BAB II

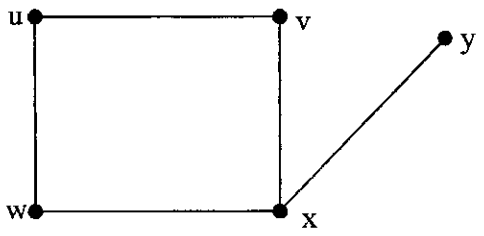
DASAR TEORI

2.1. Konsep Graph

Definisi 2.1 :

Suatu **graph** G merupakan himpunan tak kosong berhingga V(G) dari obyek-obyek yang disebut dengan titik-titik (vertex atau point atau node), dan suatu himpunan (mungkin kosong) E(G) yang merupakan himpunan pasangan elemen-elemen dari V(G) yang disebut dengan garis (edge atau line). Himpunan V(G) disebut dengan himpunan titik dari graph G dan E(G) disebut dengan himpunan garis dari graph G.

Contoh 2.1 :



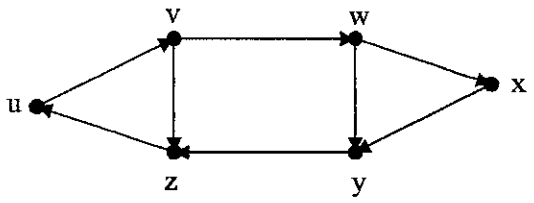
Gambar 2.1 Graph

Pada gambar 2.1, graph G(V,E) mempunyai himpunan vertex V(G) = {u,v,w,x,y} dan mempunyai himpunan edge E(G) = {(u,v),(u,w),(v,x),(w,x),(x,y)}.

Definisi 2.2 :

Suatu **directed graph** (graph berarah) G didefinisikan secara abstrak sebagai pasangan berurutan (V,E) dengan V adalah suatu set (himpunan) dan E adalah binary relation (relasi biner) pada V. Suatu directed graph dapat digambarkan secara geometrik sebagai suatu himpunan titik-titik V yang nyata, dengan suatu himpunan anak panah E antara pasangan titik-titik.

Contoh 2.2 :



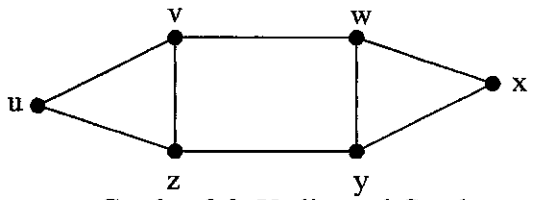
Gambar 2.2. Directed Graph

Pada gambar 2.2, adalah sebuah graph berarah G(V,E) dengan himpunan vertexnya adalah V(G) = {u,v,w,x,y,z} dan mempunyai himpunan edge berarahnya E(G) = {(u,v),(v,w),(v,z),(w,x),(w,y),(x,y),(y,z),(z,u)}.

Definisi 2.3 :

Suatu **undirected graph** (graph tak berarah) G secara abstrak didefinisikan sebagai pasangan berurutan (V,E) dimana V adalah suatu himpunan dan E adalah himpunan atau multi himpunan dari dua elemen V. Sebuah underected graph dapat direpresentasikan secara geometris sebagai himpunan titik-titik yang nyata V dengan sebuah himpunan garis E antara titik-titik anggota V.

Contoh 2.3 :



Gambar 2.3. Undirected Graph

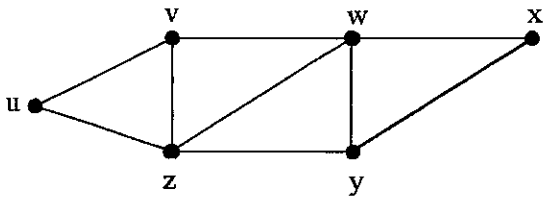
Pada gambar 2.3, adalah sebuah graph tak berarah G(V,E) dengan himpunan vertexnya adalah V(G) = {u,v,w,x,y,z} dan mempunyai himpunan edge E(G) = { (u,v),(u,z),(v,z),(v,w),(w,y),(y,z),(w,x),(x,y)}.

Untuk selanjutnya, akan digunakan istilah "graph" saja untuk menyatakan directed graph atau undirected graph atau untuk keduanya

Definisi 2.4 :

Dua buah vertex dikatakan **adjacent** jika kedua vertex dihubungkan oleh sebuah edge. Selain itu, sehubungan dengan edge (u,v), vertex u dikatakan adjacent ke vertex v dan vertex v adjacent dari vertex u. Sebuah vertex dikatakan **vertex terisolasi** jika tidak ada satupun edge yang incident dengan vertex tersebut.

Contoh 2.4:



Gambar 2.4 Vertex Adjacent

Pada gambar 2.4 vertex z adjacent dengan vertex u,v,w dan y tetapi tidak adjacent dengan vertex x.

Definisi 2.5. :

Jika e = (u,v) adalah sebuah edge dari graph G, maka dikatakan bahwa e dan u (e dan v) adalah **incident** satu sama lain, sedangkan jika dua buah edge berbeda, e dan f, yang incident dengan titik u atau v, maka e dan f dikatakan garis yang adjacent.

Pada gambar 2.4 edge (u,v) incident dengan vertex u dan vertex v, edge (u,z) incident dengan vertex u dan vertex z, tetapi edge (w,z) tidak incident dengan vertex x

Definisi 2.6 :

Suatu **walk** dalam graph G merupakan barisan vertex-vertex dan edge-edge :

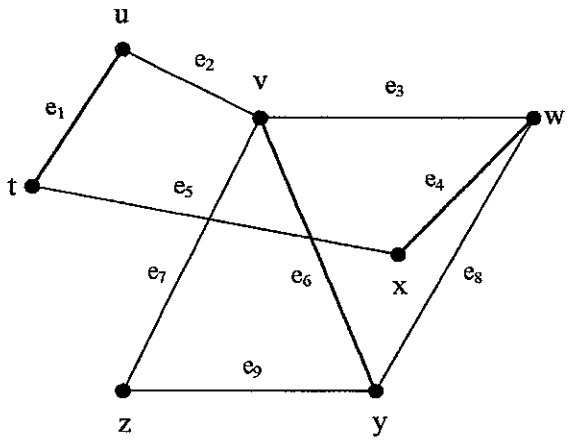
W : $v_0, e_1, v_1, e_2, v_2, ..., v_{n-1}, e_n, v_n$ $(n \geq 0)$

dimulai dan diakhiri dengan titik, sedemikian sehingga $e_i = v_{i-1}v_i$ untuk $i = 1, 2, 3, ..., n$.

Definisi 2.7:

W : $v_0, e_1, v_1, e_2, v_2, ..., v_{n-1}, e_n, v_n$ $(n \geq 0)$ dikatakan sebuah walk dengan panjang n apabila W mempunyai n edge. Sebuah walk dengan panjang nol disebut dengan **Trivial Walk**.

Contoh 2.5 :



Gambar 2.5 Walk

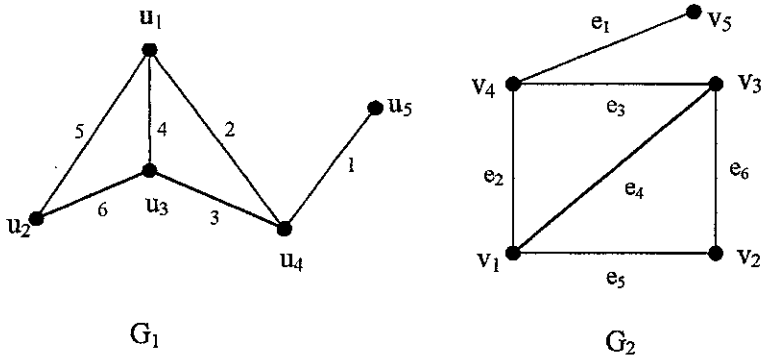
Pada gambar 2.5, yang merupakan salah satu walk dengan panjang 5 adalah: u, e_2, v, e_3, w, e_8, y, e_6, v, e_7, z dan bisa disajikan sebagai u,v,w,y,v,z atau u-v-w-y-v-z.

Definisi 2.8 :

Dua buah graph G_1 dan G_2 dikatakan **isomorphic** jika

a. Jumlah vertex dalam G_1 dan G_2 sama banyak
b. Ada korespondensi satu-satu antara vertex dan antara edge sedemikian sehingga adjacency-nya dipertahankan atau tetap terjaga.

Contoh 2.6 :



Gambar 2.6 Graph Ishomorpic

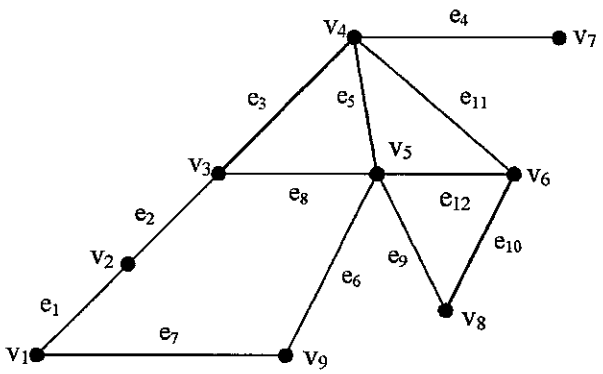
Dari gambar 2.6 terlihat bahwa kedua graph G_1 dan G_2 mempunyai jumlah vertex yang sama dan terdapat korespondensi satu-satu antara vertex-vertex dan antara edge dalam kedua graph tersebut, yaitu: vertex u_1,u_2,u_3,u_4,u_5 dalam G_1 berkorespondensi dengan v_1,v_2,v_3,v_4,v_5 dalam G_2. dan edge 1,2,3,4,5 dalam G_1 berkorespondensi dengan edge e_1,e_2,e_3,e_4,e_5 dalam G_2.

Definisi 2.9 :

Dalam directed graph, **path** adalah barisan dari edge-edge $(e_1, e_2, e_3, \ldots, e_k)$ sedemikian sehingga terminal vertex dari e_k akan bertepatan dengan initial vertex dari e_{k+1}, dengan $k=1,2,3\ldots.n$.

Suatu path dikatakan **simple** jika tidak ada edge yang sama muncul dua kali dan dikatakan **elementary** jika tidak menjumpai vertex yang sama dua kali. Suatu path dikatakan **re-entrant** jika ada vertex yang muncul lebih dari satu kali. Jadi dapat dikatakan bahwa elementary path adalah path yang tidak re-entrant.

Contoh 2.7 :



Gambar 2.7 Graph dengan Pathnya

Dari gambar 2.7, barisan dari edge-edge (e_1,e_2,e_3,e_4) adalah sebuah path, jika dinyatakan sebagai deretan vertex yang adjacent: (v_1,v_2,v_3,v_4,v_7). Barisan (e_1,e_2,e_3,e_5,e_8,e_3,e_4) adalah path, tetapi bukan simple path karena edge e_3 muncul dua kali, juga bukan elementary path karena vertex v_3 dan vertex v_4 muncul dua kali. Barisan (e_1,e_2,e_3,e_5,e_9,e_{10},e_{11},e_4) adalah simple path dan path re-entrant, tetapi bukan elementary karena v_4 muncul dua kali.

Definisi 2.10:

Suatu **circuit** adalah suatu path (e_1,e_2,e_3, …, e_k) dengan k=1,2,3…n, dimana terminal vertex e_k sama dengan initial vertex e_1. Circuit dikatakan simple jika tidak ada edge yang sama muncul dua kali. Circuit dikatakan elementary jika tidak menjumpai vertex yang sama dua kali.

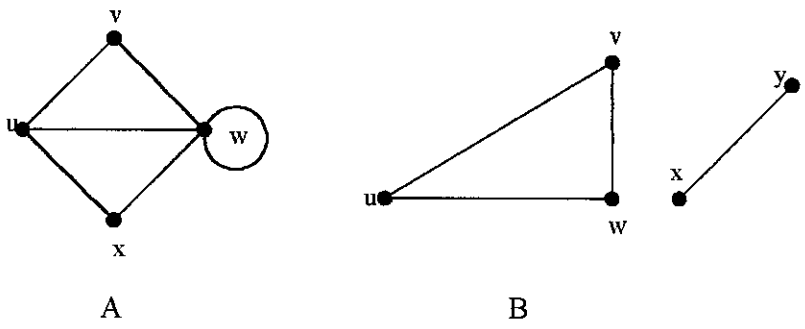
Dari gambar 2.7, (e_1,e_2,e_3,e_5,e_9,e_{10},e_{12},e_6,e_7) adalah simple circuit tapi bukan elementary circuit karena vertex v_5 dijumpai dua kali.

(e_1,e_2,e_3,e_5,e_6,e_7) adalah simple dan elementary circuit, dinyatakan dengan barisan vertex-vertex : (v_1,v_2,v_3,v_4,v_5,v_9,v_1).

Definisi 2.11:

Suatu undirected graph dikatakan **connected** jika setiap pasang vertex ada suatu path yang menghubungkan, dan dikatakan **disconnected** jika sebaliknya. Suatu directed graph dikatakan connected jika undirected graph yang diperoleh dari directed graph tersebut dengan mengabaikan arah edge-edgenya adalah connected dan dikatakan disconnected jika sebaliknya.

Contoh 2.8 :

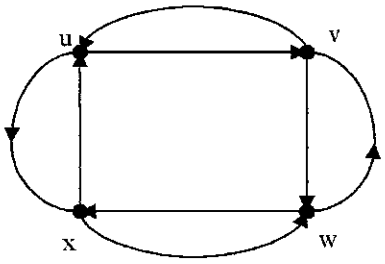


Gambar 2.8 A.Connected dan B.Disconnected Graph

Definisi 2.12:

Suatu directed graph dikatakan **strongly connected** jika untuk setiap dua vertex u dan v dalam graph ada sebuah path dari u ke v juga ada sebuah path dari v ke u.

Contoh 2.9 :

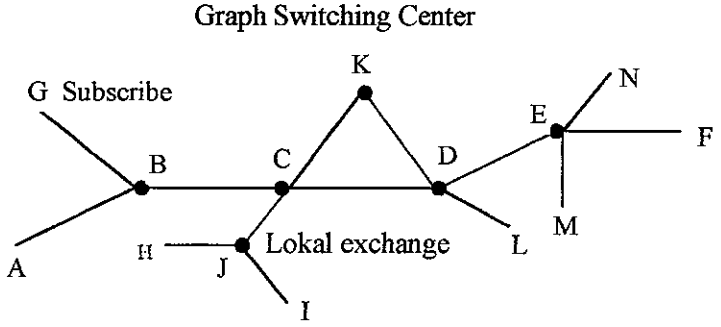


Gambar 2.9 Strongly Connected

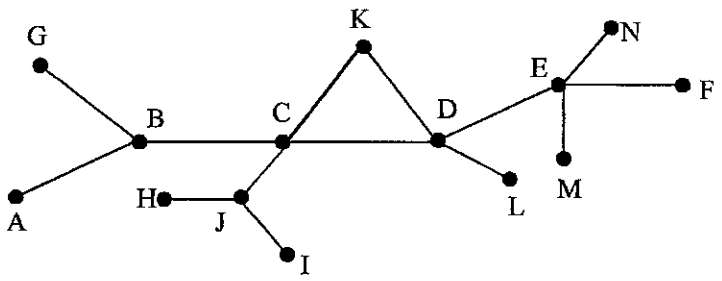
Untuk selanjutnya, penulisan vertex suatu graph akan disajikan dengan menggunakan angka, sehingga untuk penulisan walk, path dan path elementernya juga akan mengikutinya.

2.2. Representasi Graph

Jaringan telekomunikasi merupakan hubungan antar pemakai (*user*) menggunakan link komunikasi yang terseleksi sehingga dapat membentuk suatu rangkaian yang terhubung.



Gambar 2.10 Rute Sambungan Telepone



Gambar 2.11 Representasi Rute Sambungan Telepon

Gambar 2.10 di atas merupakan sebuah diagram sederhana yang memperlihatkan rute dari sebuah panggilan telepon dari A ke F melalui B, C, D dan E. Tetapi diagram tersebut juga menunjukkan sebuah sambungan untuk transmisi, switching center dan tujuan akhir (*terminal station*). Diagram tersebut dapat disederhanakan seperti pada gambar 2.11, menjadi sebuah graph yang vertex-vertexnya menggambarkan terminal station ataupun switching center dan edge-edgenya menggambarkan hubungan komunikasi.

Diagram seperti gambar 2.10 biasanya digunakan untuk menggambarkan network secara umum. Sebenarnya dalam setiap switching center yang digambarkan dalam sebuah titik dalam peta besar, ada network internal yang menghubungkan (*internal connecting network*) yang tidak kalah rumitnya. Sebuah pembicaraan telepon melibatkan banyak sekali switch, paling tidak ada beberapa puluh ribu contact yang dirangkaikan dalam beberapa ratus blok atau beberapa ratus group. Keterhubungan dari blok tersebut dikenal dengan trunking.

2.3. Matriks Adjacency

Adanya isomorphisme antara physical network dengan graphnya sudah cukup jelas, hubungan-hubungan adjacency tetap terjaga. Hubungan tersebut dapat digambarkan dalam bentuk matriks, misalnya dengan matriks adjacency A(G) yang diperoleh dari graph gambar 2.11 berikut ini :

	A	B	C	D	E	F	G	H	I	J	K	L	M	N
A	0	1	0	0	0	0	0	0	0	0	0	0	0	0
B	1	0	1	0	0	0	1	0	0	0	0	0	0	0
C	0	1	0	1	0	0	0	0	0	1	1	0	0	0
D	0	0	1	0	1	0	0	0	0	0	1	1	0	0
E	0	0	0	1	0	1	0	0	0	0	0	0	1	1
F	0	0	0	0	1	0	0	0	0	0	0	0	0	0
G	0	1	0	0	0	0	0	0	0	0	0	0	0	0
H	0	0	0	0	0	0	0	0	0	1	0	0	0	0
I	0	0	0	0	0	0	0	0	0	1	0	0	0	0
J	0	0	1	0	0	0	0	1	1	0	0	0	0	0
K	0	0	1	1	0	0	0	0	0	0	0	0	0	0
L	0	0	0	1	0	0	0	0	0	0	0	0	0	0
M	0	0	0	0	1	0	0	0	0	0	0	0	0	0
N	0	0	0	0	1	0	0	0	0	0	0	0	0	0

Gambar 2.12 Matriks Adjacency

Matriks adjacency dari graph G: A(G) atau disingkat A dimaksud sebagai suatu matriks yang elemen pada baris ke i, kolom ke j-nya, adalah:

- 1 jika vertex pada baris ke-i dengan vertex pada kolom ke-j terhubung (*adjacent*), dan

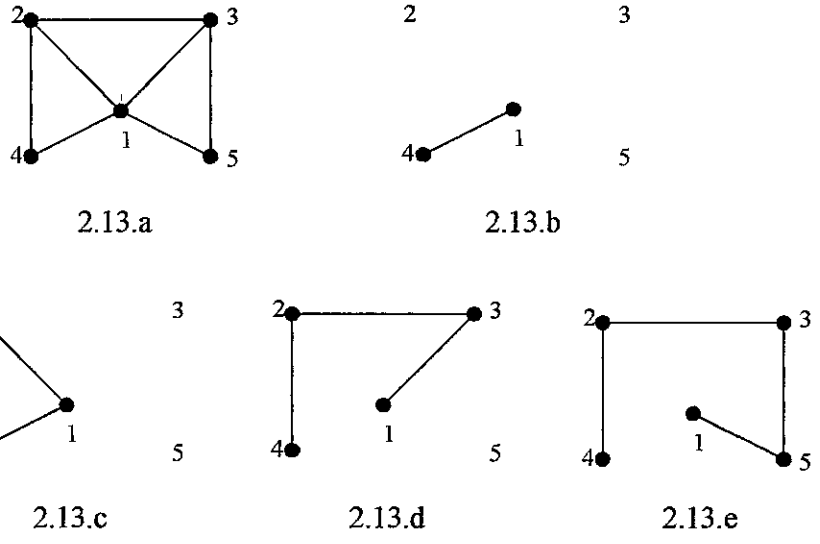
- 0 jika vertex pada baris ke-i, dengan vertex pada kolom ke-j tidak terhubung (tidak *adjacent*).

Jadi dapat terlihat jelas bahwa matriks adjacency merupakan matriks simetris, elemen pada diagonal utamanya sama dengan 0, elemennya berupa bilangan 1 dan 0.

2.4. Path Alternatif

Jaringan telekomunikasi biasanya menawarkan path-path alternatif antara pasangan terminal (tujuan) yang diutamakan. Semakin banyak path-path alternatif yang tersedia, semakin rendah kemungkinan bahwa semuanya akan terlalu sibuk dan hubungan-hubungan telepon yang baru akan terblok. Dalam jaringan sederhana banyaknya path akan sangat jelas.

Contoh 2.10 :



Gambar 2.13 Jaringan Sederhana

Jaringan sederhana di atas, dapat dibentuk sebuah matriks A.

$$A = \begin{bmatrix} 0 & 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 \end{bmatrix}$$

Dari matriks A tersebut bisa dibuat matriks A^2, A^3 ,A^4 sebagai berikut:

$$A^2 = \begin{bmatrix} 4 & 2 & 2 & 1 & 1 \\ 2 & 3 & 1 & 1 & 2 \\ 2 & 1 & 3 & 2 & 1 \\ 1 & 1 & 2 & 2 & 1 \\ 1 & 2 & 1 & 1 & 2 \end{bmatrix}$$

$$A^3 = \begin{bmatrix} 6 & 7 & 7 & 6 & 6 \\ 7 & 4 & 7 & 5 & 3 \\ 7 & 7 & 4 & 3 & 5 \\ 6 & 5 & 3 & 2 & 3 \\ 6 & 3 & 5 & 3 & 2 \end{bmatrix}$$

$$A^4 = \begin{bmatrix} 26 & 19 & 19 & 13 & 13 \\ 19 & 19 & 14 & 11 & 14 \\ 19 & 14 & 19 & 14 & 11 \\ 13 & 11 & 14 & 11 & 9 \\ 13 & 14 & 11 & 9 & 11 \end{bmatrix}$$

Gambar 2.14 Matriks A^1, A^2, A^3, A^4

Misalnya graph pada gambar 2.13 mempunyai 4 buah path antara vertex u dan x yaitu:

1). 1-4 (1 walk)

2). 1-2-4 (2 walk)

3). 1-3-2-4 (3 walk)

4). 1-5-3-2-4 (4 walk)

Tetapi biasanya diinginkan metode yang sistematis untuk menentukan jumlah path ini. Matriks adjacency menawarkan suatu pendekatan jumlah k-walk (yaitu walk-walk dengan k edge) dari vertex i ke vertex j diberikan oleh elemen ke-ij di dalam matriks A^k, karena kalau elemen ini ditulis sebagai a^k_{ij}, maka identitas $A^k = A^{k-1}.A$, menunjukkan bahwa :

$$a^k_{i,j} = \sum_r a^{k-1}_{ir}.a^1_{rj}$$

Memperhatikan persamaan tersebut dalam graph, k-walk dari i sampai j terdiri dari hubungan-hubungan tandem dari (k-1) walk dari i sampai r dengan 1 walk dari r ke j.

Contoh aplikasi persamaan tersebut:

Contoh 1. $a^4_{1,4} = 13$

Disini akan dicari banyaknya path dengan 4 walk dari vertex asal 1 dan vertex tujuan 4. Dengan menggunakan persamaan di atas bisa diketahui bahwa : $a^4_{1,4} = \sum_r a^3_{1r}.a^1_{r4}$

Disini : a^3_{1r} adalah elemen-elemen baris pertama pada aA^3

a^1_{r4} adalah elemen-elemen kolom ke empat pada A^1

$$a_{14}^{4} = \sum (6\,7\,7\,6) \begin{pmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

$= 6 + 7$

$= 13$

Jadi ada 13 path yang memiliki 4 walk yaitu:

1) 1-2-1-2-4

2) 1-2-3-1-4

3) 1-2-3-2-4

4) 1-2-4-1-4

5) 1-2-4-2-4

6) 1-3-1-2-4

7) 1-3-2-1-4

8) 1-3-5-1-4

9) 1-4-1-2-4

10) 1-4-2-1-4

11) 1-5-1-2-4

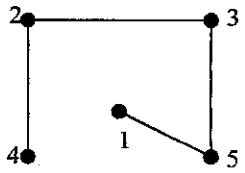
12) 1-5-3-1-4

13) 1-5-3-2-4

Dari ketigabelas path tersebut ada yang merupakan path-path re-entrant, misal path ke-12 yang menghubungkan vertex-vertex 1-5-3-1-4 Dan ada path yang tidak re-entrant, yaitu path ke-13 yang menghubungkan vertex-vertex 1-5-3-2-4.

Pada kenyataannya, dalam sebuah network suatu path dengan tidak satupun bagian yang sama akan lebih menguntungkan dibandingkan dengan suatu path yang mempunyai bagian yang sama, karena path yang mempunyai bagian yang sama dapat didisfungsikan hanya satu kesalahan saja.

Dari ketigabelas path tersebut dapat diseleksi path yang paling menguntungkan, yaitu path yang tidak re-entrant, yaitu path yang melalui vertex-vertex 1-5-3-2-4.



Gambar 2.15 Path yang tidak re-entrant

Contoh 2. $a^4_{4,5} = 9$

Path-pathnya:

1) 4-1-2-1-5
2) 4-1-2-3-5
3) 4-1-3-1-5
4) 4-1-4-1-5
5) 4-1-5-1-5
6) 4-1-5-3-5
7) 4-2-1-3-5
8) 4-2-3-1-5
9) 4-2-4-1-5

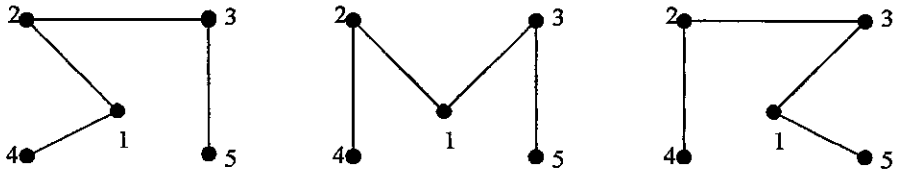
Path yang tidak re-entrant:

2) 4-1-2-3-5

7) 4-2-1-3-5

8) 4-2-3-1-5

Gambar ketiga path tersebut dalam graphnya adalah sebagai berikut:



Path ke-2 Path ke-7 Path ke-8

Gambar 2.16 Path-path yang tidak re-entrant