

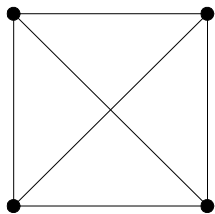
POHON: POHON MERENTANG DAN POHON BINER

MATEMATIKA DISKRIT 2

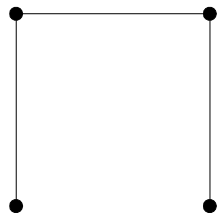
24 APRIL 2015

POHON MERENTANG (*SPANNING TREE*)

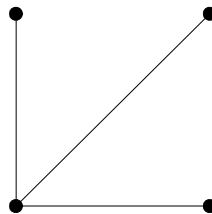
- Pohon merentang dari graf terhubung adalah upagraf merentang yang berupa pohon.
- Pohon merentang diperoleh dengan memutus sirkuit di dalam graf.



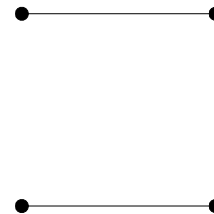
G



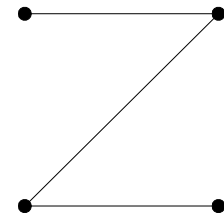
T_1



T_2



T_3

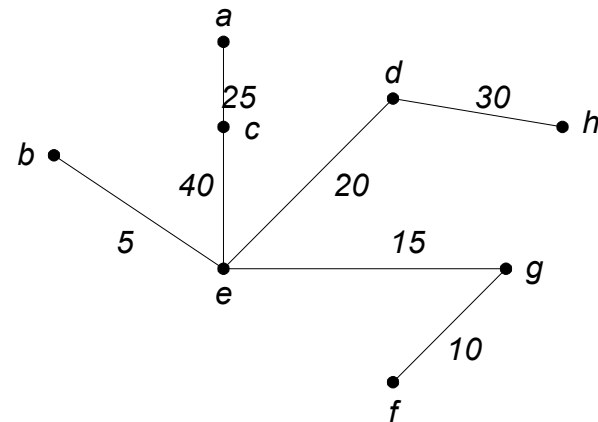
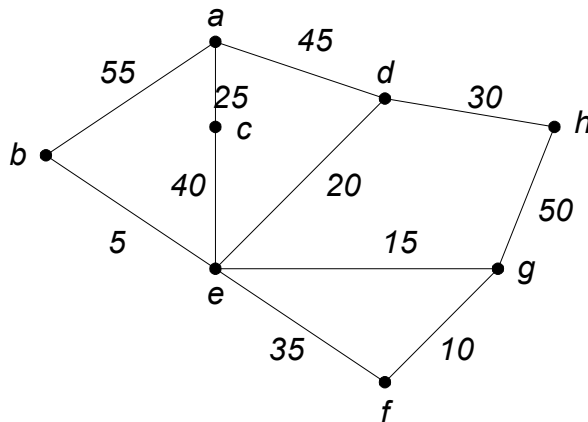


T_4

- Setiap graf terhubung mempunyai paling sedikit satu buah pohon merentang.
- Graf tak-terhubung dengan k komponen mempunyai k buah hutan merentang yang disebut hutan merentang (*spanning forest*).

POHON MERENTANG MINIMUM

- Graf terhubung-berbobot mungkin mempunyai lebih dari 1 pohon merentang.
- Pohon merentang yang berbobot minimum dinamakan **pohon merentang minimum** (*minimum spanning tree*).



Algoritma Prim

Langkah 1: ambil sisi dari graf G yang berbobot minimum, masukkan ke dalam T .

Langkah 2: pilih sisi (u, v) yang mempunyai bobot minimum dan bersisian dengan simpul di T , tetapi (u, v) tidak membentuk sirkuit di T . Masukkan (u, v) ke dalam T .

Langkah 3: ulangi langkah 2 sebanyak $n - 2$ kali.

```
procedure Prim(input G : graf, output T : pohon)
{ Membentuk pohon merentang minimum T dari graf terhubung-
berbobot G.
Masukan: graf-berbobot terhubung  $G = (V, E)$ , dengan  $|V| = n$ 
Keluaran: pohon rentang minimum  $T = (V, E')$ 
}
```

Deklarasi

i, p, q, u, v : integer

Algoritma

Cari sisi (p,q) dari E yang berbobot terkecil

$T \leftarrow \{(p,q)\}$

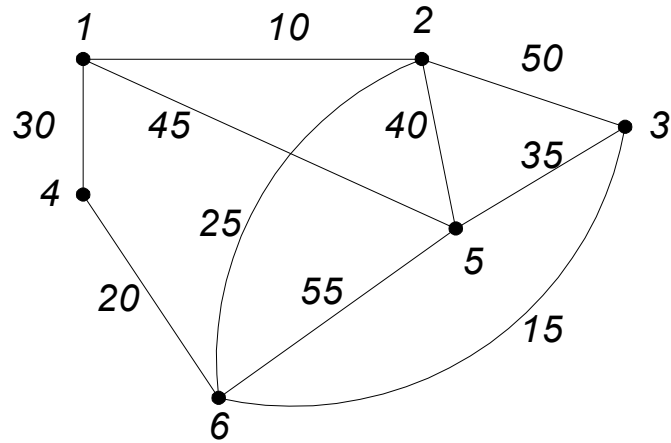
for i \leftarrow 1 to n-2 do

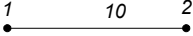
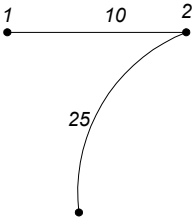
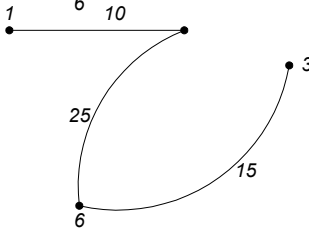
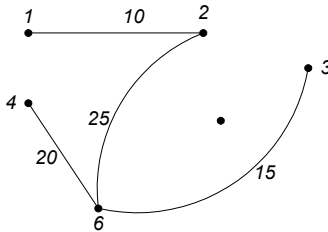
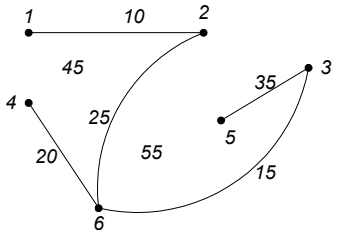
 Pilih sisi (u,v) dari E yang bobotnya terkecil namun
 bersisian dengan simpul di T

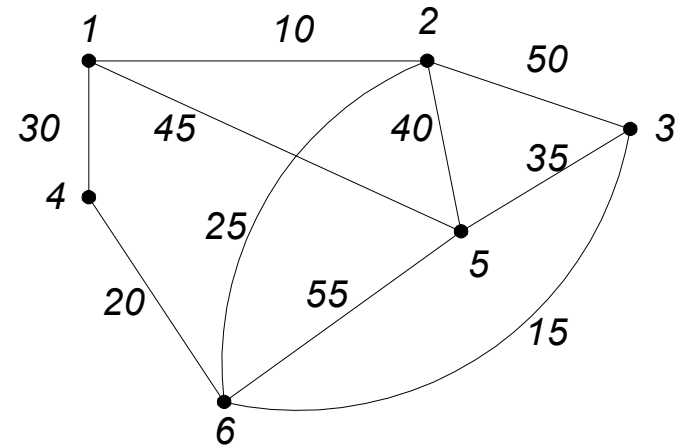
$T \leftarrow T \cup \{(u,v)\}$

endfor

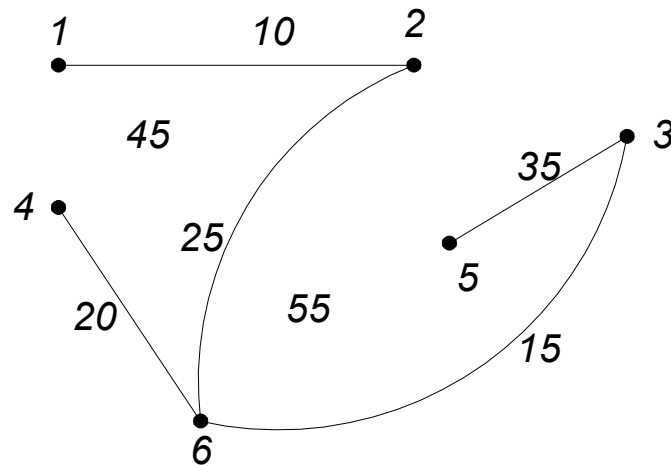
Contoh:



Langkah	Sisi	Bobot	Pohon rentang
1	(1, 2)	10	
2	(2, 6)	25	
3	(3, 6)	15	
4	(4, 6)	20	
5	(3, 5)	35	



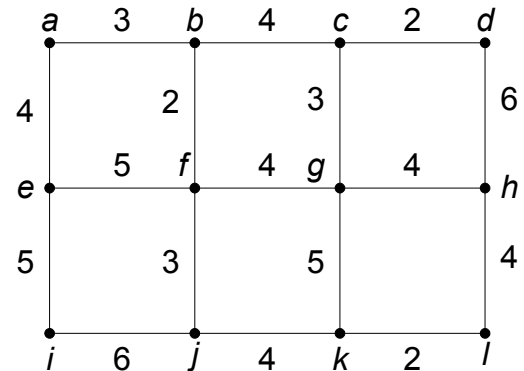
Pohon merentang minimum yang dihasilkan:



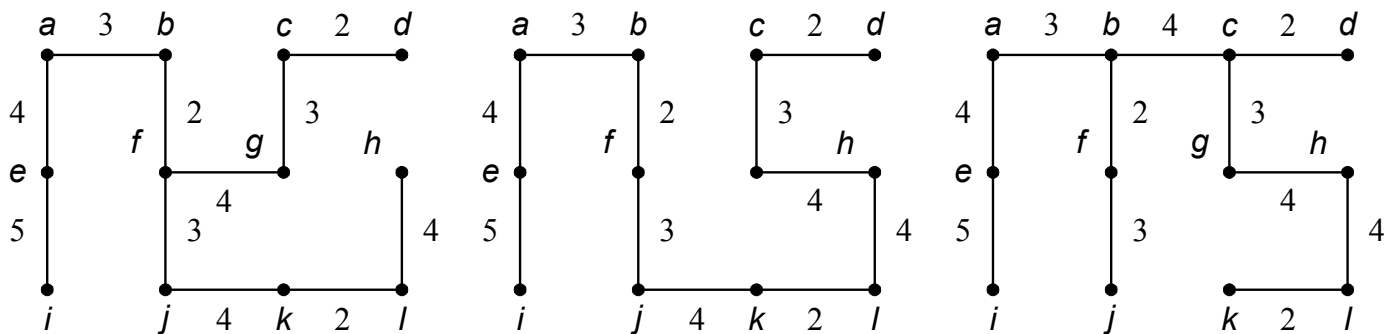
$$\text{Bobot} = 10 + 25 + 15 + 20 + 35 = 105$$

- **Pohon merentang yang dihasilkan tidak selalu unik meskipun bobotnya tetap sama.**
- **Hal ini terjadi jika ada beberapa sisi yang akan dipilih berbobot sama.**

Contoh:



Tiga buah pohon merentang minimumnya:



Bobotnya sama yaitu = 36

Algoritma Kruskal

(Langkah 0: sisi-sisi dari graf sudah diurut menaik berdasarkan bobotnya – dari bobot kecil ke bobot besar)

Langkah 1: T masih kosong

Langkah 2: pilih sisi (u, v) dengan bobot minimum yang tidak membentuk sirkuit di T . Tambahkan (u, v) ke dalam T .

Langkah 3: ulangi langkah 2 sebanyak $n - 1$ kali.

```
procedure Kruskal(input G : graf, output T : pohon)
{ Membentuk pohon merentang minimum T dari graf terhubung -
berbobot G.
```

```
Masukan: graf-berbobot terhubung  $G = (V, E)$ , dengan  $|V| = n$ 
Keluaran: pohon rentang minimum  $T = (V, E')$ 
}
```

Deklarasi

```
i, p, q, u, v : integer
```

Algoritma

```
( Asumsi: sisi-sisi dari graf sudah diurut menaik
berdasarkan bobotnya - dari bobot kecil ke bobot
besar)
```

```
T  $\leftarrow$  {}
```

```
while jumlah sisi T < n-1 do
```

```
    Pilih sisi (u,v) dari E yang bobotnya terkecil
```

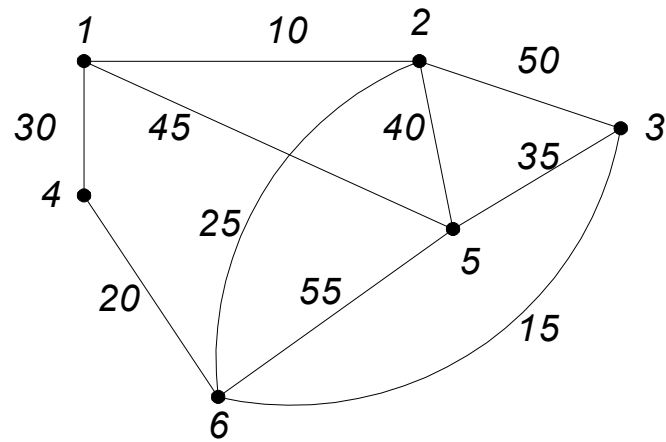
```
    if (u,v) tidak membentuk siklus di T then
```

```
        T  $\leftarrow$  T  $\cup$  {(u,v)}
```

```
    endif
```



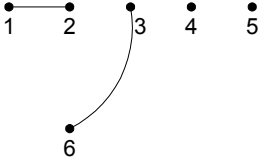
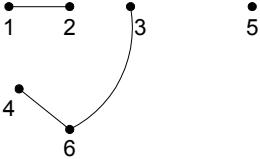
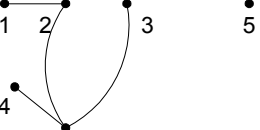
```
endfor
```

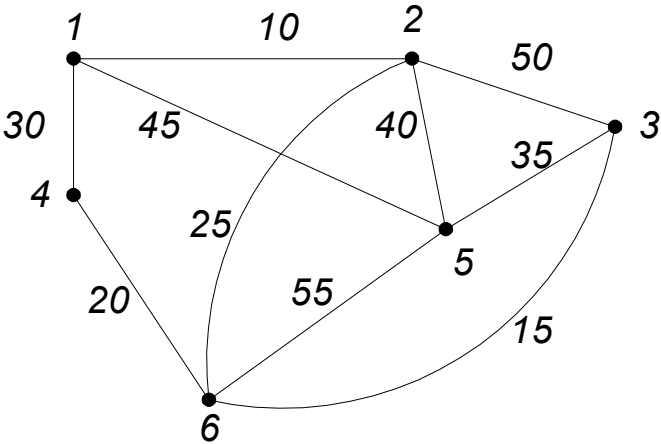
Contoh:



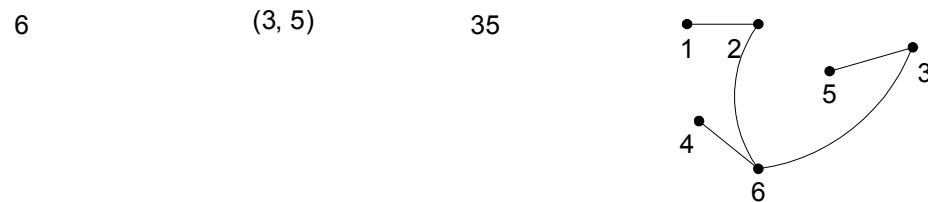
Sisi-sisi diurut menaik:

Sisi	(1,2)	(3,6)	(4,6)	(2,6)	(1,4)	(3,5)	(2,5)	(1,5)	(2,3)	(5,6)
Bobot	10	15	20	25	30	35	40	45	50	55

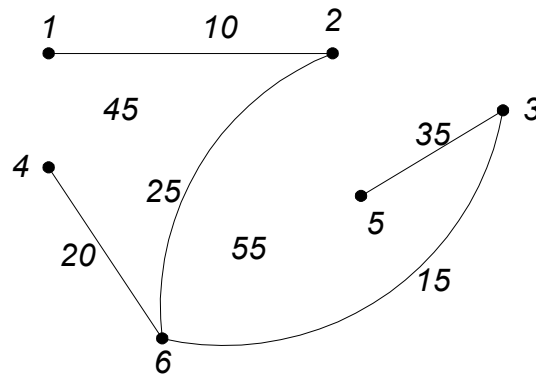
Langkah	Sisi	Bobot	Hutan merentang
0			
1	(1, 2)	10	
2	(3, 6)	15	
3	(4, 6)	20	
4	(2, 6)	25	



5 (1, 4) 30 ditolak



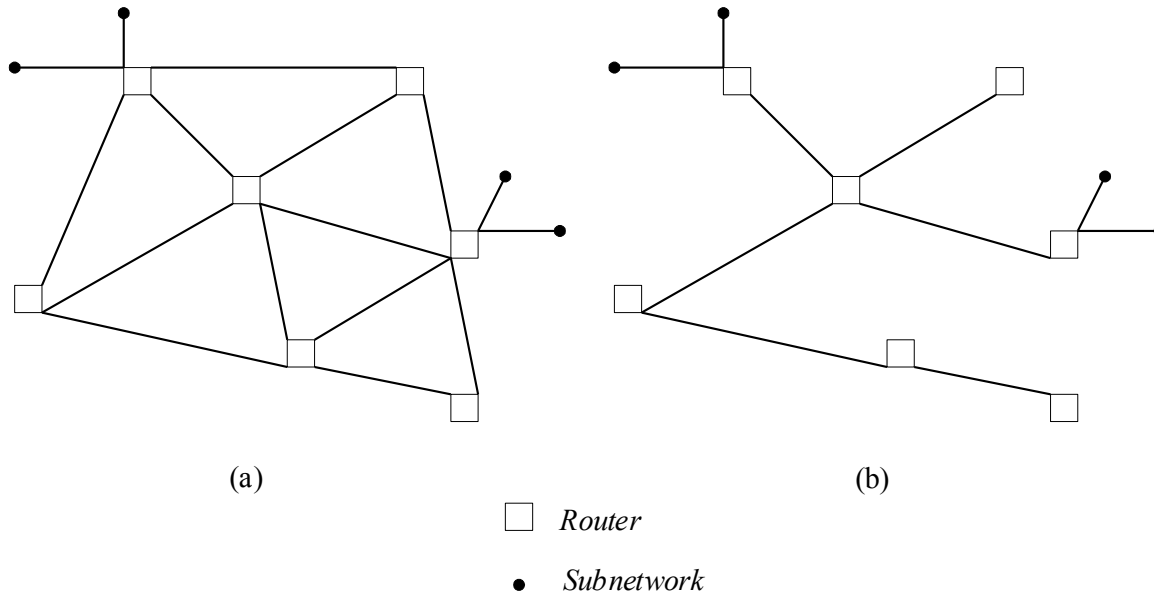
Pohon merentang minimum yang dihasilkan:



$$\text{Bobot} = 10 + 25 + 15 + 20 + 35 = 105$$

APLIKASI POHON MERENTANG

1. Jumlah ruas jalan semimumimum mungkin yang menghubungkan semua kota sehingga setiap kota tetap terhubung satu sama lain.
2. Perutean (*routing*) pesan pada jaringan komputer.

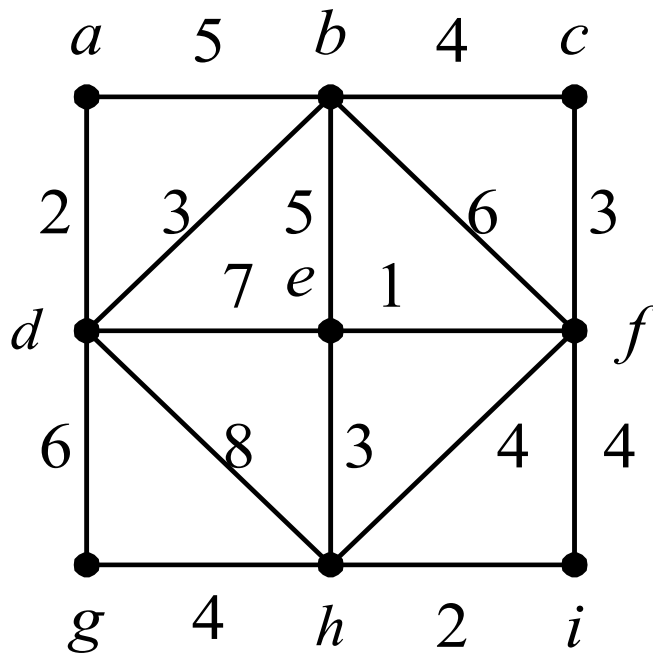


(a) Jaringan komputer, (b) Pohon merentang *multicast*

LATIHAN

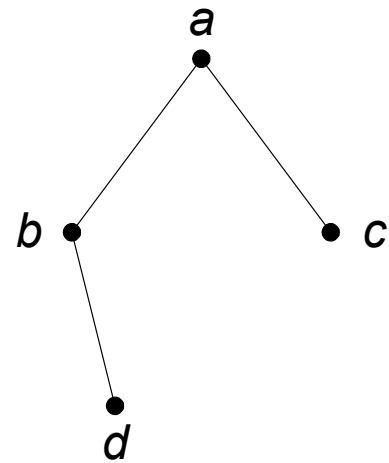
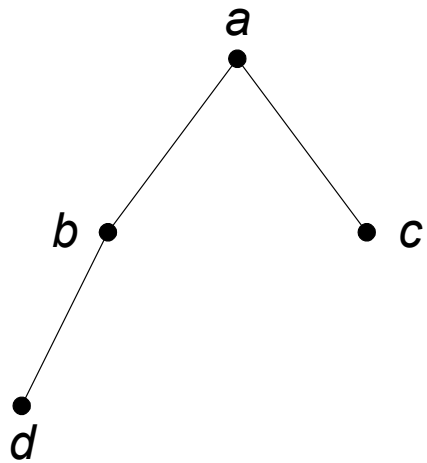
Tentukan dan gambarkan pohon merentang minimum dari graf di bawah ini dengan algoritma:

- Prim
- Kruskal

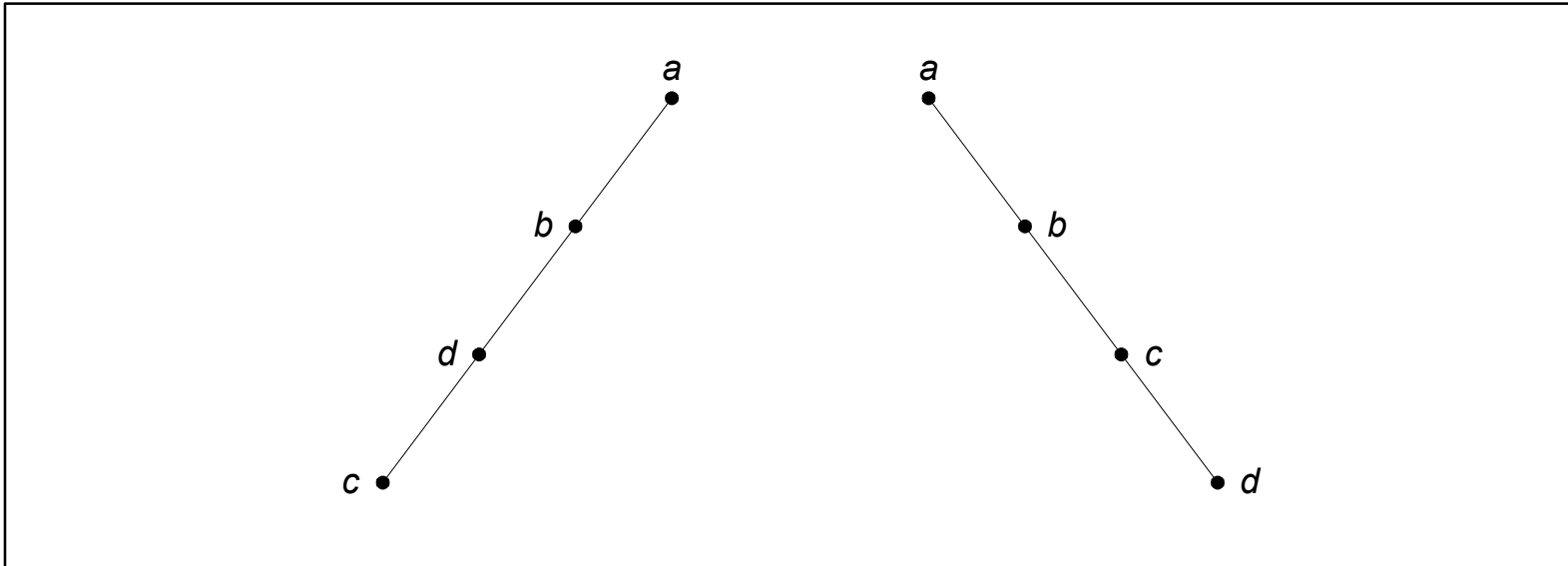


POHON BINER (*BINARY TREE*)

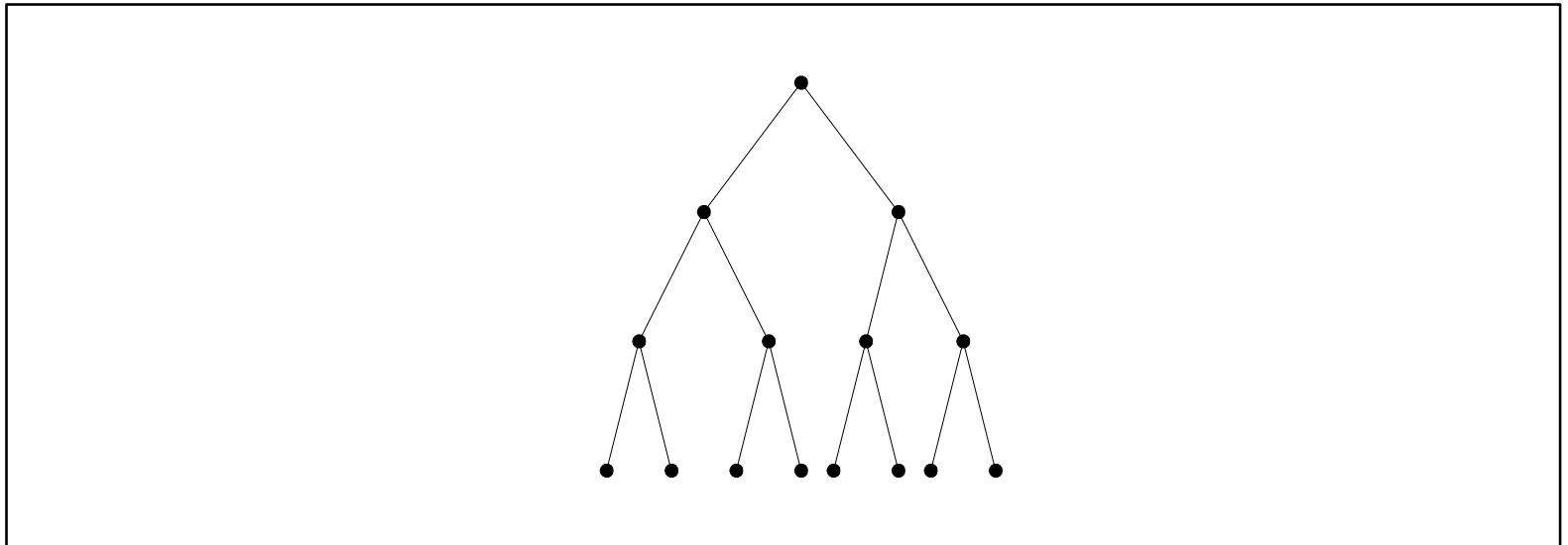
- Adalah pohon *n-ary* dengan $n = 2$.
- Pohon yang paling penting karena banyak aplikasinya.
- Setiap simpul di adlam pohon biner mempunyai paling banyak 2 buah anak.
- Dibedakan antara anak kiri (*left child*) dan anak kanan (*right child*)
- Karena ada perbedaan urutan anak, maka pohon biner adalah pohon terurut.



Gambar Dua buah pohon biner yang berbeda



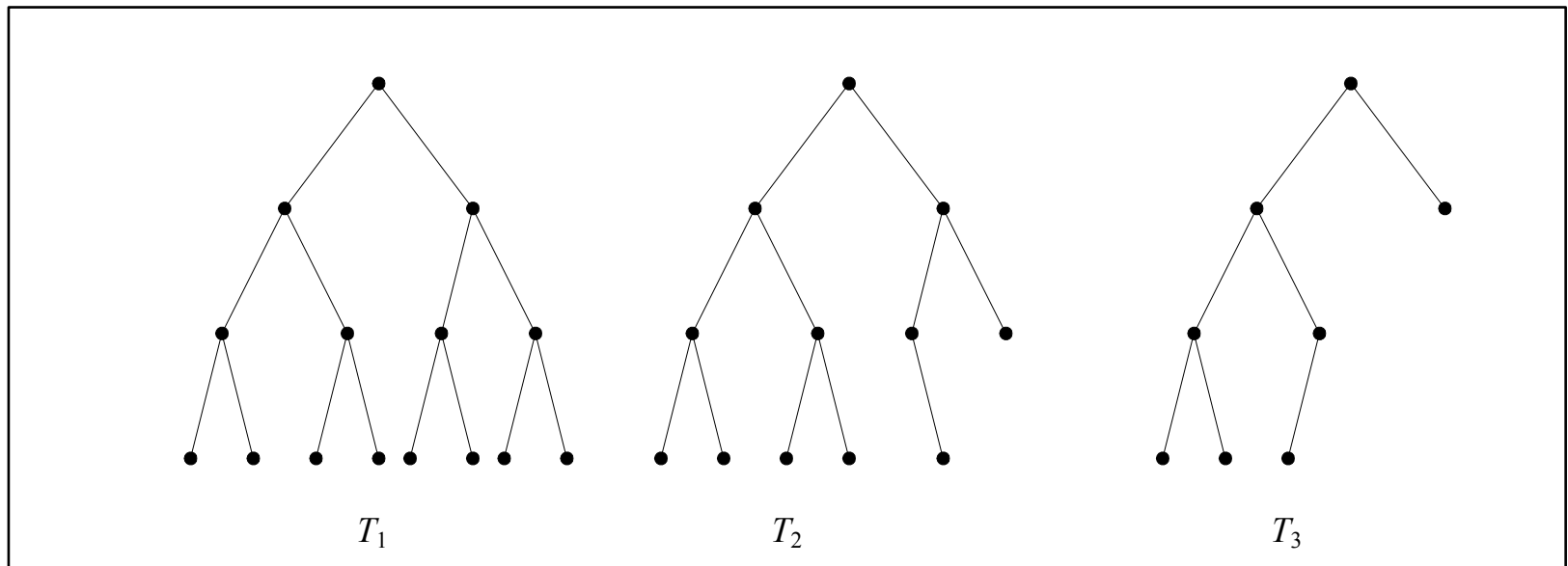
Gambar (a) Pohon condong-kiri, dan (b) pohon condong kanan



Gambar Pohon biner penuh

Pohon Biner Seimbang

Pada beberapa aplikasi, diinginkan tinggi upapohon kiri dan tinggi upapohon kanan yang seimbang, yaitu berbeda maksimal 1.



Gambar T_1 dan T_2 adalah pohon seimbang, sedangkan T_3 bukan pohon seimbang.

PENELUSURAN (TRAVERSAL) POHON BINER

1. *Preorder* : R, T_1, T_2

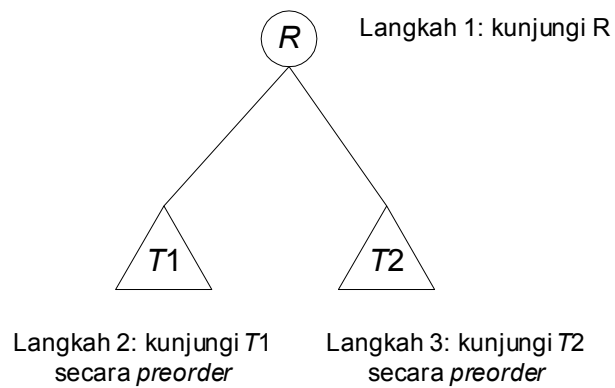
- kunjungi R
- kunjungi T_1 secara *preorder*
- kunjungi T_2 secara *preorder*

2. *Inorder* : T_1, R, T_2

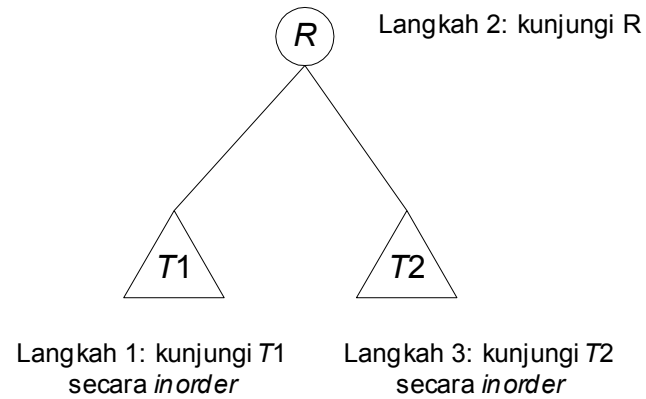
- kunjungi T_1 secara *inorder*
- kunjungi R
- kunjungi T_2 secara *inorder*

3. *Postorder* : T_1, T_2, R

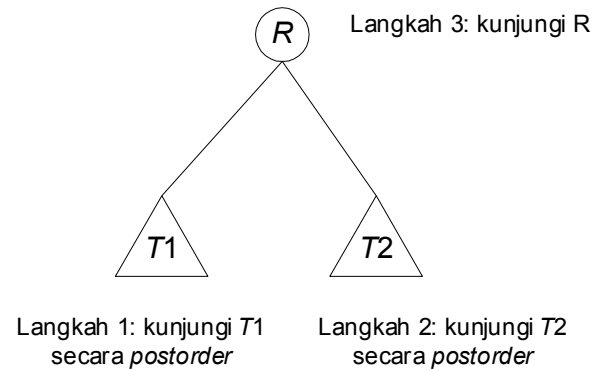
- kunjungi T_1 secara *postorder*
- kunjungi T_2 secara *postorder*
- kunjungi R



(a) *preorder*

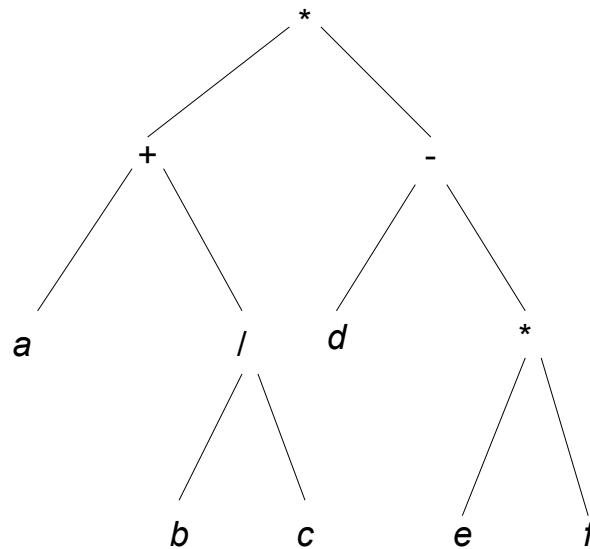


(b) *inorder*



(c) *postorder*

<i>preorder</i>	: $* + a / b \ c - d * e f$	(<i>prefix</i>)
<i>inorder</i>	: $a + b / c * d - e * f$	(<i>infix</i>)
<i>postorder</i>	: $a \ b \ c / + \ d \ e \ f * - *$	(<i>postfix</i>)



LATIHAN

Tentukan hasil kunjungan *preorder*, *inorder*, dan *postorder* pada pohon 4-ary berikut ini:

