Perbandingan Metode Ant Colony Optimization dan Dijkstra untuk Pengembangan Sistem Pengiriman Barang di Kantor Pos Area Surabaya Timur Berbasis J2ME

Aries Pratiarso, M.Zen Samsono Hadi, Mike Yuliana, dan Neny Wahyuningdiyah

Abstrak—Kantor pos memiliki berbagai layanan pada masyarakat, salah satunya adalah layanan pengiriman barang. Layanan ini menuntut pegawai pos untuk mengetahui kondisi wilayah Surabaya dengan baik agar bisa menghemat waktu dan biaya pengiriman. Pada penelitian ini dibuat pengembangan sistem mengenai rute pengiriman barang dengan jarak terpendek yang memudahkan pegawai kantor pos untuk mendistribusikan barang (paket) ke alamat yang dituju sehingga mampu meningkatkan kualitas layanan kantor pos. Penelitian ini bekerja sama dengan PT. Kantor Pos Jemur Sari dalam mendapatkan data wilayah pelanggan di area Surabaya Timur. Algoritma yang dipilih adalah *Ant Colony Optimization* (ACO), dimana dengan metode tersebut pencarian jalur terpendek menjadi lebih singkat walaupun menggunakan data yang banyak sekalipun. Urutan rute jalan yang dihasilkan oleh algoritma tersebut kemudian dapat diakses oleh pegawai pengirim paket melalui handphone berbasis Java 2 Micro Edition (J2ME). Sebagai pembanding, disertakan algoritma Dijkstra untuk menguji performa ACO.

Dari hasil pengujian didapatkan jarak terpendek yang sama. Namun, ACO membutuhkan waktu rata-rata 16,326 detik untuk mendapatkan jarak terpendek daripada waktu rata-rata Dijkstra yaitu 0,036 detik karena parameter yang digunakan *Ant Colony* lebih banyak dibandingkan dengan Dijkstra. Parameter ACO yang paling mempengaruhi jalannya eksekusi program adalah banyaknya siklus dan jumlah semut serta total node yang digunakan. Untuk interaksi *handphone client* dengan *server*, kecepatan mengakses informasi tergantung dari *throughput* yang diterima yaitu rata-rata 27,88 kbps. Login membutuhkan waktu lebih lama, rata-rata 14,57 detik sedangkan untuk mendapatkan rute membutuhkan waktu rata-rata 4,9 detik.

Kata Kunci—*Ant Colony Optimization*(ACO), Dijkstra, J2ME

1 PENDAHULUAN

PENELITIAN ini dibuat untuk membantu petugas kantor pos dalam memperbaiki layanan pengiriman paket pos. Dengan memanfaatkan teknologi telekomunikasi dan informatika, PT. Pos Indonesia (Persero) dapat meningkatkan kualitas pelayanan dalam hal pendistribusian barang (paket) dengan cara menempuh jarak terpendek dari Kantor Pos Pusat (penulis menggunakan kantor pos Jemur Sari untuk area Surabaya Timur) menuju pelanggan untuk efisiensi waktu dan meminimalisasi penggunaan BBM. Pihak pelanggan pun mendapatkan keuntungan yaitu barang kirimannya akan sampai dalam waktu yang diharapkan. Jika pelanggan puas karena layanan ini, maka *image* kantor pos yang lambat dalam proses pendistribusian surat dan paket akan sirna dan kepercayaan pelanggan akan meningkat.

Pada penelitian sebelumnya yang juga dilakukan oleh penulis telah ditentukan parameter terbaik dalam ACO (jumlah siklus, banyaknya semut, $\tau_{ij}, Q, \alpha, \beta dan \rho$) dalam menentukan jarak terpendek [4]. Dalam peneli-

- *Aries Pratiarso, Jurusan Teknik Telekomunikasi, Politeknik Elektronika Negeri Surabaya (email:aries@eepis-its.edu, Jl. Raya ITS Keputih Sukolilo, telp:031-5947280/ext:4109, fax:031-5946114) E-mail: aries@eepis-its.edu*
- *M. Zen Samsono Hadi, Mike Yuliana, Neny Wahyuningdiyah, Jurusan Teknik Telekomunikasi, Politeknik Elektronika Negeri Surabaya (Jl. Raya ITS Keputih Sukolilo, telp:031-5947280/ext:1501, fax:031-5946114). E-mail: zenhadi@eepis-its.edu, mieke@eepis-its.edu.*

tian yang lainnya, algoritma ACO digunakan untuk menentukan rute terpendek di pelabuhan Jordania [6]. Selain hal tersebut diatas, pada penelitian [7] telah dilakukan kombinasi algoritma ACO dan tabu search untuk menentukan rute terpendek. Algoritma ACO juga digunakan sebagai sistem navigasi perjalanan berbasis web yang terintegrasi dengan SIG (Sistem Informasi Geografis)[9].

Berdasarkan pada penelitian diatas, pada penelitian ini algoritma ACO diterapkan pada sistem pengiriman barang pada Kantor Pos Area Surabaya timur dalam menentukan rute terpendek sehingga memudahkan petugas dalam mendistribusikan paket-paketnya, dan juga lebih efisien dalam sisi waktu dan biaya. Algoritma tersebut juga dibandingkan dengan algoritma Dijkstra untuk mengetahui unjuk kerja terbaik dari kedua algoritma tersebut dalam kasus di kantor pos diatas.

J2ME (*Java 2 Micro Edition*) diaplikasikan dalam handphone untuk memudahkan *user* (pegawai pengirim paket pos) dalam mengakses informasi yang telah diolah oleh algoritma ACO. Pegawai kantor pos akan menerima informasi berupa rute jarak terpendek yang harus ditempuh dalam proses pendistribusian paket.

2 TEORI PENUNJANG

2.1 Algoritma Dijkstra

Ada beberapa kasus pencarian lintasan terpendek yang diselesaikan menggunakan algoritma Dijkstra, yaitu: pencarian lintasan terpendek antara dua buah simpul tertentu (*a pair shortest path*), pencarian lintasan terpendek antara semua pasangan simpul (*all pairs shortest path*), pencarian lintasan terpendek dari simpul tertentu ke semua simpul yang lain (*single-source shortest path*), serta pencarian lintasan terpendek antara dua buah simpul yang melalui beberapa simpul tertentu (*intermediate shortest path*). Intinya, algoritma *greedy* ini berupaya membuat pilihan nilai optimum lokal pada setiap langkah dan berharap agar nilai optimum lokal ini mengarah kepada nilai optimum global.

Input algoritma ini adalah sebuah graf berarah yang berbobot (*weighted directed graph*) G dan sebuah sumber vertex s dalam G dan V adalah himpunan semua vertices dalam graph G. Setiap sisi dari graf ini adalah pasangan *vertices* (u, v) yang melambangkan hubungan dari vertex u ke vertex v. Himpunan semua tepi disebut E. Bobot (*weights*) dari semua sisi dihitung dengan fungsi pada Persamaan (1)[8].

$$w : E \longrightarrow [0, \infty) \tag{1}$$

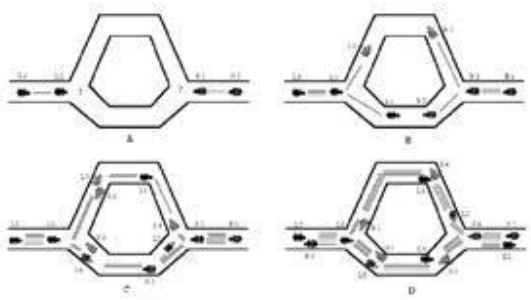
jadi $w(u, v)$ adalah jarak tak-negatif dari vertex u ke vertex v. Ongkos (*cost*) dari sebuah sisi dapat dianggap sebagai jarak antara dua vertex, yaitu jumlah jarak semua sisi dalam jalur tersebut. Untuk sepasang vertex s dan t dalam V, algoritma ini menghitung jarak terpendek dari s ke t. Berikut adalah algoritma Dijkstra [8]:

```
function Djikstra (G, w, s)
//Initializations
for each vertex v in V[G]
    d[v] := infinity
    previous[v] := undefined
d[s] := 0 // Jarak dari s ke s
S := empty set
Q := V[G] //Set semua vertex
while Q is not an empty set
    u := Extract_Min(Q)
    S := S union u
    for each edge (u,v) outgoing from u
        if d[u] + w(u,v) ¡ d[v]
            d[v] := d[u] + w(u,v)
            previous[v] := u
```

2.2 Algoritma Koloni Semut (*Ant Colony*)

Dasar dari perumusan algoritma *ant colony system* adalah kemampuan dari sekumpulan semut (*colony*) yang dapat menemukan jalur terpendek dari sumber makanan ke sarangnya. Hal ini dapat dilakukan karena seekor semut akan meninggalkan jejak *pheromone* ketika dia melalui suatu lintasan. Dengan bantuan *pheromone* ini juga sekumpulan semut dapat beradaptasi terhadap perubahan dalam jalur yang telah mereka lalui. Untuk lebih jelasnya terlihat dalam ilustrasi pada Gambar 1.

Ant colony system(koloni semut), algoritma yang digunakan untuk menyelesaikan permasalahan pada penelitian ini, merupakan algoritma yang berdasarkan algorima *ant system* dengan meningkatkan efisiensi pencarian



Gambar 1. Perilaku semut pada dunia nyata [10]

rute yang dilalui. Konsep dasar dari algoritma koloni semut adalah dengan menggunakan sekumpulan semut yang bertujuan mencari lintasan terpendek secara pararel.

Secara garis besar, *ant system* dapat diterangkan sebagai berikut. Setiap semut akan membentuk rute dengan memilih kota-kota yang dikunjungi sesuai dengan *state transition rule*: Seekor semut akan lebih memilih kota yang terhubung dengan *arc* yang lebih pendek dan memiliki jumlah *pheromone* yang lebih besar.

Setelah semut-semut tersebut menyelasaikan penyusunan rutenya, maka proses selanjutnya adalah *global pheromone updating*. Pada tahap ini terjadi penguapan sejumlah *pheromone* pada setiap cabang, kemudian tiap-tiap semut akan menambahkan sejumlah *pheromone* pada cabang yang dilaluinya dengan jumlah yang berbanding terbalik dengan jarak yang ditempuh.

State transition rule yang digunakan dalam ant system adalah [6]:

$$Pk(r,s) = \begin{cases} \frac{[\tau(r,s)][\eta(r,s)]^{\beta}]}{\Sigma_{u \epsilon Jk(r)}[\tau(r,u)][\eta(r,u)]^{\beta}}, & \text{jika } s \epsilon J_k(r) \\ 0, & \text{lainnya} \end{cases} \quad (2)$$

$P_k(r,s)$ merupakan probabilitas semut k yang berada di node r memilih node s untuk tujuan selanjutnya. Sedangkan $\tau(r,s)$ adalah *pheromone* yang terdapat pada $arc(r,s)$, $J_k(r)$ merupakan himpunan *node* yang belum dikunjungi oleh semut k yang berada pada node r dan β adalah parameter yang menentukan besarnya pengaruh jarak terhadap jumlah *pheromone*. Untuk *visibility measure*, $\eta(r,s)$, dapat dihitung dengan Persamaan (3) [6]:

$$\eta(r,s) = \frac{1}{\delta(r,s)} \quad (3)$$

dengan $\delta(r,s)$ merupakan biaya pada (r,s). Dalam *ant system*, apabila semua semut telah menyelesaikan rute yang dibentuk maka terjadi perubahan jumlah pheromone, disebut *global pheromone updating rule*dengan Persamaan (4) [6]:

$$\tau(r,s) \leftarrow (1,\alpha).\tau(r,s) + \alpha.\Sigma_{k=1}^{\infty}\Delta\tau_k(r,s) \quad (4)$$

dengan $\Delta\tau(r,s) = \frac{1}{L_k}$, jika $(r,s)\epsilon$ rute yang dilalui semut, dan $\Delta\tau(r,s) = 0$, jika rute tidak dilalui semut.

Pada persamaan (4), parameter α disebut *global pheromone decay*, Lk merupakan panjang dari tour yang dilakukan oleh semut k dan m adalah jumlah semut. Perubahan pheromone diatas bertujuan untuk memberikan jumlah *pheromone* yang lebih besar pada *tour* yang lebih pendek.

2.3 Java 2 Micro Edition (J2ME)

J2ME adalah satu set spesifikasi dan teknologi yang fokus kepada perangkat konsumen. Program-program J2ME, seperti semua program Java, di*compile* ke dalam bytecode dan diterjemahkan dengan Java Virtual Machine (JVM).

Inti dari J2ME terletak pada konfigurasi dan profil-profil. Suatu konfigurasi menggambarkan lingkungan *runtime* dasar dari suatu sistem J2ME. Ia menggambarkan *core library*, *virtual machine*, fitur keamanan dan jaringan.

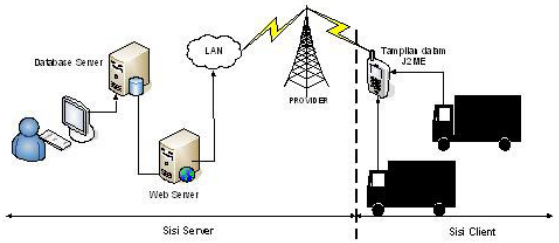
3 IMPLEMENTASI SISTEM

3.1 Perancangan Sistem

Secara keseluruhan, sistem dibedakan menjadi dua sisi, yaitu sisi *client* dan sisi *server*. Di sisi *server*, ada pegawai yang bertugas untuk memasukkan data pengiriman paket pos dari pelanggan ke dalam *database server*, disebut sebagai administrator/admin. Oleh *server*, data yang mengandung nama jalan akan diolah menggunakan algoritma ACO untuk mendapatkan rute dengan jarak terpendek menuju

pelanggan. Hasilnya dapat ditampilkan pada ponsel berbasis J2ME. Algoritma ACO akan menerima masukan data yaitu node awal untuk memulai perjalanan dan node akhir sebagai akhir perjalanan, setelah diproses akan dihasilkan rute terpendek dan ditampilkan node-node yang akan dilewati oleh pegawai pos.

Selain itu, rute terpendek yang dihasilkan ACO akan dibandingkan dengan hasil perhitungan menggunakan algoritma Dijkstra. Sedangkan di sisi *client*, pengemudi dapat mengakses informasi berupa rute terpendek dari database tersebut melalui ponsel berbasis J2ME dengan cara memasukkan *username* dan *password* pegawai untuk mengetahui kemana rute yang harus ditempuh. Blog diagram dari sistem ditunjukkan pada Gambar 2.



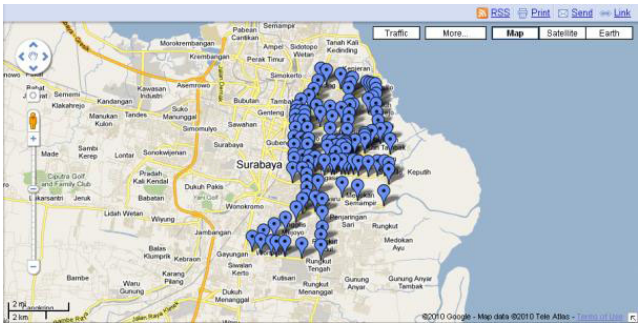
Gambar 2. Blok diagram sistem

3.2 Data Peta Surabaya Timur

Wilayah Surabaya Timur dibuat beberapa *node* yang digunakan untuk keperluan pemodelan jaringan jalan. Pemetaan *node* ini memanfaatkan *Google Maps* secara online yaitu dengan cara menandai setiap persimpangan dan tempat-tempat yang familiar dikunjungi oleh masyarakat (point), misalnya: pasar, rumah sakit, SPBU, dll. Selain itu, dicari pula jarak yang menghubungkan dua persimpangan tersebut.

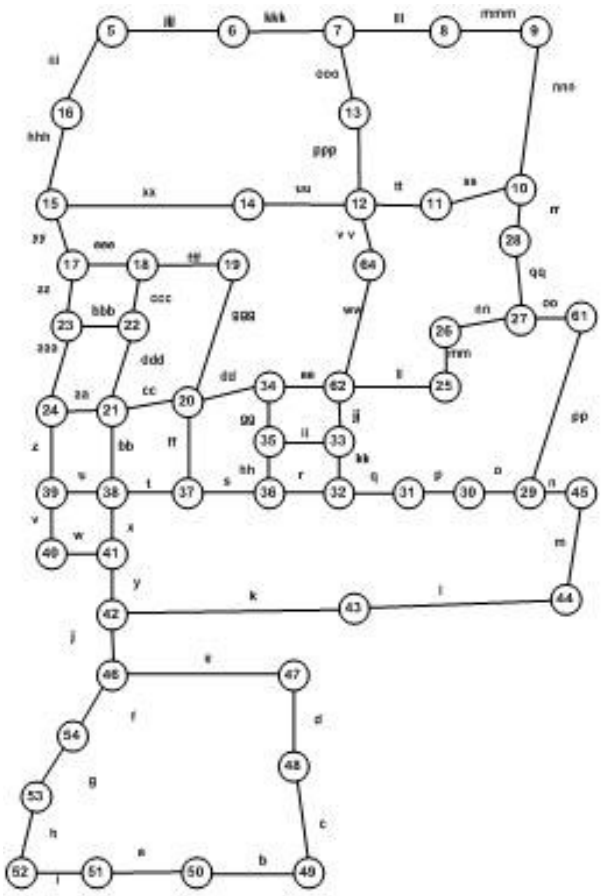
3.3 Pemodelan Jaringan Jalan

Jalan yang terdapat dalam peta sebenarnya seperti yang ditunjukkan pada Gambar 3 dapat dimodelkan dalam bentuk *graph*. *Node-node* yang berbentuk lingkaran dengan angka ditengah disebut dengan ”simpangan”. Garis yang menghubungkan dua node disebut dangan ”jalan”. Setiap jalan memiliki satu ruas,



Gambar 3. Peta node Surabaya Timur di Google maps

kecuali jika jarak jalan tersebut sangatlah panjang, maka jalan akan dibagi menjadi beberapa ruas (syarat dan ketentuan berlaku). Pemodelan ini dibuat sebagai representasi dari jalan-jalan yang ada di area Surabaya Timur seperti yang terlihat pada Gambar 4, dengan pemodelan ini diharapkan sudah mewakili area yang dicakup dalam penelitian ini.

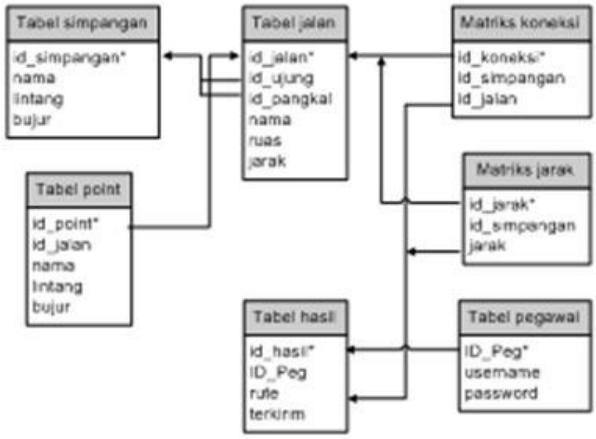


Gambar 4. Representasi graph

3.4 Relasi dengan Database

Dalam sistem yang dirancang terdapat 7 tabel yang berkaitan dengan pemodelan jaringan jalan yang digunakan.Relasi tabel pada database ditunjukkan pada Gambar 5. Tabel tersebut adalah sebagai berikut:

1) Tabel SIMPANGAN, berisi field id_simpangan, nama, lintang dan bujur.
2) Tabel JALAN, terdiri dari field id_jalan, id_ujung, id_pangkal, nama, ruas dan jarak.
3) Tabel POINT, terdiri dari field id_point, id_jalan, nama, lintang dan bujur.
4) Tabel PEGAWAI, terdiri dari field ID_Peg, username, dan password.
5) Tabel HASIL, terdiri dari field id_hasil, ID_Peg, rute, dan terkirim.
6) Tabel KONEKSI, berisi field idkoneksi yang membentuk matriks yang berisi id_jalan.
7) Tabel JARAK, berisi field id_jarak yang membentuk matriks yang berisi jarak.



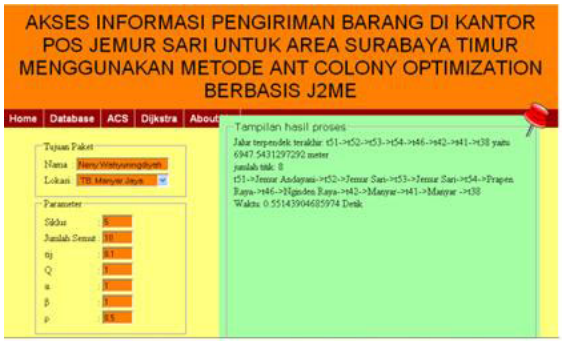
Gambar 5. Relasi tabel pada database

3.5 Interaksi Client-Server

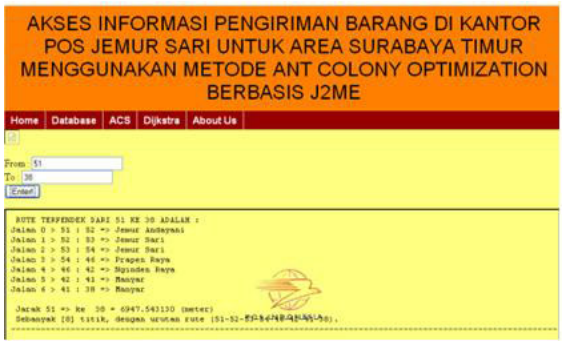
Dalam sistem ini dibuat interaksi antara server yang berbasis PHP dengan *client* yang berbasis *J2ME*. Untuk bisa login, pegawai pos yang akan mengirimkan paket harus memasukkan *username* dan *password* pada halaman awal dari tampilan J2ME, setelah itu, pegawai pos akan mendapatkan data pelanggan mana saja yang harus dituju dan rute mana saja yang harus ditempuh. Data ini merupakan data hasil pengolahan dengan menggunakan algoritma *Ant Colony Optimization* (ACO). Gambar 6 menunjukkan hasil tampilan pada handphone client yang dibuat.

Gambar 6. Tampilan pada handphone client

Untuk Admin, *interface* yang digunakan untuk mengakses *server* berupa website yang akan mengolah input untuk perhitungan algoritma ACO dan Dijkstra. Tampilan hasil pengujian ACO yang dibuat ditunjukkan pada Gambar 7, sedangkan tampilan hasil engujian Djikstra ditunjukkan pada Gambar 8.



Gambar 7. Tampilan hasil pengujian ACO



Gambar 8. Tampilan hasil pengujian Dijkstra

4 HASIL DAN ANALISA

4.1 Pengujian Algoritma

Untuk melakukan pengujian pada algoritma, maka perlu diketahui parameter-parameter yang digunakan oleh algoritma koloni semut dan Dijkstra. Parameter pada *Ant Colony* antara lain:

1) Siklus yaitu banyaknya siklus maksimum pencarian jalur terpendek.
2) Semut yaitu banyaknya semut yang akan melakukan dalam satu kali siklus.
3) τ_{ij} adalah intensitas jejak semut antar titik dan perubahannya.
4) Q adalah tetapan siklus semut dalam melakukan pencarian jalur.
5) α (alfa) adalah tetapan pengendali intensitas jejak semut, dimana $\alpha \geq 0$
6) β (beta) adalah tetapan pengendali visibilitas, dimana $\beta \geq 0$.
7) ρ (rho) adalah tetapan penguapan jejak semut untuk mencegah jejak semut yang tak terhingga, dimana $0 < \rho < 1$.
8) Asal adalah kantor pos sebagai depo.
9) Point adalah titik tujuan yang diasumsikan sebagai node pelanggan.

Dijkstra hanya memerlukan masukan berupa titik asal dan titik tujuan (simpangan) untuk menghasilkan jarak terpendek.

Tiga macam rute yang dijadikan acuan pengujian adalah:

1) Jarak Dekat: dari 51 ke 38 (point: TB. Manyar Jaya)
2) Jarak Menengah: dari 51 ke 62 (point: Natasha Skin Care)
3) Jarak Jauh: dari 51 ke 7 (point: Hotel Puspa Asri)

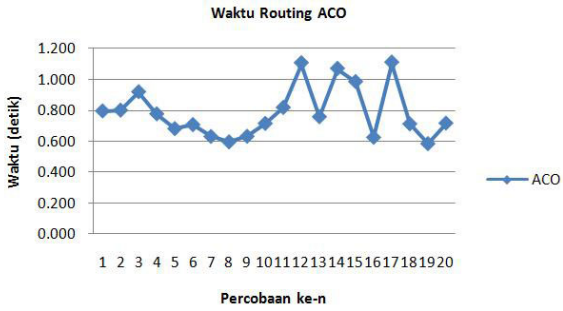
4.1.1 Pengujian Jarak Dekat

Jarak dekat berasal dari simpangan 51 menuju simpangan 38: t51 → t52 → t53 → t54 → t46 → t42 → t41 → t38, artinya melalui jalan Jemur Andayani → Jemur Sari → Jemur Sari → Prapen Raya → Manyar. Penulis menggunakan parameter terbaik yang diperoleh dari penelitian sebelumnya [4], yaitu:

- Siklus = 5
- Jumlah semut = 10
- $\tau_{ij} = 0.01$
- Q=1
- $\alpha = 1$
- $\beta = 1$
- $\rho = 1$

Dari uji coba yang ditunjukkan pada Gambar 9, Jarak terpendek yang dihasilkan oleh ACO dan Dijkstra adalah sama, sekitar 6947,54 meter. ACO memerlukan waktu antara 0,5 sampai 1,2 detik untuk menghasilkan jarak terpendek, sedangkan Dijkstra yang hanya membutuhkan waktu berkisar antara 0,03 dan 0,035 detik. Rata-rata waktu untuk ACO adalah 0,787 detik dan 0,033 detik untuk Dijkstra. Bila dinyatakan dalam grafik, rata-rata grafik waktu tempuh untuk ACO dan untuk Dijkstra untuk jarak pendek ditunjukkan pada Gambar 10 dan Gambar 11.

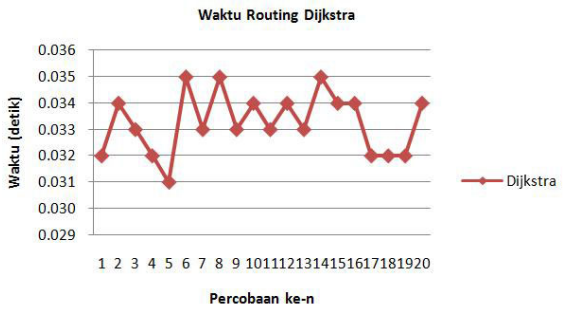
Gambar 9. Grafik jarak terpendek untuk jarak dekat



Gambar 10. Grafik waktu tempuh ACO untuk jarak dekat

4.1.2 Pengujian Jarak Menengah

Jarak menengah berasal dari simpangan 51 menuju simpangan 62 menghasilkan rute sebagai berikut: t51 → t52 → t53 → t54 → t46 → t42 → t41 → t38 → t37 → t36 → t35 → t33



Gambar 11. Grafik waktu tempuh Dijkstra untuk jarak dekat

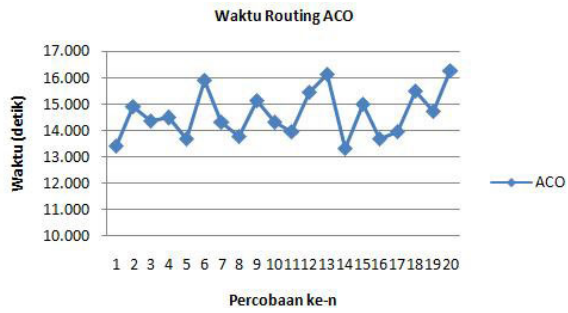
Gambar 12. Grafik jarak menengah untuk jarak menengah

→ t62 atau melalui jalan Jemur Andayani → Jemur Sari → Jemur Sari → Prapen Raya → Manyar → Manyar → Menur Pumpungan → Menur Pumpungan → Kertajaya Indah Tengah → Manyar Kertoadi → Kertajaya Indah Timur.

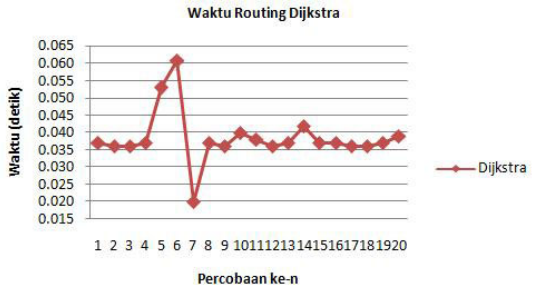
Parameter terbaik yang digunakan diperoleh dari penelitian sebelumnya [4], yaitu:

- Siklus = 30
- Jumlah semut = 40
- $\tau_{ij} = 0.1$
- $Q = 1$
- $\alpha = 1$
- $\beta = 1$
- $\rho = 0.5$

Jarak terpendek yang dihasilkan oleh ACO dan Dijkstra dalam menempuh "jarak menengah" adalah 9826,98 meter. Yang memiliki perbedaan jauh adalah waktu yang dibutuhkan kedua algoritma untuk memperoleh jarak terpendek ketika menjalankan program. ACO memerlukan waktu lebih lama, yaitu puluhan detik dalam range waktu antara 13 sampai 17 detik yang sangat kontras dengan Dijkstra yang hanya berkisar antara 0,02 dan 0,061 detik. Dalam hal ini, rata-rata waktu yang dibutuhkan ACO adalah 14,63 detik sedangkan rata-rata waktu algoritma Dijkstra adalah 0,038 detik. Bila dinyatakan dalam grafik, rata-rata grafik waktu tempuh untuk ACO dan untuk Dijkstra untuk jarak menengah ditunjukkan pada Gambar 13 dan Gambar 14.



Gambar 13. Grafik waktu tempuh ACO untuk jarak menengah



Gambar 14. Grafik waktu tempuh Dijkstra jarak menengah

4.1.3 *Pengujian Jarak Jauh*

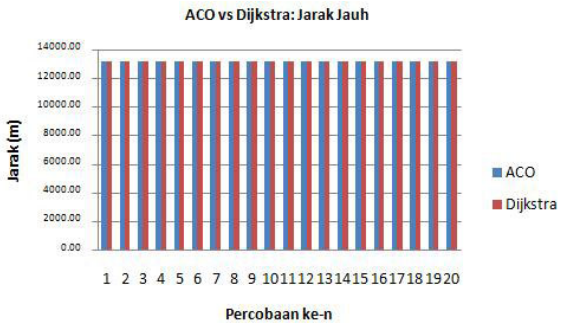
Hasil pengujian algoritma ACO dan Dijkstra untuk rute "jarak jauh" yang berasal dari simpangan 51 menuju simpangan 7 adalah: t51 → t52 → t53 → t54 → t46 → t42 → t41 → t38 → t37 → t36 → t35 → t33 → t62 → t64 → t12 → t13 → t7, yaitu melalui jalan Jemur Andayani → Jemur Sari → Jemur Sari → Prapen Raya → Manyar → Manyar → Menur Pumpungan → Menur Pumpungan → Kertajaya Indah Tengah → Manyar Kertoadi → Kertajaya Indah Timur → Dharmahusada Indah Timur → Dharmahusada Indah Utara → Raya Kalijudan MERR II → Raya Kalijudan MERR II.

Parameter terbaik telah diperoleh dari penelitian sebelumnya [4], yaitu:

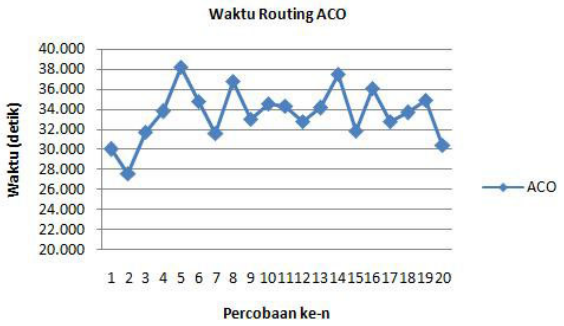
- Siklus = 50

- Jumlah semut = 45
- $\tau_{ij} = 0.01$
- $Q = 1$
- $\alpha = 1$
- $\beta = 1$
- $\rho = 0.5$

Jarak terpendek yang dihasilkan oleh ACO dan Dijkstra untuk rute "jarak jauh" adalah sepanjang 13288,02 meter. Selain itu, ACO memerlukan waktu antara 27 sampai 39 detik untuk menghasilkan rute sedangkan Dijkstra yang hanya memerlukan waktu 0,02 sampai 0,046 detik. Atau dapat disimpulkan bahwa rata-rata waktunya adalah 33,56 detik untuk ACO dan 0,037 detik untuk Dijkstra.Bila dinyatakan dalam grafik, rata-rata grafik waktu tempuh untuk ACO dan untuk Dijkstra untuk jarak jauh ditunjukkan pada Gambar 16 dan Gambar 17.



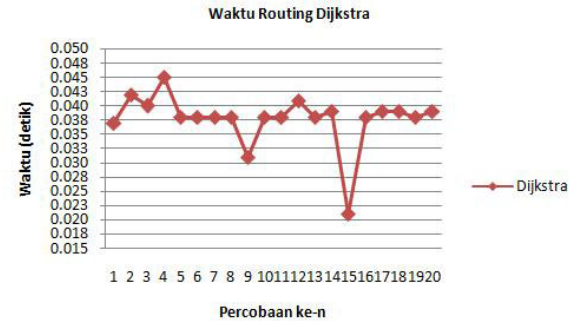
Gambar 15. Grafik jarak jauh untuk jarak jauh



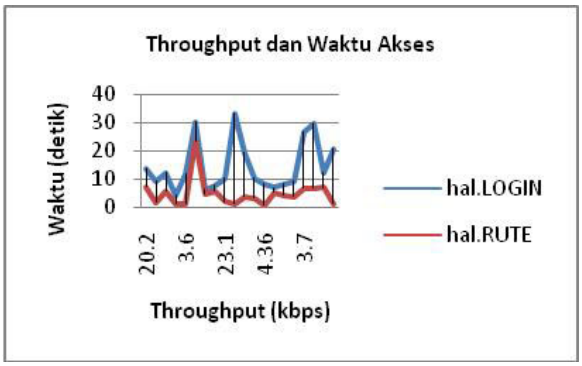
Gambar 16. Grafik waktu tempuh ACO untuk jarak jauh

4.2 Pengujian Koneksi

Parameter yang digunakan adalah lama waktu pengaksesan informasi dari *server* saat pertama kali login dengan *username* dan *password* pegawai tersebut (hal.login) dan lama waktu yang dibutuhkan untuk memperoleh rute jarak terpendek (hal.rute). Selain itu juga dicari *throughput*, yaitu *bandwidth* aktual yang diukur secara spesifik menggunakan software "Bandwidth Meter". Pengujian dilakukan selama jam kerja pegawai pengantar paket, yaitu mulai dari pukul 08.00 sampai pukul 12.00.



Gambar 17. Grafik waktu tempuh Dijkstra jarak jauh



Gambar 18. Grafik *throughput* dan waktu akses

Gambar 18 menyatakan bahwa *client* membutuhkan waktu yang lebih lama untuk login daripada waktu untuk mendapatkan informasi rute jarak terpendek dari *server*. Rata-rata waktu untuk login adalah 14,57 detik dan 4,9 detik untuk masuk ke halaman rute. Hal ini terjadi karena ketika login, *client* membutuhkan waktu untuk terhubung dengan *server* dan mengambil data dari *database*. Sedangkan saat masuk halaman rute, *client* sudah terhubung dengan *server*, hanya perlu mengambil data berupa rute jarak terpendek di dalam database.

Perbedaan waktu tersebut juga dipengaruhi oleh *throughput* yang didapatkan dari jaringan.

Rata-rata *throughput* pada 20 kali pengujian adalah 27,88 kbps.

5 KESIMPULAN

1) Pencarian jalur terpendek dengan metode koloni semut tergantung dari parameter-parameter yang dimasukkan antara lain: banyaknya titik, banyak semut, Tij (tetapan awal intensitas feromon), Alfa (tetapan pengendali intensitas feromon), Beta (tetapan pengendali visibilitas), Rho (tetapan penguapan feromon).
2) Perbandingan algoritma koloni semut dengan Dijkstra menghasilkan jarak terpendek yang sama baik untuk rute jarak dekat, jarak menengah, maupun jarak jauh.
3) Algoritma koloni semut membutuhkan waktu rata-rata 16,326 detik untuk mendapatkan jarak terpendek daripada waktu rata-rata Dijkstra yaitu 0,036 detik karena parameter yang digunakan Ant Colony lebih banyak dibandingkan dengan Dijkstra.
4) Secara umum, client menghabiskan waktu rata-rata 14,57 detik untuk login karena pertama kali melakukan sambungan dengan server melalui jaringan internet dan 4,9 detik untuk mendapatkan rute. Kecepatan akses data juga dipengaruhi oleh throughput. Rata-rata throughput yang diperoleh adalah 27,88 kbps.

DAFTAR PUSTAKA

[1] Dorigo, M., Gambardella, L. M. (1997).Ant colonies for the TSP Tech.Rep/IRIDIA/1996-003, Universit Libre de Bruxelles, Belgium.
[2] Putro, Fidi W. "Sistem Navigasi Berbasis Web dengan Algoritma Koloni Semut". T.Informatika PENS-ITS Surabaya. 2009
[3] Mutakhiroh, Iing. "Pemanfaatan Metode Heuristik dalam Pencarian Jalur Terpendek dengan Algoritma Semut dan Algoritma Genetika". Lab.Pemrograman dan Informatika, Universitas Islam Indonesia. 2007.
[4] Hadi, M.Zen, Haryadi Amran, Titik Sri Mulyani, "Akses Informasi Pengiriman Barang di Kantor Pos Jemursari untuk Area Surabaya Timur Menggunakan Metode Ant Colony Optimization Berbasis WAP", Seminar IES 2010, PENS-ITS.
[5] Mutakhiroh, I., Saptono, F., Hasanah, N., dan Wiryadinata, R,. (2007). Pemanfaatan Metode Heuristik Dalam Pencarian Jalur Terpendek Dengan Algoritma Semut dan Algoritma Genetik Seminar Nasional Aplikasi Teknologi Informasi. ISSN: 1907-5022. Yogyakarta.
[6] Saad Ghaleb Yaseen, Nada M. A.AL-Slamy, Ant Colony Optimization, IJCSNS International Journal of Computer Science and Network Security, VOL.8 No.6, June 2008
[7] Masaya Yoshikawa, Kazuo Otani, Ant Colony Optimization Routing Algorithm with Tabu Search, Proceedings of the International MultiConference of Engineers and Computer Scientists 2010 Vol III, IMECS 2010, March 17-19, Hongkong
[8] Aries Pratiarso, M. Agus Zainudin, Transmisi Citra Menggunakan Joint LDPC Decoding, Proceeding Seminar Nasional Aplikasi Teknologi Informasi (SNATI), UII Yogyakarta, 2009
[9] Arna Fariza, Entin Martiana, Fidi Wincoko Putro, Sistem Navigasi Perjalanan Berbasis Web dengan Algoritma Koloni Semut (Ant Colony Algorithm), Seminar IES 2009, PENS-ITS
[10] http://id.wikipedia.org/wiki/Algoritma_Dijkstra
[11] http://en.wikipedia.org/wiki/Ant_colony_optimization

Aries Pratiarso lahir di Surabaya, ia memperoleh gelar Sarjana Teknik (ST) pada Jurusan Teknik Elektro pada tahun 1994 dan Magister Teknik (MT) pada tahun 2004, keduanya dari Institut Teknologi Sepuluh Nopember (ITS) Surabaya. Ia adalah pengajar pada jurusan Teknik Telekomunikasi, Politeknik Elektronika Negeri Surabaya. Bidang penelitian yang ditekuni adalah Wireless Communication, teknik koding, dan image processing.

M. Zen Samsono Hadi di Kediri, ia memperoleh gelar Sarjana Teknik (ST) pada Jurusan Teknik Elektro pada tahun 2000 dan magister teknik (MT) pada tahun 2009, keduanya dari Institut Teknologi Sepuluh Nopember (ITS) Surabaya. Pada tahun 2007, ia mendapat kesempatan untuk program *double degree* ke Jerman, dan mendapatkan gelar *master of science* pada tahun 2008. Ia adalah pengajar pada jurusan Teknik Telekomunikasi, Politeknik Elektronika Negeri Surabaya. Bidang penelitian yang ditekuni adalah Network Security, Network Design dan internet application. Pernah melakukan penelitian pada bidang network design di T-Systems Enterprise GmbH, Darmstadt Jerman pada tahun 2008.

Mike Yuliana lahir di Jember, ia memperoleh gelar Sarjana Teknik pada Jurusan Teknik Elektro pada tahun 2001 dan magister pada tahun 2007, keduanya dari Institut Teknologi Sepuluh Nopember (ITS) Surabaya. Ia adalah pengajar pada jurusan Teknik Telekomunikasi, Politeknik Elektronika Negeri Surabaya. Bidang penelitian yang ditekuni adalah Telephony Network, dan Network Security. Banyak melakukan penelitian yang berbasis aplikasi VoIP, mulai dari pembuatan server VoIP sampai pembuatan program untuk menambahkan fitur dari server VoIP.

Neny Wahyuningdiyah lahir di Sidoarjo, ia memperoleh gelar Sarjana Science Terapan (SST) pada Jurusan Teknik Telekomunikasi pada tahun 2010 dari PENS-ITS Surabaya. Bidang penelitian yang ditekuni adalah Jaringan Komputer dan aplikasi algoritma Ant Colony Optimization (ACO).