

# TRAVELING SALESMAN PROBLEM

Algoritma Ant Colony Optimization

**Nur Afika Salsabila**

NIM : D0222325

KELAS : INFORMATIKA A

# APA ITU TRAVELING SALESMAN PROBLEM?

TSP (Travelling Salesman Problem) adalah sebuah permasalahan dalam bidang matematika dan ilmu komputer yang mencari rute terpendek untuk seorang penjual yang harus mengunjungi sejumlah kota dengan kondisi tertentu. Tujuan utamanya adalah mencari rute terpendek yang melewati setiap kota sekali dan kembali ke kota awal.

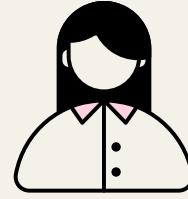
# SEJARAH ALGORITMA ANT COLONY OPTIMIZATION

ACO (Ant Colony Optimization) adalah algoritma yang terinspirasi oleh perilaku koloni semut dalam mencari jalur terpendek antara sarang mereka dan sumber makanan. Algoritma ini pertama kali diperkenalkan oleh Marco Dorigo pada tahun 1992 dalam tesisnya yang berjudul "Optimization, Learning and Natural Algorithms".

Konsep utama dalam ACO adalah bahwa semut individu mengikuti jalur feromon yang ditinggalkan oleh semut lain saat mereka mencari makanan.

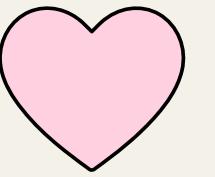
Pada awalnya, setiap semut dalam simulasi ACO diberikan kesempatan untuk memilih jalur secara acak. Namun, seiring berjalannya waktu, semut-semut ini belajar untuk mengikuti jalur yang memiliki tingkat feromon yang lebih tinggi. Ketika semut mencapai sumber makanan, mereka kembali ke sarang mereka, dan sebagian feromon yang mereka bawa kembali ke jalur yang mereka lewati.

# STRUKTUR DATA YANG DIGUNAKAN



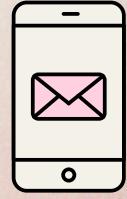
Array dua dimensi:  
adjacency, pheromone,  
dan visibility

Digunakan untuk  
menyimpan informasi  
tentang jarak antara kota-  
kota, tingkat pheromone,  
dan tingkat visibility antara  
kota-kota.



Stack: visited

Digunakan untuk melacak  
jalur yang telah dilewati  
oleh semut. Kota-kota yang  
dikunjungi oleh semut  
ditambahkan ke tumpukan  
ini saat semut bergerak dari  
satu kota ke kota lainnya.



ArrayList: bestSolution

ArrayList yang digunakan  
untuk menyimpan solusi  
terbaik yang ditemukan  
oleh algoritma ACO. Ketika  
ditemukan jalur yang lebih  
pendek, ArrayList ini  
diperbarui dengan jalur  
yang baru.

# DESAIN ALGORITMA

>>>

## Inisialisasi:

- Pada bagian ini, jumlah vertex (kota) ditentukan dan adjacency matrix digunakan untuk merepresentasikan jarak antara setiap pasangan vertex.
- Nilai-nilai awal seperti jumlah semut ( $S$ ), jumlah siklus semut maksimum ( $NCMax$ ), konstanta pengendali pheromone ( $\alpha$ ), konstanta pengendali intensitas visibilitas ( $\beta$ ), konstanta penguapan pheromone ( $\rho$ ), dan konstanta siklus semut ( $Q$ ) juga ditetapkan.
- Array pheromone dan visibility diinisialisasi dengan nilai awal yang sesuai.

## Loop Siklus Semut:

- Di dalam loop ini, pencarian dilakukan oleh setiap semut (sebanyak  $S$ ).
- Masing-masing semut memilih kota tujuan berdasarkan probabilitas yang dihitung menggunakan nilai pheromone dan visibility.
- Jika jalur yang ditempuh oleh semut membentuk solusi yang lebih baik (jarak yang lebih pendek), solusi tersebut disimpan sebagai solusi terbaik.

## Perhitungan Delta Tau:

- Setelah semua semut melakukan pencarian, deltaTau (peningkatan pheromone) dihitung berdasarkan solusi yang ditemukan oleh masing-masing semut.
- DeltaTau diupdate dengan membagi nilai Q (konstanta siklus semut) dengan panjang jalur ( $L_k$ ) yang ditempuh oleh semut.
- DeltaTau disimpan dalam matriks deltaTau untuk digunakan dalam langkah selanjutnya.

## Output:

- Setelah seluruh siklus semut selesai, solusi terbaik (jalur terpendek) dan jaraknya dicetak.

TERIMA KASIH