h(n)$	80	80	70	70	75	78	0	70	70

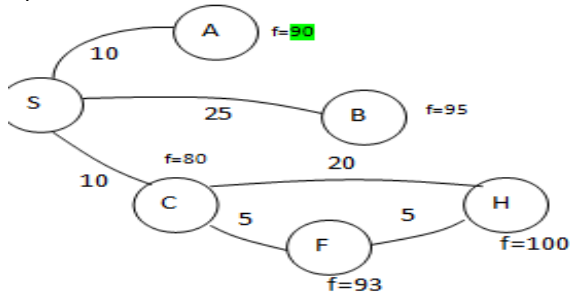


Gambar 2.1 Langkah Pertama

Langkah Pertama, karena di *OPEN* hanya terdapat 1 simpul (yaitu S), maka S Terpilih sebagai *BestNode*. *Best Node* Selanjutnya adalah $f(C)=80$

Closed : S

Open : A, B, C

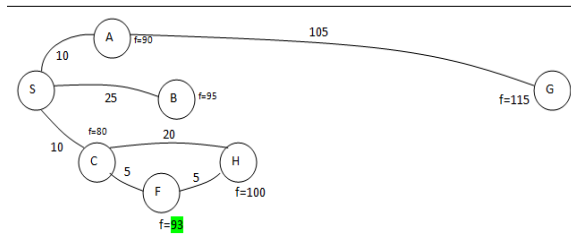


Gambar 2.2 Langkah Kedua

Langkah Kedua, C dengan biaya terkecil (yaitu 80) terpilih sebagai *BestNode* dan dipindahkan ke *CLOSED*, semua suksesor C dibuka yaitu F & H dan dimasukkan ke *OPEN*. *BestNode* Selanjutnya adalah $f(A)=90$

Closed : S, C,

Open : A, B, F, H

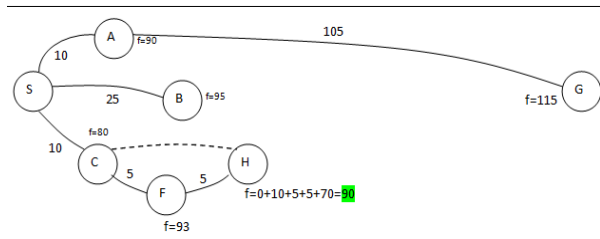


Gambar 2.3 Langkah Ketiga

Langkah Ketiga, A dengan biaya terkecil (yaitu 90) sebagai *Best Node* dan dipindahkan ke *CLOSED*, suksesor A dibuka yaitu G dimasukkan ke *OPEN*. $f(G)=115$. *Best Node* Selanjutnya adalah $f(F)=93$

Closed : S, C,,A

OPEN: B, F, H, G

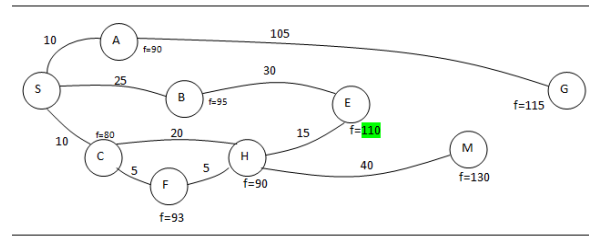


Gambar 2.4 Langkah Keempat

Langkah Keempat, F dengan biaya terkecil (yaitu 93) sebagai *BestNode* dan dipindahkan ke *CLOSED*, suksesor F dibuka yaitu H, karena H sudah *OPEN* maka dicek, Ternyata biaya dari C ke H melalui F (yaitu $5 + 5=10$) lebih kecil dari C ke H (yaitu 20). Oleh Karena itu parent dari H yang sebelumnya C menjadi F. *BestNode* selanjutnya $f(H)=90$

Closed : S, C, A, F

Open : B, G, H

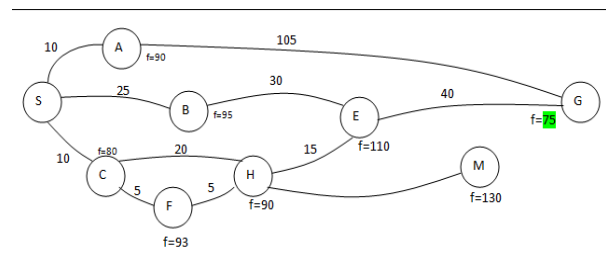


Gambar 2.5 Langkah Kelima

Langkah Kelima, B dengan biaya terkecil yaitu (95) sebagai *BestNode* dan dipindahkan ke *CLOSED*, suksesor B yaitu E, karena sudah *OPEN*, maka dicek. Jarak dari S ke E melalui B masih lebih besar yaitu ($25+30+75=130$). Maka tidak ada perubahan *parent* dari E.

Closed : S, C, F, A, H, B

Open: E, M,G



Gambar 2.6 Langkah Keenam

Langkah Keenam, E dengan dengan biaya terkecil (yaitu 110) sebagai *BestNode* dan dipindahkan ke *CLOSED*,suksesor E dibuka yaitu G, karena G sudah *OPEN* maka harus di cek, apakah *parent* dari G perlu diganti. Ternyata biaya S ke G melalui E lebih kecil (yaitu $0+10+5+5+15+40=75$) sedangkan dari S ke E melalui A (yaitu 115) jadi *parent* G dirubah ke E.

Closed : S, C, F, A, H, B, E

Open : G, M

Selanjutnya, G dengan biaya terkecil (yaitu 75) terpilih sebagai *BestNode*. Karena *BestNode* sama dengan *GOAL*, berarti Solusi sudah ditemukan. Rute dan total biaya bisa ditelusuri balik dari G ke S karena setiap simpul hanya memiliki satu parent dan setiap simpul memiliki informasi biaya sebenarnya (g). Penelusuran balik menghasilkan rute S-C-F-H-E-G dengan jarak 75 KM.

2.2.3 Notasi Algoritma A* (A-Star)

Pseudo Code A*

```
function A* (masalah) returns solusi
  OPEN <- S
  CLOSED <- array kosong
  loop sampai ditemukan atau sampai tidak ada
    if OPEN = kosong then
      Gagal
    else
      BestNode simpul yang ada di OPEN
      dengan f minimal
      Pindahkan simpul terbaiktersebut dari
      OPEN ke CLOSED
      if BestNode goal then
        Sukses
      else
        Bangkitkan semua suksesor BestNode
        tapi jangan buat pointer
        Untuk setiap suksesor kerjakan:
        Hitung g(suksesor) g(BestNode) +
        actual cost(dari BestNode ke suksesor)
        {Periksa suksesor}
        if suksesor ada di OPEN then (sudah
        pernah dibangkitkan tapi belum diproses)
          OLD simpul di OPEN yang sama
          dengan suksesor tersebut
          Tambahkan OLD sebagai suksesor
          BestNode
          Buat pointer dari OLD ke BestNode
          Bandingkan nilai g(OLD) dengan
          g(suksesor)
          if g(OLD) lebih baik then
            Ubah parent OLD ke
            BestNode
            Ubah nilai g dan f yang ada
            pada OLD
          end
        else
          if suksesor ada di CLOSED then
            (sudah pernah dibangkitkan dan sudah
```

```
diproses)
          OLD simpul di CLOSED yang
          sama dengan suksesor
          tersebut
          Tambahkan OLD sebagai
          suksesor BestNode
          Bandingkan nilai g(OLD)
          dengan g(suksesor)
          if g(OLD) lebih baik then
            Ubah parent OLD ke
            BestNode
            Ubah nilai g dan f yang
            ada pada OLD
            Propagasi untuk semua
            suksesor OLD dengan
            penelusuran DFS dengan
            aturan:
            loop sampai simpul sukseso
            tidak ada di OPEN atau simp
            tidak punya suksesor
            if suksesor ada di OPEN
              then
                Propagasi
              diteruskan
            else
              if nilai g via
              suksesor lebih baik then
                Propagasi
              diteruskan
            else
              Propagasi
            dihentikan
          end
        end
      end
    else {suksesor tidak ada di OPEN maupun
    CLOSED}
      Masukkan suksesor ke OPEN
      Tambahkan suksesor tersebut sebagai
      suksesornya BestNode
      Hitung f: g(suksesor) + h(suksesor)
    end
  end
end
end
```

2.3.4 Kompleksitas Waktu Algoritma A* (A-Star)

Kompleksitas waktu dari algoritma A tergantung dari heuristicnya. Dalam kasus terburuk *worst case*, jumlah simpul yang diekspansi bias eskponensial dalam solusinya jalan tependek. Akan tetapi, kompleksitasnya bisa berupa polinomial bila fungsi heuristic h bertemu kondisi berikut:

| $h_x - h$

$x | = O(\log x)$

Dimana h adalah *heuristik* optimal, atau *cost* pasti untuk menuju tujuan dari x . Dengan kata lain, kesalahan error dari h tidak boleh tumbuh lebih cepat dari algoritma "*perfect heuristic*" h yang mengembalikan jarak sebenarnya dari x menuju tujuan

3. KESIMPULAN

Berdasarkan hasil simulasi algoritma A* pada penelitian ini dapat disimpulkan hal-hal berikut :

1. Penentuan rute terbaik dapat dilakukan dengan Algoritma A* sehingga dapat diterapkan dengan baik.
2. Simulasi ini dapat menentukan rute (jalur) terbaik dari titik awal (*start*) menuju titik akhir (*finish*) dengan hambatan-hambatan yang diberikan disetiap rute. Dari hasil pengujian, rute yang ditemukan merupakan rute yang terbaik dengan nilai $f(n)$ terkecil dibandingkan dengan rute-rute (jalur-jalur) lainnya.
3. Untuk pengembangan lebih lanjut disarankan menggunakan algoritma lain selain algoritma A* untuk menentukan jalur (rute) yang terbaik. Dan juga dapat membandingkan algoritma lain tersebut apakah lebih baik dalam penentuan jalur tercepat.

REFERENSI

- [1] Andi, Rumani dkk. 2017. Perbandingan Kompleksitas Algoritma A-Star, Floyd-Warshall, Viterbi Pada SDN (Software Defined Networking). Universitas Telkom Bandung.
- [2] Deibi T Salaki. 2011. Penentuan Lintasan Terpendek dari FMIPA ke Rektorat dan Fakultas Lain di UNSRAT Manado Menggunakan Algoritma Dijkstra. Universitas Sam Ratulangi. Manado.
- [3] Ilham Saifudin, Dafik. 2015. Bilangan Khromatik Pewarnaan Sisi pada Graf Khusus dan Operasinya. Universitas Jember. Jember
- [4] Reza Wafdan, Mahyus Ihsan, Rahma Zuhra. 2014. Connectivity Algorithm in Simple

Graphs. Universitas Syiah Kuala Darussalam. Banda Aceh

- [5] AgustianAji P, SholehHadi Pramono, M. Aziz Muslim. 2015. Optimasi Jalur Tercepat dengan Menggunakan Modifikasi Algoritma Bellman Ford (Studi Kasus Lintasan antar Kecamatan Kota Malang). Malang.
- [6] Moh. Bandrigo Talai dkk. 2015 Aplikasi Pencarian Jalur Terpendek Pada Rumah Sakit Umum BAHTERAMAS Menggunakan Algoritma A* (A-Star). Kendari.
- [7] Muhammad Hisyam Fadhlurrahman dkk. 2014. Implementasi dan Analisis Penggunaan Algoritma A-Star Dengan Prioritas Pada Pemilihan Rute Lintas Kendaraan Roda Dua. Universitas Telkom. Bandung.
- [8] Diana Okta Pugas dkk. 2011. Pencarian Rute Terpendek Menggunakan Algoritma Dijkstra dan Astar (A*) pada SIG Berbasis Web untuk Pemetaan Pariwisata Kota Sawahlunto. Universitas Diponegoro. Semarang.
- [9] Kikik Chomsan Maaruf. 2016. Kecerdasan Buatan Menggunakan Algoritma A Star (A*) Dalam Permainan Ular Tangga (Snake 3D). STMIK AMIKOM Yogyakarta.
- [10] Kadek Ayu Yanti Pawitri, Joko Purwadi, M.Kom. 2007. Implementasi Algoritma Physical-A Star (PHA*) Untuk Menemukan Lintasan Terpendek. Universitas Kristen Duta Wacana Yogyakarta.
- [11] Zakaria., 2006, Teknologi Informasi dan Komunikasi, Arya Duta, Jakarta
- [12] Yenie Syukriyah, Falahah, Hermi Solihin, 2016 Penerapan Algoritma A* (STAR) Untuk Mencari Rute Tercepat Dengan Hambatan. Teknik Informatika Universitas Widyatama. Bandung.
- [13] Pasnur. 2016 Implementasi Algoritma WELCH-POWELL Dalam Pembuatan Jadwal Ujian Akhir Semester. Jurusan Sistem Informasi, STMIK AKBA. Makassar

- [14] Ardiansyah, Fery Sofian Efendi, Syaifullah, Mateus Pinto, Pujianto. Implementasi Algoritma GREEDY Untuk Melakukan *GRAPH COLORING*: Studi Kasus Peta Provinsi Jawa Timur.
- [15] Rosita Ayu Nugraheni, Mulyono, Rohmad. 2015. Penerapan Algoritma A* Dalam Penyelesaian Rute Terpendek Pendistribusian Barang. Universitas Negeri Semarang