Algoritma *Branch and Bound* dalam Kegunaannya Memecahkan *Assignment Problem*

Made Mahendra Adyatman – 13505015

Program Studi Teknik Informatika, Sekolah Teknik Elektro dan Informatika
Insitut Teknologi Bandung
Jl. Ganesha 10, Bandung
Email : if15015@students.if.itb.ac.id

ABSTRAK

Masalah penugasan (*assignment problem*) adalah merupakan suatu masalah yang sangat nyata wujudnya dalam kehidupan keprofesian. Secara umum masalah ini berkisar tentang bagaimana memasangkan 1 orang yang ada dengan 1 pekerjaan (*job*) yang ada dalam suatu lingkungan kerja sehingga tepat satu orang bersesuaian dengan pekerjaan yang ada. Karena dalam kehidupan nyata setiap pekerjaan itu berbeda-beda dan setiap orang itu memiliki keahlian yang berbeda-beda pula, maka biaya untuk pemilihan pekerjaan untuk setiap orang berbeda pula. Sebagai contoh, misalkan si A adalah seorang *desainer* sedangkan B adalah seorang *programmer*, X adalah pekerjaan yang melibatkan interface sedangkan Y adalah pekerjaan memrogram suatu aplikasi, maka apabila A dipasangkan dengan Y dan B dipasangkan dengan X tentunya akan terjadi ketidakcocokan dan apabila dipaksakan maka akan menimbulkan biaya tambahan yang cukup besar, misalkan untuk biaya *privat* untuk mempelajari job tersebut. Fungsi objektif dari masalah ini adalah bagaimana meminimumkan biaya yang terjadi selama penugasan sehingga kita sebagai pihak yang melakukan penugasan mengeluarkan biaya seminimum mungkin. Dalam pencarian solusi assignment yang ada, banyak sekali algoritma yang sudah mencoba untuk memecahkannya, berhasil tetapi dengan kompleksitas waktu yang kurang baik. Terkadang ada yang kompleksitas algoritmanya baik, tetapi ketika worst-case maka kompleksitas waktunya sangatlah teramat buruk. Algoritma *branch and bound* adalah contoh algoritma yang merupakan pendekatan paling baik dalam pencarian solusi ini. Dengan menggunakan metode *Breadth First Search* dengan lebih diperbaiki dan menggunakan metode pencarian solusi dalam ruang solusi yang sistematis atau disini ruang solusinya adalah pohon, algoritma ini sangatlah tepat untuk masalah ini dengan segala kelebihannya itu.

Kata kunci: *assigment problem*, orang, *job*, biaya minimum, fungsi objektif, kompleksitas waktu, *branch and bound*,*Breadth First Search*, pohon

1. PENDAHULUAN

1.1 *Assignment problem*

Assigment problem adalah suatu persoalan dimana kita harus melakukan suatu penugasan terhadap sekumpulan orang yang kepada sekumpulan job yang ada, sehingga tepat 1 orang yang bersesuaian dengan tepat 1 job yang ada. Penugasan orang 1 dengan job yang ada (1,2,…,*n*) menghasilkan biaya yang berbeda untuk setiap job yang ada sesuai dengan tingkat keahlian orang itu. Misalkan orang 1 dengan job 1 menghasilkan biaya (*cost*) sebesar c(1,1) dan orang 1 dengan job 2 menghasilkan biaya c(1,2) dan seterusnya untuk setiap orang dan job yang ada. Sehingga ada c(1,1), c(1,2), …, c(1,*n*), c(2,1), c(2,2), …, c(2,*n*), …,c(*n*,*n*). Maka untuk misalkan setiap 4 orang dengan 4 job yang ada menghasilkan 4! kemungkinan yaitu 24 kemungkinan yang ada. Namun yang dicari disini atau fungsi objektif yang ada yaitu mencari biaya seminimum mungkin dalam penugasan ini sehingga kita sebagai orang yang melakukan penugasan mengeluarkan biaya seminimum mungkin. Diperlukan suatu algoritam yang efektif untuk memecahkan persoalan ini, sehingga menggunakan algoritma branch and bound yang paling mendekati dalam pencarian solusi ini.

1.2 Algoritma Breadth First Search

Algoritma ini adalah dasar dari algoritma branch and bound yang akan kita bicarakan nantinya. Dengan menggunakan sistem pencarian solusi dalam ruang solusi yang sistematis atau dalam BFS menggunakan pohon, algoritma BFS sudah dapat dipastikan dapat mencari simpul solusi yang ada tapi dengan kompleksitas waktu yang agak lama karena dalam pencarian solusinya, BFS akan membangkitkan semua simpul-simpul anak yang

ada dan berhenti apabila sudah mendapat simpul solusi pertama. Prinsip BFS adalah :

1. Memasukkan simpul akar ke dalam antrian Q. Jika simpul akar adalah simpul solusi (goal node), maka soulsi sudah ditemukan. Stop.
2. Jika antrian Q kosong, maka tidak ada solusi. Stop.
3. Ambil simpul v dari kepala (head) antrian, bangkitkan semua simpul anaknya, jika sudah selesai, kembali ke langkah 2. Tempatkan semua anak dari v di belakang antrian sesuai dengan urutan kelahirannya.
4. Jika suatu simpul anak dari v adalah simpul solusi, maka solusi terlah ditemukan, kalau tidak maka kembali ke langkah 2.

1.3 Algoritma Branch and Bound

Sebagaimana pada algortima runut-balik, algoritma *Branch & Bound* juga merupakan metode pencarian di dalam ruang solusi secara sistematis. Ruang Solusi diorganisasikan ke dalam pohon ruang status. Pembentukan pohon ruang status. Pembentukan pohon ruang status pada algoritma B&B berbeda dengan pembentukan pohon pada algoritma runutbalik. Bila pada algoritma runut-balik ruang solusi dibangun secara *Depth-First Search*(DFS), maka pada algoritma B&B ruang solusi dibangun dengan skema *Breadth-First Search* (BFS).

Pada algoritma B&B, pencarian ke simpul solusi dapat dipercepat dengan memilih simpul hidup berdasarkan nilai ongkos (*cost*). Setiap simpul hidup diasosiasikan dengan sebuah ongkos yang menyatakan nilai batas (*bound*). Pada prakteknya, nilai batas untuk setiap simpul umumnya berupa taksiran atau perkiraan. Fungsi heuristik untuk menghitung taksiran nilai tersebut dinyatakan secara umum sebagai :

$$\hat{c}(i) = f(i) + \hat{g}(i)$$

yang dalam hal ini,

$\hat{c}(i)$ = ongkos untuk simpul i

$f(i)$ = ongkos mencapai simpul i dari akar

$\hat{g}(i)$ = ongkos mencapai simpul tujuan dari simpul akar i (perkiraan)

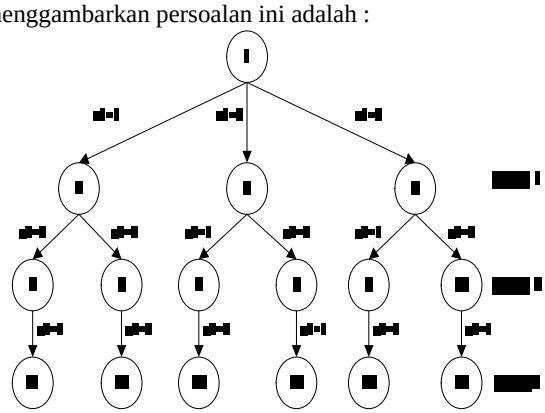
Nilai $\hat{c}$ digunakan untuk mengurutkan pencarian. Simpul berikutnya yang dipilih untuk diekspansi adalah simpul yang memiliki $\hat{c}$ minimum (Simpul-E). Strategi memilih simpul-E seperti ini dinamakan strategi pencarian berdasarkan biaya terkecil (least *cost* search).

Prinsip dari algoritma branch and bound ini adalah :

1. Masukkan simpul akar ke dalam antrian *Q*. Jika simpul akar adalah simpul solusi (*goal node*), maka solusi telah ditemukan. Stop.
2. Jika *Q* kosong, tidak ada solusi . Stop.
3. Jika *Q* tidak kosong, pilih dari antrian *Q* simpul *i* yang mempunyai $\hat{c}(i)$ paling kecil. Jika terdapat beberapa simpul *i* yang memenuhi, pilih satu secara sembarang.
4. Jika simpul *i* adalah simpul solusi, berarti solusi sudah ditemukan, stop. Jika simpul *i* bukan simpul solusi, maka bangkitkan semua anak-anaknya. Jika *i* tidak mempunyai anak, kembali ke langkah 2.
5. Untuk setiap anak *j* dari simpul *i*, hitung $\hat{c}(j)$, dan masukkan semua anak-anak tersebut ke dalam antrian *Q*.
6. Kembali ke langkah 2.

1.4 Pohon

Pohon adalah graf tak-berarah terhubung yang tidak mengandung sirkuit. Dengan setiap kemungkinan solusi dianggap sebagai sebuah simpul dan akar dari pohon, banyak algoritma yang menggunakan ruang solusi berupa pohon ini karena akan lebih memudahkan dalam penelusuran solusi yang ada berupa simpul-simpul pohon. Dalam contoh *assignment problem* dengan jumlah orang da job yang ada sebanyak 3, pohon yang menggambarkan persoalan ini adalah :



Gambar 1. Pohon *Assignment problem* dengan n=3

2. METODE

Kita ambil contoh penyelesaian persoalan *assignment problem* dengan $n = 4$ untuk ditinjau lebih lanjut. Misalkan terdapat *n* orang dan *n* buah job. Setiap orang akan di-assign dengan sebuah pekerjaan. Pengasan orang ke-i dengan job ke-j membutuhkan biaya sebesar *c(i,j).* Misalkan instansiasi persoalan dinyatakan sebagai matriks C sebagai berikut:

	Job 1	Job 2	Job 3	Job 4	
	8	2	7	9	Orang 1
C=	6	3	4	7	Orang 2
	1	8	5	4	Orang 3
	7	4	9	6	Orang 4

Terlebih dahulu kita akan mencari lower bound pada matriks ini. Mari kita tinjau matriks ini secara seksama. Untuk mencari lower bound dari persoalan ini, carilah 4 biaya paling minimum dari matriks ini. Mengapa harus 4? Karena mewakilkan setiap pekerjaan yang ada. Jadi dari matriks diatas, lower bound yang ada yaitu:

$\hat{c}$(root) = 1+2+3+4 = 10

yang artinya persoalan diatas paling tidak memiliki total biaya minimum 10. Pohon ruang status saat ini baru berisi satu buah simpul (akar) dengan nilai batas (bound) bernilai 10

Gambar 2. Akar Pohon *Assignment problem* dengan n=4

Selanjutnya, misalkan untuk orang ke-i dan job ke-j maka biaya minimum yang ada adalah $\hat{c}$(i,j) ditambah biaya-biaya minimum dari job dan orang yang tersisa (biaya minimum ini mewakilkan sebagai pencarian sisa solusi yang ada). Jadi persamaan dari fungsi pembatas yang ada adalah

$\hat{c}(i) = f(j) + \hat{g}(i)$

dalam hal ini ,

$\hat{c}(i)$ = biaya minimum untuk simpul *i,j*

$f(i)$ = biaya simpul *i,j*

$\hat{g}(i)$ = jumlah biaya minimum untuk mencapai simpul tujuan dari simpul akar i,j

Perhitungan selanjutnya untuk simpul-simpul yang lain pada pohon ruang status adalah sebagai berikut:

1. Simpul 2, orang ke-1, job ke-1

	Job 1	Job 2	Job 3	Job 4	
	-	-	-	-	Orang 1
C=	-	**3**	4	7	Orang 2
	-	8	**5**	**4**	Orang 3
	-	4	9	6	Orang 4

Karena yang kita pilih adalah $\hat{c}$(1,1) maka untuk setiap nilai pada baris 1 dan kolom 1 diubah menjadi – (artinya tidak dihiraukan untuk mencari biaya minimum yang lainnya). Nilai batas untuk simpul 2 pada pohon ruang status adalah :

$\hat{c}(1,1) = f(1,1) + \hat{g} = (8) + 3 + 5 + 4 = 20$

dimana angka 3, 5, 4 adalah biaya minimum setiap job yang tersisa

2. Simpul 3, orang ke-1, job ke-2

	Job 1	Job 2	Job 3	Job 4	
	-	-	-	-	Orang 1
C=	6	-	4	7	Orang 2
	1	-	**5**	**4**	Orang 3
	7	-	9	6	Orang 4

Nilai batas untuk simpul 3 pada pohon ruang status adalah :

$\hat{c}(1,2) = f(1,2) + \hat{g} = (2) + 1 + 5 + 4 = 12$

3. Simpul 4, orang ke-1, job ke-3

	Job 1	Job 2	Job 3	Job 4	
	-	-	-	-	Orang 1
C=	6	**3**	-	7	Orang 2
	1	8	-	**4**	Orang 3
	7	4	-	*6*	Orang 4

Nilai batas untuk simpul 4 pada pohon ruang status adalah :

$\hat{c}(1,3) = f(1,3) + \hat{g} = (7) + 1 + 3 + 4 = 15$

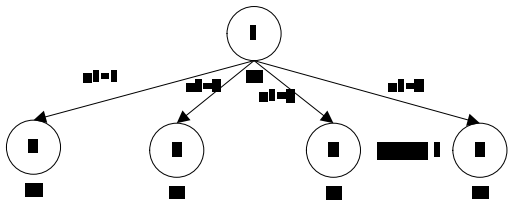
4. Simpul 5, orang ke-1, job ke-4

	Job 1	Job 2	Job 3	Job 4	
	-	-	-	-	Orang 1
C=	6	**3**	**4**	-	Orang 2
	1	8	5	-	Orang 3
	7	4	9	-	Orang 4

Nilai batas untuk simpul 5 pada pohon ruang status adalah :

$\hat{c}(1,4) = f(1,4) + \hat{g} = (9) + 1 + 3 + 4 = 17$

Pohon ruang status yang terbentuk sampai saat ini adalah:



Gambar 3. Pohon *Assignment problem* orang 1

Nilai pada setiap simpul di dalam pohon ruang status di atas bermakna sebagai berikut: Jika orang 1 dicocokan dengan job1 maka biaya yang diperlukan paling sedikit 20, dan seterusnya sampai dengan job 4 maka biaya minimum yang diperlukan adalah 17.

Sekarang simpul hidup yang ada adalah simpul 2, 3, 4, 5. Langkah selanjutnya dalam algoritma branch and bound adalah memilih simpul hidup yang memiliki *cost* paling kecil, yaitu simpul 3. Simpul 3 menjadi simpul-E yang akan diekspansi lagi. Perhitungan selanjutnya untuk simpul-simpul anak pada pohon ruang status dari simpul 3 yaitu sebagai berikut:

5. Simpul 6, orang ke-2, job ke-1

	Job 1	Job 2	Job 3	Job 4	
	-	-	-	-	Orang 1
C=	-	-	4	7	Orang 2
	-	-	**5**	**4**	Orang 3
	-	-	9	6	Orang 4

Karena yang kita pilih adalah $\hat{c}$(2,1) maka dan sebelumnya kita sudah memilih orang 1 dengan job ke 2 maka untuk setiap nilai pada baris 2 dan kolom 1 dan baris 1 kolom 2 diubah menjadi – (artinya tidak dihiraukan untuk mencari biaya minimum yang lainnya).

Nilai batas untuk simpul 6 pada pohon ruang status adalah :

$\hat{c}(2,1) = f(2,1) + \hat{g} = (2 + 6) + 4 + 5 = 17$

dimana 2 didapat dari hasil orang ke1 dengan job ke2 $\hat{c}(1,2)$ dan 6 didapat dari orang ke2 dengan job ke1. Angka 4 dan 5 adalah biaya minimum lainnnya yang menunjuk ke ruang solusi.

6. Simpul 7, orang ke-2, job ke-3

	Job 1	Job 2	Job 3	Job 4	
	-	-	-	-	Orang 1
C=	6	-	-	7	Orang 2
	1	-	-	**4**	Orang 3
	7	-	-	6	Orang 4

Nilai batas untuk simpul 7 pada pohon ruang status adalah :

$\hat{c}(2,3) = f(2,3) + \hat{g} = (2 + 4) + 1 + 4 = 11$

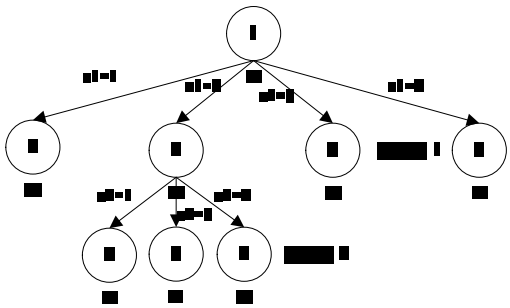
7. Simpul 8, orang ke-2, job ke-4

	Job 1	Job 2	Job 3	Job 4	
	-	-	-	-	Orang 1
C=	-	-	-	-	Orang 2
	1	-	**5**	-	Orang 3
	7	-	9	-	Orang 4

Nilai batas untuk simpul 8 pada pohon ruang status adalah :

$\hat{c}(2,4) = f(2,4) + \hat{g} = (2 + 7) + 1 + 5 = 15$

Pohon ruang status yang terbentuk sampai saat ini adalah:



Gambar 4. Pohon *Assignment problem* orang 1&2

Sekarang simpul yang hidup adalah simpul 2, 6, 7, 8, 4, 5. Langkah berikutnya kita mencari simpul dengan nilai $\hat{c}(i)$ terkecil, yaitu simpul 7 dengan nilai 11, maka simpul inilah yang akan kita kembangkan untuk menjadi simpul-E. Perhitungan selanjutnya untuk simpul-simpul anak pada pohon ruang status dari simpul 7 yaitu sebagai berikut:

8. Simpul 9, orang ke-3, job ke-1

	Job 1	Job 2	Job 3	Job 4	
	-	-	-	-	Orang 1
C=	-	-	-	-	Orang 2
	-	-	-	-	Orang 3
	-	-	-	**6**	Orang 4

Nilai batas untuk simpul 9 pada pohon ruang status adalah :

$\hat{c}(3,1) = f(3,1) + \hat{g} = (2 + 4 + 1) + 6 = 13$

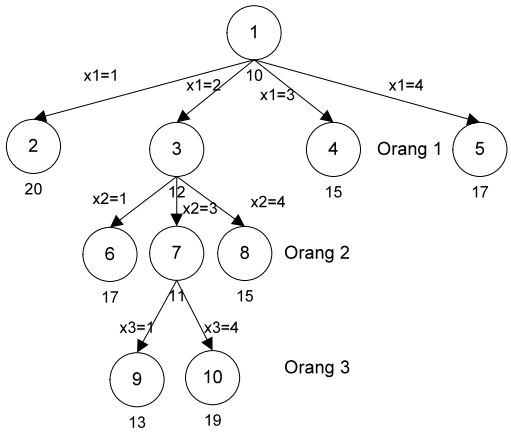
9. Simpul 10, orang ke-3, job ke-4

	Job 1	Job 2	Job 3	Job 4	
	-	-	-	-	Orang 1
C=	-	-	-	-	Orang 2
	-	-	-	-	Orang 3
	7	-	-	-	Orang 4

Nilai batas untuk simpul 10 pada pohon ruang status adalah :

$\hat{c}(3,4) = f(3,4) + \hat{g} = (2 + 4 + 6) + 7 = 19$

Pohon ruang status yang terbentuk sampai saat ini adalah:



Gambar 5. Pohon *Assignment problem* orang 1&2&3

Kita bisa lihat disini kalau simpul-simpul yang hidup adalah simpul 2, 6, 9, 10, 8, 4, dan 5. dengan melihat $\hat{c}(i)$ yang ada, maka yang paling kecil adalah simpul 9 dengan nilai 13, sehingga yang menjadi simpul-E adalah simpul 9. Berikut adalah pengembangan simpul 9:

10. Simpul 11, orang ke-4, job ke-4

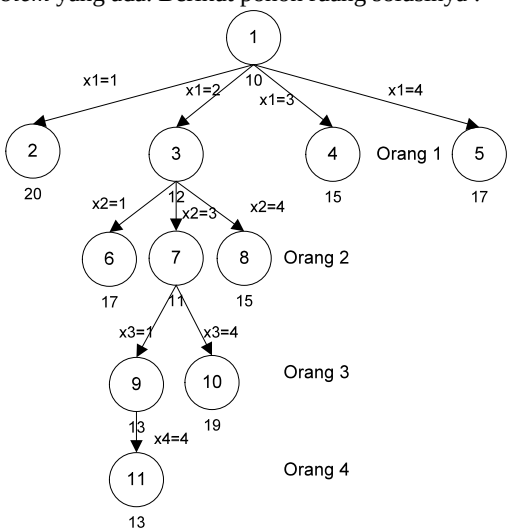
	Job 1	Job 2	Job 3	Job 4	
	-	-	-	-	Orang 1
C=	-	-	-	-	Orang 2
	-	-	-	-	Orang 3
	-	-	-	-	Orang 4

Nilai batas untuk simpul 11 pada pohon ruang status adalah :

$\hat{c}(4,4) = f(4,4) + \hat{g} = (2 + 4 + 1 + 6) = 13$

Karena dengan algoritma branch and bound ini sudah mencapai simpul solusi yang ada, sehingga Stop. Pohon

yang terbentuk merupakan sebuah solusi dari *assignment problem* yang ada. Berikut pohon ruang solusinya :



Gambar 6. Pohon Ruang Solusi *Assignment problem*

Jadi jawaban dari *assignment problem* dengan contoh di atas yaitu biaya minimum yang dikeluarkan adalah 13 dengan susunan penugasannya adalah sebagai berikut:
Orang 1 → *Job* 2 (2)
Orang 2 → *Job* 3 (4)
Orang 3 → *Job* 1 (1)
Orang 4 → *Job* 4 (6)

Total biaya *job* = 2 + 4 + 1 + 6 = 13

IV. KESIMPULAN

Banyak sekali algoritma yang dapat menyelesaikan persoalan ini. Contohnya algoritma BFS, tapi dengan algoritma BFS hanya akan berhenti sampai satu solusi ditemukan, dan belum tentu solusi itu yang paling optimum. Begitu juga dengan *brute force* yang mencari semua kemungkinan solusi, kemudian mencari hasil paling optimumnya sehingga harus kerja berkali-kali dengan kompleksitas waktu yang jauh lebih lama kalau ruang solusinya semakin dalam.

. Menurut saya algoritma paling tepat untuk mengatasi persoalan ini adalah dengan mengimplementasikan strategi algoritmik, tepatnya algoritma *Branch & Bound*. Dalam permasalahan ini, algoritma B&B dipakai untuk menentukan *cost* dari tiap simpul dengan menghitung *cost* yang diperlukan sampai simpul itu ditambah perkiraan *cost* minimum yang diperlukan dari simpul tersebut sampai ke simpul tujuan. Jika tidak, maka anaknya tidak akan dibangkitkan terlebih dahulu. Simpul dengan *cost* terkecillah yang dibangkitkan semua anakanya, sehinga pengerjaan menjadi lebih efisien, dan ditambah kita tidak perlu kerja dua kali karena solusi pertama yang ditemukan sudah pasti yang paling optimal.

REFERENSI

[1] Munir, Rinaldi, *Strategi Algoritmik*,Bandung, 2007.