TUGAS AKHIR

OPTIMASI RUTE KUNJUNGAN *CLUSTER SALES OFFICER* (CSO) MENGGUNAKAN *ANT COLONY OPTIMIZATION* (ACO)

(STUDI KASUS: INDOSAT OOREDOO HUTCHISON *MICRO CLUSTER* MAMUJU)

Diajukan untuk Memenuhi Salah Satu Syarat Ujian Guna Memperoleh
Gelar Sarjana Teknik Pada Departemen Teknik Industri Fakultas Teknik
Universitas Hasanuddin

Disusun Oleh:

IRWIN YAPUTERA

D071181323

DEPARTEMEN TEKNIK INDUSTRI

FAKULTAS TEKNIK

UNIVERSITAS HASANUDDIN

2022

TUGAS AKHIR

OPTIMASI RUTE KUNJUNGAN *CLUSTER SALES OFFICER* (CSO) MENGGUNAKAN *ANT COLONY OPTIMIZATION* (ACO)

(STUDI KASUS: INDOSAT OOREDOO HUTCHISON *MICRO CLUSTER* MAMUJU)

Diajukan untuk Memenuhi Salah Satu Syarat Ujian Guna Memperoleh
Gelar Sarjana Teknik Pada Departemen Teknik Industri Fakultas Teknik
Universitas Hasanuddin

Disusun Oleh:

IRWIN YAPUTERA

D071181323

DEPARTEMEN TEKNIK INDUSTRI

FAKULTAS TEKNIK

UNIVERSITAS HASANUDDIN

2022

LEMBAR PENGESAHAN

Judul Tugas Akhir:

OPTIMASI RUTE KUNJUNGAN *CLUSTER SALES OFFICER* (CSO)

MENGGUNAKAN *ANT COLONY OPTIMIZATION* (ACO)

(STUDI KASUS: INDOSAT OOREDOO HUTCHISON *MICRO CLUSTER* MAMUJU)

Disusun Oleh:

IRWIN YAPUTERA

D071181323

Tugas akhir ini diajukan untuk memenuhi salah satu persyaratan dalam menyelesaikan studi guna memperoleh gelar Sarjana Teknik pada Departemen Teknik Industri Fakultas Teknik Universitas Hasanuddin.

Telah diperiksa dan disetujui oleh:

Dosen Pembimbing I	Dosen Pembimbing II
Dr. Ir. Rosmalina Hanafi, M.Eng	**Dr. Eng. Ir. Muhammad Rusman, ST., MT., IPU**
NIP. 19660128 199103 2 003	NIP. 19741024 200312 1 002

Mengetahui,

Ketua Departemen Teknik Industri Fakultas Teknik

Universitas Hasanuddin

Dr. Ir. Saiful, ST., MT., IPM

NIP. 19810606 200604 1 004

PERNYATAAN KEASLIAN

Yang bertanda tangan dibawah ini:

Nama	: Irwin Yaputera
NIM	: D071181323
Program Studi	: Teknik Industri
Jenjang	: S1
Judul Skripsi	: Optimasi Rute Kunjungan *Cluster Sales Officer* (CSO) Menggunakan *Ant Colony Optimization* (ACO) (Studi Kasus: Indosat Ooredoo Hutchison *Micro Cluster* Mamuju)

Menyatakan dengan sesungguhnya bahwa Skripsi ini merupakan hasil, pemikiran, dan pemaparan asli dari saya sendiri. Saya tidak mencantumkan tanpa pengakuan bahan-bahan yang telah dipublikasikan sebelumnya atau ditulis oleh orang lain atas sebagai bahan yang pemah diajukan untuk gelar atau ijazah pada Universitas Hasanuddin atau perguruan tinggi lainnya.

Apabila dikemudian terdapat penyimpangan dan ketidakbenaran dalam pemyataan ini, maka saya bersedia menerima sanksi akademik sesuai dengan peraturan yang berlaku di Universitas Hasanuddin.

Demikian pernyataan ini saya buat.

Gowa, 23 Agustus 2022
Yang membuat pernyataan

Irwin Yaputera

ABSTRAK

Indosat Ooredoo Hutchison adalah sebuah perusahaan swasta yang bergerak di bidang telekomunikasi. Perusahaan swasta ini memiliki dua alur bisnis untuk memperoleh keuntungannya, yaitu *modern channel* dan *traditional channel*. Kedua alur bisnis memiliki kebermanfaatannya masing-masing, dimana *modern channel* melakukan kegiatan bisnis melalui aplikasi dan media elektronik sedangkan *traditional channel* melakukan kegiatan bisnis melalui toko konvensional (*outlets*). Proses bisnis pada *traditional channel* dipengaruhi oleh performa dari *salesman*, dimana dalam perusahaan disebut dengan *Cluster Sales Officer* (CSO). CSO menentukan rute kunjungan *outlets* berdasarkan pengalaman sehingga rute kunjungannya belum optimal yang menyebabkan jarak dan waktu tempuh serta biaya masih belum efisien.

Penelitian ini dilakukan untuk menyelesaikan permasalahan TSP (*Travelling Salesman Problem*) dengan menentukan rute kunjungan yang optimal menggunakan algoritma *Ant Colony Optimization.* Tujuannya untuk meminimalkan jarak dan waktu tempuh serta biaya. Data yang diperoleh adalah data koordinat dan urutan kunjungan CSO pada Rute I (Senin, Rabu dan Jumat) dan Rute II (Selasa, Kamis dan Sabtu). Berdasarkan data tersebut, peneliti kemudian menghitung jarak dan waktu tempuh serta biaya dikeluarkan pada rute awal CSO. Langkah selanjutnya adalah menghitung jarak dan waktu tempuh serta biaya yang dikeluarkan pada rute usulan dengan algoritma *Ant Colony Optimization*.

Hasil perhitungan menunjukkan bahwa dengan menggunakan algoritma *Ant Colony Optimization*, Rute I Usulan ACO yang memiliki total jarak yang ditempuh, waktu yang diperlukan serta biaya yang harus dikeluarkan diperoleh efisiensi sekitar 8% dari Rute I Awal CSO. Sedangkan untuk Rute II Usulan ACO yang memiliki total jarak yang ditempuh, waktu yang diperlukan serta biaya yang harus dikeluarkan diperoleh efisiensi sekitar 4% dari Rute II Awal CSO. Sehingga diperoleh efisiensi Rute Usulan ACO dalam sebulan sekitar 6% terhadap Rute Awal CSO.

Kata Kunci: Perusahaan Telekomunikasi, *Travelling Salesman Problem* (TSP), Algoritma *Ant Colony Optimization*, *Matlab*

ABSTRACT

Indosat Ooredoo Hutchison is a private company engaged in the telecommunications sector. This private company has two business channel to gain profits, namely modern channels and traditional channels. Both business channel have their respective benefits, where the modern channel conducts business activities through applications and electronic media, while the traditional channel conducts business activities through conventional stores (outlets). Business processes in traditional channels are influenced by the performance of salesmen, which in the company are called Cluster Sales Officers (CSOs). CSO determines the route of visiting outlets based on experience so that the visit route is not optimal which causes distance and travel time and costs are still not efficient.

This research was conducted to solve the TSP (Travelling Salesman Problem) problem by determining the optimal route using the Ant Colony Optimization algorithm. The goal is to minimize the distance and travel time and costs. The data obtained are the coordinates and order of CSO visits on Route I (Monday, Wednesday and Friday) and Route II (Tuesday, Thursday and Saturday). Based on these data, the researchers then calculated the distance and travel time as well as the costs incurred on the initial CSO route. The next step is to calculate the distance and travel time as well as the costs incurred on the proposed route with the Ant Colony Optimization algorithm.

The calculation results show that by using the Ant Colony Optimization algorithm, Route I Proposed ACO which has the total distance traveled, time required and costs incurred, obtained an efficiency of around 8% from Route I Initial CSO. As for the ACO Proposed Route II which has a total distance traveled, the time required and the costs to be incurred, an efficiency of around 4% is obtained from the CSO Initial Route I. So that the efficiency of the proposed ACO Route in a month is around 6% compared to the CSO Initial Route.

Keywords: Telecommunication Company, Traveling Salesman Problem (TSP), Ant Colony Optimization Algorithm, Matlab

KATA PENGANTAR

Puji dan syukur kita panjatkan kepada Tuhan Yang Maha Esa, karena atas berkat dan rahmat-Nya, sehingga penulis dapat menyelesaikan skripsi dengan judul "Optimasi Rute Kunjungan *Cluster Sales Officer* (CSO) Menggunakan *Ant Colony Optimization* (ACO) (Studi Kasus: Indosat Ooredoo Hutchison *Micro Cluster* Mamuju)." Penulisan skripsi ini dilakukan dalam rangka memenuhi salah satu syarat untuk memperoleh gelar Sarjana Teknik pada Departemen Teknik Industri Fakultas Teknik Universitas Hasanuddin.

Tidak dapat disangkal bahwa butuh usaha yang keras, kegigihan, dan kesabaran, dalam pengerjaan skripsi ini. Namun disadari karya ini tidak akan selesai tanpa orang-orang tercinta disekeliling yang senantiasa mendukung dan membantu. Terima kasih yang sebesar-besarnya penulis sampaikan kepada:

a. Kedua orang tua dan keluarga penulis yang telah memberikan bantuan material dan moral serta doa yang tiada hentinya;
b. Bapak Dr. Saiful, S.T., M.T., IPM selaku Ketua Departemen Teknik Industri Fakultas Teknik Universitas Hasanuddin;
c. Ibu Dr. Ir. Rosmalina Hanafi, M.Eng. selaku Dosen Pembimbing I dan Bapak Dr. Eng. Ir. Muhammad Rusman, S.T., M.T., IPU selaku Dosen Pembimbing II tugas akhir ini yang telah menyediakan waktu, tenaga, dan pikiran selama proses bimbingan;
d. Bapak Dr. Ir. Syarifuddin M. Parenreng, S.T., M.T., IPU dan Ibu Dwi Handayani, S.T., M.T. selaku Dosen Penguji dalam seminar hasil dan sidang

skripsi penulis yang telah memberikan masukan perbaikan tugas akhir penulis;

e. Segenap Dosen Departemen Teknik Industri Fakultas Teknik Universitas Hasanuddin yang telah mendidik dan memberikan ilmu selama berkuliah di sini dan seluruh staff yang selalu dengan sabar melayani segala administrasi selama proses penelitian ini.
f. Rekan seperjuangan, FEAZ18LE, yang telah memberikan banyak cerita dalam hidup yang sulit dijelaskan dengan kata-kata, selain kita untuk selamanya;
g. Keluarga besar HMTI FT–UH yang telah menjadi wadah untuk berproses memperoleh pengalaman berharga;
h. Keluarga besar WELCOME09 SMFT-UH yang telah menjadi wadah pengembangan diri yang luar biasa dan teman-teman hebat;
i. Serta semua pihak yang turut membantu penulis yang tidak dapat ditulis satu per satu.
j. Last but not least, I wanna thank me. I wanna thank me for believing in me. I wanna thank me for all doing this hard work. I wanna thank me for having no days off. I wanna thank me for never quitting. I wanna thank me for just being me at all times.

Semoga segala kebaikan dan pertolongan semuanya mendapat balasan limpahan berkah dari Tuhan Yang Maha Esa dan akhirnya penulis menyadari bahwa skripsi ini masih jauh dari kata sempurna, karena keterbatasan ilmu yang dimiliki. Untuk itu penulis dengan kerendahan hati mengharapkan saran dan kritik

yang sifatnya membangun dari semua pihak demi membangun laporan penelitian ini. Semoga skripsi ini membawa manfaat bagi pengembangan ilmu.

Makassar, 15 Agustus 2022

Penulis

Irwin Yaputera

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

BAB I

PENDAHULUAN

1.1 Latar Belakang

Setiap hari manusia akan dihadapkan pada banyak permasalahan hidup yang semakin kompleks. Penyelesaian dari setiap permasalahan harus menghasilkan solusi yang terbaik untuk setiap sumber dayanya. Salah satu permasalahan tersebut adalah pencarian rute optimal, dimana sumber daya dari waktu dan bahan bakar harus diefisienkan seminimum mungkin. Penentuan rute optimal merupakan salah satu keilmuan teknik industri dalam hal optimasi yang bertujuan untuk menghasilkan suatu rute yang efisien baik dalam hal jarak, biaya, waktu atau lain (Tisen, 2013). Masalah seperti ini dikategorikan dalam suatu permasalahan kombinatorial yang dikenal dengan *Traveling Salesman Problem* (TSP). TSP merupakan masalah klasik untuk mencari rute terpendek yang dapat dilewati *salesman* ke sejumlah tempat tanpa harus mendatangi tempat yang sama lebih dari satu kali (Kusrini & Istiyanto, 2007).

Pada perkembangannya, ternyata TSP merupakan persoalan yang banyak diaplikasikan pada berbagai persoalan dunia nyata, misalnya: efesiensi pengiriman surat dan barang, perencanaan pemasangan saluran pipa, masalah transportasi, persoalan *delivery order* (jasa pengantar makanan), dan seterusnya. Salah satu teori algoritma yang membantu dalam penyelesaian TSP adalah *Ant Colony Optimization* (ACO) (Septima, 2008).

Ant Colony Optimization (ACO) merupakan algoritma yang terinspirasi dari perilaku semut dalam menemukan jalan dari sarangnya menuju tempat makanannya. Semut memiliki zat *pheromone* dimana zat ini merupakan zat kimia yang berasal dari kelenjar endokrin dan digunakan oleh makhluk hidup untuk mengenali sesama jenis. Semut-semut tersebut akan memilih jalan berdasarkan kuatnya aroma *pheromone*. Semakin banyak semut yang menempuh suatu jalan tertentu, maka aroma *pheromone* pada lintasan tersebut akan semakin kuat sehingga semut-semut berikutnya akan mengikuti lintasan tersebut (Hotang, 2016). Algoritma *Ant Colony Optimization* (ACO) memiliki keunggulan dapat memberikan solusi dengan batasan sumber daya dapat diatur sendiri serta memberikan solusi rute optimal yang cenderung konstan untuk beberapa kali pengujian.

Penelitian ini dilaksanakan di salah satu perusahaan yang bergerak di bidang distribusi industri telekomunikasi, yakni Indosat Ooredoo Hutchison. Perusahaan swasta ini memiliki dua alur bisnis untuk memperoleh keuntungannya, yaitu *modern channel* dan *traditional channel.* Kedua alur bisnis memiliki kebermanfaatannya masing-masing, dimana *modern channel* melakukan kegiatan bisnis melalui aplikasi dan media elektronik sedangkan *traditional channel* melakukan kegiatan bisnis melalui toko konvensional (*outlets*). Proses bisnis pada *traditional channel* dipengaruhi oleh performa dari *salesman,* dimana dalam perusahaan disebut dengan *Cluster Sales Officer* (CSO). Beban kerja CSO tersebut berorientasi pada target. Oleh sebab itu, CSO pada perusahaan ini harus bekerja cepat dalam menjalankan

perjalanan kunjungan hariannya. Kunjungan harian dilakukan untuk mengecek stok dan menjual saldo, kartu perdana, maupun *voucher* kepada *outlet*, dimana pada umumnya memakan waktu sekitar 15 menit untuk setiap *outlet* yang dikunjungi. CSO memiliki tantangan untuk menyelesaikan sekitar 25 titik kunjungan ke beberapa *outlet* setiap harinya. Permasalahan yang sering terjadi khususnya pada *Micro Cluster* Mamuju, yaitu durasi rute kunjungan yang masih dapat diefisienkan dikarenakan pengambilan rute yang masih belum optimal sebab penentuan rute sekarang masih menggunakan pengalaman CSO itu sendiri.

Berdasarkan uraian yang dijelaskan di atas maka penulis menggunakan *Ant Colony Optimization* (ACO) untuk memecahkan *Travelling Salesman Problem* (TSP) yang dihadapi CSO *Micro Cluster* Mamuju dengan bantuan *software* MATLAB.

1.2 Rumusan Masalah

1. Bagaimana bentuk rute kunjungan usulan yang optimal untuk CSO *Micro Cluster* Mamuju dengan menggunakan algoritma *Ant Colony Optimization* (ACO)?
2. Bagaimana perbandingan jarak dan waktu tempuh serta biaya antara rute usulan dengan algoritma *Ant Colony Optimization* (ACO) dengan rute awal CSO?

1.3 Tujuan Penelitian

1. Menentukan rute kunjungan usulan yang optimal untuk CSO *Micro Cluster* Mamuju dengan menggunakan algoritma *Ant Colony Optimization* (ACO).
2. Membandingkan jarak dan waktu tempuh serta biaya antara rute usulan dengan algoritma *Ant Colony Optimization* (ACO) dengan rute awal CSO.

1.4 Batasan Masalah

Adapun batasan masalah yang digunakan dalam penelitian ini adalah sebagai berikut:

Ruang lingkup pembahasan dititikberatkan pada *Cluster Sales Officer* (CSO) Indosat Ooredoo Hutchison *Micro Cluster* Mamuju yang memenuhi syarat kunjungan dimulai dari kantor dan kembali ke kantor lagi menggunakan kendaraan sepeda motor.

1.5 Manfaat Penelitian

1. Bagi perusahaan
    a. Diharapkan dengan penelitian ini dapat menjadi solusi alternatif bagi perusahaan untuk menentukan rute optimal dalam kunjungan *Cluster Sales Officer* (CSO) Indosat Ooredoo Hutchison *Micro Cluster* Mamuju.
    b. Memberikan gambaran tentang pentingnya penentuan rute dalam pendistribusian *Cluster Sales Officer* (CSO) sehingga produktivitas perusahaan dapat meningkat.

2. Bagi Mahasiswa

   Memahami teori dan penerapan ilmu pengetahuan dan kajian ilmiah akademis dalam penyelesaian masalah TSP dengan algoritma *Ant Colony Optimization* (ACO) menggunakan *software* MATLAB.

1.6 Sistematika Penulisan

Untuk mempermudah dalam memahami alur penelitian, maka penelitian ini terdiri dari beberapa bab dengan sistematika penulisan sebagai berikut:

BAB I PENDAHULUAN

Bab pertama menjelaskan latar belakang dilakukannya penelitian serta terdapat penjelasan mengenai rumusan masalah yang dibahas dalam penelitian, batasan masalah, manfaat serta pembaca sistematika penulisan laporan akhir.

BAB II TINJAUAN PUSTAKA

Bab kedua memuat penjelasan dan dasar teori yang digunakan dalam melakukan penelitian untuk membantu pemahaman dalam mengelola dan analisis data. Landasan teori diperoleh dari studi intearur melalui buku, jurnal, dan skripsi.

BAB III METODOLOGI PENELITIAN

Bab ketiga berisi tentang tempat dan waktu penelitian dilakukan, subjek dan objek penelitian, data penelitian (jenis-jenis data, metode pengambilan data dan pengukuran data

BAB IV PENGUMPULAN DAN PENGOLAHAN DATA

Bab keempat berisi tentang metode pengumpulan data serta pengolahan data dengan metode yang digunakan.

BAB V ANALISA DAN PEMBAHASAN

Bab kelima berisi tentang hasil penelitian yang didapatkan dari penelitian berdasarkan metode yang digunakan.

BAB VI KESIMPULAN DAN SARAN

Bab keenam berisi tentang kesimpulan dari penelitian dan saran untuk pekerja dalam hal ini adalah *Cluster Sales Officer* (CSO) agar nantinya dapat mempertimbangkan hasil penelitian guna kepentingan kedepannya.

BAB II

TINJAUAN PUSTAKA

2.1 Teori *Graph*

Graph merupakan bahasan yang sudah tua usianya namun memiliki banyak terapan sampai saat ini. Representasi visual dari *graph* adalah dengan menyatakan obyek sebagai noktah, bulatan, atau titik. Sedangkan hubungan antar obyek dinyatakan dengan garis. *Graph* merupakan salah satu model matematika yang kompleks dan cukup sulit, akan tetapi bisa juga menjadi solusi yang sangat bagus untuk masalah tertentu. Saat ini teori *graph* semakin berkembang dan menarik karena keunikan dan banyak sekali penerapannya. Salah satu alasan perkembangan teori *graph* yang begitu pesat adalah aplikasinya yang sangat luas dalam kehidupan sehari-hari maupun dalam berbagai bidang ilmu (Budayasa, 2007).

Graph merupakan representasi dari suatu masalah yang digambarkan sebagai sekumpulan titik atau simpul (*vertex*) yang dihubungkan dengan sekumpulan garis atau sisi (*edge*). Secara singkat suatu *graph* dapat ditulis sebagai

$G = (V, E)$ yang dalam hal ini

$V =$ Himpunan berhingga dan titik kosong dari simpul-simpul *vertices*

$= \{v_1, v_2, \dots, v_n\}$

$E =$ Himpunan sisi yang menghubungkan sepasang simpul $= \{e_1, e_2, \dots, e_n\}$

Simpul pada *graph* dapat dinomori dengan huruf seperti v, w, ..., dengan bilangan asli 1,2,3, ..., atau gabungan keduanya. Sisi (*edge*) yang

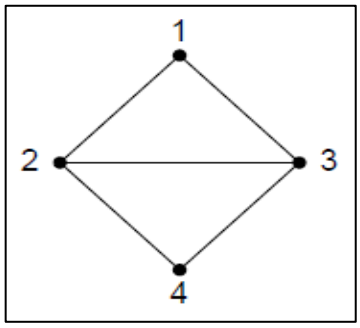
menghubungkan dua simpul (*vertex*) v_i dan v_j, dan dinyatakan dengan pasangan (v_i v_j) atau dengan lambang e_1, e_2, dengan kata lain, jika e adalah sisi yang menghubungkan simpul v_i dan v_j, maka e dapat ditulis sebagai $e = (v_i, v_j)$

2.1.1 Jenis-jenis *Graph*

Sisi pada *graph* dapat mempunyai orientasi arah. Menurut Munir (2007), berdasarkan orientasi arah pada sisi, maka secara umum *graph* dibedakan atas 2 jenis

1. *Graph* Tak Berarah

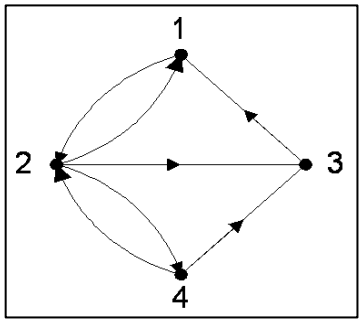
   Graph yang sisinya tidak mempunyai orientasi arah disebut *graph* tak-berarah. Pada *graph* tak berarah, urutan pasangan simpul yang dihubungkan oleh sisi tidak di perhatikan. Jadi $(v_i, v_j) = (v_j, v_i)$.



Gambar 2. 1 *Graph* Tak Berarah

2. *Graph* Berarah

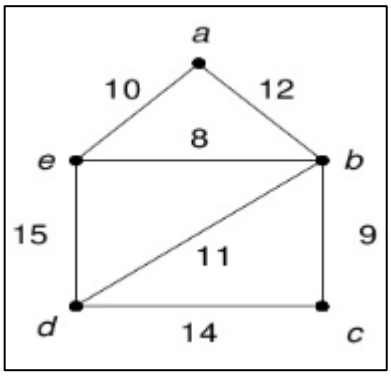
   Graph yang setiap sisinya diberikan orientasi arah disebut sebagai *graph* berarah. Pada *graph* berarah, (v_i , v_j) dan (v_j , v_i), menyatakan duah sisi yang berbeda, dengan kata lain $(v_i , v_j) \neq (v_j , v_i)$.



Gambar 2. 2 *Graph* Berarah

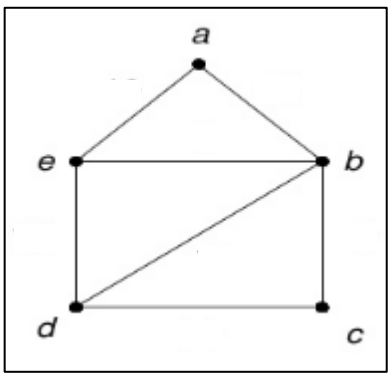
Menurut Siang (2009), berdasarkan bobot setiap sisi secara umum *graph* dibedakan atas 2 jenis:

1. *Graph* berbobot adalah *graph* yang setiap sisinya diberi sebuah bobot atau nilai.



Gambar 2. 3 *Graph* Berbobot

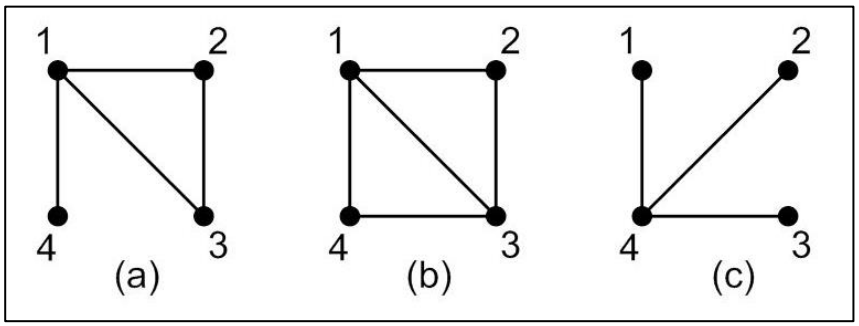
2. *Graph* tak berbobot adalah *graph* yang setiap sisinya tidak diberi bobot atau nilai.



Gambar 2. 4 *Graph* Tak Berbobot

2.1.2 Lintasan dan Sirkuit Hamilton

Lintasan Hamilton adalah lintasan yang melalui tiap simpul di dalam *graph* G tepat satu kali. Bila lintasan itu kembali lagi ke simpul awal dan membentuk lintasan tertutup, maka lintasan tertutup itu dinamakan Sirkuit Hamilton (Helene *et. al*, 2008). Jadi, Sirkuit Hamilton adalah sirkuit yang melalui tiap simpul di dalam *graph* G tepat satu kali, kecuali simpul awal dan simpul akhir.



Gambar 2. 5 Contoh *Graph*

Keterangan Gambar:

(a) *Graph* yang memiliki lintasan Hamilton (misal: 3, 2, 1, 4)

(b) *Graph* yang memiliki sirkuit Hamilton (misal: 1, 2, 3, 4, 1)

(c) *Graph* yang tidak memiliki lintasan maupun sirkuit Hamilton

2.1.3 Representasi *Graph*

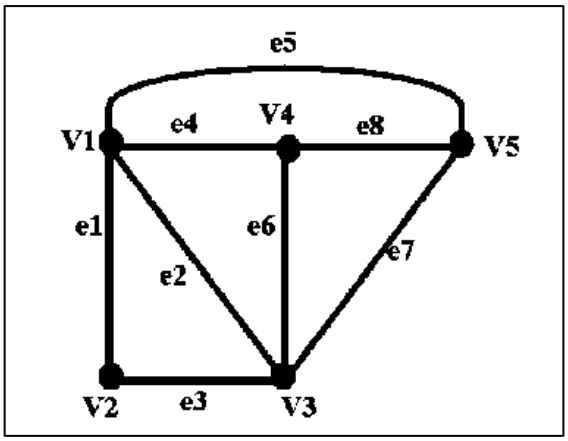
Terdapat beberapa cara mempresentasikan *graph*, dua di antaranya yang sering digunakan adalah matriks ketetanggaan dan matriks bersisian.

1. Matriks Ketetanggaan

   Matriks ketetanggaan didefinisikan sebagai berikut, misalkan $A = (a_{ij})$ adalah matriks $m \times m$ yang didefinisikan oleh,

$$a_{ij} = \begin{cases} 1, v_i \; bertetangga \; dengan \; v_j \\ 0, lainnya \end{cases}$$

Perhatikan Gambar 2.6 menunjukkan *graph* berarah yang terdiri dari 5 simpul dan 8 sisi serta Gambar 2.7 yang menunjukkan matriks ketetanggannya.



Gambar 2. 6 *Graph* dengan 5 simpul dan 8 sisi

	V1	V2	V3	V4	V5
V1	0	1	1	1	1
V2	1	0	1	0	0
V3	1	1	0	1	1
V4	1	0	1	0	1
V5	1	0	1	1	0

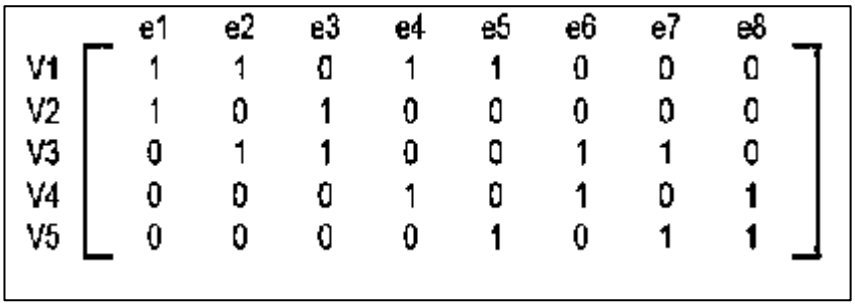
Gambar 2. 7 Matriks Ketetanggaan

2. Matriks Bersisian

Matriks bersisian didefinisikan sebagai berikut, misalkan $B = (b_{ij})$ adalah matriks $m \times n$ yang didefinisikan oleh,

$$b_{ij} = \begin{cases} 1, simpul \; v_i \; bersisian \; dengan \; sisi \; e_j \\ 0, lainnya \end{cases}$$

Perhatikan Gambar 2.6 menunjukkan *graph* berarah yang terdiri dari 5 simpul dan 8 sisi serta Gambar 2.8 yang menunjukkan matriks bersisiannya.



	e1	e2	e3	e4	e5	e6	e7	e8
V1	1	1	0	1	1	0	0	0
V2	1	0	1	0	0	0	0	0
V3	0	1	1	0	0	1	1	0
V4	0	0	0	1	0	1	0	1
V5	0	0	0	0	1	0	1	1

Gambar 2. 8 Matriks Bersisian

2.2 Optimisasi

Optimisasi adalah suatu proses untuk mencapai hasil yang optimal (nilai efektif yang dapat dicapai). Dalam disiplin matematika optimisasi merujuk pada studi permasalahan yang mencoba untuk mencari nilai minimal atau maksimal dari suatu fungsi riil. Untuk dapat mencapai nilai optimal baik minimal atau maksimal tersebut, secara sistematis dilakukan pemilihan nilai variabel integer atau riil yang akan memberikan solusi optimal (Wardy, 2007).

Di dalam kontek matematika, optimisasi ini bisa dinyatakan sebagai suatu usaha sistematis untuk mencari nilai minimum atau maksimum dari suatu fungsi. Fungsi ini secara sederhana dapat dinyatakan dengan:

$$min \,/\, max\, f(x)$$

sebagai contoh adalah fungsi kuadrat $f(x) = x^2$, dimana x anggota bilangan riil ($x \in R$), di dalam contoh ini, $f(x) = x^2$ merupakan fungsi tujuaannya,

sedangkan x adalah daerah asal yang didefinisikan sebagai anggota bilangan riil.

2.2.1 Definisi Nilai Optimal

Menurut Wardy (2007), nilai optimal adalah nilai yang didapat melalui suatu proses dan dianggap menjadi solusi jawaban yang paling baik dari semua solusi yang ada.

2.2.2 Macam-macan Permasalahan Optimisasi

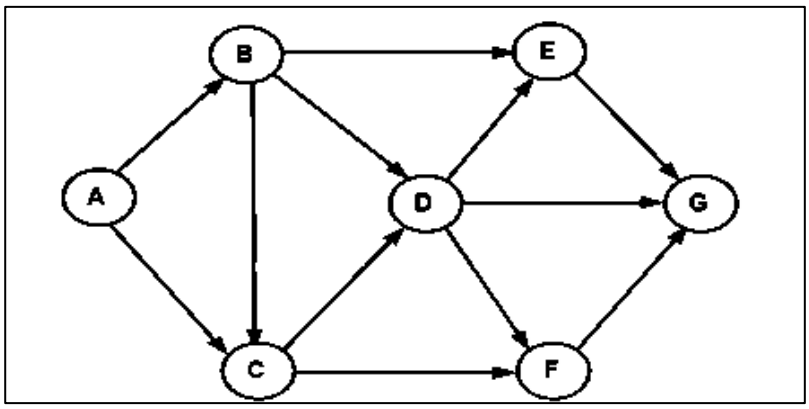
Permasalahan yang berkaitan dengan optimisasi sangat kompleks dalam kehidupan sehari-hari. Nilai optimal yang didapat dalam optimisasi dapat berupa besaran panjang, waktu, jarak, dan lain-lain. Berikut ini adalah termasuk beberapa persoalan optimisasi:

1. Menentukan lintasan terpendek dari suatu tempat ke tempat yang lain.
2. Menentukan jumlah pekerja seminimal mungkin untuk melakukan suatu proses produksi agar pengeluaran biaya pekerja dapat diminimalkan dan hasil produksi tetap maksimal.
3. Mengatur jalur kendaraan umum agar semua lokasi dapat dijangkau.
4. Mengatur *routing* jaringan kabel telepon agar biaya pemasangan kabel tidak terlalu besar dan penggunaannya tidak boros.

Selain berbagai contoh di atas, masih banyak persoalan lainnya yang terdapat dalam berbagai bidang.

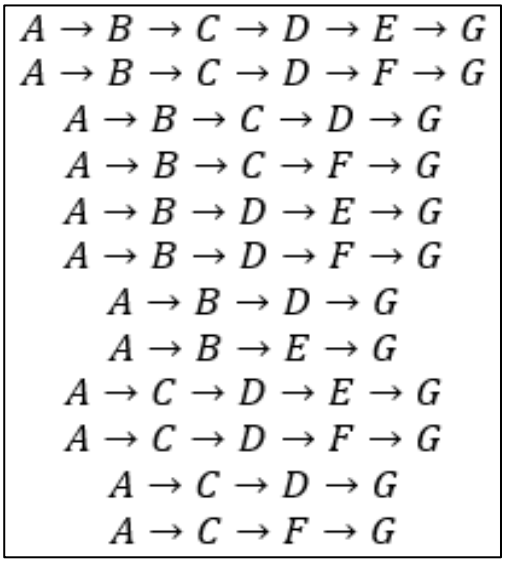
2.2.3 Permasalahan Lintasan Terpendek

Jalur terpendek merupakan suatu pencarian nilai variabel yang dianggap dapat menghasilkan nilai yang maksimal. Jalur terpendek memiliki peranan penting dalam penyusunan *system*. Dengan jalur terpendek dapat diperoleh hal-hal yang memiliki nilai *profit* tinggi serta meminimalkan jarak. Banyak masalah yang berhubungan dengan pencarian jalur. Berbagai pendekatan algoritma yang ditawarkan untuk mendapatkan solusi untuk pencarian jalur terpendek ini, salah satunya yaitu Algoritma ACO. Masalah jalur terpendek merupakan masalah yang berkaitan dengan penentuan *edge-edge* dalam sebuah jaringan yang membentuk rute terdekat antara sumber dan tujuan. Tujuan dari permasalahan jalur terpendek adalah mencari jalur yang memiliki jarak terdekat antara titik asal dan titik tujuan. Gambar 2.9 merupakan suatu jalur titik ABCDEFG.



Gambar 2. 9 Jalur Titik ABCDEFG

Pada kasus Gambar 2.6 dimisalkan rute yang di ambil adalah dari kota A ingin menuju Kota G. Untuk menuju kota G, dapat dipilih beberapa rute yang tersedia sebagai berikut:



Gambar 2. 10 Rute Jalur Titik ABCDEFG

Berdasarkan beberapa rute di atas, dapat dihitung rute terpendek dengan mencari jarak antara rute-rute tersebut. Apabila jarak antar rute belum diketahui, jarak dapat dihitung berdasarkan koordinat kota-kota tersebut, kemudian menghitung jarak terpendek yang dapat dilalui.

2.2.4 Penyelesaian Masalah Optimisasi

Secara umum, penyelesaian masalah pencarian rute terpendek dapat dilakukan dengan menggunakan dua metode, yaitu metode konvensional dan metode heuristik. Metode konvensional dihitung dengan perhitungan matematis biasa, sedangkan metode heuristik dihitung dengan menggunakan sistem pendekatan.

1. Metode Konvensional

   Metode konvensional adalah metode yang menggunakan perhitungan matematika eksak. Ada beberapa metode konvensional yang biasa digunakan untuk melakukan pencarian

rute terpendek, diantaranya: algoritma Djikstra, algoritma *Floyd-Warshall*, dan algoritma *Bellman- Ford* (Mutakhiroh *et al.*, 2007).

2. Metode Heuristik

   Metode Heuristik adalah suatu metode yang menggunakan sistem pendekatan dalam melakukan pencarian dalam optimasi. Ada beberapa algoritma pada metode heuristik yang biasa digunakan dalam permasalahan optimasi, diantaranya Algoritma Genetika, *Ant Colony Optimization* (ACO), logika *Fuzzy*, jaringan syaraf tiruan, *Tabu Search*, *Simulated Annealing*, dan lain- lain (Mutakhiroh *et al.*, 2007).

2.3 Distribusi

Distribusi pada dasarnya berarti pengiriman atau penyampaian suatu barang atau produk kepada orang atau beberapa orang, ataupun pada suatu atau beberapa tempat (Mangkunegara, 2001). Distribusi dianggap penting dikarenakan dengan adanya distribusi yang tepat, maka proses pengiriman atau penyampaian barang dapat dilakukan dengan lancar. Adanya kelancaran dalam distribusi memberikan kemampuan saing yang baik terhadap perusahaan yang melakukan distribusi.

Suatu perusahaan melakukan distribusi pastinya dikarenakan adanya tujuan yang ingin dicapai. Menurut Tjiptono dan Chandra (2008), tujuan distribusi dikelompokkan menjadi empat jenis:

1. Tujuan *account development*, yaitu tujuan yang dimaksudkan dalam penekanan terhadap penambahan atau peningkatan jumlah distributor ataupun pelanggan baru.
2. Tujuan *distributor support*, yaitu tujuan yang dimaksudkan dalam upaya menjalin kerjasama antara perusahaan dengan para distributor grosir maupun eceran guna menetapkan strategi pemasaran.
3. Tujuan *account-maintenance*, yaitu tujuan yang dimaksudkan dalam mempertahankan posisi atau kondisi penjualan yang efektif. Hal ini dilakukan misalnya dengan kunjungan penjualan reguler guna menyediakan informasi tentang produk baru, pengumpulan informasi tentang perubahan kebutuhan pelanggan maupun distributor serta melakukan kegiatan pelayanan terhadap pelanggan.
4. Tujuan *account-penetration*, yaitu tujuan yang dimaksudkan dalam meningkatkan jumlah penjualan total ataupun produk-produk yang dianggap lebih memberikan keuntungan serta produk komplementer lainnya pada distributor atau konsumen saat ini.

2.4 *Traveling Salesman Problem* (TSP)

Hoffman dan Wolfe (dalam Paillin & Tupan, 2018), mendefinisikan TSP sebagai berikut: *TSP is one which has commanded much attention of mathematicians and computer scientists specifically because it is so easy to describe and so difficult to solve. The problem can simply be stated as: if a Traveling salesman wishes to visit exactly once each of a list of m cities (where the cost of Traveling from city i to city j is C_{ij}) and then return to the*

home city, what is the least costly route the Traveling salesman can take?" Berdasarkan definisi di atas, *Traveling salesman problem* merupakan suatu masalah yang mudah dideskripsikan namun sulit untuk diselesaikan, yaitu masalah bagaimana menentukan jarak terpendek dalam perjalanan melewati titik-titik tertentu di mana satu titik harus dilalui satu kali dan hanya boleh dilalui satu kali saja dan perjanalan harus berakhir dengan kembali ke titik pertama. Dalam hal ini titik-titik tersebut merupakan kota-kota dalam suatu wilayah tertentu.

2.4.1 Karakteristik *Travelling Salesman Problem*

Secara singkat karakteristik dari permasalahan TSP adalah:

1. Perjalanan berawal dan berakhir dari dan ke kota awal
2. Ada sejumlah kota yang semuanya harus dikunjungi tepat satu kali
3. Perjalanan tidak boleh kembali ke kota awal sebelum semua kota tujuan dikunjungi
4. Tujuan dari permasalahan ini adalah meminimumkan total jarak yang ditempuh *salesman* dengan mengatur urut-urutan kota yang harus dikunjung

2.4.2 Jenis-jenis *Travelling Salesman Problem*

Menurut Dian (2013), *Travelling Salesman Problem* dikatakan ada dua jenis, yaitu:

1. *Travelling Salesman Problem* Asimetris

   Pada *Travelling Salesman Problem* jenis ini, biaya dari kota 1 ke kota 2 tidak sama dengan biaya dari kota 2 ke kota 1. Dengan n

kota, besarnya ruang pencarian adalah $\frac{n!}{n} = (n-1)!$ jalur yang mungkin.

2. *Travelling Salesman Problem* Simetris

Sedangkan pada *Travelling Salesman Problem* jenis simetris, biaya dari kota 1 ke kota 2 adalah sama dengan biaya dari kota 2 ke kota 1. Apabila dengan n kota, jumlah jalur yang mungkin adalah $\frac{n!}{2n} = \frac{(n-1)!}{2}$ jalur yang mungkin.

2.4.3 Persamaan *Travelling Salesman Problem*

Travelling Salesman Problem dapat dituliskan dalam model matematika sebagai berikut (Davendra, 2010):

$$min \sum d_{ij} x_{ij}$$

$\sum_{j=1}^{n} x_{ij} = 1 \qquad i = 1,2,\dots,n$

$\sum_{i=1}^{n} x_{ij} = 1 \qquad j = 1,2,\dots,n$

$x_{ij} \in \{0,1\} \qquad i,j = 1,2,\dots,n \qquad i \neq j$

Dimana:

d_{ij} = jarak antara titik i dan j

x_{ij} = perpindahan dari titik i menuju titik j.

Bernilai 1 apabila terjadi perpindahan, bernilai 0 apabila tidak terjadi perpindahan.

2.4.4 Algoritma dalam *Travelling Salesman Problem*

Menurut David Bolton (dalam Delima, 2020), algoritma adalah gambaran dari suatu langkah-langkah yang memperoleh suatu

keberhasilan dari sebuah hasil. Menurut Donald E. Knuth (dalam Delima, 2020), ada 5 ciri-ciri yang sangat penting, yaitu:

1. Keterbatasan (*Finiteness*)

   Keterbatasan yaitu suatu algoritma yang akan selesai jika sudah membuat beberapa proses.

2. Kepastian (*Definiteness*)

   Kepastian yaitu semua cara algoritma harus di jelaskan dengan tepat dan bukan berarti ganda.

3. Masukan (*Input*)

   Algoritma mempunyai nilai nol atau lebih beberapa data masukkan (input)

4. Keluaran (*Output*)

   Algoritma mempunnyai nilai nol atau lebih beberapa data hasil keluaran (output).

5. Efektivitas (*Effectiveness*)

   Langkah-langkah algorithma harus efektif dan dikerjakan dalam waktu yang wajar.

Untuk menyelesaikan permasalahan *Travelling Salesman Problem* (TSP) diperlu pendekatan algoritma untuk memperoleh hasil yang efisien dan efektif. Berikut adalah algoritma-algoritma yang dapat digunakan untuk penyelesaian *Travelling Salesman Problem* (TSP):

1. Algoritma *Greedy*

   Algoritma *greedy* merupakan sebuah algoritma yang dapat menentukan sebuah jalur terpendek antara *node-node* yang akan digunakan dengan mengambil secara terus menerus dan menambahkannya ke dalam jalur yang akan dilewati. Mengacu pada konsep *greedy* yang menganggap bahwa pada setiap langkah akan dipilih tempat atau kota yang belum pernah dikunjungi, dimana tempat atau kota tersebut memiliki jarak terdekat dari tempat atau kota sebelumnya. Algoritma ini tidak mempertimbangkan nilai *heuristic*, yang dalam hal ini bisa berupa jarak langsung antar dua tempat.

   Sehingga dengan kata lain, dapat dikatakan bahwa langkah dari algoritma *greedy* ini adalah mengambil pilihan yang terbaik yang dapat diperoleh pada saat itu tanpa memperhatikan konsekuensi ke depan, atau dengan prinsip "*take what you can get now*", berharap bahwa dengan memilih optimum lokal pada setiap langkah akan berakhir dengan optimum global. Dengan prinsip seperti ini dapat dikatakan bahwa algoritma *greedy* lebih berguna untuk menghasilkan solusi hampiran (*approximation*). Hal ini dikarenakan algoritma *greedy* tidak selalu berhasil memberikan solusi yang optimal (Wiyanti, 2013).

2. *Algoritma Artificial Bee Colony*

Pada algoritma ABC, pendekatan yang dilakukan adalah *population-based metaheuristic*, dimana pendekatan ini terinspirasi oleh perilaku cerdas kawanan lebah madu dalam mencari makanan (Wiyanti, 2013). Ada 3 tahapan utama pada basic algoritma ABC, yaitu:

a. Menghasilkan inisial solusi dari sumber makanan secara acak. Untuk memperbarui solusi yang mungkin, setiap *employed bee* memilih calon posisi sumber makanan baru, yang mana posisi tersebut berbeda dengan sebelumnya.

b. Setiap *onlooker bee* memilih salah satu sumber makanan yang diperoleh dari *employed bee*. Setelah memilih sumber makanan, *onlooker bee* pergi ke sumber makanan yang dipilih dan memilih sumber calon makanan baru.

c. Terdapat limit yang telah ditetapkan. Pada tahapan terakhir, limit adalah batasan yang telah ditetapkan dalam siklus algoritma ABC dan mengendalikan banyaknya solusi tertentu yang tidak diperbarui. Setiap sumber makanan yang tidak meningkat melewati limit akan ditinggalkan dan diganti dengan posisi baru dan *employed bee* menjadi *scout bee*.

3. Algoritma *Cheapest Insertion Heuristics* (CIH)

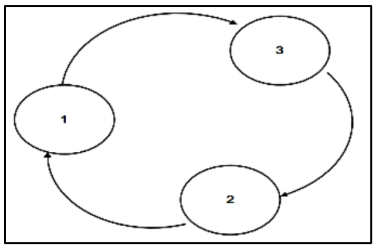
Menurut (Wiyanti, 2013), konsep CIH memiliki algoritma sebagai berikut:

a. Penelusuran

Dimulai dari sebuah kota pertama yang dihubungkan dengan sebuah kota terakhir.

b. Dibuat hubungan *subtour*

Sebuah hubungan *subtour* dibuat antara 2 kota tersebut. Yang dimaksud *subtour* adalah perjalanan dari kota pertama dan berakhir di kota pertama, misal (1,3) → (3,2) → (2,1) seperti tergambar dalam gambar di bawah ini.



Gambar 2. 11 *Subtour*

c. Mengganti arah hubungan

Salah satu arah hubungan (*arc*) dari dua kota diganti dengan kombinasi dua *arc*, yaitu *arc* (i,j) dengan *arc* (i,k) dan *arc* (k,j), dengan k diambil dari kota yang belum masuk *subtour*, dan dengan tambahan jarak terkecil.

Yang mana jarak diperoleh dari

$$c_{ik} + c_{kj} - c_{ij}$$

dimana:

c_{ik} adalah jarak dari kota i ke kota k

c_{kj} adalah jarak dari kota k ke kota j

c_{ij} adalah jarak dari kota i ke kota j

d. Ulangi langkah 3 sampai seluruh kota masuk dalam *subtour*.

4. Algoritma Genetika

Kasus TSP dengan jumlah kota yang banyak, algoritma genetika dapat menghasilkan rute paling optimum. Namun yang perlu diingat adalah pemilihan parameter input harus dilakukan dengan tepat. Konsep algoritma genetika sendiri adalah algoritma pencarian heuristik yang didasarkan pada mekanisme evolusi biologis. Keberagaman pada evolusi biologis adalah variasi dari kromosom dalam individu organisme. Variasi kromosom ini akan mempengaruhi laju reproduksi dan tingkat kemampuan organisme untuk tetap hidupn (Wiyanti, 2013). Adapun langkah-langkah dari algoritma genetika adalah sebagai berikut:

a. Langkah pertama adalah melakukan penentuan nilai awal (inisialisasi). Bagian ini merupakan pemberian input yang dilakukan oleh pengguna.

b. Proses berikutnya adalah proses pembentukan populasi awal. Proses ini berfungsi berfungsi untuk membentuk populasi generasi pertama.

c. Selanjutnya adalah proses seleksi, dimana setelah terbentuk populasi awal, maka hasil populasi awal itu akan diseleksi.
d. Setelah melakukan proses seleksi, maka hasil dari proses tersebut akan digunakan dalam proses *crossover*.
e. Sebelum melakukan proses *crossover* dilakukan pengundian dengan bilangan *random* untuk setiap kromosom, apakah kromosom tersebut terjadi *crossover* atau tidak.
f. Jika dari proses pengundian menunjukkan bahwa terjadi *crossover* maka akan dibuat bilangan random lain untuk menentukan dimana *crossover* akan terjadi.
g. Setelah proses *crossover* dijalankan selalu dilakukan pengecekan apakah kromosom yang terkena *crossover* tersebut merupakan kromosom yang valid, dalam arti kromosom hasil *crossover* tersebut membentuk suatu rute.
h. Jika terjadi pengulangan individu dalam kromosom, maka dilakukan proses normalisasi untuk membuat kromosom tersebut menjadi valid.
i. Pada saat penyalinan kromosom dilakukan, kromosom tersebut dapat mengalami mutasi, yaitu perubahan isi kromosom, dimana isi dari kromosom tersebut digantikan dengan suatu nilai yang dipilih secara acak dari titik-titik yang ada.

j. Setelah proses mutasi, maka akan dilakukan lagi proses seleksi dan dilakukan pengecekan apakah proses keseluruhan telah selesai atau belum.

5. Algoritma *Ant Colony Optimization* (ACO)

Algoritma *Ant Colony Optimization* bekerja sebagai berikut; setiap semut memulai turnya melalui sebuah kota yang dipilih secara acak (setiap semut memiliki kota awal yang berbeda). Secara berulang kali, satu-persatu kota yang ada dikunjungi oleh semut dengan tujuan untuk menghasilkan tur yang lengkap (yaitu mengunjungi masing-masing kota sekali saja). Semut lebih suka untuk bergerak menuju ke kota- kota yang dihubungkan dengan sisi yang pendek atau memiliki tingkat feromon yang tinggi. Setiap semut memiliki sebuah memori, dinamai daftar semut, yang berisi semua kota yang telah dikunjunginya pada setiap tur. Daftar semut ini mencegah semut untuk mengunjungi kota-kota yang sebelumnya telah dikunjungi selama tur tersebut berlangsung.

Setelah semua semut menyelesaikan tur mereka dan daftar semut menjadi penuh, sebuah aturan pembaruan feromon dilaksanakan pada setiap semut. Penguapan feromon pada semua sisi dilakukan, dan kemudian setiap semut menghitung panjang tur yang telah mereka lakukan lalu menaruh sejumlah feromon pada sisi-sisi yang merupakan bagian dari tur mereka yang sebanding dengan kualitas dari solusi yang mereka hasilkan. Semakin pendek sebuah tur yang

dihasilkan oleh seekor semut, jumlah feromon yang diletakkan pada sisi-sisi yang dilaluinya pun semakin besar, dengan demikian sisi yang merupakan bagian dari tur-tur yang pendek adalah sisi-sisi yang menerima jumlah feromon yang lebih besar. Hal ini menyebabkan sisi yang diberi feromon lebih banyak akan lebih diminati/dipertimbangkan pada tur-tur selanjutnya, dan sebaliknya sisi-sisi yang tidak diberi feromon menjadi kurang diminati. Dan juga, jalur terpendek yang ditemukan oleh semut disimpan dan semua daftar semut dikosongkan kembali (Refianti, 2005).

Kelebihan dan kekurangan dari penyelesaian TSP menggunakan algoritma *greedy*, *Artificial Bee Colony* (ABC), *Cheapest Insertion Heuristics* (CIH), Genetika dan *Ant Colony Optimization* (ACO) adalah sebagai berikut.

Tabel 2. 1 Perbandingan Algoritma Penyelesaian TSP

Algoritma	Kelebihan	Kekurangan
Greedy	Waktu komputasi yang dibutuhkan dalam menyelesaikan kasus TSP lebih cepat. Lebih sesuai untuk kasus yang membutuhkan solusi hampiran.	Hasil yang didapatkan tidak selalu optimal. Hal ini karena algoritma greedy masih terjebak dalam optimum lokal.
Artificial Bee Colony (ABC)	Mencapai nilai optimal apabila data pada kasus TSP merupakan data dengan *size* yang tidak terlalu besar.	Kesalahan atau akurasinya semakin besar seiring dengan data *size* yang besar pula. Sehingga algoritma ini kurang cocok untuk kasus TSP dengan jumlah kota yang besar dan jarak yang terlalu lebar
Cheapest Insertion Heuristics (CIH)	Algoritma ini masih stabil digunakan untuk kasus TSP dengan jumlah kota yang besar.	Prinsip pencariannya lebih rumit dan lebih lama karena memungkinkan terjadinya perulangan perhitungan tambahan jarak (probabilitas *edge* yang akan digantikan) pada iterasi yang berbeda

Lanjutan Tabel 2. 1 **Perbandingan Algoritma Penyelesaian TSP**

Algoritma	Kelebihan	Kekurangan
Genetika	Waktu komputasi yang dibutuhkan cenderung stabil. Mampu memberikan jarak terpendek meski dengan jumlah kota yang besar	Sangat bergantung pada pemilihan parameter input, yaitu ukuran populasi, besar maksimum generasi, ukuran peluang *crossover*, dan ukuran peluang *mutation* Hasil yang di peroleh bisa saja lebih panjang dari jarak sebenarnya.
Ant Colony Optimization (ACO)	Algoritma ini selalu menemukan solusi yang mendekati optimal untuk semua permasalahan yang mempunyai jumlah titik sedikit. Mampu memberikan nilai dengan solusi tunggal untuk beberapa kali pengujian	Kompleksitas yang cukup banyak sehingga *running time*-nya juga cukup lama karena ada beberapa proses tahapan yang agak rumit untuk dipecahkan secara matematis biasa dan dibutuhkan bantuan *software*. Dan proses *running* program ACO boros dalam penggunaan *memory*

2.5 *Ant Colony Optimization* (ACO)

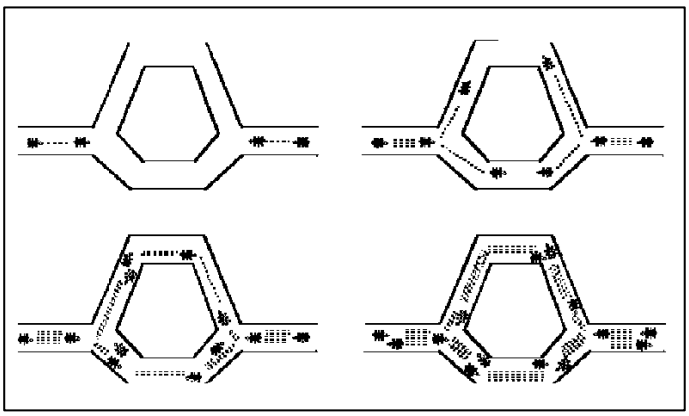
Seiring berkembangnya pemikiran, ditemukan sejumlah algoritma dalam AI (*Artifical Intelligence*) yang mendapat inspirasi dari alam. Algoritma tersebut di antaranya adalah: cara kerja otak yang luar biasa mengilhami *neural network*, *genetic algorithm* belajar dari proses evolusi, dan dari semut menyelesaikan masalah optimisasi (Helene *et al.*, 2008). Pada tahun 1996, dunia AI diperkenalkan dengan algoritma Semut oleh Moyson dan Manderick dan secara meluas dikembangkan oleh Marco Dorigo, merupakan algoritma yang terinspirasi oleh perilaku semut dalam menemukan jalur dari sarangnya menuju makanan (Fikri, 2010).

Semut adalah serangga sosial yang hidupnya berkoloni, dapat bekerja sama dengan sesamanya dalam melakukan pekerjaan secara efektif. Perilaku semut dalam menemukan makanan dari sarangnya menghasilkan jalur yang optimal dengan menemukan jalur terpendek (Helene *et al.*, 2008). Semut juga

mampu mengindera lingkungannya yang kompleks untuk mencari makanan dan kemudian kembali ke sarangnya dengan meninggalkan zat feromon pada jalur-jalur yang mereka lalui (Alfriany, 2008). Feromon adalah zat kimia yang berasal dari kelenjar endokrin dan digunakan oleh makhluk hidup untuk mengenali sesama jenis, individu lain, kelompok, dan untuk membantu proses reproduksi. Proses peninggalan feromon ini dikenal sebagai stigmergy, yaitu sebuah proses memodifikasi lingkungan yang tidak hanya bertujuan untuk mengingat jalan pulang ke sarang, tetapi juga memungkinkan para semut berkomunikasi dengan sesamanya (Fikri, 2010). Seiring waktu, bagaimanapun juga jejak feromon akan menguap dan akan mengurangi kekuatan daya tariknya. Lebih lama seekor semut pulang pergi melalui jalur tersebut, lebih lama jugalah feromon menguap (Refianti, 2005). Agar semut mendapatkan jalur optimal dalam perjalanannya, diperlukan beberapa proses:

1. Pada awalnya, semut berkeliling secara acak, hingga menemukan makanan.
2. Ketika menemukan makanan mereka kembali ke sarangnya sambil memberikan tanda dengan jejak feromon.
3. Jika semut-semut lain menemukan jalur tersebut, maka mereka tidak akan bepergian dengan acak lagi, melainkan akan mengikuti jejak tersebut.
4. Jika pada akhirnya mereka pun menemukan makanan, maka mereka kembali dan menguatkan jejaknya.

5. Feromon yang berkonsentrasi tinggi pada akhirnya akan menarik semut-semut lain untuk berpindah jalur, menuju jalur paling optimal, sedangkan jalur lainnya akan ditinggalkan. Gambar 2.12 menujukkan perjalanan semut dalam menemukan jalur terpendek dari sarang ke sumber makanan.



Gambar 2. 12 Perjalanan Semut Menentukan Rute Makanan

2.6 Pemrograman

Menurut Sunarto (dalam Delima, 2020), mengatakan bahwa program beberapa kumpulan arahan yang akan dicapai baik itu dalam kode, bagan dan bahasa. Jika beberapa arahan tersebut dikumpulkan atau di gabungkan menjadi satu menggunakan media dan dapat di baca oleh sistem komputer maka itu akan membuat komputer dapat bekerja untuk melakukan suatu fungsi dalam sebuah persiapan merancang instruksi-instruksi tersebut.

Program merupakan "*source code*" biasa dibuat oleh seorang *programmer* yaitu sekumpulan arahan-arahan atau instruksi yang tersendiri. Program adalah realisasi dari algoritma atau dapat dijelaskan dengan persamaann berikut.

$$Program = algoritma + bahasa$$

2.6.1 Cara dalam Membuat Program

Dalam membuat sebuah program, terdapat langkah-langkahnya, sebagai berikut.

1. Menjelaskan suatu permasalahan
    a. Kondisi awal, yaitu input yang tersedia
    b. Kondisi akhir, yaitu output yang diinginkan
    c. Data lain yang tersedia
    d. Operator yang tersedia
    e. Syarat atau kendala yang harus dipenuhi.
2. Struktur cara penyelesaian dalam membuat algoritma

    Dalam penyelesaikan masalah jika permasalahannya rumit maka akan di bagi kedalam beberapa buku.
3. Menulis Program
    a. Menggunakan bahasa yang mudah di pahami dan di pelajari
    b. Pilihlah yang mudah digunakan dan sudah dikuasai penulisannya
4. Dalam menggunakan perangkat keras dan beberapa media lainnya harus memiliki tingkat kompatibilitas yang tinggi.
5. Mencari kesalahan
    a. Kesalahan sintaks, yaitu kesalahan dalam penulisan program.
    b. Kesalahan pelaksanaan terdiri dari *semantic*, akal dan ketelitian.

6. Menguji program terlebih dahulu lalu melakukan verifikasi program
7. Membuat dokumentasi sebuah program
8. Membantu pemeliharaan program

2.6.2 Bahasa Pemrograman

Bahasa pemrograman dapat digolongkan menjadi dua kelompok berdasarkan terapannya yaitu:

1. Bahasa Pemrograman bertujuan untuk khusus yaitu:
   a. *Cobol* yang digunakan untuk menerapkan bisnis dan administrasi
   b. *Fortan* berfungsi untuk menerapkan komputasi ilmiah
   c. *Assembly* memiliki tujuan untuk menerapkan pemrograman mesin
   d. *Prolog* untuk menerapkan kecerdasan buatan dan bahasa-bahasa simulasi
   e. *High-level matrix/array language* bertujuan melakukan komputasi numerik berbasis matriks atau tabel
2. Bahasa pemrograman mempunyai tujuan umum yang bisa digunakan atau di manfaatkan oleh berbagai aplikasi. Aplikasi yang dimaksud adalah *basic*, bahasa *pascal*, java dan C. Karena memiliki tujuan khusus ini bukan tidak bisa digunakan oleh aplikasi lain tetapi bisa digunakan oleh aplikasi lain. Misalnya *cobol* bisa digunakan untuk terapan ilmiah, tetapi kemampuannya

terbatas. Bahasa pemrograman dibuat dengan bermaca-macam untuk terapan yang berbeda-beda.

Pada bahasa pemrograman lebih mendekati bahasa mesin atau bahasa manusia untuk tingkat kerumitannya (Delima, 2020). Dalam bahasa pemrograman terbagi menjadi dua bagian utama, yaitu:

1. Bahasa tingkat tingkat tinggi yaitu bahasa pemrograman yang tidak dapat dikatakan lebih baik di bandingkan bahasa tingkat rendah. Bahasa pemrograman tingkat tinggi berurusan dengan ekspresi aritmatika atau aljabar Boolean. Bahasa tingkat tinggi tidak bisa secara langsung membuat bahasa tersebut menjadi dimengerti sebuah mesin jika tidak mempunyai kode operasi. Ada beberapa karakteristik yang terdapat dalam bahasa tingkat tinggi ini yaitu, memiliki kegunaan-kegunaan penanganan *string*, fungsi-fungsi penanganan *string*, pengenalan objek pemrograman, masukan (*input*) atau keluaran (*output*) yang ada dalam berbagai macam bahasa tingkat tinggi ini. Pada umumnya dalam bahasa tingkat tinggi ini pemrograman sistem komputer yang rumit dibuat agar lebih sederhana. Bahasa pemrograman tingkat tinggi berorientasi ke bahasa manusia yaitu bahasa inggris agar pemrograman lebih mudah dipahami atau lebih manusiawi. Tetapi pemrograman yang berupa bahasa inggris dalam bahasa tingkat tinggi biasanya tidak langsung dapat dilakukan oleh komputer. Bahasa tersebut harus di terjemahkan terlebih dahulu dalam sebuah *translator*. *Translator*

bahasa yang disebut dengan kompilator atau *compiler*. Bahasa tangkat tinggi ini harus di terjemahkan terlebih dahulu ke dalam bahasa mesin sebelum pada akhirnya akan di eksekusi oleh sistem CPU.

2. Bahasa tingkat rendah, yaitu bahasa yang harus berurusan dengan *register*, *stack-stack* pangggilan, dan alat memori. Bahasa tingkat rendah lebih cenderung menbuat kode yang lebih efisien. Elemen-elemen yang kompleks dapat dipecah menjadi berberapa elemen yang lebih sederhana dimana bahasa tersebut menyediakan suatu abstraksi yang masih dianggap kompleks dalam suatu bahasa tingkat tinggi. Maka dari itu karena alasan ini, kode yang ditulis dalam bahasa pemrograman tingkat rendah ini bisa berjalamn dengan lebih efektif dan efisien. Bahasa tingkat rendah berorientasi sangat dekat dengan mesin dan sangat sulit di mengerti oleh orang dan juga bersifat sangat sederhana. Bahasa yang dimasukkan kedalam bahasa tingkat rendah ini adalah bahasa *Assembly* karena menotasi yang dipakai pada bahasa ini lebih dekat kepada mesin. Untuk melaksanakan instruksinya masih perlu terjemahan ke dalam bahasa mesin.

2.6.3 *Software Programming* untuk Algoritma *Travelling Salesman Problem*

Untuk membuat program penyelesaian TSP dibutuhkan algoritma dan bahasa pemrogramannya. Program TSP ini dapat berjalan dalam perangkat lunak (*software*) programming. Software programming yang

dapat digunakan untuk penyelesaian *Travelling Salesman Problem* adalah sebagai berikut.

1. *Visual Studio Code*

   Visual Studio Code adalah *software* yang sangat ringan, namun kuat editor kode sumbernya yang berjalan dari desktop. Muncul dengan *built-in* dukungan untuk *JavaScript*, naskah dan *Node.js* dan memiliki *array* beragam ekstensi yang tersedia untuk bahasa lain, termasuk C ++, C # , *Python*, dan PHP (Robitoh, 2018). Banyak sekali fitur-fitur yang disediakan oleh *Visual Studio Code*, diantaranya *Intellisense*, *Git Integration*, *Debugging*, dan fitur ekstensi yang menambah kemampuan teks editor. Fitur-fitur tersebut akan terus bertambah seiring dengan bertambahnya versi *Visual Studio Code*. Pembaruan versi *Visual Studio Code* ini juga dilakukan berkala setiap bulan, dan inilah yang membedakan VS *Code* dengan teks editor yang lain (Permana & Romadlon, 2019).

2. *Matrix Laboratory* (MATLAB)

   Matlab adalah singkatan dari *Matrix Laboratory* (Laboratorium Matriks) dan merupakan bahasa pemrograman yang dibuat dengan tujuan sebagai alat bantu perhitungan yang rumit atau simulasi dari suatu sistem yang ingin di simulasikan, dalam matlab mutlak dibutuhkan pengetahuan tentang matriks yang dapat dipelajari dalam ilmu matematika (Noviansyah, 2019).

Kelebihan dan kekurangan dari penggunaan software *Visual Studio Code* dan MATLAB adalah sebagai berikut.

Tabel 2. 2 Perbedaan *Software* Penyelesaian TSP

Software	**Kelebihan**	**Kekurangan**
Visual Studio Code	Bersifat *open source* dan gratis. Memiliki banyak *extensions* serta mudah mengelolanya. VS *code* sudah mendukung banyak bahasa pemrograman seperti C, C++, PHP, *javascript* dan masih banyak yang lainnya	Performa untuk menjalankan harus memiliki spesifikasi komputer yang cukup tinggi. *Shortcut key* yang cukup berbeda dengan *software* lainnya. Tidak terkhusus dalam pengerjaan matriks
Matrix Laboratory (MATLAB)	MATLAB sangat handal untuk komputasi kompleks yang terkait dengan *array* atau matriks. Kehandalan ini bisa terlihat mulai dari proses *assignment* variabel terhadap nilai bertipe *array* atau matriks yang sederhana, sampai dengan operasi perhitungannya yang cepat. Kesederhanaan dalam proses *assignment* ini menyebabkan tidak diperlukannya pendefinisian ukuran (*size*) *array* atau matriks pada variabel tersebut.	Software ini cukup kompleks dalam menggunakannya. Bahasa pemrogramannya kurang familiar diantara pengguna *software* pemrograman pada umumnya. MATLAB merupakan sebuah program berbayar, sehingga untuk bisa menggunakan program ini secara *full*, membutuhkan biaya yang tidak sedikit.

2.7 MATLAB

Matlab adalah suatu *software* pemrograman perhitungan dan analisis yang banyak digunakan dalam semua area penerapan matematika baik bidang pendidikan maupun penelitian pada universitas dan industri. Dengan matlab, maka perhitungan matematis yang rumit dapat diimplementasikan dalam program dengan lebih mudah (Megalina, 2010)

MATLAB (*Matrix Laboratory*) adalah suatu program untuk analisis dan komputasi numerik dan merupakan suatu bahasa pemrograman matematika lanjutan yang dibentuk dengan dasar pemikiran menggunakan sifat dan bentuk matriks. Pada awalnya, program ini merupakan *interface* untuk

koleksi rutin-rutin *numeric* dari proyek LINPACK dan EISPACK, dan dikembangkan menggunkan bahasa FORTRAN namun sekarang merupakan produk komersial dari perusahaan *Mathworks, Inc*. yang dalam perkembangan selanjutnya dikembangkan menggunakan bahasa C++ dan assembler (utamanya untuk fungsi-fungsi dasar MATLAB) (Cahyono, 2016).

Data dalam MATLAB sama saja dengan bahasa pemrograman yang lain, yaitu ada tipe '*integer'*, *'double'*, *'single*, dan lain-lain. Yang membuat berbeda dengan yang lain adalah struktur data dalam MATLAB. Menurut Noviansyah (2019), struktur data dalam MATLAB yang biasa digunakan adalah:

1. *Array* Multi Dimensi

   Inilah salah satu kelebihan MATLAB dibandingkan dengan bahasa pemrograman lainnya, MATLAB dapat dengan mudah merepresentasikan suatu *array* multidimensi

2. *Array Cell*

   Pada tipe data ini kita dapat mengisi komponen-komponen yang ada pada *array* dengan suatu nilai, atau bahkan suatu matriks, jadi jika kita membutuhkan data yang sangat bervariasi kita dapat menggunakan *array cell* ini. Untuk membuat sebuah *cell* kita harus menggunakan tanda kurung kurawal untuk menandakkan kalau itu adalah sebuah *cell*

3. *Structures*

   Tipe data yang satu ini juga sering digunakan dalam MATLAB. *Structures* adalah tipe data yang sebenarnya adalah multidimensional

array, hanya saja dalam penamaannya menggunakan penandaan dengan suatu *field*.

2.8 Penelitian Terdahulu

Berikut adalah tabel perbandingan antara beberapa penelitian terdahulu yang membahas *Travelling Salesman Problem*.

Tabel 2.3 Penelitian Terdahulu

No	Peneliti	Judul	Metode	Hasil	Perbedaan Penelitian Terdahulu
1	Irfan (2018)	Penyelesaian *Travelling Salesman Problem* (TSP) Menggunakan Algoritma *Hill Climbing* dan MATLAB	Algoritma *Simple Hill Climbing* (SHC) dan *Steepest-Ascent Hill Climbing* (SAHC)	Didapatkan perbandingan hasil rute dari penggunaan algoritma *Simple Hill Climbing* (SHC) dan *Steepest-Ascent Hill Climbing* (SAHC), baik dari segi panjang rute, lama proses perhitungan dan level optimumnya. SAHC lebih efektif dalam menyelesaikan masalah TSP dibandingkan dengan SHC	Penelitian lebih berfokus pada perbandingan hasil dari kedua pembagian algoritma *Hill Climbing* Objek penelitian berbeda
2	Nurlaelasari *et al*. (2018)	Penerapan Algoritma *Ant Colony Optimization* Menentukan Nilai Optimal Dalam Memilih Objek Wisata Berbasis *Android*	*System Development Life Cycle* (SDLC) *Waterfall* dan Algoritma *Ant Colony Optimization*	Hasil dari penelitian ini adalah pembuatan sistem berbasis *Android* untuk penentuan rute objek wisata dengan algoritma *Ant Colony Optimization*. Dasar penentuan rute optimalnya adalah biaya tiap lintasan.	Penelitian lebih berfokus pada pembangunan sistem *Android* Objek penelitian berbeda
3	Sanggala *et al.* (2021)	*Genetic Algorithm* untuk Memperbaiki Rute *Travelling Salesman Problem* Yang Dihasilkan dari *Nearest Neighbour*	Algoritma *Genetic* dan *Nearest Neighbour*	Pada penelitian ini dilakukan uji perbandingan hasil pada 10 *instance*. Diperoleh hasil pada seluruh 10 *instance*, algoritma GA dapat memperbaiki rute yang dihasilkan metode NN.	Penelitian lebih berfokus pada keberhasilan GA dalam memperbaiki rute dari NN Objek penelitian menggunakan *instance* dari internet

Irfan (2018) melakukan penelitian penyelesaian *Traveling Salesman Problem* dengan pendekatan algoritma *Hill Climbing* dimana terdapat 2 pembagiannya yakni *Simple Hill Climbing* (SHC) dan *Steepest-Ascent Hill Climbing* (SAHC) dengan bantuan *software* MATLAB. Studi kasus diambil dari pengiriman surat pada *Chinese Postman*. Pada penelitian diperoleh perbadingan hasil dari SHC dan SAHC, baik dari segi panjang rute, lama proses perhitungan dan level optimumnya. Dari penelitian terdahulu ini, lebih berfokus pada perbandingan hasil dari kedua metode yang digunakan, sedangkan pada penelitian berfokus pada pengoptimalan/perbaikan rute awal. Dari penelitian terdahulu ini diambil acuan bahwa *Travelling Salesman Problem* dapat dibantu penyelesaiannya menggunakan *software programming*, dimana dalam hal ini adalah MATLAB. Sehingga penulis juga melakukan perbaikan rute awal/penyelesaian TSP dengan bantuan *software* MATLAB.

Nurlaelasari *et al*. (2018) melakukan penelitian pembuatan sistem berbasis *Android* yang digunakan untuk menyelesaiakn *Traveling Salesman Problem* pada penentuan rute objek wisata dengan algoritma *Ant Colony Optimization.* Hasil dari penelitian ini adalah terciptanya satu sistem aplikasi, dimana aplikasi ini akan memberikan usulan rute tujuan berikutnya dengan acuan tarif. Dari penelitian terdahulu ini, tidak dapat dikatakan diperoleh nilai yang paling optimum karena keterbatasan sistem yang mengelola dengan parameter tetap dan iterasi yang cukup sedikit. Variabel yang menjadi acuan penentuan rute hanya satu, yakni tarif sedangkan penelitian ini mengambil 3

variabel, yakni jarak, waktu dan biaya. Penelitian ini melakukan penyelesaian TSP guna mengoptimalkan rute awal sedangkan penelitian terdahulu ini membuat rute baru. Dari penelitian terdahulu ini dapat diperoleh acuan bahwa algoritma *Ant Colony Optimization* dapat digunakan untuk menyelesaikan *Travelling Salesman Problem*.

Sanggala *et al.* (2021) melakukan penelitian penyelesaian *Traveling Salesman Problem* dengan melakukan perbaikan hasil rute dari metode *Nearest Neighbour* (NN) dengan metode *Genetic Algorithm* (GA). Pada penelitian ini dilakukan uji perbandingan hasil pada 10 *instance* yang diperoleh dari internet. Diperoleh hasil pada seluruh 10 *instance*, algoritma GA dapat memperbaiki rute yang dihasilkan metode NN. Dari penelitian terdahulu ini memiliki kesamaan dalam hal memperbaiki rute yang sebelumnya sudah ada. Penggunaan metode NN serupa dengan objek penelitian ini, yakni rute awal CSO yang cenderung mencari titik terdekat tanpa memikirkan nilai heuristiknya.

Pada penelitian kali ini memiliki kesamaan pada ketiga penelitian sebelumnya yaitu dengan melakukan penyelesaian *Traveling Salesman Problem* untuk menentukan rute opimal dari beberapa titik kunjungan (*nodes*) dengan menggunakan metode heuristik. Perbedaan pada penelitian sebelumnya yaitu pengambilan data diambil pada CSO (*Cluster Sales Officer)* Indosat Ooredoo Hutchison *Micro Cluster* Mamuju.