POLITEKNIK ELEKTRONIKA NEGERI SURABAYA

DIKTISAINTEK BERDAMPAK KAMPUS INOVASI



Traveling Salesman Problem

Sistem Pendukung Keputusan

Yunia Ikawati

Referensi: Tri Hadiah Muliawati

Teknik Informatika-PENS

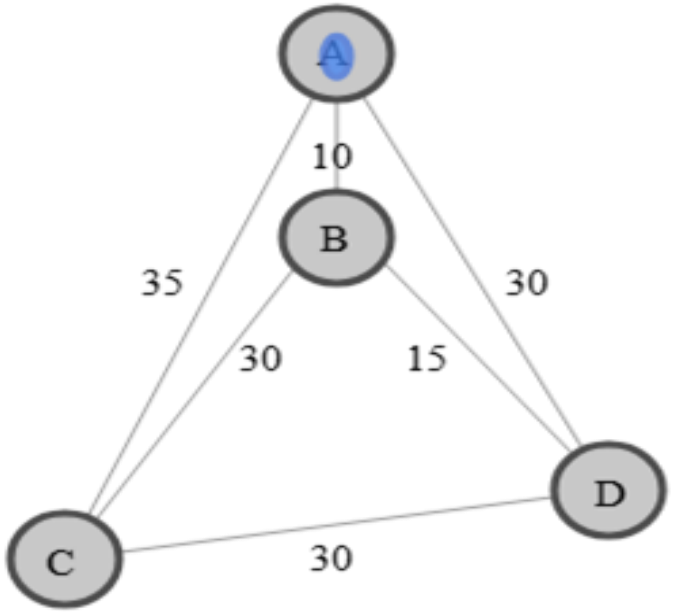
Overview

- TSP merupakan salah satu bagian dari permasalahan penentuan rute kendaraan (*vehicle routing*), dimana tujuan utamanya adalah menemukan rute terbaik untuk kendaraan dalam mengunjungi semua titik tujuan (*node* atau *vertex*) dimana masing-masing titik tujuan hanya boleh dikunjungi satu kali.
- Rute terbaik didefinisikan sebagai rute dengan total jarak atau biaya terendah.
- Komponen TSP:
  - Node atau vertex: Titik tujuan yang perlu dikunjungi
  - Edge: jalur antara satu titik tujuan dengan titik tujuan lainnya
  - Weight: Bobot atau biaya yang dibutuhkan dari satu titik tujuan ke titik tujuan lainnya

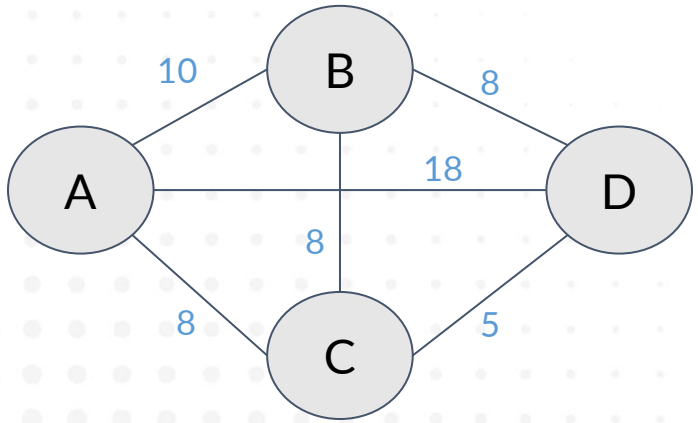
Overview

Graph di samping menggambarkan ilustrasi studi kasus TSP. Titik tujuan (node atau vertex) yang tersedia adalah A, B, C, dan D dimana terdapat 6 jalur (edge) yang menghubungkan keempat titik tersebut. Adapun bobot antar titik tujuan (weight) dapat dituliskan sbb:

- w(A, B) = w(B, A) = 10
- w(A, C) = w(C, A) = 35
- w(B, C) = w(C, B) = 30
- w(B, D) = w(D, B) = 15
- w(D, C) = w(C, D) = 30
- w(A, D) = w(D, A) = 30

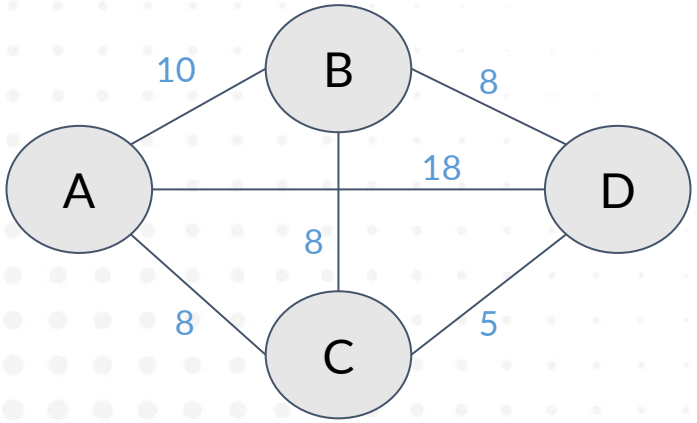


Symmetric TSP



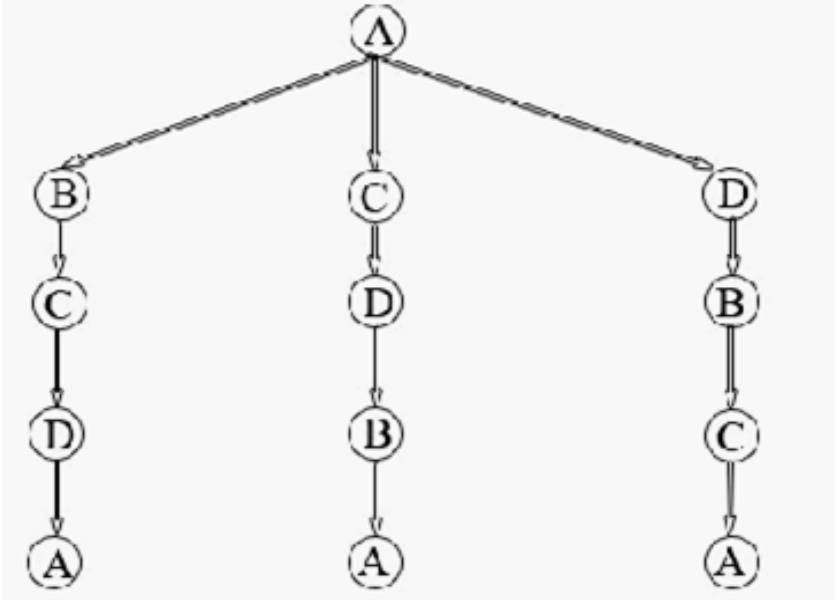
- Pada symmetric TSP, graph bersifat *undirected* (tidak memiliki arah). Oleh karena itu, bobot untuk *edge* dari node A ke B akan sama dengan bobot untuk edge dari node B ke A.
  w(A, B) = w(B,A)
- Total maksimal alternatif rute yang tersedia untuk mengunjungi semua node N adalah = (N-1)!/2

Symmetric TSP

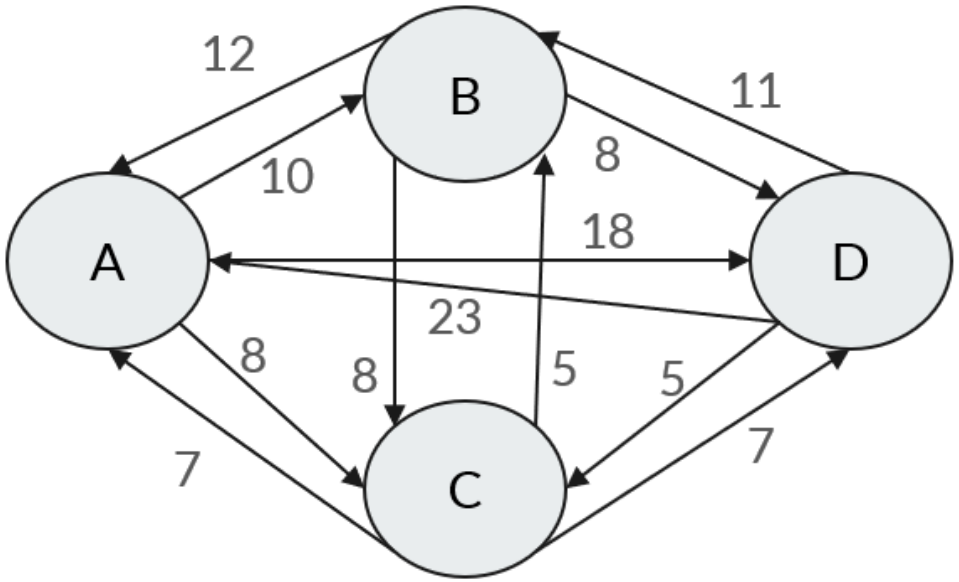


Untuk N=4, Total maksimal alternatif rute yang tersedia adalah

= (4-1)!/2 = 3!/2 = 6/2 = 3

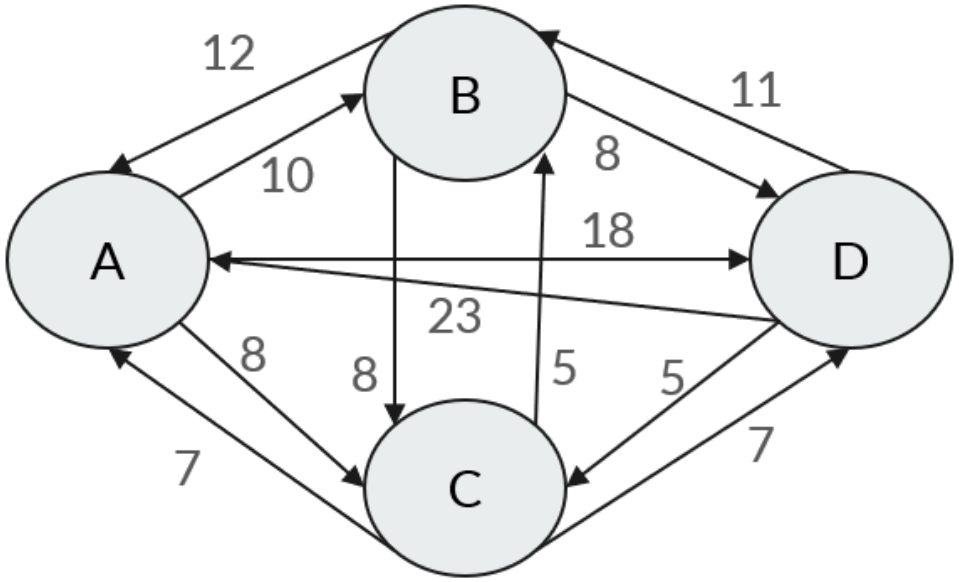


Asymmetric TSP



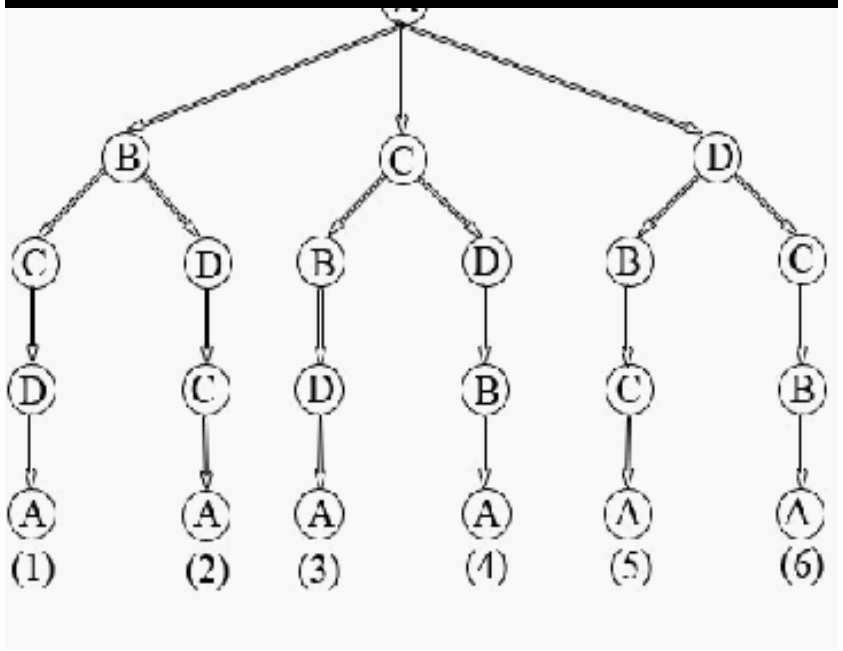
- Pada asymmetric TSP, graph bersifat *directed* (memiliki arah). Oleh karena itu, bobot untuk *edge* dari node A ke B tidak selalu sama dengan bobot untuk edge dari node B ke A.
  w(A, B) != w(B,A)
- Total maksimal alternatif rute yang tersedia untuk mengunjungi semua node N adalah = (N-1)!

Asymmetric TSP



Untuk N=4, Total maksimal alternatif rute yang tersedia adalah

= (4-1)! = 3! = 6



NP-Hard Problem

TSP termasuk dalam kategori NP (Non deterministic polynomial time)-hard problem karena waktu yang dibutuhkan untuk menyelesaikan masalah bertambah secara faktorial seiring dengan bertambahnya jumlah titik (node) apabila penyelesaian masalah dilakukan menggunakan metode eksak seperti brute force.

N	Total rute (Asymmetric TSP)	Total rute (Symmetric TSP)
2	1	1
3	2	1
4	6	3
5	24	12
6	120	60
7	720	360
8	5.040	2.520
9	40.320	20.160
10	362.880	181.440

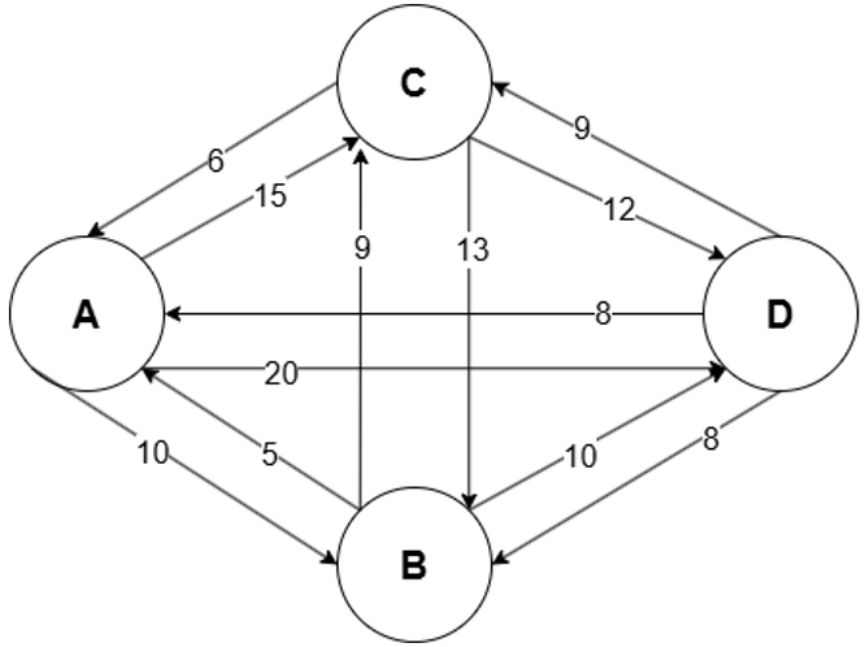
Metode Penyelesaian

- Eksak:
  - Brute force
  - Branch and bound
  - Dynamic programming
- *Heuristic*:
  - Nearest neighbors
  - Cheapest Insertion
  - dsb
- *Metaheuristic*:
  - Algoritma genetika
  - Particle Swarm Optimization
  - dsb

Studi Kasus

Seorang kurir bernama Dika bekerja untuk sebuah perusahaan pengiriman di kota Surabaya (A). Setiap hari, Dika harus mengantarkan paket ke tiga lokasi berbeda sebelum kembali ke kantor pusat. Namun, karena kondisi jalan dan arah lalu lintas, jarak antara setiap lokasi tidak selalu sama dalam dua arah.

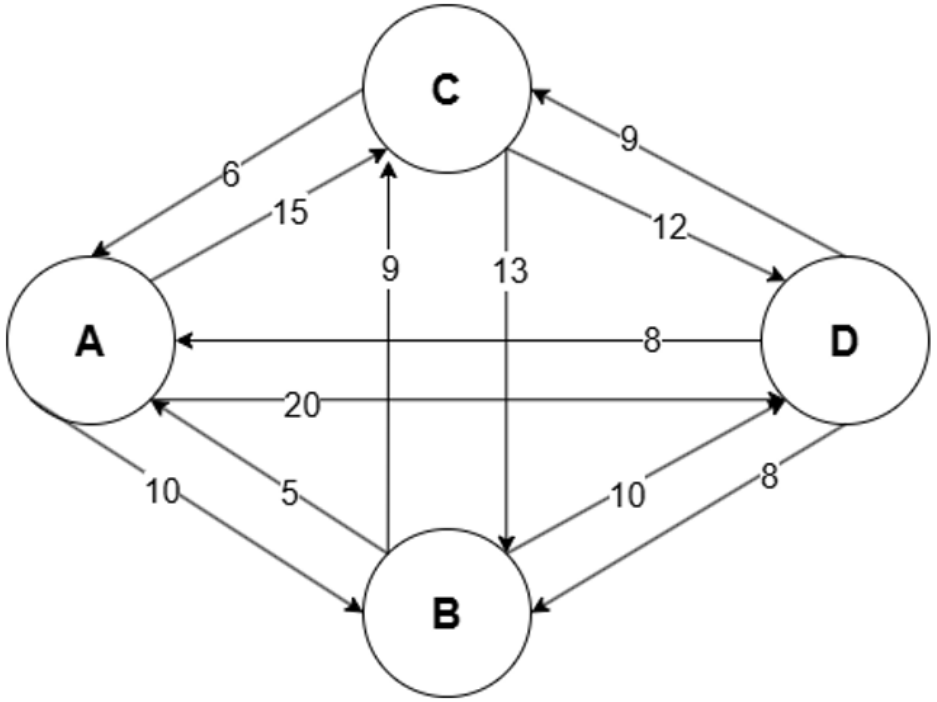
Ilustrasi jarak antar kota digambarkan pada graph di samping.



Studi Kasus: Matriks Representasi Biaya

Matriks representasi biaya atau cost matrix dari studi kasus tsb dapat digambarkan sbb:

	A	B	C	D
A	-	10	15	20
B	5	-	9	10
C	6	13	-	12
D	8	8	9	-



Solusi: Metode Eksak - Brute Force

Metode ini mencoba semua kemungkinan solusi untuk menemukan hasil terbaik. Metode ini akan mengevaluasi setiap kemungkinan rute yang mengunjungi semua kota sekali dan kembali ke titik awal, lalu memilih rute dengan jarak minimum.

Metode ini menjamin solusi optimal tetapi memiliki kompleksitas O(n!), sehingga jumlah perhitungan meningkat sangat cepat seiring bertambahnya jumlah kota (node) yang dikunjungi.

Opsi rute yang tersedia:

1. A → B → C → D → A = 10 + 9 + 12 + 8 = 39
2. A → B → D → C → A = 10 + 10 + 9 + 6 = 35
3. A → C → B → D → A = 15 + 13 + 10 + 8 = 46
4. A → C → D → B → A = 15 + 12 + 8 + 5 = 40
5. A → D → B → C → A = 20 + 8 + 9 + 6 = 43
6. A → D → C → B → A = 20 + 9 + 13 + 5 = 47

Rute terbaik adalah A → B → D → C → A dengan total cost 35.

Solusi: Metode Heuristic - Cheapest Insertion

Metode ini bekerja dengan prinsip menyisipkan kota yang belum dikunjungi ke dalam rute yang sudah ada dengan biaya tambahan terkecil (cheapest cost increase).

Langkah-langkah penyelesaian meliputi:

1. Pilih kota awal dan akhir yang akan dikunjungi.
2. Cari kota yang belum dikunjungi:
   - ✓ Untuk setiap kota yang belum masuk dalam rute, hitung biaya penambahan kota tersebut ke setiap sisi (edge) dalam rute saat ini.
   - ✓ Sisipkan kota dengan biaya tambahan terkecil:
3. Pilih kota dan posisi penyisipan yang menghasilkan peningkatan biaya paling kecil.
4. Ulangi langkah 2–3 sampai semua kota dimasukkan ke dalam rute.

Solusi: Metode Heuristic - Cheapest Insertion

- Kota awal dan akhir adalah: A
- Opsi rute dari dan ke A adalah:
  - A-B-A = 10 + 5 = 15 (biaya terkecil)
  - A-C-A = 15 + 6 = 21
  - A-D-A = 20 + 8 = 28
- Rute yang terpilih adalah A-B-A karena memiliki total biaya terkecil.

	A	B	C	D
A	-	10	15	20
B	5	-	9	10
C	6	13	-	12
D	8	8	9	-

Solusi: Metode Heuristic - Cheapest Insertion

- Kota awal dan akhir adalah: A
- Rute sementara: A-B-A
- Opsi rute yang bisa disisipkan adalah:
  - A-C-B-A = 15+13+5 =33
  - A-D-B-A = 20+8+5 =33
  - A-B-C-A = 10+9+6 =25 (biaya terkecil)
  - A-B-D-A = 10+10+8 =28
- Rute yang terpilih adalah A-B-C-A karena memiliki total biaya terkecil.

	A	B	C	D
A	-	10	15	20
B	5	-	9	10
C	6	13	-	12
D	8	8	9	-

Solusi: Metode Heuristic - Cheapest Insertion

- Kota awal dan akhir adalah: A
- Rute sementara: A-B-C-A
- Opsi rute yang bisa disisipkan adalah:
  - A-D-B-C-A = 20+8+8+6 =42
  - A-B-D-C-A = 10+10+9+6 =35 (biaya terkecil)
  - A-B-C-D-A = 10+9+12+8 =39
- Rute yang terpilih adalah A-B-D-C-A karena memiliki total biaya terkecil.
- Karena semua kota sudah dikunjungi, maka rute akhir yang direkomendasikan adalah A-B-D-C-A dengan total biaya 35.

	A	B	C	D
A	-	10	15	20
B	5	-	9	10
C	6	13	-	12
D	8	8	9	-

Solusi: Metode Heuristic - Nearest Neighbor

Metode Nearest Neighbor memilih kota terdekat yang belum dikunjungi pada setiap langkah. Algoritma ini bekerja sebagai berikut:

1. Mulai dari kota awal yang telah ditentukan.
2. Pilih kota terdekat yang belum dikunjungi.
3. Ulangi proses hingga semua kota telah dikunjungi.
4. Kembali ke kota awal setelah semua kota dikunjungi.

Metode ini cepat dan mudah diterapkan, tetapi tidak selalu menghasilkan solusi optimal karena bisa menghasilkan rute yang kurang efisien. Kompleksitas waktu algoritma ini adalah $O(n^2)$, sehingga masih cukup efisien untuk jumlah kota yang tidak terlalu besar (apabila dibandingkan dengan metode brute force).

Solusi: Metode Heuristic - Nearest Neighbor

- Kota dengan biaya terkecil dari A adalah B (10). Total biaya sementara = 10
- Kota dengan biaya terkecil dari B (selain A) adalah C (9). Total biaya sementara = 10 + 9 = 19
- Kota dengan biaya terkecil dari C (selain A dan B) adalah D (12). Total biaya sementara = 10 + 9 + 12 = 31
- Seluruh kota sudah dikunjungi, maka kembali ke A dari D (8). Total biaya akhir = 10 + 9 + 12 + 8 = 39

Rute terbaik adalah A → B → C → D → A dengan total cost 39.

	A	B	C	D
A	-	10	15	20
B	5	-	9	10
C	6	13	-	12
D	8	8	9	-

Pada studi kasus ini solusi metode heuristic: Nearest Neighbor tidak lebih baik dari solusi metode eksak: Brute Force karena metode heuristic bisa terjebak pada *local optima*. Akan tetapi pada studi kasus dengan jumlah node yang lebih banyak, solusi ini akan menghasilkan solusi yang cukup baik dengan waktu yang lebih cepat apabila dibandingkan dengan metode eksak.

Solusi: Metode Metaheuristic - Algoritma Genetika

Metode Algoritma Genetika bekerja dengan membentuk populasi awal dari solusi acak, kemudian melalui proses seleksi dan reproduksi (crossover dan mutasi), solusi terbaik dipertahankan dan diperbaiki seiring generasi. Tahapan Algoritma genetika meliputi:

1. Inisialisasi Populasi awal – Membentuk sekumpulan individu yang berisi kumpulan solusi awal yang dibentuk (di-generate) secara acak. Solusi yang dimaksud adalah opsi rute yang tersedia.
2. Evaluasi Fitness – Pada TSP nilai fitness didapatkan dengan menghitung total biaya perjalanan untuk setiap individu / solusi / rute pada populasi.
3. Seleksi individu – Memilih individu / solusi / rute terbaik berdasarkan nilai fitness untuk digunakan sebagai dasar membentuk individu / solusi / rute baru. Contoh metode seleksi roulette wheel atau tournament selection.
4. Reproduksi - Menghasilkan individu / solusi / rute baru berdasarkan individu / solusi / rute yang terpilih pada tahap sebelumnya.

a. Crossover – Menggabungkan dua individu / solusi / rute untuk menghasilkan individu / solusi / rute baru. Contoh metode Crossover yang bisa digunakan adalah Order Crossover (OX)
b. Mutasi – Mengubah sebagian kecil gen dari individu / solusi / rute untuk menjaga keberagaman. Contoh metode Mutasi yang bisa digunakan adalah Swap Mutation.

Pendekatan ini tidak menjamin solusi optimal tetapi sering memberikan hasil yang sangat baik dalam waktu yang lebih singkat dibandingkan metode eksak. Referensi: Link, Link, Link

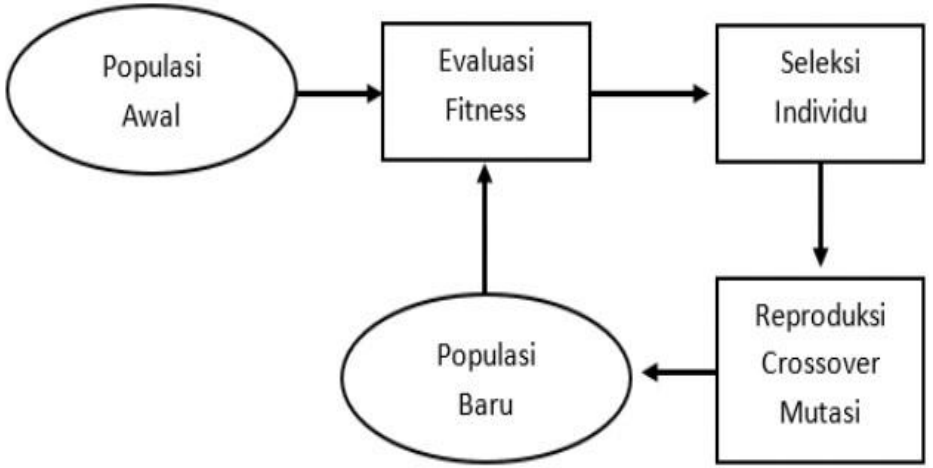
Solusi: Metode Metaheuristic - Algoritma Genetika

Secara visual siklus Algoritma Genetika bisa digambarkan sebagaimana gambar di samping.

Siklus akan terus berulang hingga individu / solusi / rute terbaik sudah ditemukan atau iterasi mencapai jumlah maksimal yang sudah ditentukan.

Penentuan populasi baru yang digunakan pada iterasi ke-2 dan seterusnya bisa menggunakan beberapa pendekatan, antara lain:

- Populasi baru berisi individu terbaik dari hasil reproduksi saja
- Populasi baru berisi individu terbaik dari gabungan hasil reproduksi dan populasi pada iterasi sebelumnya



Solusi: Metode Metaheuristic - Algoritma Genetika

Iterasi ke-1

1. Inisialisasi populasi awal (misal 4 individu / solusi / rute)
- A-B-C-D-A
- A-C-D-B-A
- A-C-B-D-A
- A-D-B-C-A

2. Evaluasi fitness tiap individu / solusi / rute
- A-B-C-D-A = 10 + 9 + 12 + 8 = 39
- A-C-D-B-A = 15 + 12 + 8 + 5 = 40
- A-C-B-D-A = 15 + 13 + 10 + 8 = 46
- A-D-B-C-A = 20 + 8 + 9 + 6 = 43

	A	B	C	D
A	-	10	15	20
B	5	-	9	10
C	6	13	-	12
D	8	8	9	-

3. Seleksi
Misal memilih 2 individu / solusi / rute terbaik dari populasi untuk menghasilkan individu / solusi / rute baru, maka individu / solusi / rute yang terpilih sebagai parent adalah:

- A-B-C-D-A
- A-C-D-B-A

Solusi: Metode Metaheuristic - Algoritma Genetika

Iterasi ke-1

4a. Reproduksi: Crossover

- Parent 1: A-B-C-D-A
- Parent 2: A-C-D-B-A

Misal node yang akan ditukar adalah node ke-3 (Node ke-1 dan 5 tidak boleh ditukar karena merupakan node awal dan akhir), maka ada 2 individu / solusi / rute baru yang dihasilkan. Penentuan titik / node yang akan ditukar bisa dilakukan secara random.

	A	B	C	D
A	-	10	15	20
B	5	-	9	10
C	6	13	-	12
D	8	8	9	-

Solusi baru yang dihasilkan adalah:

- Child 1: A-B-D-C-A
  Node ke-4 digantikan dengan C karena Node yang dikunjungi tidak boleh berulang.
- Child 2: A-D-C-B-A
  Node ke-2 digantikan dengan D karena Node yang dikunjungi tidak boleh berulang.

Solusi: Metode Metaheuristic - Algoritma Genetika

Iterasi ke-1

4a. Reproduksi: Crossover

Crossover diterapkan pada solusi secara berpasangan. Jumlah individu / solusi / rute yang mengalami crossover sangat bergantung pada Crossover Rate (CR). Semakin besar CR maka semakin besar pula probabilitas individu / solusi / rute yang akan dikenai crossover.

Misal:
CR = 0,9 maka 90% pasangan individu / solusi / rute akan mengalami crossover.

	A	B	C	D
A	-	10	15	20
B	5	-	9	10
C	6	13	-	12
D	8	8	9	-

CR umumnya berada di rentang 0,5 - 0,9. Hal ini bertujuan untuk mengoptimalkan fungsi eksplorasi sehingga diharapkan individu / solusi / rute baru yang dihasilkan akan lebih baik dari opsi individu / solusi / rute yang tersedia pada iterasi sebelumnya.

Solusi: Metode Metaheuristic - Algoritma Genetika

Iterasi ke-1

4b. Reproduksi: Mutasi

Berbeda dengan crossover, mutasi tidak perlu dilakukan secara berpasangan. Ilustrasinya sebagai berikut:

Parent 1: A-B-C-D-A
Misal node yang akan ditukar adalah node ke-2 dan ke-3 (Node ke-1 dan 5 tidak boleh ditukar karena merupakan node awal dan akhir). Penentuan titik / node yang akan ditukar bisa dilakukan secara random.

	A	B	C	D
A	-	10	15	20
B	5	-	9	10
C	6	13	-	12
D	8	8	9	-

Hasil mutasi:

Child 1: A-C-B-D-A
Apabila diamati individu / solusi / rute baru yang dihasilkan sama dengan opsi individu / solusi / rute ke-3 pada populasi awal. Hal ini mungkin terjadi, khususnya pada permasalahan dengan opsi individu / solusi / rute yang terbatas.

Solusi: Metode Metaheuristic - Algoritma Genetika

Iterasi ke-1

4b. Reproduksi: Mutasi

Parent 2: A-C-D-B-A
Misal node yang akan ditukar adalah node ke-2 dan ke-3 (Node ke-1 dan 5 tidak boleh ditukar karena merupakan node awal dan akhir). Penentuan titik / node yang akan ditukar bisa dilakukan secara random. Berikut individu / solusi / rute baru yang dihasilkan:

Child 2: A-D-C-B-A

	A	B	C	D
A	-	10	15	20
B	5	-	9	10
C	6	13	-	12
D	8	8	9	-

Solusi: Metode Metaheuristic - Algoritma Genetika

Iterasi ke-1

4b. Reproduksi: Mutasi

Jumlah mutasi bisa diterapkan pada opsi solusi sangat bergantung pada Mutation Rate (MR). Semakin besar MR maka semakin besar pula probabilitas solusi yang akan dikenai mutasi.

Misal:
MR = 0,05 maka 5% kemungkinan gen akan dimutasi.

	A	B	C	D
A	-	10	15	20
B	5	-	9	10
C	6	13	-	12
D	8	8	9	-

Berbeda dengan CR, MR umumnya diset cukup rendah (< 0.1). Hal ini bertujuan untuk menjaga stabilitas, tetapi tetap ada kemungkinan untuk bisa terbebas dari permasalahan local optima.

MR ditujukan untuk mengontrol fungsi eksploitasi dari algoritma genetika.

Solusi: Metode Metaheuristic - Algoritma Genetika

Iterasi ke-1

Berdasarkan hasil reproduksi, maka nilai fitness individu / solusi / rute baru yang dihasilkan adalah:

- A-B-D-C-A = 10 + 10 + 9 + 6 = 35
- A-D-C-B-A = 20 + 9 + 13 + 5 = 47
- A-C-B-D-A = 15 + 13 + 10 + 8 = 46
- A-D-C-B-A = 20 + 9 + 13 + 5 = 47

Di sisi lain, berikut nilai fitness individu / solusi / rute pada populasi awal

- A-B-C-D-A = 10 + 9 + 12 + 8 = 39
- A-C-D-B-A = 15 + 12 + 8 + 5 = 40
- A-C-B-D-A = 15 + 13 + 10 + 8 = 46
- A-D-B-C-A = 20 + 8 + 9 + 6 = 43

	A	B	C	D
A	-	10	15	20
B	5	-	9	10
C	6	13	-	12
D	8	8	9	-

Apabila diamati, tidak semua individu / solusi / rute baru yang dihasilkan lebih baik daripada individu / solusi / rute pada populasi awal. Hal ini bisa saja terjadi karena sifat stokastik (acak) yang dimiliki oleh Algoritma Genetika sebagaimana metode metaheuristic lainnya. Berdasarkan hasil sementara, individu / solusi / rute terbaik pada iterasi ke-1 adalah A-B-D-C-A dengan total biaya 35.

Solusi: Metode Metaheuristic - Algoritma Genetika

Iterasi ke-2

Pada iterasi ke-2, populasi baru berisi individu terbaik dari gabungan hasil reproduksi dan populasi iterasi sebelumnya. Berikut merupakan individu dari populasi baru dengan nilai fitness terbaik:

1. A-B-D-C-A = 10 + 10 + 9 + 6 = 35
2. A-B-C-D-A = 10 + 9 + 12 + 8 = 39
3. A-C-D-B-A = 15 + 12 + 8 + 5 = 40
4. A-D-B-C-A = 20 + 8 + 9 + 6 = 43

	A	B	C	D
A	-	10	15	20
B	5	-	9	10
C	6	13	-	12
D	8	8	9	-

Misal dilakukan crossover pada individu ke-1 (A-B-D-C-A) dan individu ke-2 (A-B-C-D-A) pada node ke-2 dan ke-3. Maka individu baru yang dihasilkan adalah:

- Child 1: A-B-C-D-A
- Child 2: A-B-D-C-A

Solusi: Metode Metaheuristic - Algoritma Genetika

Iterasi ke-2

Misal dilakukan Mutasi pada individu ke-1 (A-B-D-C-A) dan individu ke-2 (A-B-C-D-A) pada node ke-4 dan ke-2. Maka individu baru yang dihasilkan adalah:

- Child 1: A-C-D-B-A
- Child 2: A-D-C-B-A

	A	B	C	D
A	-	10	15	20
B	5	-	9	10
C	6	13	-	12
D	8	8	9	-

Solusi: Metode Metaheuristic - Algoritma Genetika

Iterasi ke-2

Berdasarkan hasil reproduksi, maka nilai fitness individu / solusi / rute baru yang dihasilkan adalah:

- A-B-C-D-A = 10 + 9 + 12 + 8 = 39
- A-B-D-C-A = 10 + 10 + 9 + 6 = 35
- A-C-D-B-A = 15 + 12 + 8 + 5 = 40
- A-D-C-B-A = 20 + 9 + 13 + 5 = 47

Di sisi lain, berikut nilai fitness individu / solusi / rute pada populasi iterasi ke-2

- A-B-D-C-A = 10 + 10 + 9 + 6 = 35
- A-B-C-D-A = 10 + 9 + 12 + 8 = 39
- A-C-D-B-A = 15 + 12 + 8 + 5 = 40
- A-D-B-C-A = 20 + 8 + 9 + 6 = 43

	A	B	C	D
A	-	10	15	20
B	5	-	9	10
C	6	13	-	12
D	8	8	9	-

Apabila diamati, individu / solusi / rute terbaik hasil reproduksi iterasi ke-2 sama dengan individu / solusi / rute terbaik populasi iterasi ke-2. Hal ini menunjukkan solusi sudah optimal sehingga siklus algoritma genetika bisa dihentikan.

Berdasarkan metode algoritma genetika, individu / solusi / rute terbaik adalah A-B-D-C-A dengan total biaya 35.

Studi Kasus: Solusi

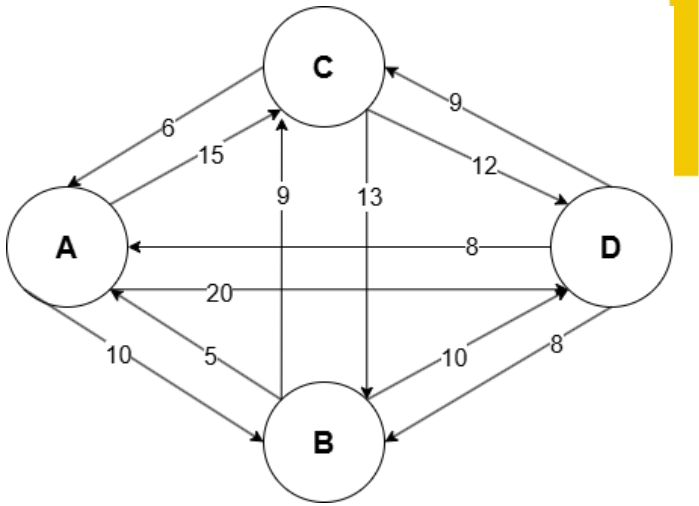
Berdasarkan uji coba yang telah dilakukan, pada studi kasus berikut metode brute force dan algoritma genetika (metaheuristic) berhasil menemukan rute terbaik, yakni A-B-D-C-A dengan total biaya 35.

Sedangkan, metode nearest neighbor (heuristic) menemukan rute yang cukup baik, yakni A-B-C-D-A dengan total biaya 39.

Meskipun metode nearest neighbor (heuristic) belum berhasil menemukan rute terbaik, solusi yang dihasilkan sudah cukup mendekati solusi terbaik dengan waktu perhitungan yang jauh lebih cepat dibandingkan dengan algoritma genetika (metaheuristic).

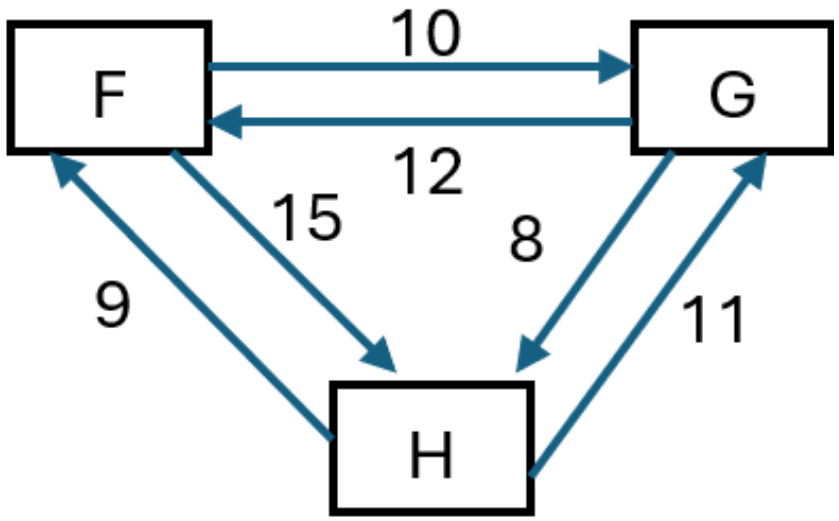
Di sisi lain, metode algoritma genetika (metaheuristic) berhasil menemukan rute terbaik, tapi waktu yang dibutuhkan untuk menemukan solusi tersebut lebih lama daripada metode bruteforce (eksak) pada studi kasus dengan jumlah node yang sedikit.

Untuk studi kasus yang sederhana, penggunaan metode eksak masih menjadi salah satu opsi terbaik untuk menemukan solusi yang optimal dengan waktu yang relatif tidak terlalu lama. Keunggulan dari metode heuristic dan metaheuristic akan lebih terlihat pada studi kasus dengan jumlah node yang lebih banyak.



LATIHAN

- Seorang salesman harus mengunjungi 3 kota: F, G, dan H. Ia harus memulai dari satu kota, mengunjungi semua kota lainnya tepat satu kali, dan kembali ke kota asal. Namun, biaya perjalanan antar kota bersifat *asymmetric* — artinya, biaya dari F ke G tidak sama dengan dari G ke F.



References

- https://developers.google.com/optimization/routing
- https://www.cs.ubc.ca/labs/algorithms/Courses/CPSC532D-05/Slides/tsp-camilo.pdf