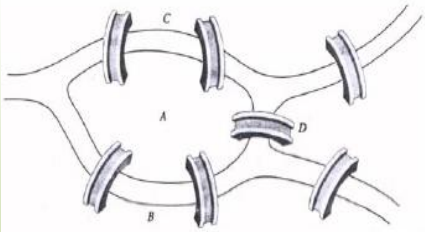
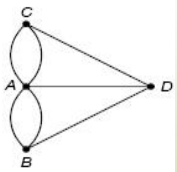
Diktat Kuliah

GRAF TERAPAN

(Digunakan untuk kalangan sendiri)





Ari Mulyoto, S.Pd, M.Si.

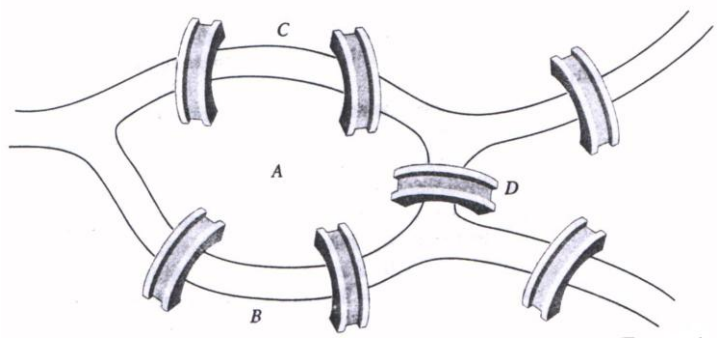
JURUSAN TEKNIK REKAYASA PERANGKAT LUNAK

UNIVERSITAS PAMULANG

DAFTAR ISI

halaman

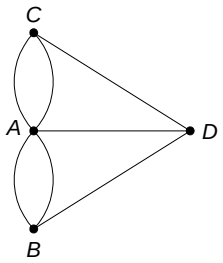
PENDAHULUAN



Gambar 0.1. Masalah Jembatan Kőnigsberg.

Graf digunakan untuk merepresentasikan objek-objek diskrit dan hubungan antara objek-objek tersebut. Istilah graf pertama kali muncul pada tahun 1736 yang dikaji oleh ilmuwan Swiss yang bernama ***Leonard Euler***. Dia mengemukakan permasalahan yang dikenal dengan "*Masalah Jembatan Kőnigsberg"*. Dapatkah seseorang melalui 7 jembatan tersebut tepat satu kali dan kembali ke tempat semula? (seperti pada Gambar 0.1).

Berikut adalah sketsa yang merepresentasikan ilustrasi jembatan Königsberg. Himpunan titik yaitu {A, B, C, D} merepresentasikan sebagai daratan, dan garis yang menghubungkan titik-titik tersebut adalah sebagai jembatan.



Gambar 0.2. Sketsa representasi Jembatan Kőnigsberg.

Jawaban pertanyaan Euler adalah *tidak mungkin*. Agar bisa melalui setiap jembatan tepat sekali dan kembali lagi ke tempat semula maka jumlah jembatan yang menghubungkan setiap daratan harus genap.

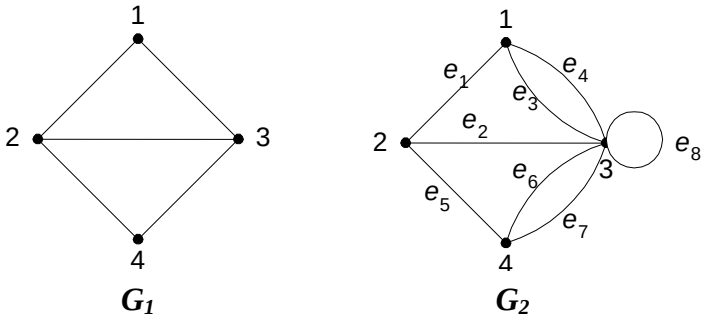
Graf merupakan struktur diskrit yang terdiri himpunan sejumlah berhingga obyek yang disebut simpul (*vertices, vertex*) dan himpunan sisi (*edges*) yang menghubungkan simpul-simpul tersebut.

Bab. 1 TEORI GRAF

A. DEFINISI GRAF

Notasi sebuah graf adalah $G = (V, E)$, dimana :

- ***V*** merupakan himpunan tak kosong dari simpul-simpul (*vertices/vertex*), misalkan $V = \{ v_1, v_2, \ldots, v_n \}$
- ***E*** merupakan himpunan sisi – sisi (*edges*) yang menghubungkan sepasang simpul, misalkan $E = \{e_1, e_2, \ldots, e_n\}$



Gambar 1.1. Graf Sederhana dan Graf Tidak Sederhana

Contoh 5.1.

Pada Gambar 5.3, G_1 adalah graf dengan

$V = \{ 1, 2, 3, 4 \}$

$E = \{ (1, 2), (1, 3), (2, 3), (2, 4), (3, 4) \}$

G_2 adalah graf dengan

$V = \{ 1, 2, 3, 4 \}$

$E = \{ (1, 2), (2, 3), (1, 3), (1, 3), (2, 4), (3, 4), (3, 4), (3, 3) \}$

$= \{e_1, e_2, e_3, e_4, e_5, e_6, e_7, e_8\}$

Pada G_2, sisi e_3 dan sisi $e_4 = (1, 3)$ dinamakan **sisi-ganda** (*multiple edges* atau *paralel edges*) karena kedua sisi ini menghubungi dua buah simpul yang sama, yaitu simpul 1 dan simpul 3, dan sisi $e_8 = (3, 3)$ dinamakan **gelang** atau **kalang** (*loop*) karena ia berawal dan berakhir pada simpul yang sama.

B. JENIS-JENIS GRAF

Berdasarkan ada tidaknya gelang atau sisi ganda pada suatu graf, maka graf digolongkan menjadi dua jenis:

1. **Graf Sederhana** (*simple graph*).

   Graf yang tidak mengandung gelang maupun sisi-ganda dinamakan graf sederhana. G_1 pada Gambar 1.1. adalah contoh graf sederhana

2. **Graf Tak-Sederhana** (*Unsimple-Graph*).

Graf yang mengandung sisi ganda atau gelang dinamakan graf tak-sederhana (*unsimple graph*). G_2 pada Gambar 1.1. adalah contoh graf tak-sederhana

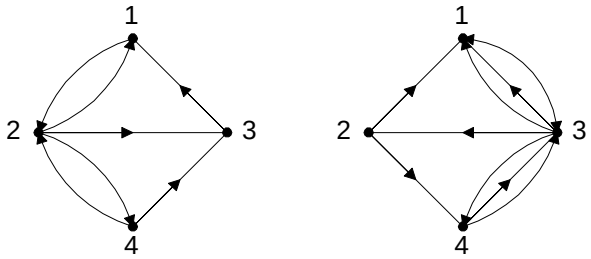
Berdasarkan orientasi arah pada sisi, maka secara umum graf dibedakan atas 2 jenis:

1. **Graf Tak-Berarah** (*Undirected Graph*)

Graf yang sisinya tidak mempunyai orientasi arah disebut graf tak-berarah. Dua buah graf pada Gambar 1.1. adalah graf tak-berarah.

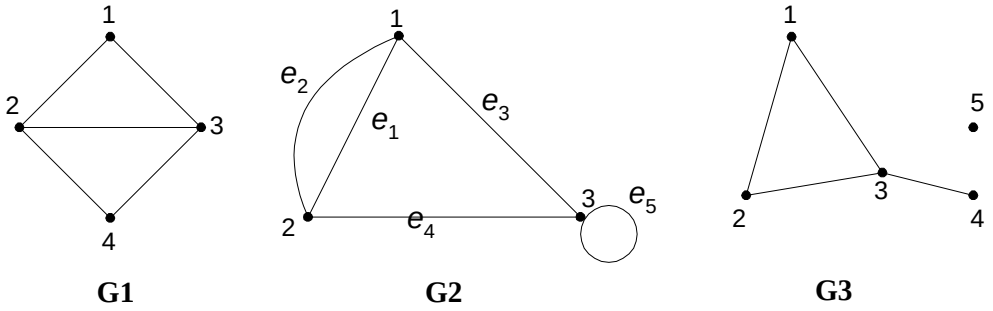
2. **Graf Berarah** (*Directed Graph* Atau *Digraph*)

Graf yang setiap sisinya diberikan orientasi arah disebut sebagai graf berarah. Dua buah graf pada Gambar 1.2. adalah graf berarah.



Gambar 1.2. Graf Berarah

C. TERMINOLOGI GRAF



Gambar 1.3. Terminologi Graf

1. Ketetanggaan (*Adjacent*)

Dua buah simpul (vertex) dikatakan *bertetangga* bila keduanya terhubung langsung oleh sisi.

Pada graf G_1 : simpul 1 bertetangga dengan simpul 2 dan 3,
simpul 1 tidak bertetangga dengan simpul 4.

2. Bersisian (*Incidency*)

Untuk sembarang sisi $e = (v_i , v_j)$ dikatakan
e bersisian dengan simpul v_j , atau
e bersisian dengan simpul v_i

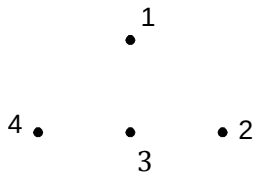
Pada graf G_2: e_2 bersisian dengan simpul 1 dan simpul 2
e_3 bersisian dengan simpul 1 dan simpul 3,
tetapi e_4 tidak bersisian dengan simpul 1.

3. Simpul Terpencil (*Isolated Vertex*)

Simpul terpencil ialah simpul yang tidak mempunyai sisi yang bersisian dengannya.
Tinjau graf G_3: simpul 5 adalah simpul terpencil.

4. Graf Kosong (*null graph* atau *empty graph*)

Graf kosong adalah graf yang himpunan sisinya merupakan himpunan kosong (N_n). Dengan kata lain graf kosong adalah graf yang tidak memiliki sisi.



Gambar 1.4.. Graf kosong N_4

5. Derajat (*Degree*)

Derajat suatu simpul adalah jumlah sisi yang bersisian dengan simpul tersebut.
Notasi: $d(v)$

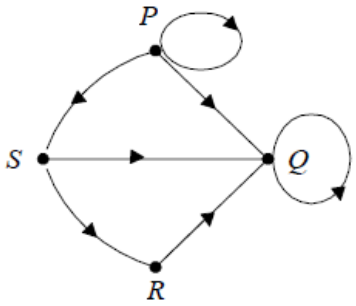
Tinjau graf G_1:
$d(1) = d(4) = 2$
$d(2) = d(3) = 3$

Tinjau graf G_3: $d(5) = 0$ → simpul terpencil
$d(4) = 1$ → simpul anting-anting (*pendant vertex*)

Tinjau graf G_2: $d(1) = 3$ → bersisian dengan sisi ganda
$d(3) = 4$ → bersisian dengan sisi gelang (*loop*)

Pada graf berarah,
$d_{in}(v)$ = derajat-masuk (*in-degree*)
= jumlah busur yang masuk ke simpul v
$d_{out}(v)$ = derajat-keluar (*out-degree*)
= jumlah busur yang keluar dari simpul v

$d(v) = d_{in}(v) + d_{out}(v)$



Gambar 1.5. Derajat simpul graf berarah

Pada Gambar 1.5 :

$d_{in}(P) = 1$ dan $d_{out}(P) = 3$ maka $d(P) = 4$
$d_{in}(Q) = 4$ dan $d_{out}(Q) = 1$ maka $d(Q) = 5$
$d_{in}(R) = 1$ dan $d_{out}(R) = 1$ maka $d(R) = 2$
$d_{in}(S) = 1$ dan $d_{out}(S) = 2$ maka $d(S) = 3$

Lemma Jabat Tangan : *Jumlah derajat semua simpul pada suatu graf adalah genap, yaitu dua kali jumlah sisi pada graf tersebut.*

Dengan kata lain, jika $G = (V, E)$, maka :

$$\sum_{v \in V} d(v) = 2|E|$$

Tinjau graf G_1: $d(1) + d(2) + d(3) + d(4) = 2 + 3 + 3 + 2 = 10$
$= 2 \times$ jumlah sisi $= 2 \times 5$

Tinjau graf G_2: $d(1) + d(2) + d(3) = 3 + 3 + 4 = 10$
$= 2 \times$ jumlah sisi $= 2 \times 5$

Tinjau graf G_3: $d(1) + d(2) + d(3) + d(4) + d(5)$
$= 2 + 2 + 3 + 1 + 0 = 8$
$= 2 \times$ jumlah sisi $= 2 \times 4$

Contoh 5.2.

Diketahui graf dengan lima buah simpul. Dapatkah kita menggambar graf tersebut jika derajat masing-masing simpul adalah:

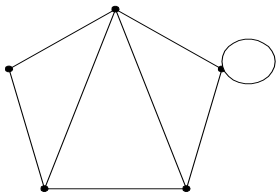
(a) 2, 3, 1, 1, 2

(b) 2, 3, 3, 4, 4

Penyelesaian:

(a) ***tidak dapat***, karena jumlah derajat semua simpulnya ganjil
(2 + 3 + 1 + 1 + 2 = 9).

(b) ***dapat***, karena jumlah derajat semua simpulnya genap
(2 + 3 + 3 + 4 + 4 = 16).



6. Lintasan (*Path*)

Lintasan yang panjangnya n dari simpul awal v_0 ke simpul tujuan v_n di dalam graf G ialah barisan berselang-seling simpul-simpul dan sisi-sisi yang berbentuk v_0, e_1, v_1, e_2, v_2,... , v_{n-1}, e_n, v_n sedemikian sehingga $e_1 = (v_0, v_1)$, $e_2 = (v_1, v_2)$, ... , $e_n = (v_{n-1}, v_n)$ adalah sisi-sisi dari graf G. Lintasan 1, 2, 4, 3 pada G_1 adalah lintasan dengan barisan sisi (1,2), (2,4), (4,3).

Panjang lintasan adalah jumlah sisi dalam lintasan tersebut. Lintasan 1, 2, 4, 3 pada G_1 memiliki panjang 3.

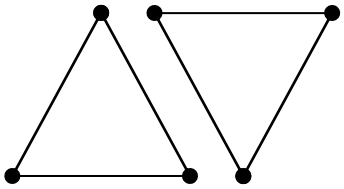
7. Siklus (*Cycle*) atau Sirkuit (*Circuit*)

Lintasan yang berawal dan berakhir pada simpul yang sama disebut **sirkuit** atau **siklus**.
Pada graf G_1 : Lintasan 1, 2, 3, 1 adalah sebuah sirkuit.
Panjang sirkuit adalah jumlah sisi dalam sirkuit tersebut. Sirkuit 1, 2, 3, 1 pada G_1 memiliki panjang 3.

8. Terhubung (*Connected*)

Dua buah simpul v_1 dan simpul v_2 disebut **terhubung** jika terdapat lintasan dari v_1 ke v_2. G disebut **graf terhubung** (*connected graph*) jika untuk setiap pasang simpul v_i dan v_j dalam himpunan V terdapat lintasan dari v_i ke v_j. Jika tidak, maka G disebut **graf tak-terhubung** (*disconnected graph*).

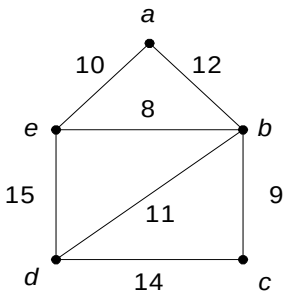
Contoh graf tak-terhubung:



Gambar 1.6.. Graf tak terhubung.

9. Graf Berbobot (*Weighted Graph*)

Graf berbobot adalah graf yang setiap sisinya diberi sebuah harga (bobot).

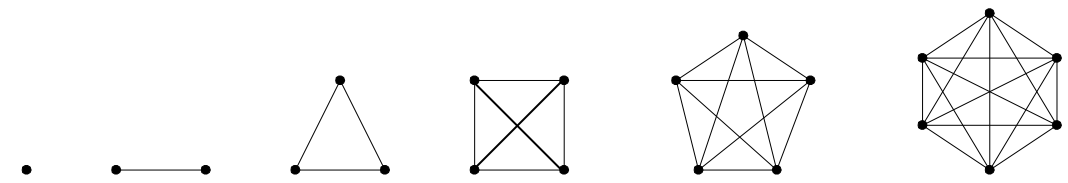


Gambar 1.7. Graf berbobot.

D. BEBERAPA GRAF SEDERHANA KHUSUS

a. Graf Lengkap (*Complete Graph*)

Graf lengkap ialah graf sederhana yang setiap simpulnya mempunyai sisi ke semua simpul lainnya. Graf lengkap dengan n buah simpul dilambangkan dengan K_n. Jumlah sisi pada graf lengkap yang terdiri dari n buah simpul adalah $n(n-1)/2$.

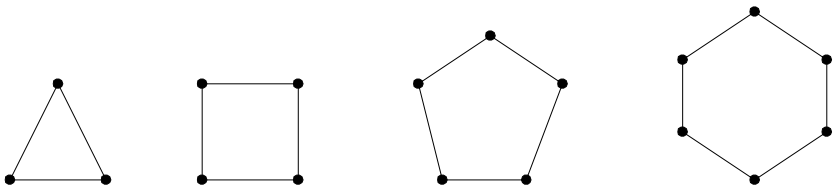


K_1 K_2 K_3 K_4 K_5 K_6

Gambar 1.8. Graf lengkap.

b. Graf Lingkaran (Cycle Graft)

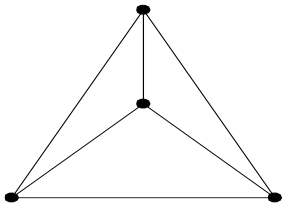
Graf lingkaran adalah graf sederhana yang setiap simpulnya berderajat dua. Graf lingkaran dengan n simpul dilambangkan dengan C_n.



Gambar 1.9. Graf lingkaran.

c. Graf Teratur (*Regular Graphs*)

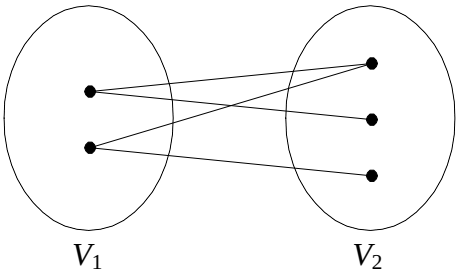
Graf yang setiap simpulnya mempunyai derajat yang sama disebut **graf teratur**. Apabila derajat setiap simpul adalah r, maka graf tersebut disebut sebagai graf teratur derajat r. Jumlah sisi pada graf teratur adalah $nr/2$.



Gambar 1.10. Graf teratur.

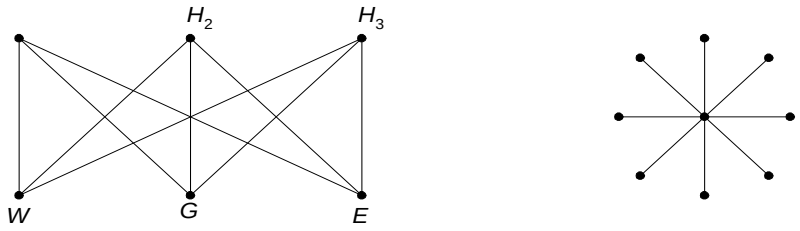
d. Graf *Bipartit* (*Bipartite Graph*)

Graf G yang himpunan simpulnya dapat dipisah menjadi dua himpunan bagian V_1 dan V_2, sedemikian sehingga setiap sisi pada G menghubungkan sebuah simpul di V_1 ke sebuah simpul di V_2 disebut **graf bipartit** dan dinyatakan sebagai $G(V_1, V_2)$.



Gambar 1.11. Graf bipartit.

Graf G_1 dan G_2 berikut adalah graf bipartit, karena simpul-simpunya dapat dibagi menjadi $V_1 = \{a, b, d\}$ dan $V_2 = \{c, e, f, g\}$



G_1
graf persoalan utilitas,

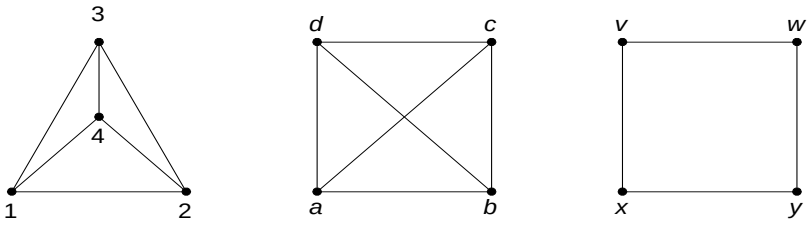
G_2
topologi bintang

Gambar 1.12 : Graf bipartit lain.

e. Graf Isomorfik (*Isomorphic Graph*)

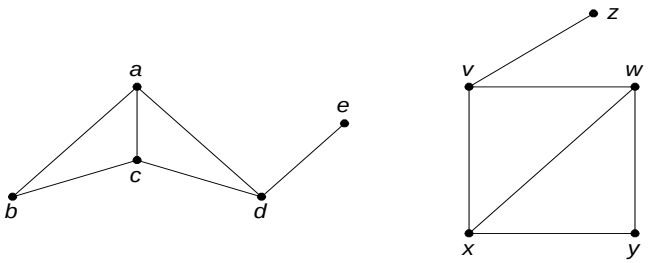
Dua buah graf yang sama tetapi secara geometri berbeda disebut graf yang saling **isomorfik**. Dua buah graf, G_1 dan G_2 dikatakan isomorfik jika terdapat korespondensi satu-satu antara simpul-simpul keduanya dan antara sisi-sisi keduaya sedemikian sehingga hubungan kebersisian tetap terjaga. Dengan kata lain, misalkan sisi *e* bersisian dengan simpul *u* dan *v* di G_1, maka sisi *e*' yang berkoresponden di G_2 harus bersisian dengan simpul *u*' dan *v*' yang di G_2.

Dua buah graf yang isomorfik adalah graf yang sama, kecuali penamaan simpul dan sisinya saja yang berbeda. Ini benar karena sebuah graf dapat digambarkan dalam banyak cara.



(a) G_1 (b) G_2 (c) G_3

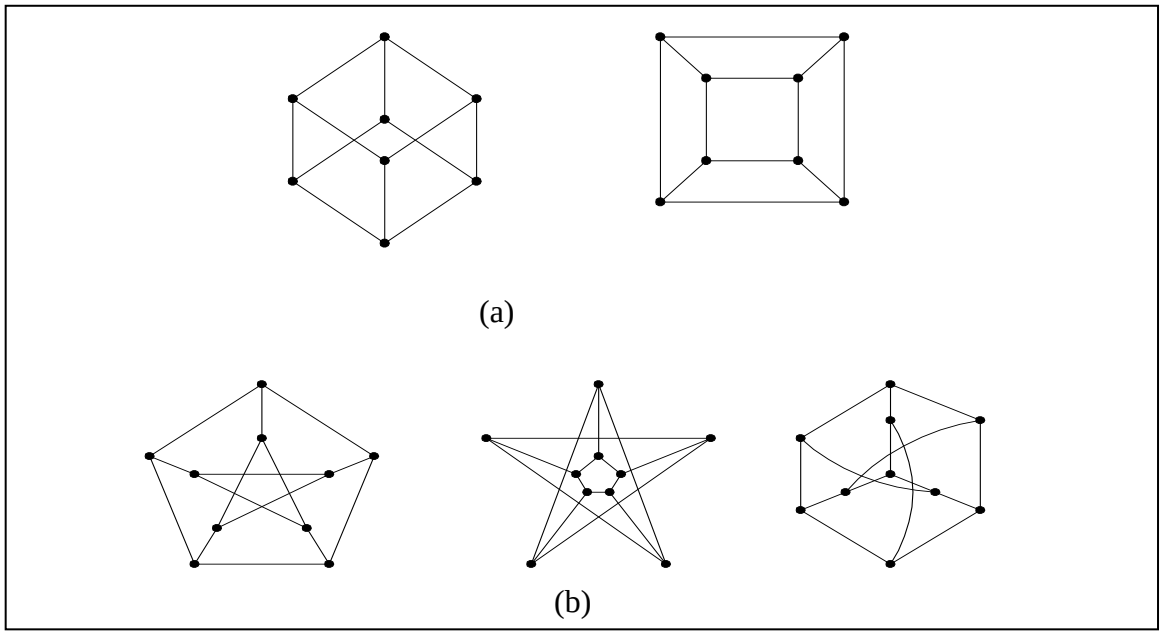
Gambar 1.13 : *G_1 isomorfik dengan G_2, tetapi G_1 tidak isomorfik dengan G_3*



(a) G_1 (b) G_2

Gambar 1.14. *Graf (a) dan graf (b) isomorfik*

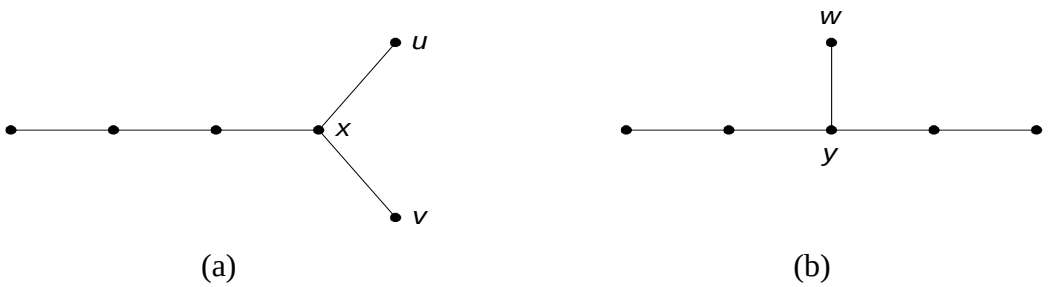
$$A_{G1} = \begin{array}{c} \\ a \\ b \\ c \\ d \\ e \end{array} \begin{array}{c} \begin{array}{ccccc} a & b & c & d & e \end{array} \\ \begin{bmatrix} 0 & 1 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix} \end{array} \qquad A_{G2} = \begin{array}{c} \\ x \\ y \\ w \\ v \\ z \end{array} \begin{array}{c} \begin{array}{ccccc} x & y & w & v & z \end{array} \\ \begin{bmatrix} 0 & 1 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix} \end{array}$$



Gambar 1.15. *(a) Dua buah graf isomorfik, (b) tiga buah graf isomorfik*

Dari definisi graf isomorfik dapat dikemukakan bahwa dua buah graf isomorfik memenuhi ketiga syarat berikut [DEO74]:
1. Mempunyai jumlah simpul yang sama.
2. Mempunyai jumlah sisi yang sama
3. Mempunyai jumlah simpul yang sama berderajat tertentu

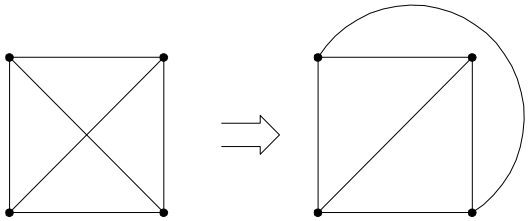
Namun, ketiga syarat ini ternyata belum cukup menjamin. Pemeriksaan secara visual perlu dilakukan.



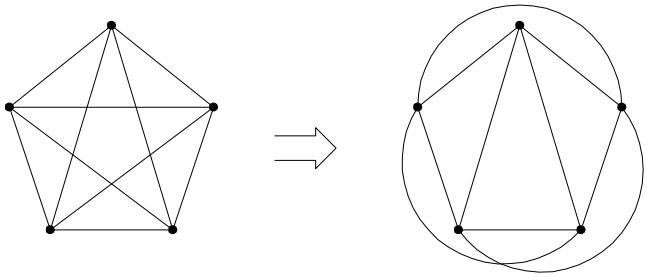
Gambar 1.16. *Dua buah graf tidak isomorfik,*

f. Graf Planar (*Planar Graph*) dan Graf Bidang (*Plane Graph*)

Graf yang dapat digambarkan pada bidang datar dengan sisi-sisi tidak saling memotong disebut sebagai **graf planar**, jika tidak, ia disebut **graf tak-planar**.

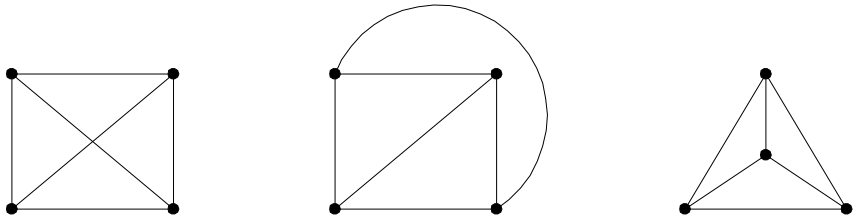


Gambar 1.17. *K_4 adalah graf planar*



Gambar 1.18. *K_5 bukan graf planar*

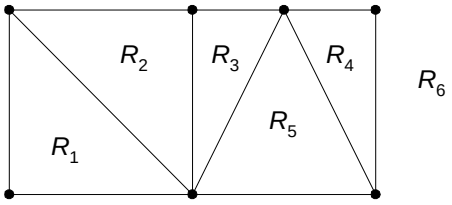
Graf planar yang digambarkan dengan sisi-sisi yang tidak saling berpotongan disebut **graf bidang** (*plane graph*).



(a) (b) (c)

Gambar 1.19. *Tiga buah graf planar. Graf (b) dan (c) adalah graf bidang*

Sisi-sisi pada graf planar membagi bidang menjadi beberapa wilayah (*region*) atau muka (*face*). Jumlah wilayah pada graf planar dapat dihitung dengan mudah.



Gambar 1.20. *Graf planar yang terdiri atas 6 wilayah*

Beberapa hal tentang graf planar $G(V, E)$, antara lain :

- (**Formula Euler**) Misalkan G merupakan graf planar terhubung dengan e buah sisi dan v buah simpul, dan r merupakan jumlah daerah pada graf planar tersebut maka $r = e - v + 2$.
- Jika G merupakan graf planar terhubung dengan e buah sisi dan v buah simpul ($v \geq 3$) maka $e \leq 3v - 6$ (**ketaksamaan Euler**).
- Jika G merupakan graf planar terhubung dengan e buah sisi dan v buah simpul ($v \geq 3$) dan tidak memuat sirkuit dengan panjang 3 maka $e \leq 2v - 4$.

E. REPRESENTASI GRAF

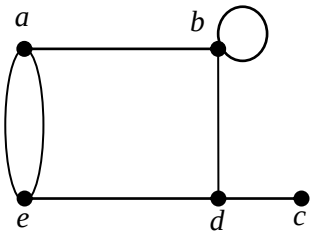
Bila graf akan diproses dengan program computer, maka graf harus direpresentasikan didalam memori. Ada tiga macam representasi graf yang biasa digunakan, yaitu dengan matriks ketetanggan, matriks bersisian, atau dengan senarai ketetanggan.

Matriks dapat digunakan untuk menyatakan suatu graf. Namun kesulitan utama merepresentasikan graf dalam matriks adalah keterbatasan matriks untuk mencakup semua informasi yang ada dalam graf.

1. Matriks Hubung (Adjacency Matrix)

Matriks ketetanggaan adalah representasi graf yang paling umum digunakan. Matriks Hubung (Adjacency Matrix) digunakan untuk merepresentasikan graf dengan cara menyatakannya dalam jumlah garis yang menghubungkan titik-titiknya. Banyaknya baris dan kolom matriks hubung sama dengan jumlah titik-titik pada graf. Karena banyaknya baris dan kolom matriks hubung sama dengan jumlah titik-titik pada graf, maka jelas bahwa matriks hubung selalu merupakan matriks yang simetris/persegi (berordo n x n).

Contoh 1:

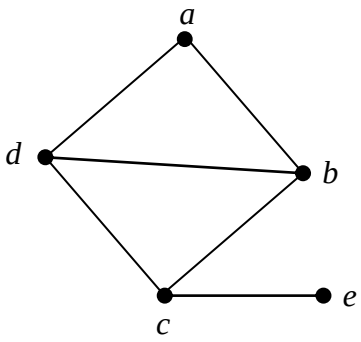


Graf G_1

	a	b	c	d	e
a	0	1	0	0	2
b	1	1	0	1	0
c	0	0	0	1	0
d	0	1	0	0	1
e	2	0	0	1	0

Gambar 1.21. Graf G_1 dan adjacency matriks graf G_1

Contoh 2:



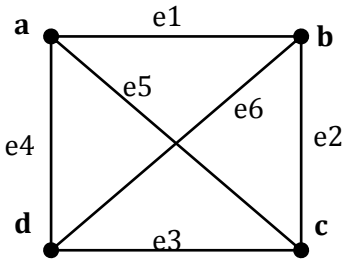
$$A_{G1} = \begin{array}{c} \\ a \\ b \\ c \\ d \\ e \end{array} \begin{array}{c} \begin{array}{ccccc} a & b & c & d & e \end{array} \\ \begin{bmatrix} 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{bmatrix} \end{array}$$

Gambar 1.22. Graf G_2 dan adjacency matriks graf G_2

Disini terdapat ***n*!** cara pengurutan nomor simpul, yang berarti ada ***n*!** matriks ketetanggaan yang berbeda untuk graf dengan ***n*** simpul.

2. Matriks Biner

Matriks biner merupakan representasi graf dimana setiap elemen dalam matriks menyatakan keterhubungan antara titik dalam graf dengan sisi-sisi yang terhubung pada titik tersebut. Matriks biner merupakan matriks *zero-one* (*0/1*) berordo *m* x *n* dimana banyaknya baris menyakan banyaknya titik, dan banyak kolom menyatakan banyaknya sisi-sisi pada graf.



	1	**2**	**3**	**4**	**5**	**6**
a	1	0	0	1	1	0
b	1	1	0	0	0	1
c	0	1	1	0	1	0
d	0	0	1	1	0	1

Gambar 1.22. Graf G_2 dan matriks biner dari graf G_2

F. LINTASAN DAN SIRKUIT EULER

Lintasan Euler ialah lintasan yang melalui masing-masing sisi di dalam graf tepat satu kali. Sirkuit Euler ialah sirkuit yang melewati masing-masing sisi tepat satu kali. Graf yang mempunyai sirkuit Euler disebut **graf Euler** (*Eulerian graph*). Graf yang mempunyai lintasan Euler dinamakan juga graf **semi-Euler** (*semi-Eulerian graph*).

Contoh 1.3.

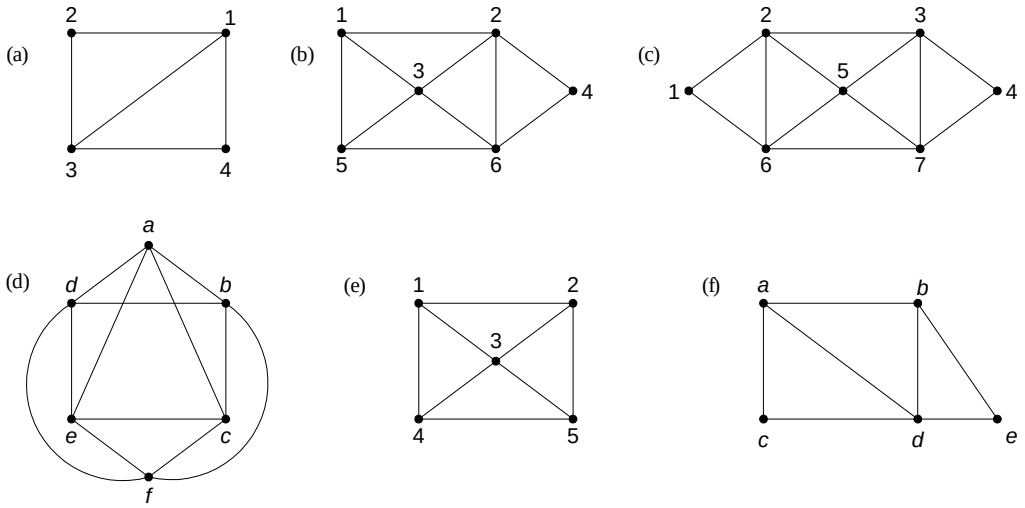
Lintasan Euler pada graf Gambar 1.21 (a) : 3, 1, 2, 3, 4, 1
Lintasan Euler pada graf Gambar 1.21 (b) : 1, 2, 4, 6, 2, 3, 6, 5, 1, 3
Sirkuit Euler pada graf Gambar 1.21 (c) : 1, 2, 3, 4, 7, 3, 5, 7, 6, 5, 2, 6, 1
Sirkuit Euler pada graf Gambar 1.21 (d) : *a, c, f, e, c, b, d, e, a, d, f, b, a*
Graf (e) dan (f) tidak mempunyai lintasan maupun sirkuit Euler

Teorema 1.1.

Graf tidak berarah memiliki lintasan Euler jika dan hanya jika terhubung dan memiliki dua buah simpul berderajat ganjil atau tidak ada simpul berderajat ganjil sama sekali.

Teorema 1.2.

Graf tidak berarah *G* adalah graf Euler (memiliki sirkuit Euler) jika dan hanya jika setiap simpul berderajat genap.

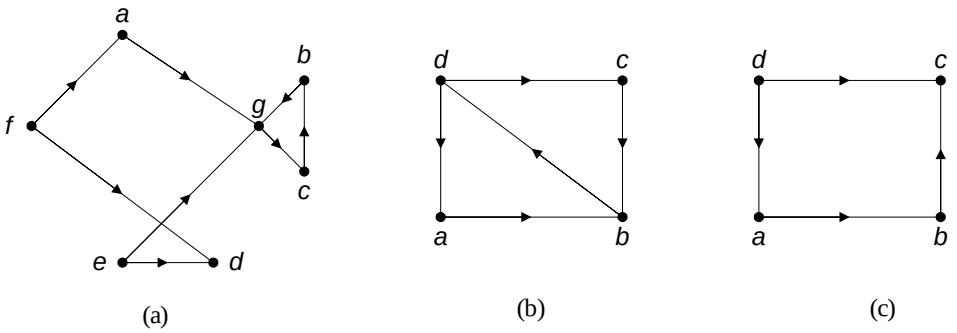


Gambar 1.23. (a) dan (b) graf semi-Euler, (c) dan (d) graf Euler
(e) dan (f) bukan graf semi-Euler atau graf Euler

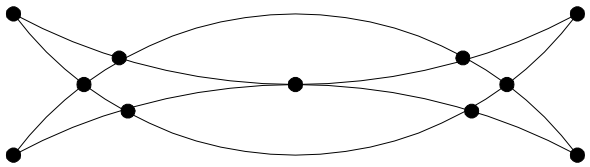
(Catatlah bahwa graf yang memiliki sirkuit Euler pasti mempunyai lintasan Euler, tetapi tidak sebaliknya)

Teorema 1.3.

Graf berarah *G* memiliki sirkuit Euler jika dan hanya jika *G* terhubung dan setiap simpul memiliki derajat-masuk dan derajat-keluar sama. *G* memiliki lintasan Euler jika dan hanya jika *G* terhubung dan setiap simpul memiliki derajat-masuk dan derajat-keluar sama kecuali dua simpul, yang pertama memiliki derajat-keluar satu lebih besar derajat-masuk, dan yang kedua memiliki derajat-masuk satu lebih besar dari derajat-keluar.



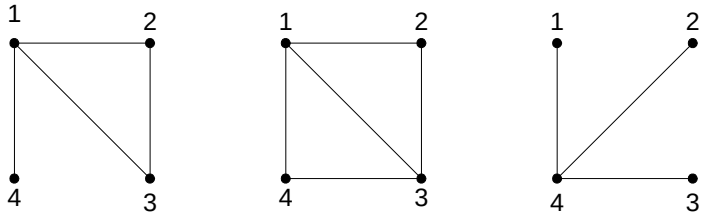
Gambar 1.24. (a) Graf berarah Euler (*a*, *g*, *c*, *b*, *g*, *e*, *d*, *f*, *a*)
(b) Graf berarah semi-Euler (*d*, *a*, *b*, *d*, *c*, *b*)
(c) Graf berarah bukan Euler maupun semi-Euler



Gambar 1.25. Graf Bulan sabit Muhammad

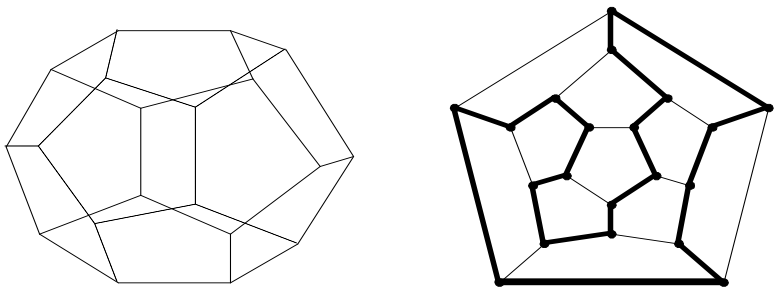
G. LINTASAN DAN SIRKUIT HAMILTON

Lintasan Hamilton ialah lintasan yang melalui tiap simpul di dalam graf tepat satu kali.
Sirkuit Hamilton ialah sirkuit yang melalui tiap simpul di dalam graf tepat satu kali, kecuali simpul asal (sekaligus simpul akhir) yang dilalui dua kali. Graf yang memiliki sirkuit Hamilton dinamakan **graf Hamilton**, sedangkan graf yang hanya memiliki lintasan Hamilton disebut **graf semi-Hamilton**.



(a) (b) (c)

Gambar 1.26.
(a) graf yang memiliki lintasan Hamilton (misal: 3, 2, 1, 4)
(b) graf yang memiliki sirkuit Hamilton (1, 2, 3, 4, 1)
(c) graf yang tidak memiliki lintasan maupun sirkuit Hamilton



(a) (b)

Gambar 1.25. (a) *Dodecahedron* Hamilton, dan (b) graf yang mengandung sirkuit Hamilton

TEOREMA 1.4.

Syarat cukup (jadi bukan syarat perlu) supaya graf sederhana *G* dengan *n* (≥ 3) buah simpul adalah graf Hamilton ialah bila derajat tiap simpul paling sedikit *n*/2 (yaitu, *d*(v) ≥ *n*/2 untuk setiap simpul *v* di *G*).

TEOREMA 1.5.

Setiap graf lengkap adalah graf Hamilton.

TEOREMA 1.6.

Di dalam graf lengkap *G* dengan *n* buah simpul (*n* ≥ 3), terdapat (*n* - 1)!/2 buah sirkuit Hamilton.

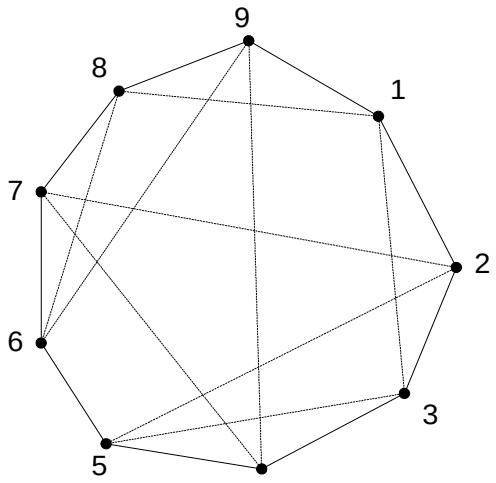
TEOREMA 1.7.

Di dalam graf lengkap *G* dengan *n* buah simpul (*n* ≥ 3 dan n ganjil), terdapat (*n* - 1)/2 buah sirkuit Hamilton yang saling lepas (tidak ada sisi yang beririsan). Jika *n* genap dan *n* ≥ 4, maka di dalam G terdapat (*n* - 2)/2 buah sirkuit Hamilton yang saling lepas.

Contoh 1.4.

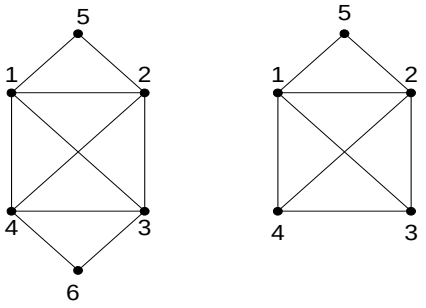
(Persoalan pengaturan tempat duduk). Sembilan anggota sebuah klub bertemu tiap hari untuk makan siang pada sebuah meja bundar. Mereka memutuskan duduk sedemikian sehingga setiap anggota mempunyai tetangga duduk berbeda pada setiap makan siang. Berapa hari pengaturan tersebut dapat dilaksanakan?

Jumlah pengaturan tempat duduk yang berbeda adalah (9 - 1)/2 = 4.



Gambar 1.26. Graf yang merepresentasikan persoalan pengaturan tempat duduk.

Beberapa graf dapat mengandung sirkuit Euler dan sirkuit Hamilton sekaligus, mengandung sirkuit Euler tetapi tidak mengandung sirkuit Hamilton, mengandung sirkuit Euler dan lintasan Hamilton, mengandung lintsan Euler maupun lintasan Hamilton, tidak mengandung lintasan Euler namun mengandung sirkuit Hamilton, dan sebagainya. Graf pada Gambar 1.27.(a) mengandung sirkuit Hamilton maunpun sirkuit Euler, sedangkan graf pada Gambar 1.27.(b) mengandung sirkuit Hamilton dan lintasan Euler (periksa!).



(a) (b)

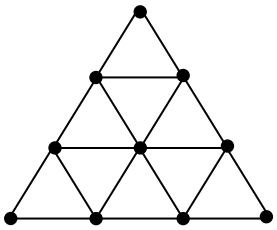
Gambar 1.27.
(a) Graf Hamilton sekaligus graf Euler
(b) Graf Hamilton sekaligus graf semi-Euler

LATIHAN

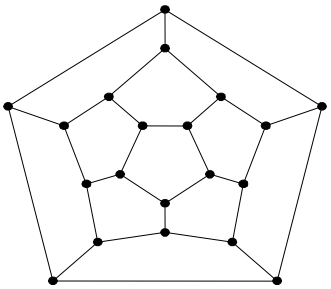
1. Gambarkan (bila dapat) graf dengan lima simpul yang masing-masing simpul berderajat berikut :
   a. 3, 3, 3, 3, 2
   b. 3, 3, 3, 3, 3
   c. 1, 2, 3, 4, 5

2. Tentukan mana diantara graf-graf berikut yang memiliki sirkuit Euler dan atau sirkuit Hamilton !

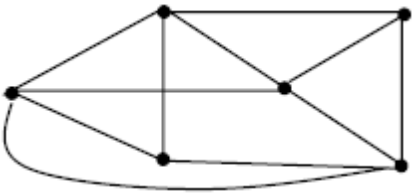
   a.



   b.



   c.

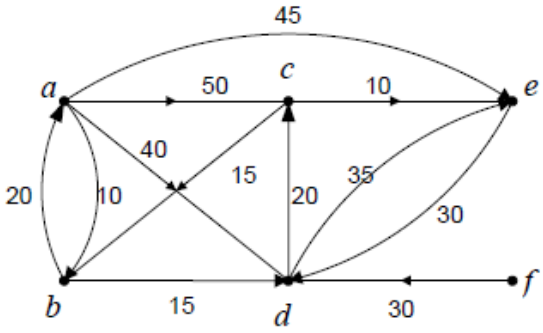


3.

Bab. 2 APLIKASI GRAF

A. LINTASAN TERPENDEK (*Shortest Path*)

Misalkan ***G*** merupakan graf berbobot (*weighted graph*), yaitu setiap sisi dari graf ***G*** memiliki bobot tertentu, seperti pada ilustrasi dibawah ini :



Gambar 2.1. *Graf berbobot*

Hal yang biasanya dilakukan adalah menentukan lintasan terpendek pada graf tersebut. Dengan kata lain, menentukan lintasan yang memiliki total bobot minimum.

Contoh 2.1 :

1. Menentukan jarak terpendek/waktu tempuh tersingkat/ongkos termurah antara dua buah kota
2. Menentukan waktu tersingkat pengiriman pesan (*message*) antara dua buah terminal pada jaringan komputer.

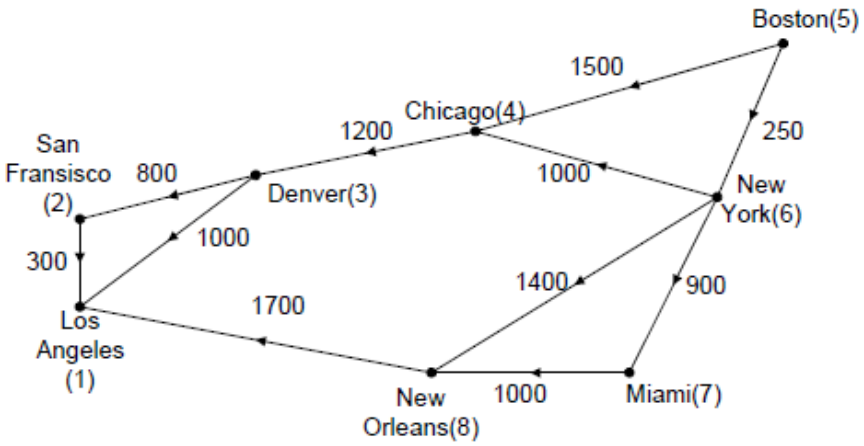
Beberapa jenis persoalan lintasan terpendek, antara lain:

a. Lintasan terpendek antara dua buah simpul tertentu.
b. Lintasan terpendek antara semua pasangan simpul.
c. Lintasan terpendek dari simpul tertentu ke semua simpul yang lain.
d. Lintasan terpendek antara dua buah simpul yang melalui beberapa simpul tertentu.

B. ALGORITMA LINTASAN TERPENDEK DIJKSTRA

Algoritma Dijkstra merupakan suatu algoritma yang digunakan untuk menentukan lintasan terpendek dari suatu simpul ke semua simpul lain. Untuk mempermudah dalam pemahaman Algoritma Dijkstra, berikut ini adalah graf dimana simpul-simpulnya merepresentasikan kota-kota di Amerika Serikat dan sisi dari graf tersebut merepresentasikan jarak antar dua kota (dalam kilometer).

Contoh 2.2 :



Gambar 2.2. Graf Kota-kota di Amerika

Dengan menggunakan Algoritma Dijkstra akan ditentukan jarak terpendek dari kota Boston ke kota-kota yang lainnya.

Lelaran	Simpul yang dipilih	Lintasan	*S*								*D*							
			1	2	3	4	5	6	7	8	1	2	3	4	5	6	7	8
Inisial	-	-	0	0	0	0	0	0	0	0	∞	∞	∞	1500	0	250	∞	∞
1	5	5	0	0	0	0	1	0	0	0	∞	∞	∞	1500	∞	250	∞	∞
2	6	5, 6	0	0	0	0	1	1	0	0	∞	∞	∞	1250	∞	250	1150	1650
3	7	5, 6, 7	0	0	0	0	1	1	1	0	∞	∞	∞	1250	∞	250	1150	1650
4	4	5, 6, 4	0	0	0	1	1	1	1	0	∞	∞	2450	1250	∞	250	1150	1650
5	8	5, 6, 8	0	0	0	1	1	1	1	1	3350	∞	2450	1250	∞	250	1150	1650
6	3	5, 6, 4, 3	0	0	1	1	1	1	1	1	3350	∞	2450	1250	∞	250	1150	1650
7	2	5, 6, 4, 3, 2	0	1	1	1	1	1	1	1	3350	3250	2450	1250	∞	250	1150	1650

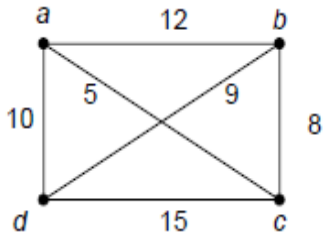
Jadi, lintasan terpendek dari:
5 ke 6 adalah 5, 6 dengan jarak = 250 km
5 ke 7 adalah 5, 6, 7 dengan jarak = 1150 km
5 ke 4 adalah 5, 6, 4 dengan jarak = 1250 km
5 ke 8 adalah 5, 6, 8 dengan jarak = 1650 km
5 ke 3 adalah 5, 6, 4, 3 dengan jarak = 2450 km
5 ke 2 adalah 5, 6, 4, 3, 2 dengan jarak = 3250 km
5 ke 1 adalah 5, 6, 8, 1 dengan jarak = 3350 km

C. PERSOALAN PERJALANAN PEDAGANG (*Travelling Salesperson Problem*)

Seperti halnya contoh pada (a), misalkan diberikan sejumlah kota dan jarak antar kota. Tentukan sirkuit terpendek yang harus dilalui oleh seorang pedagang bila pedagang itu berangkat dari sebuah kota asal dan ia harus menyinggahi setiap kota tepat satu kali dan kembali lagi ke kota asal keberangkatan. Ini merupakan masalah menentukan sirkuit Hamilton yang memiliki bobot minimum. Jumlah sirkuit Hamilton di dalam graf lengkap dengan n simpul: $(n - 1)!/2$.

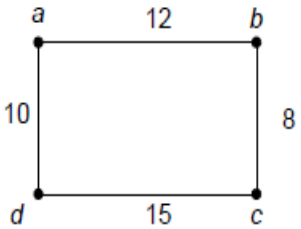
Contoh 2.3 (Munir, 2003) :

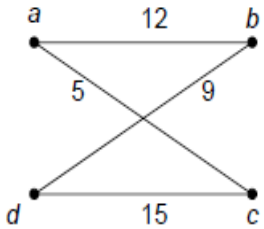
Jumlah sirkuit Hamilton di dalam graf lengkap dengan *n* simpul: $(n - 1)!/2$

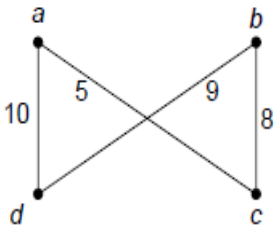


Graf di atas memiliki $(4 - 1)!/2 = 3$ sirkuit Hamilton, yaitu:

- I_1 = (*a*, *b*, *c*, *d*, *a*) atau (*a*, *d*, *c*, *b*, *a*) ==> panjang = 10 + 12 + 8 + 15 = 45
- I_2 = (*a*, *c*, *d*, *b*, *a*) atau (*a*, *b*, *d*, *c*, *a*) ==> panjang = 12 + 5 + 9 + 15 = 41
- I_3 = (*a*, *c*, *b*, *d*, a) atau (*a*, *d*, *b*, *c*, *a*) ==> panjang = 10 + 5 + 9 + 8 = 32





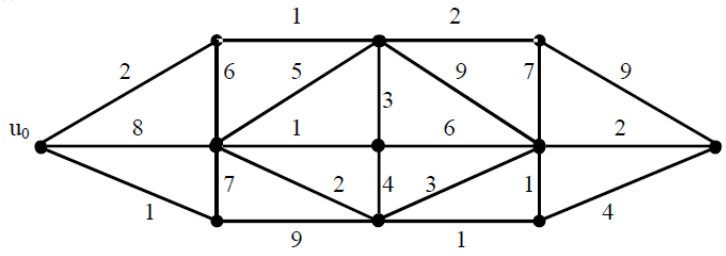


Jadi, sirkuit Hamilton terpendek adalah $\boldsymbol{I}_3$ = (*a*, *c*, *b*, *d*, *a*) atau (*a*, *d*, *b*, *c*, *a*) dengan panjang sirkuit = 10 + 5 + 9 + 8 = 32.

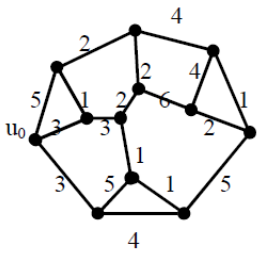
LATIHAN

1. Untuk setiap graf di bawah ini, gunakan algoritma Djikstra untuk menemukan lintasan terpendek dari u_0 ke setiap simpul lainnya. Anda diharuskan untuk menggambarkan graf per tahap, sehingga masing-masing pelabelan dapat digambarkan dengan jelas.

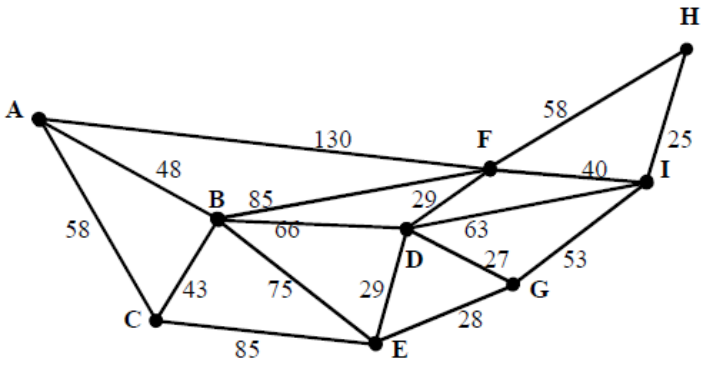
(i)



(ii)



2. Sebuah paket berisi barang-barang asli di Los Angeles dapat dikirimkan ke Boston melalui beberapa rute berbeda, yang ditunjukkan di bawah ini. Bobot tiap busur menyatakan biaya transportasi atau pengiriman paket antar kota yang berhubungan. Gunakan algoritma Djikstra untuk menemukan rute terpendek yang dapat meminimumkan biaya dari Los Angeles ke Boston!



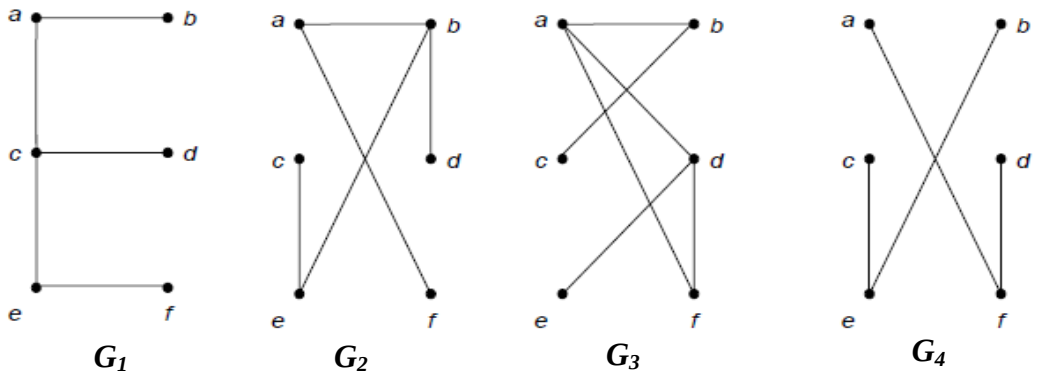
A	=	Portland	F	=	Chicago
B	=	Salt Lake City	G	=	Memphis
C	=	Los Angeles	H	=	Boston
D	=	Kansas City	I	=	Washington
E	=	Dallas			

3.

Bab.3 POHON

A. DEFINISI POHON

Pohon (*tree*) merupakan salah satu bentuk khusus dari struktur suatu graf. Misalkan A merupakan sebuah himpunan berhingga simpul (*vertex*) pada suatu graf ***G*** yang terhubung. Untuk setiap pasangan simpul di A dapat ditentukan suatu lintasan yang menghubungkan pasangan simpul tersebut. Suatu graf terhubung yang setiap pasangan simpulnya hanya dapat dihubungkan oleh suatu lintasan tertentu, maka graf tersebut dinamakan pohon (*tree*). Dengan kata lain, pohon (*tree*) merupakan graf tak-berarah yang terhubung dan tidak memiliki sirkuit.



Gambar 3.1. G_1 dan G_2 adalah pohon, sedangkan G_3 dan G_4 bukan pohon

Hutan (*forest*) merupakan kumpulan pohon yang saling lepas. Dengan kata lain, hutan merupakan graf tidak terhubung yang tidak mengandung sirkuit. Setiap komponen di dalam graf terhubung tersebut adalah pohon. Pada gambar 2.1, G_4 merupakan salah satu contoh hutan, yaitu hutan yang terdiri dari dua pohon.

Sifat-sifat Pohon

Teorema 3.1.

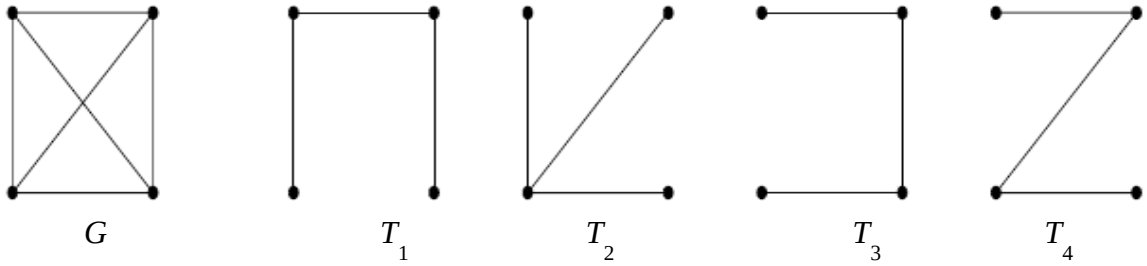
Misalkan $G = (V, E)$ adalah graf tak-berarah sederhana dan jumlah simpulnya n. Maka, semua pernyataan di bawah ini adalah ekivalen:

1. G adalah pohon.
2. Setiap pasang simpul di dalam G terhubung dengan lintasan tunggal.
3. G terhubung dan memiliki $m = n - 1$ buah sisi.
4. G tidak mengandung sirkuit dan memiliki $m = n - 1$ buah sisi.
5. G tidak mengandung sirkuit dan penambahan satu sisi pada graf hanya akan membuat satu sirkuit.
6. G terhubung dan semua sisinya adalah jembatan.

3.1 Pohon Merentang Minimum (*Minimum Spanning Tree*)

Spanning Tree dari suatu graf terhubung merupakan subgraf merentang yang berupa pohon. Pohon merentang diperoleh dengan cara menghilangkan sirkuit di dalam graf tersebut.

Contoh *spanning tree* dari suatu graf terhubung (**Munir, 2003**) :



Gambar 3.2. Spanning Tree

Perhatikan graf di atas :

Terlihat bahwa T_1, T_2, T_3, T_4 merupakan s*panning tree* dari graf G. Perlu diperhatikan bahwa setiap graf terhubung berbobot paling sedikit mempunyai satu buah *spanning tree*. Pohon rentang yang memiliki bobot minimum dinamakan pohon merentang minimum (*minimum spanning tree*). Dalam kehidupan nyata, salah satu contoh aplikasi s*panning tree* adalah menentukan rangkaian jalan dengan jarak total seminimum mungkin yang menghubungkan semua kota sehingga setiap kota tetap terhubung satu sama lain.

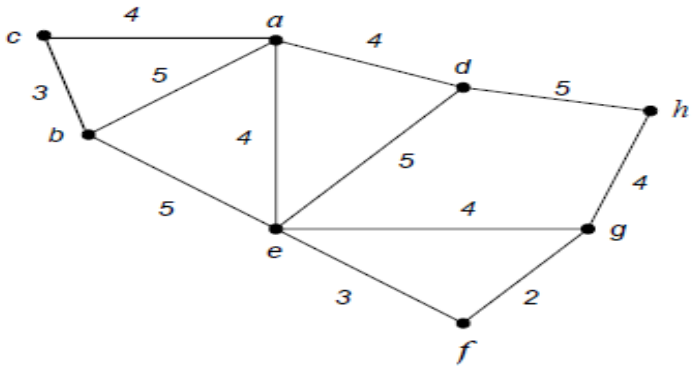
Dalam menentukan suatu *minimum spanning tree* dari suatu graf terhubung, kita dapat menentukannya dengan mengunakan dua cara yaitu algoritma Prim dan algoritma Kruskal. Algoritma Prim memiliki langkah-langkah sebagai berikut :

1. Pilih sisi dari graf *G* yang berbobot minimum, masukkan ke dalam *T*.
2. Pilih sisi (*u*, *v*) dalam G yang mempunyai bobot minimum dan bersisian dengan simpul di *T*, dengan syarat sisi tersebut tidak membentuk sirkuit di *T*. Masukkan (*u*, *v*) ke dalam *T*.
3. Ulangi langkah 2 sebanyak $n - 2$ kali.

Jumlah langkah seluruhnya dalam algoritma Prim adalah sebanyak jumlah sisi di dalam *spanning tree* dengan *n* buah simpul, yaitu $(n - 1)$ buah.

Contoh 3.1:

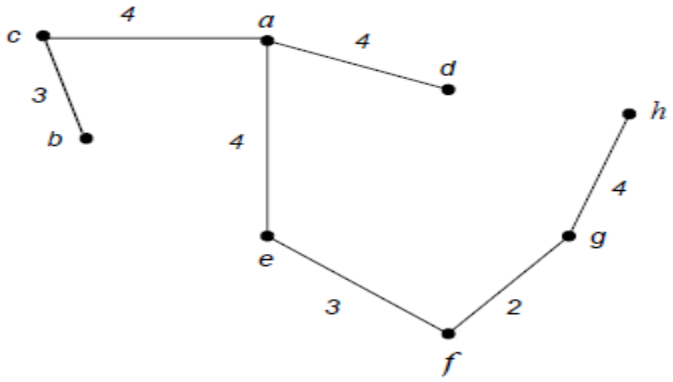
Tentukan *minimum spanning tree* dari graf dibawah ini :



Jawab :

- Pilih sisi *fg* sehingga kita mempunyai T ($\{f, g\}$, fg)
- Langkah selanjutnya dapat dipilih sisi *ef* karena sisi tersebut berbobot minimum yang bersisian dengan simpul *f* .
- Selanjutnya pilih sisi *ae* atau *gh* karena sisi tersebut berbobot minimum yang bersisian dengan simpul pada *T*, yaitu *e* dan *g*.

Jika proses ini dilanjutkan terus maka akan diperoleh *minimum spanning tree* seperti dibawah ini :

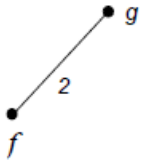


Terlihat bahwa *spanning tree* tersebut mempunyai total bobot :
2 + 3 + 4 + 4 + 4 + 4 + 3 = 24.

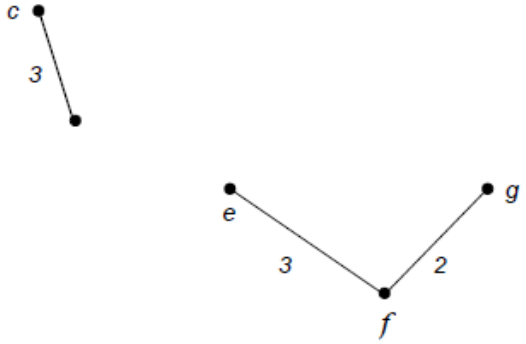
Langkah-langkah dalam algoritma Kruskal agak berbeda dengan algoritma Prim. Pada algoritma Kruskal, semua sisi dengan bobot yang minimal dimasukan kedalam *T* secara berurutan.

Langkah-langkah dalam menentukan *minimum spanning tree* dengan algoritma Kruskal adalah sebagai berikut :

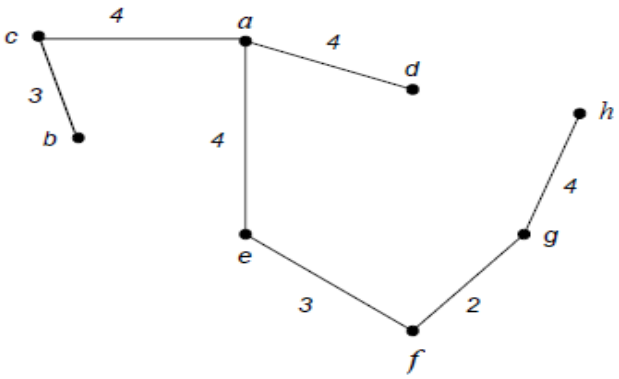
Langkah I : T berbentuk seperti pohon berikut



Langkah II : memasukan sisi-sisi yang berbobot 3 kedalam T sehingga berbentuk

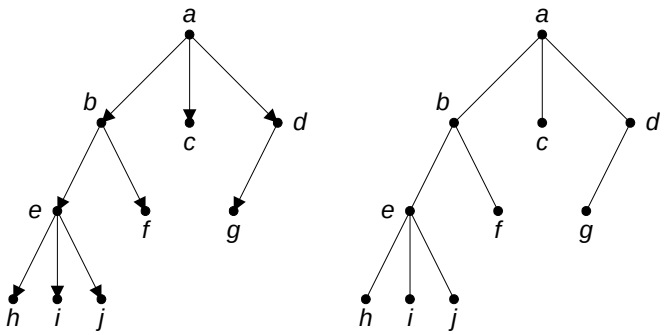


Langkah III : memasukan sisi-sisi yang berbobot 4 kedalam sehingga akhirnya diperoleh *minimum spanning tree* berikut :



3.2 Pohon Berakar

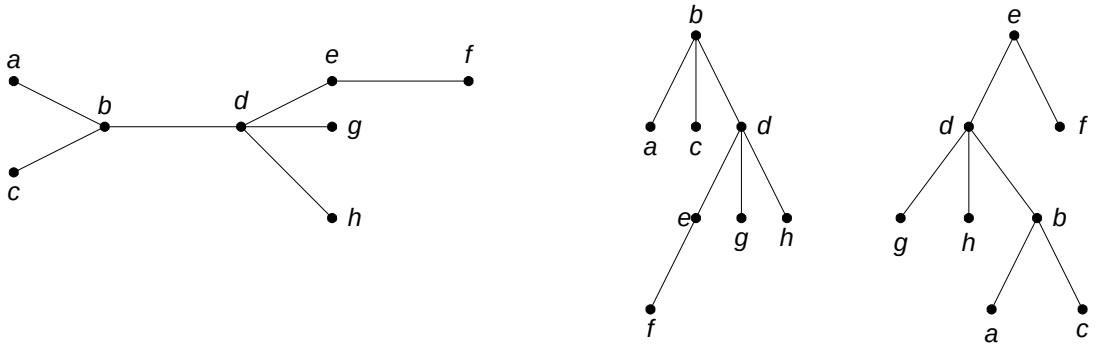
Pada suatu pohon, yang sisi-sisinya diberi arah sehingga menyerupai graf berarah, maka simpul yang terhubung dengan semua simpul pada pohon tersebut dinamakan **akar**. Suatu pohon yang satu buah simpulnya diperlakukan sebagai akar maka pohon tersebut dinamakan pohon berakar (*rooted tree*). Simpul yang berlaku sebagai akar mempunyai derajat masuk sama dengan nol. Sementara itu, simpul yang lain pada pohon itu memiliki derajat masuk sama dengan satu. Pada suatu pohon berakar, Simpul yang memiliki derajat keluar sama dengan nol dinamakan **daun**.



Gambar 3.3 : Pohon Berakar (**Munir, 2003**)

Pada pohon berakar diatas :

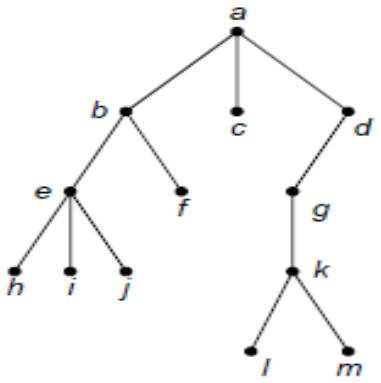
- *a* merupakan akar
- *c, f, g, h, i,* dan *j* merupakan daun



Gambar 3.4 : Pohon dan dua buah pohon berakar yang dihasilkan dari pemilihan dua simpul berbeda sebagaia akar

B. TERMINOLOGI PADA POHON BERAKAR

Perhatikan graf pohon berakar berikut :



Gambar 3.5. *Pohon berakar untuk terminologi*.

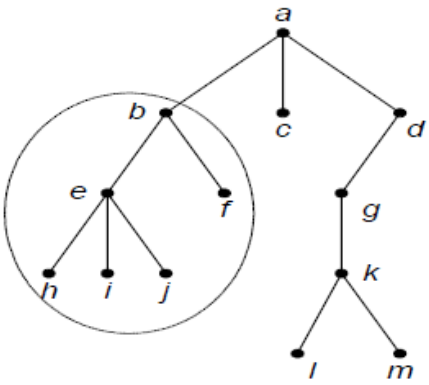
a. Anak (*child* atau *children*) dan Orangtua (*parent*)

b, *c*, dan *d* adalah anak-anak simpul *a*,
a adalah orangtua dari anak-anak itu

b. Lintasan (*path*)

Lintasan dari *a* ke *h* adalah *a*, *b*, *e*, *h*. dengan pnjang lintasannya adalah 3.
f adalah saudara kandung *e*, tetapi, *g* bukan saudara kandung *e*, karena orangtua mereka berbeda.

c. *Subtree*



c. Derajat (*degree*)

Derajat sebuah simpul adalah jumlah anak pada simpul tersebut.

Contoh :

Simpul yang berderajat 0 adalah simpul *c, f, h, I, j, l,* dan *m*.
Simpul yang berderajat 1 adalah simpul *d* dan g.
Simpul yang berderajat 2 adalah simpul *b* dan *k*.
Simpul yang berderajat 3 adalah simpul *a* dan *e*.

Jadi, derajat yang dimaksudkan di sini adalah derajat-keluar.
Derajat maksimum dari semua simpul merupakan derajat pohon itu sendiri. Pohon di atas berderajat 3

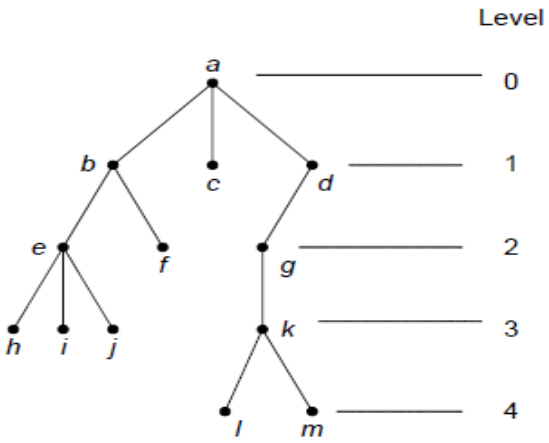
d. Daun (*leaf*)

Simpul yang berderajat nol (atau tidak mempunyai anak) disebut daun. Simpul *h, i, j, f, c, l,* dan *m* adalah daun.

e. Simpul Dalam (*internal nodes*)

Simpul yang mempunyai anak disebut simpul dalam. Simpul *b, d, e, g,* dan *k* adalah simpul dalam.

f. Aras (*level*) atau Tingkat



g. Tinggi (*height*) atau Kedalaman (*depth*)

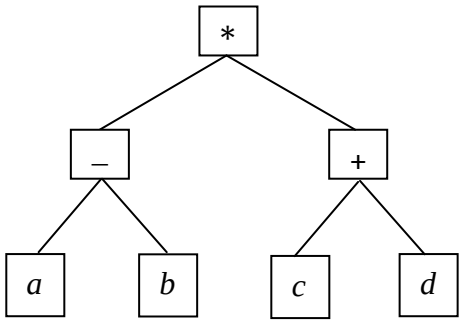
Aras maksimum dari suatu pohon disebut tinggi atau kedalaman pohon tersebut. Pohon di atas mempunyai tinggi 4.

Pohon berakar yang urutan anak-anaknya penting (diperhatiakn) dinamakan **pohon terurut** (*ordered tree*). Sedangkan pohon berakar yang setiap simpul cabangnya mempunyai paling banyak *n* buah anak disebut pohon ***n-ary*.** Jika *n* = 2, pohonnnya disebut pohon biner (*binary tree*).

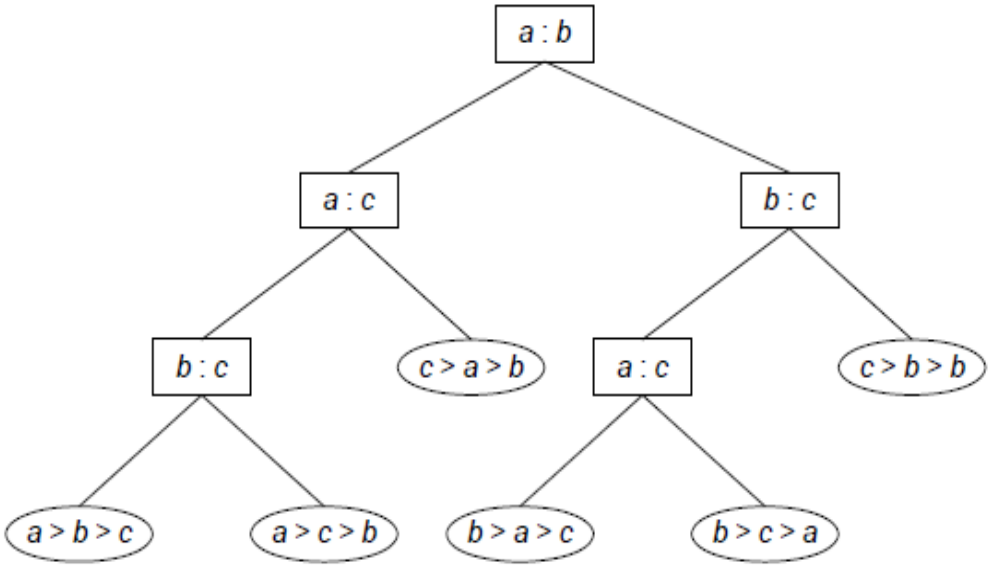
Berikut adalah beberapa contoh pohon biner :

1. Pohon Ekspresi

   Ekspresi aritmetika $(a - b)*(c + d)$ dapat dinyatakan dalam suatu pohon biner, dimana peubah sebagai daun dan operator aritmetika sebagai simpul dalam dan akar.



2. Pohon keputusan (**Munir, 2004**)

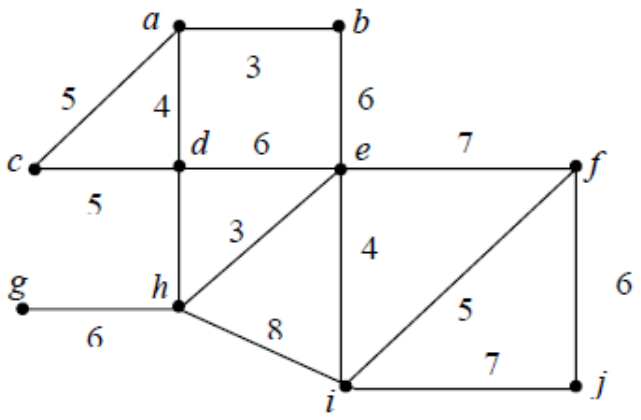


C. LATIHAN

1. Buat sketsa graf biner (pohon ekspresi) yang merepresentasikan ekpresi :
   a. p / (q – r)*(s + t)
   b. (p + q) / r – (s + t * u)

2. Tentukan hasil dari pohon ekspresi pada soal no. 1 dalam bentuk *preorder*, *inorder*, dan *postorder* !

3. Pada graf dibawah ini, himpunan simpul mendefinisikan himpunan desa pada suatu kecamatan. Dalam rangka pembuatan jalan antar desa dibuatlah anggaran pembiayaan seperti tertulis sebagai bobot (dalam satuan juta rupiah) setiap sisi. Tentukan biaya minimum yang harus disiapkan dalam pembangunan jalan antar desa tersebut sehingga setiap desa pada kecamatan tersebut terhubung (ingat definisi terhubung pada suatu graf).



DAFTAR PUSTAKA

Munir, Rinaldi. *Matematika Diskrit.* Bandung: Informatika, 2005.

Siang, Jong Jek. *Matematika Diskrit dan Aplikasinya pada Ilmu komputer.* Yogyakarta: Andi Offset, 2004.

Wibisono, Samuel. *Matematika Diskrit.* Yogyakarta: Graha Ilmu, 2008.