Implementasi Ant Colony Optimization untuk Routing pada Optimasi Perutean in Network Processing pada Sistem Monitoring Kesehatan Struktur Jembatan

1st M Firmansyah Arrozi
Fakultas Informatika
Universitas Telkom
Bandung, Indonesia
firmanarrozi@students.telkomuniversity.ac.id

2nd Setyorini
Fakultas Informatika
Universitas Telkom
Bandung, Indonesia
setyorini@telkomuniversity.ac.id

3rd Seno Adi Putra
Fakultas Informatika
Universitas Telkom
Bandung, Indonesia
adiputra@telkomuniversity.ac.id

Abstrak*— *Structral Health Monitoring System* pada umumnya diterapkan pada jembatan untuk memperpanjang usia bangunan tersebut dengan mengidentifikasi kerusakan yang terjadi pada bangunan lebih awal. *Structural Health Monitoring System* berbasis *wireless sensor network* pada jembatan lebih diunggulkan karena lebih murah dari segi biaya yang dikeluarkan. Akan tetapi karakteristik sumber daya yang dimiliki oleh *wireless sensor network* itu terbatas. Sehingga diperlukan efesiensi dalam konsumsi energi pada *wireless sensor network*. Salah satunya adalah dengan menggunakan perutean yang optimal untuk mengirimkan data dari sensor node menuju ke sink node. Hal tersebut dimaksudkan untuk memiminimalkan konsusi energi pada *wireless sensor network*. Dikarenakan banyaknya sensor node yang harus diproses dan pengolahan datanya dilakukan pada setiap sensor node. Algoritma ant colony optimization merupakan salah satu dari beberapa algoritma optimasi yang bisa dipilih untuk melakukan perutean secara optimal. Hasil dari implementasi algoritma ACO dilakukan pada beberapa skenario pengujian dan dibandingkan dengan algoritma genetika. Hasil pengujian yang diperoleh menunjukkan bahwa algoritm ACO dan GA memiliki hasil yang hampir sama. Dan didapatkan performa algoritma ACO yang telah diimplementasikan untuk WSN pada *Structural Health Monitoring System.

***Kata Kunci*— optimasi perutean, *wireless sensor network*, *ant colony optimization*, metaheuristik, *structural health monitoring system*.**

I. PENDAHULUAN

A. Latar Belakang

Structural Health Monitoring merupakan salah satu bidang keilmuan yang memiliki tujuan untuk mendeteksi kerusakan yang terjadi pada struktur bangunan untuk memonitor kesehatan dari suatu bangunan. *Structural Health Monitoring System* umumnya digunakan untuk memperpanjang umur dari suatu bangunan karena kemampuannya untuk mengidentifikasi kerusakan dan perubahan yang terjadi pada suatu bangunan lebih awal, sebelum terjadi kerusakan yang lebih parah dan mungkin membutuhkan biaya yang lebih besar untuk memperbaikinya.

Salah satu bangunan yang sering menggunakan *Structural Health Monitoring System* adalah jembatan. Untuk mengurangi biaya yang dikeluarkan dalam penerapan *Structural Health Monitoring System* pada jembatan, maka digunakan SHMS berbasis *wireless sensor network*. *Wireless sensor network* dipilih dikarenakan minimnya biaya yang dibutuhkan dalam penerapannya. Dan keunggulan dari penerapan aplikasi berbasis *wireless sensor network* dapat memberikan hasil yang sangat optimal dikarenakan kemampuannya yang dapat diterapkan pada banyak hal[2].

Di dalam *wireless sensor network* terdapat *in-network processing* yakni pemrosesan data yang terjadi pada setiap *node*-nya. Pengaplikasian *wireless sensor network* berdasar pada node-node kecil yang memiliki tugas untuk mengumpulkan data yang diterima. Yang kemudian data tersebut dikirimkan menuju sink node [2]. Salah satu yang menjadi pertimbangan ketika menggunakan *wireless sensor network* adalah karakteristik dari sumber dayanya yang terbatas. Dikarenakan banyaknya node yang harus diproses dan pengolahan data yang dilakukan pada setiap *sensor nodes*-nya. Sehingga efesiensi dari konsumsi sumber daya yang tersedia perlu dilakukan. Salah satu aspek yang dapat meminimalisir konsumsi sumber daya pada WSN adalah mengoptimalkan perutean *in-network processing*. Perutean ini diperlukan untuk mengurangi konsumsi sumber daya yang terbatas. Sehingga sumber daya dari sensor node yang ada pada WSN bisa bertahan lebih lama.

Salah satu cara untuk melakukan perutean adalah dengan menggunakan metode metaheuristik. Di dalam metode metaheuristik terdapat banyak sekali algoritma *routing* yang bisa digunakan. Diantaranya *Simulated Annealing* (SA), *Genetic Algorithm* (GA), *Cross Entropy* (CE), *Particle Swarm Optimization* (PSO), dll. Algoritma-algoritma tersebut dapat digunakan untuk menentukan rute paling optimal ketika melakukan pengiriman data. Sehingga konsumsi sumber daya pada WSN bisa diminimalisir.

Penerapan algoritma *routing* membuat pengiriman data dari sensor dilakukan secara *hop-by-hop.* Ant colony optimization merupakan salah satu dari sekian banyak algoritma yang bisa digunakan untuk melakukan *routing*. Pada penelitian ini algoritma ACO dipilih dikarenakan pada penelitian-penelitian sebelumnya belum ditemukan penerapan ACO pada optimasi *routing in network processing* pada sistem *monitoring* kesehatan struktur jembatan. Alasan lain mengapa dipilih algoritma ACO adalah dikarenakan pada beberapa paper ditemukan bahwa algoritma ACO memiliki performa yang lebih baik daripada algoritma yang lebih modern dalam kondisi-kondisi tertentu. Salah satunya algoritma ACO mampu mencari solusi berupa jarak tempuh yang lebih pendek daripada algoritma genetika untuk data kurang dari 20 kota[18]. Penerapan algoritma ACO pada penelitian ini untuk menentukan rute paling optimal pada *in-network processing* di dalam Sistem *Monitoring* Kesehatan Struktur Jembatan.

1. Topik dan Batasannya

a. Rumusan Masalah pada Tugas Akhir ini yaitu:

Bagaimana mengimplementasikan algoritma ant colony optimization untuk optimasi perutean in network processing dalam sistem *monitoring* kesehatan struktur jembatan?

Bagaimana hasil analisis performansi dari implementasi algoritma ant colony optimization untuk optimasi perutean *in-network processing* dalam sistem *monitoring* kesehatan struktur jembatan?

b. Batasan Masalah pada Tugas Akhir ini yaitu:

Pengujian algoritma ant colony optimization yang dilakukan terbatas pada emulator sunspot manager application dan pada sistem *monitoring* kesehatan struktur jembatan.

Pengujian algoritma yang dilakukan hanya terbatas pada parameter waktu yang dibutuhkan dalam optimasi perutean pada sistem *monitoring* kesehatan struktur jembatan. Dengan asumsi bahwa semakin cepat waktu yang dibutuhkan untuk melakukan proses routing, maka semakin sedikit konsumsi sumber daya yang terdapat pada WSN.

Hasil dari pengujian algoritma ant colony optimization dikomparasikan dengan performansi algoritma genetika untuk optimasi perutean pada sistem *monitoring* kesehatan struktur jembatan.

2. Tujuan

Tujuan yang ingin dicapai pada Tugas Akhir ini yaitu:

a. Mengimplementasikan algoritma ant colony optimization dalam optimasi perutean in-network processing untuk sistem monitoring kesehatan struktur jembatan.

b. Menganalisis hasil pengujian dari implementasi algoritma ant colony optimization berdasarkan parameter waktu yang dibutuhkan pada optimasi perutean in-network processing pada sistem monitoring kesehatan struktur jembatan.

3. Organisasi Tulisan

Pada bab berikutnya adalah bab 2 yang berisikan penjelasan terkait studi literatur yang mendukung pengerjaan penelitian ini. Pada bab 3 dijelaskan sistem yang dibangun pada penelitian ini. Bab 4 berisikan hasil dari uji coba, evaluasi, dan analisis hasil dari uji coba yang telah dilakukan. Dan untuk bab 5, berisikan penjelasan terkait kesimpulan dari keseluruhan proses penelitian yang dilakukan beserta dengan saran untuk penelitian selanjutnya.

II. KAJIAN TEORI

A. Studi Terkait

Structural Health Monitoring adalah bidang keilmuan yang melibatkan pemantauan otomatis terhadap beban struktural dan respon yang diterimanya melalui sejumlah besar sensor dan instrumen, sekaligus melakukan diagnosis kesehatan struktural berdasarkan data yang telah dikumpulkan [14]. Menurut studi dari (Y. Bao et. al, 2019) secara umum SHM terdiri dari berbagai sensor, perangkat akuisisi data, sistem transmisi data, *database* untuk manajemen data, analisis data dan pemodelan data, penilaian kondisi, prediksi kinerja, *alarm devices*, UI visualisasi, *software*, dan sistem operasi. SHMS telah banyak diimplementasikan secara luas pada bidang kedirgantaraan, teknik sipil, dan teknik mesin.

WSN biasanya terdiri dari banyak node yang relatif kecil, dan masing-masing node tersebut dilengkapi kemampuan untuk menangkap data. Di setiap nodenya terdapat baterai dan sebagian besar dari jaringan sensor node tersebut menggunakan komunikasi *wireless*. Keterbatasan dalam sumber daya, kemampuan komunikasi, dan konsumsi daya yang terpakai menuntut efesiensi dalam hal pemakaiannya [1].

Perbandingan yang dilakukan antara ACO dan PSO untuk menemukan jarak terpendek memberikan hasil bahwa algoritma ACO memberikan kinerja yang lebih baik dibandingkan dengan PSO untuk masalah optimasi jarak. Karena pada penerapannya algoritma ACO mencapai *total minimum* dalam jumlah iterasi yang lebih sedikit dan nilai yang lebih rendah jika dibandingkan dengan PSO [16].

Dalam penerapan ACO pada TSP, jika terdapat n sensor akan ada (nx (n - 1) / 2) buah ruas, dan juga memiliki (n - 1)! /2 rute yang mungkin. Jarak yang digunakan pada TSP standar merupakan jarak simetris, jadi jarak antara sensor A dan sensor B sama dengan jarak sensor B dan sensor A artinya d (A, B) = d (B, A).

Algoritma ACO pada TSP akan menggunakan pointer berupa *ant* yang memulai rutenya pada sebuah sensor dan akan mencari rute ke sensor terdekat, dan seterusnya sampai sensor terakhir dengan kondisi semua sensor sudah dilewati oleh *ant*. Pemilihan sensor-sensor yang dilalui oleh *ant* didasarkan pada fungsi probabilitas, dengan mempertimbangkan *visibiliy* sensor tersebut dan jumlah *pheromone* yang terdapat pada ruas yang menghubungkan antara sensor A dan B. Probabilitas bergerak dari sensor A ke B akan membesar jika jarak antara sensor A dan B lebih kecil dari jarak sensor A ke sensor lainnya.

Untuk mencegah *ant* mengunjungi sebuah sensor lebih dari sekali selama iterasi berlangsung bisa menggunakan *tabu list*. *Tabu list* di sini berisi data sensor-sensor yang telah dilalui oleh *ant.* Setelah selesai satu iterasi maka semua *pheromone* yang terdapat pada ruas-ruas antar sensor akan menguap. Dan untuk *ant* dihitung panjang seluruh rute yang telah ditempuh. Tujuan penguapan dari *pheromone* sendiri adalah untuk mencegah iterasi yang dilakukan berikutnya melalui rute yang sama secara terus menerus. Setelah rute paling optimal dari *ant* ditemukan, rute tersebut disimpan dan *tabu list* dikosongkan kembali. Proses tersebut diulang

kembali sampai pada jumlah maksimum iterasi yang telah ditentukan. Aturan untuk transisi status yang digunakan oleh *ant* disebut sebagai *random-proportional rule*, yang ditunjukkan oleh persamaan (1)[17],

$$p_k(A,B) = \begin{cases} \tau(A,B)\eta(A,B)^{\beta} / \sum_{u \notin Mk} \tau(A,u)\eta(A,u)^{\beta}, \text{jika } s \notin M^k \\ 0, \qquad \text{lainnya} \end{cases} \quad (1)$$

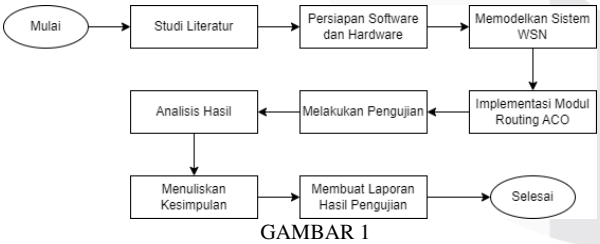
pk (A, B) adalah peluang perpindahan dari sensor A ke B dari *ant k,* τ merupakan tingkat dari *pheromone,* η(A, B) adalah bobot yang mengontrol visibility terhadap tingkat *pheromone* τ. Langkah pertama perlu dilakukan perhitungan matrik yang menyatakan kedekatan antar sensor, η(A, B), dimana setiap entrinya merupakan 1/jarak dari A ke B. Dari persamaan di atas ruas yang lebih pendek dan memiliki jumlah *pheromone* yang lebih besar akan memiliki probabilitas lebih besar untuk dipilih dalam rute [17].

Menurut studi (B. Santosa, et. Al., 2011) pembaruan *pheromone* pada ACO untuk TSP terjadi secara lokal dan global. Pembaruan secara global dimaksudkan untuk membuat jalur tersebut menjadi ruas terbaik, dan pembaruan pheromone secara lokal dimaksudkan untuk mencegah semua *ant* melalui rute yang sama berkali-kali. Sehingga secara tidak langsung hal tersebut membuat *ant* lainnya memilih rute lain yang belum dilalui. Hal tersebut dilakukan untuk membuat kemungkinan rute yang berbeda-beda. Karena ketika *ant* melalui rute yang berbeda-beda kemungkinan mendapatkan rute paling optimal akan lebih tinggi daripada jika setiap *ant* melewati rute yang sama. Karena ketika setiap *ant* melewati rute yang sama tidak ada pembanding untuk mendapatkan rute paling optimal diantara rute-rute lain yang telah dilaluinya.

III. METODE

A. Sistem yang Dibangun

1. Metode yang digunakan.



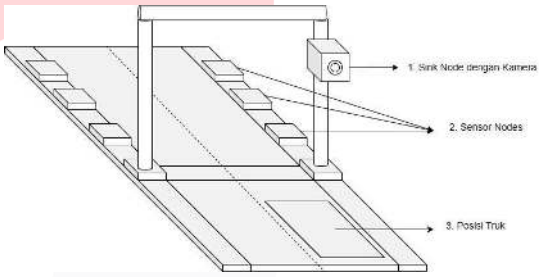
GAMBAR 1
Metodologi yang Digunakan

Metodologi yang digunakan pada penulisan Tugas Akhir ini dapat dilihat pada **Gambar 1**. Proses ke- 1 yang dilakukan adalah studi literatur. Dalam proses ini dilakukan pengumpulan paper dan jurnal serta referensi yang memiliki keterkaitan dengan pengerjaan Tugas Akhir ini. Proses ke-2 dan ke-3 adalah persiapan *hardware*, *software*, dan memodelkan sistem WSN. Pada proses ini *hardware* dan *software* yang dibutuhkan untuk Tugas Akhir dipersiapkan sekaligus melakukan pemodelan sistem *wireless sensor network*. Pemodelan wsn ini dibutuhkan untuk terbentuknya lingkungan *wireless sensor network* yang berfungsi sebagai tempat implementasi program yang telah dibuat.

Proses ke-4 adalah implementasi modul routing ACO. Pada proses ini program algoritma *routing* ant colony optimization diimplementasikan ke dalam lingkungan *wireless sensor network*. Proses ke-5 dan ke-6 adalah mengumpulkan dataset dan melakukan pengujian. Pada proses ini dilakukan pengujian untuk mendapatkan dataset yang dikumpulkan oleh sensor dan hasil kinerja *routing* dari algoritma ant colony optimization.

Proses ke-7 dan ke-8 adalah membuat analisis hasil dan menuliskan kesimpulan. Hasil dari performansi *routing* algoritma ACO yang sudah diuji dianalisis dan kemudian dari hasil tersebut diambil kesimpulan untuk Tugas Akhir ini. Kesimpulan yang dibuat juga harus didasarkan pada paper-paper yang terdahulu yang menjadi acuan. Proses ke-9 adalah membuat laporan hasil pengujian. Pada proses ini dilakukan penulisan laporan hasil dari pengujian secara lengkap dan runtut dari proses awal sampai akhir untuk syarat diselesaikannya Tugas Akhir ini.

2. Arsitektur Jembatan



GAMBAR 2
Arsitektur Jembatan

Arsitektur pengujian pada penelitian ini diperlihatkan seperti pada Gambar 2. Posisi yang ditunjukkan oleh nomor 1 adalah lokasi dari sink node dan kamera. Kamera berfungsi sebagai pendeteksi ketika truk mendekati jembatan. Dan sink node berfungsi untuk menghidupkan setiap sensor node dan juga sebagai penerima data dari setiap sensor node. Posisi yang ditunjukkan angka 2 adalah lokasi dari sensor node. Sensor node memiliki tugas untuk melakukan *sensing* data ketika truk melewati jembatan. Data yang ditangkap kemudian dikirimkan menuju sink node secara *hop-by-hop*. Algoritma ant colony optimization diimplementasikan untuk optimasi *routing* yang berjalan pada sensor node. Posisi yang ditunjukkan oleh nomor 3 adalah lokasi dari truk yang akan melewati jembatan dan bisa terdeteksi oleh kamera.

3. Rancangan Implementasi Routing dengan ACO

Algoritma ACO yang diterapkan pada penelitian ini menerapkan model *Travelling Salesman Problem* (TSP). Model TSP yang diterapkan pada WSN dimaksudkan untuk membuat informasi yang dikirimkan ke sink node melewati setiap node yang terdapat di dalam jaringan. Hal ini bertujuan untuk mencegah terjadinya tabrakan informasi pada sink node dikarenakan sink node harus menerima data *real-time* secara terus menerus dari setiap node. Karena apabila informasi tidak dikirimkan secara hop-by-hop maka akan terjadi kemungkinan beberapa node mengirimkan datanya secara bersamaan. Pengiriman informasi dari beberapa node secara bersamaan menyebabkan kemungkinan sink node tidak dapat menangkap informasi tersebut secara bersamaan.

Oleh karena itu mode TSP diterapkan untuk menghindari kemungkinan adanya informasi yang tidak tertangkap oleh sink node ketika menerima data secara bersamaan dari beberapa node. Sebagian besar pemrosesan data terjadi di dalam setiap node. Hal ini juga diterapkan karena mempertimbangkan performa dari sink node yang digunakan pada jaringan dalam penelitian ini.

Algoritma ACO yang diimplementasikan ke dalam sistem ditunjukkan seperti pada Algoritma 1**.** Dari algoritma tersebut dibuat sebuah program berbasis bahasa java. Kemudian dari program tersebut dilakukan pengujian terlebih dahulu untuk memastikan bahwa program telah dibuat sesuai dengan kebutuhan yang terdapat pada penelitian yang dilakukan. Pembuatan program algoritma ACO ini menggunakaan Java Netbeans. Hal ini dimaksudkan untuk mempermudah pengujian yang dilakukan pada sensor Sun Microsystem.

Algoritma 1: Algoritma Ant Colony Optimization

```
Input : jumlah ant, matriks jarak, itmax
Output : rute terbaik
%tahap insialisasi
Foreach i=1; jumlah semut do
    τ(i,j) = τ0
    Hitung nilai visibility h(i,j) = 1/jarak(i,j)
end
While it < itmax do
    Tempatkan semua semut pada kota 1
    Foreach i=1; jumlah semut do
        Mulailah dari kota r
        Hitung tingkat pheromone untuk setiap ruas yang terhubung dengan kota r
        Hitung probabilitas transisi dari kota r ke kota lain
        Bangkitan bilangan random
        Gunakan roulette wheel selection untuk memilih kota berikutnya
    end
    Foreach i=1: jumlah semut do
        Hitung jarak rute untuk setiap semut
        Update nilai τ untuk iterasi berikutnya
    end
end
```

ALGORITMA 1
Ant Colony Optimization Algorithm

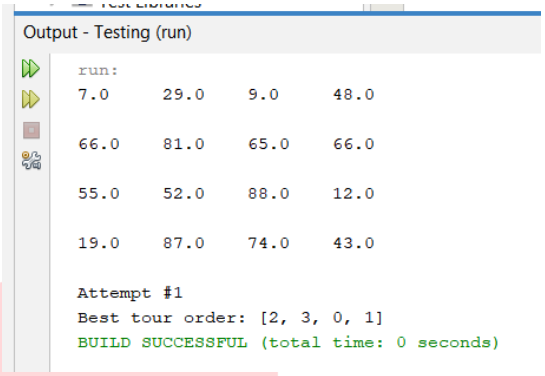
IV. HASIL DAN PEMBAHASAN

A. Evaluasi

1. Hasil dan Analisis Hasil Pengujian

Pengujian pertama yang dilakukan adalah menguji algoritma ACO menggunakan Java Netbeans. Hal ini dilakukan untuk memastikan algoritma ACO yang sudah dibuat berjalan sesuai dengan konsep *Travelling Salesman Problem*. Pengujian menggunakan Java Netbeans dipilih karena bahasa yang digunakan sesuai dengan bahasa yang digunakan pada sensor sunspot microsystem.

Pengujian pertama dilakukan untuk memastikan konsep TSP sudah diterapkan ke dalam algoritma ACO yang disusun. Skenario pengujian pertama dilakukan dengan menggunakan 4 titik sebagai tujuan dari *ant,* yaitu titik 0, 1, 2, dan 3. Untuk jarak antar tiap titik tersebut diset sebagai angka random. Dan rute paling optimal dihitung berdasarkan total jarak tempuh paling minimum dari rute yang telah dilalui oleh *ant.* Hasil yang didapatkan pada pengujian tersebut seperti pada Gambar 4**,**



GAMBAR 4
Percobaan ACO pada Java Netbeans

Attempt merepresentasikan urutan percobaan yang dilakukan. *Best tour order* merupakan rute terbaik yang dilalui oleh *ant.* Pada pengujian ini rute terbaik ditempuh secara berurutan dari titik 2, 3, 0, dan 1. Titik pertama dan terakhir dari pengujian ini ditentukan berdasarkan jarak paling minimum yang ditempuh oleh masing-masing *ant* untuk mencapai semua titik. Dari pengujian tersebut didapatkan hasil bahwa algoritma ACO yang disusun sudah sesuai dengan konsep TSP karena melalui seluruh titik yang sudah di-*set* di dalam program. Untuk potongan *source code* dari ACO yang digunakan pada pengujian bisa dilihat pada lampiran 4.

Untuk memastikan algoritma ACO sudah sesuai dengan kebutuhan untuk penelitian yang dilakukan. Maka dilakukan percobaan perhitungan secara manual untuk menentukan rute paling optimal dengan melihat jarak terpendek dari setiap kemungkinan rute yang ada. Pengujian ini dilakukan dengan cara membandingkan hasil pencarian rute paling optimal secara manual dengan program algoritma ACO yang telah dibuat pada java Netbeans. Langkah awal diberikan tabel seperti pada Tabel 1 yang didapatkan dari matriks pada Gambar 4,

TABEL 1
Matriks Jarak pada Ant

	0	**1**	**2**	**3**
0	0	29	9	48
1	66	0	65	66
2	55	52	0	12
3	19	87	74	0

Pada tabel tersebut terdapat kota 0,1,2, dan 3 yang merepresentasikan setiap node yang akan dilalui oleh *ant* untuk melakukan perutean. Untuk masing-masing node terdapat jarak ke setiap node lainnya. Untuk jarak dari satu kota ke kota tersebut diasumsikan sebagai 0 dikarenakan tidak terdapat jarak yang ditempuh untuk menuju kota yang sama. Misal dari kota 1 ke kota 1 jaraknya adalah 0. Dan jarak dari kota 1 ke kota 2 adalah 65 dan seterusnya. Dari tabel tersebut akan dicari setiap kemungkinan rute yang ada untuk menentukan rute paling optimal dan dibandingkan hasilnya

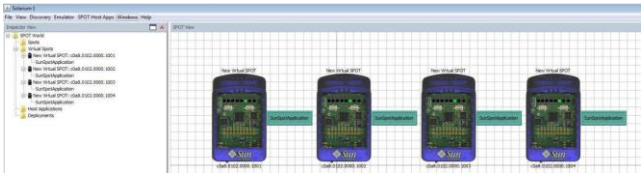
dengan hasil dari program algoritma ACO yang telah dibuat. Dan kemungkinan rute yang dilalui untuk melewati maka didapatkan seperti pada tabel 2,

TABEL 1
Hasil Pencarian Rute secara Manual

No	Rute	Total Jarak yang Ditempuh
1	0-1-2-3	106
2	0-1-3-2	169
3	0-3-2-1	174
4	0-3-1-2	200
5	0-2-1-3	127
6	0-2-3-1	108
7	1-0-2-3	87
8	1-0-3-2	188
9	1-2-0-3	168
10	1-2-3-0	96
11	1-3-0-2	94
12	1-3-2-0	195
13	2-0-1-3	150
14	2-0-3-1	190
15	2-1-0-3	166
16	2-1-3-0	137
17	2-3-0-1	60
18	2-3-1-0	165
19	3-0-1-2	113
20	3-0-2-1	80
21	3-1-0-2	162
22	3-1-2-0	207
23	3-2-0-1	158
24	3-2-1-0	192

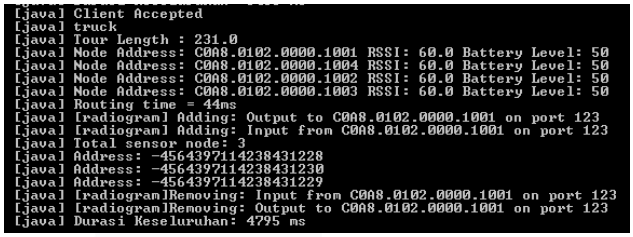
Dari tabel tersebut diperlihatkan semua kemungkinan rute yang bisa dilalui dan total jarak tempuh yang harus dilalui ketika melewati sebuah rute. Total jarak yang ditempuh didapatkan dari perhitungan total dari rute yang dilalui berdasarkan Tabel 1. Dari percobaan tersebut didapatkan bahwa rute dengan jarak terpendek yang bisa dilalui adalah pada rute 2-3-0-1 dengan jarak tempuh 60. Hasil percobaan tersebut sesuai dengan pengujian yang dilakukan menggunakan algoritma ACO yang ditunjukkan seperti pada Gambar 4. Dari hasil tersebut juga menunjukkan bahwa rute 2-3-0-1 adalah rute paling optimal. Sehingga dapat diambil kesimpulan bahwa program algoritma ACO yang telah dibuat sudah sesuai dengan kebutuhan penelitian yang dilakukan.

Pengujian kedua yang dilakukan adalah menguji algoritma ACO menggunakan software Sunspot Manager Application. Software tersebut adalah emulator yang dapat mensimulasikan program dan cara kerja dari sensor sunspot microsystem. Pengujian ini dilakukan untuk memastikan program yang sudah dibuat dapat berjalan dengan baik ketika disimulasikan pada emulator Sunspot Manager Application. Jika sudah dapat berjalan maka program bisa di-*deploy* ke dalam sensor sunspot microsystem.



GAMBAR 5
Percobaan ACO pada emulator Sunspot Manager Application

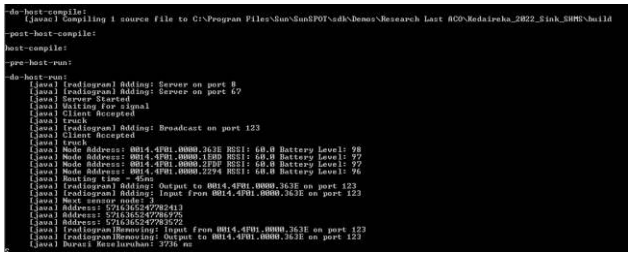
Pada pengujian kedua ini dilakukan percobaan selama 5 kali untuk mendapatkan hasil yang objektif. Pada pengujian ini menggunakan program tambahan untuk memancarkan sinyal bahwa kendaraan akan segera melewati jembatan yang sudah terpasang sensor sunspot microsystem. Hal tersebut dilakukan untuk mensimulasikan kondisi *real* yang terjadi di lapangan. Batasan pada pengujian ini adalah parameter input yang digunakan bukan berasal dari RSSI dan *battery level* melainkan dari jarak random yang sudah ditentukan di dalam algoritma ACO. Dikarenakan pada emulator Sunspot Manager Application tidak dapat mengambil data RSSI dan *battery level* dari masing-masing sensor. Dan untuk penamaan dari titik-titik yang menjadi tujuan ditentukan berdasarkan penamaan yang terdapat pada masing-masing sensor yang ditunjukkan oleh *Node Address*. Untuk rute paling optimal yang dipilih diambil berdasarkan rute paling minimum dari total keseluruhan rute yang ditempuh oleh *ant.* Pengujian yang dilakukan dapat dilihat pada gambar. Untuk dokumentasi secara keseluruhan dari total 5 kali pengujian ACO pada emulator bisa dilihat pada lampiran 1.



GAMBAR 6
Hasil Percobaan pada Emulator Sunspot Manager Application

Server started dan *Waiting for signal* menandakan sink node sudah aktif dan siap untuk menangkap sinyal. *Client Accepted* dan *truck* mensimulasikan bahwa terdapat objek yang mendekat ke arah sensor. *Broadcast on port 123* memberikan perintah kepada sensor untuk bersiap-siap melakukan sensing. Dan *node address* menunjukkan alamat dari masing-masing sensor. RSSI dan battery level di-*set* dengan nilai sama. Node address yang ditampilkan sudah berurutan berdasarkan pada rute paling optimal yang dilewati oleh *ant.* Dan berikutnya terdapat *Routing time* yaitu waktu yang dibutuhkan untuk melakukan proses routing menggunakan ACO. Pada pengujian kedua ini didapatkan hasil bahwa algoritma ACO yang dibuat dapat berjalan pada emulator sunpot manager application dan sudah berjalan dengan menerapkan konsep TSP.

Pengujian ketiga dilakukan pada sensor sunspot microsystem. Percobaan ini dilakukan sebanyak dua kali, berdasarkan hasil percobaan tersebut didapatkan bahwa routing time yang dibutuhkan oleh sensor node untuk mendapatkan rute optimal menggunakan ACO adalah 45 ms. Dan untuk Durasi keseluruhan waktu yang dibutuhkan adalah 3736 ms. Percobaan kedua yang dilakukan pada sensor menghasilkan durasi routing time yang dilakukan oleh algoritma ACO sebesar 42 ms. Dan untuk total durasi secara keseluruhan sebesar 3255 ms. Hasil yang didapatkan ditunjukkan seperti pada Gambar 9,



GAMBAR 7
Pengujian pada Sensor Node

Pada pengujian pertama yang dilakukan pada Java netbeans didapatkan hasil bahwa algoritma ACO yang dibuat sudah berjalan sesuai dengan konsep TSP. Hal ini terlihat dari *ant* melewati seluruh titik yang menjadi tujuan. Pada pengujian berikutnya untuk ACO yang diterapkan pada emulator Sun Microsystem didapatkan hasil seperti Tabel 3,

TABEL 3
Pengujian ACO pada Emulator Sun Microsystem

	Percobaan 1	Percobaan 2	Percobaan 3	Percobaan 4	Percobaan 5	Rata-R
Routing Time (ms)	44	65	52	51	51	52.6

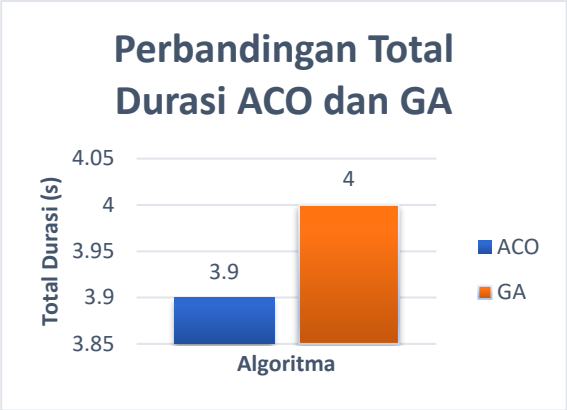
Pada Tabel 3 diberikan data terkait hasil pengujian dari 5 kali percobaan, pada hasil yang didapatkan terlihat waktu yang dihasilkan dari masing-masing percobaan memiliki perbedaan yang tidak terlalu signifikan. Dan dari hasil percobaan tersebut didapatkan bahwa untuk melakukan sekali *routing* menggunakan algoritma ACO diperlukan rata-rata waktu sebesar 52.6 ms.

Untuk analisis hasil pengujian algoritma ACO yang sudah diterapkan dibandingkan dengan performansi Algoritma Genetika seperti yang sudah dibahas pada batasan masalah pada penelitian ini. Hasil yang didapatkan dari ACO meskipun tidak terlalu optimal namun hasilnya mendekati optimal[7]. Percobaan ini bertujuan untuk menguji performansi dari ACO. Apabila total durasi yang dihasilkan oleh ACO sangat berbanding jauh dengan GA, maka bisa disimpulkan bahwa terdapat kesalahan di dalam pemilihan rute paling optimalnya. Namun apabila hasil yang didapatkan tidak terlalu jauh perbedaannya, maka ACO yang diterapkan sudah sesuai untuk pencarian rute yang paling optimal. Pada pengujian ini diasumsikan bahwa performansi *device* yang digunakan stabil.

Perbandingan yang dilakukan didasarkan pada pengujian yang dilakukan pada emulator sensor sunpot microsystem. Kedua algoritma tersebut masing-masing di-*run* selama 5 kali untuk mendapatkan hasil yang objektif dari sisi optimasi peruteannya. Setelah dilakukan perbandingan, hasil yang didapatkan seperti pada Tabel 4,

TABEL 4
Hasil Perbandingan Algoritma ACO dan GA

Algoritma	Percobaan 1 (ms)	Percobaan 2 (ms)	Percobaan 3 (ms)	Percobaan 4 (ms)	Percobaan 5 (ms)	Rata-rata (ms)
ACO	3880	3719	3989	3455	4795	3967.6



GAMBAR 8
Perbandingan Total Durasi ACO dan GA

Pada Gambar 8 dapat dilihat untuk percobaan yang dilakukan selama 5 kali menghasilkan perbandingan durasi yang tidak terlalu jauh pada total durasi yang dibutuhkan oleh masing-masing algoritma untuk melakukan proses *routing* sampai data tersebut dikirimkan menuju sink node. Sehingga dapat ditarik kesimpulan bahwa implementasi algoritma ACO sudah berhasil dilakukan pada emulator sunspot manager application. Dan terdapat batasan pada percobaan ini. Untuk total durasi yang didapatkan bisa berbeda-beda dikarenakan faktor performa dari *device* yang digunakan. Namun secara durasi, tidak terjadi perbedaan yang signifikan.

V. KESIMPULAN

A. Kesimpulan

Berdasarkan hasil pengujian yang telah dilakukan dapat diambil kesimpulan bahwa:

Algoritma ACO yang diimplementasikan sudah sesuai dengan kebutuhan sistem, yaitu menemukan rute paling optimal pada proses routing dengan mengacu pada total waktu yang dibutuhkan untuk melakukan proses *routing* pada SHMS. Dengan mengasumsikan bahwa proses routing yang memiliki durasi semakin cepat maka energi yang dibutuhkan untuk melakukan routing akan semakin sedikit. Sehingga mengurangi konsumsi energi pada WSN.

Kesimpulan didapatkan didukung dengan perhitungan yang dilakukan secara manual untuk menentukan jarak terpendek dari rute yang dilalui.

Kesimpulan yang telah didapatkan juga didukung dengan adanya perbandingan performansi dengan algoritma GA dari total durasi keseluruhan dalam melakukan *routing* dari sink node menerima sinyal *trigger* untuk membangunkan sensor sampai data yang ditangkap oleh sensor dikirimkan ke sink node.

Waktu rata-rata yang dibutuhkan oleh ACO untuk melakukan *routing* adalah 52.6 ms.

Hasil pengujian yang dilakukan pada ACO dan GA secara berurutan didapatkan rata-rata total durasi sekitar 3.9 s dan 4.0 s.

Hasil Pengujian pada sensor didapatkan bahwa algoritma ACO membutuhkan rata-rata waktu untuk sekali routing sebesar 43.5 ms dan total durasi keseluruhan sebesar 3495.5 ms.

Dan Saran untuk penelitian berikutnya yang berkaitan dengan *routing* untuk optimasi perutean *in-network processing* pada SHMS adalah ketika melakukan pengujian

menggunakan banyak sensor, sehingga hasil yang didapatkan bisa lebih akurat.

REFRENCES

[1] A. Mohajerani and D. Gharavian, "An ant colony optimization based routing algorithm for extending network lifetime in wireless sensor networks," Wireless Networks 22, vol. 8, pp. 2637-2647, 2016.
[2] T. Kamel, "Hierarchical routing optimization in wireless sensor networks," PhD diss, 2021.
[3] R. Okoro, O. Ubadike, A. Onumanyi and M. Aibinu, "Routing Optimization in a Wireless Network Using Genetic Algorithm and ILA Routing Metric," Nigerian Journal of Technology 40, vol. 5, pp. 938- 946, 2021.
[4] D. Marco and K. Socha, "An introduction to ant colony optimization," in In Handbook of Approximation Algorithms and Metaheuristics, Second Edition, Chapman and Hall/CRC, 2018, pp. pp. 395-408.
[5] R. Srikakulapu and U. Vinatha, "Optimized design of collector topology for offshore wind farm based on ant colony optimization with multiple travelling salesman problem," Journal of Modern Power Systems and Clean Energy 6, vol. no. 6, pp. 1181-1192, 2018.
[6] M. Sama, P. Pellegrini, A. D'Ariano, J. Rodriguez and D. Pacciarelli, "Ant colony optimization for the real-time train routing selection problem," Transportation Research Part B: Methodological 85, pp. 89- 108, 2016.
[7] G. Stampa, M. Arias, D. Sánchez-Charles, V. Muntés-Mulero and A. Cabellos, "A deep-reinforcement learning approach for software-defined networking routing optimization," arXiv preprint arXiv:1709.07080 , 2017.
[8] S. Liu, H. Leng and L. Han, "Pheromone model selection in ant colony optimization for the travelling salesman problem," Chinese Journal of Electronics 26, vol. 2, pp. 223-229, 2017.
[9] G. Li, P. Liu, C. Le and B. Zhou, "A novel hybrid meta-heuristic algorithm based on the cross-entropy method and firefly algorithm for global optimization," Entropy 21, vol. 5, p. 494, 2019.
[10] G. Li, F. Shuang, P. Zhao and C. Le, "An improved butterfly optimization algorithm for engineering design problems using the cross-entropy method," Symmetry 11, vol. 8, p. 1049, 2019.
[11] A. S. Yadav, V. Sharma, P. Agarwal, A. Swami and P. K. Yadav, "Pharmaceutical drug two- warehouse inventory model under FIFO dispatching policy using ant colony optimization for travelling salesman problem," Linguistics and Culture Review 5, vol. S2, pp. 1148-1171., 2021.
[12] A. S. Yadav, V. Sharma, P. Agarwal, A. Swami and P. K. Yadav, "Pharmaceutical drug two- warehouse inventory model under FIFO dispatching policy using ant colony optimization for travelling salesman problem," Linguistics and Culture Review 5, vol. S2, pp. 1148-1171., 2021.
[13] M. Basuki, M. J. Hidayat and F. B. Aji, "Application of saving matrix methods and cross entropy for capacitated vehicle routing problem (CVRP) resolving," In IOP Conference Series: Materials Science and Engineering, vol. 462, no. 1, p. 012025 IOP Publishing, 2019.
[14] Y. Bao, Z. Chen, S. Wei, Y. Xu, Z. Tang, and H. Li, "The state of the art of data science and engineering in structural health monitoring," Engineering 5, no. 2, pp. 234-242, 2019.
[15] Y. Sun, W. Dong, and Y. Chen, "An improved routing algorithm based on ant colony optimization in wireless sensor networks," IEEE communications Letters 21, no. 6, pp. 1317-1320, 2017.
[16] A. Gupta and S. Srivastava, "Comparative analysis of ant colony and particle swarm optimization algorithms for distance optimization," Procedia Computer Science 173, pp. 245-253, 2020.
[17] B. Santosa, P. Willy, "Metoda Metaheuristik konsep dan implementasi," Guna Widya, Surabaya, 2011.
[18] A. Alexander, H. Sriwindono, "The Comparison of Genetic Algorithm and Ant Colony Optimization in Completing Travelling Salesman Problem." Proceedings of the 2nd International Conference of Science and Technology for the Internet of Things, 2020.
[19] Al-Karaki, N. Jamal, E. A. Jamal, "Routing techniques in wireless sensor networks: a survey", IEEE wireless communications 11, no. 6, pp. 6-28, 2004.
[20] M. Shafiq, H. Ashraf, A. Ullah, S. Tahira, "Systematic Literatur Review on energy efficient routing shemes in WSN-a survey", Mobile Networks and Applications 25, pp. 882-895, 2020.