Jurnal
MATEMATICS PAEDAGOGIC

Vol I. No. 2, Maret 2017, hlm. 162 - 168
Available online at www.jurnal.una.ac.id/indeks/jmp

PENYELESAIAN *TRAVELLING SALESMAN PROBLEM* DENGAN *ALGORITMA BRANCH AND BOUND*

Yogo Dwi Prasetyo
Pendidikan Matematika, Universitas Asahan
e-mail: abdullah.prasetyo@gmail.com

Abstract

The shortest route search by a salesman from one city to the n-city exactly once and return to the city early departure is one of combinatorial optimization problems. Travelling Salesman Problem (TSP) can be solved with branch and bound algorithm using a scheme Breadth First Search (BFS) more intelligent use of barrier function is bound to determine the expanded node (branch). The accuracy of the optimal solution depends on the barrier function selected. Barrier function selected by instinc and experience that sometimes do not provide optimal results. This algorithm can be an option in solving combinatorial optimization when viewed from the aspect of their turnaround times.

Keywords: travelling salesman problem, branch and bound algorithm, combinatorial optimization, breadth first search, exhaustive search.

Abstrak

Pencarian rute terpendek oleh seorang *salesman* dari suatu kota ke *n*-kota tepat satu kali dan kembali ke kota awal keberangkatan merupakan salah satu masalah optimisasi kombinatorial. *Travelling Salesman Problem* (*TSP*) dapat diselesaikan dengan algoritma *branch and bound* yang menggunakan skema *Breadth First Search* (BFS) yang lebih pintar yaitu menggunakan fungsi pembatas *bound* untuk menentukan simpul yang diperluas (*branch*). Keakuratan penyelesaian optimal bergantung pada fungsi pembatas yang dipilih. Fungsi pembatas dipilih berdasarkan *instinc* dan pengalaman sehingga terkadang tidak memberikan hasil yang optimal. Algoritma ini bisa menjadi pilihan dalam menyelesaikan optimisasi kombinatorial jika dipandang dari aspek waktu penyelesaiannya.

Kata kunci: travelling salesman problem, algoritma branch and bound, optimisasi kombinatorial, breadth first search, exhaustive search.

Travelling Salesman Problem (*TSP*) adalah pencarian rute terpendek atau jarak minimum oleh seorang *salesman* dari suatu kota ke *n*-kota tepat satu kali dan kembali ke kota awal keberangkatan. *TSP* dapat diterapkan pada graf lengkap berbobot yang memiliki total bobot sisi minimum, dimana bobot pada sisi adalah jarak. Rute TSP ini memuat semua titik pada graf tersebut tepat satu kali. Meskipun persoalan ini bernama persoalan perjalanan pedagang, namun penerapannya tidak hanya pada kasus yang berhubungan dengan pedagang. Banyak penerapan

Jurnal
MATEMATICS PAEDAGOGIC

Vol I. No. 2, Maret 2017, hlm. 162 - 168
Available online at www.jurnal.una.ac.id/indeks/jmp

TSP yang muncul dalam kehidupan sehari-hari misalnya, efisiensi pengiriman surat dan barang, perencanaan pemasangan saluran pipa, masalah transportasi, persoalan *delivery order* (jasa pengantar makanan), analisis rangkaian air atau listrik dan lain sebagainya.

Kebanyakan *TSP* merupakan simetris, artinya untuk dua kota A dan B, jarak kota A ke kota B adalah sama dengan jarak dari kota B ke kota A. Sehingga kita akan mendapatkan jarak perjalanan keliling yang sama persis jika kita membalikkan rute perjalanan tersebut. Banyak algoritma yang dapat digunakan untuk menyelesaikan TSP, diantaranya adalah, *Branch and Bound*, *Simulated Annealing*, *Genetic Algorithm*, *Ant Colony Optimization*, *Simple Hill Climbing*, dan lain-lain. Dalam penelitian ini penulis menggunakan algoritma *branch and bound* untuk menyelesaikan *TSP*.

TSP dapat direpresentasikan ke dalam sebuah graf. Kota-kota yang harus dikunjungi digambarkan dengan simpul atau titik, dan jarak tiap kota digambarkan sebagai garis atau sisi yang menghubungkan titik-titik tersebut. Model graf di lapangan mungkin saja melibatkan titik-titik dan sisi-sisi yang sangat banyak dan membutuhkan waktu yang lama jika hanya berpedoman pada representasi grafis dari graf. Oleh karena itu diperlukan representasi matriks dari sebuah graf sehingga sebuah graf dapat dengan mudah diolah dengan menggunakan komputer.

TSP dapat didefinisikan sebagai berikut. Ada 1 set kota $C_1, C_2, C_3, ..., C_n$ dan $d(C_i, C_j)$ adalah jarak antar kota ke *i* dan kota ke *j*. Tujuannya adalah menentukan urutan π dari rumus berikut untuk mendapatkan nilai yang paling minimal.

$$\sum d(C_{\pi(i)}, C_{\pi(i)}) + d(C_{\pi(N)}, C_{\pi(1)})$$

Hasil dari rumus tersebut disebut sebagai panjang dari perjalanan seorang *salesman* mengunjungi kota-kota sesuai urutan π, dimana setelah mengunjungi semua kota salesman tersebut akan kembali ke kota awal.

Algoritma Branch And Bound

Branch and bound merupakan merupakan metode algoritma yang umum digunakan untuk menentukan penyelesaian optimal dari masalah optimisasi, khususnya pada diskrit dan optimisasi kombinatorial. Pada intinya algoritma ini menggunakan pendekatan enumerasi dengan cara mematikan *search space* yang tidak mengarah pada penyelesaian. Pada tahun 1960, algoritma *branch and bound* diperkenalkan oleh A.H. Land dan A.G. Doig.

Sesuai dengan namanya *branch and bound* memiliki dua alat yaitu *branching* dan *bounding*. *Branching* dilakukan dengan cara meng-*cover* daerah penyelesaian yang layak dengan beberapa sub daerah layak yang lebih kecil. *Bounding* dilakukan dengan cara menentukan nilai batas atas dan batas bawah untuk penyelesaian optimal di dalam sub daerah yang layak.

Proses *branching* menggunakan skema *Breadth First Search* (*BFS*). Simpul atau titik yang dibangkitkan terlebih dahulu merupakan simpul yang bertetangga dengan simpul akar. Namun proses pemilihan simpul yang akan diperluas

Jurnal
MATEMATICS PAEDAGOGIC

Vol I. No. 2, Maret 2017, hlm. 162 - 168
Available online at www.jurnal.una.ac.id/indeks/jmp

(simpul *expand*) tidak seperti pada *BFS* murni. Tidak seperti *BFS* murni yang memilih simpul *expand* berdasarkan urutan pembangkitan, pada algoritma *branch and bound* pemilihan simpul *expand* didasarkan pada nilai fungsi objektif.

Masalah optimisasi biasanya memiliki fungsi untuk menghasilkan nilai batas yang unik. Sedangkan pada algoritma *branch and bound* pemilihan formula fungsi menggunakan metode *heuristic*, berdasarkan pada pengalaman dan *instinc* pembuat program dan tidak ada bukti matematisnya. Hasil optimisasi yang diperoleh melalui algoritma ini bergantung pada keakuratan pemilihan fungsi batas tersebut. Tidak ada cara baku untuk menentukan fungsi batas karena masalah yang sama bisa saja memiliki rumus perhitungan nilai batas yang berbeda.

Langkah-langkah algoritma *branch and bound* untuk menyelesaikan *TSP* adalah sebagai berikut.

Langkah 1:
Tetapkan penyelesaian awal masalah. Penyelesaian yang ditetapkan merupakan rute perjalanan lengkap. Tentukan batas tertinggi pada nilai minimum fungsi objektif dengan mencari berbagai kemungkinan rute perjalanan. Batas atas ini dinotasikan dengan f_U.

Langkah 2:
Buat cabang awal dengan mengatur $x_1 = 1$ untuk masing-masing kota $j = 2,3,...,n$. Untuk $i = j$, nilai $C_{ij} = M$ untuk menyatakan rute yang tidak mungkin. Hitung batas terendah yang dinotasikan dengan f_L pada nilai minimum fungsi objektif di setiap titik. Dari data awal, hapus baris pertama dan kolom ke *j*, serta ganti $C_{ji} = M$. Tentukan penyelesaian masalah dan tambahkan harga f ke C_{ij} untuk memperoleh f_L sehingga $f_L = C_{ij} + f$. Jika $f_L \le f_U$ aktifkan simpul, dan jika sebaliknya hapus simpul.

Langkah 3:
Jika tidak ada simpul aktif pada langkah ini maka penyelesaian terbaik saat ini adalah optimal. Jika tidak, pilih simpul dengan nilai f_L terkecil dan buat cabang baru dengan mengatur $x_{jk} = 1$ untuk setiap kota yang belum dikunjungi sebelumnya.

Langkah 4:
Buat batasan f pada setiap simpul dengan menghapus baris j dan kolom k dari data pada simpul aktif diatasnya. Tambahkan nilai f ke C_{jk} dengan seluruh nilai sebelumnya.

HASIL DAN PEMBAHASAN

Penerapan Algoritma *Branch And Bound* Pada *TSP*

Misalkan terdapat 5 kota yang harus dikunjungi dengan jarak antar kota seperti pada tabel 1.

Tabel 1. Jarak Antar Kota

	1	2	3	4	5
1	M	131	120	144	131
2	131	M	139	153	154
3	120	139	M	127	118
4	144	153	127	M	130
5	131	154	118	130	M

Jurnal
MATEMATICS PAEDAGOGIC

Vol I. No. 2, Maret 2017, hlm. 162 - 168
Available online at www.jurnal.una.ac.id/indeks/jmp

Langkah 1.
Penyelesaian awal untuk masalah *TSP* ditetapkan
$x_{12} = x_{23} = x_{34} = x_{45} = x_{51} = 1$ dengan nilai 131 + 139 + 127 + 130 + 131 = 658, sehingga $f_U^1 = 658$.

Langkah 2.
Buat cabang $x_{12} = 1$ (simpul 2), $x_{13} = 1$ (simpul 3), $x_{14} = 1$ (simpul 4), $x_{15} = 1$ (simpul 5).
Pada simpul 2 hapus baris pertama dan kolom kedua dari data awal serta ganti $C_{21} = M$, sehingga diperoleh :

	1	3	4	5
2	M	139	153	154
3	120	M	127	118
4	144	127	M	130
5	131	118	130	M

Penyelesaiannya adalah
$x_{23} = x_{35} = x_{54} = x_{41} = 1$ dengan nilai 139 + 118 + 130 + 144 = 531, dengan f = 531.
Jadi $f_L = C_{12} + f = 131 + 531 = 662$.
Atur $f_U^2 = 662$, kemudian simpan penyelesaian ini sebagai penyelesaian baru dan simpul 2 sebagai simpul aktif sementara.
Pada simpul 3 hapus baris pertama dan kolom ketiga dari data awal serta ganti $C_{31} = M$, sehingga diperoleh:

	1	2	4	5
2	131	M	153	154
3	M	139	127	118
4	144	153	M	130
5	131	154	130	M

Penyelesaiannya adalah
$x_{35} = x_{54} = x_{42} = x_{21} = 1$ dengan nilai 118 + 130 + 153 + 131 = 532, dengan f = 532.
Jadi $f_L = C_{13} + f = 120 + 532 = 652$.
Karena $f_L < f_U^2$, maka simpan penyelesaian ini sebagai penyelesaian baru. Hapus simpul 2 dan atur simpul 3 sebagai simpul aktif sementara.
Pada simpul 4 hapus baris pertama dan kolom keempat dari data awal serta ganti $C_{41} = M$, sehingga diperoleh:

	1	2	3	5
2	131	M	139	154
3	120	139	M	118
4	M	153	127	130
5	131	154	118	M

Penyelesaiannya adalah
$x_{43} = x_{35} = x_{52} = x_{21} = 1$ dengan nilai 127 + 118 + 154 + 131 = 530, dengan f = 530.
Jadi $f_L = C_{14} + f = 144 + 530 = 674$.
Karena $f_L > f_U^2$, maka hapus simpul 4.

Pada simpul 5 hapus baris pertama dan kolom kelima dari data awal serta ganti $C_{51} = M$, sehingga diperoleh:

	1	2	3	4
2	131	M	139	153
3	120	139	M	127
4	144	153	127	M
5	M	154	118	130

Penyelesaiannya adalah
$x_{53} = x_{34} = x_{42} = x_{21} = 1$ dengan nilai 118 + 127 + 153 + 131 = 529, dengan f = 529.

Jurnal
MATEMATICS PAEDAGOGIC

Vol I. No. 2, Maret 2017, hlm. 162 - 168
Available online at www.jurnal.una.ac.id/indeks/jmp

Jadi $f_L = C_{15} + f = 131 + 529 = 660$. Karena $f_L > f_U^2$, maka hapus simpul 5.

Langkah 3.
Terdapat satu simpul aktif yaitu simpul 3 dengan $f_L = 652$. Buat cabang $x_{32} = 1$ (simpul 6), $x_{34} = 1$ (simpul 7), $x_{35} = 1$ (simpul 8). Buat batasan pada simpul 6, 7, 8, dengan memodifikasi data pada simpul 3.
Pada simpul 6 hapus baris ketiga dan kolom kedua dari data di simpul 3 serta ganti $C_{21} = M$, sehingga diperoleh:

	1	4	5
2	M	153	154
4	144	M	130
5	131	130	M

Penyelesaiannya adalah
$x_{24} = x_{45} = x_{51} = 1$ dengan nilai 153 + 130 + 131 = 414, dengan f = 414. Jadi
$f_L = C_{13} + C_{32} + f = 120 + 139 + 414 = 673$
Atur $f_U^3 = 673$, kemudian simpan penyelesaian ini sebagai penyelesaian baru dan simpul 6 sebagai simpul aktif sementara.
Pada simpul 7 hapus baris ketiga dan kolom keempat dari data di simpul 3 serta ganti $C_{41} = M$, sehingga diperoleh:

	1	4	5
2	131	M	154
4	M	153	130
5	131	154	M

Penyelesaiannya adalah
$x_{45} = x_{52} = x_{21} = 1$ dengan nilai 130 + 154 + 131 = 415, dengan f = 415. Jadi
$f_L = C_{13} + C_{34} + f = 120 + 127 + 415 = 662$
Karena $f_L < f_U^3$, maka simpan penyelesaian ini sebagai penyelesaian baru. Hapus simpul 6 dan atur simpul 7 sebagai simpul aktif sementara.
Pada simpul 8 hapus baris ketiga dan kolom kelima dari data di simpul 3 serta ganti $C_{51} = M$, sehingga diperoleh:

	1	2	4
2	131	M	153
4	144	153	M
5	M	154	130

Penyelesaiannya adalah
$x_{54} = x_{42} = x_{21} = 1$ dengan nilai 130 + 153 + 131 = 414, dengan f = 414. Jadi
$f_L = C_{13} + C_{35} + f = 120 + 118 + 414 = 652$
Karena $f_L < f_U^3$, maka simpan penyelesaian ini sebagai penyelesaian baru. Hapus simpul 7 dan atur simpul 8 sebagai simpul aktif.

Langkah 4.
Terdapat satu simpul aktif yaitu simpul 8 dengan $f_L = 652$. Buat cabang $x_{52} = 1$ (simpul 9), $x_{54} = 1$ (simpul 10).
Pada simpul 9 hapus baris kelima dan kolom kedua dari data di simpul 8 serta ganti $C_{21} = M$, sehingga diperoleh :

	1	4
2	M	153
4	144	M

Jurnal
MATEMATICS PAEDAGOGIC

Vol I. No. 2, Maret 2017, hlm. 162 - 168
Available online at www.jurnal.una.ac.id/indeks/jmp

Penyelesaiannya adalah $x_{24} = x_{41} = 1$ dengan nilai 153 + 144 = 297, dengan f = 297.

Jadi

$f_1 = C_{13} + C_{35} + C_{52} + f$
= 120 + 118 + 154 + 297
= 689

Karena $f_L > f_U^3$, maka hapus simpul 9.

Pada simpul 10 hapus baris kelima dan kolom keempat dari data di simpul 8 serta ganti $C_{41} = M$, sehingga diperoleh :

	1	2
2	131	M
4	M	153

Penyelesaiannya adalah $x_{42} = x_{21} = 1$ dengan nilai 153 + 131 = 284, dengan f = 284.

Jadi

$f_L = C_{13} + C_{35} + C_{54} + f = 120 + 118 + 130 + 284 = 652$

Karena $f_L > f_U^3$, maka hapus simpul 10.

Tidak ada simpul yang aktif. Terdapat alternatif perjalanan yang optimal yaitu 1 - 3 - 5 - 4 - 2 - 1 dengan nilai 652.

Pohon algoritma *branch and bound* diperlihatkan pada gambar 1.

Perbandingan Hasil Algoritma *Branch And Bound* Dengan *Exhaustive Search*

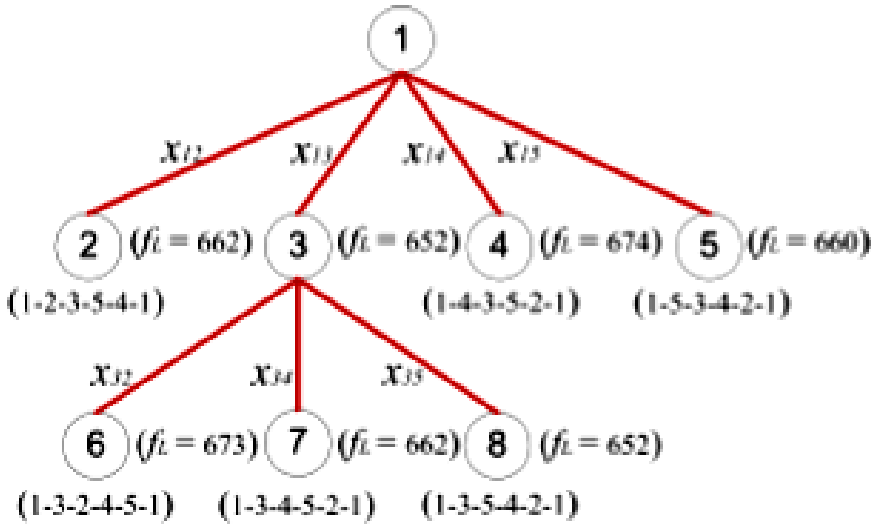
Hasil iterasi pada *exhaustive search* diperlihatkan pada tabel 2.

Alternatif Solusi	Nilai	Alternatif Solusi	Nilai
1-2-3-4-5-1	658	1-4-3-5-2-1	674
1-2-3-5-4-1	662	1-4-3-2-5-1	695
1-2-4-5-3-1	652	1-4-5-2-3-1	687
1-2-4-3-5-1	660	1-4-5-3-2-1	662
1-2-5-3-4-1	674	1-4-2-3-5-1	685
1-2-5-4-3-1	662	1-4-2-5-3-1	689
1-3-4-5-2-1	662	1-5-4-3-2-1	658
1-3-4-2-5-1	685	1-5-4-2-3-1	673
1-3-5-2-4-1	689	1-5-3-2-4-1	685
1-3-5-4-2-1	652	1-5-3-4-2-1	660
1-3-2-4-5-1	673	1-5-2-4-3-1	695
1-3-2-5-4-1	687	1-5-2-3-4-1	695

Dari tabel di atas terlihat bahwa penyelesaian optimal yang diperoleh dengan *exhaustive search* sama dengan algoritma *branch and bound* yaitu 1-3-5-4-2-1 atau dengan rute terbalik 1-2-4-5-3-1 memiliki nilai optimal 652. Karena *exhaustive search* akan memberikan hasil 100% benar, maka untuk contoh kasus ini hasil yang diperoleh melalui algoritma *branch and bound* juga pasti benar.

Jurnal
MATEMATICS PAEDAGOGIC

Vol I. No. 2, Maret 2017, hlm. 162 - 168
Available online at www.jurnal.una.ac.id/indeks/jmp



Gambar 1. Pohon algoritma *branch and bound*

SIMPULAN

Banyak metode untuk menyelesaikan permasalahan optimisasi kombinatorial, salah satu diantaranya adalah algoritma *branch and bound*. Algoritma ini menggunakan skema *Breadth First Search* (*BFS*) yang lebih pintar, yaitu menggunakan fungsi pembatas *bound* untuk menentukan simpul *expand* yaitu simpul yang akan diperluas berikutnya. Kelemahan dari algoritma ini adalah sama seperti *exhaustive search* yaitu n!, namun hal ini sangat jarang terjadi karena algoritma ini menggunakan fungsi pembatas dalam pencarian solusi.

Keakuratan penyelesaian optimal dari algoritma *branch and bound*, sangat tergantung pada fungsi pembatas yang dipilih. Formula dipilih berdasarkan insting dan pengalaman sehingga terkadang tidak memberikan hasil yang optimal. Namun dari aspek waktu, algoritma ini bisa menjadi pilihan dalam menyelesaikan masalah optimisasi kombinatorial.

DAFTAR RUJUKAN

Bayram, H. dan Sahin, R. 2013. ”A New Simulated Annealing Approach for Travelling Salesman Problem”, *Mathematical and computational Applications*, 18 (3): 313 – 322

Brualdi, R.A. 2010. *Introductory Combinatorics, Fifth Edition. Pearson Education*, Inc., Prentice Hall.

Munawar, H. 2007. *Penerapan Algoritma Branch and Bound dalam Optimasi Assignment Problem*. Bandung: STEI-ITB.

Riyanti, E. 2004. *Penerapan Algoritma Branch and Bound untuk Penentuan Rute Objek Wisata*. Jakarta: UNIKOM.

Salaki, D.T. dan Rindengan, A. J. 2010. Optimasi Rute Distribusi Barang dengan Menggunakan Metode Heuristik. *Jurnal Ilmiah Sains*. 10 (1); 75 – 80

Jurnal
MATEMATICS PAEDAGOGIC

Vol I. No. 2, Maret 2017, hlm. 162 - 168
Available online at www.jurnal.una.ac.id/indeks/jmp