ALGORITMA OPTIMASI UNTUK PENYELESAIAN TRAVELLING SALESMAN PROBLEM

(Optimization Algorithm for Solving Travelling Salesman Problem)

Dian Tri Wiyanti
Program Studi Teknik Informatika, Jurusan Teknologi Informasi
Fakultas Teknologi Informasi dan Komunikasi, Universitas Semarang
Email: dian@usm.ac.id

Abstract

Travelling Salesman Problem (TSP) is still a hot topic for discussion. TSP includes part of optimization problems in a real world. In the TSP, there are cities that must be passed by a salesman, then back to the city where he first set out. In that way, a salesman should be choose the shortest route. There are many algorithms for solving TSP problem. And among the many algorithms, in this study will be discussed on how the algorithm implementation of Greedy, Artificial Bee Colony (ABC), Cheapest Insertion Heuristics (CIH), and Genetics algorithm to resolve the case of TSP. Analysis that is performed are a comparison of methods, implementation of the TSP case, as well as the advantages and disadvantages of each algorithm

Keywords: *TSP, Algorithm Optimization, Comparison*

1. PENDAHULUAN

Travelling Salesman Problem (TSP) merupakan masalah klasik mencari rute terpendek yang bisa dilalui *salesman* ketika ingin mengunjungi beberapa kota tanpa harus mendatangi kota yang sama lebih dari satu kali [4]. Penyelesaian eksak untuk masalah TSP ini mengharuskan perhitungan terhadap semua kemungkinan rute yang dapat diperoleh, kemudian memilih salah satu rute yang terpendek.

Tinjauan Pustaka. Terdapat banyak algoritma untuk melakukan pencarian rute terpendek. Pemilihan algoritma yang paling optimum selalu menjadi permasalahan dalam pencarian rute terpendek, dimana masing-masing algoritma memiliki kelebihan dan kekurangannya masing-masing. Dalam lingkup pencarian rute terpendek ini tidak dapat dikatakan secara langsung algoritma mana yang paling optimum untuk keseluruhan kasus, karena belum tentu suatu algoritma yang memiliki optimasi yang tinggi untuk suatu kasus memiliki optimasi yang tinggi pula untuk kasus yang lain. Penelitian yang dilakukan oleh [5] memberikan kesimpulan yaitu secara umum algoritma genetika bekerja lebih baik daripada kedua algoritma yang lain, yaitu algoritma *Hopfield* dan *Exhaustive* ditinjau dari jarak yang dihasilkan serta waktu yang dibutuhkan untuk melakukan perhitungan pada algoritma. Sedangkan penelitian yang dilakukan [2] memilih menggunakan algoritma ABC karena sederhana dan fleksibel, memiliki kemampuan untuk keluar dari *local minimum* dan dapat secara efisiensi digunakan untuk multimodal dan multivariable optimasi fungsi. Kemudian penelitian yang dilakukan oleh [1], menyebutkan bahwa algoritma *greedy* lebih bagus apabila digunakan pada kurang dari 10 tempat tujuan. Karena hasil dari algoritma *greedy* dan dibanding dengan algoritma lain, algoritma *greedy* memiliki hasil optimasi yang lebih bagus dan dengan waktu komputasi yang lebih cepat. Selain itu, penelitian oleh [4] menyebutkan waktu proses dari algoritma CIH jauh lebih cepat, namun untuk memproses kota dalam jumlah besar CIH masih membutuhkan waktu yang cukup besar pula.

Kontribusi. Setiap algoritma memiliki kelebihan dan kekurangan masing-masing. Pembahasan dalam paper ini dilakukan mulai dari perbandingan metode, implementasinya terhadap kasus TSP, serta kelebihan dan kekurangan masing-masing algoritma untuk studi kasus pada TSP. Dengan demikian diharapkan akan menambah informasi pada *user* mengenai algoritma yang paling sesuai digunakan dengan solusi yang paling diinginkan untuk kasus TSP.

2. PEMBAHASAN

Beberapa hal yang menjadi perhatian dalam pembahasan ini mencakup 3 hal, yaitu :

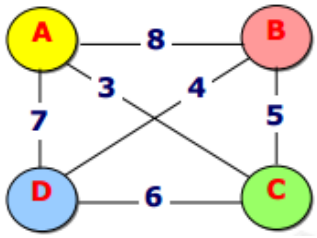
a. Konsep TSP
b. Konsep dan implementasi beberapa algoritma terhadap TSP
c. Kelebihan dan kekurangan masing-masing algoritma

2.1. Konsep TSP

Apa yang dilakukan dalam TSP adalah membentuk sebuah *tour*. Operator yang bisa digunakan untuk masalah TSP adalah pencarian urutan semua lokasi untuk memilih lokasi yang belum pernah terpilih satu demi satu sehingga dihasilkan satu rute kunjungan yang lengkap dari lokasi awal kemudian mengunjungi semua lokasi yang lain tepat satu kali dan akhirnya kembali ke lokasi awal. Sehingga dengan definisi tersebut dapat dikatakan bahwa konsep permasalahan TSP memiliki aturan sebagai berikut:

1. Harus mengunjungi setiap kota tepat satu kali, tidak boleh kurang ataupun lebih.
2. Semua kota harus dikunjungi dalam satu kali perjalanan (*tour*).
3. Dimulai dan diakhiri pada kota yang sama.

Sebagai ilustrasi dengan Gambar 1, diasumsikan bahwa simpul awal dan simpul akhir adalah 1. Suatu graf TSP dengan 4 simpul tersebut dikonversi menjadi sebuah pohon pencarian yang menghasilkan (4-1)! = 6 kemungkinan urutan kunjungan. Sedangkan untuk 50 kota, terdapat sebanyak (50-1)! = $6{,}08\times10^{62}$ kemungkinan urutan kunjungan.



Gambar 1. Contoh Graf Rute Perjalanan

Berdasarkan [6], TSP dikatakan ada 2 jenis, yaitu:

1. TSP asimetris

Pada TSP 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. TSP simetris

Sedangkan pada TSP 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.

2.2. Konsep dan Implementasi Beberapa Algoritma Terhadap TSP

2.2.1. 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 akna 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. Hal ini telah dibuktikan dalam penelitian oleh [1] yang mengaplikasikan algoritma ini terhadap layanan taksi wisata, dimana hasil implementasi algoritma *greedy* ini dikhususkan pada kasus TSP yang jarak antar node-nodenya pendek.

2.2.2. 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. Ada 3 tahapan utama pada *basic* algoritma ABC, yaitu :

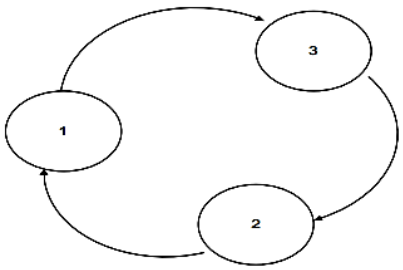
1. Menghasilkan inisial solusi dari sumber makanan secara acak.
   Untuk memperbarui solusi yang mungkin, setiap *employed bee* memilih calon posisi sumber makanan baru, yang mana posisi tersebut berbeda dengan sebelumnya.
2. Setiap *onlooker bee* memilih salah satu sumber makanan yang diperoleh dari *employed bee*.
   Setelah memilih sumber makanan, *onlooker bee* pergi ke sumber makanan yang dipilih dan memilih sumber calon makanan baru.
3. 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*.

Dalam penelitian oleh [2], hasil yang didapatkan mencapai nilai optimal, dalam hal ini jarak terpendek. Namun sama seperti penelitian [1], permasalahan pada algoritma ABC yang diterapkan masih terpaku pada jumlah kota yang terbatas. Karena dalam penelitian [2] ini terdapat kesimpulan bahwa jika *size problem* semakin besar, dalam hal ini jumlah kota yang diproses, maka tingkat kesalahan juga semakin meningkat.

2.2.3. Algoritma Cheapest Insertion Heuristics

Kemudian dalam [4], algoritma CIH dikombinasikan dengan basis data. Dimana basis data digunakan sebagai penyimpanan data proses sehingga pengambilan informasi jarak minimal dari beberapa alternatif yang ada dapat dilakukan dengan mudah yaitu dengan menggunakan *query*. Konsep CIH sendiri memiliki algoritma sebagai berikut :

1. Penelusuran
   Dimulai dari sebuah kota pertama yang dihubungkan dengan sebuah kota terakhir.
2. 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 2.



Gambar 2. *Subtour*

3. 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:

$$c_{ik} + c_{kj} - c_{ij} \quad (1)$$

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

4. Ulangi langkah 3 sampai seluruh kota masuk dalam subtour.

2.2.4. Algoritma Genetika

Sedangkan dalam [5], algoritma genetika berhasil mengoptimalkan hasilnya. Terbukti untuk 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 hidup. Pada dasarnya ada 4 kondisi yang mempengaruhi proses evaluasi, yaitu :

1. Kemampuan organisme untuk melakukan reproduksi.
2. Keberadaan populasi organisme yang bisa melakukan reproduksi.
3. Keberagaman organisme dalam suatu populasi.
4. Perbedaan kekuatan dan kemampuan organisme untuk bertahan hidup.

Adapun langkah-langkah dari algoritma genetika adalah sebagai berikut :

1. Langkah pertama adalah melakukan penentuan nilai awal (inisialisasi). Bagian ini merupakan pemberian input yang dilakukan oleh pengguna.
2. Proses berikutnya adalah proses pembentukan populasi awal. Proses ini berfungsi untuk membentuk populasi generasi pertama.
3. Selanjutnya adalah proses seleksi, dimana setelah terbentuk populasi awal, maka hasil populasi awal itu akan diseleksi.
4. Setelah melakukan proses seleksi, maka hasil dari proses tersebut akan digunakan dalam proses *crossover*.
5. Sebelum melakukan proses *crossover* dilakukan pengundian dengan bilangan random untuk setiap kromosom, apakah kromosom tersebut terjadi *crossover* atau tidak.
6. Jika dari proses pengundian menunjukkan bahwa terjadi *crossover* maka akan dibuat bilangan random lain untuk menentukan dimana *crossover* akan terjadi.
7. 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.
8. Jika terjadi pengulangan individu dalam kromosom, maka dilakukan proses normalisasi untuk membuat kromosom tersebut menjadi valid.
9. 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.
10. Setelah proses mutasi, maka akan dilakukan lagi proses seleksi dan dilakukan pengecekan apakah proses keseluruhan telah selesai atau belum.

2.3. Kelebihan dan kekurangan masing-masing algoritma

Perbandingan untuk menyelesaikan kasus TSP dengan menggunakan algoritma *greedy*, *Artificial Bee Colony* (ABC), *Cheapest*

Insertion Heuristics (CIH), dan algoritma genetika ada pada Tabel 1 berikut.

Tabel 1. Perbandingan Algoritma

ALGORITMA	*Greedy*	ABC	CIH	Genetika
KELEBIHAN	Waktu komputasi yang dibutuhkan dalam menyelesaikan kasus TSP lebih cepat. Lebih sesuai untuk kasus yang membutuhkan solusi hampiran.	Mencapai nilai optimal apabila data pada kasus TSP merupakan data dengan *size* yang tidak terlalu besar.	Berbeda dengan algoritma ABC, algoritma ini masih stabil digunakan untuk kasus TSP dengan jumlah kota yang besar.	Waktu komputasi yang dibutuhkan cenderung stabil. Mampu memberikan jarak terpendek meski dengan jumlah kota yang besar, bila dibandingkan dengan algoritma yang lain.
KEKURANGAN	Hasil yang didapatkan tidak selalu optimal. Hal ini karena algoritma *greedy* masih terjebak dalam optimum lokal.	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.	Dengan kelebihan yang ada pada algoritma ini, banyaknya jumlah kota sangat berpengaruh pada waktu komputasi.	Sangat bergantung pada pemilihan parameter *input*, yaitu ukuran populasi, besar maksimum generasi, ukuran peluang *crossover*, dan ukuran peluang *mutation*.

3. KESIMPULAN

Berdasarkan asumsi yang ditinjau dari referensi-referensi yang ada, dapat disimpulkan bahwa tidak ada satupun algoritma yang berlaku umum dan bisa digunakan untuk menyelesaikan semua jenis masalah. Dengan apa yang disajikan oleh Tabel I, diharapkan peneliti dan praktisi dapat memilih algoritma optimasi yang paling tepat dan sesuai untuk digunakan, dalam hal ini pada kasus TSP.

DAFTAR PUSTAKA

[1] A.C. Purnomo, M. Yuliana, dan I. Prasetyaningrum, *Implementasi Algoritma Greedy Pada Layanan Taksi Wisata Berbasis Web*. [Online]. Available: www.eepis-its.edu/uploadta/downloadmk. [Accessed August 23, 2013].

[2] F. Amri, E.B. Nababan, M.F. Syahputra, *Artificial Bee Colony Algorithm untuk Menyelesaikan Travelling Salesman Problem*, Jurnal Dunia Teknologi Informasi Vol. 1, No. 1, hal. 8-13, 2012.

[3] J.J. Siang. *Riset Operasi Dalam Pendekatan Algoritmis*. Yogyakarta: Penerbit Andi, 2011.

[4] Kusrini, J.E. Istiyanto, *Penyelesaian Travelling Salesman Problem Dengan Algoritma Cheapest Insertion Heuristics Dan Basis Data*, Jurnal Informatika Vol. 8, No. 2, hal. 109 – 114, 2007.

[5] R.Adipranata, F. Soedjianto, dan W. Tjondro, *Perbandingan Algoritma*

Exhaustive, Algoritma Genetika Dan Algoritma Jaringan Syaraf Tiruan Hopfield Untuk Pencarian Rute Terpendek. [Online]. Available: http://fportfolio.petra.ac.id/user.../**Perbandingan**%20**Rute**%20**Terpendek**.pdf.

[6] Suyanto, *Algoritma Optimasi : Deterministik atau Probabilistik*, Yogyakarta: Graha Ilmu, 2010.