p-ISSN : 2443 – 0366
e-ISSN : 2528 – 0279

SIMULASI PENENTUAN LINTASAN TERPENDEK PADA *COMPLETE GRAPH* DENGAN MENGGUNAKAN *ANT COLONY OPTIMIZATION ALGORITHM*

Ami Riana[1], Hermawan Syahputra[2]
[1,2]Fakultas Matematika dan Ilmu Pengetahuan Alam, Universitas Negeri Medan
email: rianaami16@gmail.com

ABSTRAK

Ant Colony Optimization Algorithm merupakan suatu metodologi yang dihasilkan berdasarkan pengamatan terhadap perilaku semut. *Ant Colony Optimization Algorithm* berfungsi untuk menemukan solusi dalam permasalahan pencarian lintasan terpendek dengan berdasarkan nilai probabilistic dan dengan bantuan semut buatan yang terdapat pada algoritma ini. Penelitian ini bertujuan untuk membangun suatu program simulasi lintasan terpendek pada *complete graph* K_{20}. Data jarak pada simulasi ini ditentukan secara random dengan ketentuan nilai 0 – 100. Simulasi yang dilakukan adalah perhitungan algoritma menggunakan nilai parameter dengan kondisi *pheromone* awal yang berbeda-beda. Parameter pada *Ant Colony Optimization Algorithm* diatur dengan nilai alpha = 1, beta = 2, nilai kondisi pheromone awal = 0.0001 untuk simulasi pertama, nilai alpha = 1, beta = 2, kondisi pheromone awal = 1 untuk simulasi kedua, dan nilai alpha = 1, beta = 5, kondisi *pheromone* awal = 0.00000001 untuk simulasi ketiga. Hasil simulasi menunjukkan bahwa semakin besar nilai kondisi *pheromone* awal yang digunakan maka semakin besar pula nilai *temporary* yang dihasilkan. Meskipun dengan kondisi *pheromone* awal berbeda, namun lintasan terpendek yang didapatkan dengan data jarak yang digunakan pada penelitian ini adalah sama yaitu 15 – 14 – 1 – 3 – 20 – 12 – 16 – 6 – 18 – 10 – 9 – 13 – 11 – 7 – 8 – 4 – 2 – 17 – 5 – 19 dengan panjang 613 (dalam km).

Kata kunci: Ant Colony Optimization Algorithm, Complete Graph, Simulasi, Lintasan Terpendek.

ABSTRACT

Ant Colony Optimization Algorithm is a methodology generated based on observations of ant behavior. Ant Colony Optimization Algorithm serves to find solutions in the problem of finding the shortest path based on probabilistic value and with the help of artificial ants found in this algorithm. This study aims to build a shortest path simulation program on the complete graph K_{20}. *The distance data in this simulation is determined randomly with terms 0 - 100. The simulation performed is the algorithm calculation using parameter values with different initial pheromone conditions. The parameters in Ant Colony Optimization Algorithm are set with the value of alpha = 1, beta = 2, initial pheromone condition value = 0.0001 for the first simulation, alpha = 1, beta = 2, initial pheromone = 1 for second simulation, and alpha = 1 , beta = 5, initial pheromone conditions = 0.00000001 for the third simulation. Simulation results show that the greater the initial pheromone condition used, the greater the temporary value generated. Although the initial pheromone conditions were different, the shortest path obtained with the distance data used in this study was the same, that was 15 – 14 – 1 – 3 – 20 – 12 – 16 – 6 – 18 – 10 – 9 – 13 – 11 – 7 – 8 – 4 – 2 – 17 – 5 – 19 with length 613 (in km).*

p-ISSN : 2443 – 0366
e-ISSN : 2528 – 0279

Keywords: Ant Colony Optimization Algorithm, Complete Graph, Simulation, Shortest Path.

PENDAHULUAN

Traveling Salesman Problem (TSP) dikenal sebagai salah satu masalah yang banyak menarik perhatian para peneliti. *Traveling Salesman Problem* merupakan salah satu permasalahan optimisasi yang muncul seiring dengan pemasaran produk yang dilakukan oleh para *salesman*, dengan permasalahan yang semakin kompleks dalam penentuan jalur terpendek yang harus dilalui para *salesman*. Permasalahan ini biasanya membahas mengenai kota awal dan sejumlah *n* 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: perjalanan berawal dan berakhir di kota awal, ada sejumlah kota yang semuanya harus dikunjungi tepas satu kali, perjalanan tidak boleh kembali ke kota awal sebelum semua kota tujuan dikunjungi. Adapun tujuan dari permasalahan *Traveling Salesman Problem* ini adalah meminimumkan total jarak tempuh *salesman* dengan mengatur urutan-urutan kota yang harus dikunjungi.

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

Terdapat beberapa penelitian terdahulu terkait rute terpendek. Triansyah [1] menggunakan algoritma Djikstra untuk menghitung jarak terdekat dari suatu kota ke kota lainnya pada Sumatera bagian selatan. Hasil penelitian diimplementasi-kan ke dalam bentuk perangkat lunak, dan metode yang digunakan untuk mengembangkan perangkat lunak adalah metode *Waterfall*. Belalawe [2] dalam penelitiannya tentang penentuan jalur wisata terpendek menggunakan metode *Forward Chaning*. Pada penelitian ini dibangun sebuah system pakar untuk menentukan jalur terpendek objek wisata Kota Kupang dengan menggunakan metode *Forward Chaning*. Metode *Forward Chaning* adalah metode yang menggunakan teknik pencarian algoritma *depth first search* (DFS). Hasil akhir dari penelitian ini disajikan dalam sebuah system berbasis *web*.

Selain algoritma-algoritma yang digunakan pada penelitian terdahulu, ada algoritma lain yang dapat juga digunakan untuk menemukan jarak terpendek pada sebuah graf. Beberapa diantaranya Algoritma Ford dan Algoritma Floyd seperti yang dinyatakan oleh Ardiani [3]. Namun terdapat juga algoritma lainnya yang dapat digunakan untuk menemukan jarak terpendek dalam melakukan perjalanan yaitu Algoritma *Ant Colony Optimization* (ACO).

Algoritma *Ant Colony Optimization* (ACO) atau disebut juga dengan Algoritma Semut merupakan suatu metode yang digunakan untuk menentukan jalur terpendek. Kristiawan (2015), penggunaan metode *Ant Colony Optimization* (ACO) atau disebut juga dengan Algoritma Semut dapat dipergunakan untuk mengatasi permasalahan dalam optimasi jarak/rute. Penggunaan metode ACO juga dapat digunakan untuk mencari solusi Traveling Salesman Problem (TSP) terutama dalam menvari jalur terpendek.

Pada dasarnya algoritma semut ini mengadaptasi cara kerja koloni semut untuk menemukan jarak

p-ISSN : 2443 – 0366
e-ISSN : 2528 – 0279

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 seut 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. penggunaan mesin dapat dimaksimumkan lebih baik dari pada rencana perusahaan.

Berdasarkan latar belakang masalah tersebut, penulis tertarik untuk mencoba suatu simulasi untuk mencari lintasan terpendek pada *complete graph*.

TINJAUAN PUSTAKA

Graf

Secara kasar, graf adalah suatu diagram yang memuat informasi tertentu jika diinterpretasikan secara tepat. Dalam kehidupan sehari-hari, graf digunakan untuk menggambarkan berbagai macam struktur yang ada. Tujuannya adalah sebagai visualisasi objek-objek agar lebih mudah dimengerti.

Secara matematis, graf didefinisi-kan sebagai berikut : Graf G didefinisikan sebagai pasangan himpunan (V, E) ditulis dengan notasi $G = (V, E)$, yang dalam hal ini V adalah himpunan tidak kosong dari simpul-simpul *vertices* atau *node* dan E adalah himpunan sisi *edges* atau *arcs* yang menghubungkan sepasang simpul. Definisi ini menyatakan bahwa V tidak boleh kosong, sedangkan E boleh kosong. Jadi, sebuah graf dimungkinkan tidak mempunyai sisi satu buah pun, tetapi simpulnya harus ada, minimal satu. Graf yang hanya mempunyai satu buah simpul tanpa sebuah sisi dinamakan graf trivial [4].

Graf Tak Sederhana

Graf yang mengandung sisi ganda atau gelang dinamakan graf tak sederhana *unsimple graph*. Salah satu jenis graf tak sederhana, yaitu graf ganda *multi graph*. Graf ganda *multigraph* adalah graf yang mengandung sisi ganda. Sisi ganda yang menghubungkan sepasang simpul bisa lebih dari dua buah. Graf ganda $G = (V, E)$, terdiri dari himpunan tidak kosong simpul-simpul dan E adalah himpunan ganda *multiset* yang mengandung sisi ganda.

Graf Tak Berarah (*Undirected Graph*)

Berdasarkan jenis garis-garisnya, graf dibedakan dalam 2 (dua) kategori, yaitu graf tak berarah *undirected graph* dan graf berarah *directed graph* atau *digraph* [5]. Graf tak berarah adalah graf yang sisinya tidak memiliki orientasi arah. Pada graf tak berarah, urutan pasangan simpul yang dihubungkan oleh sisi tidak diperhatikan. Jadi, $(u, v) = (v, u)$ adalah sisi yang sama.

Graf Terhubung (*Connected Graph*)

Keterhubungan dua buah simpul adalah penting di dalam graf. Dua buah simpul u dan simpul v dikatakan ter-hubung jika terdapat lintasan dari u ke v. Jika dua buah simpul terhubung maka

p-ISSN : 2443 – 0366
e-ISSN : 2528 – 0279

pasti simpul yang pertama dapat dicapai dari simpul yang kedua. Jika setiap pasang simpul di dalam graf terhubung, maka graf tersebut dapat dikatakan graf terhubung.

Graf Lengkap

Graf lengkap adalah graf sederhana yang setiap verteksnya mempunyai *edge* ke semua *verteks* lainnya. Graf lengkap dengan n buah *verteks* dilambangkan dengan K_n. Jumlah *edge* pada graf lengkap yang terdiri dari n buah verteks adalah $n(n-1)/2$.

Graf Berbobot (*Weighted Graph*)

Graf berbobot adalah graf yang setiap sisinya diberi sebuah harga (bobot) [4]. Bobot pada tiap sisi dapat berbeda-beda bergantung pada masalah yang dimodelkan dengan graf. Bobot dapat menyatakan jarak antara dua buah kota, biaya perjalanan antara dua buah kota, dan lain sebagainya.

Lintasan dan Sirkuit Hamilton

Lintasan Hamilton adalah sebuah lintasan sederhana dalam graf G yang melalui tiap simpul di dalam graf G tepat satu kali [6]. Bila lintasan itu kembali lagi ke simpul awal dan membentuk lintasan tertutup, maka lintasan itu dinamakan sirkuit Hamilton [7].

Lintasan Terpendek (Shortest Path)

Lintasan terpendek adalah lintasan minimum yang diperlukan untuk mencapai suatu tempat dari tempat tertentu. Lintasan minimum yang dimaksud dapat dicari dengan menggunakan graf. Graf yang digunakan adalah graf yang berbobot, yaitu graf yang memiliki suatu nilai atau bobot.

Metode Pencarian Jalur Terpendek

Pada dasarnya permasalahan pencarian jalur terpendek antar kota merupakan pencarian jalur terpendek antar titik yang telah diketahui koordinatnya. Dengan mengetahui konsep pencarian jalur terpendek antar titik, untuk selanjutnya dapat diterapkan pada pencarian jalur terpendek pada berbagai kota yang ingin diketahui.

Metode Pencarian Heuristik

Heuristik adalah sebuah teknik yang mengembangkan efisiensi dalam proses pencarian. Untuk dapat menerapkan heuristik tersebut dengan baik dalam suatu domain tertentu, diperlukan suatu fungsi heuristik.

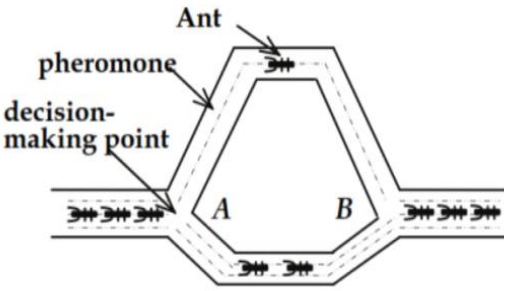
Sejarah Algoritma Semut

Algoritma semut pertama kali diperkenalkan oleh Moyson dan Manderick dan secara meluas dikembangkan oleh Marco Dorigo sebagai sebuah pendekatan awal terhadap berbagai masalah sulit, salah satunya seperti masalah *Travelling Salesman Problem*.

Cara Kerja Algoritma Semut

Prinsip dari algoritma semut adalah zat *pheromone* yang ditinggalkan pada lintasan yang dilalui, dimana zat *pheromone* ini akan membantu semut lainnya dalam menemukan tempat tujuan [8].

p-ISSN : 2443 – 0366
e-ISSN : 2528 – 0279



Gambar 1. Proses semut mencari lintasan terpendek

Tahapan Algoritmta Semut

1. **Inisialisasi**
   Tahap inisialisasi ini dibutuhkan nilai visibilitas atau *invers* jarak antar *node*. Nilai visibilitas dinyatakan dengan η_{ij}. Nilai visibilitas η_{ij} dihitung berdasarkan persamaan 1.1 berikut:

$$\eta_{ij} = \frac{1}{d_{ij}}$$

   keterangan:
   η_{ij} = invers jarak dari *node* i ke *node j*
   d_{ij} = jarak dari *node node* i ke *node j*

2. **Pembangunan solusi**
   Pada aturan ini, semut akan memilih *node* tujuan secara acak. Maka untuk menentukan node tujuan ini digunakan persamaan 2.1 probabilitas node untuk dikunjungi sebagai berikut:

$$p_{ij,k}(t) = \frac{[\tau_{ij}(t)]^{\alpha}.[\eta_{ij}]^{\beta}}{\sum[\tau_{ik}(t)]^{\alpha}.[\eta_{ik}]^{\beta}}$$

$$temporary\ (ij) = [\tau_{ik}(t)]^{\alpha}.[\eta_{ik}]^{\beta}$$

   keterangan:
   $p_{ij,k}(t)$ = peluang semut ke-k untuk mengunjungi *node j* dari *node i* pada iterasi ke-t
   $\tau_{ij}(t)$ = inisialisasi *pheromone* antara *node* i dan *node j* pada iterasi ke-t
   Jika $q \leq q_0$ dengan q adalah bilangan pecahan acak 0 – 1, dan q_0 adalah bilangan pembatas, maka pemilihan titik yang akan dituju menerapkan aturan yang ditunjuk-kan oleh persamaan 2.2 berikut:

$$v = max[\tau_{ik}(t)]^{\alpha}.[\eta_{ik}]^{\beta}$$

   v merupakan *node* yang akan dituju.

3. **Pembaharuan *pheromone* lokal**
   Dilakukan suatu perubahan tingkat *pheromone* pada *node* dengan persamaan 3.1 berikut:

$$\Delta\tau(ij,k) = \frac{1}{L_k \times C.}, jika\ (ij) \in T_k$$

   dimana:
   $\Delta\tau(ij,k)$ merupakan perubahan tingkat *pheromone*
   L_k merupakan panjang rute terpendek
   T_k merupakan rute
   C merupakan jumlah *node*

   Selanjutnya penguapan pheromone pada seluruh sisi dengan menggunakan persamaan 3.2 berikut:

$$\tau(ij)(baru) \leftarrow (1-\rho).\tau(ij) + \rho.\Delta\tau(ij,k)$$

   keterangan:
   $\tau(ij)$ adalah tingkat *pheromone*

p-ISSN : 2443 – 0366
e-ISSN : 2528 – 0279

ρ merupakan parameter laju penguapan *pheromone*

4. **Pembaharuan *Pheromone* global**
   Setelah semua semut melewati semua *node* dan mendapatkan rute terpendek, maka dilakukan suatu pembaharuan tingkat *pheromone* secara global. Perubahan ini dilakukan hanya pada rute terpendek. Perhitungan dapat dilakukan dengan persamaan 4.1 berikut:

$$\Delta\tau(ij,k) = \frac{1}{L_k}$$

   Selanjutnya, perubahan tingkat *pheromone* dengan persamaan 3.2.

METODE PENELITIAN

Penelitian ini dilakukan di Digital Library Universitas Negeri Medan yang berlokasi di Jl. Willem Iskandar Psr. V Medan. Penelitian ini dilaksanakan selama kurang lebih dua bulan. Jenis penelitian yang digunakan dalam penelitian ini adalah penelitian kepustakaan atau riset kepustakaan (*library research*). Penelitian kepustakaan yaitu melakukan penelusuran dengan penelaahan terhadap beberapa literature yang mempunyai relevansi dengan topik pembahasan. Informasi untuk penelitian ini dikumpulkan dari beberapa referensi, jurnal, maupun dokumen-dokumen lainnya yang berkaitan dengan topik pembahasan.

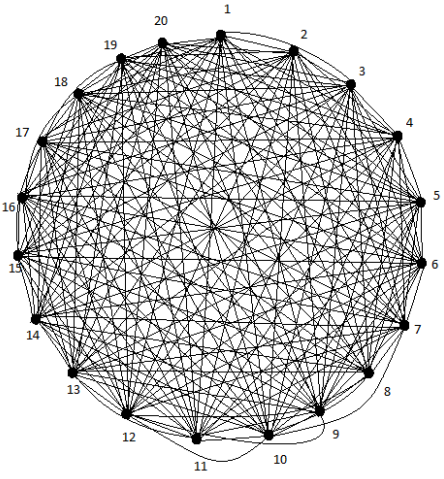
Agar penelitian dapat berjalan dengan baik, maka diperlukan beberapa prosedur penelitian sebagai berikut:

1. Studi kepustakaan.
2. Pengolahan data. Data yang digunakan merupakan *complete graph* K_{20} beserta bobotnya yang merupakan data simulasi jarak antara *node*. Kemudian data akan diolah dengan menggunakan metode analisis hasil menggunakan *Ant Colony Optimization* (ACO) *Algorithm.*
3. Pembuatan program simulasi penentuan lintasan terpendek dengan menggunakan bahasa pemrograman Visual Basic (VB.net).
4. Membuat kesimpulan pada program simulasi dengan menggunakan *Ant Colony Optimization* (ACO) *Algorithm*.

HASIL DAN PEMBAHASAN

1. Perancangan Simulasi

Simulasi yang dilakukan pada penelitian ini adalah dengan meng*input* parameter algoritma. Data parameter yang di-*input* berupa nilai alpha, beta, nilai kondisi *pheromone* awal, serta banyaknya semut (*k*).



Gambar 2. *Complete Graph* K_{20}

Data simulasi pada Tabel 1. berikut merupakan jarak antara *node* pada *complete graph* K_{20}.

p-ISSN : 2443 – 0366
e-ISSN : 2528 – 0279

Node	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	0	40	30	50	35	55	60	50	73	85	42	33	52	28	70	75	40	44	38	35
2	40	0	50	30	70	67	85	35	55	65	60	62	57	54	38	55	68	71	28	33
3	30	50	0	55	65	45	75	55	80	67	64	60	40	47	70	72	65	43	63	29
4	50	30	55	0	50	50	42	35	56	50	87	52	56	38	43	40	51	67	55	31
5	35	70	65	50	0	55	65	52	72	75	63	60	72	78	81	49	36	38	40	45
6	55	67	45	50	55	0	65	58	87	70	63	48	40	38	37	30	35	28	30	32
7	60	85	75	42	65	65	0	35	40	45	43	47	56	63	48	70	72	82	85	46
8	50	35	55	35	52	58	35	0	55	46	58	60	63	72	80	52	43	31	38	47
9	73	55	80	56	72	87	40	55	0	36	70	75	24	37	39	81	38	52	50	64
10	85	65	67	50	75	70	45	46	36	0	55	52	72	80	40	50	38	28	40	51
11	42	60	64	87	63	63	43	58	70	55	0	73	45	50	49	62	56	43	68	40
12	33	62	60	52	60	48	47	60	75	52	73	0	70	35	48	30	65	89	92	29
13	52	57	40	56	72	40	56	63	24	72	45	70	0	73	48	62	58	47	55	50
14	28	54	47	38	78	38	63	72	37	80	50	35	73	0	29	30	32	37	88	65
15	70	38	70	43	81	37	48	80	39	40	49	48	48	29	0	55	56	80	47	43
16	75	55	72	40	49	30	70	52	81	50	62	30	62	30	55	0	73	45	50	52
17	40	68	65	51	36	35	72	43	38	38	56	65	58	32	56	73	0	80	90	85
18	44	71	43	67	38	28	82	31	52	28	43	89	47	37	80	45	80	0	51	43
19	38	28	63	55	40	30	85	38	50	40	68	92	55	88	47	50	90	51	0	45
20	35	33	29	31	45	32	46	47	64	51	40	29	50	65	43	52	85	43	45	0

2. Implementasi Program

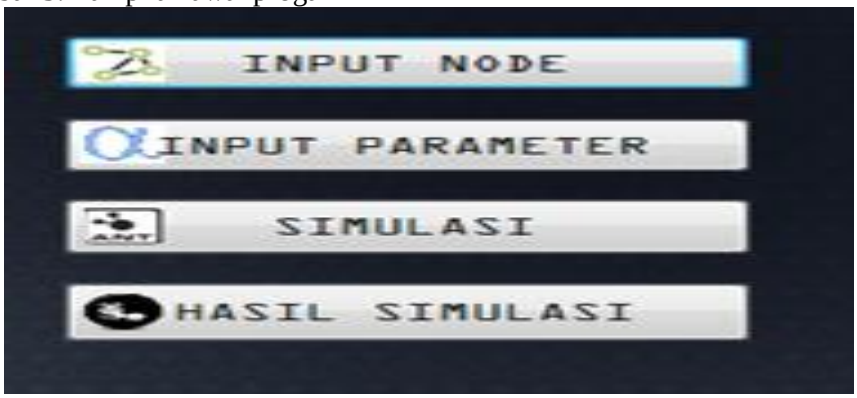
Impementasi *Ant Colony Optimization* (ACO) *Algorithm* diapikasikan dalam bahasa pemrogram VB.net. Aplikasi dari algoritma ini dibatasi hanya pada pencarian jalur terpendek dari data yang di*input*. Simulasi yang dilakukan adalah pencarian lintasan terpendek dengan menggunakan tiga nilai parameter dan nilai kondisi *pheromone* awal yang berbeda-beda.

Nilai alpha = 1, beta = 2, nilai kondisi *pheromone* awal = 0.0001 untuk perhitungan yang pertama, nilai alpha = 1, beta = 2, nilai kondisi *pheromone* awal = 1 untuk perhitungan yang kedua, dan nilai alpha = 1, beta = 5, nilai kondisi *pheromone* awal = 0.00000001 untuk perhitungan yang ketiga.

Tampilan awal program simulasi adalah sebagai berikut:

p-ISSN : 2443 – 0366
e-ISSN : 2528 – 0279

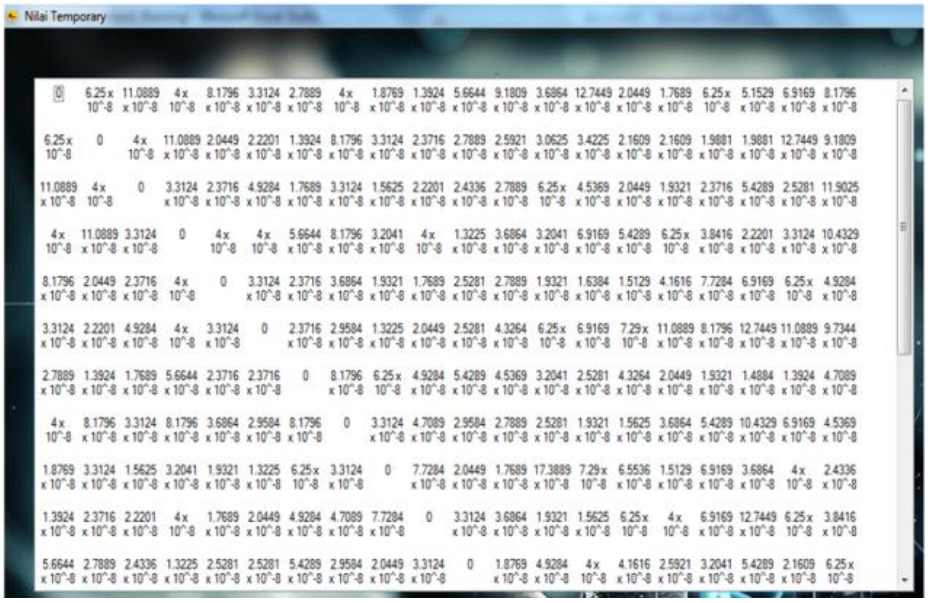
Gambar 3. Tampilan awal progam



Gambar 4. Menu-menu yang terdapat pada progam simulasi

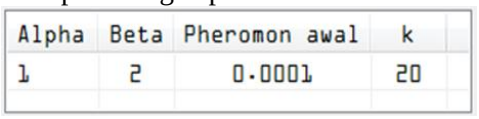
Proses simulasi nilai parameter dan kondisi *pheromone* awal adalah sebagai berikut:

p-ISSN : 2443 – 0366
e-ISSN : 2528 – 0279



Gambar 5. Nilai *Temporary* untuk perhitungan pertama

Pada Gambar 5 didapat nilai *temporary* dari parameter alpha = 1, beta = 2, dan nilai kondisi *pheromone* awal = 0.0001.



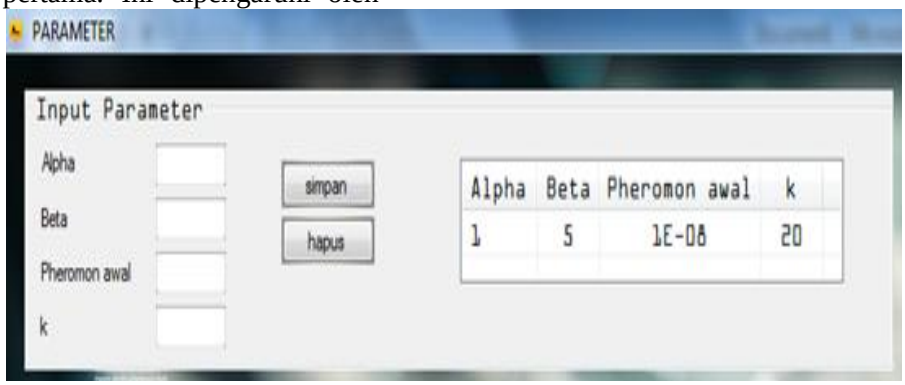
Gambar 6. Nilai parameter dan kondisi *pheromone* awal

Gambar 7. Nilai Parameter dan kondisi *pheromone* awal kedua

p-ISSN : 2443 – 0366
e-ISSN : 2528 – 0279

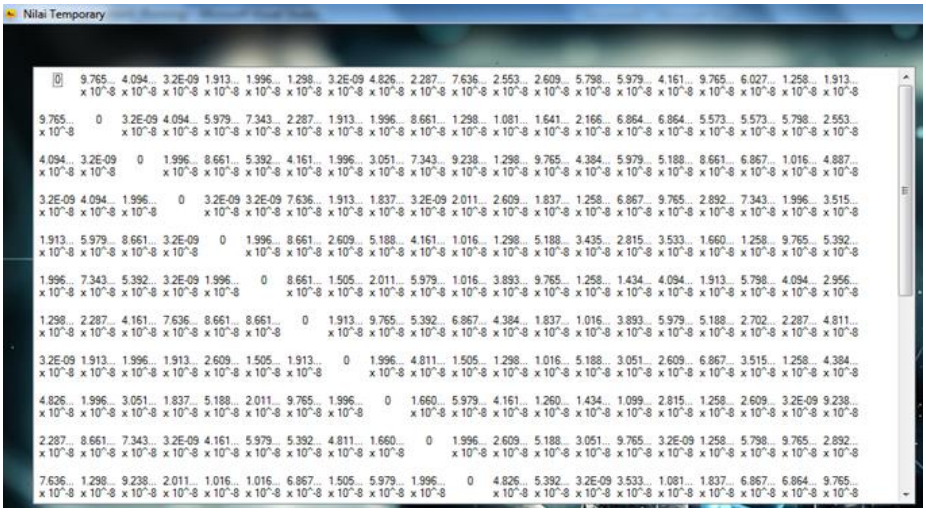
Gambar 8. Nilai *Temporary* untuk perhitungan kedua

Pada perhitungan nilai *temporary* yang kedua ini memiliki hasil yang berbeda dengan perhitungan yang pertama. Ini dipengaruhi oleh nilai *pheromone* awal yaitu 0.0001 (untuk perhitungan yang pertama) dan 1 (untuk perhitungan yang kedua).



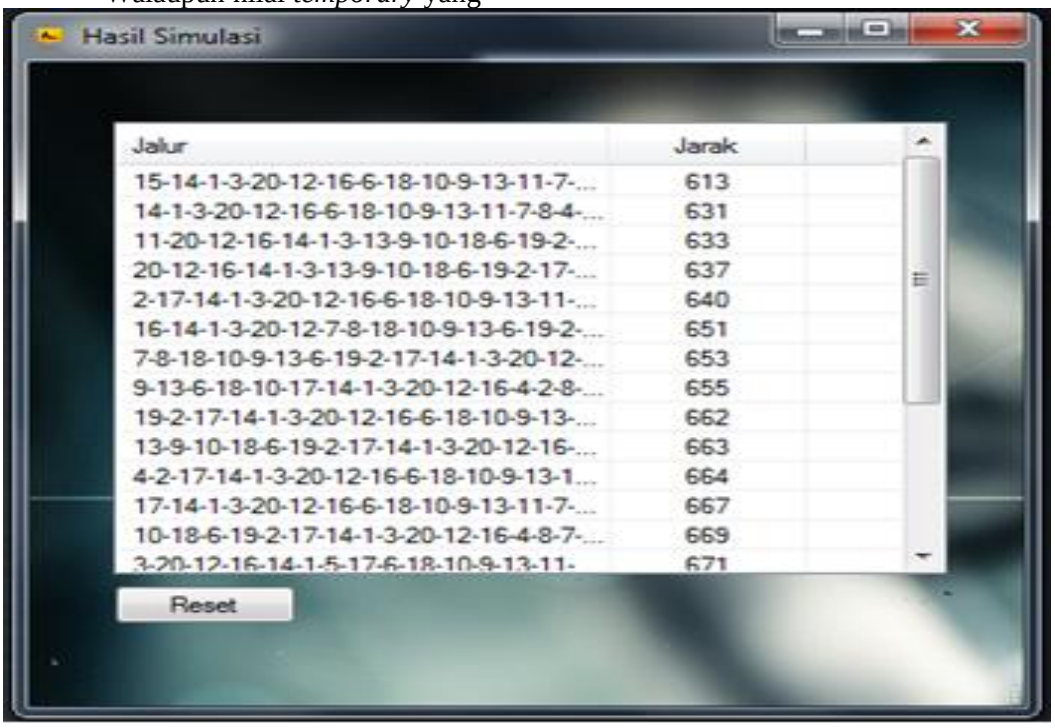
Gambar 9. Nilai parameter dan kondisi *pheromone* awal ketiga

p-ISSN : 2443 – 0366
e-ISSN : 2528 – 0279



Gambar 10. Nilai *Temporary* untuk perhitungan ketiga

Dari hasil perhitungan nilai *temporary* dari ketiga parameter dan kondisi *pheromone* awal yang berbeda, nilai *temporary* yang paling besar adalah dengan nilai kondisi *pheromone* awal = 1. Walaupun nilai *temporary* yang dihasilkan berbeda, namun lintasan terpendek yang didapatkan sama yaitu 15 – 14 – 1 – 3 – 20 – 12 – 16 – 6 – 18 – 10 – 9 – 13 – 11 – 7 – 8 – 4 – 2 – 17 – 5 – 19 dengan panjang 613(dalam km).



Gambar 11. Hasil Simulasi yang diperoleh

p-ISSN : 2443 – 0366
e-ISSN : 2528 – 0279

KESIMPULAN

Pada penelitian ini dilakukan simulasi lintasan terpendek pada *complete graph* dengan *ant colony optimization* (ACO) *algorithm* atau yang biasa disebut dengan algoritma semut menggunakan bahasa pemrograman *Visual Basic* (VB.net) dengan *database* SQL *Server*. *Complete Graph* yang digunakan dalam penelitian ini adalah graf K_{20}. Simulasi yang dilakukan pada penelitian ini menggunakan nilai parameter alpha, beta, dan nilai kondisi *pheromone* awal yang berbeda. Hal ini dilakukan untuk melihat peran penting nilai parameter dan pengaruh nilai kondisi *pheromone* awal terhadap nilai *temporary* yang dihasilkan. Nilai *temporary* digunakan untuk mencari dan mengetahui *node* yang akan dikunjungi oleh semut. Penentuan *node* awal ditentukan secara acak dan sebanyak simpul yang ada pada graf. Setelah semua *node* ditentukan *node* awalnya, selanjutnya sistem secara otomatis memproses dan menamplkan hasil pencarian.

Algoritma *ant colony optimization* dapat diterapkan pada penyelesaian *travelis salesman problem* serta digunakan untuk menyelesaikan masalah pencarian lintasan terpendek untuk *complete graph* K_{20}. Pencarian lintasan terpendek menggunakan data jarak, nilai alpha, beta, *pheromone* awal, dan juga banyaknya semut (*k*) yang digunakan 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*.

DAFTAR PUSTAKA

[1] 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.

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

[3] Ardiani, F., 2011. Penentuan Jarak Terpendek dan Waktu Tempuh Menggunakan Algoritma Dijkstra Dengan Pemrograman Berbasis Objek. Skripsi. Yogyakarta: Universitas Islam Negeri Sunan Kalijaga

[4] Munir, R., (2005), *Matematika Diskrit (Edisi Revisi Kelima)*, Penerbit Informatika, Bandung.

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

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

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

p-ISSN : 2443 – 0366
e-ISSN : 2528 – 0279

Negeri Syarif Hidayatullah, Jakarta.

[8] Ostfeld, A., (2011), *Ant Colony Optimization – Methods and Applications*, InTech, India.