Akses Informasi Pengiriman Barang Di Kantor Pos Jemur Sari Untuk Area Surabaya Timur Menggunakan Metode Ant Colony Optimization Berbasis J2ME

Neny Wahyuningdiyah[1], M.Zen Samsono Hadi[2], Mike Yuliana[2]
[1]Mahasiswa Politeknik Elektronika Negeri Surabaya, Jurusan Teknik Telekomunikasi
[2]Dosen Politeknik Elektronika Negeri Surabaya Institut Teknologi Sepuluh Nopember
Kampus ITS, Surabaya 60111
e-mail : neny@student.eepis-its.edu e-mail : zenhadi@eepis-its.edu, mieke@eepis-its.edu

Abstrak

Saat ini, kantor pos dapat dikatakan sebagai ”sentra layanan masyarakat” karena pelbagai macam layanan terdapat disana, seperti layanan pembayaran rekening listrik, air, dan telepon serta menabung. Dengan memanfaatkan teknologi informatika (IT) dan telekomunikasi, semua data terkomputerisasi. Sistem yang dibangun mampu meningkatkan kualitas layanan kantor pos.

Pada tugas akhir ini akan dibuat sistem informasi mengenai rute pengiriman barang dengan jarak terpendek yang memudahkan pegawai kantor pos untuk mendistribusikan barang (paket) ke alamat yang dituju. Algoritma yang dipilih adalah *Ant Colony Optimizatian* (ACO) yang menghasilkan urutan rute jalan yang dapat diakses oleh pegawai pengirim paket melalui *handphone* berbasis *Java 2 Micro Edition* (J2ME). Keuntungan lain adalah untuk menghemat biaya BBM armada pengirim. 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*, dan *Java 2 Micro Edition* (J2ME).

1. Pendahuluan

Tugas akhir ini dibuat untuk membantu 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 depo (penulis menggunakan kantor pos Jemur Sari) 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.

Jarak terpendek yang dimaksud adalah hasil dari perhitungan beberapa parameter melalui *Ant Colony Optimization* (ACO). ACO merupakan salah satu algoritma yang didasarkan atau terinspirasi dari perilaku sosial semut dalam suatu koloni yang saling berinteraksi, bernegoisasi, dan berkoordinasi satu sama lain dalam mengerjakan suatu pekerjaan bersama. Parameter yang terdapat dalam algoritma ini antara lain: jumlah siklus, banyaknya semut, τ_{ij}, Q, α, β dan ρ yang sangat

mempengaruhi perolehan jarak terpendek. Setelah menemukan parameter terbaiknya [4], lalu jarak terpendek tersebut dibandingkan dengan algoritma Dijkstra untuk mengetahui apakah hasilnya sama dengan algoritma ACO. 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

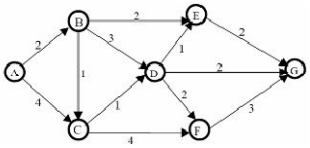
2.1. *Graph*

Graf adalah kumpulan simpul (nodes) yang dihubungkan satu sama lain dengan garis atau busur (edge) (Zakaria, 2006). Suatu Graf terdiri dari dua himpunan yaitu himpunan V dan E:

a. Verteks (simpul) adalah himpunan simpul yang terbatas dan tidak kosong.
b. Edge (Sisi/busur) adalah himpunan busur yang menghubungkan sepasang simpul.

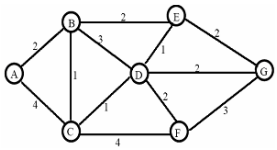
Simpul-simpul pada graf adalah suatu obyek yang mewakili suatu kota atau tempat dan sebagainya. Busur dapat mewakili obyek seperti, jalan raya, sambungan telepon, dan lain-lain. Menurut arah dan beban yang dimiliki oleh busur, maka Graf dibedakan sebagai berikut:

a. Graf berarah dan berbobot: tiap busur mempunyai anak panah dan bobot. Lihat contoh gambar 2.1.



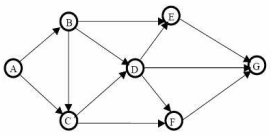
Gambar 2.1 Graf Berarah dan Berbobot

b. Graf tidak berarah dan berbobot: busur tidak mempunyai panah, tetapi tetap memiliki bobot atau beban pada setiap busurnya. Lihat contoh gambar 2.2.



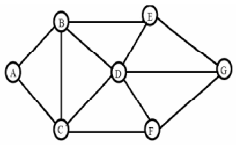
Gambar 2.2 Graf tidak berarah dan berbobot

c. Graf berarah dan tidak berbobot: Busur memiliki arah, tetapi tidak memiliki bobot atau beban pada setiap busurnya. Lihat contoh gambar 2.3.



Gambar 2.3 Graf berarah dan tidak berbobot.

d. Graf tidak berarah dan tidak berbobot: Busur tidak memiliki arah dan tidak memiliki bobot pada setiap busurnya. Lihat contoh gambar 2.4.



Gambar 2.4 Graf tidak berarah dan tidak berbobot.

2.2. Algoritma *Dijkstra*

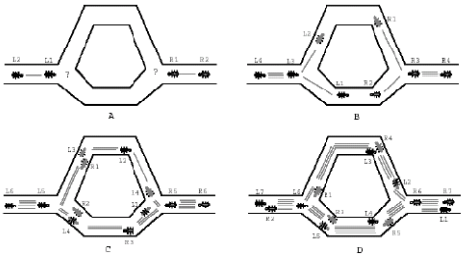
Algoritma Dijkstra, dinamai menurut penemunya, Edsger Dijkstra, merupakan salah satu varian dari algoritma *greedy,* yaitu salah satu bentuk algoritma populer dalam pemecahan persoalan yang terkait dengan masalah optimasi. Sifatnya sederhana dan lempang (*straight-forward*). Sesuai dengan artinya yang secara harfiah berarti tamak atau rakus (namun tidak dalam konteks negatif), algoritma *greedy* ini hanya memikirkan solusi terbaik yang akan diambil pada setiap langkah tanpa memikirkan konsekuensi ke depan. Prinsipnya, ambillah apa yang bisa Anda dapatkan saat ini (*take what you can get now!*), dan keputusan yang telah diambil pada setiap langkah tidak akan bisa diubah kembali.

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.

2.3. 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 di bawah ini



Gambar 2.5 Perilaku semut pada dunia nyata

Ant colony system (koloni semut), algoritma yang digunakan untuk menyelesaikan permasalahan pada tugas akhir ini, merupakan algoritma yang berdasarkan algorima *ant system* dengan meningkatkan efisiensi pencarian rute yang dilalui. Konsep dasar dari algoritma koloni semut adalah dengan menggunakan sekumpulan semut yang bertujuan mencari lintasan terpendek secara pararel.

2.4. JAVA 2 MICRO EDITION (J2ME)

J2ME adalah satu set spesifikasi dan teknologi yang fokus kepada perangkat konsumen.

Program-program J2ME, seperti semua program Java, dikompile ke dalam *bytecode* dan diterjemahkan dengan Java Virtual Machine (JVM).

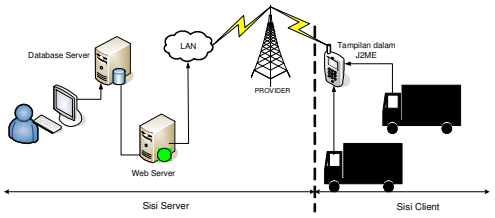
Inti dari J2ME terletak pada konfigurasi dan profil-profil. Suatu konfigurasi menggambarkan lingkungan *runtime* dasar dari suatu sistem J2ME. Ia menggambarkan *core library, virtual machine,* fitur keamanan dan jaringan.

3. Perancangan 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.

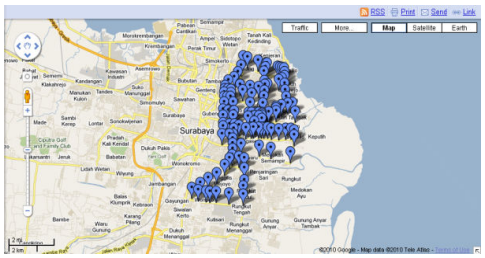
Tidak hanya itu, rute terpendek yang dihasilkan ACO akan dibandingkan dengan hasil perhitungan menggunakan algoritma *Dijkstra*. Sedangkan di sisi *client*, pengemudi dapat mengakses informasi berupa rute terpendek dari *database* tersebut melalui ponsel berbasis J2ME dengan cara memasukkan *username*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



Gambar 3.1 Blok Diagram Sistem

3.2. Data Peta Surabaya Timur

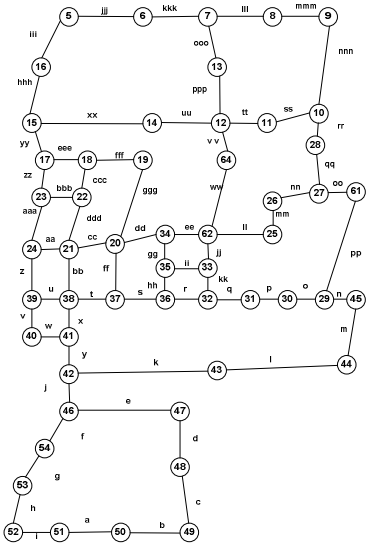
Wilayah Surabaya Timur dibuat beberapa *node* yang digunakan untuk keperluan pemodelan jaringan jalan. Pemetaan *node* ini memanfaatkan *Google Maps* secara online yaitu dengan cara menandai setiap persimpangan dan tempat-tempat yang *familiar* dikunjungi oleh masyarakat (*point*), misalnya: pasar, rumah sakit, SPBU, dll. Selain itu, dicari pula jarak yang menghubungkan 2 persimpangan tersebut.



Gambar 3.2 Peta Node Surabaya Timur di Google Maps

3.3. Pemodelan Jaringan Jalan

Jalan yang terdapat dalam peta sebenarnya 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, kecuali jika jarak jalan tersebut sangatlah panjang, maka jalan akan dibagi menjadi beberapa ruas (syarat dan ketentuan berlaku).

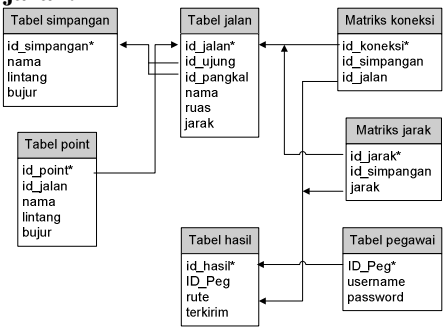


Gambar 3.3 Representasi *Graph*

3.4. Relasi dengan Database

Dalam sistem yang dirancang terdapat 7 tabel yang berkaitan dengan pemodelan jaringan jalan yang digunakan. 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* **id_koneksi** yang membentuk matriks yang berisi **id_jalan.**
7. Tabel *JARAK*, berisi *field* **id_jarak** yang membentuk matriks yang berisi **jarak.**



Gambar 3.4. Relasi Tabel pada *Database*

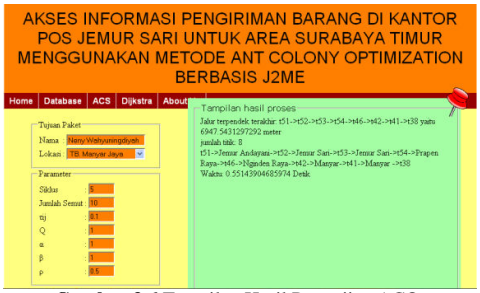
3.5. Interaksi Client-Server

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).

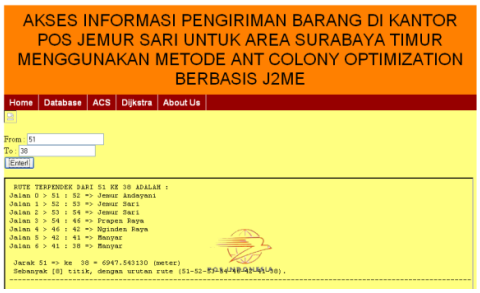
Gambar 3.5 Tampilan pada *Handphone Client*

Untuk *Admin*, *interface* yang digunakan untuk mengakses *server* berupa

website yang akan mengolah input untuk perhitungan algoritma ACO dan Dijkstra



Gambar 3.6 Tampilan Hasil Pengujian ACO



Gambar 3.7 Tampilan Hasil Pengujian Dijkstra

4. Pembuatan Sistem, Implementasi, 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 α >= 0.
6. β (beta) adalah tetapan pengendali visibilitas, dimana β >= 0 .
7. ρ (rho) adalah tetapan penguapan jejak semut untuk mencegah jejak semut yang tak terhingga, dimana 0 < ρ < 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:

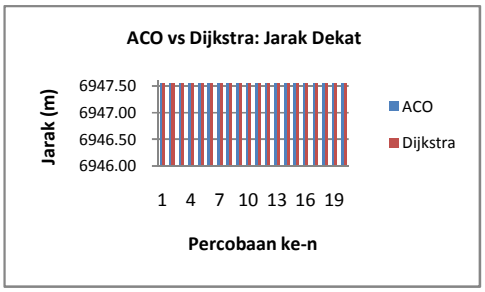
1. Jarak Dekat : dari 51 ke 38 (point: TB. Manyar Jaya)
2. Jarak Menengah : dari 51 ke 62 (point: Natasha Skin Care)
3. Jarak Jauh : dari 51 ke 7 (point: Hotel Puspa Asri)

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 -> Manyar.

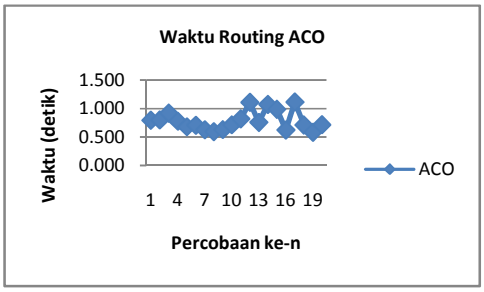
Penulis menggunakan parameter terbaik yang diperoleh dari penelitian sebelumnya [4], yaitu:

- Siklus = 5
- Jumlah semut = 10
- τ_{ij} = 0,01
- Q = 1
- α = 1
- β = 1
- ρ = 0,5

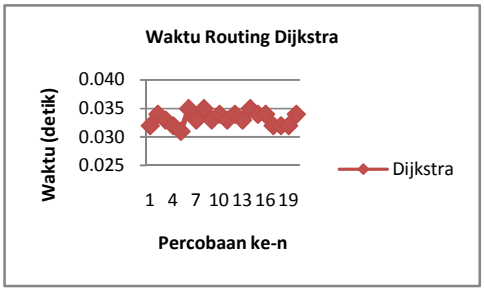
Jarak terpendek yang dihasilkan oleh ACO dan Dijkstra adalah sama, sekitar 6947,54 meter. ACO memerlukan waktu antara 0,5 sampai 1,2 detik untuk menghasilkan jarak terpendek sedangkan Dijkstra yang hanya membutuhkan waktu berkisar antara 0,03 dan 0,035 detik. Rata-rata waktu untuk ACO adalah 0,787 detik dan 0,033 detik untuk Dijkstra. Bila dinyatakan dalam grafik, hasilnya adalah sebagai berikut:



Gambar 4.2 Grafik Jarak Terpendek untuk Jarak Dekat



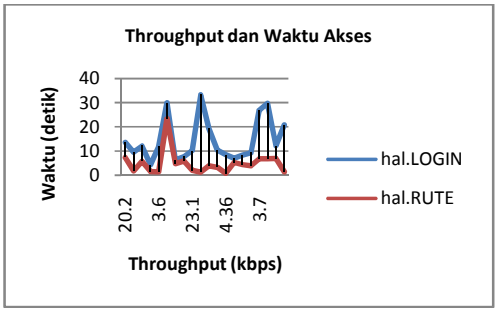
Gambar 4.3 Grafik Waktu Tempuh ACO untuk Jarak Dekat



Gambar 4.3 Grafik Waktu Tempuh Dijkstra untuk Jarak Dekat

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 4.4 Grafik *Throughput* dan Waktu Akses

Gambar 4.14 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.

6. Kesimpulan

a. 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).
b. Perbandingan algoritma koloni semut dengan Dijkstra menghasilkan jarak terpendek yang sama baik untuk rute jarak dekat, jarak menengah, maupun jarak jauh.
c. 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.
d. 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.

7. Daftar Pustaka :

[1] Fredivianus,Nugroho. "Penggunaan Model Multiple Vehicle Routing Problem dalam Optimasi Jaringan Distribusi Part di PT. Astra Internasional Tbk.

Toyota Sales Operation AUTO2000 Surabaya". T.Elektro ITS Surabaya. 2005

[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] Mulyani, Titik Sri. "Akses Informasi Pengiriman Barang di Kantor Pos Jemursari untuk Area Surabaya Timur Menggunakan Metode Ant Colony Optimization Berbasis WAP". T. Telekomunikasi PENS-ITS. Surabaya. 2010.