Program Dinamis (*Dynamic Programming*)

Bagian 1

Bahan Kuliah IF2211 Strategi Algoritma

Program Studi Teknik Informatika
STEI-ITB

Program Dinamis

- **Program Dinamis** (*dynamic programming*):
  - metode pemecahan masalah dengan cara menguraikan solusi menjadi sekumpulan tahapan (*stage*)
  - sedemikian sehingga solusi persoalan dapat dipandang sebagai serangkaian keputusan yang saling berkaitan.

- Kata “program” tidak ada kaitannya dengan pemrograman
- Istilah “dinamis” muncul karena pencarian solusinya melakukan perhitungan dengan menggunakan tabel (yang dapat berkembang)

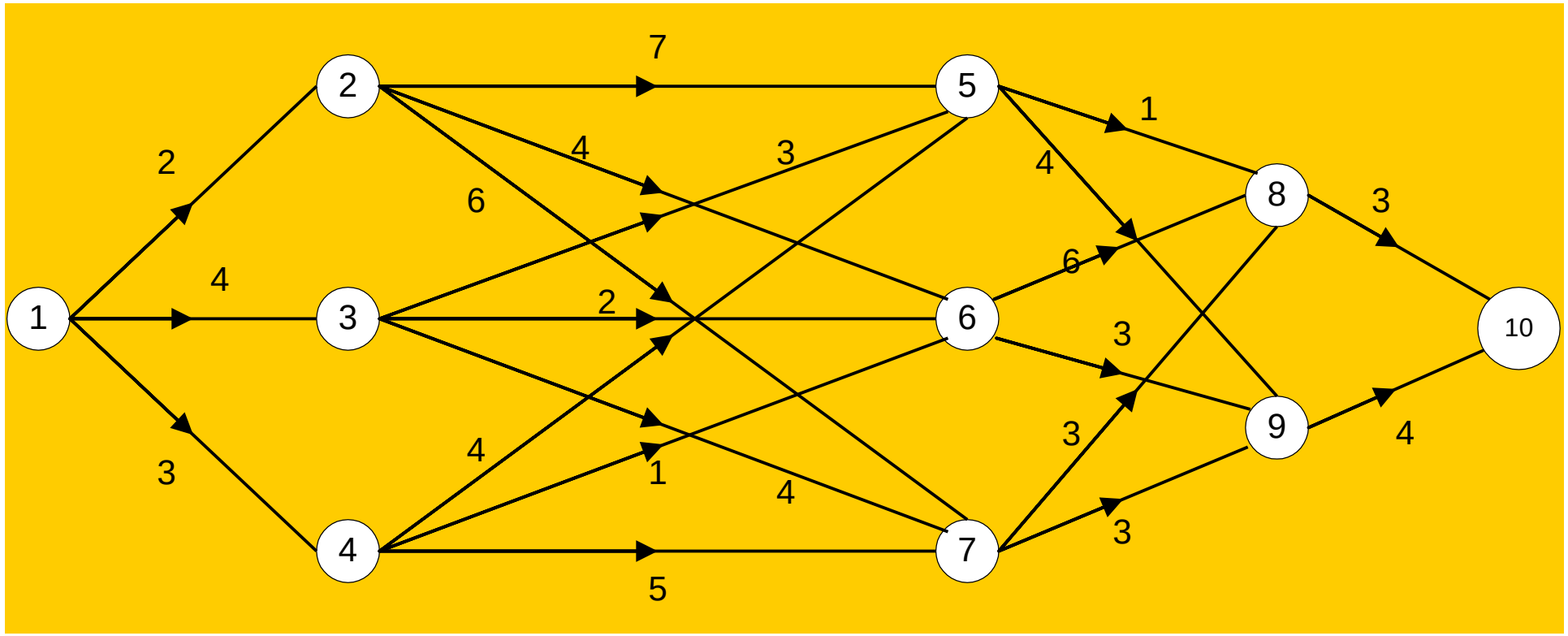
- Program dinamis digunakan untuk menyelesaikan persoalan-persoalan optimasi (maksimasi atau minimisasi)

- Perbedaan Algoritma *Greedy* dengan Program Dinamis:

  → **Greedy**: hanya satu rangkaian keputusan yang dihasilkan

  → **Program dinamis**: lebih dari satu rangkaian keputusan yang dipertimbangkan.

Tinjau graf di bawah ini. Kita ingin menemukan **lintasan terpendek** dari 1 ke 10.



Greedy: dari setiap simpul, ambil sisi dengan bobot terkecil ke simpul berikutnya
Solusi *greedy*: 1 – 2 – 6 – 9 – 10 dengan *cost* = 2 + 4 + 3 + 4 = 13 → tidak optimal!

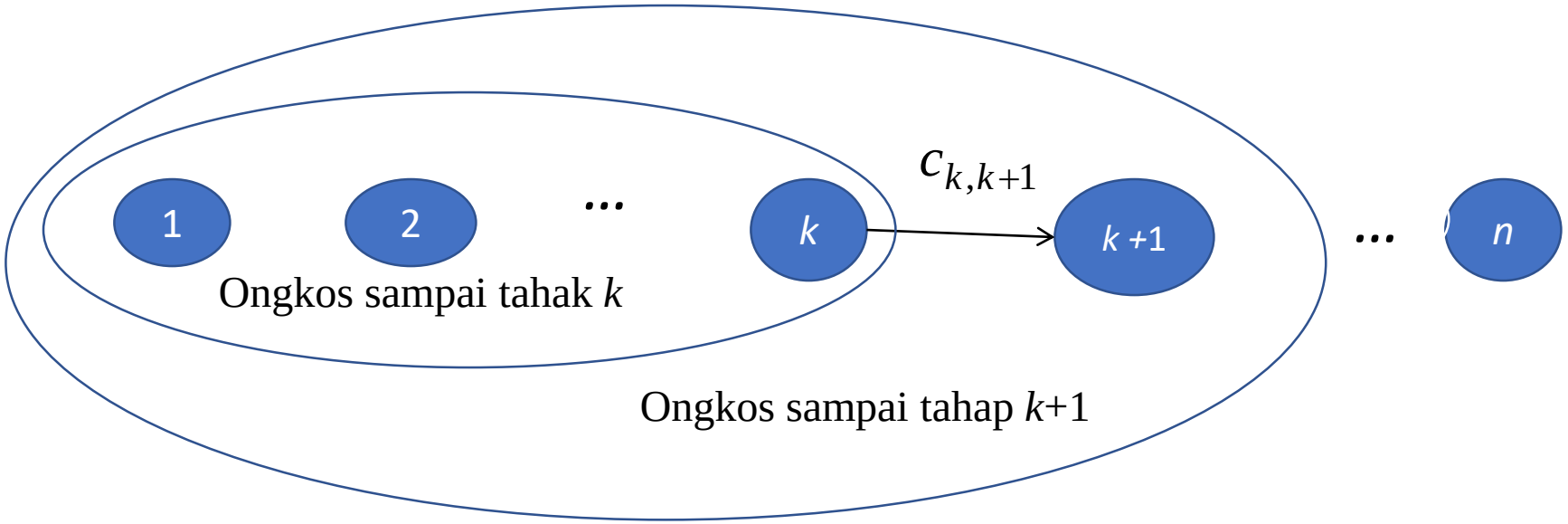
Prinsip Optimalitas

- Pada program dinamis, rangkaian keputusan yang optimal dibuat dengan menggunakan **Prinsip Optimalitas**.

- Prinsip Optimalitas: *jika solusi total optimal, maka bagian solusi sampai tahap ke-k juga optimal*.

- Prinsip optimalitas berarti bahwa jika kita bekerja dari tahap *k* ke tahap *k* + 1, kita dapat menggunakan hasil optimal dari tahap *k* tanpa harus kembali ke tahap awal.

- Ongkos pada tahap k +1 = (ongkos yang dihasilkan pada tahap k) + (ongkos dari tahap k ke tahap k + 1, atau $c_{k,k+1}$)

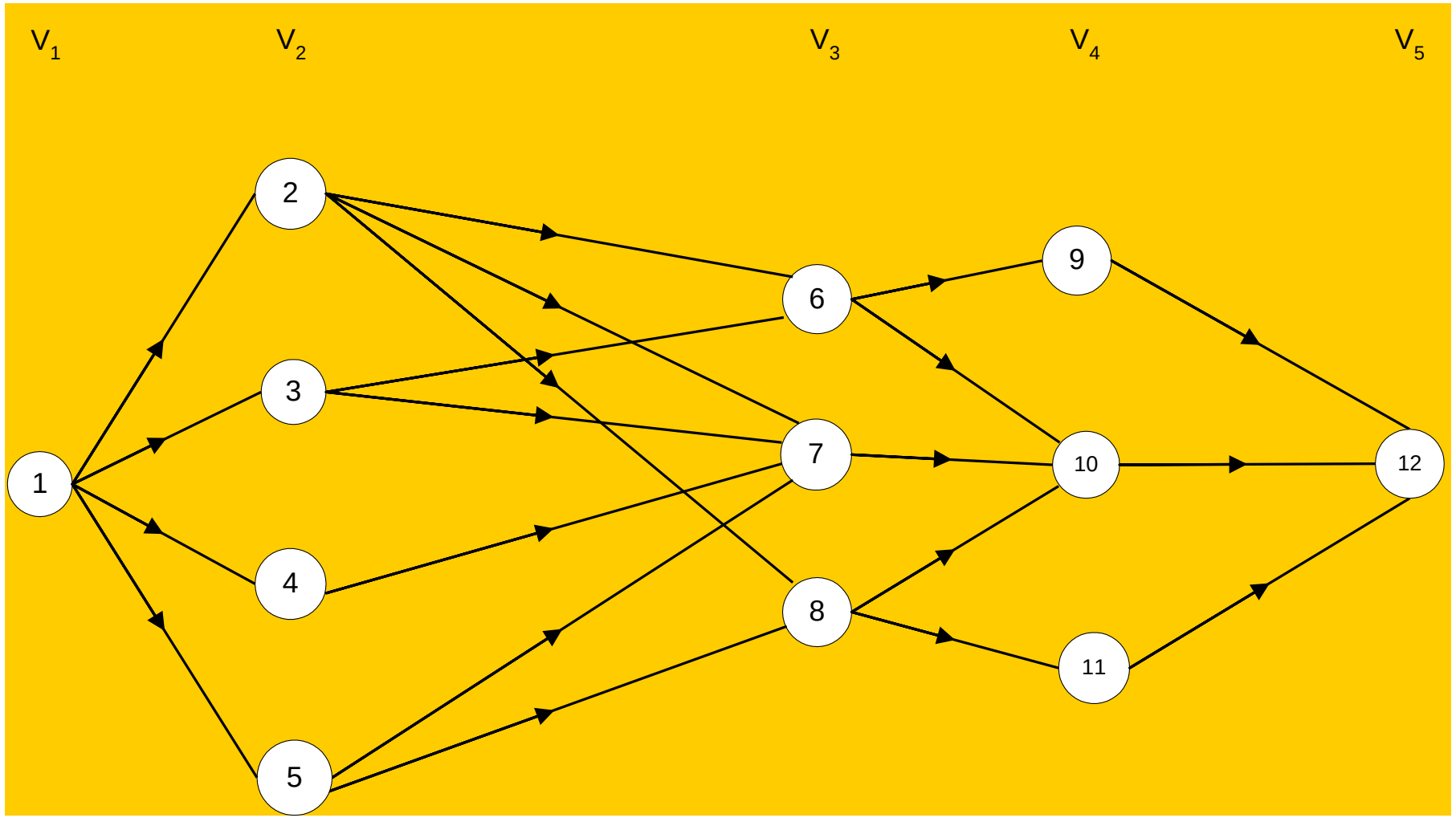


Karakteristik Persoalan dengan Program Dinamis

1. Persoalan dapat dibagi menjadi beberapa tahap (*stage*), yang pada setiap tahap hanya diambil satu keputusan.

2. Masing-masing tahap terdiri dari sejumlah status (*state*) yang berhubungan dengan tahap tersebut. Secara umum, status merupakan bermacam kemungkinan masukan yang ada pada suatu tahap.

Graf multitahap (*multistage graph*). Tiap simpul di dalam graf tersebut menyatakan status, sedangkan V_1, V_2, ... menyatakan tahap.



3. Hasil dari keputusan yang diambil pada setiap tahap ditransformasikan dari status yang bersangkutan ke status berikutnya pada tahap berikutnya.

4. Ongkos (*cost*) pada suatu tahap meningkat secara teratur (*steadily*) dengan bertambahnya jumlah tahapan.

5. Ongkos pada suatu tahap bergantung pada ongkos tahap-tahap yang sudah berjalan dan onkos dari tahap tersebut ke tahap berikutnya.

6. Adanya hubungan rekursif yang mengidentifikasikan keputusan terbaik untuk setiap status pada tahap k memberikan keputusan terbaik untuk setiap status pada tahap $k + 1$.

7. Prinsip optimalitas berlaku pada persoalan tersebut.

Dua pendekatan PD

Dua pendekatan yang digunakan dalam program dinamis:

1. Program dinamis maju (*forward* atau *up-down*)

   → Perhitungan dilakukan dari tahap 1, 2, …, n – 1, n

2. Program dinamis mundur (*backward* atau *bottom-up*)

   → Perhitungan dilakukan dari tahap n, n – 1, …, 2, 1.

Misalkan x_1, x_2, …, x_n menyatakan peubah (*variable*) keputusan yang harus ditentukan masing-masing untuk tahap 1, 2, …, n. Maka,

1. **Program dinamis maju**. Program dinamis bergerak mulai dari tahap 1, terus maju ke tahap 2, 3, dan seterusnya sampai tahap n.

   Rangkaian peubah keputusan adalah x_1, x_2, …, x_n.

2. **Program dinamis mundur**. Program dinamis bergerak mulai dari tahap n, terus mundur ke tahap $n - 1$, $n - 2$, dan seterusnya sampai tahap 1.

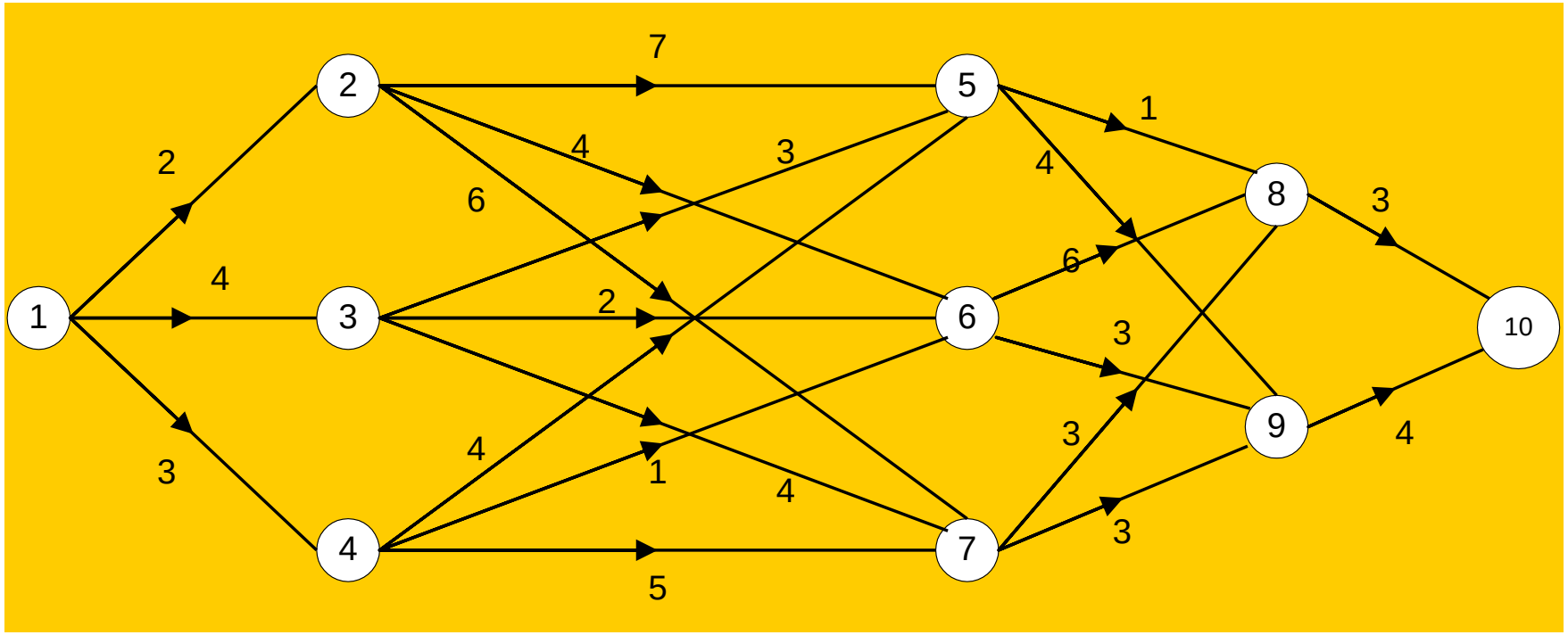
   Rangkaian peubah keputusan adalah x_n, x_{n-1}, …, x_1.

Langkah-langkah Pengembangan Algoritma Program Dinamis

1. Karakteristikkan struktur solusi optimal.
   - tahap, variable keputusan, status (state), dsb
2. Definisikan secara rekursif nilai solusi optimal.
   - hubungan nilai optimal suatu tahap dengan tahap sebelumnya
3. Hitung nilai solusi optimal secara maju atau mundur.
   - menggunakan tabel
4. Rekonstruksi solusi optimal (opsional).
   - rekonstruksi solusi secara mundur

Persoalan 1: Lintasan Terpendek (*Shortest Path*)

- Tentukan lintasan terpendek dari simpul 1 ke simpul 10:



How to find the shortest path?

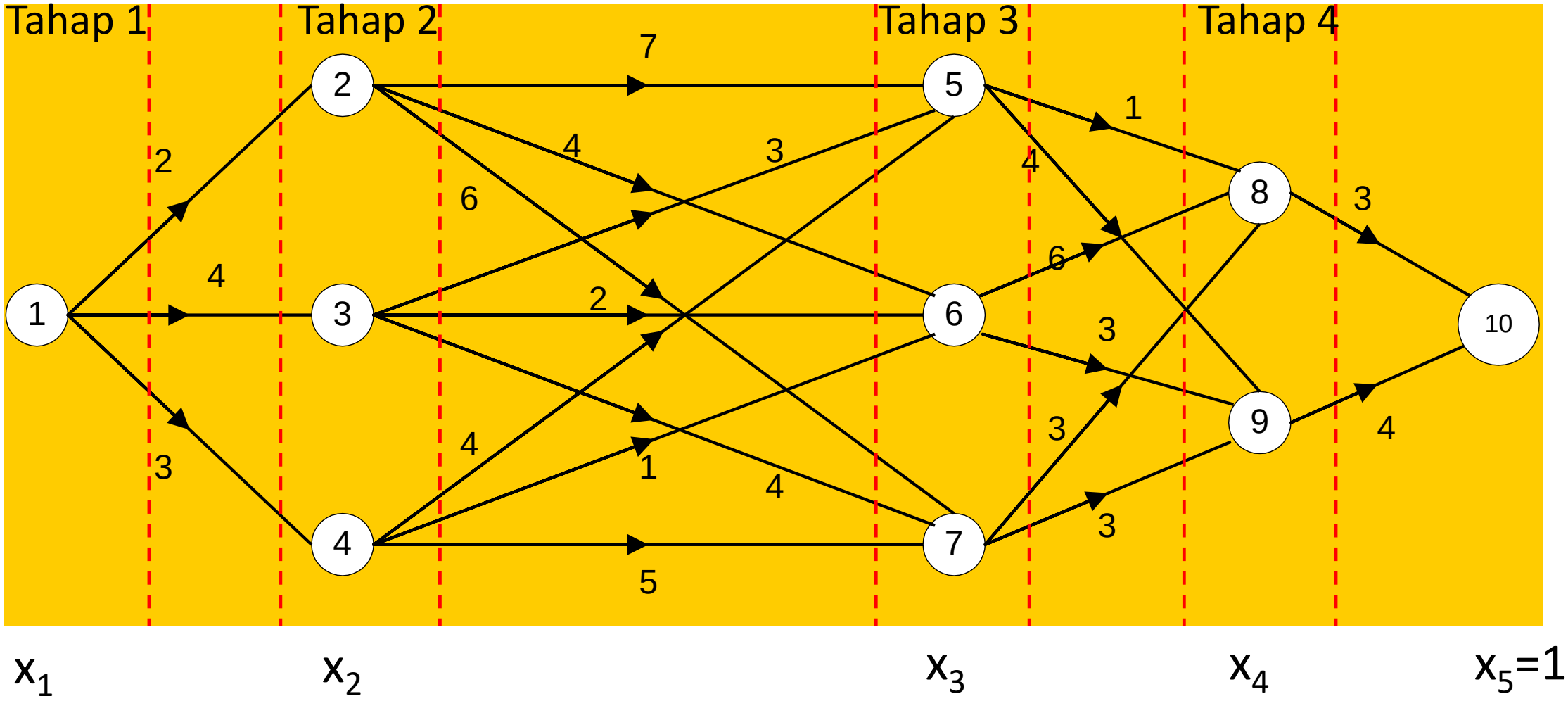
(a) Karakteristikkan struktur solusi optimal

- Misalkan x_1, x_2, x_3, x_4 adalah simpul-simpul yang dikunjungi pada tahap k (k = 1, 2, 3, 4). Tahap 5 tidak ada karena x_5=10
- Misalkan digunakan pendekatan program dinamis maju
- Maka rute yang dilalui adalah $x_1 \rightarrow x_2 \rightarrow x_3 \rightarrow x_4 \rightarrow x_5$=10

Pada persoalan ini,

- *Tahap* (k) adalah proses memilih simpul tujuan berikutnya (ada 4 tahap).

- *Status* (s) yang berhubungan dengan masing-masing tahap adalah simpul-simpul di dalam graf multi-tahap.


Tahap 1
Tahap 2
Tahap 3
Tahap 4
1
2
3
4
5
6
7
8
9
10
2
4
3
7
4
6
4
2
3
4
1
4
5
1
6
3
3
4
3
3
4
x1
x2
x3
x4
x5=10

(b) Definisikan hubungan rekursif solusi optimal

Relasi rekurens berikut menyatakan lintasan terpendek pada setiap tahap:

$$f_1(s) = c_{x1,s} \qquad \text{(basis)}$$
$$f_k(s) = \min \{f_{k-1}(x_k) + c_{xk,s}\} \qquad \text{(rekurens)}$$
$$k = 2, 3, 4$$

Keterangan:

a. x_k : peubah keputusan pada tahap k (k = 2, 3, 4).

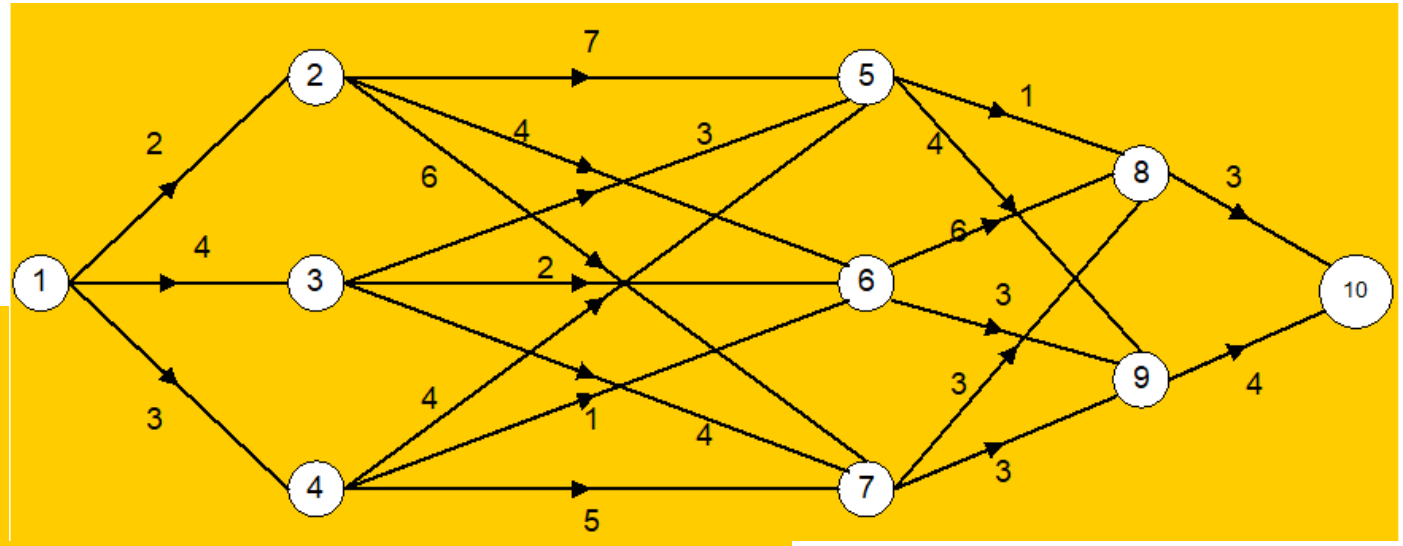
b. s : status pada setiap tahap

c. $c_{xk,s}$: bobot (*cost*) sisi dari ke x_k ke s

d. $f_k(s)$: nilai minimum dari $f_k(x_k, s)$

e. $f_{k-1}(x_k)$: nilai minimum tahap sebelumnya dari x_k ke s

(c)Hitung nilai solusi optimal



Tahap 1:

$$f_1(s) = c_{x1s}$$

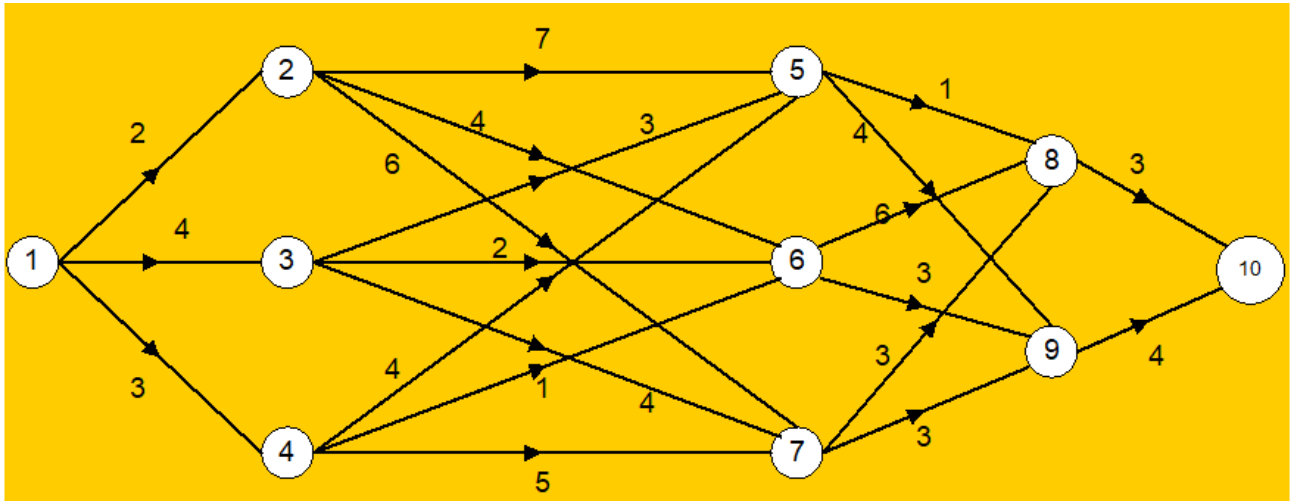
s	Solusi Optimum	
	$f_1(s)$	x_1^*
2	2	1
3	4	1
4	3	1

$f_1(s) = c_{x1,s}$

Catatan: x_k^* adalah nilai x_k yang meminimumkan f_s.

Tahap sebelumnya (Tahap 1)

s	Solusi Optimum	
	$f_1(s)$	x_1^*
2	2	1
3	4	1
4	3	1



Tahap 2:

$$f_2(s) = \min_{x_2}\{f_1(x_2) + c_{x_2 s}\}$$

x_2 / s	$f_2(s) = f_1(x_2) + c_{x2,s}$			Solusi Optimum	
	2	3	4	$f_2(s)$	x_2^*
5	9	7	7	7	3 atau 4
6	6	6	4	4	4
7	8	8	8	8	2, 3, 4

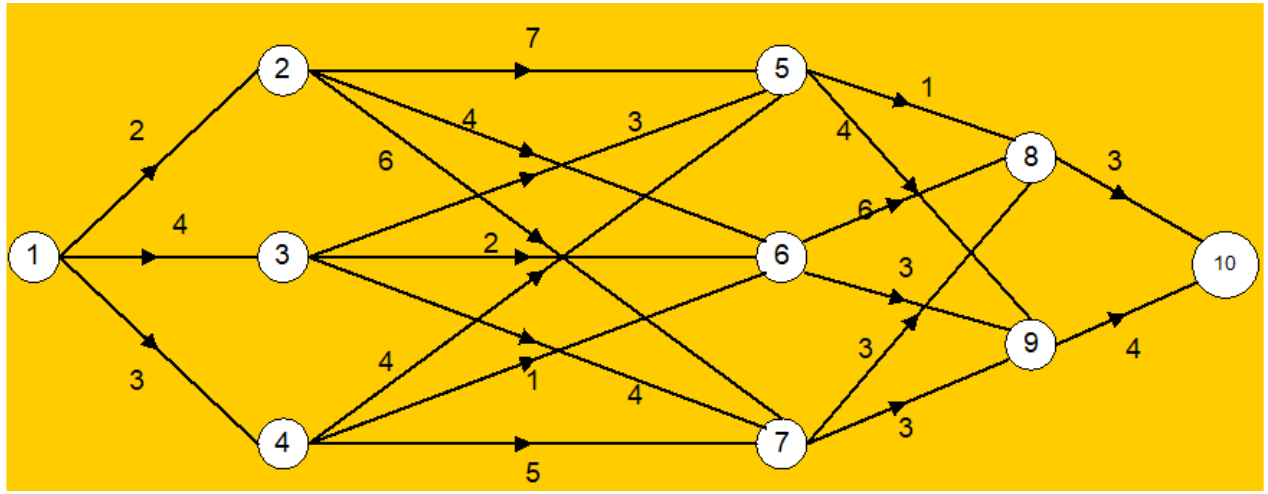
$f_k(s) = \min\{f_{k-1}(x_k) + c_{xk,s}\}$

Tahap sebelumnya (Tahap 2)

Tahap 2:

$$f_2(s) = \min_{x_2}\{f_1(x_2) + c_{x_2 s}\}$$

x_2 / s	$f_2(s) = f_1(x_2) + c_{x2,s}$			Solusi Optimum	
	2	3	4	$f_2(s)$	x_2^*
5	9	7	7	7	3 atau 4
6	6	6	4	4	4
7	8	8	8	8	2, 3, 4



Tahap 3:

$$f_3(s) = \min_{x_3}\{f_2(x_3) + c_{x_3 s}\}$$

x_3 / s	$f_3(s) = f_2(x_3) + c_{x3,s}$			Solusi Optimum	
	5	6	7	$f_3(s)$	x_3^*
8	8	10	11	8	5
9	11	7	11	7	6

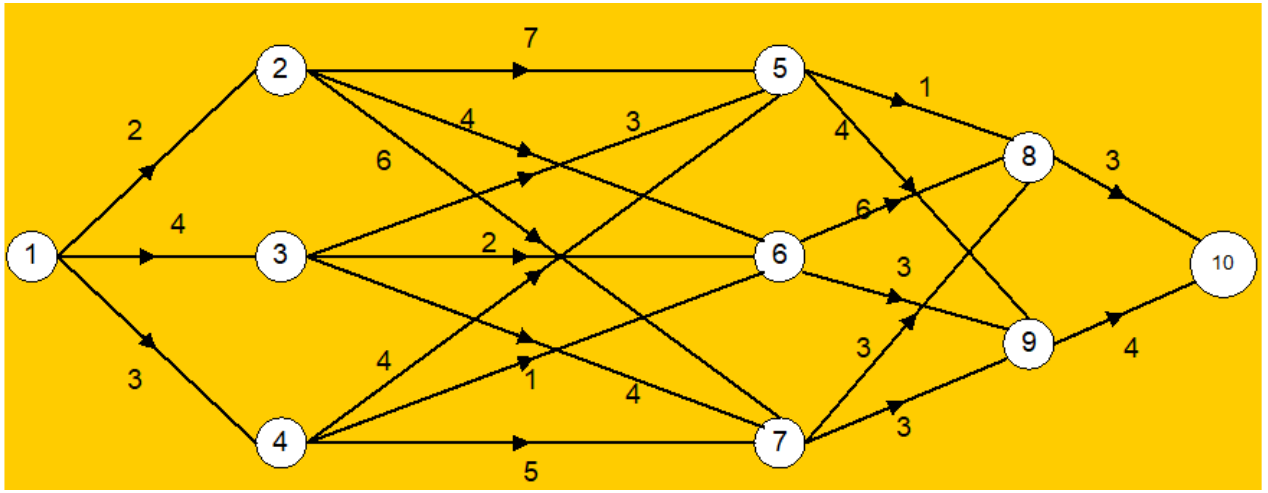
$$f_k(s) = \min\{f_{k-1}(x_k) + c_{x_k s}\}$$

Tahap sebelumnya (Tahap 3)

Tahap 3:

$$f_3(s) = \min_{x_3}\{f_2(x_3) + c_{x_3 s}\}$$

x_3 / s	$f_3(s) = f_2(x_3) + c_{x3,s}$			Solusi Optimum	
	5	6	7	$f_3(s)$	x_3^*
8	8	10	11	8	5
9	11	7	11	7	6



Tahap 4:

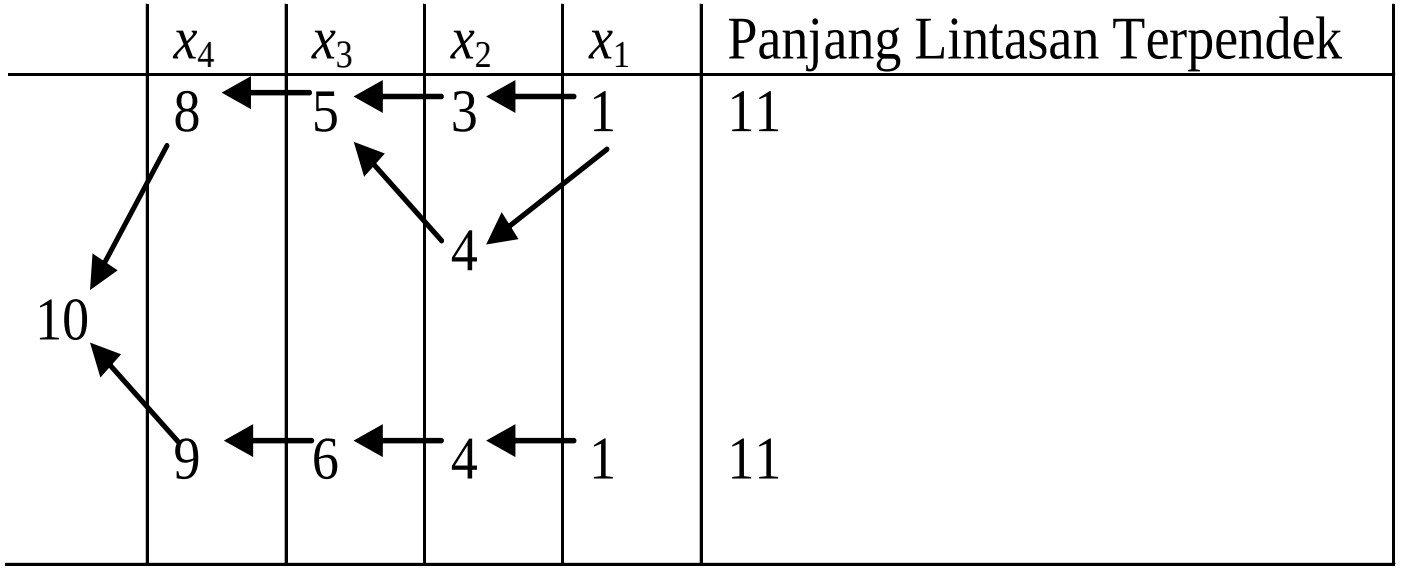
$$f_4(s) = \min_{x_4}\{f_3(x_4) + c_{x_4 s}\}$$

x_4 / s	$f_4(s) = f_3(x_4) + c_{x4,s}$		Solusi Optimum	
	8	9	$f_4(s)$	x_4^*
10	11	11	11	8 atau 9

$f_k(s) = \min\{f_{k-1}(x_k) + c_{xk,s}\}$

(d) Rekonstruksi solusi optimal

Solusi optimum dapat dibaca pada tabel di bawah ini:



Jadi ada tiga lintasan terpendek dari 1 ke 10, yaitu

$$1 \to 3 \to 5 \to 8 \to 10$$
$$1 \to 4 \to 5 \to 8 \to 10$$
$$1 \to 4 \to 6 \to 9 \to 10$$

yang mana panjang ketiga lintasan tersebut sama, yaitu 11.

Tahap 2:

$$f_2(s) = \min_{x_2}\{f_1(x_2) + c_{x_2 s}\}$$

x_2 / s	$f_2(s) = f_1(x_2) + c_{x2,s}$			Solusi Optimum	
	2	3	4	$f_2(s)$	x_2^*
5	9	7	7	7	3 atau 4
6	6	6	4	4	4
7	8	8	8	8	2, 3, 4

Tahap 3:

$$f_3(s) = \min_{x_3}\{f_2(x_3) + c_{x_3 s}\}$$

x_3 / s	$f_3(s) = f_2(x_3) + c_{x3,s}$			Solusi Optimum	
	5	6	7	$f_3(s)$	x_3^*
8	8	10	11	8	5
9	11	7	11	7	6

Tahap 4:

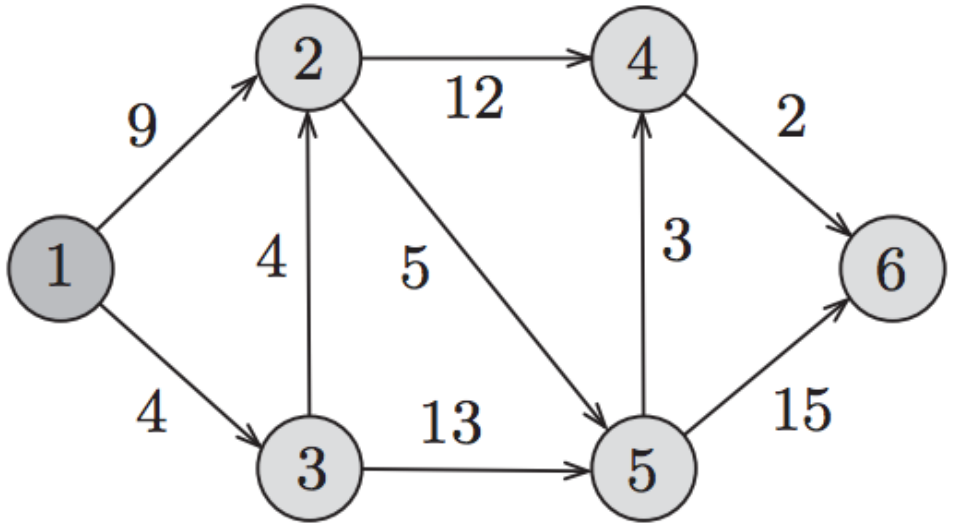
$$f_4(s) = \min_{x_4}\{f_3(x_4) + c_{x_4 s}\}$$

x_4 / s	$f_4(s) = f_3(x_4) + c_{x4,s}$		Solusi Optimum	
	8	9	$f_4(s)$	x_4^*
10	11	11	11	8 atau 9

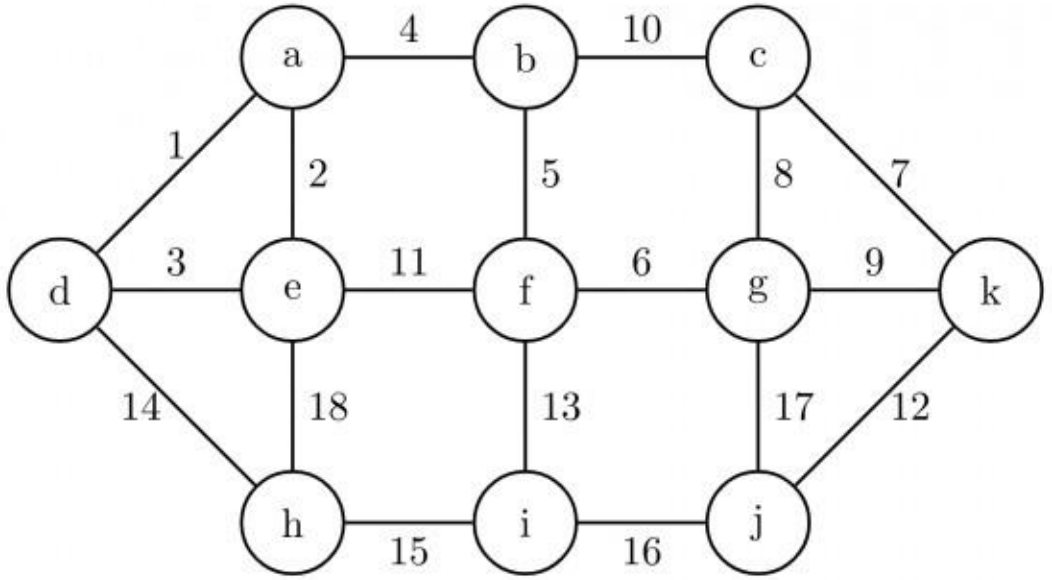
Latihan

1. Selesaikan persoalan *shortest path* tersebut dengan program dinamis mundur.

2. Carilah lintasan terpendek dari 1 ke 6 pada graf (a) dan dari d ke k pada graf (b):



(a)

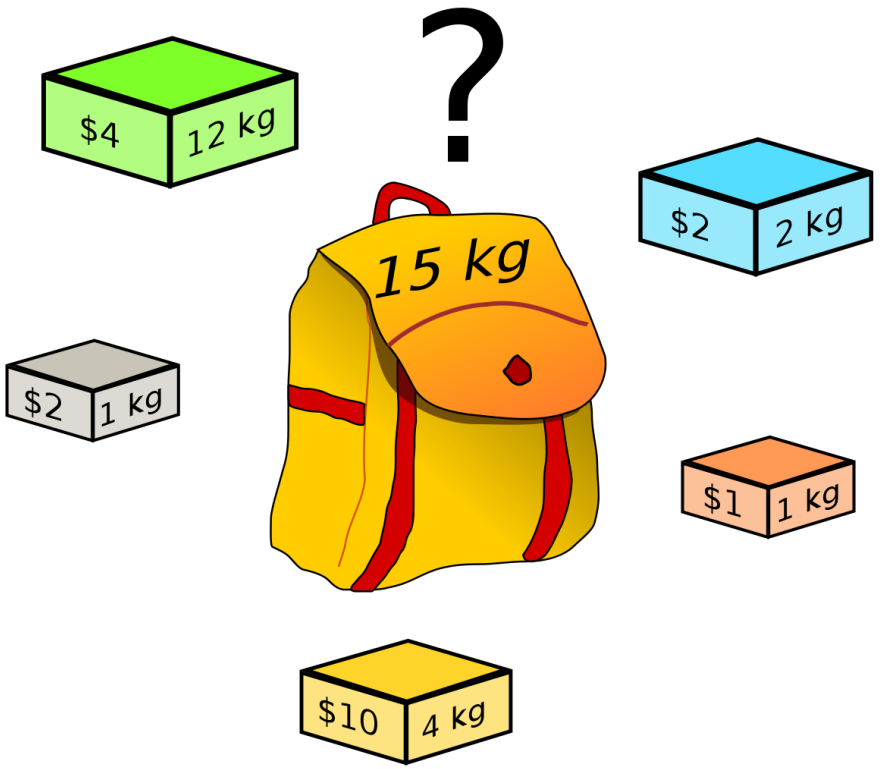


(b)

Persoalan 2: *Integer Knapsack*

Diberikan sebuah *knapsack* dengan kapasitas *M*. Terdapat *n* buah objek, setiap objek memiliki bobot w_i dan keuntungan p_i.

Bagaimana cara memilih objek-objek yang dimasukkan ke dalam *knapsack* sehingga total keuntungan yang diperoleh maksimal.



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

dengan kendala (*constraint*)

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

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

Contoh: $n = 3$

$M = 5$

Barang ke-i	w_i	p_i
1	2	65
2	3	80
3	1	30

- Pada persoalan ini,
  1. Tahap (*k*) adalah proses memasukkan objek ke dalam *knapsack* (*knapsack*) (pada contoh di atas ada 3 tahap).

  2. Status (*y*) menyatakan kapasitas muat *knapsack* yang tersisa setelah memasukkan objek pada tahap sebelumnya.

- Dari tahap ke-1, kita masukkan objek ke-1 ke dalam *knapsack* untuk setiap satuan kapasitas *knapsack* sampai batas kapasitas maksimumnya.

- Karena kapasitas *knapsack* adalah bilangan bulat, maka pendekatan ini praktis.

- Misalkan ketika memasukkan objek pada tahap *k*, kapasitas muat *knapsack* sekarang adalah $y - w_k$.

- Untuk mengisi kapasitas sisanya, kita menerapkan prinsip optimalitas dengan mengacu pada nilai optimum dari tahap sebelumnya untuk kapasitas sisa $y - w_k$

- Nilai optimum pada tahpa sebelumnya adalah $f_{k-1}(y - w_k)$.

- Selanjutnya, kita bandingkan:

  nilai keuntungan pengisian pada tahap k (yaitu p_k) + nilai $f_{k-1}(y - w_k)$

  dengan

  keuntungan pengisian hanya $k - 1$ objek, $f_{k-1}(y)$.

- Jika $p_k + f_{k-1}(y - w_k)$ lebih kecil dari $f_{k-1}(y)$, maka objek yang ke-k tidak dimasukkan ke dalam *knapsack*,

  tetapi jika $p_k + f_{k-1}(y - w_k)$ lebih besar dari $f_{k-1}(y)$, maka objek yang ke-k dimasukkan.

- Relasi rekurens untuk persoalan ini adalah

$$f_0(y) = 0, \quad y = 0, 1, 2, \ldots, M \qquad \text{(basis)}$$
$$f_k(y) = -\infty, \quad y < 0 \qquad \text{(basis)}$$

$$f_k(y) = \max\{f_{k-1}(y), p_k + f_{k-1}(y - w_k)\}, \quad \text{(rekurens)}$$
$$k = 1, 2, \ldots, n$$

- $f_k(y)$ adalah keuntungan optimum pada tahap *k* untuk kapasitas *knapsack* sebesar *y*.
- $f_0(y) = 0$ adalah nilai persoalan *knapsack* kosong (tidak ada persoalan *knapsack*) dengan kapasitas *y*)
- $f_k(y) = -\infty$ adalah nilai persoalan *knapsack* untuk kapasitas negatif.
- Solusi optimum dari persoalan *knapsack* adalah $f_n(M)$.

Contoh: $n = 3$

$M = 5$

Barang ke-i	w_i	p_i
1	2	65
2	3	80
3	1	30

Tahap 1:

$$f_1(y) = \max\{f_0(y),\ p_1 + f_0(y - w_1)\}$$
$$= \max\{f_0(y),\ 65 + f_0(y - 2)\}$$

y			Solusi Optimum	
	$f_0(y)$	$65 + f_0(y-2)$	$f_1(y)$	(x_1^*, x_2^*, x_3^*)
0	**0**	$-\infty$	0	(0, 0, 0)
1	**0**	$-\infty$	0	(0, 0, 0)
2	0	**65**	65	(1, 0, 0)
3	0	**65**	65	(1, 0, 0)
4	0	**65**	65	(1, 0, 0)
5	0	**65**	65	(1, 0, 0)

Contoh: $n = 3$

$M = 5$

Barang ke-i	w_i	p_i
1	2	65
2	3	80
3	1	30

Tahap 2:

$$f_2(y) = \max\{f_1(y), p_2 + f_1(y - w_2)\}$$
$$= \max\{f_1(y), 80 + f_1(y - 3)\}$$

y			Solusi Optimum	
	$f_1(y)$	$80 + f_1(y-3)$	$f_2(y)$	(x_1^*, x_2^*, x_3^*)
0	**0**	$80 + (-\infty) = -\infty$	0	(0, 0, 0)
1	**0**	$80 + (-\infty) = -\infty$	0	(0, 0, 0)
2	**65**	$80 + (-\infty) = -\infty$	65	(1, 0, 0)
3	65	80 + 0 = **80**	80	(0, 1, 0)
4	65	80 + 0 = **80**	80	(0, 1, 0)
5	65	80 + 65 = **145**	145	(1, 1, 0)

Contoh: $n = 3$

$M = 5$

Barang ke-i	w_i	p_i
1	2	65
2	3	80
3	1	30

Tahap 3:

$$f_3(y) = \max\{f_2(y), p_3 + f_2(y - w_3)\}$$
$$= \max\{f_2(y), 30 + f_2(y - 1)\}$$

y	$f_2(y)$	$30 + f_2(y-1)$	Solusi Optimum $f_3(y)$	Solusi Optimum (x_1^*, x_2^*, x_3^*)
0	**0**	$30 + (-\infty) = -\infty$	0	(0, 0, 0)
1	**0**	$30 + (-\infty) = -\infty$	0	(0, 0, 0)
2	**65**	30 + 0 = 30	65	(1, 0, 0)
3	80	30 + 65 = **95**	95	(1, 0, 1)
4	80	30 + 80 = **110**	110	(0, 1, 1)
5	**145**	30 + 80 = 110	145	(1, 1, 0)

Solusi optimum $X = (1, 1, 0)$ dengan $\sum p = f = 145$.

SELAMAT BELAJAR