PERBANDINGAN OPTIMASI RUTE PADA *TRAVELING SALESMAN PROBLEM* MENGGUNAKAN ALGORITMA *ANT COLONY OPTIMIZATION* DAN ALGORITMA *GREEDY*

SKRIPSI

INDAH PUSPITA SARI

H011201019

PROGRAM STUDI MATEMATIKA

DEPARTEMEN MATEMATIKA

FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM

UNIVERSITAS HASANUDDIN

MAKASSAR

MEI 2024

PERBANDINGAN OPTIMASI RUTE PADA *TRAVELING SALESMAN PROBLEM* MENGGUNAKAN ALGORITMA *ANT COLONY OPTIMIZATION* DAN ALGORITMA *GREEDY*

SKRIPSI

Diajukan sebagai salah satu syarat untuk memperoleh gelar Sarjana Sains pada Program Studi Matematika Departemen Matematika Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Hasanuddin

INDAH PUSPITA SARI

H011201019

PROGRAM STUDI MATEMATIKA

DEPARTEMEN MATEMATIKA

FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM

UNIVERSITAS HASANUDDIN

MAKASSAR

MEI 2024

HALAMAN PERNYATAAN KEOTENTIKAN

Saya yang bertanda tangan di bawah ini menyatakan dengan sungguh-sungguh bahwa skripsi yang buat dengan judul

Perbandingan Optimasi Rute pada *Traveling Salesman Problem* Menggunakan Algoritma *Ant Colony Optimization* dan Algoritma *Greedy*

Adalah benar hasil karya sendiri, bukan hasil plagiat dan belum pernah dipublikasikan dalam bentuk apapun

Makassar, 6 Mei 2024

INDAH PUSPITA SARI
NIM. H011201019

PERBANDINGAN OPTIMASI RUTE PADA *TRAVELING SALESMAN PROBLEM* MENGGUNAKAN ALGORITMA *ANT COLONY OPTIMIZATION* DAN ALGORITMA *GREEDY*

disetujui oleh:

Pembimbing Utama

Dr. Khaeruddin, M.Sc

NIP. 196509141991031003

Pada 6 Mei 2024

HALAMAN PENGESAHAN

PERBANDINGAN OPTIMASI RUTE PADA *TRAVELING SALESMAN PROBLEM* MENGGUNAKAN ALGORITMA *ANT COLONY OPTIMIZATION* DAN ALGORITMA *GREEDY*

Disusun dan diajukan oleh

INDAH PUSPITA SARI

H011201019

Telah dipertahankann di hadapan Panitia Ujian yang dibentuk dalam rangka Penyelesaian Studi Program Sarjana Program Studi Matematika Depatemen Matematika Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Hasanuddin pada Tanggal, 6 Mei 2024 dan dinyatakan telah memenuhi syarat kelulusan

Menyetujui

Pembimbing Utama

Dr. Khaeruddin, M.Sc

NIP. 196509141991031003

Ketua Program Studi

Dr. Firman, S.Si., M.Si.

NIP. 196804292002121001

HALAMAN PENGESAHAN

Skripsi ini diajukan oleh:

Nama : Indah Puspita Sari

NIM : H011201019

Program Studi : Matematika

Judul Skripsi : Perbandingan Optimasi Rute pada *Traveling Salesman Problem* Menggunakan Algoritma *Ant Colony Optimization* dan Algoritma *Greedy*

Telah berhasil dipertahankan di hadapan Dewan Penguji dan diterima sebagai bagian dari persyaratan yang diperlukan untuk memperoleh gelar Sarjana Sains pada Program Studi Matematika Departemen Matematika Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Hasanuddin

DEWAN PENGUJI

1. Ketua : Dr. Khaeruddin, M.Sc. ( )
2. Anggota : Prof. Dr. Amir Kamal Amir, M.Sc. ( )
3. Anggota : Prof. Dr. Moh. Ivan Azis, M.Sc.

Ditetapkan di : Makassar

Tanggal : 6 Mei 2024

KATA PENGANTAR

Alhamdulillah, segala puji bagi Allah SWT yang selalu melimpahkan Rahmat dan Hidayah-Nya kepada penulis sehingga skripsi dengan judul "Perbandingan Optimasi Rute pada *Traveling Salesman Problem* Menggunakan Algoritma *Ant Colony Optimization* dan Algoritma *Greedy*" dapat terselesaikan. Shalawat dan salam semoga senantiasa kita curahkan kepada teladan kita Nabi Muhammad SAW, beserta keluarga dan para sahabat. Skripsi ini adalah salah satu syarat untuk memperoleh gelar Sarjana Sains pada Program Studi Matematika Departemen Matematika Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Hasanuddin.

Penulis mengucapkan terima kasih khususnya kepada kedua orang tua Bapak **Sainuddin** dan Ibu **Karmini** yang telah membesarkan dan mendidik penulis dengan penuh kesabaran serta selalu mencurahkan kasih sayang yang tak pernah putus, memberikan dukungan dan doa sehingga penulis dapat menyelesaikan penulisan skripsi ini. Begitu pula kepada adik penulis **Muhammad Zulkifly** dan kakak penulis **Asriani** yang telah memberikan dukungan pada penulis. Terima kasih telah menjadi sumber motivasi dan kekuatan bagi penulis selama perjalanan akademik ini.

Penulis menyadari bahwa penyelesaian skripsi ini tidak lepas dari dukungan dan bantuan banyak pihak baik secara langsung maupun tidak langsung. Untuk itu pada kesempatan ini penulis mengucapkan terima kasih dan penghargaan yang setinggi-tingginya kepada:

1. Bapak **Prof. Dr. Ir. Jamaluddin Jompa, M.Sc.,** selaku Rektor Universitas Hasanuddin.
2. Bapak **Dr. Eng. Amiruddin, S.Si., M.Si.,** selaku Dekan Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Hasanuddin.
3. Bapak **Dr. Khaeruddin., M.Sc.,** selaku pembimbing untuk segala ilmu, nasihat, dan kesabaran dalam membimbing dan mengarahkan penulis, serta bersedia meluangkan waktunya untuk mendampingi penulis sehingga skripsi ini dapat diselesaikan.

4. Bapak **Prof. Dr. Moh. Ivan Azis, M.Sc.,** selaku penguji, dan Bapak **Prof. Dr. Amir Kamal Amir, M.Sc.,** selaku penguji dan penasihat akademik penulis yang telah bersedia meluangkan waktunya untuk memberikan saran dan arahan kepada penulis dalam penulisan skripsi ini.
5. Bapak/Ibu Dosen Departemen Matematika yang telah membagikan ilmu dan pengalamannya, serta Staf Departemen Matematika atas segala bantuannya.
6. Sahabat-sahabat penulis **Fahira Farensia Ediyanto, Alpiyanti, Nur Atila Ayu,** dan **Gabriella Mega Lulun Bara** yang telah membantu, menemani, menyemangati, dan tempat berbagi keluh-kesah penulis selama kurang lebih 4 tahun perkuliahan.
7. Teman-teman penulis **Asfi Saiva, Hilda Alifatin, Fatmawati, Dian Fadlu Rahman, Mardiana, Afiliani, Eko Sura' Kapuangan, Ilmi Amaliyah Rahim, Putri Andriani, Fadila Alya Putri B, Waode Allysa Salsabila,** dan **Madina Reski Tulhusnah** yang telah membantu dan menjadi teman diskusi penulis selama perkuliahan dan proses penulisan skripsi.
8. **Lee Jeno, Mark Lee, Huang Renjun, Lee Haechan, Na Jaemin, Zhong Chenle, Park Jisung**, serta **seluruh member NCT, aespa, dan RIIZE** yang tidak dapat desebutkan satu persatu yang menjadi sumber inspirasi dan penyemangat yang tak tergantikan selama perjalanan perkuliahan.
9. **Teman-teman Matematika 2020** atas segala dukungan, kebersamaan, dan kerjasamanya selama ini.
10. **Teman-teman KKN G-110 Posko Tompobulu** atas segala dukungan, kebersamaan, serta pengalaman yang diperoleh selama proses KKN hingga saat ini.
11. **Diri sendiri** atas ketekunan, kesabaran, dan kerja keras selama menyelesaikan skripsi ini.
12. Serta segala pihak yang tidak dapat disebutkan satu persatu yang telah membantu penulis dalam menyelesaikan skripsi ini.

Penulis menyadari bahwa skripsi ini masih jauh dari kesempurnaan, kritik dan saran yang membangun diharapkan oleh penulis untuk perbaikan dan pengembangan penelitian lebih lanjut. Akhir kata, semoga hasil penelitian ini dapat bermanfaat bagi kalangan akademisi, praktisi, dan semua pihak.

PERNYATAAN PERSETUJUAN PEBLIKASI TUGAS AKHIR UNTUK KEPENTINGAN AKADEMIS

Sebagai sivitas akademik Universitas Hasanuddin, saya yang bertanda tangan dibawah ini:

Nama	: Indah Puspita Sari
NIM	: H011201019
Program Studi	: Matematika
Departemen	: Matematika
Fakultas	: Matematika dan Ilmu Pengetahuan Alam
Jenis Karya	: Skripsi

demi pengembangan ilmu pengetahuan, menyetujui untuk memberikan kepada Universitas Hasanuddin **Hak Bebas Royalti Noneksklusif (*Non-exclusive Royalty-Free Right*)** atas karya ilmiah saya yang berjudul:

"Perbandingan Optimasi Rute pada *Traveling Salesman Problem* Menggunakan Algoritma *Ant Colony Optimization* dan Algoritma *Greedy*"

Beserta perangkat yang ada (jika diperlukan). Terkait dengan hal diatas, maka dipihak universitas berhak menyimpan, mengalih-meda/format-kan, mengelola dalam bentuk pangkalan data (*database*), merawat, dan mempublikasikan tugas akhir saya selama tetap mncantumkan nama saya sebagai penulist/pencipta dan sebagai pemilik Hak Cipta.

Demikian pernyataan ini saya buat dengan sebenarnya.

Dibuat di Makassar pada 6 Mei 2024

Yang menyatakan

Indah Puspita Sari

ABSTRAK

Traveling Salesman Problem (TSP) merupakan salah satu permasalahan optimasi kombinatorial yang sering muncul dalam merencanakan rute perjalanan. Fokus utama dalam TSP adalah mencari rute terpendek yang melewati sejumlah kota dengan cara mengoptimalkan total jarak yang ditempuh. Dalam penelitian ini, akan dibandingkan dua pendekatan algoritma yang berbeda, yaitu *Ant Colony Optimization* (ACO) dan *Greedy*, dalam menyelesaikan TSP dalam dua kasus, yakni simetris dan asimetris. Kedua algoritma tersebut memiliki struktur kerja yang berbeda, sehingga digunakan bahasa pemrograman Python untuk memeriksa kinerja optimal keduanya. Kedua algoritma ini diterapkan pada lima matriks jarak dengan ukuran yang berbeda-beda. Hasilnya menunjukkan bahwa ACO cenderung memberikan jarak terpendek yang optimal dibandingkan dengan *Greedy*, walaupun *Greedy* memiliki struktur kerja yang lebih sederhana dalam penerapannya.

Kata Kunci: *Traveling Salesman Problem*, *Ant Colony Optimization*, Algoritma *Greedy*, Optimasi Rute, Python

Judul : Perbandingan Optimasi Rute pada *Traveling Salesman Problem* Menggunakan Algoritma *Ant Colony Optimization* dan Algoritma *Greedy*

Nama : Indah Puspita Sari

NIM : H011201019

Program Studi : Matematika

ABSTRACT

Traveling Salesman Problem (TSP) is one of the combinatorial optimization problems that often arise in planning travel routes. The main focus in TSP is to find the shortest route through a number of cities by optimizing the total distance traveled. In this study, two different algorithmic approaches, namely Ant Colony Optimization (ACO) and Greedy, will be compared in solving TSP in two cases, namely symmetric and asymmetric. Both algorithms have different working structures, so the Python programming language is used to check their optimal performance. Both algorithms were applied to five distance matrices of different sizes. The results show that ACO tends to provide the optimal shortest distance compared to Greedy, although Greedy has a simpler working structure in its application.

Keywords: Traveling Salesman Problem, Ant Colony Optimization, Greedy Algorithm, Route Optimization, Python

Title : Comparison of Route Optimization in Traveling Salesman Problem Using Ant Colony Optimization Algorithm and Greedy Algorithm

Name : Indah Puspita Sari

NIM : H011201019

Study Program : Mathematics

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

BAB I

PENDAHULUAN

1.1 Latar Belakang

Menemukan rute dengan jarak terpendek dalam perjalanan adalah masalah umum yang sering dijumpai, masalah ini dikenal dengan istilah *Traveling Salesman Problem* (TSP). Dalam konteks TSP, optimasi kombinatorial matematika melibatkan pembentukan graf, yang mana setiap kota menjadi titik dan jarak antar kota menjadi sisi graf (Gunawan dkk, 2022). Kompleksitas TSP meningkat saat mendapati keterlibatan berbagai kondisi seperti TSP simetris dan TSP asimetris. Menurut Gambarella (1996), TSP dikatakan simetris jika kota A dan kota B memiliki jarak yang sama dengan jarak kota B ke kota A. Sedangkan pada TSP asimetris, jarak kota A ke kota B tidak sama dengan jarak kota B ke kota A. Untuk memecahkan kompleksitas ini, dapat digunakan algoritma *Ant Colony Optimization* (ACO) dan algoritma *Greedy*. Kedua algoritma ini memiliki pendekatan yang berbeda saat mencari solusi optimal dari kompleksitas TSP yang dihadapi. Oleh karena itu, kedua algoritma ini akan dibandingkan untuk memperoleh jarak tempuh dengan total paling minimum.

Ant Colony Optimization (ACO) pertama kali diperkenalkan di Italia pada tahun 1992 oleh Marco Dorigo, Vittorio Maniezzo, dan Alberto Colorni. Algoritma ACO adalah algoritma yang menggunakan pendekatan metaheuristik, yang diadaptasi dari perilaku semut dalam mencari makanan. Berdasarkan hasil penelitian yang diperoleh Dorigo diketahui bahwa dalam algoritma ACO, semut melepaskan jejak feromon untuk menandai jalur yang telah mereka tempuh, dan semut lain mengikuti jejak tersebut. Jejak-jejak tersebut akan membentuk rute yang paling efisien dalam menghubungkan titik-titik tertentu yang nantinya akan menjadi solusi dari masalah optimasi kombinatorial. Prinsip inilah yang akan diimplementasikan dalam penyelesaian masalah kompleks seperti TSP. Dalam penyelesaiannya, ACO memiliki pendekatan yang berbeda dengan algoritma *Greedy*.

Algoritma *Greedy* pertama kali diperkenalkan oleh Harold W. Kuhn pada tahun 1955. Kuhn menggunakan algoritma *Greedy* untuk menemukan pencocokan antara dua set objek. Kemudian, pada tahun 1971, Donald Knuth memperkenalkan istilah "Greedy Algorithm" yang didefinisikan sebagai algoritma yang membuat keputusan lokal yang optimal, dengan harapan bahwa keputusan tersebut akan menghasilkan solusi *global* yang optimal (Darnita dkk, 2019). Algoritma *Greedy* adalah algoritma yang mengadaptasi pendekatan heuristik untuk menentukan urutan pengambilan jalur dengan memilih kota terdekat dengan jalur terpendek, yang nantinya akan menjadi solusi optimal yang diharapkan.

Penelitian terkait algoritma *Greedy* dan algoritma *Ant Colony Optimization* telah banyak dikaji dan masih terus dikaji, diantaranya pada tahun 2008 Djamarus dan Mediawan melakukan penelitian dengan judul *Perbandingan Algoritme Ant Colony Optimization dengan Algoritme Greedy dalam Traveling Salesman Problem* dan menyimpulkan bahwa *Ant Colony Optimization* (ACO) dapat mengungguli Algoritma *Greedy* dalam menyelesaikan *Traveling Salesman Problem* (TSP) jika jumlah semut yang digunakan lebih dari 8 atau iterasi yang digunakan lebih dari 20. ACO memberikan solusi dengan kualitas yang lebih baik daripada Algoritma *Greedy*, meskipun bergantung pada parameter-parameter seperti jumlah semut dan iterasi. Pada penelitian ini, masalah TSP hanya difokuskan untuk TSP simetris saja.

Penelitian terkait penyelesaian *Traveling Salesman Problem* menggunakan algoritma *Greedy* telah dilakukan oleh Lukman dkk pada tahun 2011. Kesimpulan yang diperoleh dari penelitian tersebut adalah algoritma *Greedy* pasti memberikan suatu hasil, namun hasil tersebut tidak selalu berupa solusi yang paling optimal. Di sisi lain, algoritma *Greedy* dapat memecahkan masalah *Traveling Salesman Problem* dengan waktu yang cepat. Kemudian, pada tahun 2021 Sianturi dkk melakukan penelitian terkait optimasi rute distribusi kebutuhan pokok menggunakan algoritma *Ant Colony Optimization* (ACO). Salah satu hasil yang diperoleh dari penelitian tersebut adalah penggunaan nilai parameter sangat berpengaruh untuk menemukan kondisi optimal. Adapun kombinasi nilai parameter yang digunakan dalam penelitian ini untuk menemukan solusi yang baik adalah tetapan siklus semut (Q) bernilai 1, tetapan pengendalian intensitas semut (α)

bernilai 1, tetapan pengendali visibilitas (β) bernilai 1, dan tetapan penguapan jejak semut (ρ) bernilai 0,1. Berdasarkan penelitian terdahulu yang telah disebutkan sebelumnya, peneliti tertarik untuk memperluas cakupan penelitian dengan membandingkan algoritma *Ant Colony Optimization* dan algoritma *Greedy* pada *Traveling Salesman Problem* dalam dua kasus, yaitu simetri dan asimetri. Oleh karena itu, akan dibuat penelitian dengan judul "Perbandingan Optimasi Rute Pada *Traveling Salesman Problem* Menggunakan Algoritma *Ant Colony Optimization* dan Algoritma *Greedy*".

1.2 Rumusan Masalah

Rumusan masalah dari penelitian ini adalah solusi dari *Traveling Salesman Problem* dengan membandingkan algoritma *Ant Colony Optimization* (ACO) dan algoritma *Greedy*. Dengan fokus yang lebih spesifik dari penelitian ini mencakup

1. Bagaimana penerapan Algoritma *Ant Colony Optimization* (ACO) dan Algoritma *Greedy* pada *Traveling Salesman Problem*?
2. Bagaimana perbandingan kinerja Algoritma *Ant Colony Optimization* (ACO) dan Algoritma *Greedy* pada *Traveling Salesman Problem*?

1.3 Batasan Masalah

Batasan masalah dari penelitian ini yaitu

1. Terbatas hanya untuk kasus TSP Simetris dan Asimetris.
2. Setiap titik saling terhubung
3. Algoritma ACO yang ditemukan oleh Dorigo dkk telah mengalami banyak perkembangan penelitian yang menghasilkan banyaknya varian ACO, salah satunya adalah *Ant System* (AS). *Ant System* (AS) adalah varian ACO yang sederhana, mudah diimplementasikan, dan cepat dalam menemukan solusi dibandingkan varian ACO lainnya. Selain itu, terdapat banyak literatur dan referensi mengenai *Ant System* (AS). Maka dari itu, penelitian ini akan berfokus hanya pada *Ant System* (AS).

1.4 Tujuan Penelitian

Berdasarkan rumusan masalah, tujuan dari penelitian ini adalah untuk mengetahui algoritma manakah yang paling efisien dalam mencari jarak untuk menemukan rute paling optimal dari TSP Simetris dan Asimetris.

1.5 Manfaat Penelitian

Berdasarkan tujuan penelitian, maka diharapkan agar penelitian ini dapat menambah pemahaman tentang penyelesaian TSP, khususnya pada kasus TSP Simetris dan Asimetris dengan menggunakan Algoritma *Ant Colony Optimization* (ACO) dan Algoritma *Greedy*.

1.6 Sistematika Penelitian

Penulisan tugas akhir ini dibagi dalam 5 bab dengan rincian sebagai berikut.

BAB I PENDAHULUAN

Pada bab ini, akan diuraikan mengenai latar belakang masalah, rumusan masalah, batasan masalah, tujuan penelitian, manfaat penelitian, dan sistematika penelitian yang memberikan pemaparan singkat terkait tugas akhir ini.

BAB II TINJAUAN PUSTAKA

Pada bab ini, akan diuraikan mengenai beberapa materi dan teori yang dapat memudahkan dalam meneliti masalah tugas akhir ini, yaitu graf, optimisasi, *Traveling Salesman Problem*, algoritma G*reedy, Ant Colony Optimization* (ACO), dan Python.

BAB III METODOLOGI PENELITIAN

Pada bab ini, akan diuraikan mengenai data penelitian dan langkah-langkah penelitian yang berisi flowchart yang menggambarkan proses berjalannya penelitian ini.

BAB IV PEMBAHASAN

Pada bab ini, akan diuraikan mengenai hasil penelitian algoritma *Greedy* dan *Ant Colon Optimization* (ACO) dalam pengoptimalan rute dan jarak tempuh pada *Traveling Salsman Problem* (TSP) dalam dua kasus, yaitu TSP simetris dan asimetris.

BAB V PENUTUP

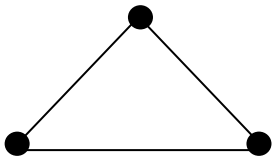
Pada bab ini, akan diberikan kesimpulan berdasarkan hasil yang diperoleh pada bab sebelumnya untuk menjawab permasalahan yang telah ditemukan pada penelitian ini serta memberikan saran untuk pengembangan penelitian selanjutnya.

BAB II

TINJAUAN PUSTAKA

2.1 Graf

Pada tahun 1736, seorang matematikawan Swiss bernama Leonardo Euler dalam karyanya "*Solutio Problematis ad Geometrian Situs Pertinentis*". Karya tersebut terinspirasi dari masalah jembatan Königsberg dan konsep inilah yang menjadi titik awal munculnya suatu cabang Matematika yakni teori graf. Secara umum, didefinisikan bahwa graf G adalah pasangan himpunan (V, E) dengan V adalah himpunan diskrit yang anggota-anggotanya disebut titik, dan E adalah himpunan dari pasangan anggota-anggota V yang disebut sisi. Graf sederhana G adalah pasangan $(V(G), E(G))$, dimana $V(G)$ adalah himpunan diskrit berhingga dan tidak kosong, yang anggotanya disebut titik (*vertex*), dan $E(G)$ adalah himpunan pasangan-pasangan tak terurut dan berbeda dari anggota-anggota $V(G)$ yang disebut sisi (*edge*) (Hasmawati, 2020). Contoh dari sebuah graf dapat dilihat pada Gambar 2.1 berikut.



Gambar 2. 1 Graf G

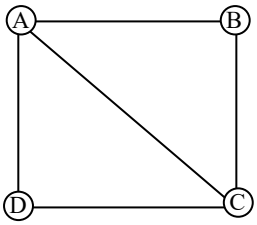
2.2.1 Jenis Graf

Berdasarkan orientasi arah pada sisi, maka secara umum graf dibedakan menjadi 2 jenis, yaitu:

A. Graf tidak berarah

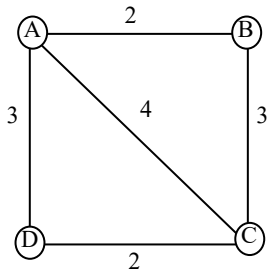
Graf tidak berarah adalah graf yang sisinya tidak mempunyai orientasi arah. Pada graf tidak berarah, urutan pasangan titik yang dihubungkan oleh sisi tidak diperlihatkan. Sehingga $(V_j, V_k) = (V_k, V_j)$ adalah sisi yang sama. Graf tidak berarah dikategorikan berdasarkan bobotnya menjadi 2 jenis, yaitu

1) Graf tidak berarah dan tidak berbobot, yaitu graf yang setiap sisinya tidak memiliki tanda panah dan tidak memiliki bobot. Contohnya dapat dilihat pada Gambar 2.2.



Gambar 2. 2 Graf tidak berbobot dan tidak berarah

2) Graf tidak berarah dan berbobot, yaitu graf yang setiap sisinya tidak memiliki tanda panah dan memiliki bobot. Contohnya dapat dilihat pada Gambar 2.3.

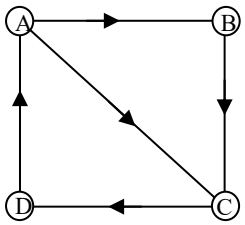


Gambar 2. 3 Graf tidak berarah dan berbobot

B. Graf berarah

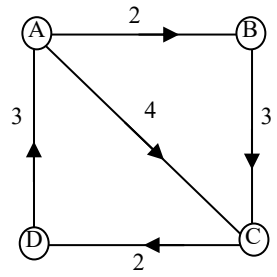
Graf berarah adalah graf yang setiap sisinya diberikan orientasi arah. Pada graf berarah, (V_j, V_k) dan (V_k, V_j) menyatakan dua sisi yang berbeda, dengan kata lain $(V_j, V_k) \neq (V_k, V_j)$. Untuk sisi V_j merupakan titik asal dan titik V_k disebut titik terminal. Graf berarah dikategorikan berdasarkan bobotnya menjadi 2 jenis, yaitu

1) Graf berarah dan tidak berbobot, yaitu graf yang setiap sisinya memiliki tanda panah tetapi tidak memiliki bobot. Contohnya dapat dilihat pada Gambar 2.4.



Gambar 2. 4 Graf berarah dan tidak berbobot

2) Graf berarah dan berbobot, yaitu graf yang setiap sisinya memiliki tanda panah dan memiliki bobot. Contohnya dapat dilihat pada Gambar 2.5 (Gunawan dkk, 2023).



Gambar 2. 5 Graf berarah dan berbobot

2.2 Optimisasi

Optimisasi adalah proses mencari nilai maksimum atau minimum dari suatu fungsi objektif dalam batas tertentu. Dalam konteks matematika, optimasi adalah cabang analisis matematika yang mencakup berbagai metode dan teknik untuk menemukan solusi optimal dari masalah-masalah terdefinisi otomatis. Tujuan utama optimisasi adalah mencari nilai variabel atau parameter dalam fungsi tujuan sehingga memenuhi batasan dan menghasilkan nilai optimal sesuai dengan kriteria tertentu, yang bisa berupa nilai maksimum atau minimum tergantung pada sifat masalah dan persyaratan yang diberikan, seperti pada masalah optimisasi kombinatorial yang mencari struktur diskrit, seperti rute (urutan titik dalam suatu graf), jalur, atau graf, yang mengoptimalkan nilai tujuan (Septianto dkk, 2023).

2.2.1 Lintasan Terpendek

Lintasan adalah urutan titik-titik dalam graf yang dihubungkan oleh sisi, dan pencarian nilai variabel untuk mencapai nilai maksimal disebut jalur terpendek. Jalur terpendek adalah rangkaian titik yang saling terhubung dalam graf dengan

jarak total minimal di antara semua kemungkinan jalur. Pencarian solusi untuk masalah ini melibatkan berbagai pendekatan algoritma, termasuk Algoritma *Ant Colony Optimization* dan Algoritma *Greedy*. Masalah jalur terpendek terkait dengan menentukan sisi-sisi dalam suatu jaringan untuk membentuk rute terdekat antara sumber dan tujuan, dengan tujuan mencari jalur terpendek antara titik asal dan tujuan.

2.2.2 Solusi Masalah Optimalisasi

Pencarian jalur terpendek dapat memanfaatkan beragam algoritma. Secara garis besar, pemecahan masalah pencarian rute terpendek dapat diakukan melalui dua metode, yaitu metode heuristik dan metode metaheuristik. Pada metode heuristik, pemecahan masalahnya menggunakan pendekatan lokal, sedangkan metode metaheuristik menggunakan pendekatan *global* dalam pemecahan masalahnya.

1) Metode Heuristik

   Metode heuristik menggunakan pendekatan sistematis untuk melakukan pencarian dalam optimalisasi. Metode heuristik sering digunakan untuk memecahkan masalah yang sulit atau tidak dapat diselesaikan dengan algoritma eksak. Beberapa algoritma heuristik yang umum digunakan dalam permasalahan optimasi adalah Algoritma *Greedy*, Algoritma *backtracking*, dan lainnya.

2) Metode Metaheuristik

   Metode metaheuristik adalah kelanjutan dari metode heuristik. Metode metaheuristik mencari solusi dengan menggabungkan interaksi antara prosedur pencaria lokal dan strategi lain untuk menciptakan proses pencarian diluar titik-titik *local optimal* dan melakukan pencarian untuk menemukan solusi *global*. Beberapa algoritma metaheuristik yang umum digunakan dalam permasalahan optimasi adalah Algoritma *Ant Colony Optimization*, Algoritma *Particle Swarm Optimization*, dan lainnya.

2.3 *Traveling Salesman Problem* (TSP)

Traveling Salesman Problem (TSP) pertama kali diusulkan pada tahun 1859 dalam penelitian "Icsian Calculus" oleh matematikawan Irlandia, Sir William

Rowan Hamilton. Hamilton mengilustrasikan pencarian rute terpendek yang menghubungkan titik *polygon* dengan 20 sisi. Penggunaan istilah TSP muncul dalam penelitian Karl Menger, "Zur Allgemeinen Kurventhoir," pada tahun 1932, ketika Menger menjelaskan konsep mencari rute terpendek yang menghubungkan titik-titik dalam diagram graf. TSP sendiri merupakan masalah optimasi kombinatorial yang meminta seorang salesman mengunjungi n kota, setiap kota hanya sekali, dan memilih rute sedemikian rupa sehingga total jarak tempuh minimal. Representasi TSP dilakukan dengan graf lengkap dan berbobot $G = (V, E)$, di mana V adalah himpunan titik (kota) dan E adalah himpunan sisi (Gunawan dkk, 2022). Tujuan dari TSP adalah menemukan rute yang melalui setiap kota tepat satu kali, kembali ke kota awal, dan memiliki total jarakminimal. Fungsi matematika yang mencerminkan tujuan ini dapat dirumuskan sebagai berikut:

n adalah jumlah titik yang akan dikunjungi.

d_{ij} adalah jarak antara titik i dan titik j.

x_{ij} adalah variabel keputusan biner.

Variabel Keputusan:

$$x_{ij} = \begin{cases} 1, & j\text{ika rute melewati tepat sekali dari titik } i \text{ ke titik } j \\ 0, & \text{lainnya} \end{cases}$$

Fungsi Tujuan:

$$\min \sum_{i=1}^{n} \sum_{j=1, j\neq i}^{n} d_{ij} x_{ij}$$

Ketentuan:

1. Setiap titik harus terhubung dengan tepat satu titik lain:

$$\sum_{j=1, j\neq i}^{n} x_{ij} = 1, \qquad \forall i \in V, \qquad i = 1,2,3, \dots, n$$

   Setiap titik harus memiliki tepat satu jalur yang keluar:

$$\sum_{i=1, i\neq j}^{n} x_{ij} = 1, \qquad \forall j \in V, \qquad j = 1,2,3, \dots, n$$

2. Tidak ada subrute (*subtour elimination*), artinya rute harus membentuk siklus:

$$u_i - u_j + nx_{ij} \leq n - 1, \qquad \forall i,j \in V, i \neq j, i,j > 1$$
$$2 \leq u_i \leq n, \qquad \forall i \in V$$

dalam batasan terakhir, u_i adalah variabel bantu untuk menunjukkan urutan kota dalam rute (Silalahi, 2019).

Misalkan $V = \{v_1, \ldots, v_n\}$ adalah himpunan kota, $E = \{(i,j): i,j \in V\}$ adalah himpunan sisi, dan $d_{ij} = d_{ji}$ adalah jarak yang terkait dengan sisi $(i,j) \in E$. Setiap sisi (i,j) memiliki nilai (jarak) $d(i,j)$ yang menyatakan jarak dari kota i ke kota j. Dalam kasus ini, $v_i \in V$ diberikan oleh koordinat (x_i, y_i) dan d_{ij} adalah jarak *Euclidean* antara i dan j, yang dikenal dengan *Euclidean* TSP. Berikut beberapa jenis masalah TSP yang umum, anataralain:

1) TSP Simetris (*Symmetric Traveling Salesman Problem*)
   TSP simetris adalah jenis TSP ketika jarak antara dua kota selalu sama, meskipun *salesman* bergerak dari kota A menuju kota B atau sebaliknya. Dalam TSP simetris, jarak dari kota i ke kota j sama dengan jarak dari kota j ke kota i $d(i,j) = d(j,i)$ untuk setiap pasangan kota.
2) TSP Asimetris (*Asymmetric Traveling Salesman Problem*)
   TSP asimetris adalah jenis TSP ketika jarak antara dua kota bervariasi tergantung arah perjalanan *salesman*, apakah bergerak dari kota A menuju kota B atau sebaliknya. Dalam TSP asimetris, jarak dari kota i ke kota j tidak sama dengan jarak dari kota i ke kota j $d(i,j) \neq d(j,i)$ untuk setiap pasangan kota (Gambarella, 1996).

2.4 Algoritma *Greedy*

Algoritma *Greedy* pertama kali diperkenalkan oleh Harold W. Kuhn dalam karyanya "The Hungarian Method for the Assignment Problem" pada tahun 1955, algoritma tersebut digunakan untuk menyelesaikan masalah *assignment*, yaitu mencari pencocokan antara dua set objek. Istilah "greedy algorithm" kemudian diperkenalkan oleh Donald Knuth pada tahun 1971 dalam bukunya "The Art of Computer Programming", yang mendefinisikan Algoritma *Greedy* sebagai algoritma yang membuat keputusan lokal yang optimal, dengan harapan bahwa

keputusan tersebut akan menghasilkan solusi *global* yang optimal. Algoritma *Greedy* membantu menentukan urutan pengambilan jalur atau jalur optimal lokal, membentuk rute terpendek atau optimal *global*. Berikut adalah urutan solusi yang diberikan oleh Algoritma *Greedy*:

1. Pada setiap langkah solusi, evaluasi berbagai pilihan untuk mengambil keputusan terbaik yang tidak dapat diubah pada langkah berikutnya.
2. Algoritma *Greedy* mengadopsi pendekatan di mana opsi optimal diambil pada setiap langkah, mencari hasil terbaik secara lokal, dengan harapan mencapai solusi global yang optimal (Darnita dkk, 2019).

2.4.1 Skema Umum Algoritma *Greedy*

Algoritma *Greedy* berfokus pada pemilihan setiap sisi secara terpisah, tanpa mempertimbangkan dampak setelahnya. Pendekatan ini tidak memeriksa semua alternatif solusi yang mungkin, dan kadang-kadang tidak memberikan solusi yang optimal, meskipun solusi yang dihasilkan cenderung mendekati nilai optimal. Dalam konteks algoritma *Greedy*, masalah optimasi dibangun oleh elemen-elemen berikut:

1. Himpunan kandidat, C. Himpunan ini terdiri dari elemen-elemen yang berpeluang membentuk solusi. Dalam konteks permasalahan lintasan terpendek pada graf, himpunan kandidat ini adalah kumpulan titik-titik dari graf tersebut.
2. Himpunan solusi, S. Himpunan ini memuat solusi dari permasalahan yang sedang diselesaikan dan terdiri dari elemen-elemen dalam himpunan kandidat. Namun, tidak semua elemen dalam himpunan kandidat termasuk ke dalam himpunan solusi. Dengan kata lain, himpunan solusi ini merupakan subbagian dari himpunan kandidat.
3. Fungsi seleksi, diartikan sebagai predikat SELEKSI, berfungsi memilih kandidat yang paling mungkin menghasilkan solusi optimal pada setiap langkah. Kandidat yang sudah dipilih pada suatu langkah tidak akan dipertimbangkan lagi pada langkah berikutnya.
4. Fungsi kelayakan, diartikan sebagai predikat LAYAK, berfungsi memeriksa apakah suatu kandidat yang telah dipilih (setelah diseleksi) dapat memberikan solusi yang layak. Layak dalam artian, kandidat tersebut, ketika ditambahkan

ke dalam himpunan solusi yang telah terbentuk, tidak akan melanggar urutan yang sudah ada.

5. Fungsi obyektif, bertujuan memaksimumkan atau meminimumkan nilai solusi. Tujuannya adalah memilih satu solusi terbaik dari masing-masing anggota himpunan solusi.

Optimal *global* diharapkan menjadi solusi terbaik dalam suatu masalah, tetapi terkadang, optimal *global* tidak selalu merupakan solusi terbaik, tetapi dapat berupa solusi sub-optimal atau pseudo-optimal. Hal ini dapat dijelaskan oleh dua faktor utama:

1) Algoritma *Greedy* tidak mengeksplorasi semua alternatif solusi yang ada secara menyeluruh.
2) Pemilihan fungsi SELEKSI, biasanya bergantung pada fungsi obyektif dan penting untuk memilih fungsi yang tepat agar menghasilkan nilai yang optimal.

Oleh karena itu, pada beberapa kasus, algoritma *Greedy* tidak selalu dapat memberikan solusi yang benar-benar optimal. Namun, algoritma *Greedy* selalu memberikan solusi yang mendekati nilai optimal, yaitu solusi pendekatan (*approximation*) (Darnita dkk, 2019).

2.4.2 Cara Kerja Algoritma *Greedy*

Diberikan sebuah graf berbobot $G(V, E)$. Untuk menentukan lintasan terpendek dari titik awal a, ke setiap titik lainnya di G, diasumsikan bahwa bobot semua sisi bernilai positif. Algoritma *Greedy* untuk mencari lintasan terpendek dapat dirumuskan sebagai berikut (Darnita dkk, 2019)

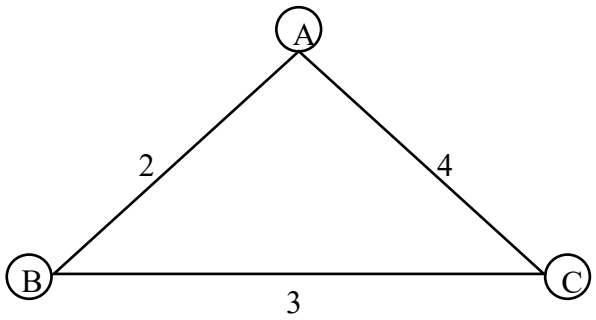
1. Periksa semua sisi yang langsung terhubung dengan titik a. Pilih sisi yang bobotnya terkecil. Sisi ini menjadi lintasan terpendek pertama, sebut saja $L(1)$.
2. Tentukan lintasan terpendek kedua dengan cara sebagai berikut:
   (i) Hitung $d(i)$ = panjang $L(1)$ + bobot sisi dari titik akhir $E(1)$ ke titik i yang lain.
   (ii) Pilih $d(i)$ yang terkecil.
   (iii) Bandingkan $d(i)$ dengan bobot sisi (a, i). Jika bobot sisi (a, i) lebih kecil daripada $d(i)$, maka $L(2) = L(1) \cup$ sisi dari titik akhir $L(i)$ ke titik i.

3. Dengan cara yang sama, ulang langkah (2) untuk menemukan lintasan terpendek berikutnya.

Contoh 2.1

1. Contoh Penerapan Algoritma *Greedy* pada TSP Simetris

   Sebuah perusahaan perlu mengirimkan paket ke lima kota: A, B, C, D, dan E. Pengiriman dimulai dari kota A dan harus melalui setiap kota sekali sebelum kembali ke kota A. Tentukan rute pengiriman yang paling efisien dengan jarak terpendek, dengan jarak setiap kota (dalam satuan km) diberikan pada Gambar 2.6 berikut (Sumber: Data Diolah 2024)



Gambar 2. 6 Graf G dengan titik A, B, C

Penyelesaian:

Langkah-langkah penyelesaian, yaitu sebagai berikut

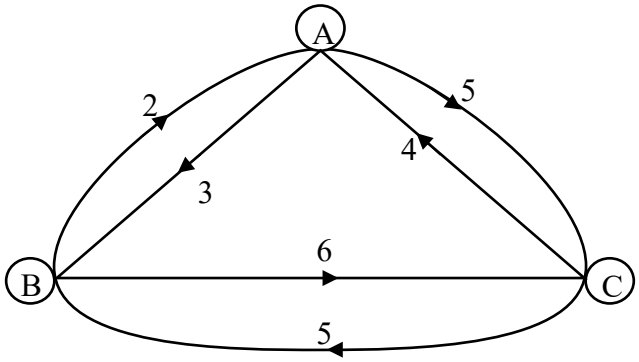
1) Memulai dari kota A. Kota yang jaraknya paling dekat dengan kota A adalah kota B, dengan jarak 3 km.
2) Mengunjungi kota B. Kota yang jaraknya paling dekat dengan kota C yang belum dikunjungi adalah kota C, dengan jarak 4 km.
3) Mengunjungi kota C. Dari kota C kembali ke kota A, dengan jarak 4 km.

Diperoleh hasil sebagai berikut: Rute terpendek untuk mengunjungi kelima kota adalah A –B –C – A, dengan total jarak 9 km.

2. Contoh Penerapan Algoritma *Greedy* pada TSP asimetris

   Sebuah perusahaan perlu mengirimkan paket ke lima kota: A, B, C, dan D. Pengiriman dimulai dari kota A dan harus melalui setiap kota sekali sebelum kembali ke kota A. Tentukan rute pengiriman yang paling efisien dengan jarak

terpendek, dengan jarak setiap kota (dalam satuan km) diberikan pada Gambar 2.7 berikut (Sumber: Data Diolah 2024)



Gambar 2. 7 Graf berarah dan berbobot dengan titik A, B, C

Penyelesaian:

Langkah-langkah penyelesaian, yaitu sebagai berikut

1) Memulai dari kota A. Kota yang jaraknya paling dekat dengan kota A adalah kota B, dengan jarak 3 km.
2) Mengunjungi kota B. Kota yang jaraknya paling dekat dengan kota C yang belum dikunjungi adalah kota C, dengan jarak 6 km.
3) Mengunjungi kota C. Dari kota C kembali ke kota A, dengan jarak 4 km.

Diperoleh hasil sebagai berikut: Rute terpendek untuk mengunjungi kelima kota adalah A –B –C – A, dengan total jarak 13 km.

Catatan: Dengan menggunakan Algoritma *Greedy*, pengiriman barang akan selalu memilih kota yang paling dekat dengan kota yang sedang dikunjungi pada saat itu. Pada awalnya, ini tampak sebagai keputusan terbaik karena dapat mengurangi total jarak yang harus ditempuh. Namun, metode ini tidak selalu menjamin solusi optimal. Sebagai contoh pada TSP asimetri, jika pada soal tidak ada instruksi yang mengharuskan pengiriman barang dimulai dari kota A, maka rute terpendek dan total jaraknya pun berubah menjadi lebih optimal daripada yang telah diperoleh.

2.5 *Ant Colony Optimization* (ACO)

Ant Colony Optimization atau ACO pertama kali diperkenalkan di Italia pada tahun 1992 oleh Marco Dorigo, Vittorio Maniezzo, dan Alberto Colorni dalam

penelitian yang berjudul "Ant System: Optimization by a Colony of Cooperating Agents". Algoritma ini terinspirasi dari perilaku semut dalam mencari makanan. Semut melepaskan feromon untuk memberi petunjuk kepada sesama semut, menandai jalur menuju sumber makanan atau sarangnya.

Feromon semut berfungsi sebagai jejak komunikasi dalam koloni semut, memungkinkan interaksi dan kerja sama terorganisir tanpa komunikasi langsung. Proses peninggalan feromon, dikenal sebagai stigmeri, membuat semut-semut dapat bekerja bersama secara terkoordinasi. Seiring waktu, feromon menguap secara alami, tetapi semut dapat memperbarui jejak dengan melepaskan feromon baru saat bergerak. Jalur yang sering dilalui semut memiliki feromon lebih kuat, menandakan keefisienan jalur tersebut. Dalam algoritma ACO, semut melepaskan jejak feromon untuk menandai jalur, dan semut lain mengikuti jejak ini mencari solusi terbaik. Intensitas feromon yang tinggi memungkinkan algoritma ACO menemukan jalur paling efisien dalam menghubungkan titik-titik, memecahkan masalah optimasi kombinatorial (Dorigo dkk, 1997).

Ant System (AS) merupakan varian pertama algoritma ACO yang efisien untuk menyelesaikan masalah kompleks seperti *Traveling Salesman Problem*. Dalam operasinya, setiap semut memulai perjalanan dari titik awal yang dipilih secara acak, dengan pemilihan titik dalam rute didasarkan pada fungsi probabilitas yang mempertimbangkan jarak antar titik dan konsentrasi feromon pada jalur yang menghubungkan titik-titik tersebut. Semut cenderung memilih jalur pendek dengan konsentrasi feromon tinggi, dan mereka menggunakan *tabu list* untuk menghindari *revisiting* titik yang sudah dikunjungi. Setelah semua semut menyelesaikan rutenya, dilakukan penggantian keseluruhan pada zat feromon, mempengaruhi pilihan rute-rute berikutnya. Jalur yang dioptimalkan mendapatkan lebih banyak feromon, memastikan eksplorasi yang efektif dalam mencari solusi optimal, sementara penguapan feromon berperan dalam mencegah stagnasi. Proses ini diulang hingga mencapai batas rute maksimum, menjadikan algoritma ini sebagai pendekatan yang terus mencari solusi alternatif (Dorigo dkk, 2006).

Untuk menentukan jalur terpendek dalam algoritma ACO, diperlukan beberapa langkah, diantaranya:

Langkah 1:

1. Parameter-parameter yang digunakan dalam algoritma ACO antara lain:
   a. Intensitas jejak semut (τ_{ij}) dan perubahannya $(\Delta\tau_{ij})$. Intensitas jejak semut (τij) harus di inisialisasi sebelum memulai siklus. τ_{ij} digunakan dalam persamaan probabilitas titik yang akan dikunjungi. $\Delta\tau_{ij}$ diinisialisasi setelah selesai satu siklus. $\Delta\tau_{ij}$ digunakan untuk menentukan τ_{ij} untuk siklus selanjutnya.
   b. Tetapan siklus semut (Q), merupakan konstanta yang digunakan dalam persamaan untuk menentukan $\Delta\tau_{ij}$.
   c. Bilangan acak (r), merupakan bilangan yang dibangkitkan secara acak untuk menentukan kota berikutnya yang akan dikunjungi oleh semut.
   d. Tetapan pengendali intensitas jejak semut (α), digunakan dalam persamaan probabilitas titik yang akan dikunjungi, dan berfungsi sebagai pengendali intensitas jejak semut. Nilai parameter α adalah $\alpha \geq 0$.
   e. Tetapan pengendali visibilitas (β), digunakan sebagai pengendali visibilitas dan digunakan dalam persamaan probabilitas titik yang akan dikunjungi. Nilai parameter β adalah $\beta \geq 0$.
   f. Visibilitas antar kota (η_{ij}), digunakan dalam persamaan probabilitas titik yang akan dikunjungi. Nilai η_{ij}merupakan hasil dari $\frac{1}{d_{ij}}$.
   g. Banyak semut (m).
   h. Banyak titik (n) termasuk koordinat (x, y).
   i. Tetapan penguapan jejak semut (ρ), digunakan untuk menentukan τ_{ij} untuk siklus selanjutnya. Demi mencegah jumlah jejak zat feromon yang tak terhingga. Maka, nilai parameter ρ harus $0 < \rho < 1$.

Langkah 2:

Pengisian kota pertama ke dalam *tabu list*. *Tabu list* diartikan sebagai catatan yang digunakan oleh semut untuk menyimpan kota-kota yang telah dikunjungi agar tidak terjadi revisiting titik. Hasil inisialisasi kota pertama setiap semut dalam langkah 1 harus diisikan sebagai elemen pertama *tabu list*. Hasil dari langkah ini adalah terisinya elemen pertama *tabu list* setiap semut dengan indeks kota tertentu,

yang berarti bahwa setiap $tabu_k$ (1) bisa berisi indeks kota antara 1 sampai n sebagaimana hasil inisialisasi pada langkah 1.

Langkah 3:

Penyusunan rute kunjungan setiap semut ke setiap kota. Koloni semut yang sudah terdistribusi ke sejumlah atau setiap kota, akan mulai melakukan perjalanan dari kota pertama masing-masing sebagai kota asal dan salah satu kota lainnya sebagai kota tujuan. Kemudian dari kota kedua masing-masing, koloni semut akan melanjutkan perjalanan dengan memilih salah satu dari kota-kota yang tidak terdapat pada $tabu_k$ sebagai kota tujuan selanjutnya.

Perjalanan koloni semut berlangsung terus menerus sampai semua kota satu persatu dikunjungi atau telah menempati $tabu_k$. Jika s menyatakan indeks urutan kunjungan, kota asal dinyatakan sebagai $tabu_k(s)$ dan kota-kota lainnya dinyatakan sebagai $\{N - tabu_k\}$, maka untuk menentukan kota tujuan digunakan persamaan probabilitas kota untuk dikunjungi sebagai berikut:

$$p_{ij}^{k} = \begin{cases} \dfrac{[\tau_{ij}]^{\alpha}[\eta_{ij}]^{\beta}}{\sum_{u \in J_i^k}[\tau_{ij}]^{\alpha}[\eta_{ij}]^{\beta}}, & \text{untuk } s \in J_i^k \\ 0, & \text{untuk } s \text{ lainnya} \end{cases} \qquad (2.1)$$

dengan $\eta_{ij} = \frac{1}{d_{ij}}$, i sebagai indeks kota asal, dan j sebagai indeks kota tujuan.

Langkah 4:

1. Perhitungan panjang rute tertutup (*length closed tour*) atau L_k setiap semut dilakukan setelah satu siklus diselesaikan oleh semua semut. Perhitungan ini dilakukan berdasarkan $tabu_k$ masing-masing dengan persamaan berikut:

$$L_k = d_{tabu_k(n)tabu_k(1)} + \sum_{s=1}^{n-1} d_{tabu_k(s),tabu_k(s+1)} \qquad (2.2)$$

2. Koloni semut akan meninggalkan jejak-jejak pada lintasan antar kota yang dilaluinya. Faktor-faktor seperti penguapan dan variasi jumlah semut yang melewati lintasan tersebut dapat menyebabkan fluktuasi nilai intensitas jejak kaki semut antar kota. Persamaan yang menggambarkan perubahan ini adalah:

$$\Delta\tau_{ij} = \sum_{k=1}^{m} \Delta\tau_{ij}^{k} \tag{2.3}$$

dengan $\Delta\tau_{ij}^{k}$ adalah perubahan nilai intensitas jejak kaki semut antar kota setiap semut yang dihitung berdasarkan persamaan:

$$\Delta\tau_{ij}^{k} = \begin{cases} \dfrac{Q}{L_k}, & \text{untuk } (i,j) \in \text{ kota asal dan kota tujuan dalam } tabu_k \\ 0, & \text{untuk } (i,j) \text{ lainnya} \end{cases} \tag{2.4}$$

Langkah 5:

1. Perhitungan nilai intensitas jejak semut antar kota untuk siklus selanjutnya. Nilai intensitas jejak kaki semut antar kota pada semua lintasan antar kota ada kemungkinan berubah karena adanya penguapan dan perbedaan jumlah semut yang melewati. Untuk siklus selanjutnya, semut yang akan melewati lintasan tersebut nilai intensitasnya telah berubah. Nilai intensitas jejak kaki semut antar kota untuk siklus selanjutnya dihitung dengan persamaan:

$$\tau_{ij} = (1-\rho) \cdot \tau_{ij} + \sum_{k=1}^{m} \Delta\tau_{ij}^{k} \tag{2.5}$$

2. Atur ulang nilai perubahan intensitas jejak semut antar kota. Untuk siklus selanjutnya perubahan nilai intensitas jejak semut antar kota perlu diatur kembali agar memiliki nilai sama dengan nol.

Langkah 6:

Pengosongan *tabu list*, dan ulangi langkah 2 jika diperlukan. *Tabu list* perlu dikosongkan untuk diisi lagi dengan urutan kota yang baru pada siklus selanjutnya, jika jumlah siklus maksimum belum tercapai atau belum terjadi konvergensi. Algoritma diulang lagi dari langkah 2 dengan harga parameter intensitas jejak kaki semut antar kota yang sudah diperbaharui.

Contoh 2.2

1. Contoh Penerapan ACO pada TSP Simetris

   Sebuah perusahaan perlu mengirimkan paket menuju tiga kota: A, B, dan C Pengiriman dimulai dari kota A dan harus melalui setiap kota sekali sebelum

kembali ke kota A. Tentukan rute pengiriman yang paling efisien dengan jarak terpendek, dengan jarak setiap kota (dalam satuan km) diberikan pada tabel 2.1 berikut:

Tabel 2. 1 Jarak antar kota A, B, dan C

Kota	**A**	**B**	**C**
A	0	2	4
B	2	0	3
C	4	3	0

Sumber: Data Diolah 2024

Penyelesaian:

1) Menentukan nilai parameter yang digunakan

$n = m = 3$ $\rho = 0{,}5$

$\alpha = \beta = Q = 1$ $\tau_{ij} = 10$

Melakukan inisialisasi

$$\tau_{ij} = \begin{pmatrix} 0 & 10 & 10 \\ 10 & 0 & 10 \\ 10 & 10 & 0 \end{pmatrix}$$

2) Menempatkan semut.

Penempatan semut untuk pertama kali dilakukan secara acak.

Semut	1	2	3
Kota Awal	B	A	C

3) Membangkitkan bilangan acak untuk menentukan titik (kota) berikutnya

Menggunakan rumus:

$$p_{ij}^{k} = \frac{[\tau_{ij}]^{\alpha}[\eta_{ij}]^{\beta}}{\sum_{l \in \mathcal{N}_i^k}[\tau_{ij}]^{\alpha}[\eta_{ij}]^{\beta}} \text{ dengan } \eta_{ij} = \frac{1}{d_{ij}}$$

- Iterasi 1

Pembagi semut 1

$$= \sum_{j\neq 2}^{3} [\tau_{ij}]^{\alpha} \left[\frac{1}{d_{ij}}\right]^{\beta} = \left(10 \times \frac{1}{2}\right) + \left(10 \times \frac{1}{3}\right) = 8{,}3333$$

Pembagi semut 2

$$= \sum_{j\neq 1}^{3} [\tau_{ij}]^{\alpha} \left[\frac{1}{d_{ij}}\right]^{\beta} = \left(10 \times \frac{1}{2}\right) + \left(10 \times \frac{1}{4}\right) = 7{,}5000$$

Pembagi semut 3

$$= \sum_{j\neq 3}^{3} [\tau_{ij}]^{\alpha} \left[\frac{1}{d_{ij}}\right]^{\beta} = \left(10 \times \frac{1}{4}\right) + \left(10 \times \frac{1}{3}\right) = 5{,}8333$$

Akumulasi dari iterasi 1 soal ACO pada TSP Simetris, yaitu sebagai berikut:

Tabel 2. 2 Perhitungan Iterasi 1 soal ACO pada TSP Simetris

Semut	Kota saat ini	Probabilitas pilih kota		
		A	B	C
1	C	0,6000	0,0000	0,4000
2	B	0,0000	0,6667	0,3333
3	A	0,4286	0,5714	0,0000

Berikutnya bangkitakan bilangan acak untuk mennetukan kota berikutnya yang akan dikunjungi oleh semut.

4) Mengisi *tabu list*

Semut	1	2	3
Rute	B, A	A, B	C, B

Pada iterasi kedua dan seterusnya, lakukan langkah-langkah yang sama seperti yang dilakukan pada iterasi pertama. Lakukan iterasi sampai *tabu list* terisi sehingga dapat diperoleh rute yang akan dilalui oleh semut. Setelah

menyelesaikan perhitungan dan pengisian *tabu list* akan diperoleh hasil seperti pada Tabel 2.3 berikut:

Tabel 2. 3 Rute dan Total Jarak

Semut	Rute				Total jarak
1	B	A	C	B	9
2	A	B	C	A	9
3	C	B	A	C	9

5) Memperbarui feromon

Menggunakan rumus: $\tau_{ij} = (1-\rho) \cdot \tau_{ij} + \sum_{k=1}^{m} \Delta\tau_{ij}^{k}$

Pertama, perhatikan Tabel 2.3, lihat apakah ada rute A dan B yang berdekatan baik dari A lalu ke B maupun sebaliknya. Lihat pada semut 1, karena kita mendapati A dan B yang berdekatan maka feromon baru untuk dari kota A ke kota B. Diperoleh hasil pada Tabel 2.4 sebagai berikut:

Tabel 2. 4 Hasil Perhitungan Pembaruan Feromon

Kota Asal	Kota Tujuan	Feromon Baru
A	B	5,2222
A	C	5,3333
B	C	5,3333

Sehingga, dapat ditarik kesimpulan bahwa rute pengiriman yang paling efisien dengan jarak terpendek adalah rute A, B, C, A dengan total jarak 9 km.

2. Contoh Penerapan ACO pada TSP Asimetris

Sebuah perusahaan perlu mengirimkan paket menuju kota: A, B, dan C. Pengiriman dimulai dari kota A dan harus melalui setiap kota sekali sebelum kembali ke kota A. Tentukan rute pengiriman yang paling efisien dengan jarak terpendek, dengan jarak setiap kota (dalam satuan km) diberikan pada Tabel 2.5 berikut

Tabel 2. 5 Jarak antar kota A, B, C, dan D

Kota	**A**	**B**	**C**
A	0	3	5
B	2	0	6
C	4	5	0

Sumber: Data Diolah 2024

Penyelesaian:

1) Menentukan nilai parameter yang digunakan

$n = m = 3$ $\rho = 0{,}5$

$\alpha = \beta = Q = 1$ $\tau_{ij} = 10$

Melakukan inisialisasi

$$\tau_{ij} = \begin{pmatrix} 0 & 10 & 10 \\ 10 & 0 & 10 \\ 10 & 10 & 0 \end{pmatrix}$$

2) Menempatkan semut.

Penempatan semut untuk pertama kali dilakukan secara acak.

Semut	1	2	3
Kota Awal	B	A	C

3) Membangkitkan bilangan acak untuk menentukan titik (kota) berikutnya

Perlu diingat, soal ini adalah bentuk TSP asimetris, sehingga kita akan mencari lebih banyak kemungkinan rute yang diharapkan menjadi solusi optimal. Pertama, kita akan mencari jarak antar kota untuk nilai yang mendatar, sehingga diperoleh hasil sebagai berikut:

Tabel 2. 6 Perhitungan Iterasi 1 (bagian 1) soal ACO pada TSP Asimetris

Semut	Kota saat ini	Probabilitas pilih kota		
		A	B	C
1	B	0,7500	0,0000	0,2500
2	D	0,0000	0,6250	0,3750
3	C	0,5556	0,4444	0,0000

Berikutnya bangkitakan bilangan acak untuk mennetukan kota berikutnya yang akan dikunjungi oleh semut.

4) Mengisi *tabu list*

Semut	1	2	3
Rute	B, A	A, B	C, A

Pada iterasi kedua dan seterusnya, lakukan langkah-langkah yang sama seperti yang dilakukan pada iterasi pertama. Lakukan iterasi sampai *tabu list* terisi sehingga dapat diperoleh rute yang akan dilalui oleh semut. Setelah menyelesaikan perhitungan dan pengisian *tabu list* akan diperoleh hasil seperti pada Tabel 2.7 berikut:

Tabel 2. 7 Rute dan Total Jarak

Semut	Rute				Total Jarak
1	B	A	C	B	12
2	A	B	C	A	13
3	C	B	A	C	12

5) Memperbarui feromon berdasarkan perhitungan pertama

Diperoleh hasil pada Tabel 2.8 sebagai berikut:

Tabel 2. 8 Hasil Perhitungan Pembaruan Feromon

Kota Asal	Kota Tujuan	Feromon Baru
A	B	5,2436
A	C	5,2436
B	C	5,2436

Karena jarak kota A ke kota B berbeda dengan jarak B ke kota A, maka kita memiliki proses penyelesaian yang lebih panjang dari penyelesaian soal TSP simetris. Berikutnya, kita akan mencari jarak antar kota untuk nilai yang berbeda (menurun).

6) Membangkitkan bilangan acak untuk menentukan titik (kota) berikutnya
Hitung pembagi semut 1 hingga 3, lalu menghitung probabilitas kota berikutnya yang akan dikunjungi, sehingga diperoleh hasil sebagai berikut:

Tabel 2. 9 Perhitungan Iterasi 1 (bagian 2) soal ACO pada TSP Asimetris

Semut	Kota saat ini	Probabilitas pilih kota		
		A	B	C
1	B	0,6250	0,0000	0,3750
2	D	0,0000	0,6667	0,3333
3	C	0,5455	0,4545	0,0000

Berikutnya bangkitakan bilangan acak untuk mennetukan kota berikutnya yang akan dikunjungi oleh semut.

7) Mengisi *tabu list*
Setelah menyelesaikan perhitungan dan pengisian *tabu list* akan diperoleh hasil seperti pada Tabel 2.10 berikut:

Tabel 2. 10 Rute dan Total Jarak

Semut	Rute				Total Jarak
1	B	C	A	B	13
2	A	C	B	A	12
3	C	B	A	C	13

8) Memperbarui feromon berdasarkan perhitungan kedua
Diperoleh hasil pada Tabel 2.11 sebagai berikut:

Tabel 2. 11 Hasil Perhitungan Pembaruan Feromon

Kota Asal	Kota Tujuan	Feromon Baru
A	B	5,2372
A	C	5,2372
B	C	5,2372

Karena jarak kota A ke kota B berbeda dengan jarak B ke kota A, maka kita memiliki proses penyelesaian yang lebih panjang dari penyelesaian soal TSP simetris. Dapat dilihat bahwa pada masalah TSP asimetris, penentuan awal keberangkatan akan mempengaruhi hasil yang akan diperoleh, disinilah letak kesulitan dalam proses penyelesaiannya. Karena kita harus sangat hati-hati dalam menentukan titik awal keberangkatan.

Pada perhitungan pertama dan kedua sama-sama menghasilkan total jarak terpendek adalah 12 km. Namun, jika diperhatikan terdapat perbedaan pada rute yang diperoleh. Pada perhitungan pertama, jika pengiriman dimulai dari kota A, akan menghasilkan rute A, B, C, A dengan total jarak 13 km. Sedangkan, pada perhitungan kedua, jika pengiriman dimulai dari kota A, akan menghasilkan rute A, C, B, A dengan total jarak 12 km. Perbedaan pada total jarak ini juga mempengaruhi hasil dari pembaruan feromon, karena pembaruan feromon dari perhitungan pertama dan kedua juga memiliki perbedaan. Sehingga dapat ditarik kesimpulan bahwa rute pengiriman yang paling efisien adalah A, C, B, A dengan jarak terpendek adalah rute dengan total jarak 12 km.

2.6 Pyhton

Menurut Donald Ervin Knuth, algoritma adalah sekumpulan aturan berhingga yang memberikan sederetan operasi-operasi untuk menyelesaikan suatu jenis masalahh yang khusus. Dalam konteks pemrograman, algoritma seringkali diterapkan dalam bahasa pemrograman, salah satunya adalah Python. Python dibuat oleh Guido Van Rossum pada tahun 1991 dan dikembangkan lebih lanjut oleh Python Software Foundation. Python adalah bahasa pemrograman tingkat tinggi adaptif yang paling banyak digunakan (Saputra dkk, 2020). Pada penelitian ini, Python akan digunakan untuk membangkitkan matriks yang akan digunakan sebagai data input dalam penelitian ini dan juga akan diterapkan pada algoritma *Greedy* dan ACO untuk masalah TSP simetris dan asimetris.