J. Ris. & Ap. Mat. Vol. 10 No. 1 (2026) pp. 189-201
Jurnal Riset dan Aplikasi Matematika (JRAM)
e-ISSN: 2581-0154
DOI: doi.org/10.26740/jram.v10n1.p189-201

KOMBINASI VOGEL*'S APPROXIMATION METHOD* (VAM) DENGAN ALGORITMA *BRANCH AND BOUND*

IDA DAROJATUN, MIZAN AHMAD*, DAN BRYAN PUDJI HARTONO

Universitas Nahdlatul Ulama Al Ghazali, Cilacap, Indonesia

*Penulis Korespondensi: mizan.ahmad36@gmail.com

ABSTRAK

Penyelesaian masalah transportasi tidak selalu dapat dilakukan dengan satu metode atau algoritma saja. Dalam beberapa kasus, dibutuhkan penerapan lebih dari satu metode atau algoritma secara bersamaan. Penelitian ini, menunjukkan bahwa penggabungan antara Vogel*'s Approximation Method* (VAM) dan Algoritma *Branch and Bound* memungkinkan untuk dilakukan apabila empat asumsi: 1) terdapat minimal satu gudang yang kelebihan kapasitas dan satu gudang yang kekurangan kapasitas; 2) model transportasi (antargudang) selalu seimbang; 3) masing-masing gudang memiliki daerah pengiriman independen (agen); 4) Untuk setiap gudang mengirimkan ke setiap agennya tepat sekali dan kembali lagi ke gudang dalam sekali perjalanan.

Kata Kunci: Algoritma *Branch and Bound*, Optimasi Distribusi, VAM.

ABSTRACT

Transportation problems cannot always be solved with a single method or algorithm. In some cases, it is necessary to apply more than one method or algorithm simultaneously. This study shows that combining Vogel's Approximation Method (VAM) and the Branch and Bound Algorithm is possible if the first four assumptions are met. These assumptions are based on the characteristics of each VAM and the Branch and Bound Algorithm, as well as the compatibility between the two.

Keywords: *Branch and Bound Algorithm, Distribution Optimization, Vogel's Approximation Method (VAM).*

1. Pendahuluan

Masalah transportasi dapat dimisalkan m adalah jumlah barang yang harus dikirim ke tujuan n untuk memenuhi permintaan. Khususnya, sumber i memuat barang berjumlah a_i, dan tujuan j membutuhkan barang sebesar b_j. Diasumsikan bahwa sistem tersebut seimbang, artinya total pemasok sama dengan total permintaan [1]. Masalah ini dapat dirumuskan sebagai berikut.

$$\sum_{i=1}^{m} a_i = \sum_{j=1}^{n} b_j \tag{1}$$

Diasumsikan angka a_i dan b_j, $i = 1, 2, 3, \cdots, m$ dan $j = 1, 2, 3, \cdots, n$ adalah non-negatif dan dalam banyak aplikasi angka-angka tersebut sebenarnya merupakan bilangan bulat non-negatif. Terdapat bilangan satuan C_{ij} yang terkait dengan pengiriman dari asal i ke tujuan j. Salah satu masalah transportasi adalah menemukan pola distribusi antara asal dan tujuan yang memenuhi semua persyaratan dan meminimalkan total biaya distribusi [2].

Menurut G. Dantzig dan Thapa [3] masalah transportasi klasik adalah menentukan jadwal optimal dari pengiriman yang:

2020 *Mathematics Subject Classification*: 90C08; 90C10; 90C27; 90B06.
Diterima: 13-10-2025, direvisi: 17-05-2026, dimuat: 19-05-2026

a. Berasal dari sumber yang memiliki jumlah barang tertentu yang tersedia.
b. Barang dialokasikan dan dikirim langsung ke masing-masing tujuan di mana jumlah total yang diterima harus sama dengan jumlah total yang dibutuhkan.
c. *Supply* dan *demand* harus dihabiskan. Oleh karena, itu total permintaan yang diberikan harus sama dengan total persediaan yang diberikan.
d. Biaya setiap pengiriman dari sumber ke tujuan dihitung berdasarkan kapasitas gudang dan total biaya alokasi adalah jumlah biaya dari semua biaya.

Model transportasi selalu seimbang yaitu jumlah permintaan dan jumlah penawaran dari permintaan. Jika model tidak seimbang, maka sumber *dummy* atau tujuan *dummy* harus ditambah untuk mengembalikan keseimbangan [1]. Salah satu metode transportasi yang populer untuk menyelesaikan masalah transportasi adalah Vogel*'s Approximation Method* (VAM). VAM menjadi populer karena pada penerapannya menghasilkan solusi mendekati optimal atau bahkan optimal [3]. Beberapa penelitian terkait VAM diantaranya optimalisasi biaya transportasi pengiriman barang berbasis sistem informasi di PT. Coca-Cola Amatil Indonesia [4], optimalisasi efisiensi biaya transporrtasi pada CV Herba Sedunia [5], optimalisasi biaya transportasi pendistribusian produk *frozen food* menggunakan metode VAM dan metode *stepping stone* di PT. Ciomas Adisatwa Balikpapan [2]. Ada banyak penelitian terkait dengan optimasi distribusi dengan menggunakan metode VAM ([6], [7], [8], [9]).

Algoritma *Branch and Bound* yang pertama kali dikemukakan oleh A. H. Land dan A. G. Doig pada tahun 1960, digunakan untuk membagi suatu masalah menjadi sub-masalah lebih kecil kemudian mengarah pada penyelesaian dengan cara melakukan percabangan (*branching*) dan pembatasan (*bounding*) untuk memperoleh solusi yang optimal [2]. Beberapa penelitian terkait Algoritma *Branch and Bound* yaitu, penentuan rute terdekat pengiriman paket di J&T Express [10], optimalisasi produk tenun sa'be di toko sutera [11], penentuan jumlah produksi untuk memaksimalkan keuntungan [12], penerapan Algoritma *Branch and Bound* untuk jalur terpendek dan memaksimalkan keuntungan [13]. Ada banyak penelitian terkait dengan optimasi rute menggunakan Algoritma *Branch and Bound* ([14], [15], [16], [17], [18]).

Pada permasalahan transportasi terdapat VAM dan Algoritma *Branch and Bound* merupakan dua teknik yang digunakan untuk menyelesaikan masalah yang berbeda namun sama pentingnya. Meskipun kedua metode ini digunakan untuk konteks yang berbeda, namun memiliki tujuan yang sama yaitu solusi optimasi. Tetapi beberapa kasus tidak dapat diselesaikan menggunakan salah satu dari keduanya. Pada penelitian-penelitian yang telah disebutkan sebelumnya, penentuan rute/jalur distribusi hanya mneggunakan satu metode/algoritma saja. Namun, pada beberapa kasus tidak hanya membutuhkan satu metode/algoritma untuk menyelesaikan rute pendistribusian yang optimal, tertapi perlu untuk mengombinasikan beberapa metode atau algoritma. Penelitian ini difokuskan untuk mengombinasikan Algoritma VAM dan Algoritma *Branch and Bound* pada penentuan rute distribusi yang optimal.

2. Tinjauan Pustaka

2.1. Vogel*'s Approximation Method* (VAM)

Vogel*'s Approximation Method* (VAM) merupakan salah satu metode transportasi yang populer karena pada penerapannya menghasilkan solusi mendekati optimal atau bahkan optimal [19]. Berikut langkah-langkah VAM:

a. Hitung selisih C_{ij} dengan biaya terkecil pada seluruh baris atau kolom.
b. Tentukan penalti C_{ij} yang selisihnya terbesar. Jika terdapat nilai terbesar lebih dari satu, maka pilihlah sembarang yang dapat memindahkan barang paling banyak.
c. Alokasikan sebanyak mungkin pada sel yang mempunyai biaya terkecil pada C_{ij} yang dipilih. Jumlah alokasi adalah sebesar *supply* atau *demand* yang mana lebih kecil.

d. Jika alokasi telah memenuhi *supply* atau *demand*, maka hapus C_{ij} yang sudah memenuhi syarat sebelumnya.
e. Ulangi langkah a sampai d hingga semua alokasi terpenuhi.

2.2. Algoritma *Branch and Bound*

Algoritma *Branch and Bound* merupakan algoritma yang membagi suatu masalah menjadi sub-masalah lebih kecil yang kemudian mengarah pada penyelesaian dengan cara melakukan percabangan (*branching*) dan pembatasan (*bounding*) untuk mencapai solusi yang optimal [1]. Percabangan (*branching*) yaitu proses membentuk suatu masalah menjadi struktur pohon pencarian (*search tree*). Proses percabangan dilakukan untuk membangun semua cabang pohon yang mengarah ke solusi, sedangkan proses pembatasan dilakukan dengan menghitung estimasi nilai (*cost*) *vertex* dengan memperhatikan *edge*. Algoritma *Branch and Bound* pertama kali dikemukakan oleh A. H. Land dan A. G. Doig pada tahun 1960 [20]. Berikut langkah-langkah Algoritma *Branch and Bound*:

a. Masukkan *vertex* akar ke dalam antrian Q. Jika *vertex* akar merupakan *vertex* solusi (*goal node*), maka solusi ditemukan.
b. Jika Q kosong, maka tidak ada solusi.
c. Jika Q tidak kosong, pilih dari antrian Q *vertex* i dengan nilai $\hat{c}(i)$ terkecil. Jika terdapat *vertex* i lebih dari satu, pilih salah satu *vertex* sembarang.
d. Jika *vertex* i merupakan *vertex* solusi, berarti solusi sudah ditemukan. Jika *vertex* i bukan *vertex* solusi, maka bangkitkan semua *vertex* anak-anaknya. Jika i tidak mempunyai anak, kembali ke langkah b.
e. Untuk setiap *vertex* anak j dari vertex i, hitung $\hat{c}(j)$ lalu masukkan semua anak-anak tersebut ke dalam Q.
f. Kembali ke langkah b.

Misal $c(i)$ adalah nilai estimasi lintasan minimum dari *vertex* ke *vertex* tujuan, maka $c(i)$ menyatakan batas (*bound*) nilai pencarian solusi dari *vertex* i sehingga dapat dirumuskan fungsi heuristik untuk menghitung nilai (*cost*) sebagai berikut.

$$c(i) = f(i) + g(i)$$

dengan

$c(i)$: Nilai untuk *vertex* i
$f(i)$: Nilai lintasan dari *vertex* akar ke *vertex* i
$g(i)$: Nilai untuk mencapai *vertex* tujuan dari *vertex* i

Pada permasalahan yang lebih komplek, sistem digambarkan dengan matriks A berukuran $n \times n$. *Vertex* yang dieksplorasi adalah *vertex* dengan nilai batas terkecil. Nilai batas didapatkan dari reduksi baris dan kolom matriks A yang merepresentasikan graf. Reduksi dilakukan dengan mengurangi nilai C_{ij} pada baris atau kolom dengan nilai C_{ij} terkecil pada baris atau kolom tersebut, sedemikian sehingga didapatkan matriks tereduksi $A(t)$ dengan sebuah nilai nol pada setiap baris dan kolom. Matriks A dikatakan tereduksi apabila setiap kolom dan barisnya memuat minimal satu nilai nol dan semua elemen lainnya non-negatif [9]. Selanjutnya, total nilai pereduksi menjadi nilai batas simpul akar. Untuk setiap simpul anak yang dibangkitkan dengan mengunjungi $A(i,j)$, dilakukan reduksi matriks $A(t)$ untuk mendapatkan matriks tereduksi $A(j, \text{simpul awal})$ diubah menjadi ∞.

Berdasarkan persamaan fungsi heuristik untuk menghitung nilai estimasi yang telah dijelaskan sebelumnya, maka nilai batas *vertex* anak dihitung menggunakan rumus

$$\hat{c}(S) = \hat{c}(R) + A(i,j) + r$$

dengan

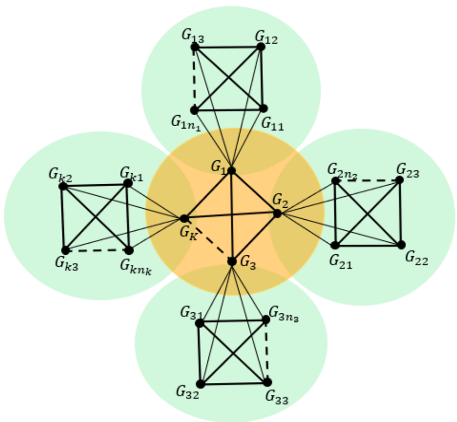
$\hat{c}(S)$: Nilai perjalanan minimum yang melalui *vertex* S, dimana S adalah anak dari R
$\hat{c}(R)$: Nilai perjalanan minimum yang melalui *vertex* R, dimana adalah orang tua dari S
$A(i,j)$: Bobot *edge* (i,j) pada matriks teredukasi

r : Jumlah semua preduksi pada proses memperoleh matriks tereduksi untuk *vertex* S

3. Hasil dan Pembahasan

3.1. Model Graf VAM-Algoritma *Branch and Bound* dan Asumsinya

Ide dasar untuk mengkombinasi dua metode bergantung pada kasusnya. Sebuah perusahaan mempunyai beberapa tempat produksi yang mana juga berfungsi sebagai gudang pemasok $(G_1, G_2, G_3, \cdots, G_k)$ di beberapa tempat berbeda. Setiap tempat produksi atau gudang memiliki kapasitas sendiri (produk atau kapasitas) dan wilayah distribusi (saling terpisah dan punya total kebutuhan). Jika setiap tempat produksi atau gudang mempunyai kapasitas yang kelebihan dan ada juga gudang atau tempat produksi yang kekurangan kebutuhan, maka gudang atau tempat produksi yang memilki kelebihan kebutuhan mengirimkan beberapa kelebihan produksinya ke gudang dengan kapasitas yang kurang dari kebutuhan. Masalah distribusi antar gudang dapat diselesaikan menggunakan masalah transportasi, dalam kasus ini VAM dapat menyelesaikannya, karena distribusi memuat kapasitas dan biaya. Lebih lanjut untuk setiap gudang mengirimkan ke setiap agennya tepat sekali dan kembali lagi ke gudang dalam sekali perjalanan. Dalam kasus ini Algoritma *Branch and Bound* dapat menyelesaikannya, dengan gudang yang menjadi titik awal karena proses distribusi dimulai dengan titik awal atau gudang, ke semua agen penjual dan hanya memuat biaya distribusi. Dengan demikian untuk menyelesaikan masalah itu sebuah kombinasi dari Algoritma VAM dan Algoritma *Branch and Bound* dapat digunakan. Sebelum mengkombinasi dua metode itu, perlu ditentukan model dan asumsi yang dibutuhkan. Berikut adalah model grafnya.



: VAM : Algoritma *Branch and Bound*

Gambar 1. Model kombinasi VAM dan Algoritma *Branch and Bound*

Berdasarkan permasalahan di atas dan model grafik, untuk dapat menggabungkan VAM dan Algoritma *Branch and Bound*, diperlukan asumsi-asumsi berikut.

1. Terdapat setidaknya satu gudang yang memiliki kelebihan kapasitas (kelebihan produksi/lebih dari jumlah yang dibutuhkan di area tersebut) yang disebut pemasok dan satu gudang yang kekurangan kapasitas (kekurangan produksi/kurang dari jumlah yang dibutuhkan di area tersebut) yang disebut penerima.
2. Model transportasi (antargudang) harus seimbang (jumlah yang di-*supply* = jumlah *demand*). Jika model tidak seimbang, maka harus ditambahkan variabel *dummy* untuk mengubahnya menjadi seimbang.
3. Setiap gudang (pengirim dan penerima) memiliki area pengirimannya sendiri (area pengiriman yang independen) dan menjadi *vertex* awal (untuk Algoritma *Branch and Bound*) di area pengirimannya sendiri.

4. Setiap gudang mengirimkan ke setiap agennya tepat sekali dan kembali lagi ke gudang dalam sekali perjalanan.

Selanjutnya, untuk menentukan solusi masalah ini, sederhananya VAM digunakan untuk menentukan solusi biaya minimal pada pendistribusian antar gudang dan Algoritma *Branch and Bound* digunakan untuk menentukan solusi rute pendistribusian terpendek dari setiap gudang ke setiap agen dari gudang tersebut. Berikut ini langkah-langkahnya.

1. Untuk semua X_i dengan $i = 1,2,\cdots,n$, tentukan solusi pendistribusian menggunakan VAM. Tujuannya untuk memenuhi kebutuhan setiap gudang.
2. Hitung B_K yang mana adalah biaya minimal distribusi antar gudang.
3. Untuk setiap $i = 1,2,\cdots,n$, tentukan rute distribusi dari X_i ke setiap X_{in_i} Algoritma *Branch and Bound* digunakan untuk setiap $j = 1,2,\cdots,n_i$.
4. Hitung B_{Ki} yang mana adalah biaya minimal distribusi dari X_i ke setiap X_{in_i}.
5. Hitung B yang mana adalah total biaya distribusi dengan rumus berikut.

$$B = B_K + \sum_{i=1}^{n} B_{Ki} \tag{1}$$

3.2. Simulasi

Berikut ini adalah contoh kasus yang dapat diselesaikan menggunakan VAM dan Algoritma *Branch and Bound*. Perum Bulog memiliki 4 tempat produksi yang juga merupakan gudang. Setiap tempat produksi (gudang) memiliki kapasitas produksi yang berbeda (kapasitas gudang). Berikut ini adalah jumlah produksi disetiap tempat produksi.

Tabel 1. Informasi gudang

Gudang	Jumlah Produksi	Jumlah yang Dibutuhkan	*Supply*	*Demand*
A	25	10	25 – 10 = 15	-
B	40	15	40 – 15 = 25	-
C	10	35	-	35 – 10 = 25
D	30	45	-	45 – 30 = 15

Berikut ini adalah biaya distribusi antar gudang atau tempat produksi.

Tabel 2. Biaya distribusi antar gudang atau tempat produksi

	A	B	C	D
A	-	10	30	25
B	10	-	15	20
C	30	15	-	35
D	25	20	35	-

Gudang A, B, C, D, masing-masing memiliki 2, 3, 2, dan 4 agen. Berikut ini adalah jarak antar gudang dan agen.

Tabel 3. Biaya distribusi antar gudang A ke masing-masing agen

	A	A_1	A_2
A	-	16	35
A_1	16	-	20
A_2	35	20	-

Tabel 4. Biaya distribusi antar gudang B ke masing-masing agen

	B	$\mathbf{B_1}$	$\mathbf{B_2}$	$\mathbf{B_3}$
B	-	18	4	15
B_1	18	-	11	7
B_2	4	11	-	2
B_3	15	7	2	-

Tabel 5. Biaya distribusi antar gudang C ke masing-masing agen

	C	$\mathbf{C_1}$	$\mathbf{C_2}$
C	-	25	30
C_1	25	-	13
C_2	30	13	-

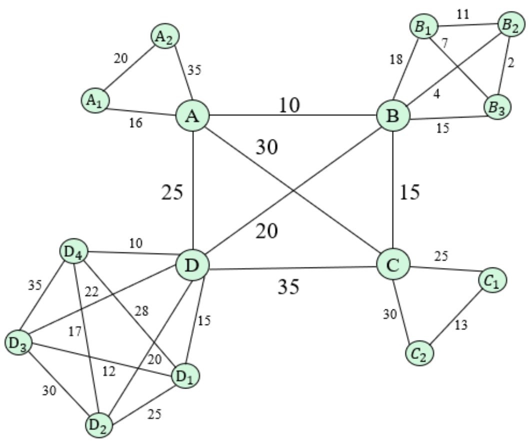
Tabel 6. Biaya distribusi antar gudang D ke masing-masing agen

	D	$\mathbf{D_1}$	$\mathbf{D_2}$	$\mathbf{D_3}$	$\mathbf{D_4}$
D	-	15	20	22	10
D_1	15	-	25	12	28
D_2	20	25	-	30	17
D_3	22	12	30	-	35
D_4	10	28	17	35	-

Perum Bulog harus mengirimkan semua hasil produksinya ke masing-masing agen dengan biaya minimum, dengan ketentuan setiap gudang atau tempat produksi mengirimkan ke setiap agennya tepat sekali dan kembali lagi ke gudang dalam sekali perjalanan.

3.3. Solusi

Berikut ini adalah model grafik untuk permasalahan di atas.



Gambar 2. Model grafik

Berdasarkan gambar di atas diperoleh

1. Ada dua gudang yang kelebihan kapasitas (A, B), dan dua gudang yang kekurangan kapasitas (C, D).
2. Model transportasi (antar dua gudang) seimbang (jumlah penawaran = jumlah permintaan).
3. Setiap gudang mempunyai daerah pengirimannya sendiri (agen) dan menjadi titik awal pada daerah pengirimannya sendiri.

4. Setiap gudang mengirimkan ke setiap agennya tepat sekali dan kembali lagi ke gudang dalam sekali perjalanan.

Oleh karena itu, keempat asumsi untuk dapat menggunakan kombinasi VAM dan Algoritma *Branch and Bound* terpenuhi.

Tabel 7. Biaya distribusi antar gudang (berdasarkan pasokan dan permintaan)

Gudang	**C**	**D**
A	30	25
B	15	20

Berdasarkan contoh di atas, diperoleh grafik biaya distribusi minimum menggunakan VAM.

Tabel 8. Iterasi 1 simulasi VAM

	C	**D**	***Supply***	**Penalti**
A	30	25	15	30-25=5
B	15	20	25	20-15=5
Demand	25	15		
Penalti	30-15=15	25-20=5		

Tabel 9. Iterasi 2 simulasi VAM

	C	**D**	***Supply***	**Penalti**
A	30	25	15	5
B	15(25)	20	25-25=0	5
Demand	25-25=0	15		
Penalti	15	5		

Tabel 10. Iterasi 3 simulasi VAM

	C	**D**	***Supply***	**Penalti**
A	-	25	15	5
B	15(25)	-	0	5
Demand	0	15		
Penalti	15	5		

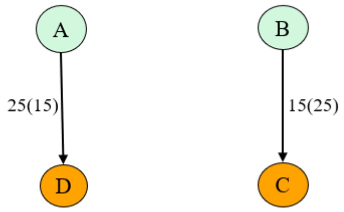
Tabel 11. Iterasi 4 simulasi VAM

	C	**D**	***Supply***	**Penalti**
A	-	25(15)	15-15=0	
B	15(25)	-	0	
Demand	0	15- 15=0		
Penalti				

Tabel 12. Iterasi 5 simulasi VAM

	C	**D**	***Supply***	**Penalti**
A	-	25(15)	0	
B	15(25)	-	0	
Demand	0	0		
Penalti				

Berdasarkan contoh kasus di atas, diperoleh grafik biaya minimum.



Gambar 3. VAM untuk gudang

Biaya distribusi minimum antar gudang diperoleh sebagai berikut:
1. Gudang A mendistribusikan 25 ke gudang D.
2. Gudang B mendistribusikan 15 ke gudang C.

Dengan biaya distribusi minimum antar gudang.

$$B_K = (15 \times 25) + (25 \times 15) = 750$$

Dengan demikian tidak ada gudang yang kekurangan stok.

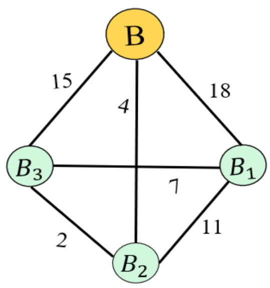
Selanjutnya, tentukan rute distribusi terpendek dari gudang ke masing-masing agen menggunakan Algoritma *Branch and Bound*. Berikut penentuan rute distribusi terpendek dari masing-masing gudang ke masing-masing agennya. Dalam hal ini ditunjukkan untuk gudang B, untuk gudang yang lainnya menggunakan metode yang sama.

Rute Pendistribusian Gudang B

Tabel 13. Gudang B ke masing-masing agen

Vertex	B	B_1	B_2	B_3
B	0	18	4	15
B_1	18	0	11	7
B_2	4	11	0	2
B_3	15	7	2	0

Tabel 13 merepresentasikan jarak antar B sampai B_2, kemudian diubah dalam bentuk graf sebagai berikut.



Gambar 4. Graf K_B

Gambar 4 merupakan representasi graf dari Tabel 13. Kemudian, diubah dalam bentuk matriks sebagai berikut.

$$M_1 = \begin{bmatrix} 0 & 18 & 4 & 15 \\ 18 & 0 & 11 & 7 \\ 4 & 11 & 0 & 2 \\ 15 & 7 & 2 & 0 \end{bmatrix}$$

Matriks M_1 merupakan representasi matriks dari Gambar 4. Selanjutnya, dicari rute terpendek dari B ke semua *vertex* kemudian kembali ke B menggunakan Algoritma *Branch and Bound* sebagai berikut.

a. Langkah 1

1. Memberikan label pada B ke B, B_1 ke B_1 dan seterusnya dengan label ∞ untuk setiap sel dalam matriks.

$$\begin{bmatrix} \infty & 18 & 4 & 15 \\ 18 & \infty & 11 & 7 \\ 4 & 11 & \infty & 2 \\ 15 & 7 & 2 & \infty \end{bmatrix}$$

2. Pada permasalahan minimalisasi terlebih dahulu mencari nilai terkecil di setiap baris, kemudian nilai terkecil tersebut digunakan untuk mengurangi semua nilai jarak yang ada pada baris yang sama.

$$\begin{bmatrix} \infty & 18 & 4 & 15 \\ 18 & \infty & 11 & 7 \\ 4 & 11 & \infty & 2 \\ 15 & 7 & 2 & \infty \end{bmatrix} \begin{matrix} B_1 - 4 \\ B_2 - 7 \\ B_3 - 2 \\ B_4 - 2 \end{matrix} \Rightarrow \begin{bmatrix} \infty & 14 & 0 & 11 \\ 11 & \infty & 4 & 0 \\ 2 & 9 & \infty & 0 \\ 13 & 5 & 0 & \infty \end{bmatrix}$$

3. Memastikan semua baris dan kolom sudah memiliki nilai nol. Apabila masih ada kolom yang belum memiliki nilai nol, maka cari nilai terkecil pada kolom tersebut lalu gunakan untuk mengurangi semua nilai yang ada pada kolom yang sama. Hasil matriks yang dikurangkan elemen terkecil di setiap baris dan setiap kolom disebut matriks (C_{ij}). Semua angka yang digunakan untuk mengurangi setiap baris dan setiap kolom tersebut dijumlahkan, kemudian hasil penjumlahan inilah dijadikan sebagai r (*root*) atau bobot dari *vertex* awal atau akar. Hal ini juga berarti bahwa solusi pada persoalan ini paling tidak memiliki bobot minimum sebesar r (*root*).

$$\begin{bmatrix} \infty & 14 & 0 & 11 \\ 11 & \infty & 4 & 0 \\ 2 & 9 & \infty & 0 \\ 13 & 5 & 0 & \infty \end{bmatrix} \begin{matrix} K_1 - 2 \\ K_2 - 5 \end{matrix} \Rightarrow \begin{bmatrix} \infty & 9 & 0 & 11 \\ 9 & \infty & 4 & 0 \\ 0 & 4 & \infty & 0 \\ 11 & 0 & 0 & \infty \end{bmatrix} = A$$

Total jumlah semua pengurangan $= 4 + 7 + 2 + 2 + 2 + 5 = 22$

$$\hat{c}(1) = 22$$

b. Langkah 2

Misalkan A adalah matriks tereduksi untuk *vertex* R. Misal adalah anak dari *vertex* R sedemikian sehingga *edge* (R, S) pada ruang pohon pencarian berkoresponden dengan *edge* (i, j). Jika bukan *vertex* daun, maka matriks bobot tereduksi untuk *vertex* S dapat dihitung dengan langkah sebagai berikut.

a. Ubah semua nilai pada baris i dan kolom j menjadi ∞, untuk mencegah agar tidak ada lintasan yang keluar dari *vertex* i atau masuk pada *vertex* j,
b. Ubah $A(j, 1)$ menjadi ∞, untuk mencegah penggunaan *edge* $(j, 1)$, dan
c. Reduksi kembali semua baris dan kolom pada matriks A kecuali untuk elemen ∞.

Jika r adalah total pengurangan, maka nilai batas untuk *vertex* S adalah

$$\hat{c}(S) = \hat{c}(R) + A(i, j) + r$$

Hasil reduksi ini menghasilkan matriks B.

Perhitungan bobot lintasan antar *vertex* dapat ditunjukkan sebagai berikut.

1. Lintasan B-B_1, $B(B, B_1) = 9$

$$\begin{bmatrix} \infty & \infty & \infty & \infty \\ \infty & \infty & 4 & 0 \\ 0 & \infty & \infty & 0 \\ 11 & \infty & 0 & \infty \end{bmatrix} = B$$

Nilai batas *vertex* 2 pada *search tree* adalah

$$\hat{c}(2) = \hat{c}(1) + B\,(B, B_1) + r = 22 + 9 + 0 = 31$$

2. Lintasan B-B_2, $B(B, B_2) = 0$

$$\begin{bmatrix} \infty & \infty & \infty & \infty \\ 9 & \infty & \infty & 0 \\ \infty & 4 & \infty & 0 \\ 11 & 0 & \infty & \infty \end{bmatrix} K_1 - 9 \Rightarrow \begin{bmatrix} \infty & \infty & \infty & \infty \\ 0 & \infty & \infty & 0 \\ \infty & 4 & \infty & 0 \\ 2 & 0 & \infty & \infty \end{bmatrix}$$

Diperoleh $r = 9$, sehingga nilai batas *vertex* 3 pada *search tree* adalah

$$\hat{c}(3) = \hat{c}(1) + B\,(B, B_2) + r = 22 + 0 + 9 = 31$$

3. Lintasan B-B_3, $B(B, B_3) = 11$

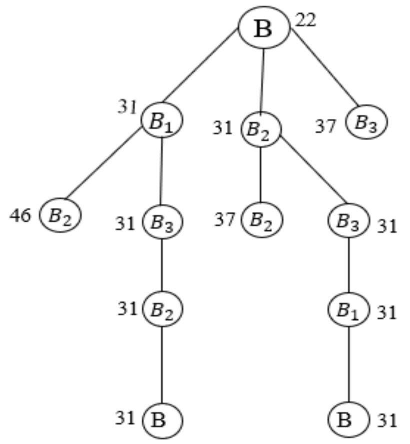
$$\begin{bmatrix} \infty & \infty & \infty & \infty \\ 9 & \infty & 4 & \infty \\ 0 & 4 & \infty & \infty \\ \infty & 0 & 0 & \infty \end{bmatrix} B_2 - 4 \Rightarrow \begin{bmatrix} \infty & \infty & \infty & \infty \\ 5 & \infty & 0 & \infty \\ 0 & 4 & \infty & \infty \\ \infty & 0 & 0 & \infty \end{bmatrix}$$

Diperoleh $r = 4$, sehingga nilai batas *vertex* 4 pada *search tree* adalah

$$\hat{c}(4) = \hat{c}(1) + B\,(B, B_3) + r = 22 + 11 + 4 = 37$$

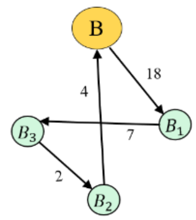
Dari 1-3, *search tree* terpendek bernilai sama 31, diperoleh dari B-B_1 dan B-B_2.

Iterasi dilanjutkan hingga berhenti dan diperoleh solusi terbaik yang dipilih adalah B- B_1-B_3-B_2-B atau B-B_2-B_3-B_1-B dengan total bobot minimal 31. Jadi, diperoleh panjang lintasannya 31.

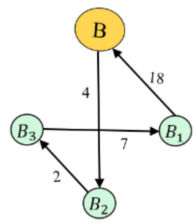


Gambar 5. Hasil akhir model graf B Algoritma *Branch and Bound*

Berdasarkan contoh kasus di atas, diperoleh rute sebagai berikut.



Gambar 6. Rute pendistribusian utama B



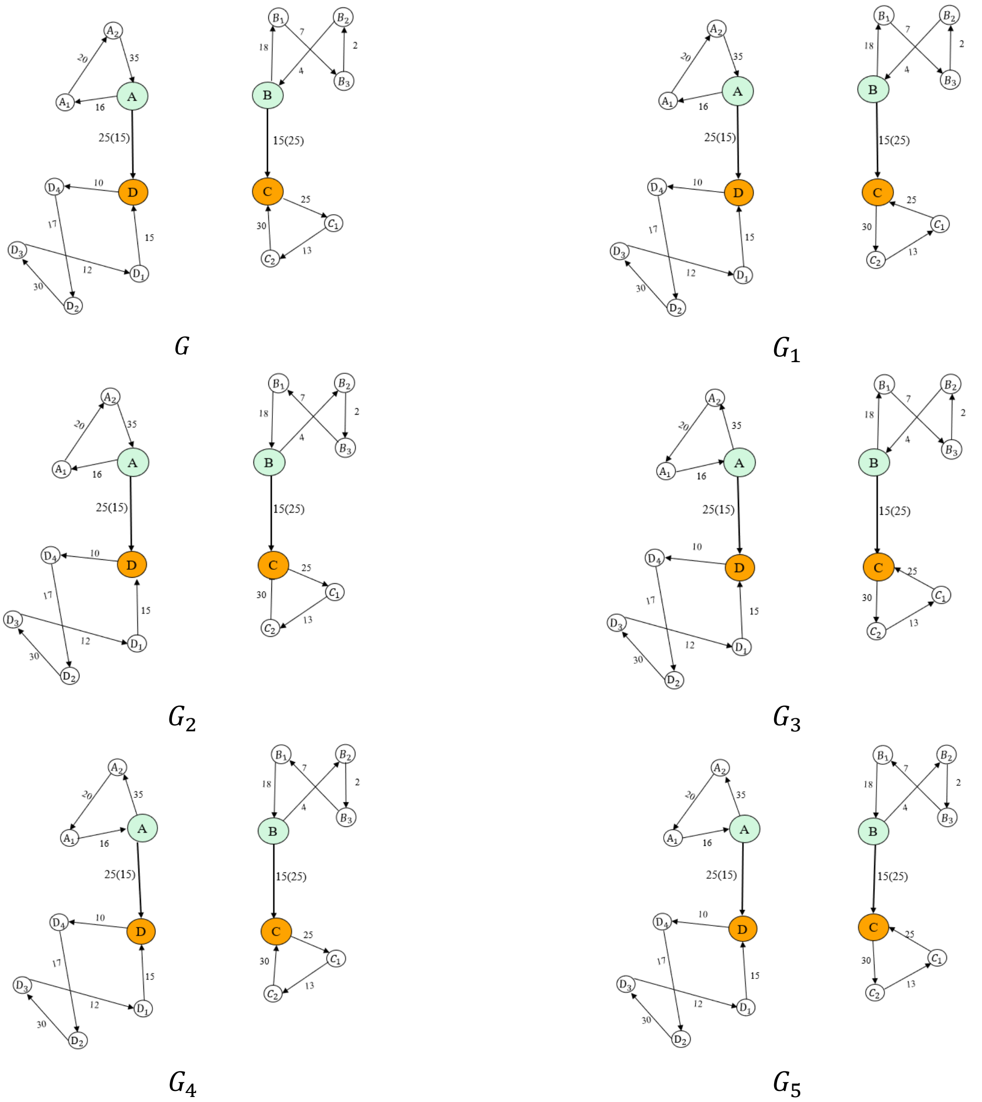
Gambar 7. Rute alternatif B

Berikut hasil rute pendistribusian terpendek dari setiap gudang ke setiap agennya menggunakan Algoritma *Branch and Bound*.

Tabel 14. Rute distribusi antar gudang

Gudang Awal	Agen Tujuan	Rute
A	$A \to A_1 \to A_2$	$A - A_1 - A_2 - A$ atau $A - A_2 - A_1 - A$
B	$B \to B_1 \to B_2 \to B_3$	$B - B_1 - B_3 - B_2 - B$ atau $B - B_2 - B_3 - B_1 - B$
C	$C \to C_1 \to C_2$	$C - C_1 - C_2 - C$ atau $C - C_2 - C_1 - C$
D	$D \to D_1 \to D_2 \to D_3 \to D_4$	$D - D_4 - D_2 - D_3 - D_1 - D$

Terakhir, berikut ini adalah graf solusi masalah yang diperoleh dengan menggabungkan VAM dan Algoritma *Branch and Bound*.



: Gudang yang kelebihan *supply* (gudang pemasok)

: Gudang yang kekurangan *supply* (gudang penerima)

: Agen

Gambar 8. Graf rute pendistribusian minimal (G), graf rute alternatif $(G_1, G_2, G_3, G_4, G_5)$

Biaya distribusi minimum dari setiap gudang ke masing-masing agennya adalah

$$B_{kA} = 35 + 20 + 16 = 71$$
$$B_{kB} = 18 + 7 + 2 + 4 = 31$$
$$B_{kC} = 25 + 13 + 30 = 68$$
$$B_{kD} = 10 + 12 + 30 + 17 + 15 = 84$$

Total biaya distribusi adalah

$$B = B_k + (B_{kA} + B_{kB} + B_{kC} + B_{kD}) = 750 + (71 + 31 + 68 + 84) = 1004$$

4. Kesimpulan

Vogel*'s Approximation Method* (VAM) dan Algoritma *Branch and Bound* dapat dikombinasikan apabila empat asumsi terpenuhi. Asumsi-asumsi tersebut didasarkan pada karakteristik masing-masing VAM dan Algoritma *Branch and Bound*, serta kesesuaian antara keduanya. Secara sederhana, keempat asumsi tersebut adalah: 1) terdapat minimal satu gudang yang kelebihan kapasitas dan satu gudang yang kekurangan kapasitas; 2) model transportasi (antargudang) selalu seimbang; 3) masing-masing gudang memiliki daerah pengiriman independen (agen); 4) Untuk setiap gudang mengirimkan ke setiap agennya tepat sekali dan kembali lagi ke gudang dalam sekali perjalanan.

Ucapan Terima Kasih

Terima kasih kepada penilai yang telah memberikan saran.

Daftar Pustaka

[1] H. A. Taha, *Operations Research*, New York: Pearson Education, 2017.

[2] Z. Anitasari et al., “Optimalisasi Biaya Transportasi Pendistribusian Produk Frozen Food Menggunakan Metode Vogel's Approximation Method (VAM) dan Metode Stepping Stone (Studi Kasus: PT. Ciomas Adisatwa Balikpapan)”, *BASIS J. Ilm. Mat.*,vol. 3, no. 1, pp. 61–71, 2024.

[3] G. B. Dantzig and M. N. Thapa, *Linear Programming, Introduction*. New York: Springer, 1997.

[4] S. Y. Prayogi and M. I. Panjaitan, “Penerapan Metode Vogel's Approximation Method (VAM) Dalam Optimalisasi Biaya Transportasi Pengiriman Barang Berbasis Sistem Informasi (Studi Kasus: PT. Coca-Cola Amatil Indonesia (CCAI) Medan)”, *J. Inf. Technol. Account*., vol. 5, no. 1, pp. 69–75, 2022.

[5] Ernis Riniawati et al., “Optimalisasi Efisiensi Biaya Transporrtasi Dengan Metode Vogel’s Approximation Method (VAM) Dalam Linear Programing Pada CV Herba Sedunia”, *J. Manuhara Pus. Penelit. Ilmu Manaj. dan Bisnis*, vol. 3, no. 1, pp. 234–246, 2025.

[6] F. Kurniawan, “Penerapan Metode Vogels’s Approximation method (VAM) Dalam Menetukan Harga Pengiriman Dokumen”, *J. Sci. Soc. Res.*, vol. 4307, no. 5, pp. 292–296, 2022.

[7] A. Roschyntawati et al., “Application of North West Corner, Least Cost and Vogel’s Approximation Method for Optimizing Transportation Cost of Biodiesel Between Biofuel Company and Blending Terminal”, *AIP Conference Proceedings.*, 2023.

[8] F. Muhtarulloh, S. N. Juliana, and E. R. Wulan, “Solusi Layak Awal Masalah

Transportasi Menggunakan Total Opportunity Cost Matrix-Modified Extemum Difference Method", *J. Sains Mat. dan Stat.*, vol. 9, no. 1, pp. 48 - 57, 2023.

[9] N. L. Azizah and M. Suryawinata, "Aplikasi Metode Transportasi Dalam Optimasi Biaya Distribusi Beras Sejahtera Pada PERUM BULOG Sub-Divre Sidoarjo", *Jurnal Ilmiah: SOULMATH.*, vol. 6, no. 1, pp. 15–24, 2018.

[10] K. W. Prayogo et al., "Implementasi Algoritma Branch & Bound Dalam Penentuan Rute Terdekat Pengiriman Paket Di J&T Express Blitar Khresna", *J. Ilm. Wahana. Pend.*, vol. 8, no. 9, pp. 10–20, 2024.

[11] Nurjanna et al., "Penerapan Algoritma Branch and Bound Dalam Optimalisasi Produk Tenun Sa'be", *JOMTA J. Math. Theory Appl.*, vol. 4, no. 1, pp. 8–14, 2022.

[12] Y. N. Firdaus et al., "Implementasi Algoritma Branch and Bound Dalam Penentuan Jumlah Produksi Untuk Memaksimalkan Keuntungan", *SETRING Satuan Tulisan Ris. dan Inov. Teknol.*, vol. 4, no. 1, pp. 65–70, 2019.

[13] A. R. Putri and M. Widyastiti, "Penerapan Algoritma Branch and Bound Untuk Jalur Terpendek dan Maksimalisasi Keuntungan", *INTERVAL J. Ilm. Mat.*, vol. 4, no. 2, pp. 86–98, 2024.

[14] W. Y. Oeitama and W. Y. Oeitama, "Optimasi Rute Pendistribusian Barang Menggunakan Kombinasi Algoritma Branch and Bound dan Cheapest Insertion Heuristic", *SQUARE J. Math. Math Educ.*, vol. 6, no. 2, pp. 89–104, 2024.

[15] J. I. Matematika and S. S. G. Witin, " Penerapan Algoritma Branch and Bound Untuk Optimasi Rute Wisata Di Kalimantan Timur Berdasarkan Traveling Salesman Problem", *MATH unesa, J. Ilm. Mat.*, vol. 13, no. 02, pp. 197–205, 2025.

[16] E. Safitri, S. Basriati, W. Widiarti, S. Sukmawati, "Kombinasi Algoritma Branch and Bound dan Cheapest Insertion Heuristic Dalam Mengoptimalkan Rute Distribusi Kurir Paket JNT Di Kecamatan Batang Cenaku", *J. Ilm. Pend. Mat. Mat. Stat. Lebesgue.*, vol. 4, no. 1, pp. 561–572, 2023.

[17] J. I. Matematika and S. S. G. Witin, "Penerapan Algoritma Branch and Bound Untuk Optimasi Rute Wisata Di Kalimantan Timur Berdasarkan Traveling Salesman Problem", *MATH unesa, J. Ilm. Mat.*, vol. 13, no. 02, pp. 197–205, 2025.

[18] Juliani and H. Hamrul, "Optimasi Distribusi Buku Menggunakan Algoritma Branch and Bound Untuk Efisiensi Rute Terpendek", *J. Comput. Inf. Syst.(J-CIS).*, vol. 5, no. 2, pp. 13–25, 2022.

[19] G. Chartrand, *Introductory Graph Theory*, New York: Dover Publication, Inc, 1985.

[20] S. Margiyani and N. S. M. Mussafi, "Aplikasi Algoritma Branch and Bound Untuk Optimasi Jalur Pemadam Kebakaran Kota Yogyakarta", *Encycl. Oper. Res. Manag. Sci.*, vol. 3, no. 1, pp. 75-85, 2014.