

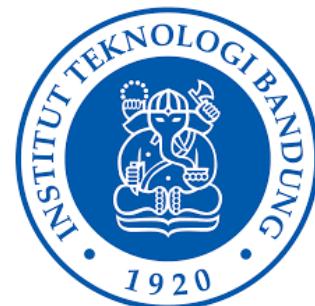
# Algoritma Branch & Bound

Bahan Kuliah IF2211 Strategi Algoritma

(Bagian 4)

Oleh: Rinaldi Munir

Update: Masayu Leylia Khoddra



Program Studi Teknik Informatika  
Sekolah Teknik Elektro dan Informatika ITB  
2025

# **Assignment Problem**

- Misalkan terdapat  $n$  orang dan  $n$  buah pekerjaan (*job*). Setiap orang akan di-*assign* dengan sebuah *job*. Ongkos (*cost*) untuk meng-*assign* setiap orang dengan sebuah *job* dinyatakan dengan sebuah matriks.
- Bagaimana meng-*assign* *job* dengan orang sehingga total ongkos *assignment* seminimal mungkin?
- Contoh:  $n = 4$

$$C = \begin{bmatrix} \textit{Job 1} & \textit{Job 2} & \textit{Job 3} & \textit{Job 4} \\ 9 & 2 & 7 & 8 \\ 6 & 4 & 3 & 7 \\ 5 & 8 & 1 & 8 \\ 7 & 6 & 9 & 4 \end{bmatrix} \begin{array}{l} \text{Orang } a \\ \text{Orang } b \\ \text{Orang } c \\ \text{Orang } d \end{array}$$

# Assignment Problem: Exhaustive Search

1. Enumerasi  $n!$

$$C = \begin{bmatrix} Job 1 & Job 2 & Job 3 & Job 4 \\ 9 & 2 & 7 & 8 \\ 6 & 4 & 3 & 7 \\ 5 & 8 & 1 & 4 \\ 7 & 6 & 9 & 4 \end{bmatrix}$$

2. Evaluasi biaya penugasan

$\langle 1, 2, 3, 4 \rangle$	$cost = 9 + 4 + 1 + 4 = 18$
$\langle 1, 2, 4, 3 \rangle$	$cost = 9 + 4 + 8 + 9 = 30$
$\langle 1, 3, 2, 4 \rangle$	$cost = 9 + 3 + 8 + 4 = 24$
$\langle 1, 3, 4, 2 \rangle$	$cost = 9 + 3 + 8 + 6 = 26$
$\langle 1, 4, 2, 3 \rangle$	$cost = 9 + 7 + 8 + 9 = 33$
$\langle 1, 4, 3, 2 \rangle$	$cost = 9 + 7 + 1 + 6 = 23$

etc.

solutions to the assignment problem as  $n$ -tuples  $j_1, \dots, j_n$  in which the  $i$ th component,  $i = 1, \dots, n$ , indicates the column of the element selected in the  $i$ th row (i.e., the job number assigned to the  $i$ th person).

3. Pilih penugasan dgn biaya minimum

# Assignment Problem: Greedy

Strategi Greedy: assign cost terkecil terlebih dahulu untuk orang yg belum mendapat job. Setiap orang mendapat tepat sebuah *job*.

$$C = \begin{bmatrix} \text{Job 1} & \text{Job 2} & \text{Job 3} & \text{Job 4} \\ 9 & 2 & 7 & 8 \\ 6 & 4 & 3 & 7 \\ 5 & 8 & 1 & 4 \\ 7 & 6 & 9 & 4 \end{bmatrix}$$

Orang *a*  
Orang *b*  
Orang *c*  
Orang *d*

solutions to the assignment problem as  $n$ -tuples  $j_1, \dots, j_n$  in which the  $i$ th component,  $i = 1, \dots, n$ , indicates the column of the element selected in the  $i$ th row (i.e., the job number assigned to the  $i$ th person).

Langkah 1:  $<0,0,3,0>$ , total cost=1

Langkah 2:  $<2,0,3,0>$ , total cost=2+1=3

Langkah 3:  $<2,0,3,4>$ , total cost=2+1+4=7

Langkah 4:  $<2,1,3,4>$ , total cost=2+6+1+4=13

Solusi:  $<2,1,3,4>$  dengan total cost 13.

## Penyelesaian dengan Branch & Bound:

- *Cost (lower bound)* setiap simpul hidup di dalam pohon ruang status dapat dihitung dengan berbagai cara, misalnya menggunakan **matriks ongkos tereduksi**.
- Cara lain yang lebih sederhana untuk menghitung *lower bound* adalah dengan **menjumlahkan nilai minimum pada setiap baris matriks**. Dasar pemikirannya adalah bahwa sembarang solusi, termasuk solusi optimal, total ongkos penugasannya tidak lebih kecil dari jumlah semua nilai terkecil pada setiap baris.
- Untuk sembarang solusi yang *legitimate* (tidak ada job yang sama di-*assign* ke 2 orang atau lebih) jika sebuah job *di-assign* dengan orang, maka ongkos peng-*assign-an* tersebut dihitung sebagai salah satu komponen nilai terkecil di dalam penjumlahan tersebut.

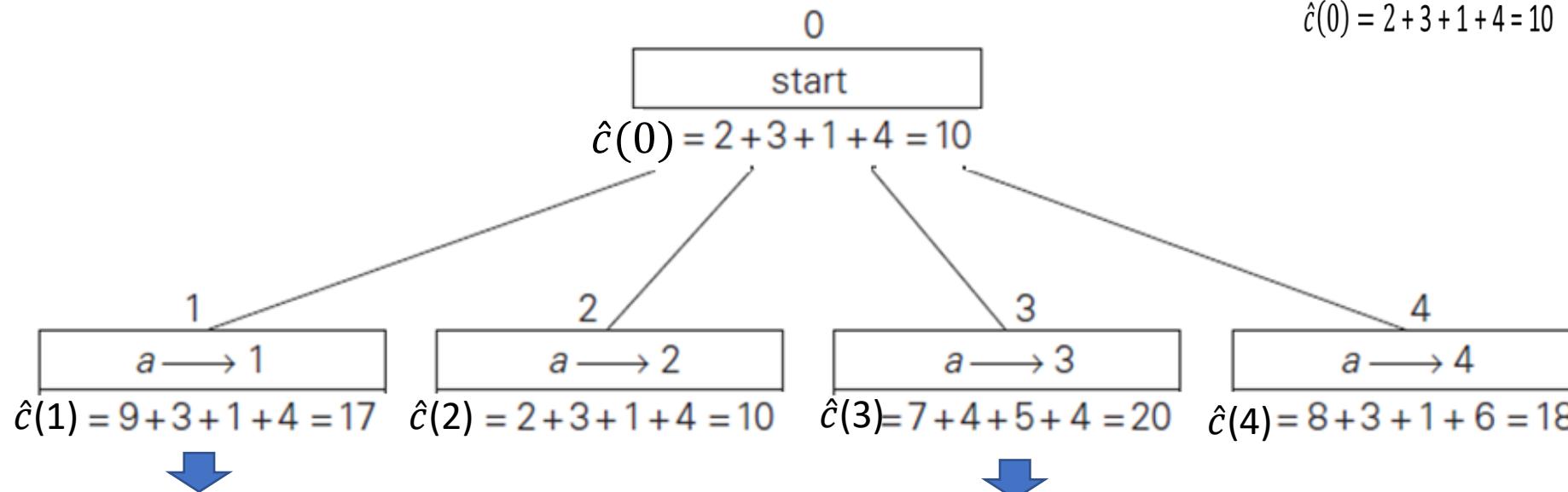
1. Cost untuk simpul akar:

$$\hat{c}(0) = 2 + 3 + 1 + 4 = 10$$

2. Bangkitkan anak-anak dari simpul akar:

	Job 1	Job 2	Job 3	Job 4	
Orang a	9	2	7	8	
Orang b	6	4	3	7	
Orang c	5	8	1	8	
Orang d	7	6	9	4	

$$\hat{c}(0) = 2 + 3 + 1 + 4 = 10$$

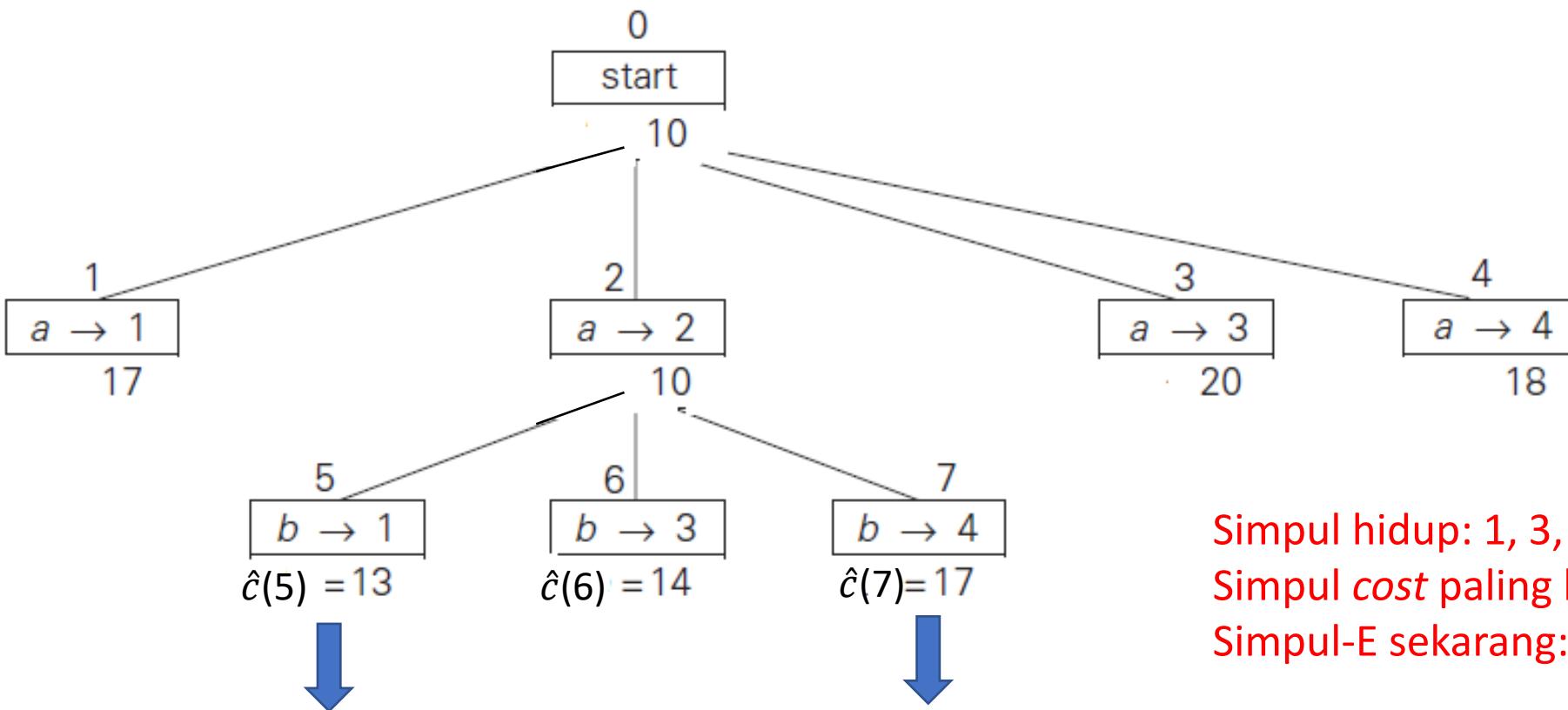


	Job 1	Job 2	Job 3	Job 4	
Orang a	9	2	7	8	
Orang b	6	4	3	7	
Orang c	5	8	1	8	
Orang d	7	6	9	4	

	Job 1	Job 2	Job 3	Job 4	
Orang a	9	2	7	8	
Orang b	6	4	3	7	
Orang c	5	8	1	8	
Orang d	7	6	9	4	

Simpul hidup: 1, 2, 3, dan 4  
Simpul cost paling kecil: 2  
Simpul-E sekarang: 2

(Sumber gambar: Levitin, 2003)



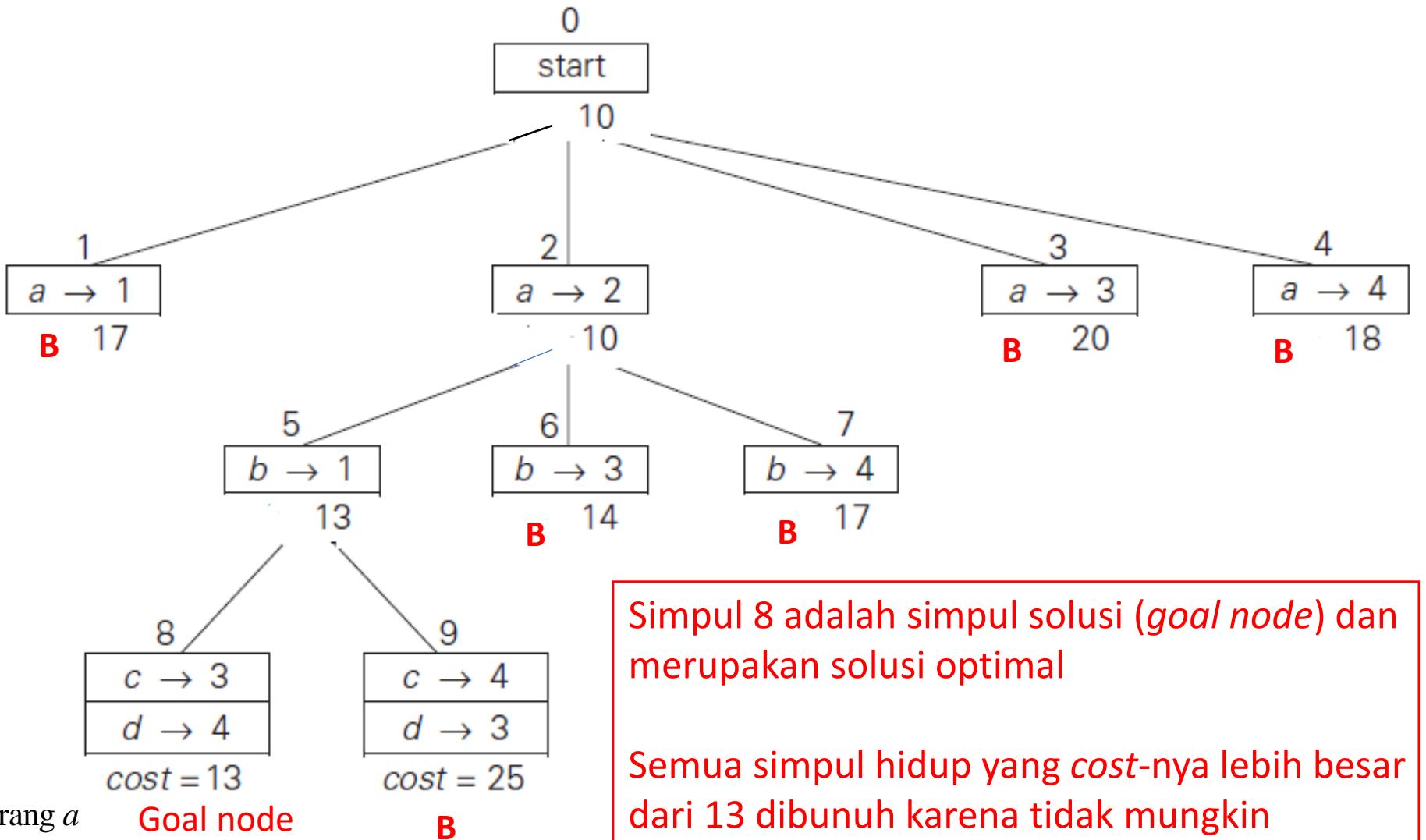
$$C = \begin{bmatrix} \text{Job 1} & \text{Job 2} & \text{Job 3} & \text{Job 4} \\ 9 & 2 & 7 & 8 \\ 6 & 4 & 3 & 7 \\ 5 & 8 & 1 & 8 \\ 7 & 6 & 9 & 4 \end{bmatrix} \begin{array}{l} \text{Orang } a \\ \text{Orang } b \\ \text{Orang } c \\ \text{Orang } d \end{array}$$

$$\hat{c}(5) = 2 + 6 + 1 + 4 = 13$$

$$C = \begin{bmatrix} \text{Job 1} & \text{Job 2} & \text{Job 3} & \text{Job 4} \\ 9 & 2 & 7 & 8 \\ 6 & 4 & 3 & 7 \\ 5 & 8 & 1 & 4 \\ 7 & 6 & 9 & 4 \end{bmatrix} \begin{array}{l} \text{Orang } a \\ \text{Orang } b \\ \text{Orang } c \\ \text{Orang } d \end{array}$$

$$\hat{c}(7) = 2 + 7 + 1 + 7 = 17$$

Simpul hidup: 1, 3, 4, 5, 6, dan 7  
 Simpul *cost* paling kecil: 5  
 Simpul-E sekarang: 5



$C =$	$\begin{bmatrix} Job\ 1 & Job\ 2 & Job\ 3 & Job\ 4 \end{bmatrix}$	$Orang\ a$	$Goal\ node$	$B$
	$\begin{bmatrix} 9 & 2 & 7 & 8 \end{bmatrix}$	Orang <i>a</i>	Goal node	
	$\begin{bmatrix} 6 & 4 & 3 & 7 \end{bmatrix}$	Orang <i>b</i>	(solusi optimal)	
	$\begin{bmatrix} 5 & 8 & 1 & 4 \end{bmatrix}$	Orang <i>c</i>		
	$\begin{bmatrix} 7 & 6 & 9 & 4 \end{bmatrix}$	Orang <i>d</i>		

$$\hat{c}(8) = 2 + 6 + 1 + 4 = 13$$

Simpul 8 adalah simpul solusi (*goal node*) dan merupakan solusi optimal

Semua simpul hidup yang *cost*-nya lebih besar dari 13 dibunuh karena tidak mungkin menghasilkan *cost* lebih kecil dari 13.  
(simpul 1, 3, 4, 7. dan 9 dibunuh → B)

Solusi optimal: X = (a → 2, b → 1, c → 3, d → 4)  
Cost = 13

# Integer Knapsack Problem

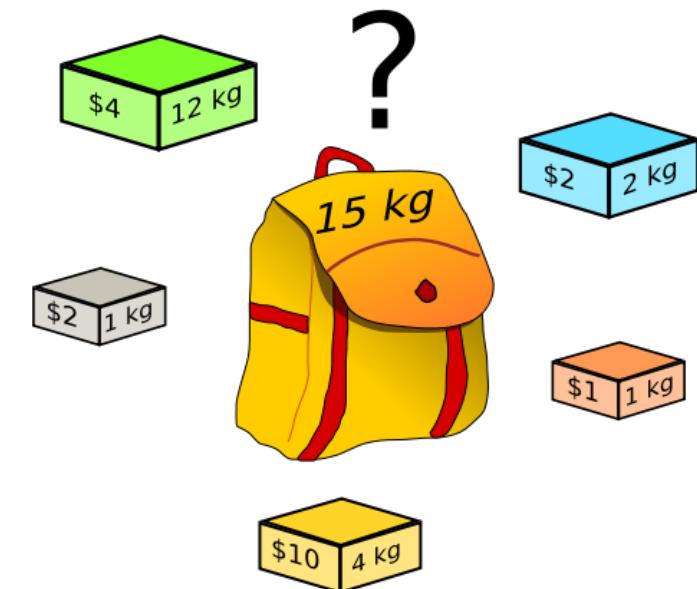
- **Persoalan:** Diberikan  $n$  buah objek dan sebuah *knapsack* dengan kapasitas bobot  $K$ . Setiap objek memiliki properti bobot (*weight*)  $w_i$  dan keuntungan(*profit*)  $p_i$ . Bagaimana cara memilih objek-objek yang dimasukkan ke dalam *knapsack* sedemikian sehingga diperoleh total keuntungan yang maksimal dengan syarat tidak boleh melebihi kapasitas *knapsack*.
- Formulasi matematis:

$$\text{Maksimasi } F = \sum_{i=1}^n p_i x_i$$

dengan kendala (*constraint*)

$$\sum_{i=1}^n w_i x_i \leq K$$

yang dalam hal ini,  $x_i = 0$  atau  $1$ ,  $i = 1, 2, \dots, n$



# Greedy: 1/0 Knapsack

**Greedy by density:** Pada setiap langkah, knapsack diisi dengan objek yang mempunyai  $p_i/w_i$  terbesar. Mencoba **memaksimumkan keuntungan** dengan memilih objek yang mempunyai **keuntungan per unit berat** terbesar.

**Contoh:**  $(w_1, p_1) = (6, 12)$ ;  $(w_2, p_2) = (5, 15)$ ;  $(w_3, p_3) = (10, 50)$ ;  $(w_4, p_4) = (5, 10)$

dan sebuah knapsack dengan kapasitas  $K = 16$ . Solusi optimal:  $X = (0, 1, 1, 0)$

Properti objek				Greedy by			Solusi Optimal
$i$	$w_i$	$p_i$	$p_i/w_i$	<i>profit</i>	<i>weight</i>	<i>density</i>	
1	6	12	2	0	1	0	0
2	5	15	3	1	1	1	1
3	10	50	5	1	0	1	1
4	5	10	2	0	1	0	0
Total bobot			15	16	15	15	
Total keuntungan			65	37	65	65	

# Backtracking: 1/0 Knapsack (N=3)

$$X = (x_1, x_2, x_3), x_i \in \{0, 1\}$$

$$(w_1, w_2, w_3) = (35, 32, 25)$$

$$(p_1, p_2, p_3) = (40, 25, 50)$$

$$M = 30$$

Constraints:  $\sum_{i=1}^k w_i x_i \leq M$

RunutBalikR(1)

T([]): x1=1, B=F (bounded)

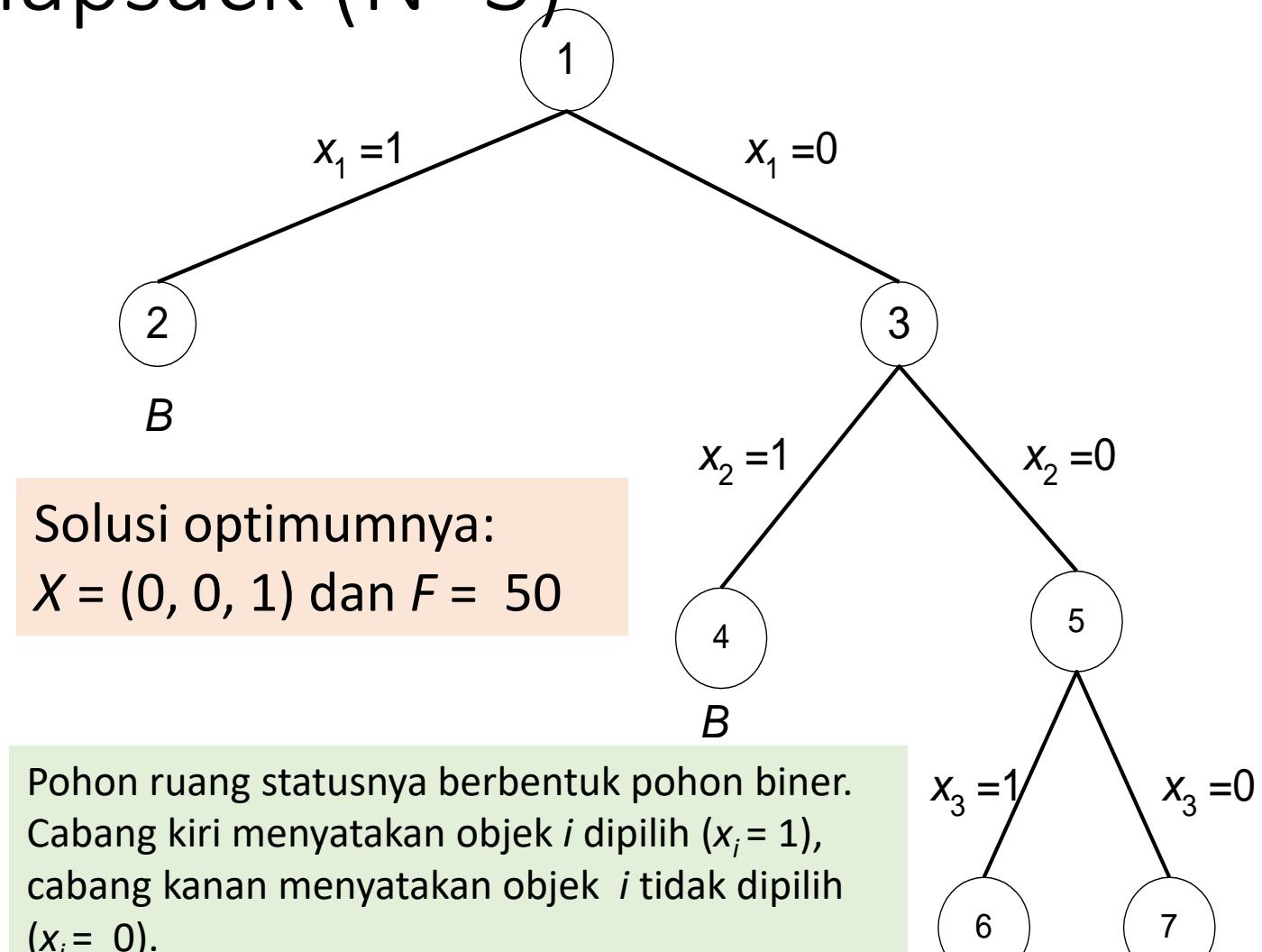
T([]): x1=0, B=T

T([x1=0]): x2=1, B=F (bounded)

T([x1=0]): x2=0, B=T

T([x1=0,x2=0]): x3=1, B=T, solusi

T([x1=0,x2=0]): x3=0, B=T, solusi



Pada pohon ruang status, sembarang lintasan dari akar ke daun menyatakan himpunan bagian (subset)

# Branch & Bound: 1/0 Knapsack

- Persoalan *knapsack* adalah persoalan **maksimasi** (mencari keuntungan maksimum)
- Oleh karena itu, *cost* setiap simpul pada pohon ruang status menyatakan **batas atas (*upper bound*)** dari solusi optimum. (Bandingkan dengan pendekatan *least cost search* (untuk persoalan minimasi) yang dalam hal ini *cost* setiap simpul menyatakan batas bawah (*lower bound*) dari solusi optimum)
- Pada persoalan maksimasi, simpul berikutnya yang diekspansi adalah **simpul hidup** yang memiliki **cost paling besar**.
- Agar pencarian solusi lebih mangkus, maka objek-objek diurutkan berdasarkan  **$p_i/w_i$  yang menurun** (dari besar ke kecil) sebagai berikut:

$$p_1/w_1 \geq p_2/w_2 \geq \dots \geq p_n/w_n$$

- Pohon ruang statusnya berbentuk pohon biner. Cabang kiri menyatakan objek  $i$  dipilih ( $x_i = 1$ ), cabang kanan menyatakan objek  $i$  tidak dipilih ( $x_i = 0$ ).
- Tiap simpul pada aras  $i$  di dalam pohon biner,  $i = 0, 1, 2, \dots, n$ , menyatakan himpunan bagian (*subset*) dari  $n$  objek yang dimasukkan ke dalam *knapsack*, yang dipilih dari  $i$  objek pertama (yang sudah diurut berdasarkan  $p_i/w_i$  yang menurun).
- Tiap simpul diisi dengan total bobot *knapsack* yang sudah terpakai ( $W$ ) dan total keuntungan yang sudah dicapai ( $F$ ).
- *Cost* atau batas atas (*upper bound*) simpul  $i$  dihitung sebagai penjumlahan total keuntungan yang sudah dicapai ( $F$ ) ditambah dengan perkalian sisa kapasitas *knapsack* ( $K - W$ ) dengan rasio keuntungan per bobot objek yang tersisa berikutnya ( $p_{i+1}/w_{i+1}$ ), atau dengan rumus:

$$\hat{c}(i) = F + (K - W)p_{i+1}/w_{i+1}$$

**Contoh:** Misalkan  $n = 4$ ,  $K = 10$

$$(w_1, w_2, w_3, w_4) = (4, 7, 5, 3),$$

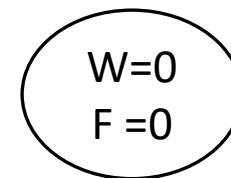
$$(p_1, p_2, p_3, p_4) = (40, 42, 25, 12),$$

i	wi	pi	Pi/wi	Greedy by density
1	4	40	10	1
2	7	42	6	0
3	5	25	5	1
4	3	12	4	0

Langkah-Langkah penyelesaian:

1. Hitung  $p_i/w_i \rightarrow (p_1/w_1, p_2/w_2, p_3/w_3, p_4/w_4) = (10, 6, 5, 4)$
2. Urutkan objek-objek berdasarkan  $p_i/w_i$  yang menurun  $\rightarrow$  kebetulan sudah terurut
3. Bangkitkan simpul akar (simpul 0),  $W = 0$ ,  $F = 0$ , (belum ada objek dipilih) dan  

$$\hat{c}(0) = F + (K - W)p_1/w_1 = 0 + (10 - 0)(10) = 100$$



$$\hat{c}(0) = 100$$

(Sumber: Levitin, 2003)

4. Bangkitkan simpul anak kiri (simpul 2) dan simpul anak kanan (simpul 3) dari simpul akar

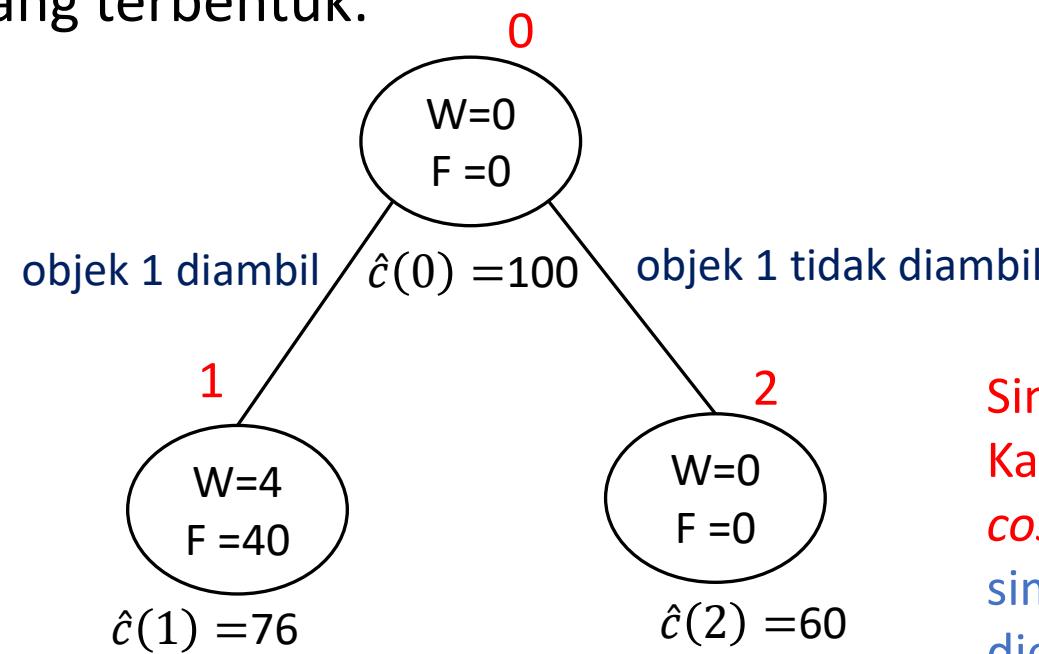
- Simpul 1 (objek 1 diambil):  $W = 0 + 4 = 4$ ;  $F = 0 + 40 = 40$

$$\hat{c}(1) = F + (K - W)p_2/w_2 = 40 + (10 - 4)(6) = 76$$

- Simpul 2 (objek 1 tidak diambil):  $W = 0 + 0 = 0$ ;  $F = 0 + 0 = 0$

$$\hat{c}(2) = F + (K - W)p_2/w_2 = 0 + (10 - 0)(6) = 60$$

Pohon ruang status yang terbentuk:



**Simpul hidup: 1 dan 2**  
 Karena simpul 1 memiliki *cost* paling besar, maka simpul 1 selanjutnya yang diekpansi

5. Bangkitkan anak-anak dari simpul 1, yaitu simpul 3 dan simpul 4

- Simpul 3 ( $w_2$  diambil):  $W = 4 + 7 = 11 >$  kapasitas knapsack ( $K = 10$ )

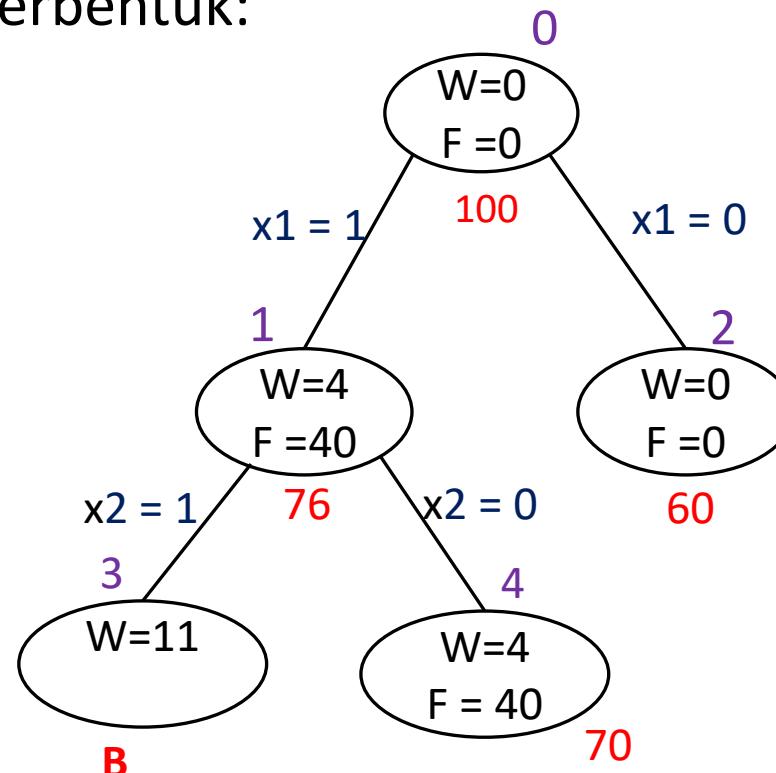
Simpul 3 langsung dimatikan (**B**).

- Simpul 4 ( $w_2$  tidak diambil):  $W = 4 + 0 = 4; F = 40 + 0 = 40$

$$\hat{c}(4) = F + (K - W)p_3/w_3 = 40 + (10 - 4)(5) = 70$$

Pohon ruang status yang terbentuk:

Simpul hidup: 2 dan 4  
Karena simpul 4 memiliki *cost* paling besar, maka simpul 4 selanjutnya yang diekpansi



Simpul 5 (objek 3 diambil):  $W = 4 + 5 = 9$ ;  $F = 40 + 25 = 65$

$$\hat{c}(5) = F + (K - W)p_4/w_4 = 65 + (10 - 9)(4) = 69$$

Simpul 6 (objek 3 tidak diambil):  $W = 4 + 0 = 4$ ;  $F = 40 + 0 = 40$

$$\hat{c}(6) = F + (K - W)p_4/w_4 = 40 + (10 - 4)(4) = 64$$

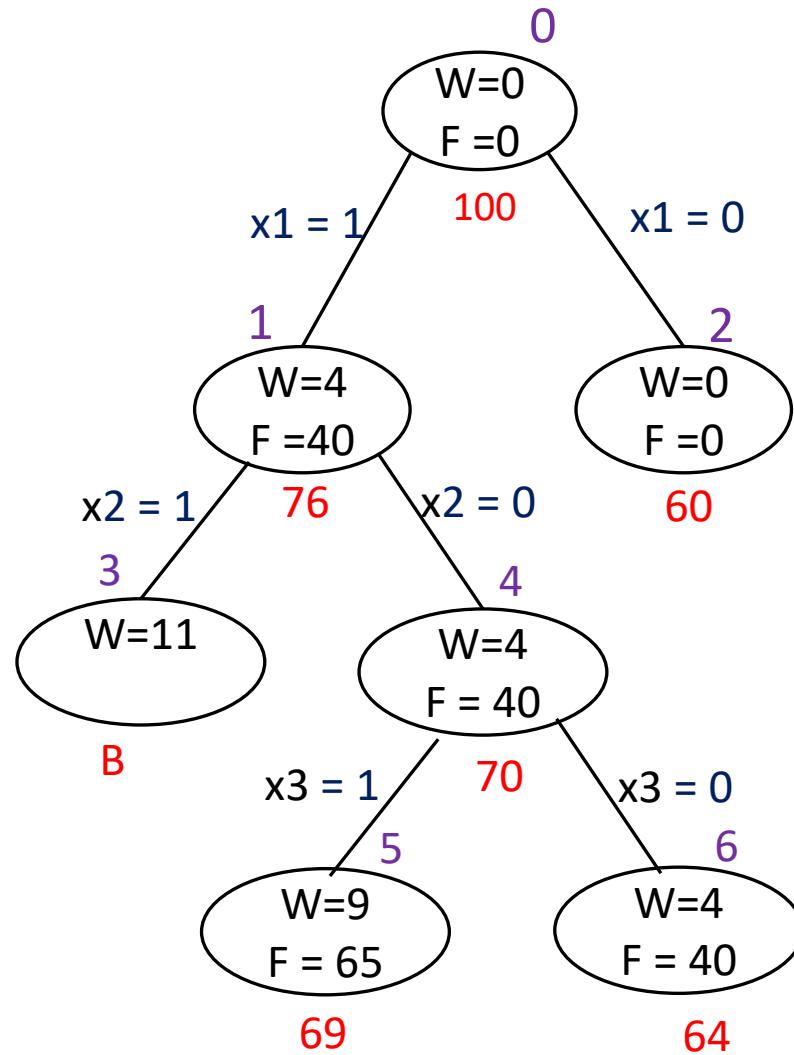
Simpul hidup: 2, 5, dan 6

Karena simpul 5 memiliki *cost* paling besar, maka simpul 5 selanjutnya yang diekspansi

$$(w_1, w_1, w_3, w_4) = (4, 7, 5, 3),$$

$$(p_1, p_1, p_3, p_4) = (40, 42, 25, 12),$$

$$(p_1/w_1, p_2/w_2, p_3/w_3, p_4/w_4) = (10, 6, 5, 4)$$



Simpul 7 ( $w_4$  diambil):  $W = 9 + 3 = 12 >$  kapasitas knapsack ( $K = 10$ )  
 Simpul 7 langsung dimatikan.

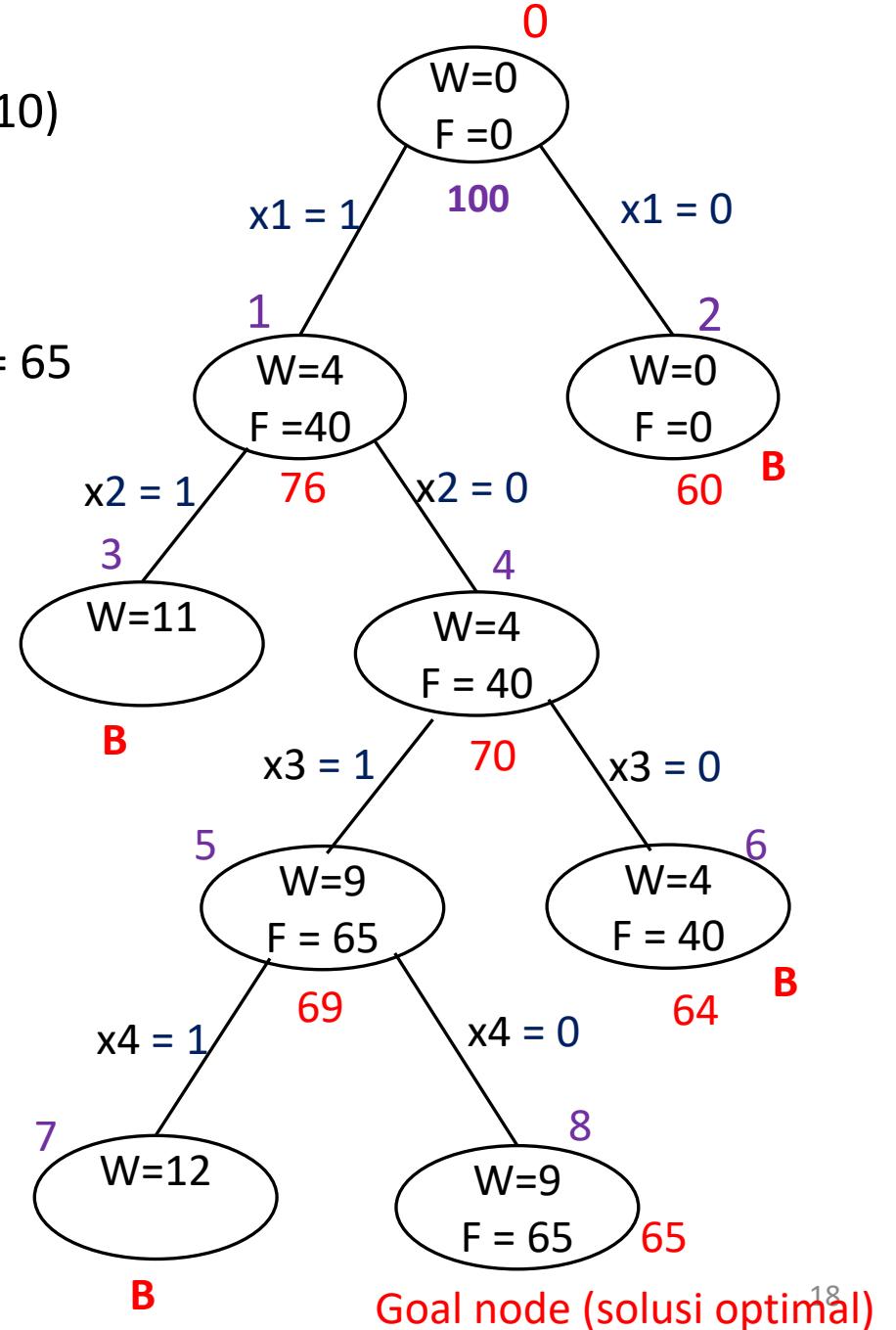
Simpul 8 ( $w_4$  tidak diambil):  $W = 9 + 0 = 9; F = 65 + 0 = 65$

$$\hat{c}(8) = F + (K - W)p_5/w_5 = 65 + (10 - 9)(0) = 65$$

Simpul 8 adalah simpul solusi (*goal node*)  
 dan merupakan solusi optimal

Semua simpul hidup yang *cost*-nya lebih  
 kecil dari 65 dibunuh  
 (simpul 2 dan simpul 6 dibunuh)

Solusi optimal:  $X = (1, 0, 1, 0), F = 65$



**Contoh 8.** Diberikan 6 buah objek sbb:

$$(w_1, p_1) = (100, 40); \quad (w_2, p_2) = (50, 35); \quad (w_3, p_3) = (45, 18);$$

$$(w_4, p_4) = (20, 4); \quad (w_5, p_5) = (10, 10); \quad (w_6, p_6) = (5, 2);$$

dan sebuah *knapsack* dengan kapasitas  $K = 100$ . Solusi dengan algoritma *greedy*:

Properti objek				Greedy by			Solusi Optimal
$i$	$w_i$	$p_i$	$p_i/w_i$	<i>profit</i>	<i>weight</i>	<i>density</i>	
1	100	40	0,4	1	0	0	0
2	50	35	0,7	0	0	1	1
3	45	18	0,4	0	1	0	1
4	20	4	0,2	0	1	1	0
5	10	10	1,0	0	1	1	0
6	5	2	0,4	0	1	1	0 <b>1</b>
Total bobot				100	80	85	100
Total keuntungan				40	34	51	55

- Ketiga strategi gagal memberikan solusi optimal!

1. Urutkan objek-objek berdasarkan  $p_i/w_i$  yang menurun

i	w <sub>i</sub>	p <sub>i</sub>	P <sub>i</sub> /w <sub>i</sub>	Greedy by density
5	10	10	1.0	1 (W=10)
2	50	35	0.7	1 (W=60)
1	100	40	0.4	0
3	45	18	0.4	0
6	5	2	0.4	1 (W=65)
4	20	4	0.2	1 (W=85)
Total profit		51		

2. Bangkitkan simpul dgn cost sbb:

$$\hat{c}(0) = 0 + (100 - 0)1 = 100$$

$$x_5=1: \hat{c}(1) = 10 + (100 - 10)0.7 = 10 + 63 = 73$$

$$x_5=0: \hat{c}(2) = 0 + (100 - 0)0.7 = 70$$

$$x_2=1: \hat{c}(3) = 45 + (100 - 60)0.4 = 45 + 16 = 61$$

$$x_2=0: \hat{c}(4) = 10 + (100 - 10)0.4 = 10 + 36 = 46$$

$$x_2=1: \hat{c}(5) = 35 + (100 - 50)0.4 = 35 + 20 = 55$$

$$x_2=0: \hat{c}(6) = 0 + (100 - 0)0.4 = 0 + 40 = 40$$

$$x_1=0: \hat{c}(8) = 45 + (100 - 60)0.4 = 45 + 16 = 61$$

$$x_3=0: \hat{c}(10) = 45 + (100 - 60)0.4 = 45 + 16 = 61$$

$$x_6=1: \hat{c}(11) = 47 + (100 - 65)0.2 = 47 + 7 = 54$$

$$x_6=0: \hat{c}(12) = 45 + (100 - 60)0.2 = 45 + 8 = 53$$

$$x_1=0: \hat{c}(14) = 35 + (100 - 50)0.4 = 35 + 20 = 55$$

$$x_3=1: \hat{c}(15) = 53 + (100 - 95)0.4 = 53 + 2 = 55$$

$$x_3=0: \hat{c}(16) = 35 + (100 - 50)0.4 = 35 + 20 = 55$$

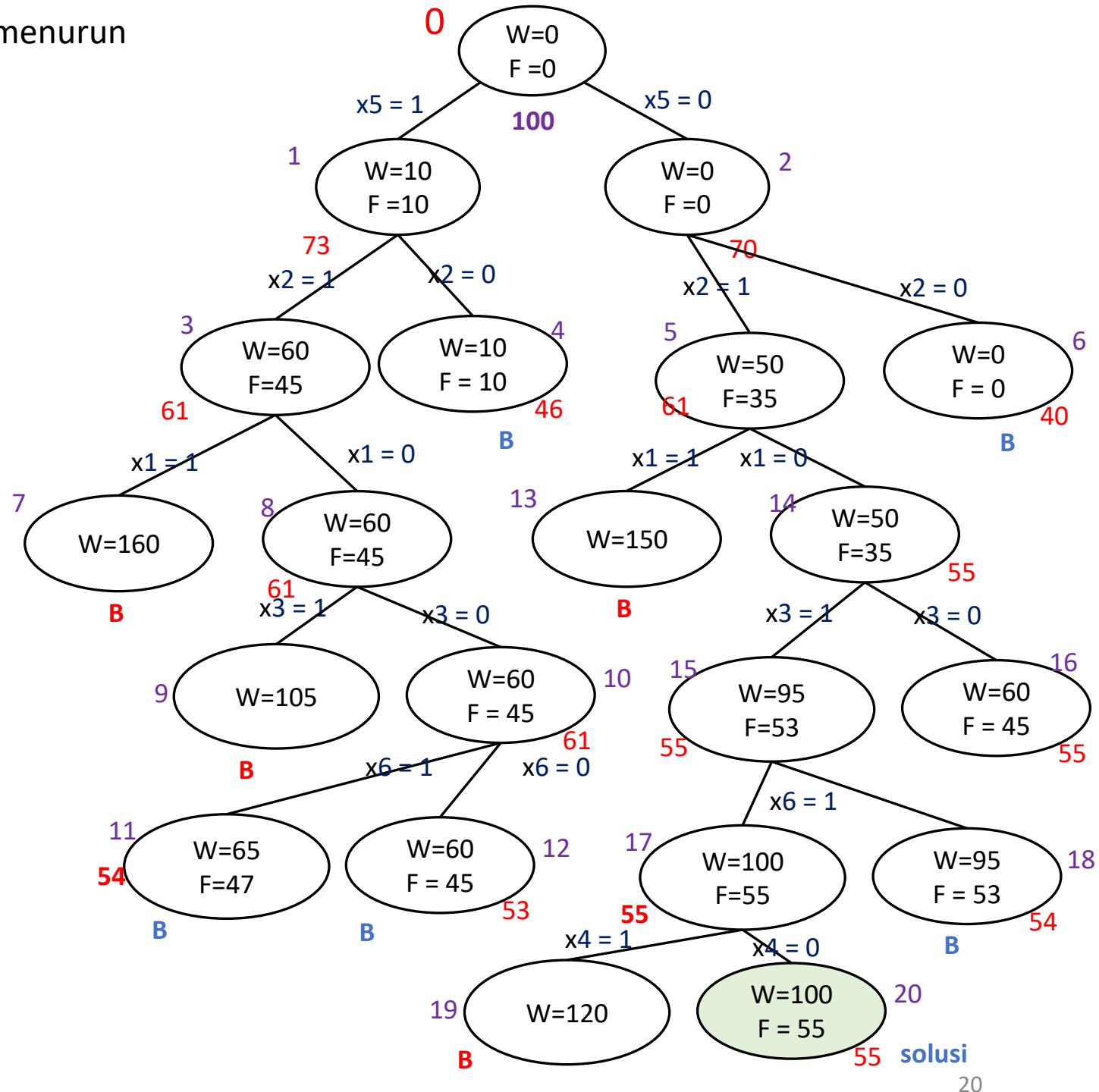
$$x_6=1: \hat{c}(17) = 55 + (100 - 100)0.2 = 55$$

$$x_6=0: \hat{c}(18) = 53 + (100 - 95)0.2 = 54$$

$$x_4=0: \hat{c}(20) = 55 + (100 - 100)0 = 55$$

$$x_6=1: \hat{c}(21) = 47 + (100 - 65)0.2 = 47 + 7 = 54$$

$$x_6=0: \hat{c}(22) = 45 + (100 - 60)0.2 = 45 + 8 = 53$$



TAMAT