Algoritma DFS (*Depth First Search*) untuk Menyelesaikan Masalah TSP (*Travelling Salesperson Problem*)

Ali Akbar[1], Muhammad Arif Romdhoni[2], Masagus Krisna Ismaliansyah[3]

Laboratorium Ilmu dan Rekayasa Komputasi
Departemen Teknik Informatika, Institut Teknologi Bandung
Jl. Ganesha 10, Bandung

E-mail : if13095@students.if.itb.ac.id[1], if13108@students.if.itb.ac.id[2],
if13115@students.if.itb.ac.id[3]

Abstrak

Travelling Salesperson Problem (TSP) adalah masalah menemukan perjalanan (*tour*) terpendek. Ada n buah kota dan setiap kota saling berhubungan satu sama lain. Seorang *salesperson* ingin memulai perjalanannya dari satu kota, melalui setiap kota lainnya hanya sekali dan kembali lagi ke kota asal keberangkatan. Jarak antara setiap kota satu sama lain diketahui. *Salesperson* itu ingin meminimalkan ongkos yang harus dikeluarkannya untuk perjalanannya tersebut.

TSP merupakan masalah klasik yang sampai saat ini belum ada algoritma yang cukup mangkus untuk menyelesaikannya. Salah satu algoritma pemecahan masalah TSP adalah algoritma *brute-force* yang mempunyai kompleksitas O(n!). Algoritma *brute-force* ini dapat diperbaiki dengan mengadopsi algoritma DFS (*Depth First Search*). Pengadopsian tersebut diharapkan dapat mengurangi jumlah perjalanan yang harus dihitung untuk mendapatkan solusi terpendek.

***Kata kunci**: travelling salesperson problem, depth first search, brute-force*

1. Pendahuluan

Algoritma *brute-force* menyelesaikan masalah TSP dengan cara:

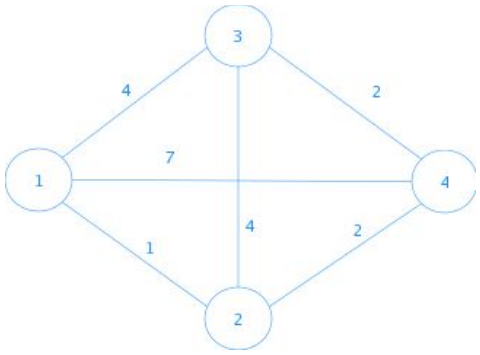
1. Mengenumerasi semua Sirkuit Hamilton dari graf lengkap TSP,
2. Menghitung bobot setiap sirkuit Hamilton yang ditemukan pada langkah 1,
3. Memilih sirkuit Hamilton yang mempunyai bobot terkecil.

Karena algoritma ini menghitung bobot untuk setiap Sirkuit Hamilton yang mungkin terjadi, maka kompleksitasnya sebesar jumlah Sirkuit Hamilton untuk graf lengkap bersimpul n yang dimulai dari sebuah simpul, yakni permutasi dari n buah simpul = n-1 * ... * 1 = (n-1)!. Maka, kompleksitasnya adalah O(n!).

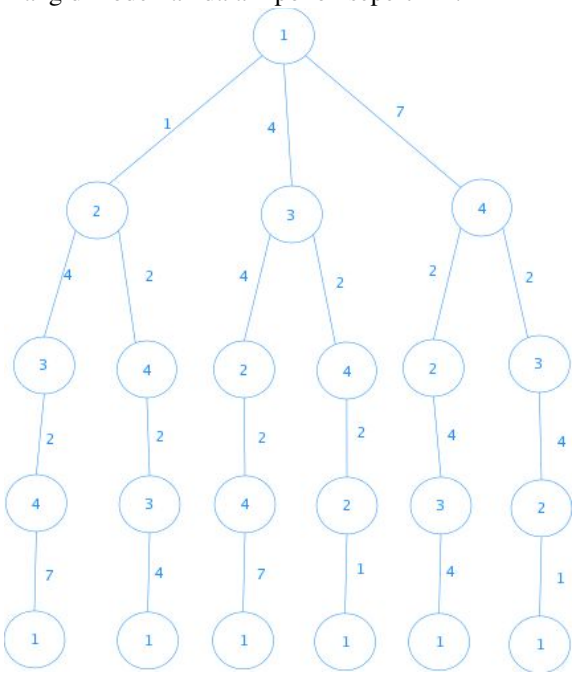
Metoda brute-force di atas dapat diperbaiki dengan cara menghilangkan penghitungan bobot untuk lintasan-lintasan yang jelas tidak mungkin menghasilkan nilai minimum. Algoritma *branch and bound* untuk masalah TSP menghilangkannya secara heuristik dengan memodelkan masalah TSP menjadi pohon, kemudian dari akar, untuk setiap simpul, setiap anaknya diberi 'bobot kira-kira' secara heuristik. Bobot yang diberikan memperkirakan jumlah simpul yang harus dibuat dari simpul itu. Kelemahan algoritma branch and bound untuk TSP, seperti algoritma-algoritma heuristik lainnya, adalah tidak mungkinannya dibuktikan.

2. Konsep Perbaikan Algoritma *Brute-Force*

Algoritma DFS untuk memecahkan permasalahan TSP ini tidak menghitung bobot lintasan tersebut jika sebelumnya sudah ada nilai minimum yang lebih kecil dari nilai lintasan yang akan diekspansi. Misalnya, untuk kasus:



Yang dimodelkan dalam pohon seperti ini:



Jika lintasan 1-2-4-3-1 sudah dipilih sebagai lintasan minimum sementara (bobotnya 9), maka ketika mengekspansi simpul 1-4-2, maka karena bobotnya sudah tidak lebih kecil dari bobot minimum sementara, maka tidak akan diekspansi lagi.

3. Algoritma DFS untuk Permasalahan TSP

Algoritma DFS akan membuat pohon itu secara dinamis, tidak statik, dimulai dari akar. Setiap membuat simpul anak, yaitu simpul pada graf yang belum dikunjungi, maka bobot akan ditambah dengan bobot lintasan ke simpul anak tersebut. Jika mendapatkan Sirkuit Hamilton untuk pertama kalinya, maka sirkuit itu akan disimpan sebagai solusi minimum sementara. Kemudian proses dilanjutkan dengan membuat simpul anak yang lainnya dari simpul bapaknya. Jika tidak ada lagi simpul anak yang belum dibuat, maka kembali ke simpul bapaknya.

Singkatnya, algoritma DFS untuk menyelesaikan TSP adalah seperti ini:

1. Bangun sebuah pohon yang cabangnya berupa simpul pada graf,
2. Lakukan metode DFS pada tiap cabang sampai semua simpul dipilih (tidak ada yang dipilih dua kali),
3. Hitung bobotnya,
4. Lakukan langkah ke-2 dan ke-3 sampai seluruh simpul asal telah dipilih. Apabila pada waktu membangkitkan simpul anak ternyata tidak lebih kecil dari minimum sementara, maka simpul tersebut dimatikan (tidak diekspansi lebih lanjut).

4. Kompleksitas Algoritma

Pada dasarnya, algoritma penyelesaian masalah TSP dengan DFS ini lebih mangkus daripada algoritma *brute-force,* karena tidak semua kemungkinan solusi harus dienumerasikan.

Kasus terbaik untuk algoritma ini adalah ketika solusi pertama yang ditemukan adalah solusi terbaik, dan pembangkitan simpul selanjutnya selalu tidak lebih kecil dari bobot solusi pertama tersebut. Hal ini dimungkinkan jika permasalahan TSP yang diselesaikan memiliki bobot yang tidak seimbang, dimana beberapa lintasan antara dua simpul memiliki bobot yang sangat kecil dibandingkan yang lain.

Tetapi, jika graf TSP yang diselesaikan cukup seimbang, maka hampir semua simpul akan dibangkitkan. Karena itu, kasus terburuk untuk algoritma DFS ini mempunyai kompleksitas yang sama dengan algoritma *brute-force.*

5. Kesimpulan

Penyelesaian masalah *Travelling Salesperson Problem* (TSP) dengan menggunakan DFS dalam kasus rata-rata lebih mangkus dibandingkan dengan algoritma *brute-force* karena hanya sirkuit-sirkuit yang mengarah ke solusi yang dibangkitkan.

Untuk menyelesaikan masalah TSP masih banyak algoritma lain yang bisa dipergunakan. Masalah TSP yang tidak seimbang, mangkus dikerjakan oleh DFS

ini. Sedangkan untuk masalah TSP yang seimbang, bisa menggunakan algoritma lain yang lebih baik.

Daftar Pustaka

[1] Artikel-artikel tentang TSP di Wikipedia
http://en.wikipedia.org