2. Terminologi Graph

Oleh : Ade Nurhopipah

Pokok Bahasan :

1. Graph dan Subgraph
2. Derajat Titik
3. Path dan Cycle
4. Graph Regular dan Graph Bipartit

Sumber :

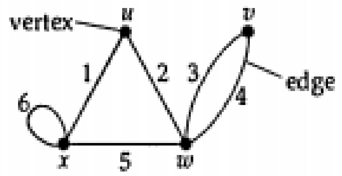
Aldous, Joan M. ,Wilson, Robin J. 2004. *Graph and Applications*. Springer: UK.

Graph dan Subgraph

Pada bab ini, kita akan mempelajari definisi dari berbagai istilah dalam teori graph.

Definisi 2.1
Sebuah graph terdiri dari sebuah himpunan elemen yang disebut *vertex*, dan sebuah himpunan elemen yang disebut *edge*. Setiap edge menghubungkan dua vertex.

Perhatikan graph yang ditunjukan pada Gambar 2.1. Graph tersebut memiliki empat vertex {*u,v,w,x*} dan enam edge {*1,2,3,4,5,6*}. Edge *1* menghubungkan vertex *u* dan *x*, edge *2* menghubungkan vertex *u* dan *w*, edge *3* dan *4* menghubungkan vertex *v* dan *w*, edge *5* menghubungkan vertex *w* dan *x*, sedangkan edge *6* menghubungkan *x* dengan dirinya sendiri. Kita sering menyebut suatu edge dengan dua vertex yang ia hubungkan. Misalnya edge *1* dinotasikan sebagai *ux* atau *xu*.



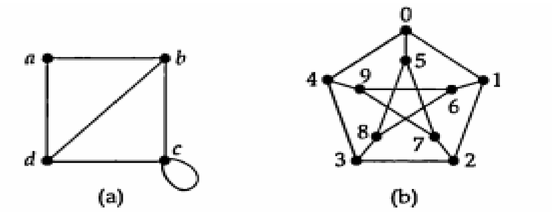
Gambar 2.1 Contoh graph dengan empat vertex dan enam edge

Definisi 2.2
Pada sebuah graph, dua atau lebih edge yang menghubungkan pasangan vertex yang sama disebut *multiple edge*.
Sebuah edge yang menghubungkan suatu vertex dengan dirinya sendiri disebut *loop*.
Suatu graph yang tidak memiliki multiple edge atau loop disebut graph sederhana (*simple graph*)

Graph pada Gambar 2.1 memiliki multiple edge yang menghubungkan *v* dan *w*, juga terdapat loop pada titik *x*, sehingga graph tersebut tidak termasuk graph yang sederhana.

Latihan 2.1

1. Tuliskan vertex dan edge pada graph berikut. Apakah graph tersebut termasuk graph sederhana?



(a) (b)

2. Gambar graph yang memiliki vertex dan edge berikut. Apakah graph tersebut termasuk graph sederhana?
   a. Vertex = {*u,v,w,x*}, edge ={*uv,vw,vx,wx*}
   b. Vertex = {1,2,3,4,5,6,7,8}, edge ={12,22,23,34,35,67,68,78}

Ketetanggaan dan Insidensi

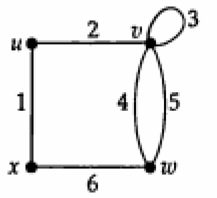
Definisi 2.3

Vertex *v* dan *w* pada suatu graph disebut bertetangga jika mereka dihubungkan dengan sebuah edge *e*. Vertec *v* dan *w* insiden dengan edge *e*, dan edge *e* insiden dengan vertex *v* dan *w*.

Latihan 2.2

Mana dari pernyataan ini yang sesuai dengan graph berikut.

a. Vertec *v* dan *w* bertetangga
b. Vertec *v* dan *x* bertetangga
c. Vertex *u* insiden dengan edge *2*
d. Edge *5* insiden dengan vertex *x*



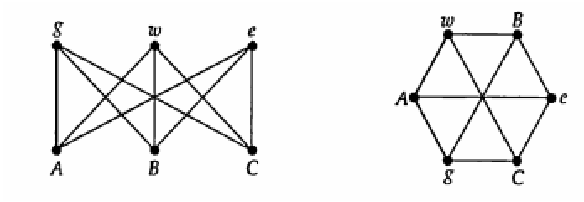
Isomorpfisma

Sesuai dengan definisi graph, sebuah graph ditentukan secara keseluruhan dengan mengetahui vertex dan edge pada graph tersebut. Dua graph merupakan graph yang sama jika mereka memiliki vertex dan edge yang sama. Sebagai contoh pada masalah utilitas, kita memiliki tiga rumah *A,B,C.* dan tiga utilitas gas (*g*), water(*w*) dan listik/electricity (*e*). Graph yang kita buat ditunjukan secara spesifik sebagai himpunan berikut.

Vertex = {*A,B,C, g,w,e*}

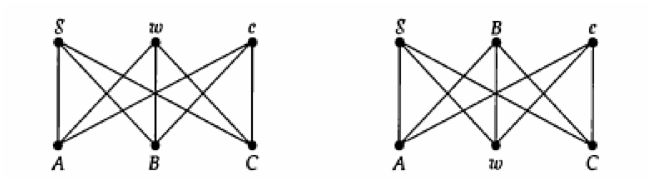
Edge { *Ag, Aw, Ae, Bg, Bw, Be, Cg, Cw, Ce*}

Kita dapat menggambar graph tersebut dengan beberapa cara, misalnya ditunjukan pada Gambar 2.2.



Gambar 2.2 Contoh dua graph yang isomorfik

Pada Gambar 2.2, kedua diagram tampak berbeda, namun sebenarnya memiliki vertex dan edge yang sama dan merepresentasikan situasi yang sama. Pada kedua graph tersebut, masing-masing rumah dihubungkan dengan ketiga utilitas, dan di antara rumah tidak saling dihubungkan demikian juga di antara utilitas. Graph yang direpresentasikan oleh diagram yang berbeda namun memiliki vertex dan edge yang sama ini disebut graph isomorfik. Artinya bahwa dua graph tersebut pada dasarnya memiliki struktur yang sama. Tinjau Gambar 2.3, di mana kedua diagram terlihat sama, namun sebenarnya merepresentasikan graph yang berbeda.

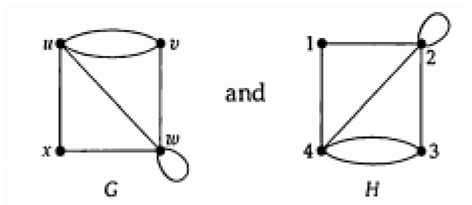


Gambar 2.3 Contoh dua graph yang tidak isomorfik

Definisi 2.4
Dua graph G dan H adalah isomorfik jika H dapat diperoleh dengan memberikan label pada vertex di G, jika terdapat korespondensi satu-satu antara vertex di G dengan vertex di H, sehingga jumlah egde yang menghubungkan setiap pasangan vertex di G sama dengan jumlah egde yang menghubungkan pasangan vertex yang berkorespondensi di H. Korespondensi satu-satu ini disebut isomorfisma.

Tinjau graph G dan H pada Gambar 2.4. Kedua graph ini tidak sama, namun keduanya isomorfik. Kita dapat mengganti label vertex graph G untuk mendapatkan graph H memakai korespondensi satu-satu sebagai berikut.

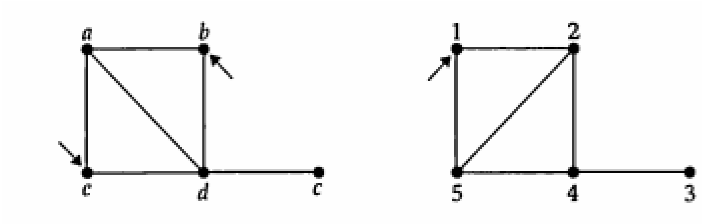
G↔H *u*↔*4* *v*↔*3* *w*↔*2* *x*↔*1*



Gambar 2.4 Dua graph yang berbeda namun isomorfik

Catat bahwa edge pada G berkoresponden dengan edge di H, misalnya dua edge yang menghubungkan *u* dan *v* berkoresponden dengan dua edge yang menghubungkan vertex *4* dan *3* di H, edge *uw* di G berkoresponden dengan edge *42* di H, dan seterusnya.

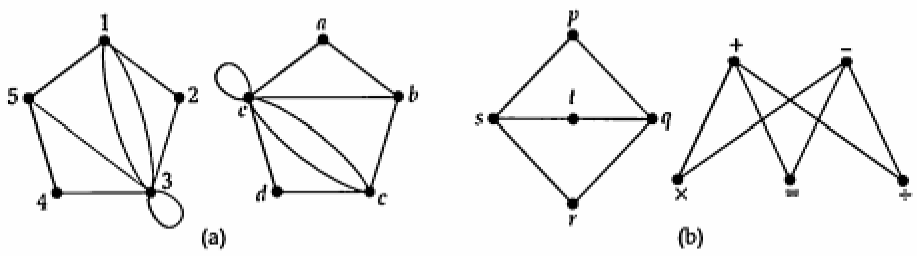
Untuk memeriksa apakah dua graph merupakan graph yang sama, kita harus melihat apakah semua label vertex berkoresponden. Namun untuk mengecek apakah dua graph adalah isomorfik kita harus memeriksa apakah kita dapat melabeli vertex pada suatu graph untuk mendapatkan graph yang lain. Untuk melakukan ini, pertama kita harus mengecek apakah kedua graph memiliki jumlah vertex dan edge yang sama, kemudian kita melihat sifat khusus pada kedua graph, seperti loop, multiple edge, atau jumlah edge yang bertemu dengan vertex. Misalnya graph pada Gambar 2.5, keduanya memiliki lima vertex dan enam edge, namun tidak isomorfik. Graph pertama memiliki dua vertex yang insiden dengan dua edge, sedangkan yang kedua hanya memiliki satu vertex yang insiden dengan dua edge.



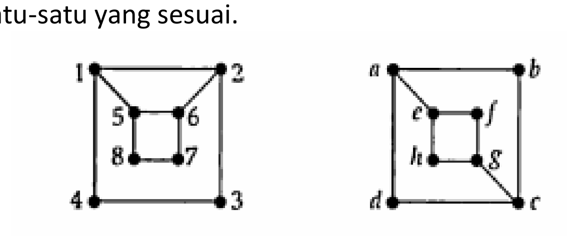
Gambar 2.5 Menentukan apakah dua graph isomorfik

Latihan 2.3

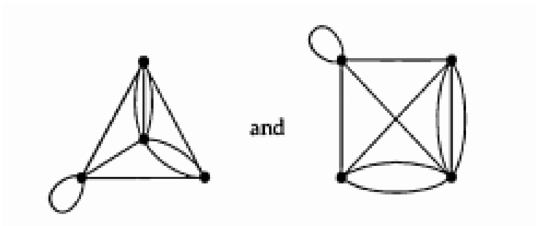
1. Dengan melakukan pelabelan kembali terhadap vertex, tunjukan bahwa pasangan graph berikut adalah isomorfik.



2. Apakah kedua graph berikut isomorfik? Jika iya, temukan korespondensi satu-satu yang sesuai antara vertex pada graph pertama dan graph kedua. Jika tidak, jelaskan mengapa tidak ada korespondesi satu-satu yang sesuai.

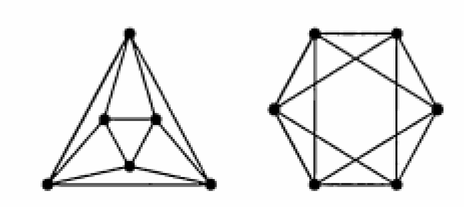


Kadang-kadang, kita tidak diharuskan atau tidak perlu untuk melabeli suatu graph. Pada kasus ini kita menghilangkan label tersebut dan mendapatkan suatu graph yang disebut *graph tidak berlabel*. Dua graph tidak berlabel seperti ditunjukan pada Gambar 2.5 dikatakan isomorfik, jika kedua graph tersebut dapat dilabeli pada vertex-nya sehingga keduanya menjadi graph yang sama.



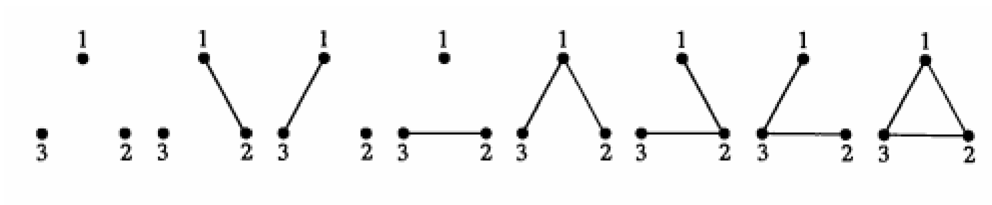
Gambar 2.5 Dua graph tidak berlabel yang isomorfik

Latihan 2.4
Dengan memberi label yang sesuai, tunjukan bahwa dua graph tidak berlebel ini adalah isomorfik.

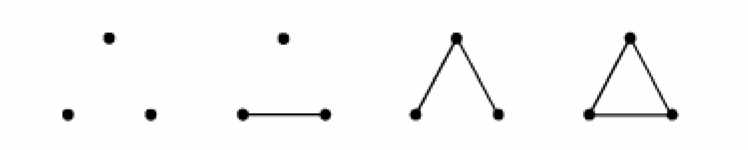


Menghitung Graph

Berapakah jumlah graph yang dapat kita buat jika kita memiliki sejumlah vertex tertentu? Jika kita memiliki tiga buah vertex, maka kita dapat membuat delapan buah graph berlabel dan empat graph tidak berlabel seperti yang ditunjukan pada Gambar 2.6.



(a) Graph berlabel



(a) Graph tidak berlabel

Gambar 2.6 Graph yang dapat dibuat dengan 3 vertex

Tabel 2.1 mendaftar jumlah graph sederhana berlabel dan tidak berlabel yang dapat dihasilkan berdasarkan jumah vertex yang dimiliki. Secara umum, menghitung jumlah graph berlabel lebih mudah daripada menghitung jumlah graph yang tidak berlabel.

Tabel 2.1 Jumlah graph yang dapat dibuat dengan *n* vertex

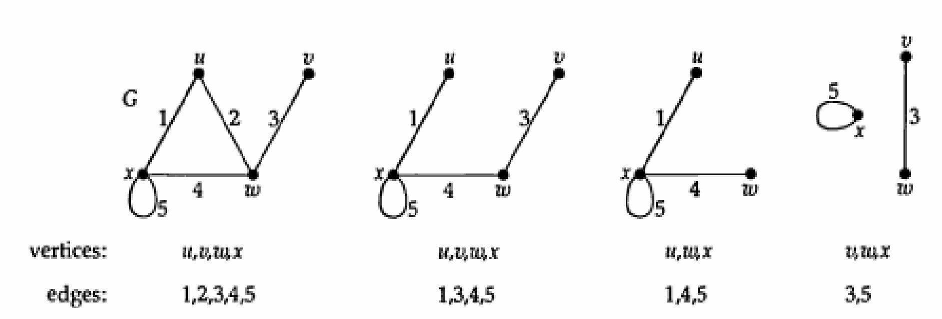
n	1	2	3	4	5	6	7	8
labelled graphs	1	2	8	64	1024	32768	2097152	268435456
unlabelled graphs	1	2	4	11	34	156	1044	12346

Catatan Sejarah
Pada tahun 1935 Matematikawan asal Hungaria, Georg Pȯlya memperoleh rumus umum yang dapat dipakai untuk menghitung jumlah graph tak berlabel dengan sejumlah vertex dan edge. Metode Pȯlya sudah diaplikasikan pada beberapa masalah perhitungan graph.

Subgraph

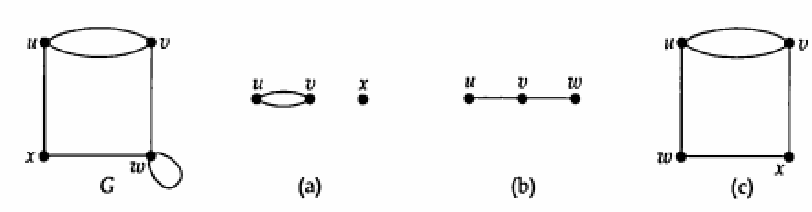
Definisi 2.5
Suatu subgraph dari graph G adalah sebuah graph di mana semua vertex-nya adalah vertex dari G dan semua edge-nya adalah edge dari G.

Sebagai contoh, Gambar 2.7 menunjukan sebuah graph G dan beberapa subgraph-nya.

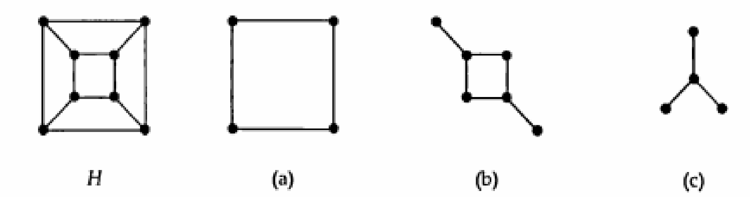


Gambar 2.7 Contoh sebuah graph dan subgraph-nya

Latihan 2.5
Mana dari graph berikut yang merupakan subgraph dari G.

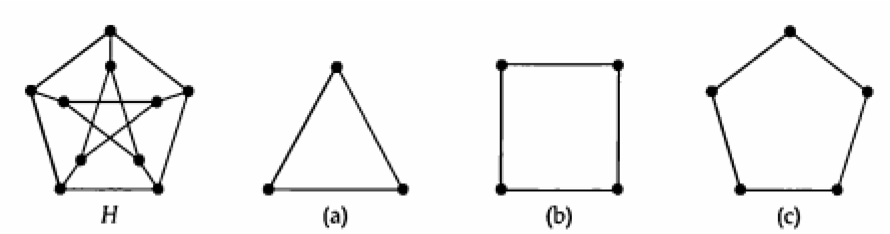


Ide dari subgraph juga dapat diperluas ke dalam graph tidak berlabel. Gambar 2.8 menunjukan graph tak berlabel H dengan subgraphnya.



Gambar 2.8 Contoh sebuah graph tidak berlabel dan subgraphnya

Latihan 2.6
Mana dari graph berikut yang merupakan subgraph dari graph tak berlabel H?

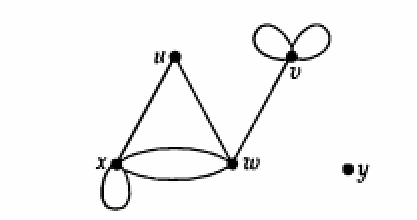


Derajat Titik

Definisi 2.6
Dalam sebuah graph, derajat dari sebuah vertex adalah jumlah edge yang insiden dengan vertex tersebut. Setiap loop memberikan dua derajat.

Sebagai contoh, Gambar 2.9 menunjukan sebuah graph yang memiliki derajat vertex sebagai berikut.

derajat *u* = 2, derajat *v* =5,
derajat *w* = 4 derajat *x* =5
derajat *y* =0



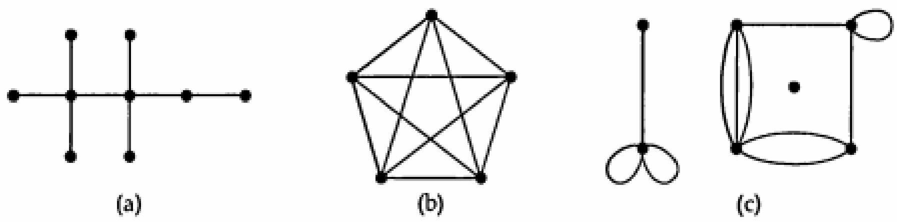
Gambar 2.9 Derajat suatu graph

Kadang kita membutuhkan daftar derajat vertex pada suatu graph dan menuliskannya dengan bentuk barisan terurut naik. Jika terdapat dua vertex yang memiliki derajat yang sama, maka pengulangan nilai tetap dituliskan. Contohnya graph pada Gambar 2.9 memiliki barisan derajat (0,2,4,5,5)

Definisi 2.7
Barisan derajat pada sebuah graph adalah barisan yang diperoleh dengan mendaftar semua derajat vertex dengan urutan naik, pengulangan dilakukan jika perlu.

Latihan 2.7

1. Tulislah barisan derajat pada graph berikut.

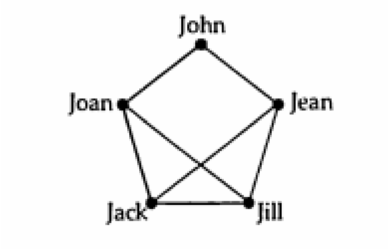


2. Tulislah jumlah edge dan jumlah derajat semua vertex dari graph pada soal bagian 1. Apa yang dapat kamu simpulkan?

Handshaking Lemma

Teorema 2.1 : Handshaking Lemma (Lemma Jabat Tangan)
Pada setiap graph, jumlah dari semua derajat vertex sama dengan dua kali jumlah edge-nya.

Pada setiap graph, setiap edge memiliki dua ujung, sehingga setiap edge menyumbang tepat dua pada jumlah derajat titiknya. Nama lemma jabat tangan muncul dari sebuah fakta bahwa graph dapat dipakai untuk mewakili sekelompok orang yang berjabat tangan seperti ditunjukan pada Gambar 2.10. Vertex pada graph tersebut mewakili orang, dan edge mewakili jabatan tangan. Dengan interpretasi ini jumlah edge mewakili jumlah jabatan, dan jumlah derajat vertex adalah jumlah dari tangan yang berjabatan.



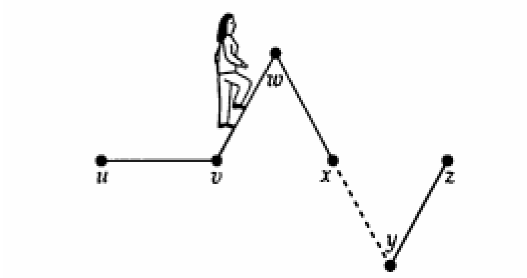
Gambar 2.10 Graph untuk representasi jabatan tangan

Catatan Sejarah
Handshaking lemma pertama muncul pada tahun 1736 pada paper Leonhard Euler tentang masalah jembatan Kõnigsberg. Paper penting ini dianggap sebagai paper pertama dalam bidang teori graph.

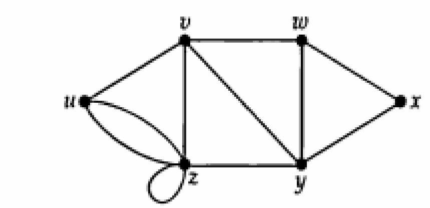
Latihan 2.8
Gunakan handshaking lemma untuk membuktikan bahwa pada setiap graph, jumlah vertex yang memiliki derajat ganjil adalah genap.

Path dan Cycle

Definisi 2.8
Suatu walk (jalan) dengan panjang *k* dalam sebuah graph adalah urutan *k* edge dengan bentuk *uv,vw,wx,…,yz*.
Walk ini dinotasikan dengan *uvwx…yz* dan disebut sebagai walk antara *u* dan *z.*



Karena edge tidak memiliki arah, kita dapat menyebut walk dari *u* ke z juga sebagai walk dari *z* ke *u* dan dinotasikan dengan *zy…xwvu*. Catat bahwa sebuah walk dapat memiliki edge atau vertex yang sama. Sebagai contoh pada Gambar 2.11, *uvwxywvzzy* adalah walk dengan panjang 9 antara vertex *u* dan *y*, yang melibatkan edge *vw* dua kali dan vertex *v,w,y* dan *z* dua kali.



Gambar 2.11 Contoh graph untuk ilustrasi walk

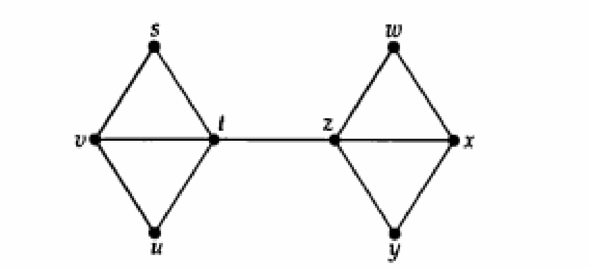
Path, Trail dan Graph terhubung

Definisi 2.9
Sebuah trail adalah sebuah walk yang semua edge-nya berbeda, namun vertex-nya boleh sama.
Sebuah path adalah sebuah walk yang semua edge dan semua vertexnya berbeda.

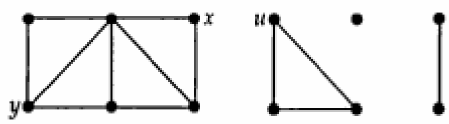
Sebagai contoh pada Gambar 2.11, walk *vzzywxy* adalah sebuah trail yang bukan path, karena vertex *y* dan *z* keduanya muncul dua kali. Sedangkan *vwxyz* tidak memiliki vertex yang sama, sehingga merupakan sebuah path.

Latihan 2.9

1. Lengkapi pernyataan ini berdasarkan graph pada Gambar 2.11.
   a. *xyzzvy* adalah sebuah ... dengan panjang … antara … dan …
   b. *uvyz* adalah sebuah … dengan panjang … antara … dan …
2. Tulis semua path antara *s* dan *y* pada graph berikut.



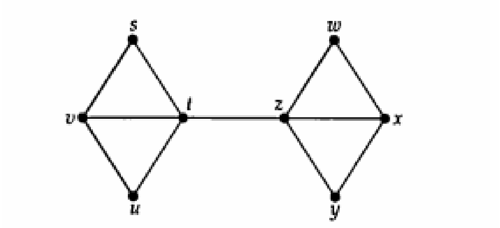
Secara intuisi, kita dapat menentukan suatu graph terhubung, jika graph tersebut menjadi satu bagian. Misalnya pada Gambar 2.12 menunjukan sebuah graph tak terhubung yang terdiri dari empat subgraph yang terhubung. Kita dapat melihat bahwa terdapat path antara *x* dan *y*, tetapi tidak ada path antara *u* dan *y*. Kita juga dapat menggunakan konsep path untuk mendefinisikan graph terhubung.



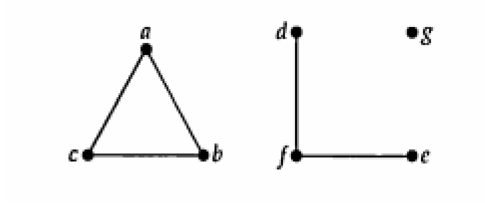
Gambar 2.12 Contoh sebuah graph tidak terhubung

Definisi 2.10
Suatu graph adalah graph terhubung jika terdapat path antara setiap pasangan vertex pada graph tersebut.
Suatu edge dalam graph terhubung disebut suatu jembatan apabila edge tersebut jika dihapus akan menyebabkan graph menjadi tidak terhunbung.
Setiap graph tidak terhubung dapat dibagi ke dalam sejumlah graph terhubung yang disebut komponen.

Sebagai contoh pada Gambar 2.13 (a), edge *tz* adalah sebuah jembatan, karena jika *tz* dihapus akan menyebabkan graph tidak terhubung. Graph pada Gambar 2.13 (b) adalah contoh graph tidak terhubung yang memiliki tiga komponen.



(a)



(b)

Gambar 2.13 Contoh graph untuk ilustrasi jembatan dan komponen

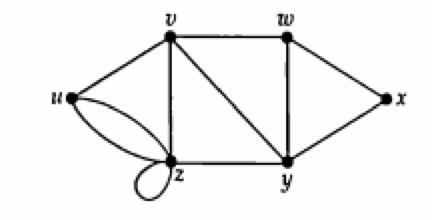
Latihan 2.10
Gambarlah
a. Sebuah graph terhubung dengan delapan vertex
b. Sebuah graph tidak terhubung dengan delapan vertex dan dua komponen
c. Sebuah graph tidak terhubung dengan delapan vertex dan tiga komponen

Trail terbuka dan Cycle

Definisi 2.11
Suatu walk tertutup pada sebuah graph adalah urutan edge dengan bentuk:
uv,vw,wx,…,yz.zu, yang berawal dan berakhir pada vertex yang sama.
Suatu trail tertutup merupakan walk tertutup di mana semua edge-nya berbeda.
Sebuah cycle adalah walk tertutup di mana semua edge dan semua vertex diantaranya berbeda.
Sebuah walk atau trail adalah terbuka jika ia berawal dan berakhir di vertex yang berbeda.

Sebagai contoh, perhatikan graph pada Gambar 2.14. Walk tertutup *vywxyzv* adalah trail tertutup yang bukan merupakan cycle, sedangkan trail tertutup *zz, vwxyv* dan *vwxyzv* semuanya merupakan cycle. Cycle dengan panjang 3 seperti *vwyv* atau *wxyw* disebut segitiga.

Dalam menjelaskan sebuah walk tertutup kita memperbolehkan vertex manapun menjadi titik awal. Contohnya pada segitiga *vwyv* dapat ditulis sebagai *wyvw* atau *yvwy,* Karena tidak terdapat arah, maka walk tersebut dapat ditulis juga sebagai *vywv*, *wvyw* atau *ywvy*.

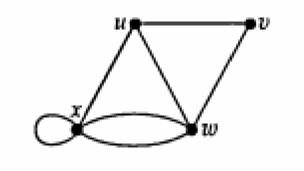


Gambar 2.14 Contoh graph untuk ilustrasi trail dan cycle

Latihan 2.11

Untuk graph berikut, tulislah:

a. Walk tertutup yang bukan trail tertutup
b. Trail tertutup yang bukan cycle
c. Semua cycle dengan panjang 1, 2, 3, dan 4.



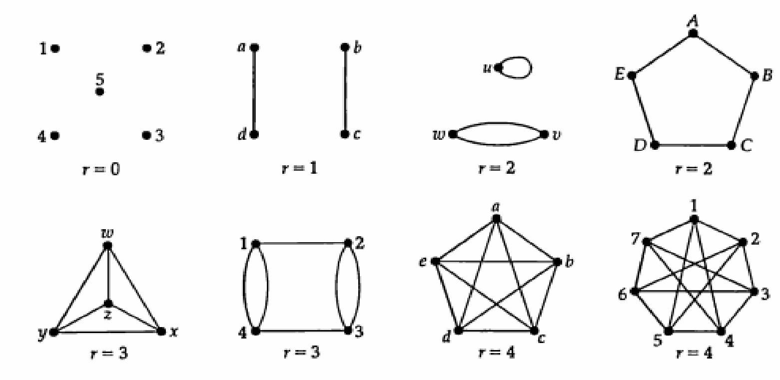
Graph Regular dan Graph Bipartit

Graph Regular

Definisi 2.12

Suatu graph merupakan graph regular jika semua vertex-nya memiliki derajat yang sama.
Graph regular disebut *r*-regular atau regular dengan derajat *r*, jika derajat semua vertex-nya adalah *r*.

Pada Gambar 2.15 kita mengilustrasikan beberapa graph *r*-regular, dengan beberapa nilai *r*.



Gambar 2.15 Contoh graph *r*-regular

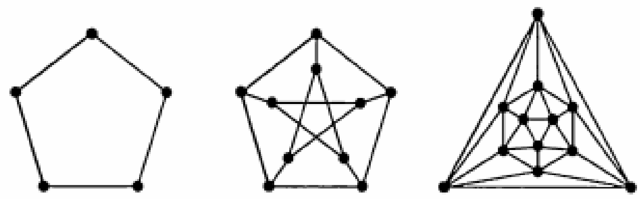
Latihan 2.12
Gambar suatu graph *r*-regular yang memiliki delapan verex dengan :
a. r =3
b. r =4
c. r =5

Teorema 2.2 :
Misal G adalah graph *r*-regular dengan *n* vertex, maka G memiliki *nr*/2 edge.

Bukti

Misalnya G adalah graph dengan *n* vertex, semua vertex memiliki derajat *r*, maka jumlah dari semua derajat vertexnya adalah *nr*. Dengan handshaking lemma diperoleh, bahwa jumlah dari edge graph tersebut seperdua dari jumlah tersebut, atau *nr*/2.

Latihan 2.13
1. Periksa bahwa teorema 2.2 berlaku untuk setiap graph regular berikut.



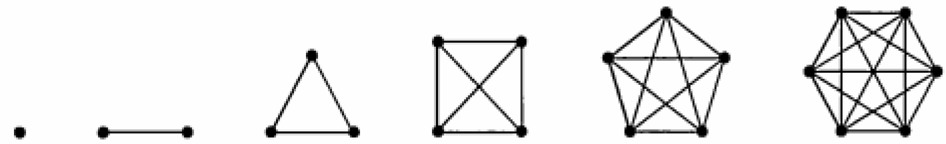
(a) (b) (c)

2. Buktikan bahwa tidak ada graph 3-regular dengan tujuh vertex
3. Buktikan bahwa, jika *n* dan *r* keduanya ganjil, maka tidak ada graph *r*-regular dengan *n* vertex.

Contoh Graph Regular

1. Graph Komplit

   Graph Komplit adalah graph yang setiap vertex-nya dihubungkan dengan semua vertex lainnya oleh tepat satu edge. Graph Komplit dengan *n* vertex dinotasikan dengan K_n. Graph K_n memiliki derajat titik *n*-1, sehingga memiliki *n*(*n*-1)/2 edge.

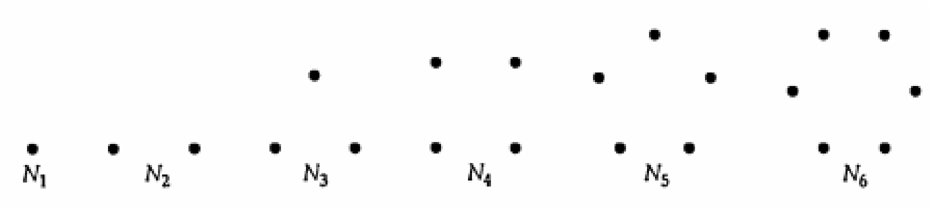


K_1 K_2 K_3 K_4 K_5 K_6

Gambar 2.16 Contoh graph Komplit

2. Graph Null

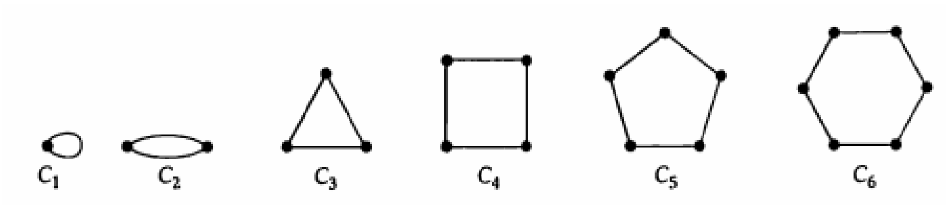
   Graph Null adalah graph yang tidak memiliki edge. Graph Null dengan n vertex dinotasikan dengan N_n. Graph N_n adalah graph regular dengan derajat 0.



Gambar 2.17 Contoh graph Null

3. Graph Cycle

   Graph cycle adalah graph yang mengandung tepat sebuah cycle pada vertex dan edge-nya. Graph cycle dengan n vertex dinotasikan dengan C_n. Graph C_n adalah graph regular dengan derajat 2 dan memiliki n edge. Untuk $n \geq 3$, C_n dapat digambar dengan bentuk polygon teratur.

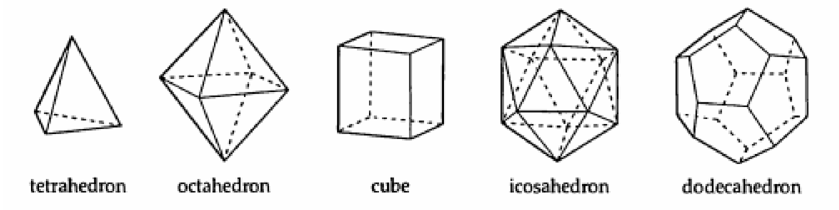


Gambar 2.18 Contoh graph Cycle

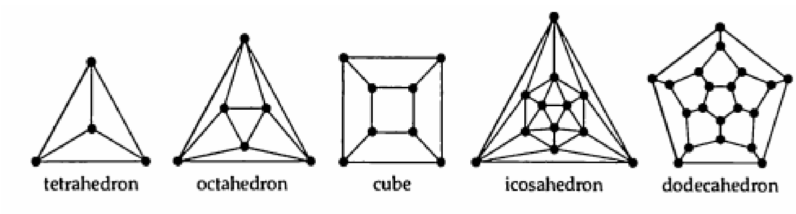
Latihan 2.14 Gambarlah graph K_7, N_7 dan C_7.

4. Graph Platonik

   Graph platonic adalah graph yang mewakili platonic solids. Kata Platonic sendiri disebut pada Plato's Timaeus dan Buku XIII dari Euclid's Elements.



(a) Platonic solid



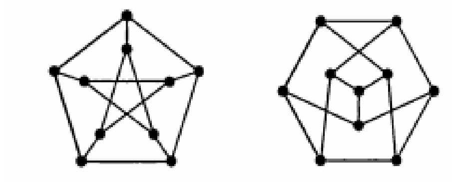
(b) Graph Platonik

Gambar 2.19 Contoh graph Platonik

Tetrahedron, cube dan dodecahedron merupakan graph 3-regular, octahedron merupakan graph 4-regular dan icosahedron merupakan graph 5-regular.

5. Graph Patersen

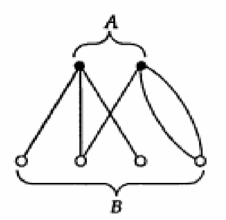
Graph Patersen adalah graph 3-regular dengan 10 vertex dan 15 edge, yang dapat digambar dengan berbagai cara. Dua cara diantaranya ditunjukan pada Gambar 2.20. Graph Patersen didiskusikan dalam paper Julius Patersen (matematkawan Denmark) pada tahun 1898.



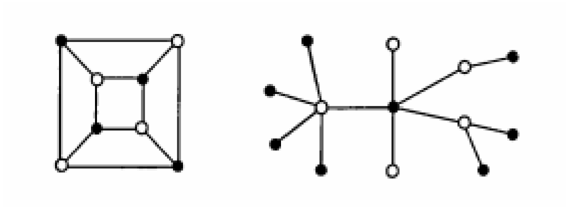
Gambar 2.20 Contoh graph Patersen

Graph Bipartit

Definisi 2.13
Graph bipartit adalah graph di mana himpunan vertexnya dapat dibagi menjadi dua bagian: A dan B, sedemikian rupa sehingga setiap edge pada graph tersebut menghubungkan sebuah vertex di A dengan sebuah vertex di B.



Kita dapat membedakan vertex pada himpunan A dari vertex pada himpunan B dengan menggambar satu himpunan vertex berwarna hitam dan satu himpunan vertex berwarna putih. Setiap edge insiden dengan vertex hitam dan vertex putih. Contoh dua graph bipartit ditunjukan pada Gambar 2.21.



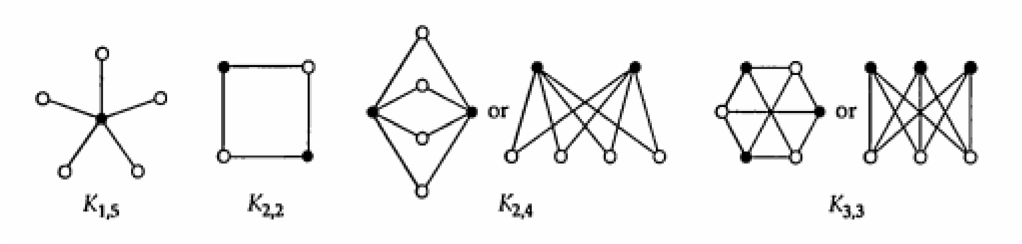
Gambar 2.21 Contoh graph bipartit

Latihan 2.15
Buktikan bahwa pada graph bipartit, setiap cycle memiliki jumlah edge yang genap.

Contoh Graph Bipartit

1. Graph Bipartit Komplit

   Graph bipartit komplit adalah graph bipartit yang setiap vertex pada himpunan A dihubungkan dengan setiap vertex di himpunan B oleh satu edge. Graph bipartit komplit dengan *r* vertex pada himpunan A dan *s* vertex pada himpunan B dinotasikan dengan $K_{r,s}$.



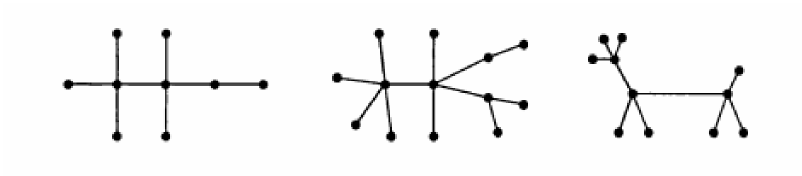
Gambar 2.22 Contoh graph bipartit komplit

Graph $K_{r,s}$ sama dengan graph $K_{r,s}$, memiliki *rs* edge dan *r+s* titik. Graph ini memiliki *r* vertex dengan derajat *s* dan *s* vertex dengan derajat *r*.

Latihan 2.16
1. Gambar graph $K_{2,3}$, $K_{1,7}$, dan $K_{4,4}$
2. Berapa banyak vertex dan edge pada masing-masing graph tersebut?
3. Dengan kondisi *r* dan *s* seperti apa, $K_{r,s}$ termasuk graph regular?

2. Graph Tree

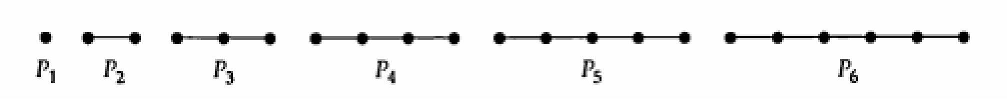
   Tree adalah graph terhubung yang tidak memiliki cycle. Karena tree adalah graph yang terhubung, setidaknya terdapat satu path antara setiap pasang vertex pada tree tersebut. Misalnya ada dua vertex yang terhubung dengan dua path, maka path tersebut akan menciptakan cycle. Sehingga pada tree, hanya ada tepat sebuah path antara setiap dua pasang vertex.



Gambar 2.23 Contoh graph Tree

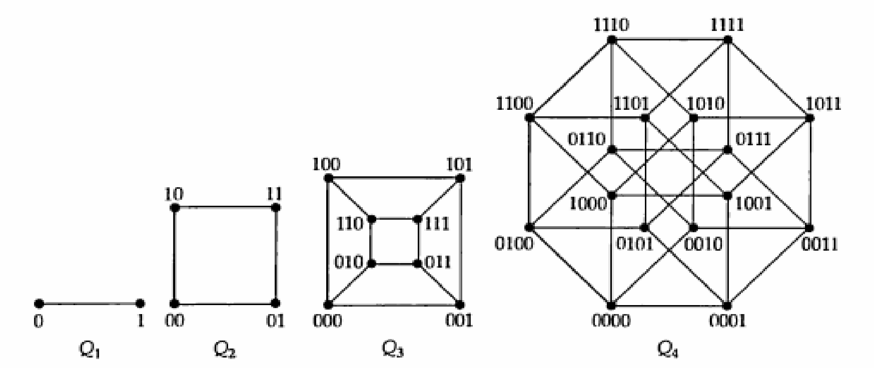
Latihan 2.17
1. Ada delapan tree tidak berlabel yang dapat digambar jika kita memiliki lima vertex atau kurang. Gambarlah graph tersebut.
2. Jelaskan mengapa setiap tree adalah graph bipartit dengan mewarnai vertex dengan warna hitam dan putih
3. Jelaskan mengapa suatu tree dengan *n* vertex memiliki *n*-1 edge.

3. Graph Path
   Graph Path adalah sebuah tree yang terdiri dari sebuah path yang melalui semua vertex-nya. Graph path dengan n vertex dinotasikan sebagai P_n. P_n memiliki n-1 edge, dan P_n dapat diperoleh dari graph cycle C_n dengan menghapus suatu edge.



Gambar 2.24 Contoh graph Path

4. Graph Cube
   Salah satu graph bipartit yang menarik adalah cube. Graph cube memiliki aplikasi penting dalam teori koding dan dibangun dengan menggambarkan vertex sebagai barisan bilangan biner (barisan angka 0 dan 1) dengan panjang tertentu. Setiap edge menghubungkan dua vertex yang angka pada barisannya berbeda pada satu tempat saja.
   Graph yang diperoleh dari bilangan biner dengan panjang k disebut k-cube atau cube k-dimensi, dan dinotasikan dengan Q_k. Graph Q_k memiliki 2^k vertex dan merupakan graph regular dengan derajat k. Q_k memiliki k x 2^{k-1} edge.



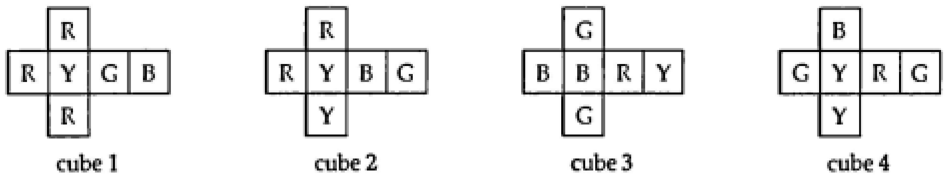
Gambar 2.25 Contoh graph Cube

Studi Kasus

Masalah Susunan Empat Kubus

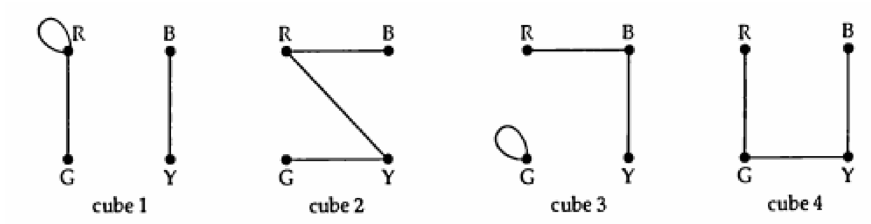
Terdapat suatu masalah hiburan yang menarik, di mana kita akan bermain dengan empat kubus yang memiliki permukaan berwarna merah (R), biru (B), hijau (G) dan kuning (Y). Bagaimana caranya kita menumpuk kubus sehingga semua warna muncul pada setiap sisi dari tumpukan yang dihasilkan.

Kita tahu bahwa banyak sekali cara dalam menyusun keempat kubus tersebut. Perhatikan bahwa jika suatu permukaan kubus muncul pada satu sisi tumpukan, maka permukaan yang berlawanan pada kubus akan muncul pasa sisi yang berlawanan pada tumpukan. Oleh karena itu, konsentrasi utama kita adalah pada pasangan permukaan kubus yang berlawanan. Perhatikan Gambar 2.26, pada setiap kubus, kita akan mendapatkan tiga pasangan permukaan yang berlawanan.

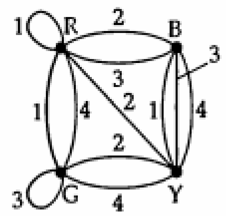


Gambar 2.26 Contoh masalah susunan empat buah kubus

Gambar 2.26 menunjukan empat buah kubus yang harus kita susun. Untuk menyelesaikan masalah ini, kita dapat merepresentasikan setiap kubus dengan graph yang menunjukan pasangan warna yang muncul pada permukaan yang berlawanan. Representasikan graph dengan vertex *R,B,G,Y* (mewakili empat warna), di mana dua vertex dihubungkan jika dua warna tersebut muncul pada permukaan yang berlawanan. Untuk kasus tersebut, kita dapat memperoleh graph pada Gambar 2.27(a). Selanjutnya, satukan semua graph yang diperoleh dari empat buah kubus menjadi graph baru G seperti ditunjukan pada Gambar 2.27 (b).



(a) Representasi graph untuk setiap kubus



(b) Satukan keempat graph menjadi graph baru

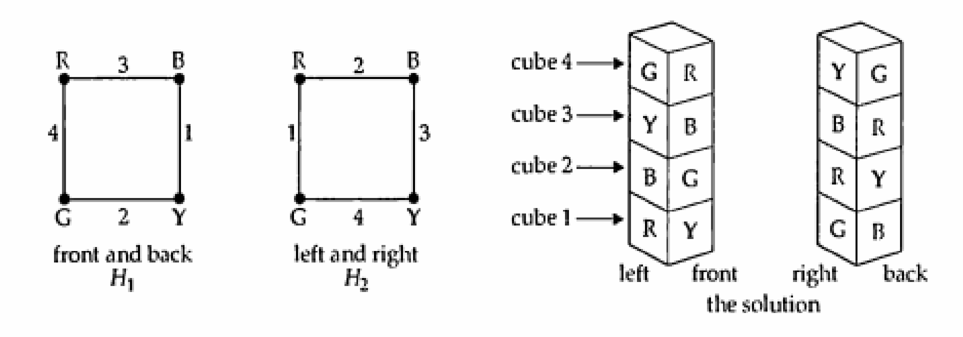
Gambar 2.27 Representasi graph untuk masalah susunan empat kubus

Solusi dari masalah ini didapatkan dengan menemukan dua buah subgraph H_1 dan H_2 dari graph G. H_1 menggambarkan pasangan warna yang muncul pada sisi depan dan belakang, dan H_2 mnunjukan pasangan warna yang muncul pada sisi kanan dan sisi kiri. Subgraph H_1 dan H_2 harus memenuhi tiga sifat berikut :

1. Setiap subgraph mengandung tepat satu edge dari keempat graph kubus
   Sifat ini menggambarkan bahwa setiap kubus memiliki sisi depan dan belakang, dan sisi kanan dan kiri. Dan subgraph H_1 dan H_2 menggambarkan pasangan warna mana yang muncul pada permukaan tersebut.

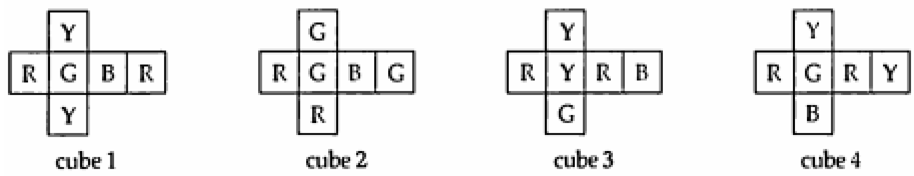
2. Kedua subgraph tidak memiliki edge yang sama
   Sifat ini menggambarkan bahwa permukaan yang muncul pada sisi depan dan belakang sebuah kubus tidak boleh sama dengan permukaan yang muncul pada sisi kanan dan kiri.
3. Setiap vertex insiden dengan dua edge.
   Sifat ini menggambarkan bahwa setiap warna muncul tepat dua kali pada sisi kanan dan kiri tumpukan (sekali pada setiap sisi) dan tepat dua kali pada sisi depan dan belakang (satu kali di depan dan satu kali dibelakang).

Salah satu solusi yang muncul pada permasalah ini disajikan pada Gambar 2.28 Subgraph H_1 dan H_2 menunjukan kubus 1 dipasang dengan posisi kuning di depan, biru di belakang (dari H_1) dan merah di kiri dan hijau di kanan (dari H_2). Demikian juga dengan kubus yang lain.



Gambar 2.28 Contoh solusi masalah empat kubus

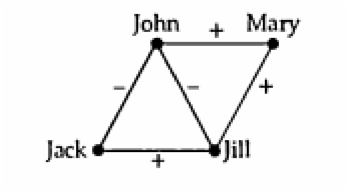
Latihan 2.18
Gunakan pendekatan yang sudah dipelajari sebelumnya untuk mencari solusi dari masalah empat kubus dengan kubus-kubus berikut.



Jaringan Sosial

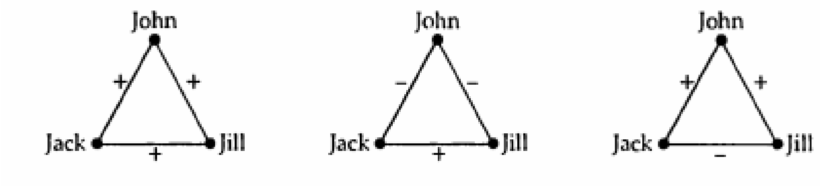
Graph telah banyak dipakai dalam ilmu sosial untuk merepresentasikan hubungan interpersonal. Vertex pada graph mewakili sekumpulan orang dan edge menghubungan dua pasang individu yang berhubungan dengan berbagai cara. Misalnya vertex *a* dihubungkan dengan vertex *b* jika *a* suka, benci, setuju, menghindari, atau berkomunikasi dengan *b.* Graph juga banyak dipakai dalam ilmu politik untuk mempelajari hubungan internasional, di mana vertex mewakili kumpulan negara dan edge menghubungkan dua pasang negara yang beraliansi, memiliki hubungan diplomatik, setuju dengan strategi tertentu, dan lain-lain.

Kita dapat menganalisa ketegangan pada suatu situasi dengan menggunakan konsep graph bertanda. Graph ini menandai edge dengan tanda + dan – yang mengindikasikan hubungan positif (suka, cinta, setuju, berkomunikasi, dan lain-lain) atau hubungan negatif (tidak suka, benci, tidak setuju, menghindar, dan lain-lain). Sebagai contoh, pada Gambar 2.29, graph bertanda tersebut menunjukan bahwa Jack suka Jill tapi tidak suka John, Jill suka Jack dan Mary tapi tidak suka John, Mary suka John dan Jill, dan John suka Mary tapi tidak suka Jack ataupun Jill. Jack dan Mary masing-masing tidak memiliki perasaan yang kuat, dan mereka tidak dihubungkan dengan edge.



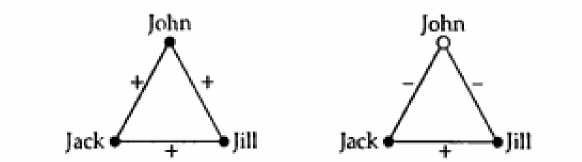
Gambar 2.29 Contoh graph bertanda

Misalnya Gambar 2.30 menggambarkan suatu situasi yang terjadi pada tiga orang John, Jack dan Jill yang bekerja bersama. Kita dpat melihat pada situasi mana yang terdapat ketegangan antara John, Jack dan Jill. Pada kasus pertama, semua saling menyukai, sehingga tidak terjadi ketegangan. Pada kasus kedua Jack dan Jill saling suka, dan keduanya tidak suka John. Pada situasi ini juga tidak terjadi ketegangan, karena Jack dan Jill bekerja bersama, dan John bekerja sendiri. Pada kasus ketiga, John suka Jack dan Jill dan mau bekerja dengan mereka. Namun Jack dan Jill saling tidak suka satu sama lain dan tidak mau bekerja sama. Oleh karena itu, pada kasus ini terjadi ketegangan.



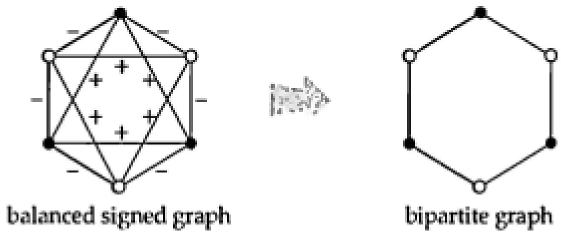
Gambar 2.30 Graph bertanda yang balance dan unbalance

Kita menyebut situasi tanpa ketegangan dengan situasi yang *balance* (seimbang) dan situasi di mana terdapat ketegangan dengan situasi *unbalance* (tidak seimbang). Secara umum, kita menyebut suatu graph bertanda itu balance jika kita dapat mewarnai vertex pada graph itu dengan warna hitam dan putih sehingga edge bertanda positif memiliki ujung dengan warna yang sama, dan edge bertanda negatif memiliki ujung hitam dan ujung putih. Pada kasus pertama dan kedua kita dapat mewarnai graph seperti pada Gambar 2.31.



Gambar 2.31 Pewarnaan graph bertanda yang balance

Definisi tersebut menyerupai definisi graph bipartit. Untuk melihat hubungan keduanya, ambil suatu graph bertanda dan hilangkan semua edge yang memiliki tanda positif, maka kita akan mendapatkan graph bipartit seperti ditunjukan pada Gambar seperti gambar 2.32. Kita perhatikan bahwa pada graph bipartit, setiap cycle memiliki jumlah edge yang genap. Sedangkan pada graph bertanda yang balance, setiap cycle memiliki jumlah edge bertanda negatif yang genap.



Gambar 2.32 Merubah graph bertanda yang balance ke dalam graph bipartit

Latihan 2.19

Tentukan graph mana pada graph bertanda berikut yang balance, dan temukan graph bipartit yang berkorespondensi dengannya.