BAB II

MATERI PENUNJANG

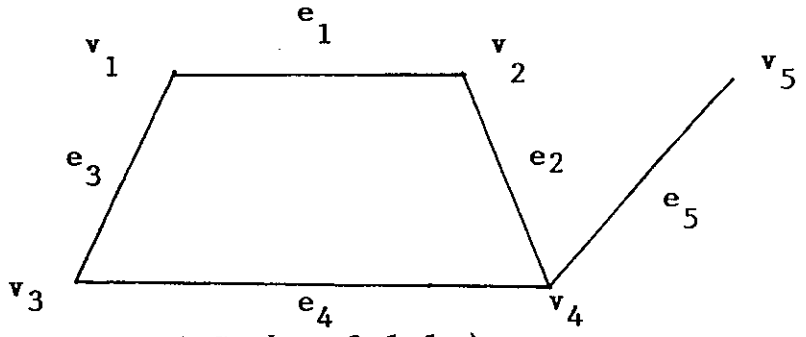
2.1 GRAPH TAK BERARAH

2.1.1 PENGERTIAN

Definisi 2.1.1

Suatu graph G dinotasikan G=(V,E) adalah himpunan vertex-vertex (V), dimana V = $\{v_1, v_2, v_3, \ldots, v_n\}$ berhingga dan tidak kosong dinotasikan V(G), dan himpunan edge-edge (E) dimana $E=\{e_1,e_2,e_3,\ldots,e_n\}$ berhingga dinotasikan E(G).

Contoh :



(Gambar 2.1.1)

Gambar 2.1.1 adalah suatu graph dengan :

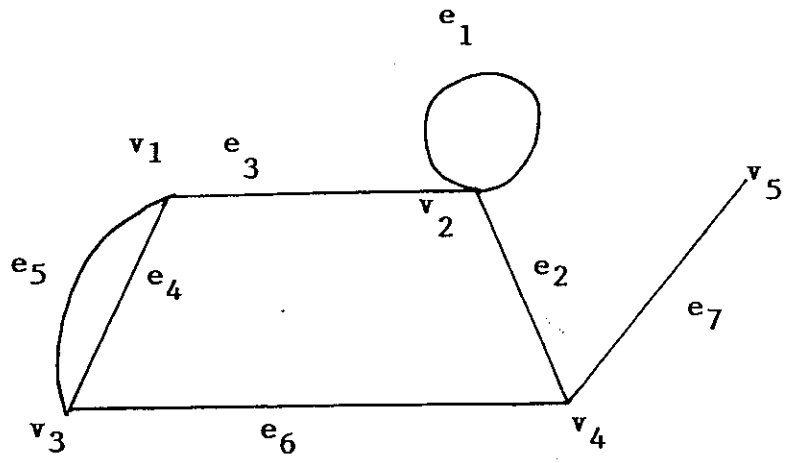
$V = \{ v_1, v_2, v_3, v_4, v_5 \}$

$E = \{ e_1, e_2, e_3, e_4, e_5 \}$

Definisi 2.1.2

Dua atau lebih edge yang menghubungkan pasangan vertex yang sama disebut edge ganda (multiple edge), dan suatu edge yang menghubungkan suatu vertex ke vertex itu sendiri disebut loop. Suatu graph tanpa loop atau edge ganda disebut graph sederhana atau graph.

Contoh :



(Gambar 2.1.2)

Gambar 2.1.2 adalah suatu graph tidak sederhana, dimana e_4 dan e_5 merupakan edge ganda dan e_1 merupakan loop.

Definisi 2.1.3

Jika semua edge-edge dalam suatu graph tidak mempunyai arah, maka disebut graph tak berarah.

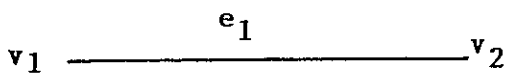
Contoh :

Graph pada gambar 2.1.1 merupakan graph tak berarah.

Definisi 2.1.4

Vertex v_i dan v_j disebut endvertex dari e_x, jika e_x menghubungkan vertex v_i dan v_j.

Contoh :



(Gambar 2.1.3)

Pada gambar 2.1.3, vertex v_1 dan vertex v_2 merupakan endvertex dari e_1.

Definisi 2.1.5

Suatu edge e_i dikatakan incident dengan vertex v_j, jika v_j endvertex dari e_i.

Contoh :

Pada gambar 2.1.1, edge e_1 incident dengan vertex v_2, demikian juga e_2 incident dengan vertex v_2.

Definisi 2.1.6

Derajat (degree) dari vertex v dinotasikan d(v) adalah banyaknya edge yang incident dengan vertex v.

Contoh :

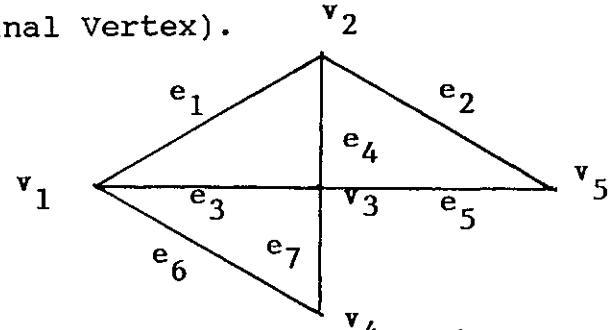
Pada gambar 2.1.1.

$d (v_1) = d (v_2) = d (v_3) = 2$

$d (v_4) = 3$

$d (v_5) = 1$

Diberikan suatu graph G seperti pada gambar 2.1.4, dimana vertex-vertexnya menunjukkan kota dan edge-edgenya menunjukkan jalan. Dari kota v_1 ke kota v_5 terdapat barisan edge-edge yang merupakan rute dari v_1 ke v_5. Suatu vertex berkorespondensi ke asal dinamakan vertex awal (Initial Vertex) dan vertex yang berkorespondensi ke tujuan dinamakan vertex akhir (Final Vertex).



(Gambar 2.1.4)

Contoh :

Pada gambar 2.1.4, dengan Initial vertex v_1 dan final vertex v_5 terdapat beberapa barisan edge misalnya (e_3,e_5), (e_3,e_4,e_2), $(e_1,e_4,e_3,e_6,e_7,e_5)$.

Yang harus dicatat bahwa setiap edge dalam suatu barisan yang dibicarakan (selain edge pertama dan edge terakhir) mempunyai satu endvertex yang bersamaan dengan edge sebelumnya dan endvertex lain dengan edge sesudahnya.

Contoh :

Dalam barisan (e_3,e_4,e_2) pada contoh di atas, vertex v_3 merupakan endvertex dari e_4 juga merupakan endvertex dari edge sebelumnya yaitu e_3.

Jika setiap edge yang demikian dalam suatu barisan muncul hanya sekali, barisan ini dinamakan suatu train edge. Sehingga didapat definisi train edge sebagai berikut :

Definisi 2.1.7

Train edge adalah barisan edge-edge dengan sifat sebagai berikut :

(1). Untuk edge e_u selain pertama dan terakhir edge-edge dalam barisan, satu endvertex dari e_u adalah endvertex dari edge sebelumnya dan endvertex lain dari e_u adalah endvertex dari edge sesudahnya.

(2). Satu endvertex dari edge pertama adalah endvertex dari edge sesudahnya dan endvertex lain dari edge pertama adalah initial vertex.

(3). Satu endvertex dari edge terakhir adalah endvertex dari edge sebelumnya dan endvertex lain dari edge terakhir adalah final vertex.

(4). Setiap edge muncul tepat satu kali.

Definisi 2.1.8

Suatu train edge disebut train edge open jika initial vertex dan final vertexnya berbeda, sebaliknya jika initial vertex dan final vertexnya sama disebut train edge closed.

Contoh :

Pada gambar 2.1.4, barisan $(e_3, e_4, e_2, e_5, e_7)$ adalah train edge open yang initial vertexnya v_1 dan final vertexnya v_4. Barisan (e_1, e_4, e_3) adalah train edge closed.

Jika train edge open mempunyai sifat bahwa derajat setiap vertex kecuali initial vertex dan final vertex adalah 2 dan derajat dari initial vertex dan final vertex adalah 1, maka himpunan semua edge dalam train edge disebut **path** antara initial vertex dan final vertex.

Definisi 2.1.9

Path antara vertex i dan j adalah himpunan semua edge dalam train edge yang memenuhi sifat sebagai berikut :

(1). Initial vertex dan final vertex adalah i dan j.

(2). Setiap vertex selain i dan j derajatnya 2 dan vertex i dan j derajatnya 1.

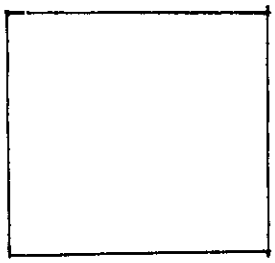
Contoh :

Himpunan semua edge dalam train edge (e_1, e_4, e_5) pada gambar 2.1.4 merupakan path antara vertex v_1 dan v_5.

Definisi 2.1.10

Graph terhubung (Connected Graph) adalah jika setiap pasang vertex-vertexnya dihubungkan oleh path. Dan sebaliknya disebut graph tak terhubung (Disconnected Graph).

Contoh :



(Gambar 2.1.5a)

Graph terhubung

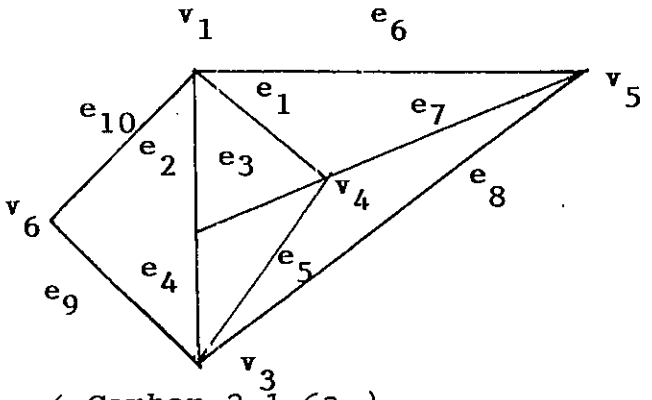
(Gambar 2.1.5b)

Graph tak terhubung

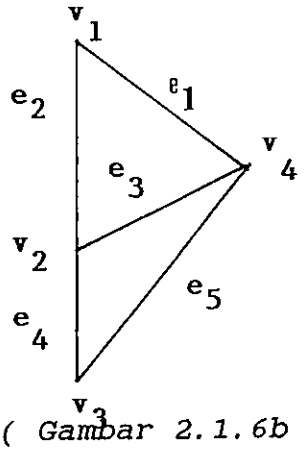
Definisi 2.1.11

Suatu graph g dikatakan subgraph dari graph G jika seluruh vertex dan edgenya berada dalam G.

Konsep dari subgraph sama dengan konsep subset pada teori himpunan. Sebuah subgraph bisa dikatakan sebagai graph dimuat (bagian dari) graph yang lain. Simbol teori himpunan $g \subset G$, digunakan dengan kalimat " g adalah sebuah subgraph dari G ".



(*Gambar 2.1.6a*)



(*Gambar 2.1.6b*)

Gambar 2.1.6b merupakan salah satu subgraph dari graph ada gambar 2.1.6a.

2.2 VERTEX - CUT

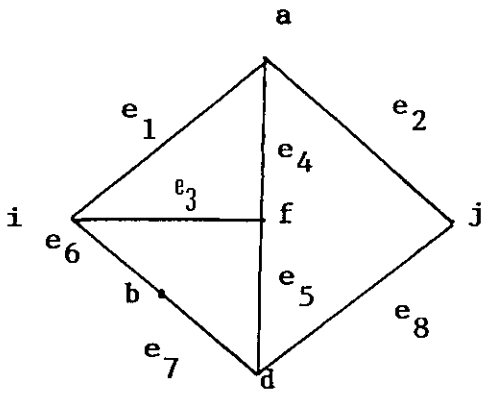
2.2.1 PENGERTIAN

Dalam jaringan VWC tak berarah untuk menentukan aliran maksimum harus dicari terlebih dahulu vertex-cutnya.
Dalam menentukan vertex-cut harus didasarkan pada definisi-definisi di bawah ini.

Definisi 2.2.1

Misalkan Ω suatu himpunan vertex dalam graph G. Maka penghapusan semua vertex dalam Ω berarti penghapusan semua vertex beserta semua edge yang incident pada vertex tersebut.

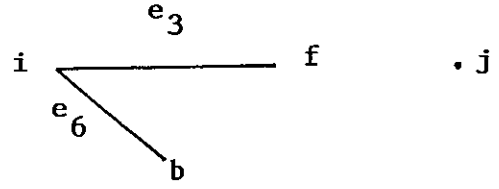
Contoh :



(*Gambar 2.2.1*)

Gambar 2.2.1 adalah graph tak berarah

Pada gambar 2.2.1, jika vertex a dan vertex d dihapus, maka graph hasil penghapusan tidak memuat edge-edge e_1, e_2, e_4, e_5, e_7, e_8 serta vertex a dan vertex d. Graph hasil ini dapat dilihat pada gambar 2.2.2 dibawah ini.



(Gambar 2.2.2)

Gambar 2.2.1 adalah graph hasil penghapusan himpunan vertex (a,d)

Definisi 2.2.2 (VERTEX-CUT)

Anggap terdapatlah sekurang-kurangnya 1 path dari vertex i ke vertex j dalam suatu graph G. Suatu vertex-cut yang memisahkan i dan j adalah suatu himpunan vertex yang minimal, sedemikian sehingga penghapusan semua vertex dalam himpunan tersebut akan merusakkan semua path dari i ke j.

Contoh :

Amati graph pada gambar 2.2.1.

Himpunan vertex (a,d,f) bukan suatu vertex-cut yang memisahkan i dan j, sebab penghapusan vertex a dan vertex d saja, sudah merusakkan semua path antara i dan j. Sebaliknya penghapusan vertex a atau vertex d saja (tidak keduanya secara bersamaan) tidak akan merusakkan semua path antara i dan j. Karenanya himpunan vertex (a,d) adalah suatu vertex-cut.

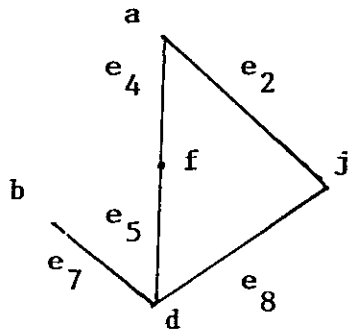
2.2.3 LANGKAH-LANGKAH MENENTUKAN VERTEX-CUT

Untuk menentukan semua vertex-cut dari suatu jaringan VWC tak berarah diperlukan langkah-langkah sebagai berikut :

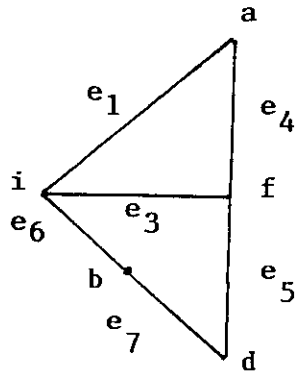
1. Ditentukan terlebih dahulu vertex awal dan vertex akhirnya Berdasarkan definisi tentang vertex-cut, penghapusan vertex awal atau vertex akhir beserta dengan edge-edge yang incident pada masing-masing vertex tersebut akan merusakkan semua path dari vertex awal ke vertex akhir dalam jaringan VWC tak berarah. Vertex awal atau vertex akhir tersebut merupakan himpunan vertex yang minimal. Berdasarkan pengertian ini vertex awal dan vertex akhir suatu jaringan VWC tak berarah merupakan vertex-cut.

   Contoh :

   Pada gambar 2.2.1, diketahui vertex awal adalah i dan vertex akhir adalah j. Kedua vertex tersebut merupakan suatu vertex-cut.



Gambar (2.2.3a)
Grap hasil penghapusan vertex i



Gambar (2.2.3b)
Graph hasil penghapusan vertex j

2. Untuk menentukan vertex-cut-vertex-cut lain terlebih dulu harus diketahui semua vertex yang ada dalam suatu jaringan VWC tak berarah.

Dimisalkan jumlah semua vertex dalam suatu graph G adalah n vertex, maka jumlah semua vertex selain vertex awal dan vertex akhir adalah (n-2) vertex, sebut sebagai **vertex sisa**. Vertex-cut - vertex-cut lain dapat dicari dengan cara mencari kemungkinan setiap vertex dalam vertex sisa menjadi suatu vertex-cut. Berarti vertex-cut tersebut merupakan himpunan yang berisi satu vertex.

Contoh :

Pada gambar 2.2.1 diketahui bahwa vertex awalnya adalah i dan vertex akhirnya adalah j .

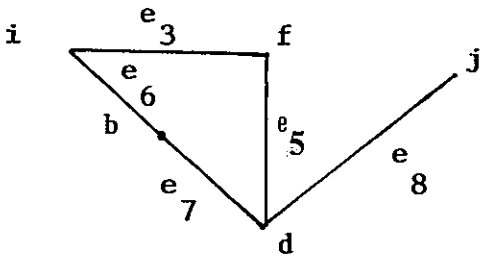
Jumlah semua vertex (n)=6, yaitu i,a,b,d, f,j.

Jumlah vertex sisa (n-2)=4, yaitu a,b,d,f.

Akan dicari kemungkinan setiap vertex dalam vertex sisa menjadi suatu vertex-cut.

a. **Himpunan vertex a**

bukan merupakan vertex-cut sebab dengan menghapus vertex a beserta semua edge yang incident terhadap vertex tersebut, tidak akan merusakkan semua path dari i ke j.

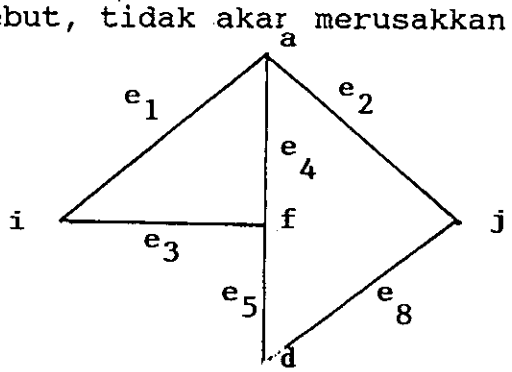


(Gambar 2.2.4a)

Gambar 2.2.4a adalah graph hasil penghapusan vertex a

b. **Himpunan vertex b**

bukan merupakan vertex-cut sebab dengan menghapus vertex b beserta semua edge yang incident terhadap vertex tersebut, tidak akan merusakkan semua path dari i ke j.

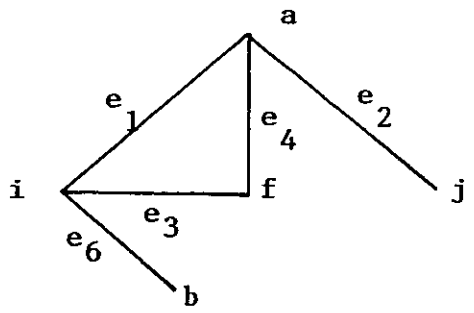


(Gambar 2.2.4b)

Gambar 2.2.4b adalah graph hasil penghapusan vertex b

c. **Himpunan vertex d**

bukan merupakan vertex-cut sebab dengan menghapus vertex d beserta semua edge yang incident terhadap vertex tersebut, tidak akan merusakkan semua path dari i ke j.

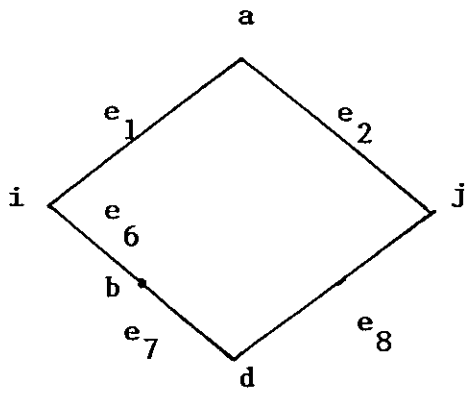


(Gambar 2.2.4c)

Gambar 2.2.4c adalah graph hasil penghapusan vertex d

d. **Himpunan vertex f**

bukan merupakan vertex-cut sebab dengan menghapus vertex f beserta semua edge yang incident terhadap vertex tersebut, tidak akan merusakkan semua path dari i ke j.



(Gambar 2.2.4d)

Gambar 2.2.4d adalah garaph hasil penghapusan vertex f

3. Mencari kombinasi vertex dari vertex-vertex sisa menjadi suatu himpunan-himpunan vertex, kemudian dicari himpunan vertex yang merupakan suatu vertex-cut.

Contoh :

Untuk menentukan suatu himpunan vertex merupakan suatu vertex-cut, terlebih dulu harus dicari semua kombinasi vertex yang mungkin dari vertex-vertex sisa tersebut.

A). Kombinasi 2 vertex dari 4 vertex sisa.

$$K_2^4 = \frac{4!}{2!\,(4-2)!} = \frac{4!}{2!\,2!} = 6$$

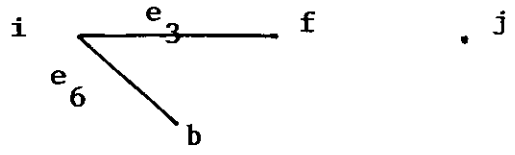
Terdapat 6 himpunan-himpunan vertex yang mungkin. Keenam himpunan vertex tersebut adalah :

1. (a,d)
2. (a,b)
3. (a,f)
4. (b,d)
5. (b,f)
6. (d,f)

Keenam himpunan vertex tersebut belum diketahui himpunan vertex mana saja yang merupakan vertex-cut. Untuk mengetahui apakah himpunan-himpunan vertex tersebut merupakan vertex-cut maka himpunan tersebut harus diselidiki satu persatu dengan berdasarkan definisi tentang vertex-cut.

1. **Himpunan vertex (a,d)**

merupakan vertex-cut, sebab dengan menghapus vertex a dan vertex d beserta semua edge yang incident terhadap vertex-vertex tersebut, akan merusakkan semua path dari i ke j serta merupakan himpunan vertex yang minimal.

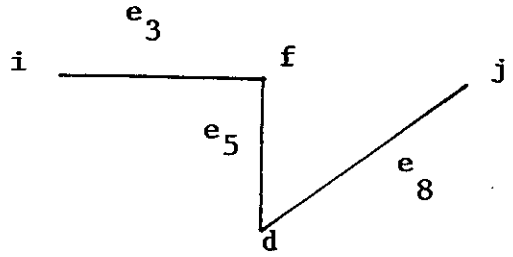


(Gambar 2.2.5a)

Gambar 2.2.5a adalah graph hasil penghapusan vertex (a,d)

2. **Himpunan vertex (a,b)**

bukan merupakan vertex-cut sebab dengan menghapus vertex a dan vertex b beserta semua edge yang incident terhadap vertex-vertex tersebut, tidak akan merusakkan semua path dari i ke j.



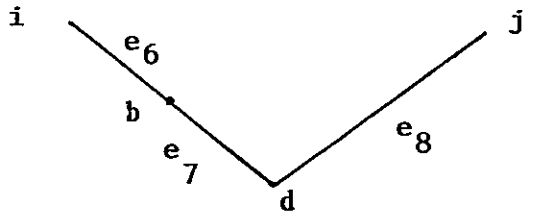
(Gambar 2.2.5b)

Gambar 2.2.5b adalah graph hasil penghapusan vertex (a,b)

3. **Himpunan vertex (a,f)**

bukan merupakan vertex-cut sebab dengan menghapus vertex a dan vertex f beserta semua edge yang

incident terhadap vertex-vertex tersebut, tidak akan merusakkan semua path dari i ke j.

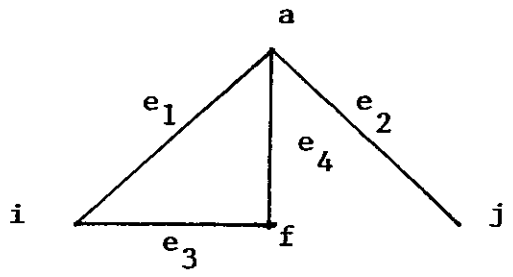


(Gambar 2.2.5c)

Gambar 2.2.5c adalah graph hasil penghapusan vertex (a,f)

4. Himpunan vertex (b,d)

bukan merupakan vertex-cut sebab dengan menghapus vertex b dan vertex d beserta semua edge yang incident terhadap vertex-vertex tersebut, tidak akan merusakkan semua path dari i ke j.

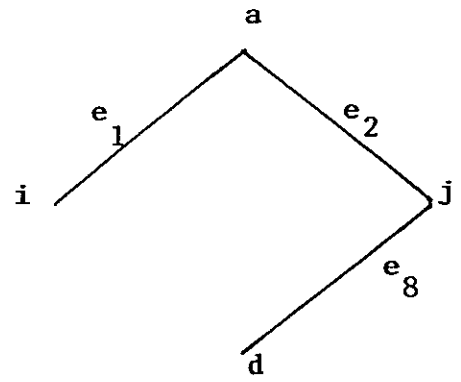


(Gambar 2.2.5d)

Gambar 2.2.5d adalah graph hasil penghapusan vertex (b,d)

5. Himpunan vertex (b,f)

bukan merupakan vertex-cut sebab dengan menghapus vertex b dan vertex f beserta semua edge yang incident terhadap vertex-vertex tersebut, tidak akan merusakkan semua path dari i ke j.

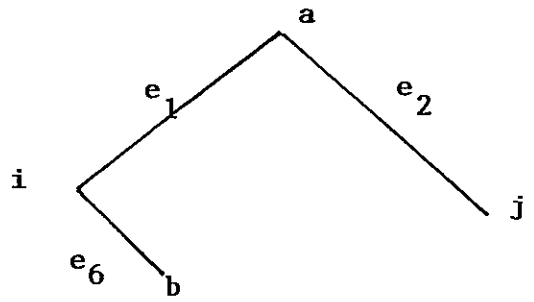


(Gambar 2.2.5e)

Gambar 2.2.5e adalah graph hasil penghapusan vertex (b,f)

6. Himpunan vertex (d,f)

bukan merupakan vertex-cut sebab dengan menghapus vertex d dan vertex f beserta semua edge yang incident terhadap vertex-vertex tersebut, tidak akan merusakkan semua path dari i ke j.



(Gambar 2.2.5f)

Gambar 2.2.5f adalah graph hasil penghapusan vertex (d,f)

Setelah menyelidiki semua himpunan vertex diatas satu persatu, ternyata hanya ada satu himpunan vertex yang merupakan vertex-cut, yaitu (a,d).

B) Kombinasi 3 vertex dari 4 vertex sisa

$$K^4_3 = \frac{4!}{3!\ (4-3)!} = \frac{4!}{3!\ 1!} = 4$$

Terdapat 4 himpunan-himpunan vertex yang mungkin. Keempat himpunan vertex tersebut adalah :

1. (a,b,d)
2. (a,b,f)
3. (b,d,f)
4. (a,d,f)

Keempat himpunan vertex tersebut belum diketahui himpunan vertex mana saja yang merupakan vertex-cut. Untuk mengetahui apakah himpunan-himpunan vertex tersebut merupakan vertex-cut maka himpunan tersebut harus diselidiki satu persatu dengan berdasarkan definisi tentang vertex-cut.

1. **Himpunan vertex (a,b,d)**

   bukan merupakan vertex-cut, karena meskipun penghapusan vertex a, b dan d beserta semua edge yang incident terhadap vertex-vertex tersebut akan merusakkan semua path dari i ke j. Namun himpunan vertex (a,b,d) bukan merupakan himpunan vertex yang minimal, sebab penghapusan himpunan vertex (a,d) saja sudah dapat merusakkan semua path dari i ke j.

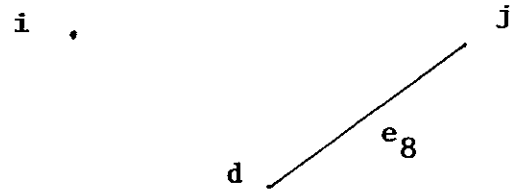
(Gambar 2.2.6a)

Gambar 2.2.6a adalah graph hasil penghapusan vertex (a,b,d)

2. **Himpunan vertex (a,b,f)**

   merupakan vertex-cut, sebab dengan menghapus vertex-vertex a, b dan f beserta semua edge yang

incident terhadap vertex-vertex tersebut, akan merusakkan semua path dari i ke j serta merupakan himpunan vertex yang minimal.

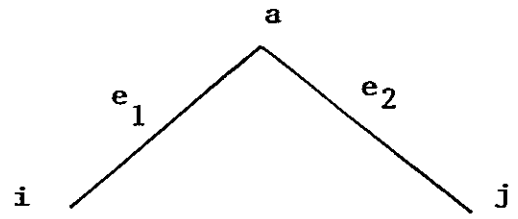


(Gambar 2.2.6b)

Gambar 2.2.6b adalah graph hasil penghapusan vertex (a,b,f)

3. **Himpunan vertex (b,d,f)**

bukan merupakan vertex-cut sebab dengan menghapus vertex-vertex b, d dan f beserta semua edge yang incident terhadap vertex-vertex tersebut, tidak akan merusakkan semua path dari i ke j.



(Gambar 2.2.6c)

Gambar 2.2.6c adalah graph hasil penghapusan vertex (b,d,f)

4. **Himpunan vertex (a,d,f)**

bukan merupakan vertex-cut, karena meskipun penghapusan vertex-vertex a, d dan f beserta semua edge yang incident terhadap vertex-vertex tersebut merusakkan semua path dari i ke j, namun himpunan vertex (a,d,f) bukan merupakan himpunan vertex yang minimal, sebab penghapusan himpunan vretex (a,d)

saja sudah merusakkan semua path dari i ke j.

(Gambar 2.2.6d)

Gambar 2.2.6d adalah graph hasil penghapusan vertex (a,d,f)

Setelah menyelidiki semua himpunan vertex diatas satu persatu, ternyata hanya ada satu himpunan vertex yang merupakan vertex-cut, yaitu (a,b,f).

C) Kombinasi 4 vertex dari 4 vertex sisa

$$K^4_4 = \frac{4!}{4!\ (4-4)!} = \frac{4!}{4!\ 0!} = 1$$

Terdapat 1 himpunan vertex yang mungkin yaitu (a,b,d,f). Himpunan vertex tersebut bukan merupakan suatu vertex-cut, karena meskipun penghapusan vertex a, b, d dan f beserta semua edge yang incident terhadap vertex-vertex tersebut akan merusakkan semua path dari i ke j. Namun himpunan vertex (a,b,d,f) bukan merupakan himpunan vertex yang minimal, sebab penghapusan himpunan vertex (a,d) atau (a,b,f) saja sudah dapat merusakkan semua path dari i ke j.

(Gambar 2.2.7)

Gambar 2.2.7 adalah graph hasil penghapusan vertex (a,b,d,f)

Jadi semua vertex-cut pada jaringan VWC tak berarah dalam gambar 2.2.1 adalah (i), (j), (a,b,f) dan (a,d).

Dengan demikian dapat ditarik kesimpulan cara menentukan vertex-cut yang mempunyai n (finite) vertex sebagai berikut :

1. Dari suatu jaringan VWC tak berarah dengan n (finite) vertex yang diberikan, ditentukan terlebih dahulu vertex awal dan vertex akhirnya. Kemudian masing-masing menjadi vertex-cut.
2. Menyelidiki setiap vertex dalam vertex sisa, apakah memenuhi definisi vertex-cut untuk menjadi suatu vertex-cut.
3. Mencari kombinasi dari vertex-vertex sisa menjadi himpunan-himpunan vertex. Dimana banyaknya himpunan vertex yang dapat dikombinasikan, yaitu :

$$K_{1}^{m} = \frac{m!}{1!\,(m-1)!}$$

dan $l=m-r$; $r = 0,1,2,\ldots,(n-1)$

dimana m : jumlah vertex sisa

l : jumlah vertex yang akan dikombinasikan

Kemudian menyelidiki himpunan-himpunan vertex tersebut apakah memenuhi definisi vertex-cut untuk menjadi suatu vertex-cut.

4. Dari langkah (1), (2) dan (3) dapat diketahui semua vertex-cut dari Jaringan VWC tak berarah.