# VEHICLE ROUTING PROBLEM BERBASIS ANT COLONY SYSTEM UNTUK OPTIMASI PENENTUAN RUTE KENDARAAN PADA SISTEM DISTRIBUSI BARANG DAN JASA

**Indra Maryati[1], Gunawan[2], C. Pickerling[3], Henry Kurniawan Wibowo[4]**

[1,2,3,4] Teknik Informatika, Sekolah Tinggi Teknik Surabaya
Jl. Ngagel Jaya Tengah 73-77, Surabaya 60284
[1] maryati@stts.edu, [2] gunawan@stts.edu, [3] pickerling@stts.edu, [4] henry_k_86@yahoo.com

## Abstrak

*Vehicle Routing Problem* (VRP) merupakan model masalah transportasi yang mengundang banyak minat ilmuwan komputer, yang bertujuan untuk mencari rute optimal untuk sejumlah kendaraan dalam melayani sejumlah customer. VRP memiliki banyak varian sehingga dapat memodelkan bermacam-macam masalah transportasi, logistik dan distribusi. Penelitian ini membahas tentang tiga varian utama VRP, yaitu *Capacitated VRP* (CVRP), *Split Delivery VRP* (SDVRP), dan *VRP with Time Window*s (VRPTW). Tiga varian ini merupakan varian dasar dari VRP yang merupakan dasar dari perumusan varian-varian VRP yang lain. Sehingga ketiga varian tersebut sangat cocok untuk diterapkan dalam sistem distribusi barang.

VRP dapat diselesaikan dengan menggunakan beberapa metode, yaitu metode pasti (*exact methods*), metode heuristik, dan metode metaheuristik. Dan masing-masing metode tersebut mempunyai algoritma yang dapat digunakan, yaitu *Brute-force Search* yang termasuk exact methods, *Nearest Neighbor Insertion* (NNI) yang termasuk metode heuristik, dan *Ant Colony Optimization* (ACO) yang termasuk metode metaheuristik.

Aplikasi pendukung dibuat pada penelitian ini dapat menyelesaikan tiga varian VRP diatas, yaitu CVRP, SDVRP, dan VRPTW. Penelitian ini membandingkan antara ketiga metode yang digunakan, yaitu Brute-force Search, NNI, dan ACO. Penelitian ini menunjukan bahwa ACO dapat menghasilkan solusi yang lebih mendekati optimal dibandingkan dengan Brute-force Search dan NNI.

**Kata kunci** : Vehicle Routing Problem, CVRP, SDVRP, VRPTW, Ant Colony Optimization, Ant Colony System.

## 1. Pendahuluan

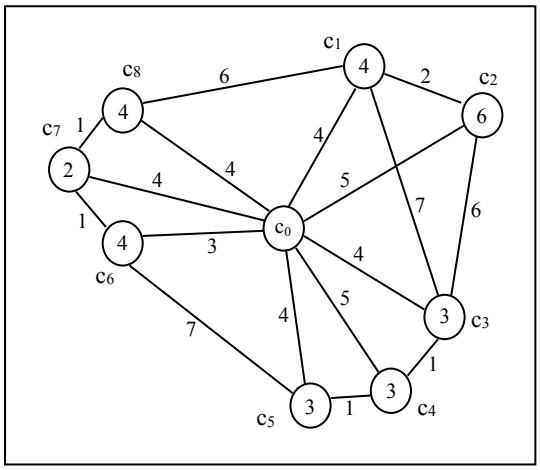
Transportasi merupakan bagian yang tidak terpisahkan dari semua sektor industri. Hal itu dikarenakan hampir semua sektor industri selalu mencakup proses distribusi dan logistik. Transportasi selalu memakan biaya, dan oleh karena itu mempengaruhi biaya produksi dan distri-busi hingga 10-20% dari total biaya suatu produk. Oleh karena itu, efisiensi di bidang transportasi sangat penting dan dapat secara signifikan mengurangi total biaya produksi dan distribusi.

Untuk mencapai pemakaian sarana transportasi yang ideal, diperlukan suatu model, yang dapat menggambarkan berbagai masalah dalam bidang transportasi. Selain itu, diperlukan metode atau algoritma untuk menyelesaikan model masalah tersebut. Dengan permodelan masalah tersebut, akan memudahkan pencarian solusi (karena dapat dikerjakan oleh komputer dengan menggunakan algoritma tertentu) untuk menemukan rute untuk sejumlah kendaraan dengan biaya minimal.

Vehicle Routing Problem (VRP) adalah suatu model yang memiliki banyak varian, yang menggambarkan masalah transportasi sebagai model graf, yang bertujuan untuk menemukan rute dengan biaya minimum untuk pengiriman suatu produk kepada sejumlah *customer* di beberapa lokasi yang berbeda, dengan menggunakan beberapa kendaraan. Setelah diperkenalkan pada tahun 1959 oleh Dantzig dan Ramser melalui makalah mereka yang berjudul ”The Truck Dispatching Problem”, telah banyak metode atau algoritma yang dipakai atau diadaptasi untuk memecahkan VRP dan varian-variannya. Yang membuat VRP menarik untuk dibahas adalah model masalah yang metodenya dapat diimplementasikan ke aplikasi yang sangat fungsional di dunia nyata.

## 2. Capacitated Vehicle Routing Problem (CVRP)

Vehicle Routing Problem (VRP) memiliki bentuk dasar yang biasa disebut Capacitated Vehicle Routing Problem (CVRP), adalah formulasi dari dua masalah optimasi yang cukup terkenal, yaitu Travelling Salesman Problem (TSP) dan Bin Packing Problem (BPP), dan merupakan masalah optimasi untuk menemukan rute dengan biaya minimal (*minimum cost*) untuk sejumlah kendaraan (*vehicles*) dengan kapasitas tertentu yang homogen (*homogeneous fleet*), yang melayani permintaan sejumlah *customer* yang kuantitas permintaannya telah diketahui sebelum proses pengiriman berlangsung.




**Gambar 1**
**Contoh Kasus CVRP**

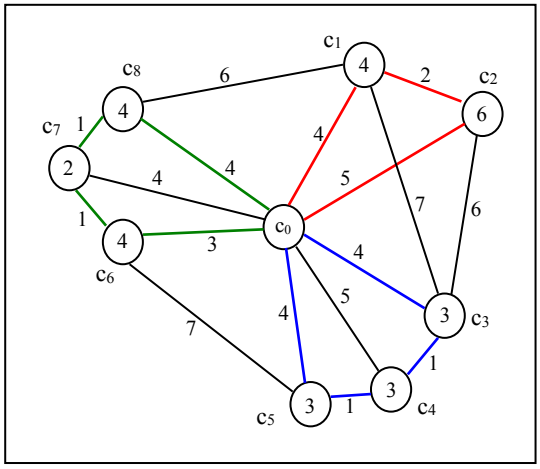
Diberikan contoh kasus sederhana yang representasi grafnya dapat dilihat pada gambar 1. Terdapat 8 *customer* yang harus dilayani. Kendaraan yang tersedia di *central depot* adalah 3. Kapasitas maksimum tiap kendaraan adalah 10 satuan. *Node*/*vertex* adalah *central depot* ($C_0$) dan *customer* ($C_1$-$C_8$). Angka di dalam *node* adalah kuantitas permintaan *customer* (*demand*). Angka di *edge* (*edge weight*) adalah biaya perjalanan antara dua *node* yang dihubungkan edge tersebut. Tidak semua *edge* digambar pada representasi graf, karena akan menjadi terlalu rumit jika semua *edge* digambar.

Dari permasalahan CVRP diatas, solusi yang dicari adalah rute untuk 3 kendaraan, yang melayani 8 *customer*, dengan total biaya perjalanan minimal, dan tidak boleh melanggar batasan kapasitas maksimum kendaraan. Solusi optimal untuk contoh masalah diatas dapat dilihat di gambar 2.

*Edge-edge* yang berwarna merah adalah rute kendaraan pertama, yang melayani *customer* 1 dan 2. *Edge-edge* yang berwarna biru adalah rute kendaraan kedua, yang melayani *customer* 3, 4, dan 5. Sedangkan *edge-edge* berwarna hijau merepresentasikan rute kendaraan ketiga, yang melayani *customer* 6, 7, dan 8. Untuk detail rute dapat dilihat dibawah ini.

Rute kendaraan 1 = 0-1-2-0
Rute kendaraan 2 = 0-3-4-5-0
Rute kendaraan 3 = 0-6-7-8-0
Total biaya = 30 satuan

Cara menghitung total biaya adalah dengan menjumlahkan semua *edge weight* (biaya perjalanan) yang merupakan bagian dari rute kendaraan. Kendaraan pertama menempuh 4+2+5=11, kendaraan kedua menempuh 4+1+1+4=10, dan kendaraan ketiga menempuh 3+1+1+4=9. Total biaya adalah 11+10+9=30 satuan.



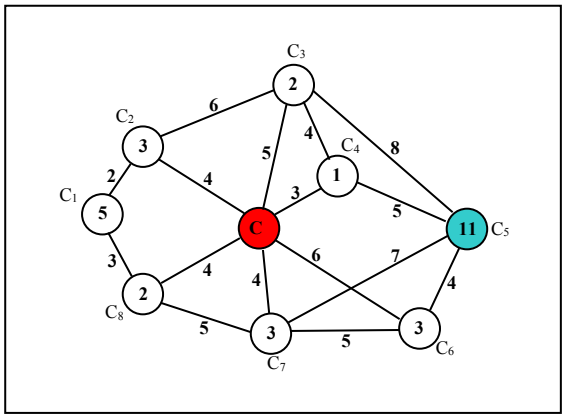

**Gambar 2**
**Solusi Kasus CVRP**

## 3. Split Delivery Vehicle Routing Problem (SDVRP)

Split Delivery Vehicle Routing Problem (SDVRP) adalah salah satu varian dari Vehicle Routing Problem yang menghapus salah satu *constraint* yang mendefinisikan VRP, yaitu tiap *customer* harus dilayani oleh satu kendaraan saja. Pada SDVRP, satu *customer* dapat dilayani oleh lebih dari satu kendaraan. Dengan demikian definisi dari SDVRP adalah masalah pencarian rute optimal atau rute dengan biaya minimum untuk sejumlah kendaraan yang melayani permintaan pengiriman barang oleh sejumlah *customer* yang kuantitas barangnya telah ditentukan sebelumnya, dan barang permintaan suatu *customer* dapat dimuat terpisah (di-*split*) di beberapa kendaraan.

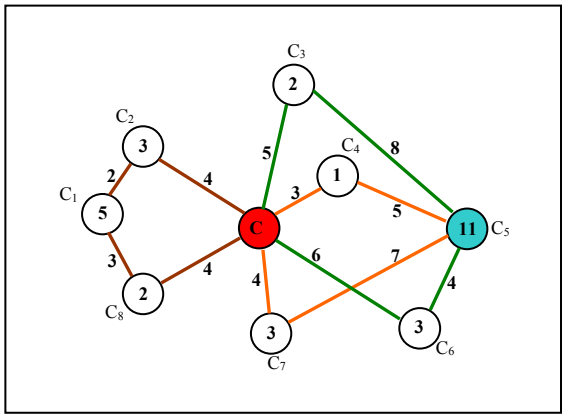
Diberikan contoh kasus sederhana yang representasi grafnya dapat dilihat pada gambar 3. Terdapat 3 kendaraan yang tersedia di suatu *central depot* untuk melayani 8 *customer*. Kapasitas maksimum kendaraan adalah 10 satuan. Terdapat 1 *customer* yang memesan barang melebihi kapasitas maksimum kendaraan, yang membuat customer tersebut harus dilayani oleh 2 kendaraan (*split delivery*). *Vertex* yang berwarna merah adalah *central depot*. *Vertex* yang berwarna biru

merepresentasikan *customer* yang dilayani oleh lebih dari satu kendaraan (*split delivery*). Angka-angka yang terdapat di dalam *vertex* merepresentasikan kuantitas permintaan customer (*demand*). Angka yang terdapat pada *edge* (*edge weight*) adalah biaya perjalanan antara dua lokasi yang dihubungkan *edge* tersebut, yang satuannya dapat berupa jarak, waktu, atau satuan lainnya.




**Gambar 3**
**Contoh Kasus SDVRP**

Dari contoh kasus SDVRP yang telah dijabarkan diatas, solusi yang dicari adalah rute optimal atau rute dengan biaya minimal untuk 3 kendaraan dalam melayani 8 *customer*, yang salah satunya memesan barang diatas kapasitas maksimum kendaraan. *Customer* $C_5$ (*vertex* berwarna biru) memesan barang dengan kuantitas 11 satuan, yang melebihi kapasitas maksimum kendaraan sebesar 10 satuan. Oleh karena itu *customer* $C_5$ akan dilayani oleh 2 kendaraan. Solusi optimal untuk contoh masalah diatas dapat dilihat di gambar 4.




**Gambar 4**
**Solusi Kasus SDVRP**

*Edge-edge* yang berwarna coklat adalah rute kendaraan pertama, yang melayani *customer* 1, 2, dan 8. *Edge-edge* yang berwarna hijau adalah rute kendaraan kedua, yang melayani *customer* 3, sebagian permintaan *customer* 5 sebesar 5 satuan, dan 6. Sedangkan *edge-edge* berwarna oranye merepresentasikan rute kendaraan ketiga, yang melayani *customer* 4, sisa permintaan customer 5 sebesar 6 satuan, dan 7. Untuk detail rute dapat dilihat dibawah ini.

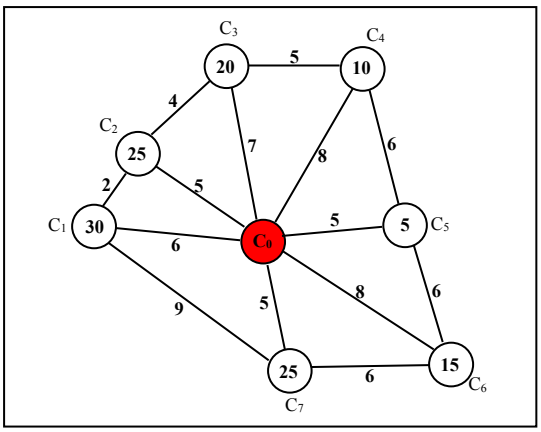
Rute #1 : $C_8 - C_1 - C_2$
Rute #2 : $C_3 - C_5$ (5) $- C_6$
Rute #3 : $C_4 - C_5$ (6) $- C_7$
Cost 55

Cara menghitung total biaya adalah dengan menjumlahkan semua *edge weight* (biaya perjalanan) yang merupakan bagian dari rute kendaraan. Kendaraan pertama menempuh 4+3+2+4=13, kendaraan kedua menempuh 5+8+4+6=23, dan kendaraan ketiga menempuh 3+5+7+4=19. Total biaya adalah 13+23+19=55 satuan.

## 4. Vehicle Routing Problem with Time Windows (VRPTW)

Vehicle Routing Problem with Time Windows adalah turunan dari Capacitated Vehicle Routing Problem dengan tambahan properti *time windows* yang diasosiasikan pada tiap-tiap *customer* dan *central depot*. *Time windows* yang diasosiasikan pada tiap *customer* adalah *range* waktu dimana *customer* dapat menerima barang kiriman. *Time windows* yang diasosiasikan pada *central depot* adalah *range* waktu antara semua kendaraan memulai rute dan waktu paling lambat untuk mengakhiri rute.

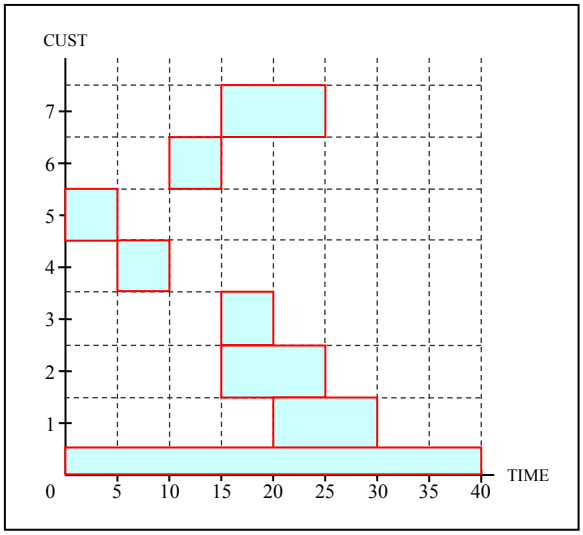
*Time windows* terdiri dari waktu buka dan waktu tutup *customer* dan *central depot*. Jika suatu *customer* buka pukul 7-10 pagi, maka kendaraan harus sampai ke *customer* tersebut sebelum jam 10 pagi. Jika kendaraan tersebut sampai sebelum pukul 7 pagi, maka kendaraan tersebut harus menunggu sampai waktu itu untuk mulai melayani *customer*. Pada varian ini juga diperkenalkan istilah *service time* (waktu pelayanan), yang merepresentasikan waktu yang dibutuhkan untuk melayani suatu *customer*. *Service time* dapat berupa waktu yang dibutuhkan untuk menurunkan barang dari suatu kendaraan.




**Gambar 5**
**Contoh Kasus VRPTW**

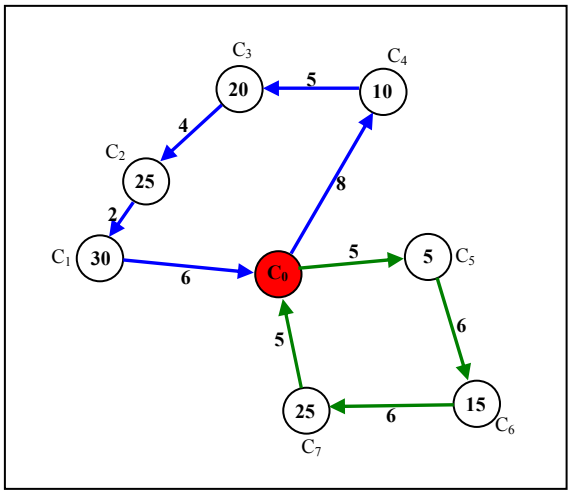
*Time windows* yang dijabarkan diatas merupakan *single time windows* (satu interval). Pada kasus lain, terdapat kemungkinan bahwa interval lebih dari satu, yang biasa disebut *multiple time windows*. Contoh dari kasus ini adalah sebuah toko yang buka pada jam 7-11 pagi (interval pertama) dan jam 3-6 sore (interval kedua). Jika tiap kendaraan "harus" sampai di suatu *customer* sebelum *customer* itu tutup, maka *time windows* yang dipakai tergolong sebagai *hard*. Jika kendaraan "boleh" sampai setelah *customer* tutup, tetapi harus membayarkan sejumlah denda/penalti, maka *time windows* tergolong sebagai *soft*.

Diberikan contoh kasus sederhana yang representasi grafnya dapat dilihat pada gambar 5. Terdapat 7 *customer* yang harus dilayani. Kendaraan yang tersedia di *central depot* adalah 2. Kapasitas maksimum tiap kendaraan adalah 10 satuan. *Vertex-vertex* merepresentasikan *central depot* dan semua *customer*. $C_0$ merepresentasikan *central depot*, sedangkan $C_1$-$C_7$ merepresentasikan *customer* yang dilayani. Angka yang berada di dalam *vertex* merepresentasikan akhir dari *time windows* (*end*). *Edge weight* (angka yang berada di tiap-tiap *edge*) merepresentasikan biaya perjalanan yang berupa waktu perjalanan antara 2 lokasi yang dihubungkan oleh *edge* tersebut dalam suatu satuan waktu. Berikut ini diberikan grafik rentang *time windows* masing-masing *customer* dan *central depot*.




**Gambar 6**
**Time Windows**

Garis horizontal pada grafik di gambar 6 menunjukkan garis waktu. Sedangkan garis vertikal menunjukkan indeks *customer*, mulai dari *central depot* (0) sampai dengan *customer* ke 7. Batang horizontal yang berwarna biru muda pada grafik tersebut menunjukkan rentang waktu antara waktu buka dan tutup para *customer* dan *central depot*. Waktu awal keberangkatan kendaraan adalah waktu awal *time windows* dari *central depot*, yaitu 0. Untuk representasi graf dari contoh kasus yang diberikan dapat dilihat pada gambar 7.

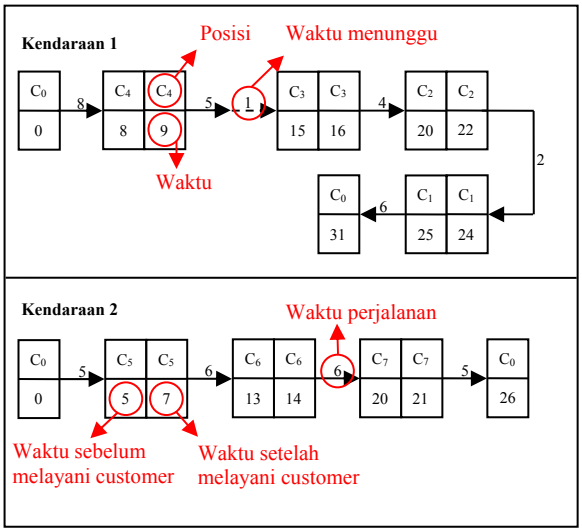



**Gambar 7**
**Solusi Kasus VRPTW**

Dari permasalahan yang telah dijabarkan, solusi yang dicari adalah rute optimal atau rute dengan total waktu perjalanan minimal untuk dua kendaraan yang berangkat dari *central depot* untuk melayani 7 *customer* yang masing-masing memiliki *time windows*.
Rute dari kendaraan pertama direpresentasikan oleh *edge-edge* yang berwarna biru, sedangkan rute dari kendaraan kedua direpresentasikan oleh *edge-edge* yang berwarna hijau. Untuk detail rute dapat dilihat dibawah ini.

Rute #1 : $C_4 - C_3 - C_2 - C_1$
Rute #2 : $C_5 - C_6 - C_7$
Cost 47

Untuk lebih jelasnya dapat dilihat pada diagram di gambar 8.




**Gambar 8**
**Diagram Solusi VRPTW**

Cara menghitung total biaya adalah dengan menjumlahkan semua *edge weight* (biaya perjalanan) yang merupakan bagian dari rute kendaraan. Kendaraan pertama menempuh 8+5+4+2+6=25, dan kendaraan kedua menempuh 5+6+6+5=22. Total biaya adalah 25+22=47 satuan.

## 5. Metode Penyelesaian VRP

Metode penyelesaian VRP dibagi menjadi tiga kategori, yaitu metode pasti (*exact methods*), metode heuristik, dan metode metaheuristik. Metode-metode yang dibahas pada penelitian ini adalah:

### 5.1 Brute Force Search

Metode penyelesaian masalah yang sangat umum dan sederhana, yang secara sistematis membangun (*generate*) semua kemungkinan kandidat solusi dari suatu masalah dan mengecek (*test*) apakah tiap kandidat itu valid, dan mengambil satu kandidat valid pertama yang ditemukan atau kandidat terbaik (untuk masalah optimasi) sebagai solusi.

### 5.2 Nearest Neighbor Insertion (NNI)

Cara kerja metode ini adalah sebagai berikut. Pertama-tama, semua rute kendaraan masih kosong. Dimulai dari rute kendaraan pertama, metode ini memasukkan (*insert*) satu persatu *customer* terdekat (*nearest neighbor*) yang belum dikunjungi ke dalam rute, selama memasukkan *customer* tersebut ke dalam rute kendaraan tidak melanggar batasan kapasitas maksimum kendaraan tersebut (atau batasan-batasan yang dijabarkan oleh varian VRP yang lain). Kemudian proses yang sama juga dilakukan untuk kendaraan-kendaraan berikutnya, sampai semua kendaraan telah penuh atau semua *customer* telah dikunjungi.

### 5.3 Ant Colony Optimization (ACO)

ACO adalah suatu metode penyelesaian masalah optimasi yang berupa kumpulan beberapa algoritma yang menggunakan teknik probabilistik dan prinsip komunikasi koloni semut dalam mencari makanan. Algoritma ACO yang dipakai untuk menyelesaikan 3 varian VRP dalam aplikasi yang dibuat untuk penelitian ini adalah Ant Colony System (ACS).

Terinspirasi oleh cara koloni semut dalam mencari rute ke sumber makanan, metode ini meniru sistem komunikasi koloni semut yang meninggalkan zat kimia yang disebut feromon di rute-rute perjalanan mereka. Semut yang menemukan sumber makanan, akan meninggalkan feromon di rute saat kembali ke koloninya. Semut lain yang mencium feromon di suatu rute, akan cenderung untuk mengikuti rute tersebut jika kandungan feromon cukup padat. Semakin padat kandungan feromon pada suatu rute, semakin besar kemungkinan semut lain mengikuti rute tersebut. Feromon akan mengalami penguapan seiring berjalannya waktu. Rute yang pendek akan mengandung feromon yang cukup padat, karena waktu yang digunakan untuk pulang-pergi (tiap kali pulang ke koloninya, semut selalu meninggalkan feromon) dari koloni ke sumber makanan lebih sedikit, yang menyebabkan penguapan feromon menjadi minimal. Selain feromon, yang juga mempengaruhi semut dalam memilih rute adalah *visibility*. *Visibility* merupakan naluri semut untuk menemukan rute terdekat menuju sumber makanan ataupun kembali ke koloninya. Dua komponen penting ini, yaitu feromon dan *visibility*, akan digunakan ACO sebagai komponen utama.

## 6. Uji Coba

Proses ujicoba akan dilakukan pada *dataset* yang merupakan *open problem*. Untuk varian CVRP dan SDVRP, *dataset* yang digunakan adalah *dataset* yang mempunyai format TSPLIB. Untuk CVRP, akan digunakan *dataset* dari Augerat (et. al.), sedangkan untuk SDVRP, akan digunakan *dataset* buatan sendiri, karena belum adanya *open problem* untuk varian ini. Sedangkan untuk VRPTW, *dataset* yang digunakan adalah *dataset* dari Solomon yang memiliki format tersendiri. Hasil ujicoba ketiga metode terhadap CVRP dapat dilihat pada tabel 1.

Untuk metode ACS, *setting* parameter yang digunakan adalah 1000 iterasi, 20 semut artifisial, *beta* = 2, *p* (*evaporation rate*) = 0,1 (10%), dan $q_0$ (probabilitas *exploitation*) = 0,9 (90%). Untuk tiap masalah, metode ACS dijalankan 10 kali, dan diambil satu solusi dengan *cost* minimum. *Setting* parameter ini juga berlaku untuk varian SDVRP dan VRPTW.

**Tabel 1**
**Hasil Ujicoba CVRP**

| Instance | Solution Cost | | | |
|---|---|---|---|---|
| | Best Known | Brute Force | NNI | ACS |
| CVRP-n8-k3 | 375 | 375 | 375 | 375 |
| CVRP-n10-k3 | 455 | 455 | 477 | 455 |
| A-n32-k5 | 784 | - | 961 | 789 |
| B-n41-k6 | 829 | - | 1184 | 834 |
| A-n65-k9 | 1177 | - | 1465 | 1250 |

Dari hasil yang ditampilkan di tabel 1, dapat dilihat bahwa ACS yang termasuk dalam metode metaheuristik, merupakan metode yang terbaik dalam memberikan solusi untuk varian CVRP jika dibandingkan Brute-Force Search dan NNI. Metode ACS mampu memberikan solusi yang optimal untuk 2 *instance* yang dibuat sendiri, dan solusi yang mendekati optimal untuk 9 *instance* dari *dataset* yang dibuat Augerat et. al. dengan format TSPLIB.

**Tabel 2**
**Hasil Ujicoba SDVRP**

| Instance | Solution Cost | | |
|---|---|---|---|
| | Brute Force | NNI | ACS |
| SDVRP-n8-k3 | 481 | 513 | 481 |
| SDVRP-n10-k3 | 421 | 486 | 421 |
| SDVRP-n32-k7 | - | 1265 | 1135 |
| SDVRP-n65-k11 | - | 1915 | 1525 |

Dari hasil proses *testing* yang dapat dilihat pada tabel 2, dapat disimpulkan bahwa metode ACS bekerja lebih efektif dibanding metode Brute Force dan NNI. Karena tidak diketahuinya solusi optimal dari masalah yang dibuat, tidak ada perbandingan antara solusi yang dihasilkan ketiga metode dengan solusi terbaik yang ditemukan (*best known*).

**Tabel 5.3**
**Hasil Testing VRPTW**

| Instance | Solution Cost | | | |
|---|---|---|---|---|
| | Best Known | Brute Force | NNI | ACS |
| VRPTW-n7 | 375 | 375 | - | 375 |
| VRPTW-n10 | 485 | 485 | - | 485 |
| C101 (25) | 192 | - | 480 | 192 |
| R106 (50) | 794 | - | 1268 | 864 |
| RC108 (100) | 1176 | - | 1697 | 1251 |

Dari hasil *testing* yang dapat dilihat pada tabel 3, dapat disimpulkan bahwa performa metode ACS jauh lebih baik dibandingkan dengan NNI dan Brute Force Search. Metode NNI menghasilkan solusi yang sangat jauh dari solusi terbaik yang ditemukan. Metode Brute Force Search hanya mampu menghasilkan solusi untuk masalah dengan skala kecil. Metode ACS menghasilkan solusi yang mendekati optimal.

## 7. Kesimpulan

Dari hasil pengamatan yang dilakukan untuk setiap tahap yang dilewati dalam pembuatan penelitian ini, maka dapat ditarik beberapa kesimpulan seperti berikut:

- VRP merupakan masalah yang lebih kompleks dan sulit jika dibandingkan TSP dan BPP. Metode pasti dapat menyelesaikan masalah TSP dan BPP dengan skala yang relatif besar, sedangkan masalah VRP skala menengah ada yang belum dapat diselesaikan oleh metode pasti.
- Metode Brute-Force Search hanya dapat menyelesaikan masalah VRP dengan skala kecil. Hal ini dikarenakan metode ini menelusuri setiap kandidat solusi yang jumlahnya menjadi sangat besar dengan bertambahnya skala masalah.
- Metode NNI dapat membutuhkan waktu komputasi yang sangat sedikit, tetapi tidak mampu memberikan solusi yang mendekati optimal untuk masalah berskala besar. Penggunaan heuristik yang sangat sederhana menyebabkan kualitas solusi yang dihasilkan metode NNI menjadi semakin jauh dari optimal seiiring bertambahnya skala masalah VRP yang diselesaikan.
- Metode ACO merupakan metode penyelesaian terbaik jika dibandingkan dengan Brute-Force Search dan NNI. Metode ACO mampu menyelesaikan masalah VRP berskala besar dan mampu menghasilkan solusi yang mendekati optimal. Selain itu waktu komputasi atau jumlah iterasi ACO dapat diatur sesuai dengan sumber daya waktu yang dimiliki.

**Daftar Pustaka:**


[1] Applegate and Bixby and Chvatal and Cook, 2007, *The Traveling Salesman Problem*, Princeton University Press.

[2] Dantzig and Ramser, 1959, *The Truck Dispatching Problem*, INFORMS.

[3] Diaz, Bernabé, 2006, *CVRPTW Instances*, (http://neo.lcc.uma.es/radiaeb/WebVRP/Problem_Instances/CVRPTWInstances.html)

[4] Diaz, Bernabé, 2006, *Description for Files of Cordeau's Instances*, (http://neo.lcc.uma.es/radi-eb/WebVRP//index.html?/Problem_Instances/CordeauFilesDesc.html)

[5] Diaz, Bernabé, 2006, *SDVRP Instances*, (http://neo.lcc.uma.es/radi-aeb/WebVRP/Problem_Instances/SDVRPInstances.html)

[6] Diaz, Bernabé, 2006, *VRPPD Instances*, (http://neo.lcc.uma.es/radi-aeb/WebVRP//index.html?/Problem_Instances/VRPPDInstances.html)

[7] Dorigo, Marco and Birattari, Mauro and Stutzle, Thomas, 2006, *Ant Colony Optimization*, IRIDIA.

[8] Dorigo, Marco and Birattari, Mauro and Stuzle, 1996, *Ant Colony System: A Cooperative Learning Approach to the Traveling Salesman Problem*, IRIDIA.

[9] Reinelt, Gerhard, 1997, *TSPLIB*, (http://elib.zib.de/pub/mp-testdata/tsp/tsplib/tsplib.html)

[10] Skorobohatyj, Georg, 1995, MP-TESTDATA - *The TSPLIB Capacitated Vehicle Routing Problem Instances*, (http://elib.zib.de/pub/mp-testdata/tsp/tsplib/vrp/index.html)

[11] Takes, Frank, 2010, Applying Monte Carlo *Techniques to the Capacitated Vehicle Routing Problem*, Leiden University

[12] Toth, Paolo and Vigo, Daniele, 2002, *The Vehicle Routing Problem*, SIAM

[13] _____. *Ant Colony Optimization*, (http://en.wikipedia.org/wiki/Ant_colony_optimization)

[14] _____. *Brute-force search*, (http://en.wikipedia.org/wiki/Brute-force_search)

[15] _____. *Travelling Salesman Problem*, (http://en.wikipedia.org/wiki/Travelling_salesman_problem)