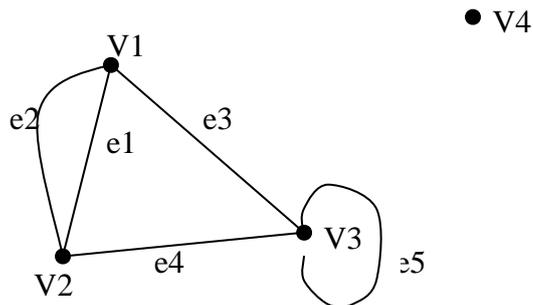


## GRAF

Graf  $G(V,E)$  didefinisikan sebagai pasangan himpunan  $(V,E)$ , dengan  $V$  adalah himpunan berhingga dan tidak kosong dari simpul-simpul (verteks atau node). Dan  $E$  adalah himpunan berhingga dari busur (vertices atau edge).

Contoh :



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

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

$$E = \{(v_1, v_2), (v_1, v_2), (v_1, v_3), (v_2, v_3), (v_3, v_3)\}$$

### ISTILAH GRAF

Gelang (*loop*) yaitu busur yang berawal dan berakhir pada simpul yang sama.

Busur ganda (*multiple edge*) yaitu suatu busur yang menghubungkan simpul yang sama

Ketetanggaan (*adjacent*) : dua buah simpul dikatakan bertetangga, jika terdapat busur  $e$  dengan ujung awal dan akhir adalah  $v_1$  dan  $v_2$ . ( $e=(v_1, v_2)$ )

Kehadiran (*incident*) : suatu busur dikatakan hadir pada suatu simpul, jika busur tersebut menghubungkan simpul tersebut.

Derajat (*degree*) yaitu banyaknya busur yang ada pada suatu simpul  $v$ . ( $d(v)$ )

Simpul terminal adalah simpul yang berderajat 1

Simpul terpencil adalah simpul yang berderajat 0, dan tidak bertetangga dengan simpul lain.

$n = |V| =$  kardinalitas simpul

$m = |E| =$  kardinalitas busur

## MACAM GRAF

Graf dapat dikelompokkan menjadi beberapa kategori, yaitu :

### 1. Graf Kosong (*Null graph*)

Graf kosong adalah graf dengan himpunan busur merupakan himpunan kosong.

Contoh :



$N_4$

### 2. Graf Sederhana (*simple graph*)

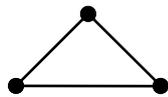
Graf sederhana adalah graf yang tidak mempunyai gelang (*loop*) dan/atau sisi ganda (*multiple edge*)

Terdapat beberapa macam graf sederhana, yaitu :

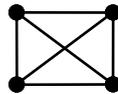
#### a) Graf lengkap (*complete graph*)

Graf lengkap adalah graf dengan setiap pasang simpulnya saling bertetangga, dengan jumlah busur  $(m) = (n \cdot (n-1))/2$ .

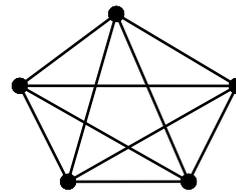
Contoh :



$K_3$



$K_4$

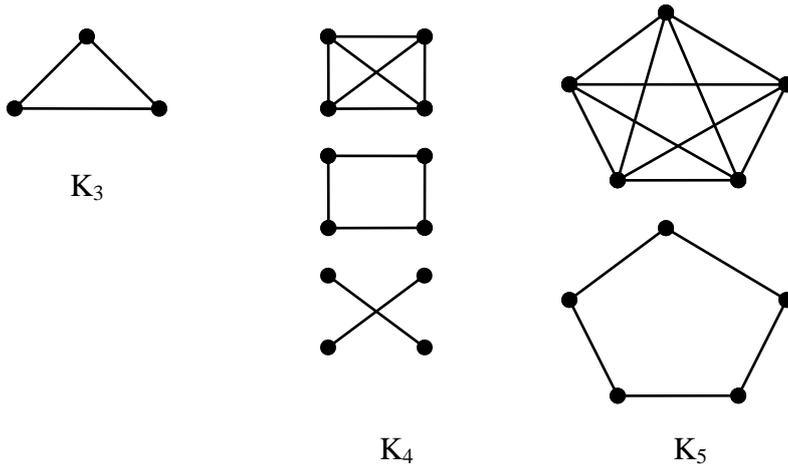


$K_5$

#### b) Graf teratur (*regular graph*)

Graf teratur adalah graf yang semua simpul dalam graf tersebut berderajat sama, dengan jumlah busur  $(m) = (n \cdot r)/2$ , dan  $r$  adalah nilai derajat simpul.

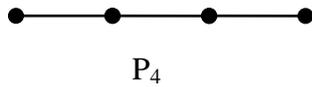
Contoh :



c) Graf Lintasan (*paths*)

Graf lintasan adalah graf yang bentuknya menyerupai garis lurus,  $m=n-1$ .

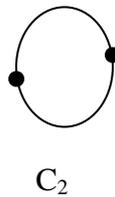
Contoh :



d) Graf lingkaran (*Cycles*)

Graf lingkaran adalah graf yang bentuknya menyerupai lingkaran, dengan  $m=n$   
Dinotasikan dengan  $C_n$

Contoh :

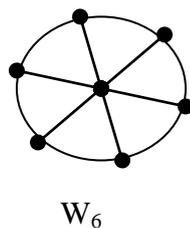


e) Graf Roda (*Wheels*)

Graf Roda adalah graf lingkaran yang setiap simpulnya dihubungkan dengan simpul di tengah lingkaran.

Dinotasikan dengan  $W_n$

Contoh :

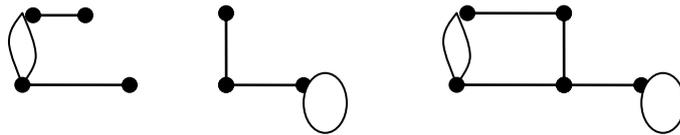


3. Graf tidak sederhana (*unsimple graph*)

Graf tidak sederhana adalah graf yang mempunyai gelang (*loop*) dan/atau sisi ganda (*multiple edge*)

1. Graf Ganda (*Multigraph*) adalah graf yang mempunyai sisi ganda
2. Graf Semu (*Pseudograph*) adalah graf yang mempunyai gelang / *loop*

Contoh :

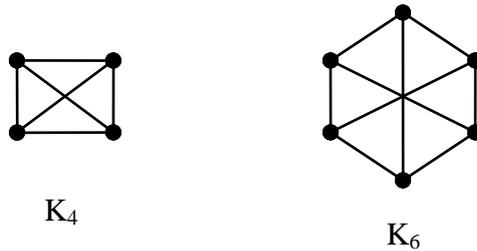


4. Graf dengan kekhususan tertentu

1. Graf Petersen

Graf Petersen adalah graf teratur yang mempunyai derajat simpul 3 pada semua simpulnya.

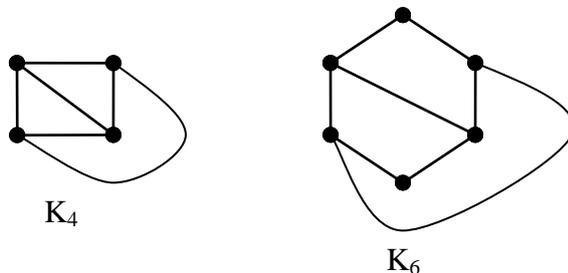
Contoh :



2. Graf Planar

Graf Planar adalah graf yang dapat digambarkan pada suatu bidang datar dengan busur-busur yang tidak saling memotong.

Contoh :



3. Graf Bipartite

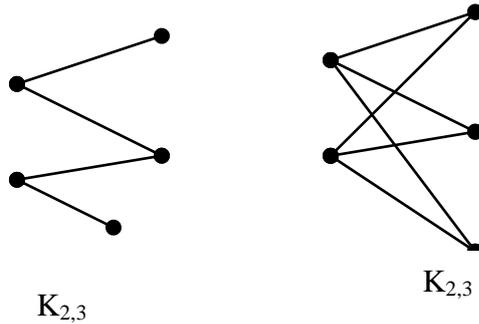
Graf bipartite adalah graf  $G$  dengan himpunan simpulnya dapat dibedakan dan dipisahkan menjadi dua himpunan bagian, yaitu  $V_1$  dan  $V_2$ , sedemikian sehingga

setiap busur di  $G$  menghubungkan ke satu simpul di  $V_1$  ke satu simpul di  $V_2$ , dengan kata lain setiap pasang simpul di  $V_1$  tidak bertetangga, dan setiap pasang simpul di  $V_2$  juga tidak bertetangga.

Dinotasikan Sebagai  $G(V_1, V_2) \leftrightarrow K_{n,m}$

Jika setiap simpul di  $V_1$  bertetangga dengan semua simpul di  $V_2$ , maka disebut graf bipartite lengkap (*complete bipartite graph*)

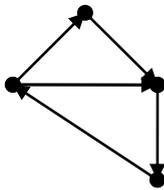
Contoh :



4. Graf Berarah (*Directed graph*)

Graf berarah adalah graf yang semua busurnya mempunyai arah.

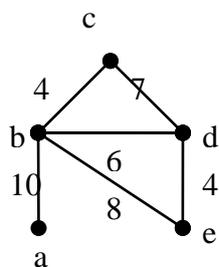
Contoh :



5. Graf Berbobot (*Weighted graph*)

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

Contoh :



## Graf Tidak Sederhana

1. Graf Ganda (multigraph)  
Graf yang mempunyai sisi ganda
2. Graf Semu (pseudograph)  
Graf yang mempunyai gelang/loop

## Representasi Graf dalam bentuk Matriks

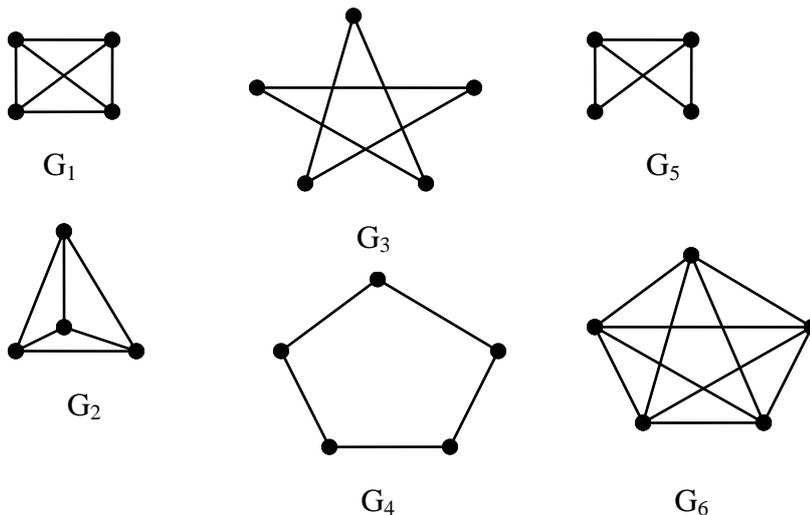
1. Matriks Adjacent  
 $M_{\text{simpul} \times \text{simpul}}$
2. Matriks Incident  
 $I_{\text{simpul} \times \text{busur}}$

## Isomorfisma

Isomorfik adalah dua buah benda yang sama tetapi secara geometri bersifat berbeda.

Dua buah graf  $G_1$  dan  $G_2$  dikatakan mempunyai isomorfisma (isomorfiks), jika terdapat pemetaan satu-satu antara simpul-simpul di  $G_1$  dan simpul-simpul di  $G_2$ , dan dipenuhinya syarat : (1) jumlah busur masing-masing graf sama, (2) jumlah node masing-masing graf sama, (3) terdapat kesesuaian antara busur-busur di dalam kedua graf tersebut

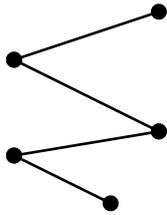
Contoh :



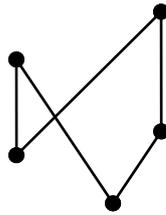
## Graf Komplemen

Komplemen graf ( $G^c$ ) adalah suatu graf sederhana dengan simpul yang sama dengan himpunan simpul graf  $G$ , dan memenuhi syarat bahwa dua buah simpul di  $G^c$  bertetangga, jika dan hanya jika kedua simpul tidak bertetangga di  $G$ , sehingga  $G^c$  dan  $G$  akan membentuk graf lengkap

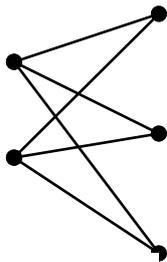
Contoh :



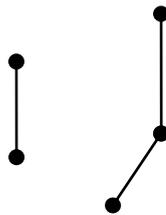
$K_{2,3}$



Komplemen  $K_{2,3}$



$K_{2,3}$



Komplemen  $K_{2,3}$

## LINTASAN

Sederetan busur atau simpul atau busur dan simpul secara berselang seling yang membentuk sambungan yang tidak putus pada graf  $G$ .

### Macam Lintasan

#### 1. Lintasan Sederhana

Lintasan yang setiap simpul yang dilalui berbeda

#### 2. Lintasan Tertutup

Lintasan yang berawal dan berakhir di simpul yang sama

#### 3. Lintasan Terbuka

Lintasan yang berawal dan berakhir di simpul berbeda

Jalan (*walk*) adalah sederetan busur-busur yang membentuk sambungan yang tidak putus di  $G$

Lintasan (*path*) adalah sederetan busur dan simpul berselang-seling dari simpul awal  $v_0$  ke simpul akhir  $v_n$ , sedemikian sehingga  $e_1 = (v_0, v_1)$ ,  $e_2 = (v_1, v_2)$ , ...,  $e_n = (v_{n-1}, v_n)$ , adalah busur-busur dalam graf.

Panjang suatu lintasan adalah banyaknya busur-busur pada jalan tersebut.

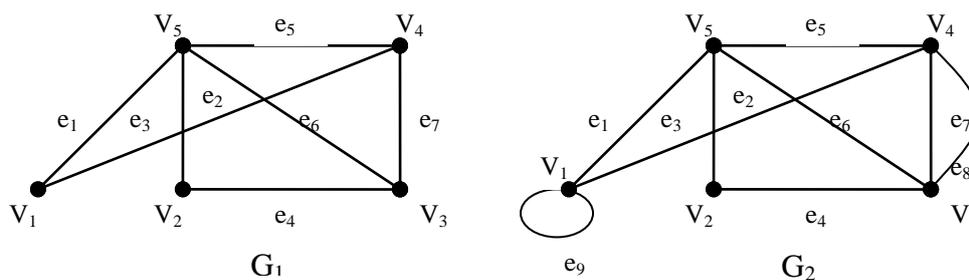
Penulisan lintasan pada graf sederhana, hanya menuliskan simpul-simpul yang dilalui, sedangkan pada graf dengan sisi ganda, harus menuliskan urutan busur dan simpul secara berselang-seling sesuai dengan jalan yang dilalui.

Lintasan sederhana adalah lintasan yang setiap simpul yang dilalui berbeda (atau setiap busur yang dilalui hanya satu kali).

Lintasan tertutup (*closed path*) adalah lintasan yang berawal dan berakhir pada simpul yang sama.

Lintasan terbuka (*open path*) adalah lintasan yang berawal dan berakhir pada simpul yang berbeda.

Contoh :



Jalan antara  $v_1$  dan  $v_4$  di  $G_1$ :  $e_3$  atau  $e_1-e_2-e_4-e_7$  atau  $e_1-e_6-e_7$

Lintasan  $v_1$  dan  $v_4$  di  $G_1$ :  $e_3$  atau  $e_1-e_2-e_4-e_7$  atau  $e_1-e_6-e_7$

Lintasan  $v_1$  dan  $v_4$  di  $G_2$ :  $e_3$  atau  $e_1-v_5-e_2-v_2-e_4-e_2-v_3-e_7$  atau  $e_1-v_5-e_6-v_3-e_7$

Lintasan sederhana :  $v_1-v_5-v_3-v_4$

Lintasan tertutup :  $v_1-v_5-v_2-v_3-v_4-v_1$

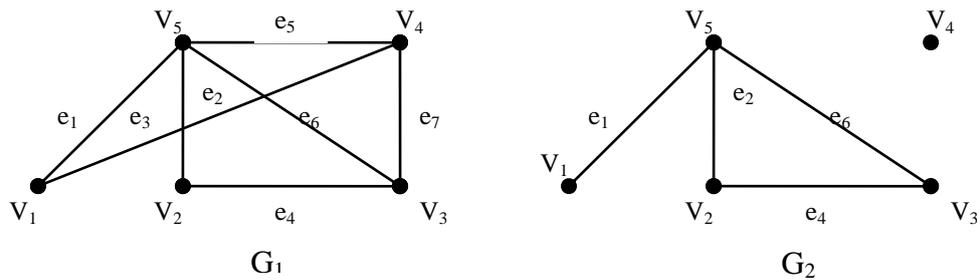
Lintasan terbuka :  $v_1-v_5-v_2-v_3-v_4$

### Definisi Keterhubungan dalam graf :

Suatu graf  $G$  disebut terhubung (*connected*) apabila setiap pasang simpul sembarang, misal:  $u$  dan  $v$ , di  $G$  mempunyai suatu lintasan dari simpul  $u$  menuju simpul  $v$ .

Lintasan tertutup adalah lintasan dengan simpul awal dan simpul akhir lintasan sama ( $u=v$ ).

Contoh:

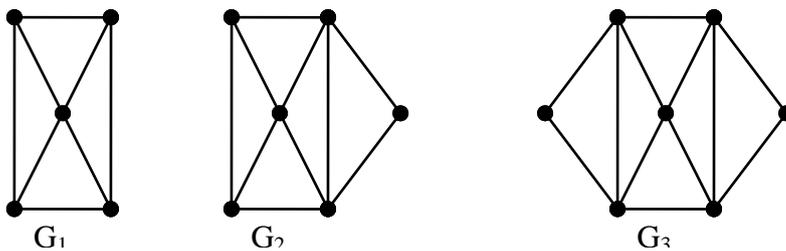


Graf  $G_1$  adalah graf terhubung, sedangkan  $G_2$  merupakan graf tidak terhubung (*disconnected graf*)

### Graf Euler

Lintasan Euler adalah lintasan yang melalui masing-masing busur dalam graf tepat satu kali. Bila lintasan tersebut kembali ke simpul asal, membentuk lintasan tertutup, maka lintasan itu dinamakan sirkuit Euler.

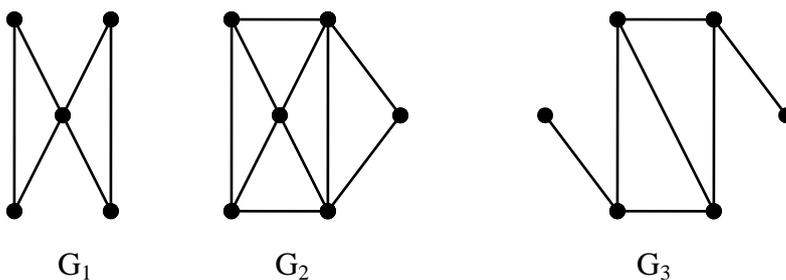
Graf yang mempunyai sirkuit Euler dinamakan graf Euler, dan graf yang mempunyai lintasan Euler dinamakan graf semi-Euler.



### Graf Hamilton

Lintasan Hamilton adalah lintasan yang melalui setiap simpul di dalam graf tepat satu kali. Bila lintasan itu kembali ke simpul awal dan membentuk lintasan tertutup maka disebut sirkuit Hamilton

Graf yang memiliki sirkuit Hamilton dinamakan graf Hamilton, sedangkan graf yang memiliki lintasan Hamilton disebut graf semi Hamilton.



# APLIKASI GRAF

## MENCARI JARAK TERPENDEK (SHORTEST PATH)

Banyak permasalahan transportasi dimodelkan sebagai bentuk graf, yaitu graf yang mempunyai berat ( weighted graf). Kota digambarkan sebagai simpul, hubungan antar kota digambarkan sebagai busur, dan jarak antar kota sebagai berat dari gambar.

### Algoritma Dijkstra

Terdapat beberapa algoritma untuk mencari jalur terpendek, diantaranya adalah yang dikemukakan oleh E. Dijkstra pada tahun 1959. Algoritma ini digunakan untuk mencari jalur terpendek yang menghubungkan dua buah simpul dalam suatu graf, sehingga sering disebut single pair's shortest path. Langkah-langkah yang digunakan sebagai berikut :

a) Iterasi pertama, simpul awal =  $v_i$ , beri label untuk simpul yang lain, yaitu :

1. Jika simpul  $v_j$  dengan  $j \in \{v_1, v_2, v_3, \dots, v_n\}$  terhubung dengan  $v_i$  oleh suatu busur  $(v_i, v_j)$ , maka label untuk  $v_j = d(v_j) =$  panjang busur tersebut
2. Jika  $v_j$  tidak terhubung dengan  $v_i$  maka  $d(v_i) = \infty$

b) Iterasi kedua, pilih simpul dengan label minimum dari hasil iterasi pertama sebagai simpul awal, label untuk setiap simpul lain ditentukan dengan membandingkan nilai labelnya dengan jumlah nilai label simpul awal ditambahkan dengan panjang busur antara simpul awal dengan simpul tersebut, atau :

$$d'(v_k) = \min\{d(v_k), d(v_j) + a(v_j, v_k)\}$$

dengan  $v_j$  : simpul awal

$v_k$  : simpul yang dicari labelnya

$d'(v_k)$  : nilai label yang baru

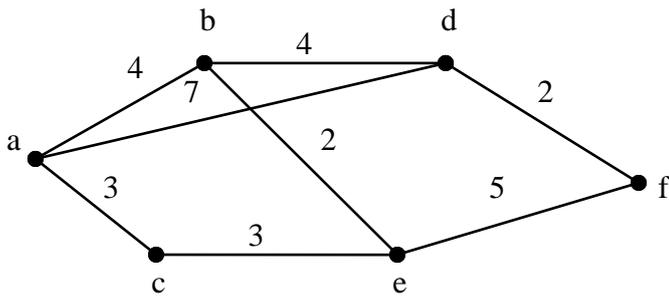
$d(v_k)$  : nilai label hasil iterasi sebelumnya

$d(v_j)$  : nilai label hasil iterasi sebelumnya

$a(v_j, v_k)$ : panjang busur

c) Ulangi iterasi kedua sampai simpul tujuan dipilih sebagai simpul awal.

Contoh :



Penyelesaian :

### Iterasi 1

Posisi awal di simpul a.

$$d(a) = 0, d(b) = 4, d(c) = 3, d(d) = 7, d(e) = \infty, d(f) = \infty,$$

Minimum di c, maka jalur yang didapat (a,c)

### Iterasi 2

Posisi awal di c

$$d(b) = \min \{d(b), d(c) + a(c, b)\} = \min \{4, 3 + \infty\} = 4$$

$$d(d) = \min \{d(d), d(c) + a(c, d)\} = \min \{7, 3 + \infty\} = 7$$

$$d(e) = \min \{d(e), d(c) + a(c, e)\} = \min \{\infty, 3 + 3\} = 6$$

$$d(f) = \min \{d(f), d(c) + a(c, f)\} = \min \{\infty, 3 + \infty\} = \infty$$

Minimum di b, jalur yang didapat (a, b)

### Iterasi 3

Posisi awal di b

$$d(d) = \min \{d(d), d(b) + a(b, d)\} = \min \{7, 4 + 4\} = 7$$

$$d(e) = \min \{d(e), d(b) + a(b, e)\} = \min \{6, 4 + 2\} = 6$$

$$d(f) = \min \{d(f), d(b) + a(b, f)\} = \min \{\infty, 4 + \infty\} = \infty$$

Minimum di e, jalur yang didapat (b, e) atau (c, e)

### Iterasi 4

Posisi awal di e

$$d(d) = \min \{d(d), d(e) + a(e, d)\} = \min \{7, 6 + \infty\} = 7$$

$$d(f) = \min \{d(f), d(e) + a(e, f)\} = \min \{\infty, 6 + 5\} = 11$$

Minimum di d, jalur yang didapat (a, d)

### Iterasi 5

Posisi awal di d

$$d(f) = \min \{d(f), d(d) + a(d, f)\} = \min \{11, 7+2\} = 9$$

Minimum di f, Iterasi dihentikan, jalur yang didapat (d, f)

Maka jalur terpendek dari a ke f adalah {(a, d), (d, f)} dengan panjang 9.

### Algoritma Dijkstra

Procedure Dijkstra (G: berat graf terhubung, dengan semua berat graf positif)

{G mempunyai busur  $a=v_0, v_1, \dots, v_n$  dan berat  $w(v_i, v_j) = \infty$ , jika  $(v_i, v_j)$  tidak terhubung dengan busur lain}

for  $i:=1$  to  $n$

$L(v_i) := \infty$

$L(a) := 0$

$S := \{ \}$

{simpul telah diinisialisasi sehingga simpul a adalah kosong dan simpul lainnya adalah  $\infty$ , dan S adalah himpunan kosong}

while  $z \notin S$

begin

$u :=$  simpul tidak ada dalam S dengan  $L(u)$  minimal

$S := S \cup \{u\}$

    For semua busur v tidak ada dalam S

        If  $L(u) + w(u, v) < L(v)$  then  $L(v) := L(u) + w(u, v)$

        {menambahkan simpul ke S dengan nilai minimal dan mengubah nilai busur yang tidak berada di S}

end. { $Lz$ }=panjang dari jalur terpendek dari a ke z}

### **Algoritma Floyd**

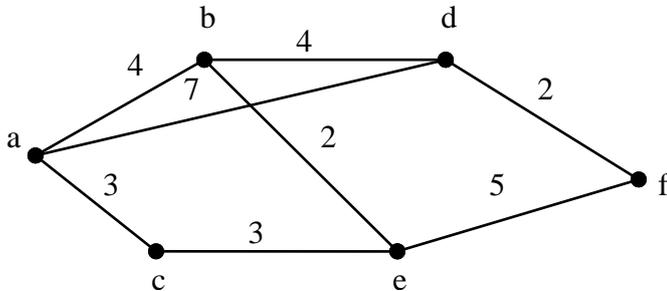
Algoritma yang juga sering digunakan untuk menentukan panjang jalur terpendek untuk setiap pasangan simpul adalah algoritma Floyd, sering disebut dengan all pair's shortest path algoritma. Langkah-langkah dalam algoritma Floyd :

- Setiap simpul diberi nomor dari 1, 2, ..., n. Susun matriks  $D^0$  yang masing-masing elemennya menunjukkan panjang busur terpendek yang menghubungkan simpul tersebut. (elemen i, j menunjukkan panjang busur terpendek yang menghubungkan  $v_i$  dengan  $v_j$ ). Jika tidak ada busur yang menghubungkan  $v_i$  dengan  $v_j$ , maka  $d_{ij}^0 = \infty$  dan  $d_{ii}^0 = 0$
- Lakukan iterasi sebanyak n kali dimana setiap iterasi disusun matrik  $D^m$  ( $m \in \{1, 2, \dots, n\}$ ) dari matriks  $D^{m-1}$  dengan rumus :

$$d_{ij}^m = \min\{d_{ij}^{m-1}, d_{im}^{m-1} + d_{mj}^{m-1}\}$$

- c) Catat setiap jalur yang didapat dari setiap iterasi. Akhiri iterasi setelah  $m=n$ , sehingga  $D^n$  menunjukkan panjang jalur terpendek yang menghubungkan simpul I dengan simpul j.

Contoh :



Penyelesaian :

Iterasi 0

Matriks  $d^0 =$

0	4	3	7	$\infty$	$\infty$
4	0	$\infty$	4	2	$\infty$
3	$\infty$	0	$\infty$	3	$\infty$
7	4	$\infty$	0	$\infty$	2
$\infty$	2	3	$\infty$	0	5
$\infty$	$\infty$	$\infty$	2	5	0

Iterasi 1

Matriks  $d^1 =$

0	4	3	7	$\infty$	$\infty$
4	0	7	4	2	$\infty$
3	7	0	10	3	$\infty$
7	4	10	0	$\infty$	2
$\infty$	2	3	$\infty$	0	5
$\infty$	$\infty$	$\infty$	2	5	0

Iterasi 2

Matriks  $d^2 =$

0	4	3	7	6	$\infty$
4	0	7	4	2	$\infty$
3	7	0	10	3	$\infty$
7	4	10	0	6	2
6	2	3	6	0	5
$\infty$	$\infty$	$\infty$	2	5	0

Iterasi 3

Matriks  $d^3 =$

0	4	3	7	6	$\infty$
4	0	7	4	2	$\infty$
3	7	0	10	3	$\infty$
7	4	10	0	6	2
6	2	3	6	0	5
$\infty$	$\infty$	$\infty$	2	5	0

Iterasi 4

Matriks  $d^4 =$

0	4	3	7	6	9
4	0	7	4	2	6
3	7	0	10	3	12
7	4	10	0	6	2
6	2	3	6	0	5
9	6	12	2	5	0

Iterasi 5

Matriks  $d^5 =$

0	4	3	7	6	9
4	0	7	4	2	6
3	7	0	9	3	8
7	4	9	0	6	2
6	2	3	6	0	5
9	6	8	2	5	0

Iterasi 6

Matriks  $d^6 =$

0	4	3	7	6	9
4	0	7	4	2	6
3	7	0	9	3	8
7	4	9	0	6	2
6	2	3	6	0	5
9	6	8	2	5	0