BAB V

PENERAPAN

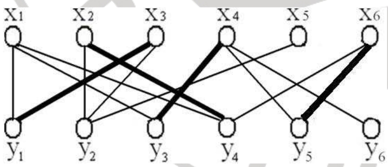
5.1 PERMASALAHAN PENUGASAN PEGAWAI

Dalam suatu perusahaan, *n* pekerja-pekerja X_1, X_2, ... X_3 tersedia untuk mengerejakan *n* pekerjaan-pekerjaan Y_1, Y_2, ... Y_3, masing-masing pekerja ter-kualifikasi untuk satu atau lebih dari pekerjaan-pekerjaan tersebut. Dapatkah semua orang-orang ditugaskan, satu orang per pekerjaan, ke pekerjaan-pekerjaan yang ter-kualifikasi bagi mereka.

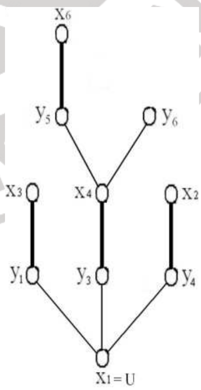
Kita membangun suatu graf bipartit *G* dengan bipartisi (X,Y), dimana $X = \{ x_1, x_2, \ldots x_3\}$, $Y = \{ y_1, y_2, \ldots y_3\}$, dan x_i dilekatkan ke y_j jika dan hanya jika pekerja X_i terkualifikasi untuk pekerjaan Y_j. Permasalahan menjadi satu dari menentukan apakah *G* memiliki *matching* sempurna atau tidak. Sesuai dengan **Teorema Perkawinan Hall (4.2)**, salah satu dari "*G* memiliki suatu *matching*" atau "ada suatu himpunan bagian *S* dari *X* sedemikian sehingga $| N(S) | < | S |$". Selanjutnya, kita akan menyajikan suatu algoritma untuk menyelesaikan permasalahan penugasan pekerja. Diberikan sembarang graf bipartit *G* dengan bipartisi (X,Y), Algoritma tersebut antara menemukan suatu *matching* dari *G* yang men-*saturate*-kan setiap titik dalam *X* atau, jika ini gagal, menemukan suatu himpunan bagian *S* dari *X* sedemikian sehingga $| N(S) | < | S |$.

Ide dasar dibalik algoritma adalah sangat sederhana. Kita memulai dengan suatu sembarang *matching M*. Jika *M* men-*saturate*-kan setiap titik dalam *X*, maka *M* tersebut adalah suatu *matching* dari tipe yang dibutuhkan. Jika tidak, kita memilih suatu titik *M-unsaturated u* dalam *X* dan mencari secara sistematis untuk suatu *M-augmenting path* dengan titik awal *u*. Metode pencarian, diuraikan dengan detail di bawah, menemukan suatu lintasan *P* jika lintasan tersebut ada; dalam kondisi ini $\tilde{M} = M \triangle E(P)$ adalah *matching* yang lebih besar dari pada *M*, dan karenanya men-*saturate*-kan lebih banyak titik-titik dalam *X*. Kita kemudian mengulang prosedur dengan $\tilde{M}$ sebagai ganti *M*. Jika lintasan tersebut tidak ada, maka himpunan *Z* dari semua titik-titik yang terhubung ke *u* oleh *M-alternating path-M-alternating path* ditemukan. Kemudian (seperti dalam pembuktian **Teorema 4.2**) $S = Z \cap X$ memenuhi $|N(S)| < |S|$.

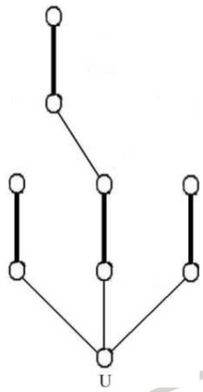
Misalkan *M* adalah *matching* dalam *G*, dan misalkan *u* adalah suatu titik *M-unsaturated* dalam *X*. Suatu pohon $H \subseteq G$ disebut suatu *M-alternating tree* yang akarnya adalah *u* jika (i) $u \in V(H)$, dan (ii) untuk setiap titik *v* dari *H*, (*u*,*v*)-*path* yang unik dalam *H* adalah suatu *M-alternating path*. Suatu *M-alternating tree* dalam suatu graf ditampilkan dalam Gambar 5.1.



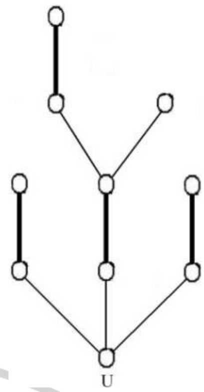
Gambar 5.1(a)
Suatu *matching M* dalam *G*



Gambar 5.1(b)
Suatu *M-alternating tree* dalam *G*



Gambar 5.2(a)
Kondisi (i)



Gambar 5.2(b)
Kondisi (ii)

Pencarian untuk *M-augmenting path* dengan titik asal u menyertakan pertumbuhan suatu *M-alternating tree* H yang akarnya adalah u. Prosedur ini pertama kali disarankan oleh Edmonds (1965). Pada awalnya, H hanya terdiri dari satu titik tunggal u. Selanjutnya dewasa dengan suatu cara, pada sembarang langkah, salah satu dari dua langkah berikut :

(i) semua titik-titik dari H kecuali u adalah *M-saturated* dan terjodohkan oleh M (seperti dalam Gambar 5.2(a)), atau

(ii) H memuat suatu titik *M-unsaturated* berbeda dengan u (seperti dalam Gambar 5.2(b)).

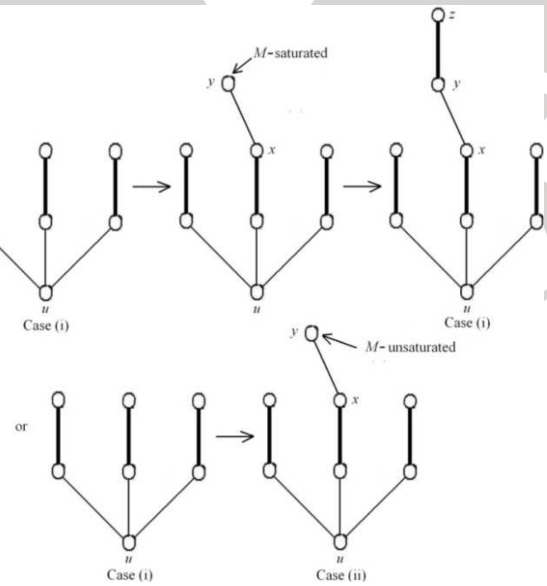
Jika yang terjadi kasus (i) (seperti pada awalan yang diberikan) kemudian, bentuklah himpunan $S = V(H) \cap X$ dan $T = V(H) \cap Y$, kita memiliki $N(S) \supseteq T$; akibatnya salah satu dari "$N(S) = T$" atau "$N(S) \supset T$".

(a) jika $N(S) = T$ maka, karena titik-titik dalam S-$\{u\}$ terjodohkan dengan titik-titik dalam T, $|N(S)| = |S|-1$, menandakan bahwa G tidak memiliki *matching* yang men-*saturate*-kan semua titik-titik dalam X.

(b) Jika $N(S) \supset T$, ada suatu titik y dalam $Y \backslash T$ berajasen ke suatu titik x dalam S. Karena semua titik-titik dari H kecuali u adalah terjodohkan oleh M, salah satu dari "$x = u$" atau "x terjodohkan dengan suatu titik dari H". Oleh karenanya $xy \notin M$. Jika y adalah *M-saturated*, dengan $yz \in M$, kita menumbuhkan H dengan penambahan titik-titik y dan z serta garis-garis xy dan yz. Kita kemudian kembali pada kasus (i). Jika y adalah *M-saturated*, kita menumbuhkan H dengan penambahan titik y dan garis xy, hasilnya pada kasus (ii). Akibatnya diperoleh bahwa (u,y)-*path* dari H adalah suatu *M-augmenting path* dengan titik awal u, seperti yang diinginkan.

Gambar 5.3 mengilustrasikan prosedur penumbuhan pohon di atas.

Algoritma yang diuraikan di atas dikenal sebagai Metode Orang Hungaria, dan dapat diringkaskan sebagai beriku :



Gambar 5.3
Prosedur penumbuhan pohon

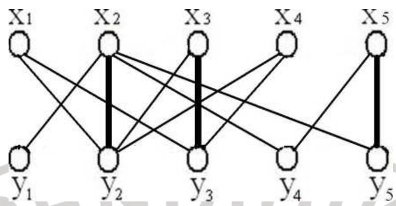
Kita mulai dengan suatu sembarang *matching* M.

1. Jika M men-*saturate*-kan setiap titik dalam X, berhenti. Jika tidak, misalkan u adalah suatu titik M-*unsaturated* dalam X. Misalkan $S = \{u\}$ dan $T = \emptyset$.

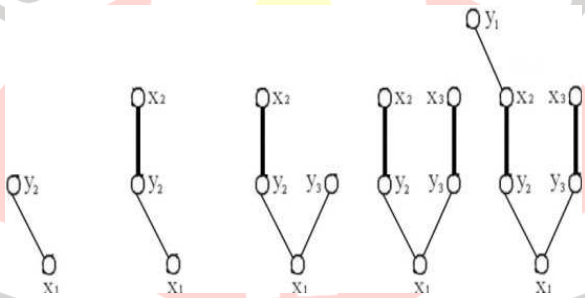
2. Jika $N(S) = T$ maka $| N(S) | < | N |$, karena $| T | = | S | - 1$. berhenti, karena berdasarkan Teorema milik Hall tidak ada *matching* yang men-*saturate*-kan setiap titik dalam X. Jika tidak, misalkan $y \in N(S)\backslash T$.

3. Jika y adalah M-*saturated*, misalkan $yz \in M$. Ganti S dengan $S \cup \{z\}$ dan T dengan $T \cup \{y\}$ dan pergi ke langkah 2. (periksa bahwa $| T | = | S | - 1$ dipertahankan setelah penggantian ini). Jika tidak, misalkan P adalah M-*augmenting* (u,y)-*path*. Ganti M dengan $\hat{M} = M \Delta E(P)$ dan pergi ke langkah 1.

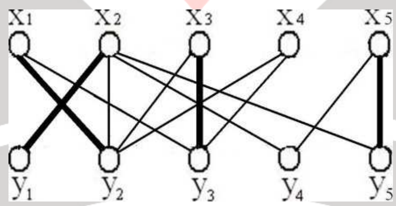
Pertimbangkan, untuk contoh, graf G dalam Gambar 5.4(a), dengan *matching* awalan $M = \{x_2y_2, x_3y_3, x_5y_5\}$. Dalam Gambar 5.4(b) suatu M-*alternating tree* ditumbuhkan, dimulai dengan x_1, sehingga diperoleh *augmenting path* $x_1y_2x_2y_1$. Hasil ini pada suatu *matching* yang baru $\hat{M} = \{x_1y_2, x_2y_1, x_3y_3, x_5y_5\}$, dan suatu $\hat{M}$-*alternating tree* sekarang ditumbuhkan dari x_4 (Gambar 5.4(c) dan Gambar 5.4(d)) karena tidak ada $\hat{M}$–*augmenting path* dengan titik awal x_4, algoritma berhenti / berakhir. Himpunan $S = \{x_1, x_3, x_4\}$, dengan himpunan tetangga $N(S) = \{y_2, y_3\}$, menunjukan bahwa G tidak memiliki *matching* sempurna.



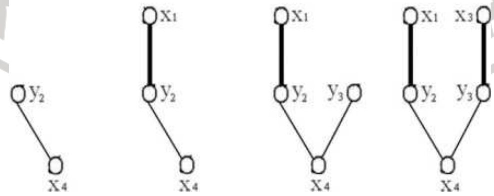
Gambar 5.4(a)
Matching M



Gambar 5.4(b)
Suatu *M-alternating tree*

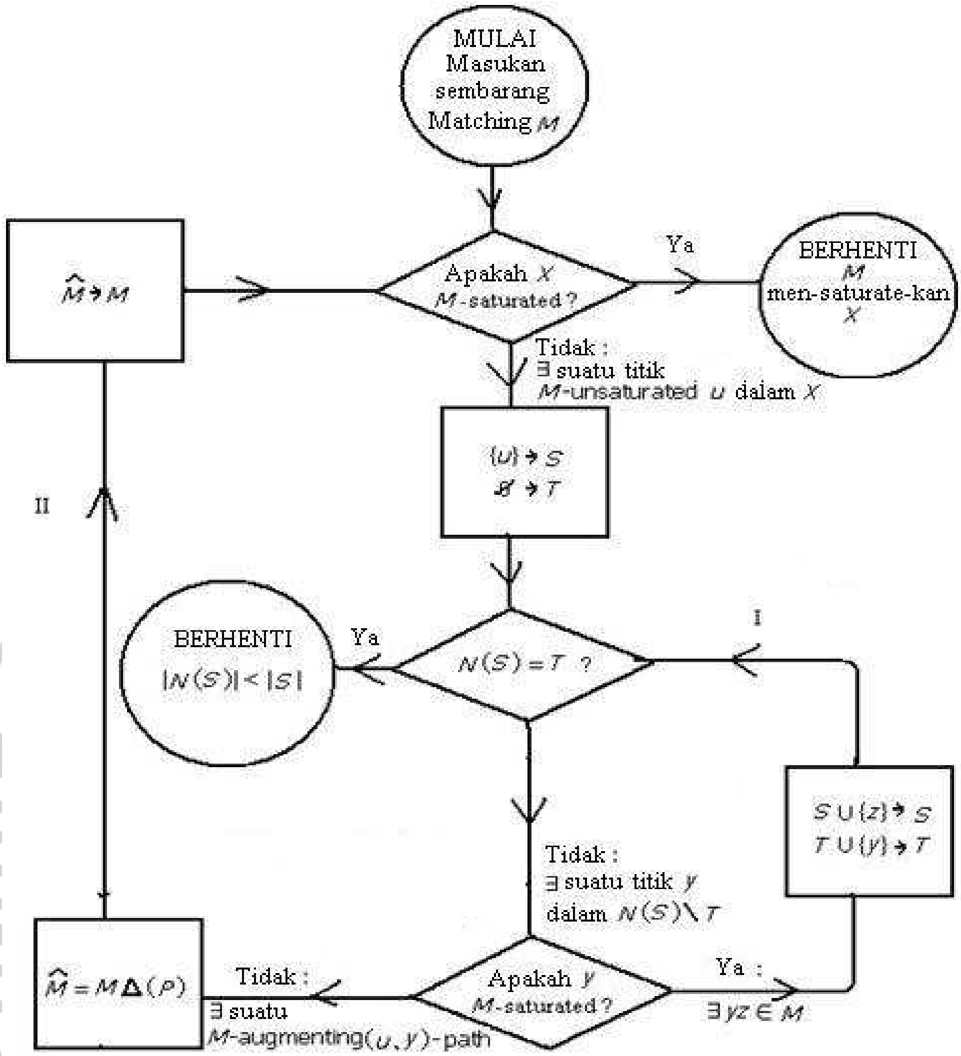


Gambar 5.4(c)
Matching $\tilde{M}$



Gambar 5.4(d)
Suatu $\tilde{M}$-*alternating tree*

Suatu diagram alur dari metode orang Hungaria diberikan dalam Gambar 5.5. Karena algoritma tersebut dapat mensiklus melalui prosedur penumbuhan pohon, I, paling banyak | X | kali sebelum menemukan salah satu dari "suatu $S \subseteq X$ sedemikian sehingga $|N(S)| < |S|$" atau "suatu *M-augmenting path*, dan karena *matching* awalan dapat di-*augment*-kan paling banyak | X | kali sebelum suatu *matching* dari tipe yang dibutuhkan ditemukan, itu jelas bahwa metode orang Hungaria adalah algoritma yang bagus.



Gambar 5.5
Metode orang Hungaria (Bondy and Murty, 1976 : 85)

5.2 PERMASALAHAN PENUGASAN OPTIMAL

Metode orang Hungaria, diuraikan dalam sesi 5.1, adalah suatu cara efisien dari menentukan suatu penugasan layak dari para pekerja ke pekerjaan-pekerjaan, jika satu ada. Bagaimanapun satu mungkin, dalam penambahan, berharap mempertimbangkan keefektifan dari para pekerja dalam berbagai pekerjaan-pekerjaan (diukur, mungkin, berdasarkan keuntungan perusahaan). Dalam kondisi ini, satu tertarik dalam suatu penugasan yang memaksimalkan keefektipan keseluruhan dari para pekerja. Masalah dari pencarian seperti suatu penugasan dikenal sebagai “masalah penugasan yang optimal”.

Mempertimbangkan suatu graf bipartit lengkap terboboti dengan bipartisi (X,Y), dimana $X = \{x_1, x_2, ..., x_n\}$, $Y= \{y_1, y_2, ..., y_n\}$ dan garis x_iy_j memiliki bobot $w_{ij} = w(x_iy_j)$, keefektipan dari pekerja X_i dalam pekerjaan Y_j. “Masalah penugasan yang optimal” jelas sama saja dengan pencarian “suatu *matching* sempurna berat-maksimum” dalam graf terboboti ini.

Untuk menyelesaikan “masalah penugasan yang optimal” itu , tentu saja, mungkin untuk menghitung satu-satu n! *matching-matching* sempurna dan menemukan suatu optimal satu diantara mereka. Bagaimanapun, untuk n sangat besar, maka prosedur tidak efisien. Dalam sesi ini kita akan menyajikan algoritma yang baik untuk pencarian suatu *matching* yang optimal dalam suatu graf bipartit lengkap terboboti.

Kita mendefinisikan suatu “pelabelan titik yang layak” sebagai suatu fungsi l bernilai-real pada himpunan titik $X \cup Y$ sedemikian sehingga, untuk semua $x \in X$ dan $y \in Y$

$$l(x) + l(y) \geq w(xy) \qquad (5.1)$$

(Bilangan real $l(v)$ disebut “label” dari titik v). Suatu pelabelan titik yang layak adalah suatu pelabelan dari titik-titik sedemikian sehingga jumlah dari label-label dari dua titik-titik ujung dari suatu garis paling sedikit sama banyaknya dengan bobot dari garis. Tidak peduli berapapu bobot dari garisnya, selalu ada suatu pelabelan titik yang layak; seperti fungsi l yang diberikan oleh

$$l(x) = \text{maks}_{y \in Y}\, w(xy) \qquad \text{jika } x \in X \qquad (5.2)$$

$$l(y) = 0 \qquad \text{jika } y \in Y$$

Jika l adalah suatu “pelabelan titik yang layak”, E_l adalah himpunan dari garis-garis yang memenuhi persamaan (5.1); yaitu

$$E_l = \{xy \in E \mid l(x) + l(y) \geq w(xy)\}$$

Graf bagian pembangun dari G dengan himpunan garis E_l diacu pada “kesamaan graf bagian” berkorespondensi dengan pelabelan titik yang layak l, yang dinotasikan oleh G_l. Hubungan antara kesamaan graf bagian dan *matching*-*matching* optimal disediakan oleh teorema berikut.

Teorema 5.1

Misalkan l adalah suatu pelabelan titik yang layak dari G. Jika G_l memuat suatu *matching* sempurna M^*, maka M^* adalah suatu *matching* optimal dari G.

Bukti :

Misalkan G_l memuat suatu *matching* sempurna M^*. Karena G_l adalah suatu graf bagian pembangun dari G, M^* juga merupakan *matching* sempurna dari G. Sekarang

$$W(M^*) = \sum_{e \in M^*} w(e) = \sum_{v \in V} l(v) \qquad (5.3)$$

Karena masing-masing $e \in M^*$ anggota dari kesamaan graf bagian dan titik-titik ujung dari garis-garis dari M^* menutup masing-masing titik tepat satu kali. Di lain pihak, jika M adalah sembarang *matching* sempurna dari G, maka

$$W(M) = \sum_{e \in M_*} w(e) = \sum_{v \in V} l(v) \qquad (5.4)$$

Berdasarkan (5.3) dan (5.4) bahwa $W(M^*) \geq W(M)$. Akibatnya M^* adalah suatu *matching* optimal. ∎

Algoritma di atas adalah dasar dari suatu algoritma, dalam kaitan dengan Kuhn (1955) dan Munkres (1957), untuk pencarian suatu *matching* optimal dalam graf bipartit lengkap yang terboboti. Langkah-langkah kita mendekati hasil yang diperoleh Edmonds (1967).

Algoritma ini dimulai dengan suatu sembarang pelabelan titik yang layak l (sebagai contoh, satu yang diberikan (5.2)), kita tetapkan G_l, pilih suatu *matching* M sembarang dalam G_l dan terapkan metode orang Hungaria. Jika suatu *matching* sempurna ditemukan dalam G_l kemudian, berdasarkan **Teorema 5.1**, *matching* ini adalah optimal. Jika tidak, metode orang Hungaria berakhir dalam suatu *matching* M' yang tidak sempurna, dan suatu M'*-alternating tree* H yang tidak memuat M'*-augmenting path* dan tidak dapat ditumbuhkan lebih lanjut (dalam G_l). Kita kemudian memodifikasi l ke suatu $\hat{l}$ pelabelan titik yang layak dengan sifat yang keduanya "M'" dan "H" tidak termuat dalam dalam G_l dan H dapat diperluas dalam G_l. Seperti modifikasi dalam pelabelan titik yang layak dibuat baik diperlukan ataupun tidak, sampai suatu *matching* sempurna ditemukan dalam kesamaan graf bagian.

$$\begin{bmatrix} 3 & 5 & 5 & 4 & 1 \\ 2 & 2 & 0 & 2 & 2 \\ 2 & 4 & 4 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 \\ 1 & 2 & 1 & 3 & 3 \end{bmatrix}$$

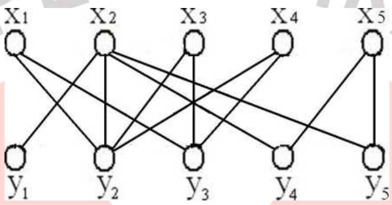
Gambar 5.6(a)

$$\begin{array}{c} \begin{bmatrix} 3 & 5 & 5 & 4 & 1 \\ 2 & 2 & 0 & 2 & 2 \\ 2 & 4 & 4 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 \\ 1 & 2 & 1 & 3 & 3 \end{bmatrix} \begin{matrix} 5 \\ 2 \\ 4 \\ 1 \\ 3 \end{matrix} \\ \begin{matrix} 0 & \;0 & \;0 & \;0 & \;0 \end{matrix} \end{array}$$

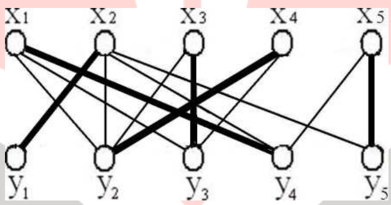
Gambar 5.6(b)

$$\begin{array}{c} \begin{bmatrix} 3 & 5 & 5 & 4 & 1 \\ 2 & 2 & 0 & 2 & 2 \\ 2 & 4 & 4 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 \\ 1 & 2 & 1 & 3 & 3 \end{bmatrix} \begin{matrix} 4 \\ 2 \\ 3 \\ 0 \\ 3 \end{matrix} \\ \begin{matrix} 0 & \;1 & \;1 & \;0 & \;0 \end{matrix} \end{array}$$

Gambar 5.6(c)



Gambar 5.6(d)



Gambar 5.6(e)

Berikut ini adalah algoritma dari Kuhn-Munkres.

Algoritma Kuhn-Munkres

Mulai dengan suatu l pelabelan titik yang layak dengan sembarang, menentukan G_l, dan memilih suatu *matching* M secara sembarang dalam G_l.

1. Jika X adalah *M-saturated*, maka M adalah suatu *matching* sempurna (karena $|X| = |Y|$) dan karenanya, berdasarkan **Teorema 5.1**, suatu *matching* optimal; dalam kondisi ini, berhenti. Jika tidak, misalkan u adalah suatu titik *M-unsaturated*. Misalkan $S = \{u\}$ dan $T = \emptyset$.

2. Jika $N_{Gl}(S) \supset T$, pergi ke langkah 3. Jika tidak, $N_{Gl}(S) = T$. Hitung

$$\alpha_1 = \min_{\substack{x \in S \\ y \notin T}} \{l(x) + l(y) - w(xy)\}$$

   dan l pelabelan titik yang layak diberikan oleh

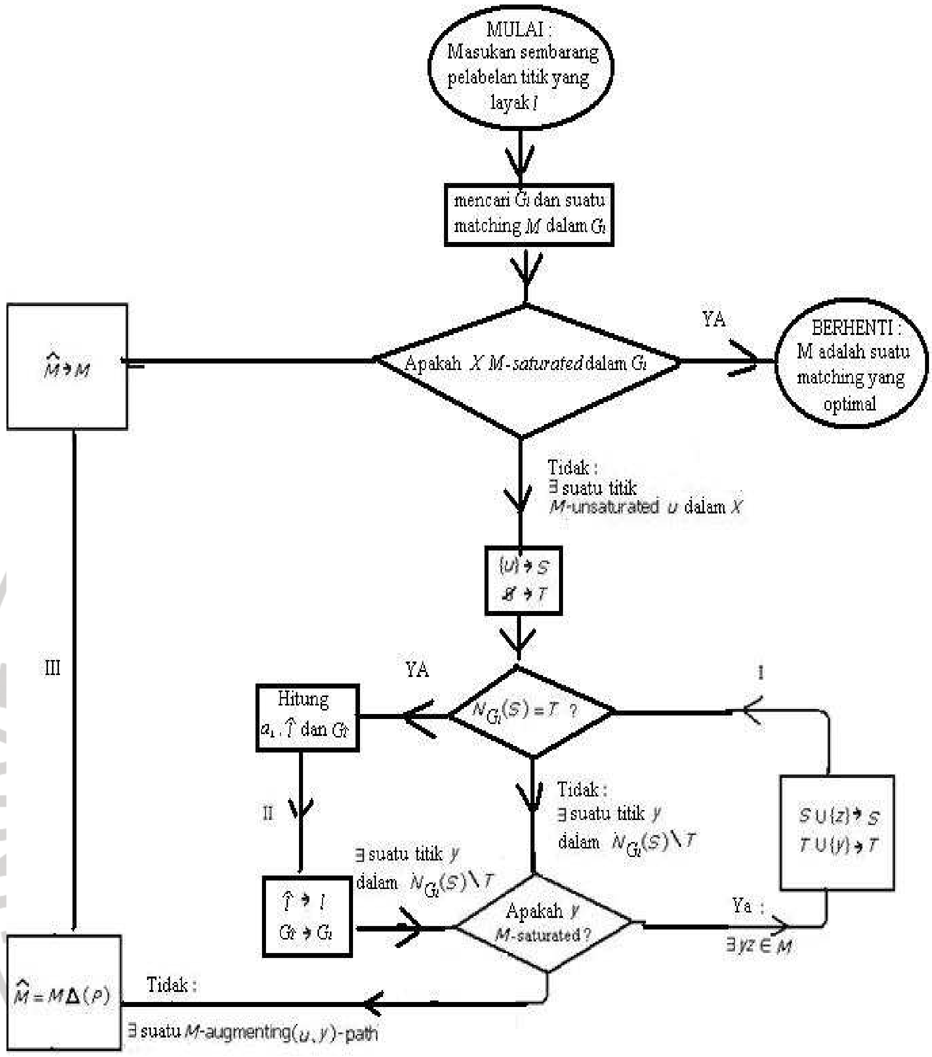
$$\hat{l}(v) = \begin{cases} l(v) - \alpha_1 \text{ jika } v \in S \\ l(v) + \alpha_1 \text{ jika } v \in T \\ \quad l(v) \text{ yang lain} \end{cases}$$

   (catatan bahwa dan $\alpha_1 > 0$ bahwa $N_{Gl}(S) \supset T$). Ganti l dengan $\hat{l}$ dan G_l dengan $G_{\hat{l}}$.

3. Memilih suatu titik y dalam $N_{Gl}(S)\backslash T$. Seperti dalam prosedur penumbuhan pohon dari sesi 5.1, mempertimbangkan apakah y adalah *M-saturated* atau bukan. Jika y adalah *M-saturated*, dengan $yz \in M$, ganti S dengan $S \cup \{z\}$, dan pergi ke langkah 2. Jika tidak, misalkan P adalah suatu *M-augmenting* (u,y)-*path* dalam G_l, ganti M dengan $\hat{M} = M \,\Delta\, E(P)$, dan pergi ke langkah 1.

Dalam pengilustrasian algoritma Kuhn-Munkres, itu sangat tepat untuk menyajikan suatu graf bipartit lengkap yang terboboti G dengan suatu matriks $W = [w_{ij}]$, dimana w_{ij} adalah bobot dari garis x_iy_j dalam G. Kita akan mulai dengan matriks dari Gambar 5.6(a). Dalam Gambar 5.6(b), pelabelan titik yang layak (5.2) ditunjukan (dengan penempatan label dari x_i ke sebelah kanan dari baris i dari matriks dan label dari y_i dibawah kolom j) dan entri-entri berkorespondensi ke titik-titik dari kesamaan graf bagian yang terkait ditandai; kesamaan graf bagian itu sendiri digambarkan (tanpa bobot) dalam Gambar 5.6(c). Itu ditunjukan dalam sesi sebelumnya bahwa graf ini tidak memiliki (himpunan $S = \{x_1, x_3, x_4\}$ memiliki himpunan tetangga $\{y_1, y_3\}$). Kemudian kita memodifikasi awalan pelabelan titik yang layak kita ke satu diberikan dalam Gambar 5.6(d). Suatu penerapan dari metode orang Hungaria sekarang menunjukan bahwa kesamaan graf bagian yang terkait (Gambar 5.6(e)) memiliki *matching* sempurna $\{x_1y_4, x_2y_1, x_3y_3, x_4y_2, x_5y_5\}$. Ini oleh karenanya suatu *matching* optimal dari G.

Suatu diagram alir untuk algoritma Kuhn-Munkres diberikan dalam Gambar 5.7. dalam siklus II, banyaknya perhitungan-perhitungan yang dibutuhkan untuk menghitung G_l adalah dengan jelas dari order v^2. Karena algoritma dapat mensiklus melalui I dan II paling banyak $|X|$ kali sebelum penemuan suatu *M-augmenting path*, dan karena *maching* yang awal dapat di-*augment*-kan paling banyak $|X|$ kali sebelum suatu matching yang optimal ditemukan, kita setuju bahwa algoritma Kuhn-Munkres adalah algoritma yang baik.



Gambar 5.7
Algoritma Kuhn-Munkres (Bondy and Murty, 1976 : 89)