Penyelesaian *Asymmetric Travelling Salesman Problem* dengan Algoritma Hungarian dan Algoritma *Cheapest Insertion Heuristic*

Caturiyati[1]
[1]Staf Pengajar Jurusan Pendidikan Matematika FMIPA UNY
E-mail: wcaturiyati@yahoo.com

ABSTRAK: Masalah *Asymmetric Travelling Salesman Problem* (ATSP) merupakan masalah mengoptimumkan rute perjalanan seorang pedagang yang membentuk sebuah sirkuit, dimana semua kota hanya disinggahi sekali saja, dan jalur pulang dan jalur pergi diantara dua kota belum tentu sama. Masalah ATSP dapat diselesaikan dengan menggunakan algoritma Hungarian, dimana masalah ATSP dipandang sebagai masalah Penugasan (*Assignment Problem*) dengan *n* pekerja diisi oleh satu orang pekerja saja, dan *n* pekerjaan diasumsikan sebagai *n* kota tujuan. ATSP juga dapat diselesaikan dengan Algoritma *Cheapest Insertion Heuristic* (CIH) dengan penelusuran siklus perjalanan dimulai dengan menghubungkan kota pertama dan kota terakhir, yang selanjutnya kota-kota persinggahan di insersi (disisipkan) dengan mencari rute terpendeknya.
Kata kunci: ATSP, masalah penugasan, Hungarian, CIH.

1. Pendahuluan

Travelling Salesman Problem (TSP) merupakan aplikasi Teori Graf dan menjadi bagian dari Riset Operasi. TSP dianggap sebagai kasus khusus dari masalah transportasi, dengan *supply* (persediaan), b_i, dan *demand* (permintaan), a_j, nya adalah satu untuk setiap *i* dan setiap *j*. TSP juga dapat dianggap sebagai kasus khusus dari masalah penugasan dengan *n* pekerja hanya akan diisi oleh satu pekerja saja dan *n* pekerjaan diasumsikan sebagai *n* kota tujuan, dan hasil optimum masalah penugasannya harus membentuk sirkuit, dalam arti pekerja ini harus mengunjungi *n*-1 kota dan kembali lagi ke kota asal.

TSP terdiri dari *Symmetric* TSP (STSP) dan *Asymmetric* TSP (ATSP). STSP adalah TSP dimana jalur pergi dan jalur pulang antara dua kota selalu sama. Sedangkan ATSP adalah TSP dimana jalur pergi dan jalur pulang tidak selalu sama. Yang selalu menjadi kendala pada STSP maupun ATSP adalah bagaimana mencapai solusi optimum yang langsung menghasilkan sirkuit?

Di dalam perkembangannya telah banyak algoritma dikembangkan untuk membantu menyelesaikan STSP maupun ATSP, terutama algoritma yang dikembangkan menjadi software komputer. Dorigo and Gambardella (1997) membahas *Ant Colony System* sebagai salah satu algoritma yang dapat digunakan untuk menyelesaikan TSP. Freisleben and Merz (1996), membahas TSP dengan Algoritma Genetik. Sierksma

(1994) membahas TSP dengan sirkuit Hamilton. Kusrini dan Istiyanto (2007) membahas algoritma *Cheapest Insertion Heuristic* untuk STSP. Dalam makalah ini akan dibahas penyelesaian ATSP dengan Algoritma Hungarian dan Algoritma *Cheapest Isertion*, yang dapat membantu penyampaian materi perkuliahan riset operasi, namun juga dapat dikembangkan menjadi software komputer.

2. ATSP Sebagai Masalah Penugasan

Masalah penugasan merupakan bentuk khusus masalah transportasi. Seperti masalah transportasi, masalah penugasan adalah suatu masalah optimasi meminimumkan, walaupun tidak menutup kemungkinan adanya masalah optimasi memaksimumkan. Kekhususan masalah penugasan adalah antara *origin* (sumber) dengan *destinasi* (tujuan) hanya akan dipenuhi oleh satu variabel basis saja yang nilainya satu, karena *supply* dan *demand* pada masalah penugasan bernilai 1 untuk setiap *origin* dan untuk setiap *destinasi*. Tabel berikut merupakan tabel transportasi untuk masalah penugasan:

	D_1	D_2	…	D_n	b_i
O_1	c_{11}	c_{12}	…	c_{1n}	1
O_2	c_{21}	c_{22}	…	c_{2n}	1
⋮	⋮	⋮	⋱	⋮	⋮
O_n	c_{n1}	c_{n2}	…	c_{nn}	1
a_j	1	1	…	1	

Keterangan tabel:

O_i : pelamar kerja ke-i, D_j : lowongan kerja ke-j, c_{ij} : gaji O_i bila diterima di D_j.

Model masalah penugasan pola minimum setimbang:

Mencari $x_{ij} \geq 0$ yang meminimalkan $f = \sum_{j=1}^{n}\sum_{i=1}^{n} c_{ij}x_{ij}$, dengan kendala: $\sum_{j=1}^{n} x_{ij} = b_i = 1, \forall i$

dan $\sum_{i=1}^{n} x_{ij} = a_j = 1, \forall j$, dan x_{ij} bernilai 0 atau 1.

Sedangkan pada ATSP diasumsikan: 1) terdapat sejumlah n lokasi, 2) tersedia jalur dari satu lokasi ke $n - 1$ lokasi lainnya, 3) tersedia ongkos c_{ij} dari lokasi ke-i ke lokasi ke-j pada jalur $i \rightarrow j$, 4) c_{ij} tidak selalu sama dengan c_{ji}, 5) seseorang harus berangkat dari suatu lokasi dan mengunjungi $n - 1$ lokasi lainya (masing-masing sekali) dan akhirnya kembali ke lokasi semula (sirkuit), 6) tujuan ATSP adalah menentukan sirkuit perjalanan yang meminimalkan ongkos total.

ATSP dapat dipandang sebagai masalah penugasan sebagai berikut: 1) Lokasi yang dikunjungi diberi label 1, 2, 3, …, *n*, 2) Pada ATSP *n* pekerja diisi 1 orang, sedangkan *n* pekerjaan merupakan *n* lokasi tujuan.

Sehingga tabel ATSP nya

	1	2	…	*n*	b_i
1	c_{11}	c_{12}	…	c_{1n}	1
2	c_{21}	c_{22}	…	c_{2n}	1
⋮	⋮	⋮	⋱	⋮	⋮
n	c_{n1}	c_{n2}	…	c_{nn}	1
a_j	1	1	…	1	

3. Penyelesaian ATSP dengan Algoritma Hungarian

1. Diusahakan supaya dalam setiap baris dan kolom ada *cost* nolnya; dengan cara mengurangi setiap baris (kolom) dengan $\min\{c_{ij}\}$ dalam setiap baris (kolom).
2. Diuji apakah ada *n* kotak dengan *cost* nol yang mewakili tiap-tiap baris dan kolom? Jika ada, berarti tabel sudah optimal (karena merupakan kombinasi nol yang paling murah) dan itulah penyelesaian optimal soal asli. Bila belum ada, diadakan lagi pengurangan *cost* (langkah 1).
   a. Pengujian dapat dilakukan dengan penutupan baris dan kolom yang memuat *cost* nol, sehingga banyaknya baris penutup sama dengan banyaknya baris atau kolom, jika demikian maka masalah sudah optimal.
   b. Jika belum, maka pada *cost* yang tidak tertutup garis, lakukan pengurangan dengan *cost* termurah yang tidak tertutup tadi (*r*), pada *cost* yang merupakan titik potong garis penutup baris dan kolom ditambahkan dengan *cost* termurah tadi. Lakukan uji optimum lagi.

Pengurangan baris dengan suatu cost, tidak mengubah solusi optimal sebab untuk p_1 konstan:

(i) $$f = \sum_{j=1}^{n}\sum_{i=1}^{n} c_{ij}x_{ij} = \sum_j c_{1j}x_{1j} + \sum_j c_{2j}x_{2j} + \ldots + \sum_j c_{nj}x_{nj}\,.$$

(ii) $$\bar{f} = \sum_j (c_{1j} - p_1)x_{1j} + \sum_j c_{2j}x_{2j} + \ldots + \sum_j c_{nj}x_{nj}$$

$$= \sum_j c_{1j}x_{1j} + \sum_j c_{2j}x_{2j} + \ldots + \sum_j c_{nj}x_{nj} - p_1 \underbrace{\sum_j x_{1j}}_{1} = f - p_1.$$

Dengan kata lain, meminimalkan $f \Leftrightarrow$ meminimalkan $\bar{f}$.

Secara umum, jika baris ke i_0 dikurangi p_i, maka:

$$\bar{f} = \sum_j \sum_i (c_{ij} - p_{i_0})x_{ij} = \sum_j c_{1j}x_{1j} + \ldots + \sum_j (c_{i_0 j} - p_{i_0})x_{i_0 j} + \ldots + \sum_j c_{nj}x_{nj}$$

$$= \sum_j \sum_i c_{ij}x_{ij} - p_{i_0} \underbrace{\sum_j x_{i_0 j}}_{1} = f - p_{i_0}.$$

Sehingga setiap solusi yang mungkin untuk ATSP juga merupakan solusi masalah penugasan yang bersangkutan. Namun demikian masalah penugasan yang bersangkutan mungkin mempunyai solusi yang tidak membentuk sirkuit, yaitu yang tidak membentuk *fisibel* untuk solusi ATSP. Oleh karenanya ATSP diselesaikan dengan cara sebagai berikut:

1. Ubahlah ATSP menjadi masalah penugasan seperti pada tabel di atas, dengan c_{ii} = bilangan bulat positif yang nilainya besar, sebut M.

2. Selesaikan masalah penugasan tersebut dengan algoritma Hungarian.

3. Jika solusi masalah penugasan tersebut membentuk sirkuit, maka solusi ini merupakan solusi optimal ATSP.

4. Jika solusi masalah penugasan tersebut belum membentuk sirkuit, misalnya diperoleh hasil optimal

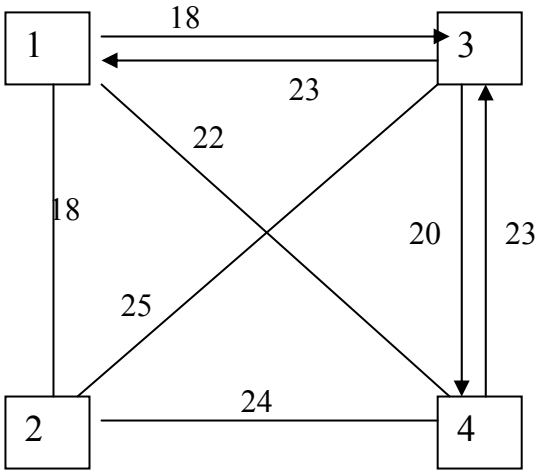
$1 \rightarrow 3$ $\quad$ $1 \leftrightarrow 3$

$3 \rightarrow 1$ $\quad$ $2 \leftrightarrow 4$

$2 \rightarrow 4$ $\quad$ $4 \rightarrow 2$

maka dilakukan sebagai berikut: pada tabel awal c_{13} atau c_{31} atau c_{24} atau c_{42} diganti dengan M, kemudian diselesaikan lagi dengan algoritma *Hungarian*, cara ini disebut "*Branch and Bound method*" atau B n B.

4. Contoh 1 : Diberikan diagram berikut ini :



Masalah pada diagram adalah masalah ATSP, dengan 4 kota tujuan. Pemodelan masalah ATSP tersebut adalah meminimalkan

$$f(x_{ij}) = 18x_{12} + 18x_{21} + 18x_{13} + 23x_{31} + 22x_{14} + 22x_{41} + 25x_{23} + 25x_{32} +$$
$$24x_{24} + 24x_{42} + 20x_{34} + 23x_{43}$$

Terhadap kendala

$$x_{11} + x_{12} + x_{13} + x_{14} = 1 \qquad x_{11} + x_{21} + x_{31} + x_{41} = 1$$
$$x_{21} + x_{22} + x_{23} + x_{24} = 1 \qquad x_{12} + x_{22} + x_{32} + x_{42} = 1$$
$$x_{31} + x_{32} + x_{33} + x_{34} = 1 \qquad x_{13} + x_{23} + x_{33} + x_{43} = 1$$
$$x_{41} + x_{42} + x_{43} + x_{44} = 1 \qquad x_{14} + x_{24} + x_{34} + x_{44} = 1$$
$$x_{ij} \geq 0, \forall i = 1,2,3,4, \forall j = 1,2,3,4 .$$

Tabel ATSP tersebut adalah

	1	2	3	4	b_i
1	…	18	18	22	1
2	18	…	25	24	1
3	23	25	…	20	1
4	22	24	23	…	1
a_j	1	1	1	1	4

$c_{11} = c_{22} = c_{33} = c_{44}$ diisi dengan M. Sehingga tabel *cost* ATSP nya adalah

M	18	18	22	-18
18	M	25	24	-18
23	25	M	20	-20
22	24	23	M	-22

Pada iterasi pertama ini, baris I, II, III, IV berturut-turut dikurangi dengan 18, 18, 20, 22. Diperoleh tabel berikut

M-18	0	0	4
0	M-18	7	6
3	5	M-20	0
0	2	1	M-22

Pada tabel cost, terlihat untuk setiap baris dan setiap kolomnya, sudah terdapat nol yang mewakili, sehingga dilakukan uji optimum dengan penutupan garis. Diperoleh banyaknya garis penutup ada $3 < n$ (banyaknya baris), tabel belum optimum. Pada *cost* tak tertutup garis, $r = 1$. Sehingga diperoleh tabel berikut:

M-17	0	0	5
0	M-19	6	6
3	4	M-21	0
0	1	0	M-22

Pada tabel terlihat uji optimum dengan penutupan garis, menghasilkan banyaknya garis penutup sebanyak $4 = n$, sehingga tabel optimum. Dengan pemilihan *cost* nol pada tabel berikut:

M-17	0√	0	5
0√	M-19	6	6
3	4	M-21	0√
0	1	0√	M-22

Pemilihan cost nol pada tabel optimum, sama maknanya dengan memilih variabel basis yang akan mewakili masalah ATSP nya, dengan urutan pemilihan $x_{34} - x_{21} - x_{43} - x_{12}$, seperti terlihat pada tabel ATSP berikut

	1	2	3	4	b_i
1	M	18 1	18	22	1
2	18 1	M	25	24	1
3	23	25	M	20 1	1
4	22	24	23 1	M	1
a_j	1	1	1	1	4

Namun hasil optimum tersebut belum membentuk sirkuit. Yaitu 1 – 2 – 1 dan 3 – 4 – 3. Sehingga perlu dilakukan B n B pada c_{12} atau c_{21} atau c_{34} atau c_{43}.

Pada makalah ini akan dilakukan B n B pada c_{12}. Sehingga tabel *cost* awal ATSP nya menjadi

M	M	18	22	-18
18	M	25	24	-18
23	25	M	20	-20
22	24	23	M	-22

Karena kolom II belum memuat *cost* nol, maka kolom II dikurangi dengan $\min\{c_{i2}\}$ $\forall i = 1,2,3,4$.

M-18	M-18	0	4
0	M-18	7	6
3	5	M-20	0
0	2	1	M-22

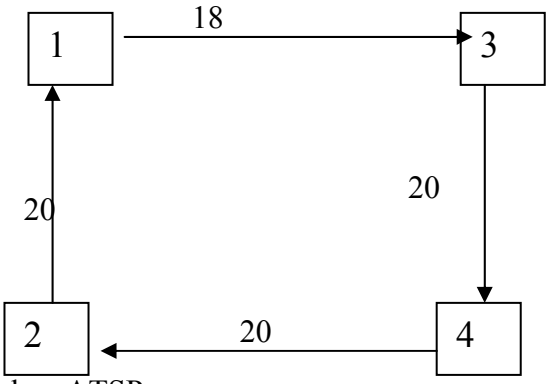
-2

M-18	M-20	0	4
0	M-20	7	6
3	3	M-20	0
0	0	1	M-22

Dengan uji optimum penutupan garis, diperoleh tabel *cost* sudah optimum. Pemilihan *cost* nol nya adalah

M-18	M-20	0√	4
0√	M-20	7	6
3	3	M-20	0√
0	0√	1	M-22

Menghasilkan variabel basis terpilih $x_{34} - x_{13} - x_{21} - x_{42}$. Sirkuit yang terbentuk adalah



Tabel lengkap ATSP nya

	1	2	3	4	b_i
1	M	18	18 1	22	1
2	18 1	M	25	24	1
3	23	25	M	20 1	1
4	22	24 1	23	M	1
a_j	1	1	1	1	4

dengan $f_{\min} = 18 + 20 + 24 + 18 = 80$.

5. Penyelesaian ATSP dengan Algoritma Cheapest Insertion Heuristic (CIH)

1. Penelusuran dimulai dari sebuah kota pertama yang dihubungkan dengan sebuah kota terakhir.
2. Dibuat sebuah hubungan *subtour* antara 2 kota tersebut. Yaitu suatu perjalanan dari kota pertama dan berakhir di kota pertama, misal (1,3) → (3,2) → (2,1).
3. Ganti salah satu arah hubungan (*arc*) antara dua kota 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 diperoleh dari perhitungan *cost* dengan rumus $d_k = c_{ik} + c_{kj} - c_{ij}$.
4. Ulangi langkah 3 sampai seluruh kota masuk dalam *subtour*.

6. Contoh 2: Diberikan diagram ATSP pada Contoh 1. ATSP tersebut akan diselesaikan dengan Algoritma CIH sebagai berikut:

Tabel jarak antar kota pada Contoh 1 adalah

Kota Asal	Kota Tujuan	Jarak
1	2	18
1	3	18
1	4	22
2	3	25
2	4	24
3	1	23
3	4	20
4	3	23

Untuk mencari jarak terpendek yang melalui 4 kota tersebut diambil langkah-langkah sebagai berikut:

1. Ambil perjalanan dari kota 1 ke kota 4.
2. Buat *subtour* (1,4) → (4,1).
3. Buat tabel yang menyimpan kota yang bisa di insersi (disisipkan) dalam *subtour* beserta tambahan jaraknya, seperti pada tabel berikut

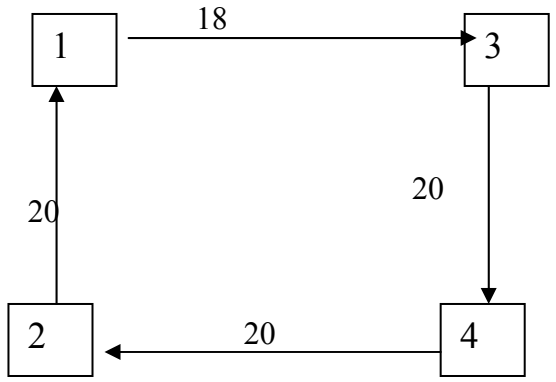
***Arc* yang akan diganti**	***Arc* yang akan ditambahkan**	**Tambahan jarak**
(1,4)	(1,2) – (2,4)	$c_{12} + c_{24} - c_{14} = 20$
(1,4)	(1,3) – (3,4)	$c_{13} + c_{34} - c_{14} = 16$
(4,1)	(4,2) – (2,1)	$c_{42} + c_{21} - c_{41} = 20$
(4,1)	(4,3) – (3,1)	$c_{43} + c_{31} - c_{41} = 24$

   Pada tabel terlihat, tambahan jarak terkecil diperoleh apabila: *Arc*(1,4) diganti dengan *arc* (1,3) dan *arc*(3,4). Maka diperoleh *subtour* baru yaitu (1,3) → (3,4) → (4,1).
4. Buat tabel yang menyimpan kota yang bisa di insersi (disisipkan) dalam *subtour* beserta tambahan jaraknya, seperti pada tabel berikut

Arc yang akan diganti	*Arc* yang akan ditambahkan	Tambahan jarak
(1,3)	(1,2) – (2,3)	$c_{12} + c_{23} - c_{13} = 25$
(3,4)	(3,2) – (2,4)	$c_{32} + c_{24} - c_{34} = 29$
(4,1)	(4,2) – (2,1)	$c_{42} + c_{21} - c_{41} = 20$

Pada tabel terlihat, tambahan jarak terkecil diperoleh apabila: Arc(4,1) diganti dengan arc (4,2) dan arc(2,1). Maka diperoleh *subtour* baru yaitu (1,3) → (3,4) → (4,2) → (2,1).

Karena semua kota sudah termasuk dalam *subtour* terakhir maka diperoleh sirkuit ATSP dengan jarak tempuhnya $c_{13} + c_{34} + c_{42} + c_{21} = 80$. Dengan sirkuitnya adalah



7. Kesimpulan

Algoritma Hungarian dan Algoritma CIH dapat digunakan untuk menyelesaikan ATSP. Penyelesaian ATSP dengan kedua algoritma apabila dikerjakan secara manual dengan jumlah kota kurang dari 10 masih mungkin untuk dilakukan, dan masing-masing algoritma mempunyai kelebihan serta kekurangan. Algoritma Hungarian memperlihatkan perhitungan yang lebih sederhana, karena menggunakan matriks *cost*, dan kesalahan perhitungan dapat diminimalkan.

Daftar Pustaka

Dorigo, M and Gambardella, L.M., 1997, *Ant Colony System: A Cooperative Learning Approach to the Travelling Salesman Problem*, Accepted for publication in the IEEE Transactions on Evolutionary Computation, Vol. 1, No.1, 1997. In press. ftp://iridia.ulb.ac.be/pub/mdorigo/journals/IJ.16-TEC97.A4.pdf diakses pada tanggal 10 Nopember 2008

Freisleben, B and Merz, P., 1996, *New Genetic Local Search Operators for the Travelling Salesman Problem*. http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.113.9500 diakses pada tanggal 10 Nopember 2008

Kusrini dan Istiyanto, J.E., 2007, *Penyelesaian Travelling Salesman Problem dengan Algoritma Cheapest Insertion Heuristic dan Basis Data*, JURNAL INFORMATIKA, Jurusan Teknik Informatika, Fakultas Teknologi Industri, Universitas Kristen Petra, Vol. 8, No. 2, Nopember 2007, 109-114. http://puslit2.petra.ac.id/ejournal/index.php/inf/article/shop/16775/16755 diakses pada tanggal 10 Nopember 2008

Sierksma, G., 1994, *Hamiltonicity and the 3-Opt Procedure for the Travelling Salesman Problem*, Jurnal Applicationes Mathematicae, 22.3 (1994), pp. 351 – 358. http://matwbn.icm.edu.pl/ksiazki/zm/zm22/zm2235.pdf diakses pada tanggal 10 Nopember 2008