ISSN 0853-3792 | e-ISSN 2549-8266

Jurnal Sains Indonesia
Volume 42(2): 44-51, 2018

Simulasi Lintasan Terpendek pada Graf Komplit Menggunakan *Ant Colony Optimization Algorithm*

Ami Riana dan Hermawan Syahputra
Jurusan Matematika, FMIPA, Universitas Negeri Medan, Medan, Indonesia, 20221
e-mail: amirianariana@gmail.com

Abstract *This study aims to build a shortest path simulation program on the complete graph K_{20}. The data of distance in this simulation was determined randomly with the provision of a value of 0–100. The simulation conducted was an algorithm calculation used parameter values with different initial pheromone conditions. The parameters in the Ant Colony Optimization Algorithm are set with alpha = 1, beta = 2, the initial pheromone condition = 0,0001 for the first simulation; alpha = 1, beta = 2, initial pheromone = 1 for the second simulation; and alpha value = 1, beta = 5, initial pheromone condition = 0,00000001 for the third simulation. The simulation results showed that if the value of the initial pheromone condition used gets greater, the value of the temporary output gets greater. Even though the initial pheromone condition was different, the shortest path obtained with distance data used in this study is the same, namely 15–14–1–3–20–12–16–6–18–10–9–13–11–7–8–4–2–17–5–19 with length 613 (in kilometers).* [THE SHORTEST PATH SIMULATION IN COMPLETE GRAPHS USING THE ANT COLONY OPTIMIZATION ALGORITHM]***(J. Sains Indon., 42(2): 44-51, 2018)***

Kata kunci: *Ant Colony Optimization Algorithm, Complete Graph, Shortest Path*

Pendahuluan

Traveling Salesman Problem (TSP) merupakan salah satu permasalahan optimasi yang muncul seiring dengan pemasaran produk yang semakin kompleks dalam penentuan jalur terpendek yang harus dilalui para *salesman*. Permasalahan ini biasanya membahas mengenai kota awal dan sejumlah kota untuk dikunjungi. Seorang salesman harus memulai perjalanan dari kota awal ke seluruh kota lain yang dikunjungi tepat satu kali. Ciri dari permasalahan TSP antara lain adalah: perjalanan berawal dan berakhir di kota awal, ada sejumlah kota yang semuanya harus dikunjungi tepat satu kali, perjalanan tidak boleh kembali ke kota awal sebelum semua kota tujuan dikunjungi. Tujuan dari permasalahan TSP adalah meminimumkan total jarak tempuh salesman dengan mengatur urutan-urutan kota yang harus dikunjungi.

Seiring berjalannya waktu, TSP menjadi persoalan yang banyak diaplikasikan pada berbagai persoalan dunia nyata, misalnya: efisiensi pengiriman surat dan barang, perencanaan pemasangan saluran pipa, masalah transportasi, persoalan *delivery order* (jasa pengantar makanan misalnya), dan lain sebagainya.

Terdapat beberapa penelitian terdahulu terkait rute terpendek. Dalam penelitian Triansyah (2013) digunakan algoritma Dijkstra untuk menghitung jarak terdekat dari suatu kota ke kota lainnya pada Sumatera bagian selatan. Hasil penelitian diimplementasikan ke dalam bentuk perangkat lunak, dan metode yang digunakan untuk mengembangkan perangkat lunak adalah metode *Waterfall*. Benyamin, dkk (2012), dalam penelitiannya tentang penentuan jalur wisata terpendek menggunakan metode *Forward Chaining*. Pada penelitian ini dibangun sebuah sistem pakar untuk menentukan jalur terpendek objek wisata Kota Kupang dengan menggunakan metode *Forward Chaining*. Metode Forward Chaining adalah metode yang menggunakan teknik pencarian Algoritma *Depth First Search* (DFS). Hasil akhir dari penelitian ini disajikan dalam sebuah sistem berbasis *web*.

Selain agoritma-algoritma yang digunakan pada penelitian terdahulu, ada algoritma lain yang dapat juga digunakan untuk menemukan jarak terpendek pada sebuah graf. Beberapa di antaranya Algoritma Ford dan Algoritma Floyd seperti yang dinyatakan oleh Ardiani (2011). Namun terdapat juga algoritma lainnya yang dapat digunakan untuk menemukan jarak terpendek dalam melakukan perjalanan yaitu *Algoritma Ant Colony Optimization* (ACO).

ACO atau disebut juga dengan Algoritma Semut merupakan suatu metode yang digunakan untuk menentukan jalur terpendek. Kristiawan (2015), menggunakan ACO untuk mengatasi permasalahan dalam optimasi jarak/rute. ACO juga dapat digunakan untuk mencari solusi TSP terutama dalam mencari jalur terpendek. Pada dasarnya ACO mengadaptasi cara kerja koloni semut untuk menemukan jarak terpendek dalam waktu yang singkat dari sumber makanan ke sarang ataupun sebaliknya. Semut dapat menemukan jarak terpendek dengan memanfaatkan jejak pheromone yang digunakan sebagai komunikasi tidak langsung antar semut.Jejak pheromone dapat digunakan oleh semut untuk menemukan jalan ke sumber makanan atau ke sarang. Selain itu, jejak pheromone dapat juga digunakan oleh semut lainnya untuk menemukan lokasi sumber makanan yang sebelumnya telah ditemukan semut lain. Algoritma semut bekerja dengan umpan balik yang positif dalam penemuan dan pencapaian solusi yang baik.Algoritma semut juga mempunyai sifat sinergi yang tinggi. Keefektifan pencarian ditunjukkan dengan memberikan sejumlah semut yang saling bekerja sama dan setiap kerja sama akan saling independen.

Berdasarkan latar belakang masalah di atas, rumusan masalah penelitian ini adalah; (1) Bagaimana penerapan Ant Colony Optamizaton (ACO) untuk pencarian lintasan terpendek pada Complete Graph, dan (2) Bagaimana membangun suatu program simulasi pencarian lintasan terpendek dengan menggunakan bahasa pemograman. Adapun batasan pada penelitian ini adalah graf yang digunakan adalah Complete Graph dan bahasa pemograman yang digunakan adalah Visual Basic (VB), SQL Server sebagai databasenya.

Metode

Penelitian ini dilakukan dengan prosedur sebagai berikut:

1. Pengolahan Data.
   Data yang berupa graf komplit K_{20} serta bobotnya akan diolah dengan menggunakan metode analisis hasil penyelesaian masalah penentuan lintasan terpendek menggunakan ACO. Selanjutnya akan dilakukan perancangan sistem dengan menuliskan diagram alir (*flowchart*) dalam perancangan aplikasi algoritma semut.
2. Implementasi ACO untuk pencarian rute terpendek.
   Pada tahap ini dilakukan pengimplementasian hasil pengolahan data yang telah dilakukan menggunakan algoritma semut ke dalam bahasa pemrograman Visual Basic (VB.net) dan SQL Server.

Desain bentuk tampilan program ini ditampilkan pada Gambar 1.

***Gambar 1**. Desain tampilan program*

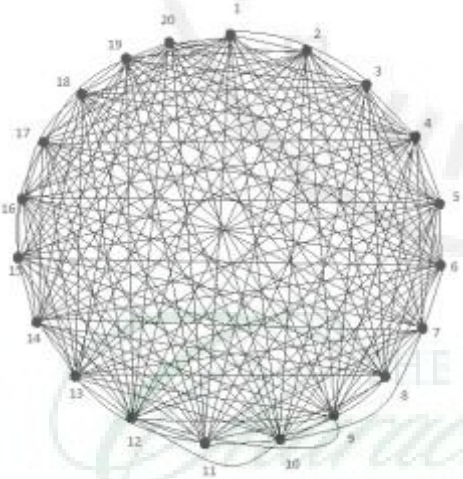
Pelaksanaan sistem dalam program ini adalah sebagai berikut:

1. Ketika program dimulai, pengguna harus menginput data graf berupa ukuran masing-masing node ke dalam tabel matriks.

2. Setelah itu, pengguna harus menginput nilai parameter algoritma, kemudian memilih salah satu node yang dijadikan node awal.
3. Setelah itu sistem akan melakukan pencarian rute terpendek.
4. Setelah selesai melakukan pencarian, program akan menampilkan hasil pencarian.

Hasil dan Pembahasan

Simulasi yang dilakukan dengan meng-*input* jumlah *node*, jarak antar *node* dan juga meng-input parameter algoritma. Data parameter yang diinput pengguna berupa nilai *alpha*, *beta*, nilai *pheromone* awal, serta banyaknya semut(k). Kemudian dicari nilai visibilitas antar *node*, mencari nilai probabilitas setiap *node* untuk dikunjungi. Setelah semut mengunjungi semua node maka didapatkan rute minimum yang ditempuh. Setelah informasi rute minimum diperoleh, kemudian mencari pembaharuan nilai *pheromone* yang baru dengan merubah nilai *pheromone* awal. graf komplit K_{20} ditunjukkan pada Gambar 2.

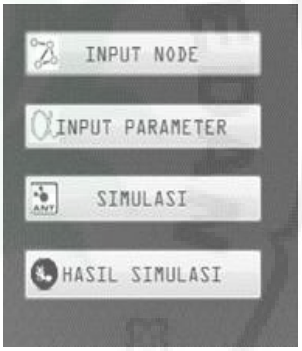


Gambar 2. *Graf komplit* K_{20}

Aplikasi dari algoritma semut ini dibatasi hanya pada pencarian jalur terpendek dari data yang diinput. Tampilannya terdiri dari beberapa *form* yang memiliki fungsi masing-masing yang tampil sesuai dengan urutan yang telah diprogram. Form utama aplikasi program ditunjukkan pada Gambar 3.

Gambar 3. *Tampilan utama simulasi ACO*

Pada *form* ACO *Simulation* ini terdapat beberapa menu antara lain Input *Node*, Input Parameter, Simulasi, dan Hasil Simulasi. Tampilannya ditunjukkan pada Gambar 4.



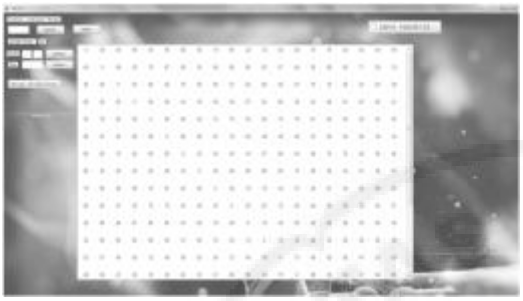
Gambar 4. *Menu-menu yang terdapat pada tampilan utama algorithm simulation*

Implementasi Gambar 3 menggunakan *Visual Basic* (VB.net) disajikan pada Gambar 5.

Gambar 5. *Source code untuk simulasi ACO*

Tampilan menu input *node* merupakan menu tampilan pada *form* kedua dalam sistem

yang memuat input jumlah *node*, tabel *node*, dan nilai visibilitas. Tampilan menu *node* ditunjukkan pada Gambar 6.

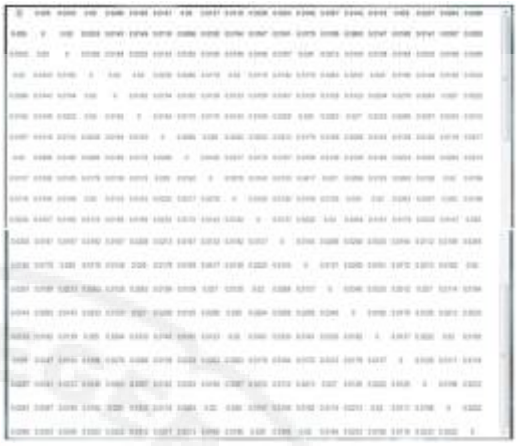


Gambar 6. *Tampilan menu Node*

Source code Gambar 6 menggunakan *Visual Basic* (VB.net) sebagaimana diperlihatkan dalam Gambar 7.

Gambar 7. *Source Code tampilan menu*

Sistem akan menampilkan nilai visibilitas seperti ditunjukkan pada Gambar 8. Menu tampilan input parameter merupakan menu tampilan pada *form* ketiga dalam sistem yang memuat input nilai parameter seperti nilai *alpha*, *beta*, banyak semut (*k*), dan nilai *pheromone* awal. Pada bagian ini, dilakukan perhitungan dengan nilai parameter yang berbeda.Ini bertujuan untuk melihat perbedaan hasil perhitungan dan melihat pengaruh nilai parameter terhadap perhitungan algoritma.



Gambar 8*. Nilai visibilitas*

Dengan mengunakan banyak semut 20, dilakukan tiga kali perhitungan. Perhitungan algoritma yang pertama menggunakan nilai alpha = 1, nilai beta = 2, dan nilai *pheromone* awal = 0,0001.

Setelah nilai parameter selesai diinput maka sistem akan menampilkan nilai parameter, nilai *temporary* dan nilai probabilitas. Nilai temporary ditunjukkan pada Gambar 9.

Gambar 9. *Nilai temporary parameter pertama*

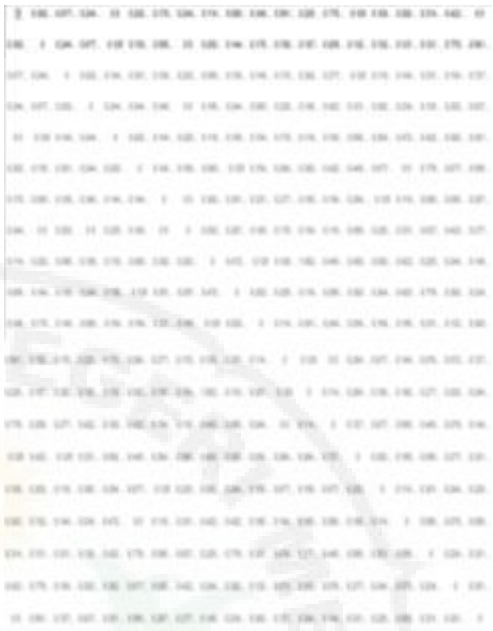
Nilai probabilitas perhitungan pertama ditunjukkan pada Gambar 10.

Gambar 10. *Nilai probabilitas paramater pertama*

Pada perhitungan algoritma yang kedua nilai parameter yang digunakan adalah nilai alpha = 1, beta = 2, pheromone awal = 1, dan banyak semut = 20.

Nilai temporary yang diperoleh ditampilkan pada Gambar 11 dan nilai probabilitasnya disajikan pada Gambar 12.

Gambar 11. *Nilai temporary parameter kedua*



Gambar 12. *Nilai probabilitas parameter kedua*

Pada perhitungan algoritma yang kedua ini memiliki hasil yang berbeda dengan perhitungan algoritma yang pertama untuk nilai temporarynya.Ini dipengaruhi oleh nilai pheromone awal yaitu 1 dan 0,0001.

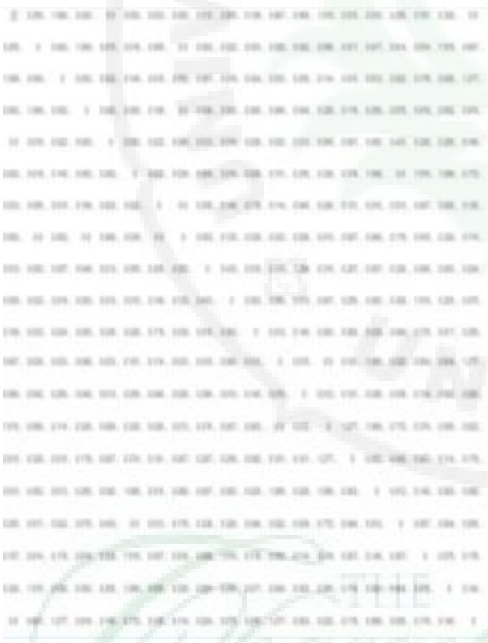
Gambar 13. *Nilai temporary parameter ketiga*

Selanjutnya dilakukan simulasi pada perhitungan algoritma yang ketiga dengan nilai parameter yang digunakan adalah nilai alpha = 1,

beta = 5, pheromone awal = ,.00000001, dan banyak semut = 20. Nilai temporary yang diperoleh ditampilkan pada Gambar 13 dan nilai probabilitasnya disajikan pada Gambar 14.

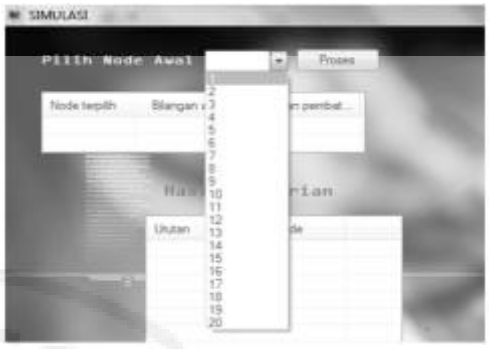
Pada perhitungan algoritma yang ketiga ini memiliki hasil yang berbeda dengan perhitungan algoritma yang pertama dan kedua untuk nilai temporarynya dan probabilistiknya. Hal ini dipengaruhi oleh nilai pheromone awal yaitu 0,00000001.

Menu tampilan simulasi merupakan menu tampilan pada form keempat yang merupakan tampilan inti dalam sistem karena digunakan untuk memuat simulasi pencarian lintasan terpendek. Pada form ini terdapat pemilihan node yang menjadi node awal perjalanan. Setelah selesai pemilihan node awal, maka program akan menampilkan hasil pencarian lintasan.



Gambar 14. *Nilai probabilitas parameter ketiga*

Gambar 15 menunjukkan bagian pemilihan *node* awal perjalanan. Pada bagian ini pengguna memilih node yang akan menjadi *node* awal. Misalkan pencarian lintasan dimulai dengan *node* awal 1. Setelah *node* awal terpilih maka program akan menampilkan nilai bilangan acak dan bilangan pembatasnya.

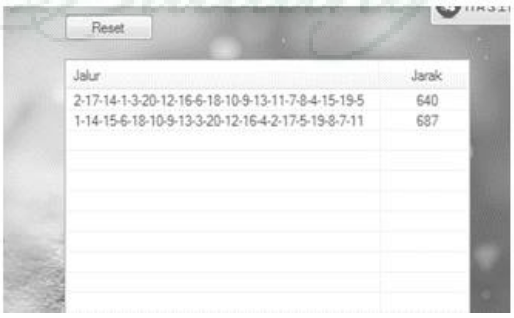


Gambar 15. *Pemilihan node awal*

Sistem akan menampilkan hasil pencarian lintasan sesuai dengan *node* awal yang terpilih. Lintasan terpendek beserta jarak dimulai *node* awal 1 sampai *node* 20 diperoleh beserta jarak tempuh. Kesimpulannya ditampilkan pada Gambar 16.

Gambar 16. *Hasil pencarian lintasan dengan node awal 1*

Berikutnya dilakukan simulasi dengan *node* awal 2. Hasil simulasi ditunjukkan dalam Gambar 17. Jarak lintasan yang diperoleh dengan *node* awal 2 adalah 640, dan ini merupakan hasil yang lebih pendek dibanding dengan lintasan dengan *node* awal 1 yaitu 687.



Gambar 17. *Lintasan yang diperoleh dengan node awal 2*

Seterusnya dilakukan simulasi hingga *node* awal 20. Hasil akhir pencarian lintasan terpendek yang diperoleh terlihat seperti pada Gambar 18.

Jalur	Jarak
15-14-1-3-20-12-16-6-18-10-9-13-11-7-8-4-2-17-5-19	613
14-1-3-20-12-16-6-18-10-9-13-11-7-8-4-2-17-5-19-15	631
11-20-12-16-14-1-3-13-9-10-18-6-19-2-17-5-4-8-7-15	633
20-12-16-14-1-3-13-9-10-18-6-19-2-17-5-4-8-7-11-15	637
2-17-14-1-3-20-12-16-6-18-10-9-13-11-7-8-4-15-19-5	640
16-14-1-3-20-12-7-8-18-10-9-13-6-19-2-17-5-4-15-11	651
7-8-18-10-9-13-6-19-2-17-14-1-3-20-12-16-4-15-11-5	653
9-13-6-18-10-17-14-1-3-20-12-16-4-2-8-7-11-15-19-5	655
19-2-17-14-1-3-20-12-16-6-18-10-9-13-11-7-8-4-15-5	662
13-9-10-18-6-19-2-17-14-1-3-20-12-16-4-8-7-11-15-5	663
4-2-17-14-1-3-20-12-16-6-18-10-9-13-11-7-8-19-5-15	664
17-14-1-3-20-12-16-6-18-10-9-13-11-7-8-4-2-15-19-5	667
10-18-6-19-2-17-14-1-3-20-12-16-4-8-7-9-13-11-15-5	669
3-20-12-16-14-1-5-17-6-18-10-9-13-11-7-8-4-2-15-19	671
18-10-9-13-6-19-2-17-14-1-3-20-12-16-4-8-7-11-15-5	675
8-18-10-9-13-6-19-2-17-14-1-3-20-12-16-4-7-11-15-5	678
6-18-10-9-13-3-20-12-16-14-1-5-17-8-7-4-2-11-15-19	679
1-14-15-6-18-10-9-13-3-20-12-16-4-2-17-5-19-8-7-11	687
12-20-3-1-14-15-6-18-10-9-13-11-7-8-4-2-17-5-19-16	699
5-1-14-15-6-18-10-9-13-3-20-12-16-4-2-17-8-7-11-19	719

Gambar 18. *Hasil simulasi lintasan terpendek*

Dari hasil yang diperoleh, terlihat bahwa *node* yang telah dicari dengan *node* awal sebelumnya diurutkan berdasarkan jarak minimal dan lintasan dengan jarak yang paling pendek adalah lintasan dengan *node* awal 15 yaitu 613 (dalam km) dengan lintasan 15-14-1-3-20-12-16-6-18-10-9-13-11-7-8-4-2-17-5-19.

Menu tampilan hasil simulasi adalah menu tampilan pada *form* kelima yang merupakan tampilan inti kedua setelah menu simulasi karena pada menu tampilan ini menampilkan hasil pencarian lintasan terpendek. Pada *form* ini terdapat sebuah tabel yang menampilkan jalur dan jaraknya. Pengurutan jalur dalam sistem penacarian lintasan ini berdasarkan jarak yang terpendek. Hasil pencarian lintasan terpendek yang diperoleh merupakan hasil yang terbaik.

Dari Gambar 19, dapat dilihat bahwa hasil simulasi yang diperoleh dari pencarian lintasan terpendek pada graf komplit K_{20} adalah lintasan dengan node awal 15 yaitu 613 (dalam km) dengan lintasan 15-14-1-3-20-12-16-6-18-10-9-13-11-7-8-4-2-17-5-19.

Jalur	Jarak
15-14-1-3-20-12-16-6-18-10-9-13-11-7-8-4-2-17-5-19	613
14-1-3-20-12-16-6-18-10-9-13-11-7-8-4-2-17-5-19-15	631
11-20-12-16-14-1-3-13-9-10-18-6-19-2-17-5-4-8-7-15	633
20-12-16-14-1-3-13-9-10-18-6-19-2-17-5-4-8-7-11-15	637
2-17-14-1-3-20-12-16-6-18-10-9-13-11-7-8-4-15-19-5	640

Gambar 19. *Hasil simulasi lintasan terpendek graf komplit* K_{20}

Hasil simulasi menampilkan lima lintasan dengan jarak terpendek:

1. Lintasan pertama dimulai dengan *node* awal 15 yaitu 15-14-1-3-20-12-16-6-18-10-9-13-11-7-8-4-2-17-5-19 dengan jarak 613 (dalam km).
2. Lintasan kedua dengan *node* awal 14 yaitu 14-1-3-20-12-16-6-18-10-9-13-11-7-8-4-2-17-5-19-15 dengan jarak 631 (dalam km).
3. Lintasan ketiga dengan *node* awal 11 yaitu 11-20-12-16-14-1-3-13-9-10-18-6-19-2-17-5-19-15 dengan jarak 633 (dalam km).
4. Lintasan keempat dengan *node* awal 20 yaitu 20-12-16-14-1-3-13-9-10-18-6-19-2-17-5-4-8-7-11-15 dengan jarak 637 (dalam km).
5. Lintasan yang kelima dengan *node* awal 2 yaitu 2-17-14-1-3-20-12-16-6-18-10-9-13-11-7-8-4-15-19-5 dengan jarak 640 (dalam km).

Penutup

Pada penelitian ini dilakukan simulasi lintasan terpendek pada graf komplit dengan *ant colony optimization algorithm* (ACO) atau yang biasa disebut dengan algoritma semut menggunakan bahasa pemograman *Visual Basic (VB.net) dengan database SQL Server*. Graf komplit yang digunakan dalam penelitian ini adalah graf K_{20}. Simulasi yang dilakukan pada penelitian ini menggunakan nilai parameter alpha dan beta yang berbeda-beda untuk melihat peran penting nilai parameter pada proses perhitungan algoritma. Simulasi yang dilakukan juga menggunakan nilai pheromone awal yang berbeda untuk melihat pengaruh nilai pheromone awal pada proses perhitungan algoritma. Penentuan *node* awal ditentukan secara acak dan sebanyak simpul yang ada pada graf.Setelah semua node ditentukan selanjutnya

sistem secara otomatis memproses dan menampilkan hasil pencarian.

Algoritma ACO dapat diterapkan pada penyelesaian *Traveling Salesman Problem* serta digunakan untuk menyelesaikan masalah pencarian lintasan terpendek untuk graf komplit K_{20}. Pencarian lintasan terpendek menggunakan data jarak, nilai alpha, beta, pheromone awal, dan juga banyaknya semut (k) yang digunakan yang sesuai dengan banyaknya simpul pada graf agar hasil simulasi yang diperoleh lebih maksimal.

Hasil simulasi yang diperoleh adalah jalur lintasan berdasarkan pilihan *node* dengan nilai *temporary* atau probabilitasnya, kemudian lintasan yang telah diperoleh sebelumnya diurutkan berdasarkan jarak paling minimum, dan setelah itu user dapat melihat jalur lintasan yang terpendek. Penggunaan bilangan *random* dan bilangan pembatas serta nilai *temporary* dan probabilitas juga mempengaruhi proses pemilihan *node* dan setiap jarak dari lintasan yang dicari diurutkan dari yang terendah sehingga memudahkan *user* mendapatkan lintasan yang terpendek.

Diharapkan penelitian ini dapat dikembangkan untuk proses pencarian jalur alternatif dan dengan menggunakan bahasa pemrograman lain agar penerapan algoritma *Ant Colony Optimization* dapat bekerja lebih baik lagi.

Daftar Pustaka

Belalawe, Benyamin J., M. d. A. F. S., (2012): Penentuan Jalur Wisata Terpendek Menggunakan Metode Forward Chaining (Studi Kasus Dinas Pariwisata Kota Kupang), *Prosiding Seminar Nasional Informatika,*.

Berlianty, I. d. M. A., (2010): *Teknik-teknik Optimasi Heuristik*, Graha Ilmu, Yogyakarta.

Dorigo, M., dan Stutzle, T., (2004): *Ant Colony Optimization*, Massachusetts Institute of Technology, London.

Intan, B., dan Arifin, M., (2010): *Teknik-teknik Optimasi Heuristik*, I, Graha Ilmu.

Kusumadewi, S. d. H. P., (2005): *Penyelesaian Masalah Optimasi dengan Teknikteknik Heuristik*, Graha Ilmu, Yogyakarta.

Mindaputra, E., (2009): *Penggunaan Algoritma Ant Colony System dalam Traveling Salesman Problem (TSP) pada PT.Eka Jaya Motor [Skripsi]*, Universitas Diponegoro, Semarang.

Mulia, D., (2011): *Aplikasi Algoritma Ant System (AS) dalam Kasus Travelling Salesman Poblem (TSP) [Skripsi]*, Universitas Islam Negeri Syarif Hidayatullah, Jakarta.

Munir, R., (2005): *Matematika Diskrit (Edisi Revisi Kelima)*, Penerbit Informatika, Bandung. Ostfeld, A., (2011): Ant Colony Optimization Methods and Applications, InTech, India.

Rosen, K. H., (2012): *Discrete Mathematics and Its Applications (Seventh Edition)*, McGraw-Hill, New York.

Siang, J. J., (2006): *Matematika Diskrit dan Aplikasinya pada Ilmu Komputer*, Penerbit Andi, Yogyakarta.

Triansyah, A. F., (2013): Implementasi Algoritma Djikstra dalam Aplikasi untuk Menentukan Lintasan Terpendek Jalan Darat Antar Kota di Sumatera Bagian Selatan, *Sistem Informasi*, 5(2), 611–621.