# Algoritma Branch & Bound

(Bagian 4)

Bahan Kuliah IF2211 Strategi Algoritma

Oleh: Rinaldi Munir

Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika ITB
2021

# *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

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

## **Penyelesaian**:

- *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 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.

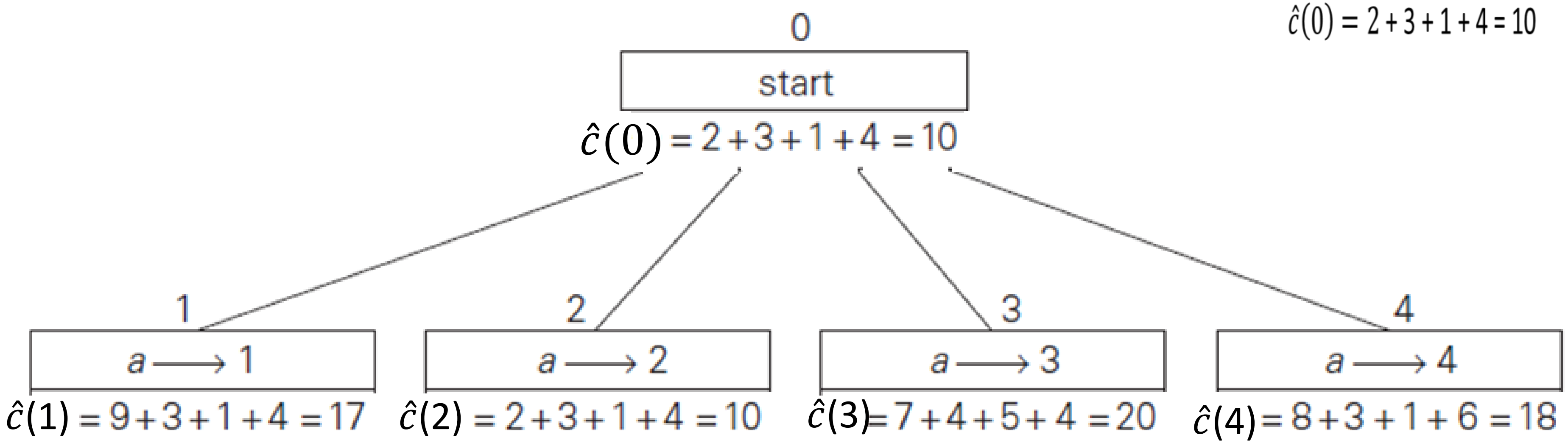
1. *Cost* untuk simpul akar:

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

2. Bangkitkan anak-anak dari simpul akar:

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

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

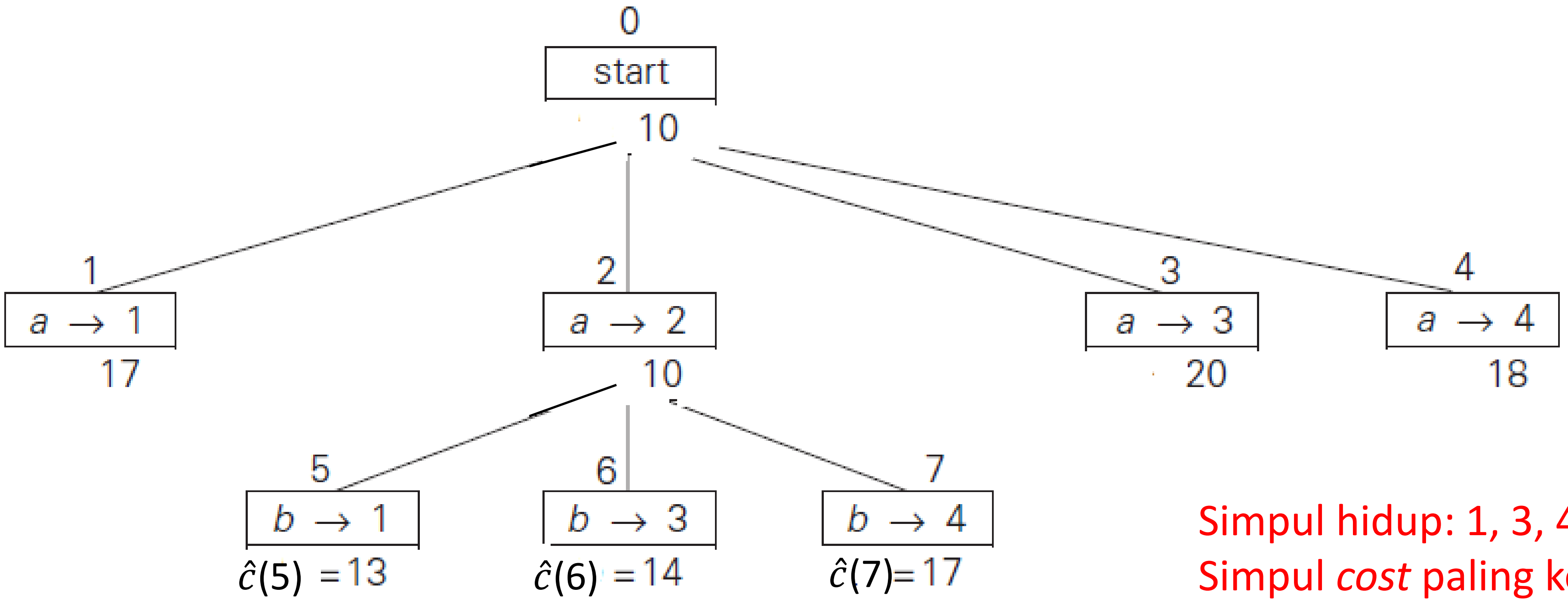



$$C = \begin{array}{c} \begin{array}{cccc} Job\,1 & Job\,2 & Job\,3 & Job\,4 \end{array} \\ \begin{bmatrix} 9 & 2 & 7 & 8 \\ 6 & 4 & 3 & 7 \\ 5 & 8 & 1 & 8 \\ 7 & 6 & 9 & 4 \end{bmatrix} \end{array} \begin{array}{l} \\ \text{Orang } a \\ \text{Orang } b \\ \text{Orang } c \\ \text{Orang } d \end{array}$$

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

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

(Sumber gambar: Levitin, 2003)




Simpul hidup: 1, 3, 4, 5, 6, dan 7

Simpul *cost* paling kecil: 5

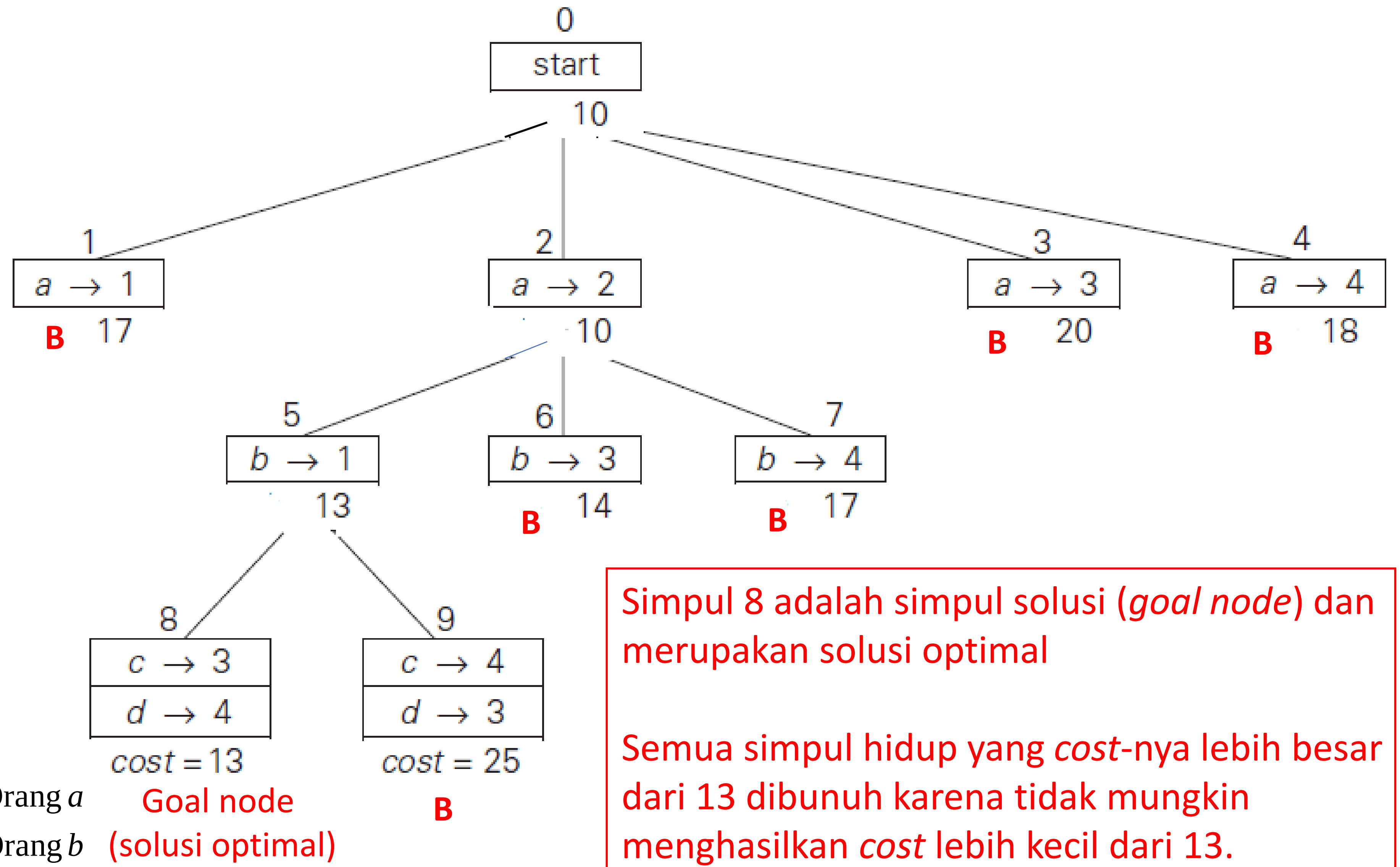
Simpul-E sekarang: 5

$$C = \begin{array}{c} \\ \end{array}\left[\begin{array}{cccc} Job\,1 & Job\,2 & Job\,3 & Job\,4 \\ 9 & 2 & 7 & 8 \\ 6 & 4 & 3 & 7 \\ 5 & 8 & 1 & 8 \\ 7 & 6 & 9 & 4 \end{array}\right]\begin{array}{l} \\ \text{Orang } a \\ \text{Orang } b \\ \text{Orang } c \\ \text{Orang } d \end{array}$$

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

$$C = \left[\begin{array}{cccc} Job\,1 & Job\,2 & Job\,3 & Job\,4 \\ 9 & 2 & 7 & 8 \\ 6 & 4 & 3 & 7 \\ 5 & 8 & 1 & 4 \\ 7 & 6 & 9 & 4 \end{array}\right]\begin{array}{l} \\ \text{Orang } a \\ \text{Orang } b \\ \text{Orang } c \\ \text{Orang } d \end{array}$$

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




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

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

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

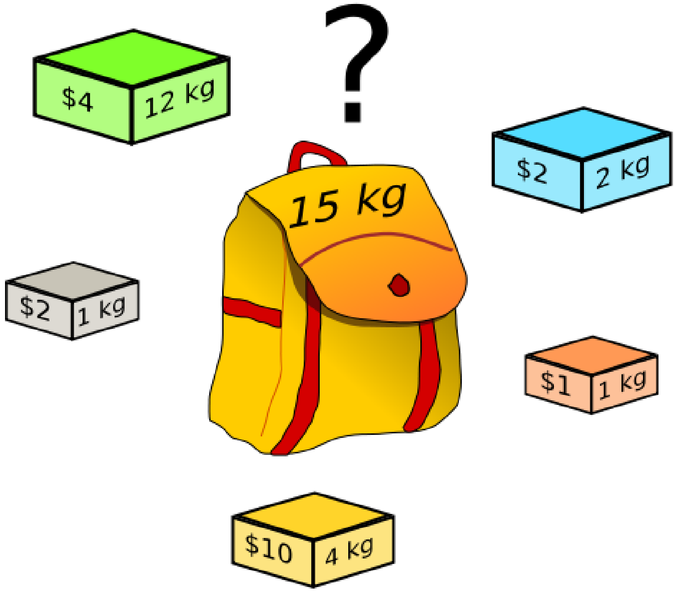
- **Persoalan:** Diberikan *n* buah objek dan sebuah *knapsack* dengan kapasitas bobot *K*. Setiap objek memiliki properti bobot (*weigth*) $w_i$ dan keuntungan(*profit*) $p_i$. Bagaimana cara memilih objek-objek yang dimasukkan ke dalam *knapsack* sedemikian sehingga diperoleh total keuntungan yang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

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, $i = 1, 2, \ldots, n$



- 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 <u>*cost* paling besar</u>.

- Agar pencarian solusi lebih mangkus, maka objek-objek diurutkan berdasarkan $p_i/w_i$ yang menurun (dari besar ke kecil) sebagai berikut:

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

- 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, …*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

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

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

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

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

W=0
F =0

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

(Sumber: Levitin, 2003)

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

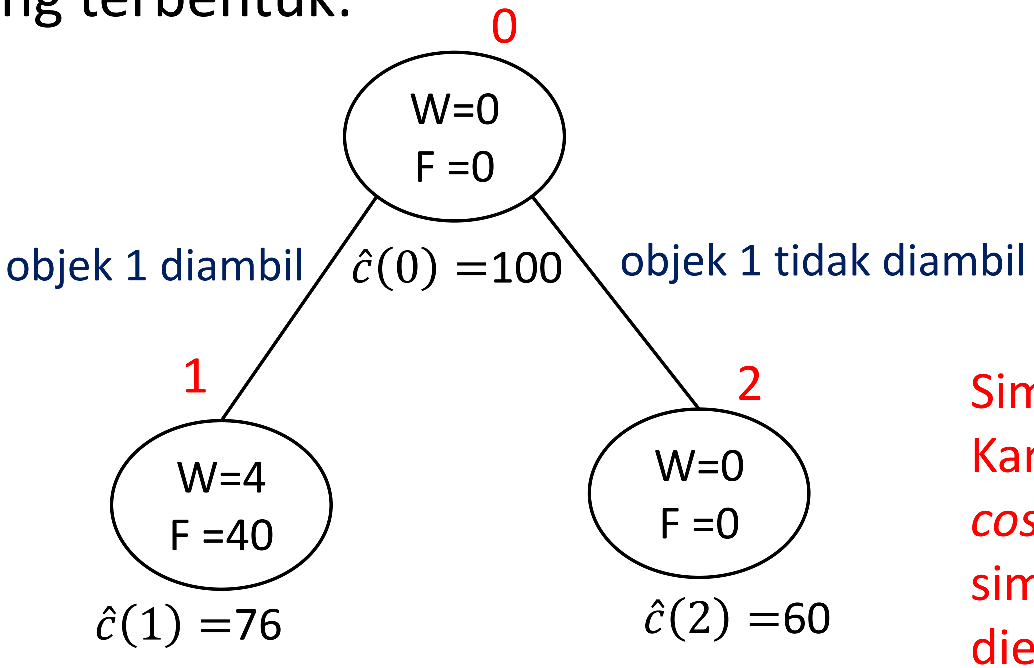
- Simpul 1 (objek 1 diambil): $W = 0 + 4 = 4$; $F = 0 + 40 = 40$

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

- 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)

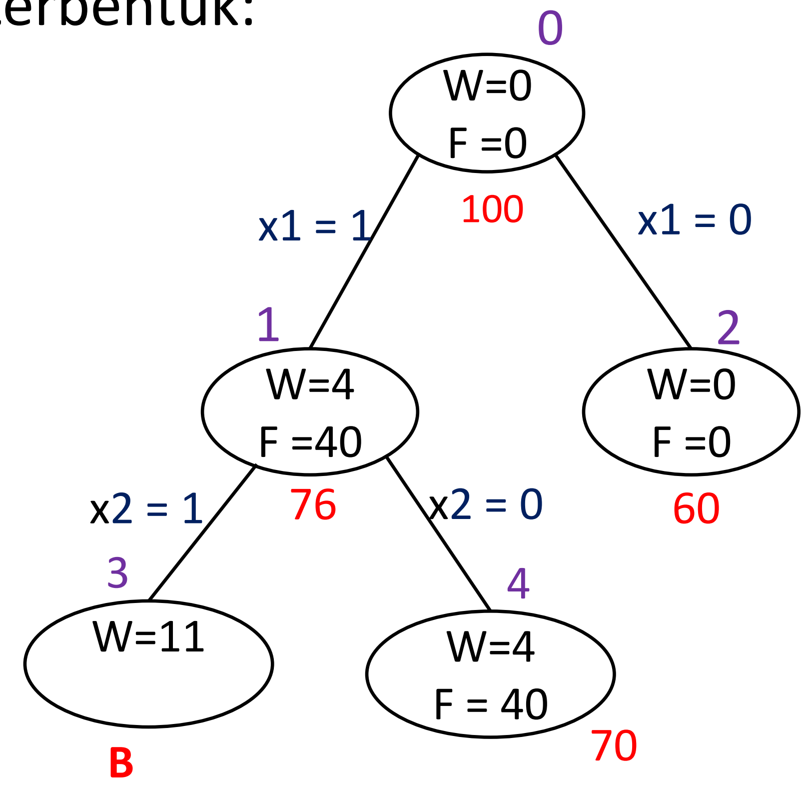
  Simpul 3 langsung dimatikan (**B**).

- 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



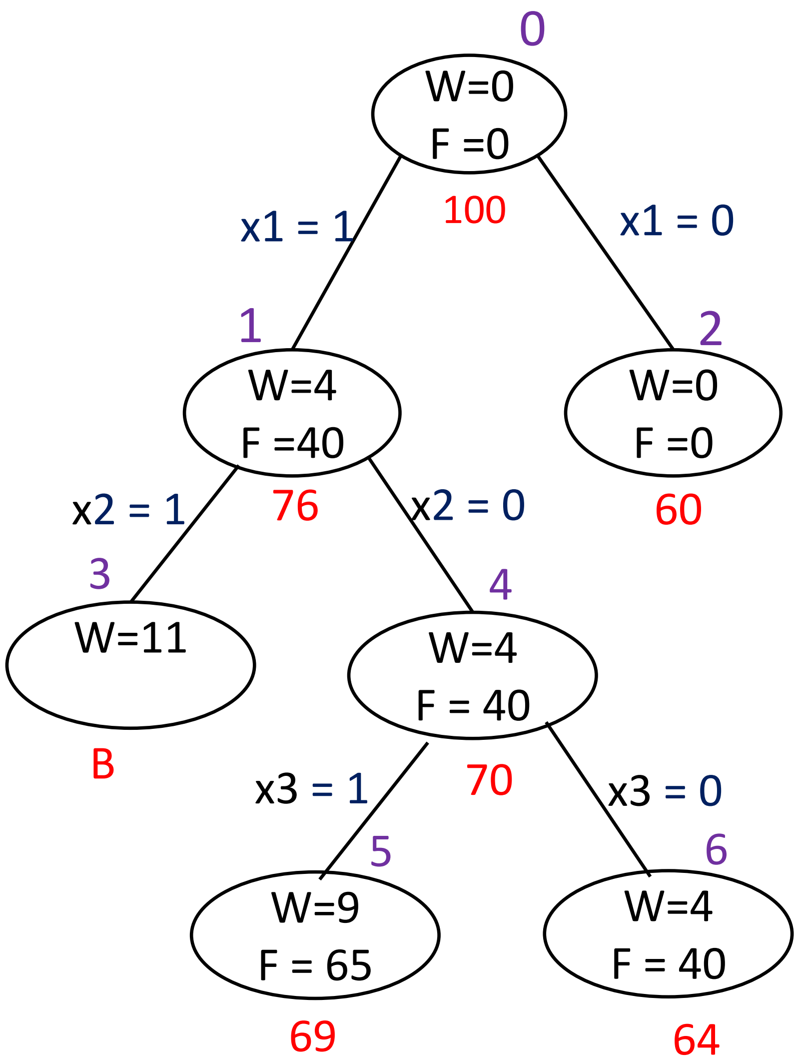
Simpul 5 (objek 3  diambil): $W$ = 4 + 5 = 9; $F$ = 40 + 25 = 65

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

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



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

Simpul 7 langsung dimatikan.

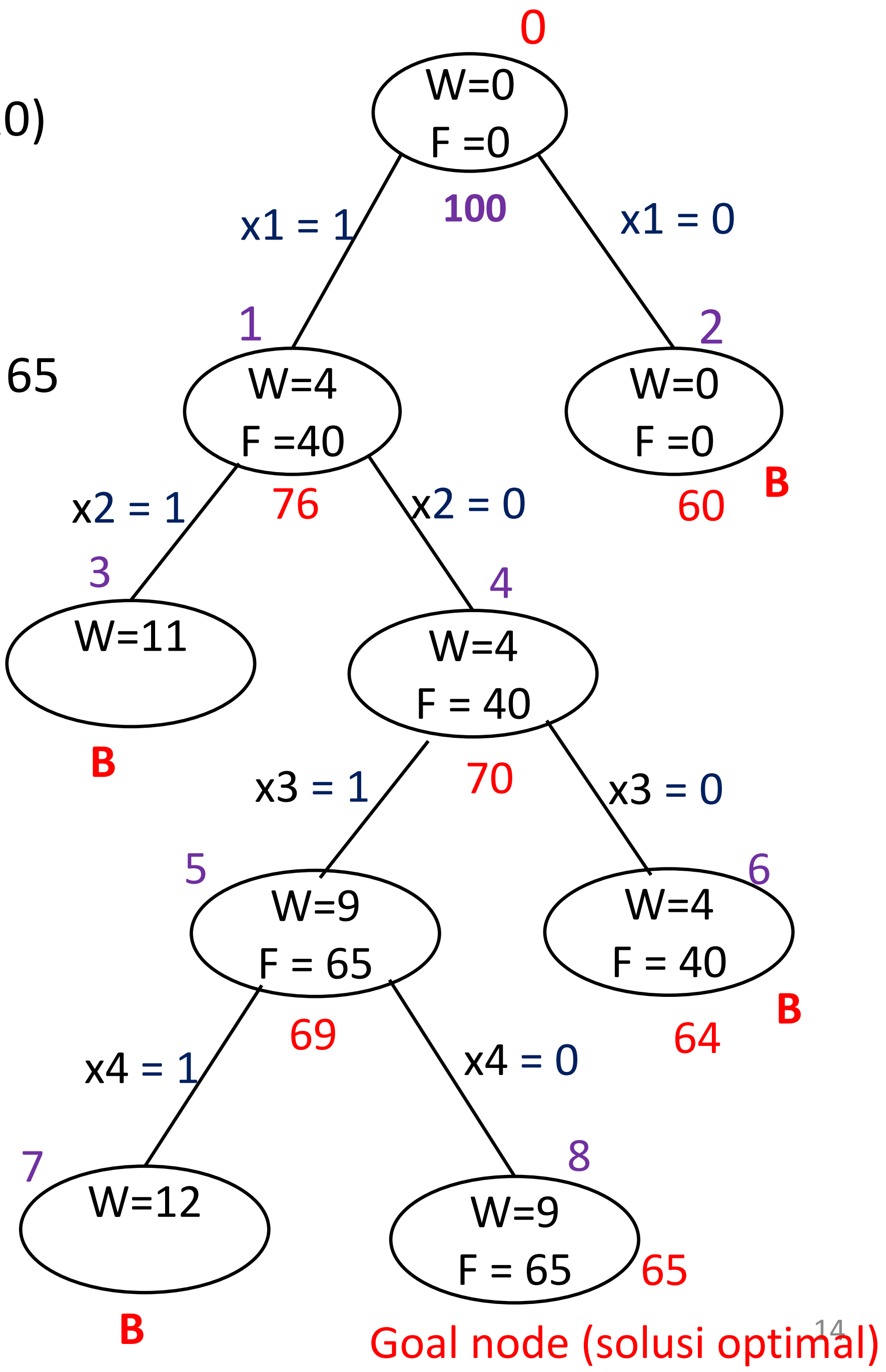
Simpul 8 ($w_4$ tidak diambil): $W$ = 9 + 0 = 9; F = 65 + 0 = 65

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

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



# TAMAT