

Program Dinamis (*Dynamic Programming*)

Bagian 1 (Update 2026)

Bahan Kuliah IF2211 Strategi Algoritma

Oleh: Rinaldi M



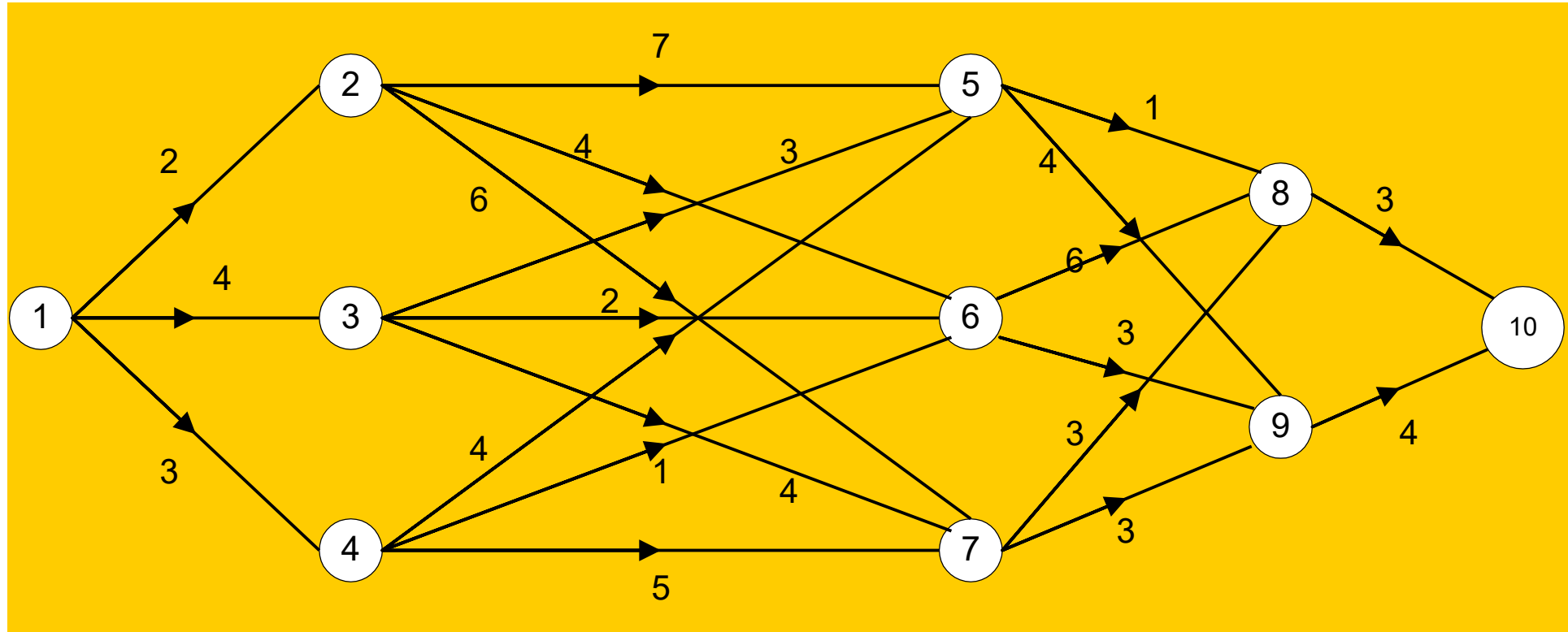
Program Studi Teknik Informatika
STEI-ITB

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. Program di sini artinya perencanaan.
- Istilah “dinamis” muncul karena pencarian solusinya melakukan perhitungan dengan menggunakan bantuan 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.

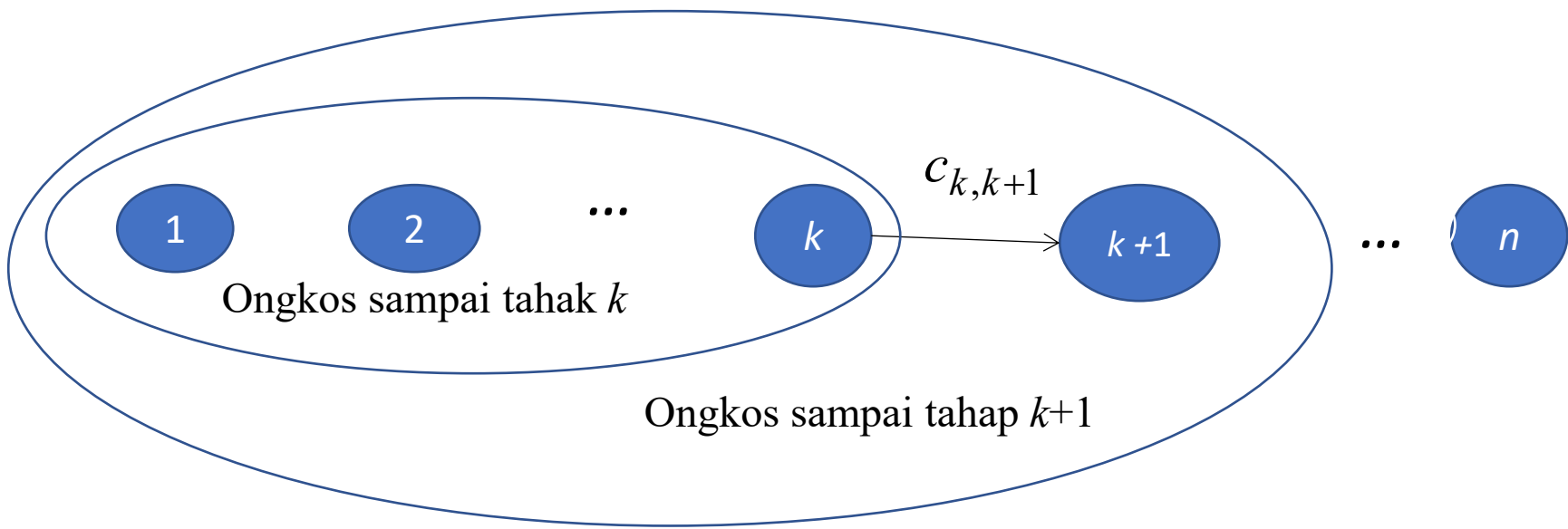


Greedy: dari setiap simpul, ambil sisi dengan bobot terkecil ke simpul berikutnya
Solusi *greedy*: 1 – 2 – 6 – 9 – 10 dengan $cost = 2 + 4 + 3 + 4 = 13 \rightarrow$ tidak optimal!

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 $c_{k,k+1}$)

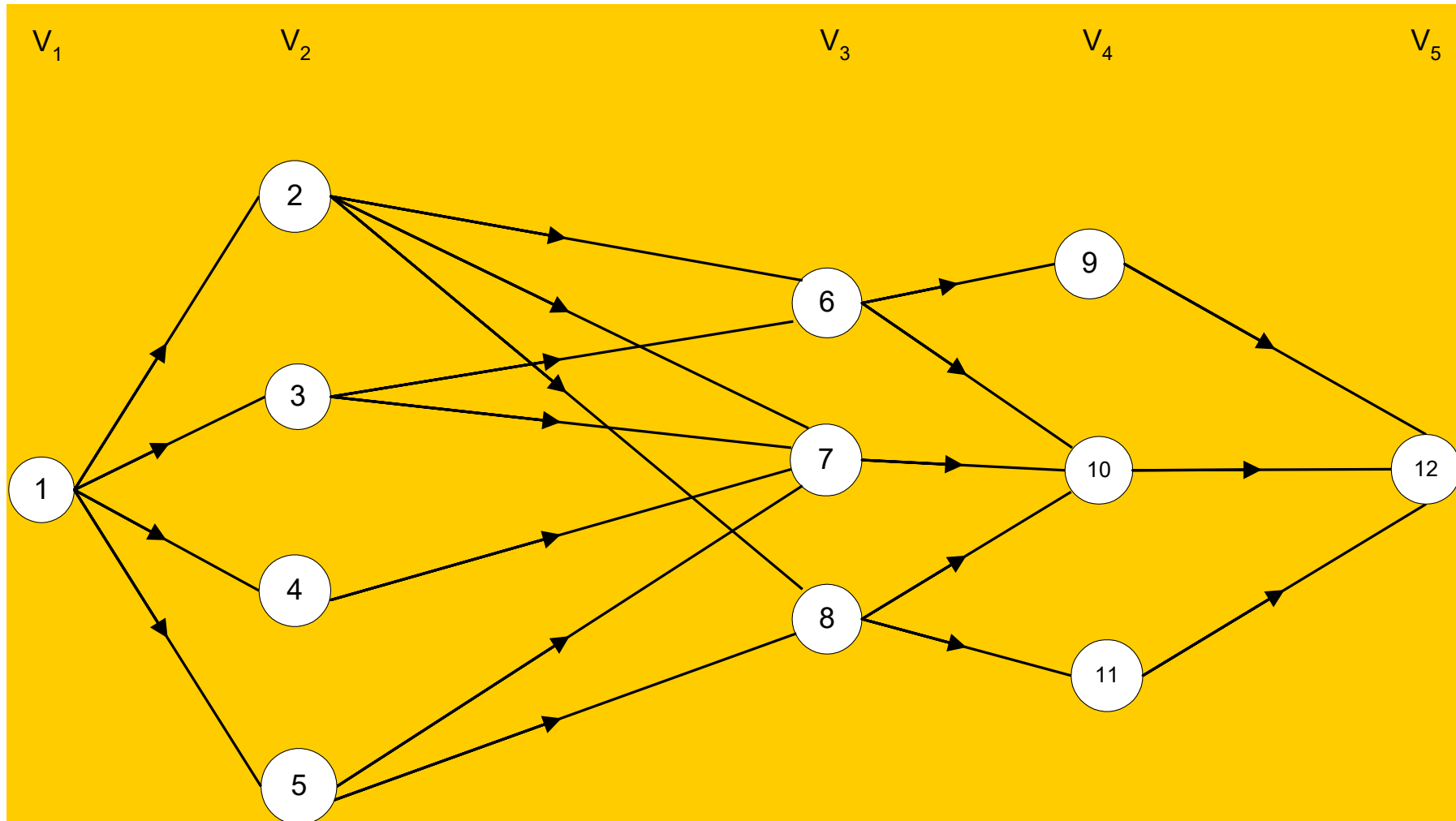


Karakteristik Persoalan dengan Program Dinamis

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.

Penyelesaian persoalan dengan Program Dinamis dapat dimodelkan dengan graf multistage (*multistage graph*)

Graf multistage (*multistage graph*). Tiap simpul di dalam graf tersebut menyatakan status, sedangkan V_1, V_2, \dots 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 ongkos dari tahap tersebut ke tahap berikutnya.
6. Solusi optimal dinyatakan dalam relasi rekursens yang menghubungkan 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.

Memoisasi (*memoization*)

- Ide kunci di dalam program dinamis adalah memoisasi, yang berarti bahwa kita menyimpan setiap nilai fungsi dalam sebuah tabel langsung setelah menghitungnya.
- Kemudian, ketika nilai tersebut dibutuhkan lagi, nilai tersebut dapat diambil dari tabel tanpa panggilan rekursif.
- Karena itulah, penggunaan tabel membantu untuk mengingat nilai perhitungan fungsi pada tahap sebelumnya

Dua pendekatan PD

Dua pendekatan yang digunakan dalam program dinamis:

1. Program dinamis maju (*forward* atau *up-down*)
→ Perhitungan dilakukan dari tahap $1, 2, \dots, n - 1, n$
2. Program dinamis mundur (*backward* atau *bottom-up*)
→ Perhitungan dilakukan dari tahap $n, n - 1, \dots, 2, 1$.

Misalkan x_1, x_2, \dots, 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, \dots, 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.

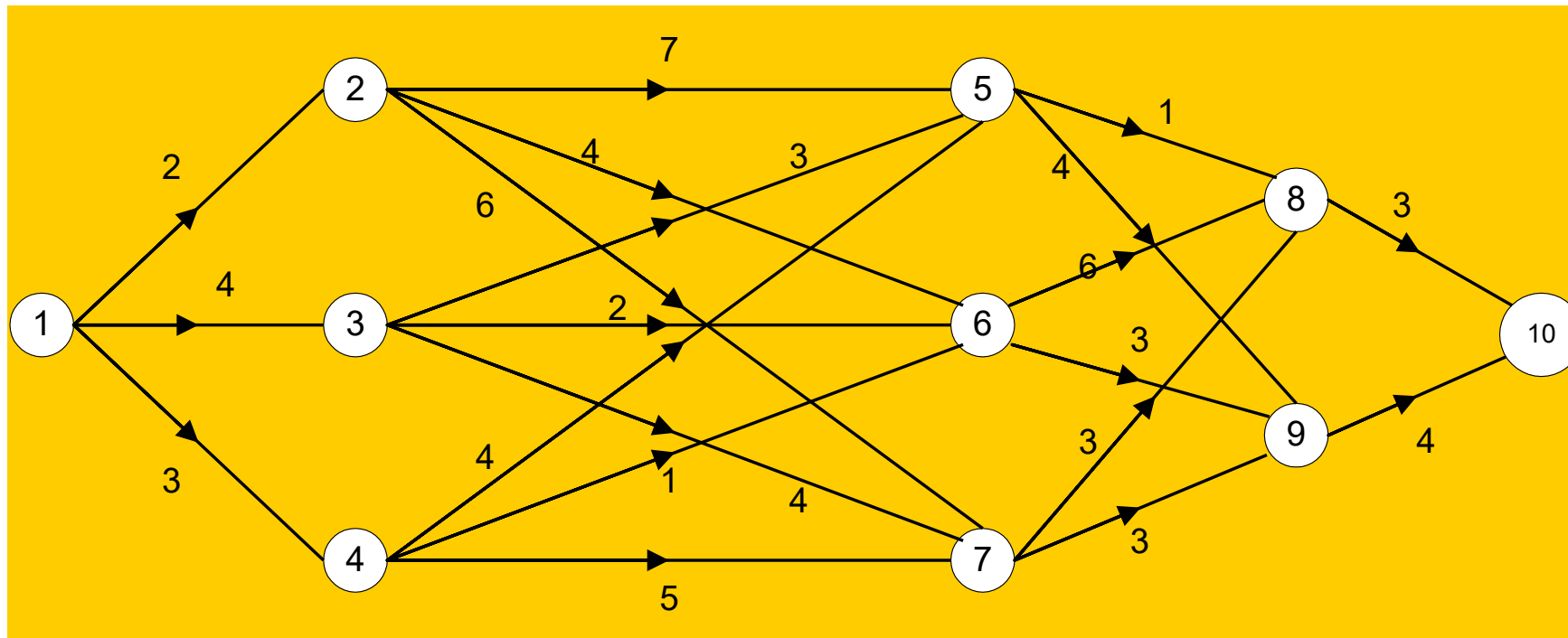
Rangkaian peubah keputusan adalah x_n, x_{n-1}, \dots, x_1 .

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:





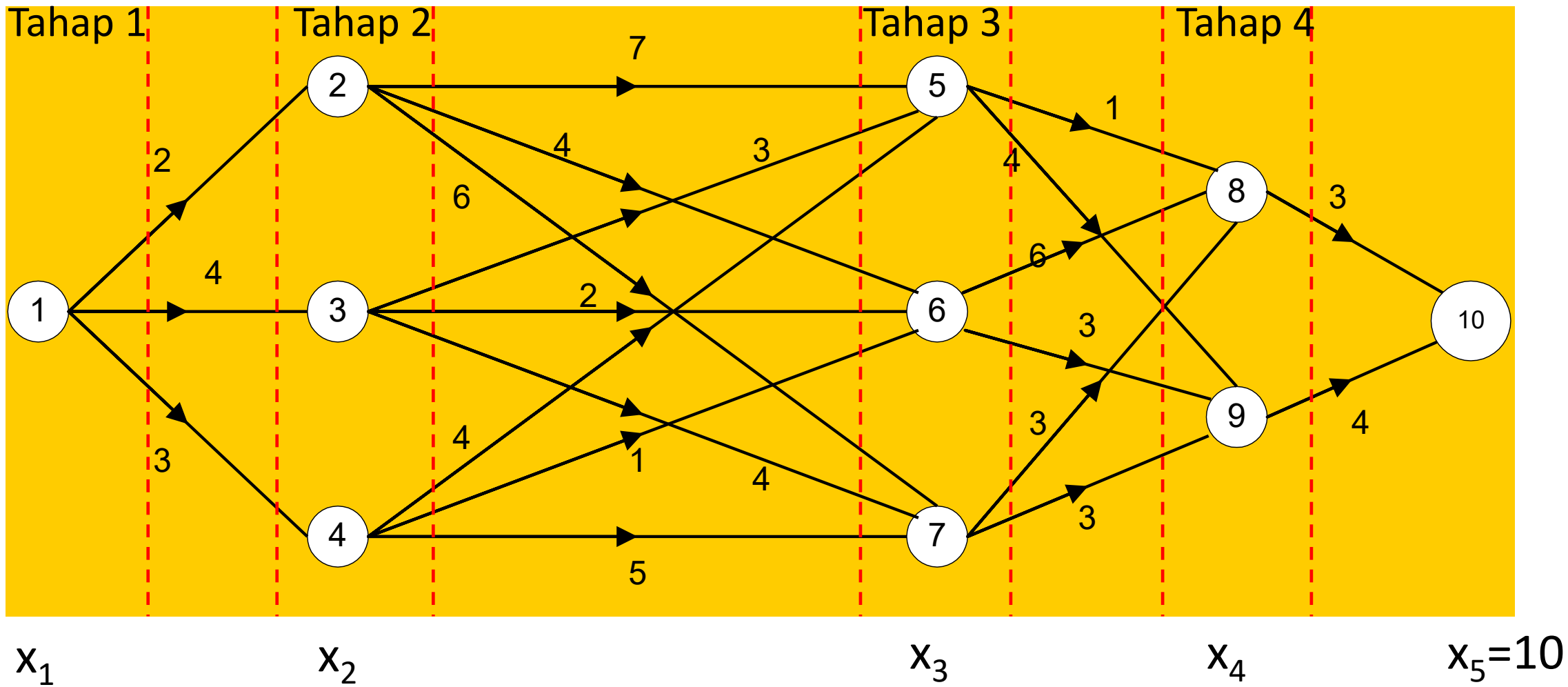
How to find the shortest path?

(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 perlu 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.



(b) Definisikan hubungan rekursif solusi optimal

Relasi rekurens berikut menyatakan lintasan terpendek pada setiap tahap:

$$\begin{aligned} f_1(s) &= c_{x_1,s} && \text{(basis)} \\ f_k(s) &= \min \{ f_{k-1}(x_k) + c_{x_k,s} \} && \text{(rekurens)} \\ k &= 2, 3, 4 \end{aligned}$$

Keterangan:

- x_k : peubah keputusan pada tahap k ($k = 2, 3, 4$).
- s : status pada setiap tahap
- $c_{x_k,s}$: bobot (*cost*) sisi dari ke x_k ke s
- $f_k(s)$: nilai minimum dari $f_k(x_k, s)$
- $f_{k-1}(x_k)$: nilai minimum tahap sebelumnya dari x_k ke s

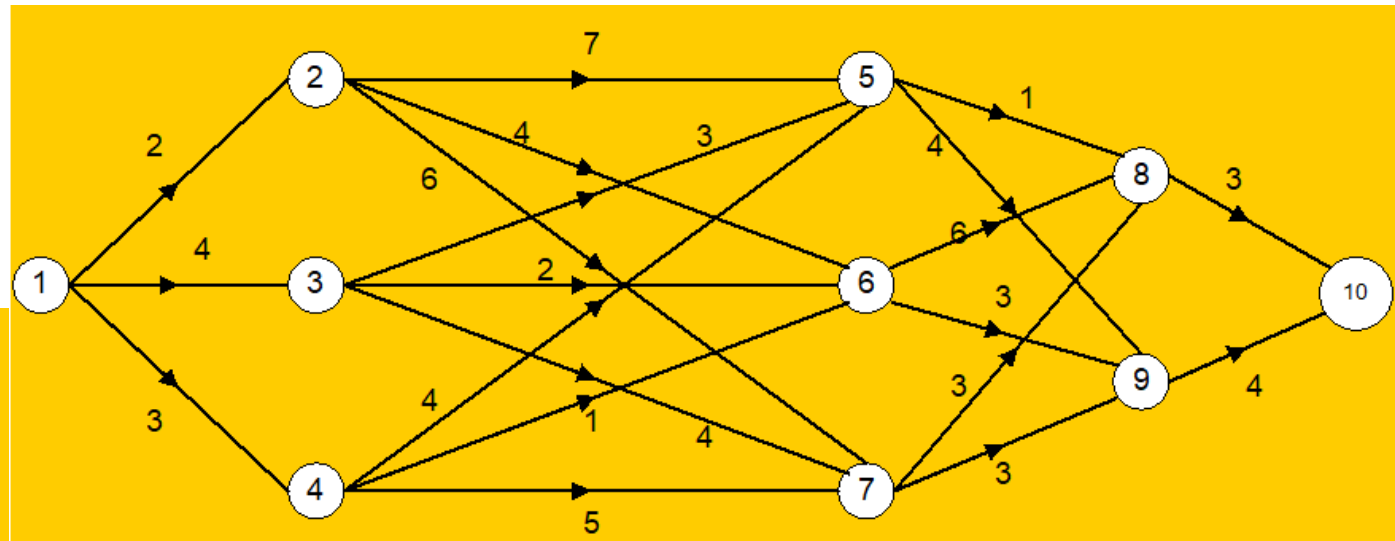
(c) Hitung nilai solusi optimal

Tahap 1:

$$f_1(s) = c_{x_1s}$$

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

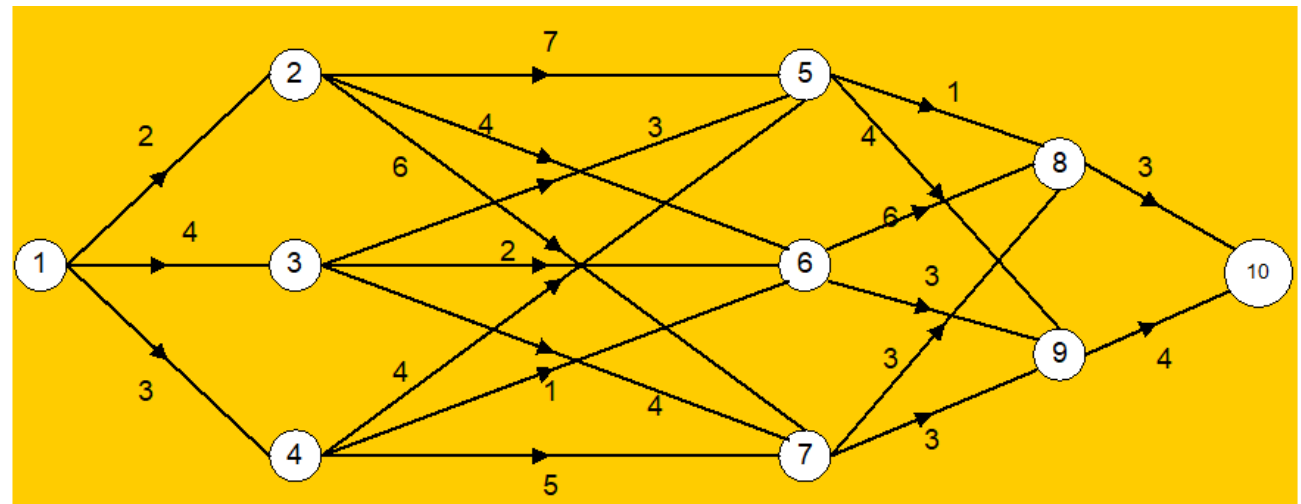
Catatan: x_k^* adalah nilai x_k yang meminimumkan f_s .



$$f_1(s) = c_{x_1,s}$$

Tahap sebelumnya (Tahap 1)

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



Tahap 2:

$$f_2(s) = \min_{x_2} \{f_1(x_2) + c_{x_2s}\}$$

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

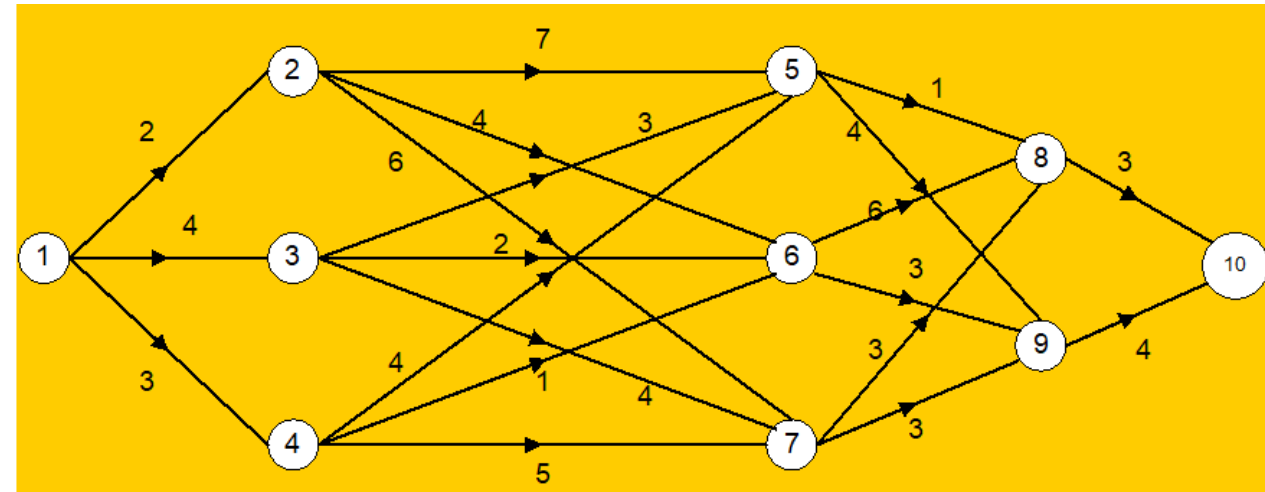
$$f_k(s) = \min \{f_{k-1}(x_k) + c_{x_k,s}\}$$

Tahap sebelumnya (Tahap 2)

Tahap 2:

$$f_2(s) = \min_{x_2} \{f_1(x_2) + c_{x_2s}\}$$

$s \backslash x_2$	$f_2(s) = f_1(x_2) + c_{x_2,s}$			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:

$$f_3(s) = \min_{x_3} \{f_2(x_3) + c_{x_3s}\}$$

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

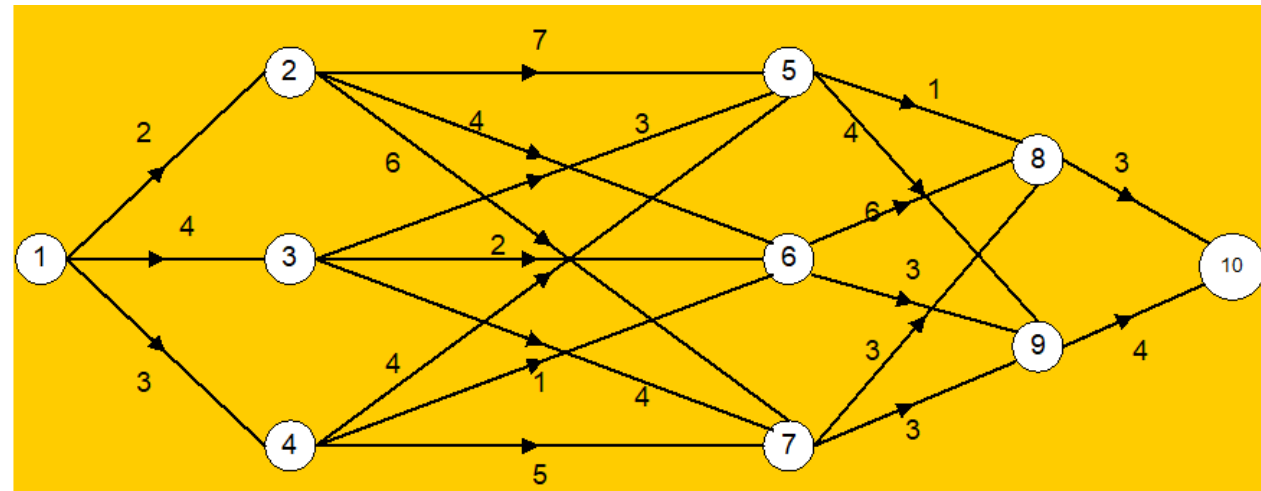
$$f_k(s) = \min \{f_{k-1}(x_k) + c_{x_k,s}\}$$

Tahap sebelumnya (Tahap 3)

Tahap 3:

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

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



Tahap 4:

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

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

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

(d) Rekonstruksi solusi optimal

Solusi optimum dapat dibaca pada tabel di bawah ini:

	x_4	x_3	x_2	x_1	Panjang Lintasan Terpendek
	8	5	3	1	11
10			4		
	9	6	4	1	11

Jadi ada tiga lintasan terpendek dari 1 ke 10, yaitu

$$1 \rightarrow 3 \rightarrow 5 \rightarrow 8 \rightarrow 10$$

$$1 \rightarrow 4 \rightarrow 5 \rightarrow 8 \rightarrow 10$$

$$1 \rightarrow 4 \rightarrow 6 \rightarrow 9 \rightarrow 10$$

yang mana panjang ketiga lintasan tersebut sama, yaitu 11.

Tahap 2:

$$f_2(s) = \min_{x_2} \{f_1(x_2) + c_{x_2s}\}$$

$x_2 \backslash s$	$f_2(s) = f_1(x_2) + c_{x_2s}$			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:

$$f_3(s) = \min_{x_3} \{f_2(x_3) + c_{x_3s}\}$$

$x_3 \backslash s$	$f_3(s) = f_2(x_3) + c_{x_3s}$			Solusi Optimum	
	5	6	7	$f_3(s)$	x_3^*
8	8	10	11	8	5
9	11	7	11	7	6

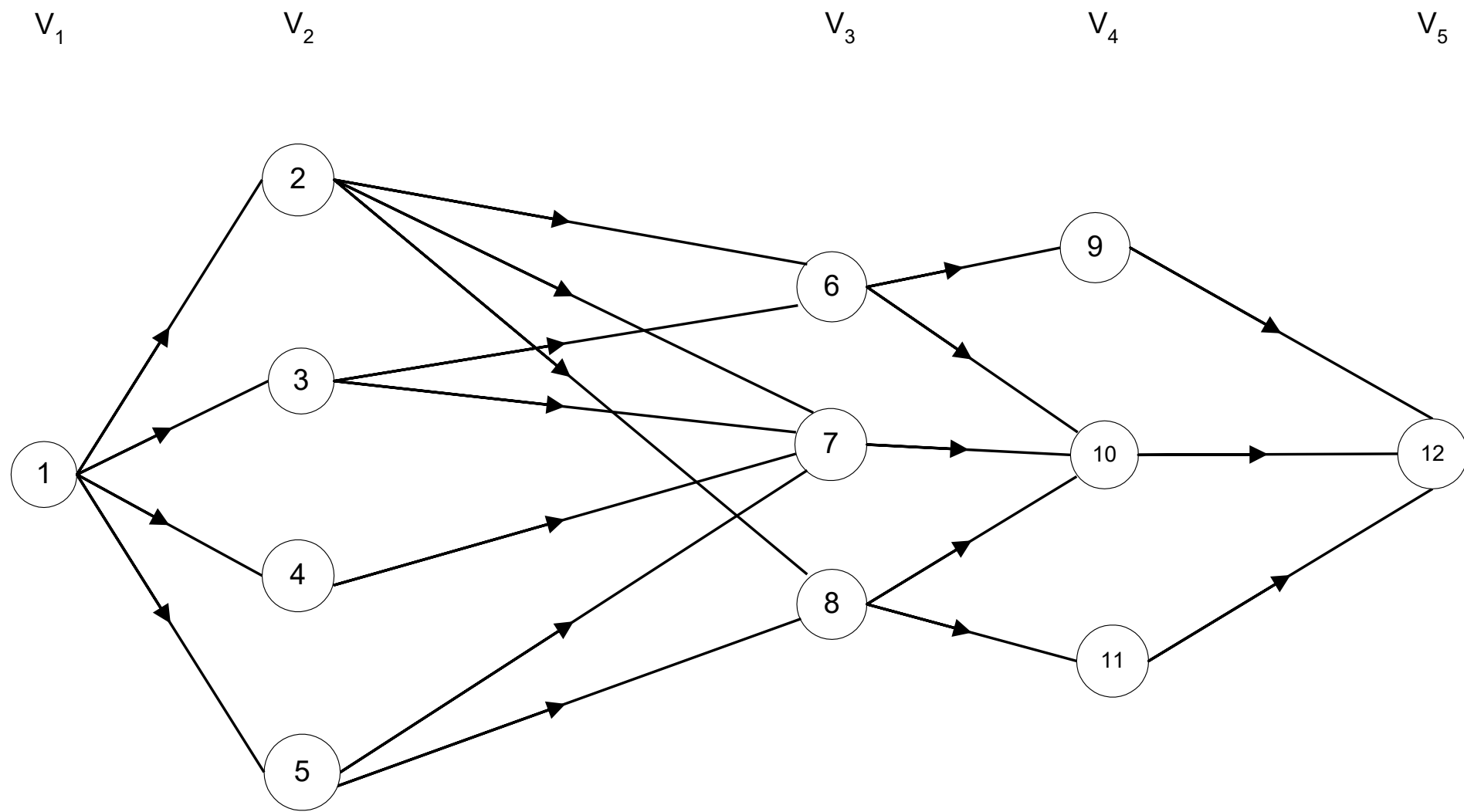
Tahap 4:

$$f_4(s) = \min_{x_4} \{f_3(x_4) + c_{x_4s}\}$$

$x_4 \backslash s$	$f_4(s) = f_3(x_4) + c_{x_4s}$		Solusi Optimum	
	8	9	$f_4(s)$	x_4^*
10	11	11	11	8 atau 9

Penyelesaian dengan Program Dinamis Mundur

- Persoalan ini akan diselesaikan dengan program dinamis mundur.
- Misalkan x_1, x_2, \dots, x_4 adalah simpul-simpul yang dikunjungi pada tahap k ($k = 1, 2, 3, 4$).
- Maka rute yang dilalui adalah $1 \rightarrow x_1 \rightarrow x_2 \rightarrow x_3 \rightarrow x_4$, yang dalam hal ini $x_4 = 10$.
- 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.



- Relasi rekurens berikut menyatakan lintasan terpendek (nilai keputusan terbaik) dari status s ke x_4 pada tahap k :

$$f_4(s) = c_{sx_4} \quad (\text{basis})$$

$$f_k(s) = \min_{x_k} \{c_{sx_k} + f_{k+1}(x_k)\}, \quad k = 1, 2, 3 \quad (\text{rekurens}) \quad |$$

yang dalam hal ini,

- peubah x_k menyatakan peubah keputusan pada tahap k ($k = 1, 2, 3$)
- c_{sx_k} menyatakan bobot (*cost*) sisi dari s ke x_k
- $f_k(s, x_k)$ menyatakan total bobot lintasan dari s ke x_k
- $f_k(s)$ adalah nilai minimum dari $f_k(s, x_k)$

Tujuan program dinamis mundur adalah mendapatkan $f_1(1)$ dengan cara mencari $f_4(s)$, $f_3(s)$, $f_2(s)$ terlebih dahulu.

Tahap 4:

$$f_4(s) = c_{x_4}$$

s	Solusi Optimum	
	$f_4(s)$	x_4^*
8	3	10
9	4	10

Catatan: x_k^* adalah nilai x_k yang meminimumkan $f_k(s, x_k)$.

Tahap 3:

$$f_3(s) = \min_{x_3} \{c_{x_3} + f_4(x_3)\}$$

s \ x ₃	$f_3(s, x_3) = c_{x_3} + f_4(x_3)$		Solusi Optimum	
	8	9	$f_3(s)$	x_3^*
5	4	8	4	8
6	9	7	7	9
7	6	7	6	8

Tahap 2:

$$f_2(s) = \min_{x_2} \{c_{sx_2} + f_3(x_2)\}$$

$s \backslash x_2$	$f_2(s, x_2) = c_{sx_2} + f_3(x_2)$			Solusi Optimum	
	5	6	7	$f_2(s)$	x_2^*
2	11	11	12	11	5 atau 6
3	7	9	10	7	5
4	8	8	11	8	5 atau 6

Tahap 1:

$$f_1(s) = \min_{x_1} \{c_{sx_1} + f_2(x_1)\}$$

$s \backslash x_1$	$f_1(s, x_1) = c_{sx_1} + f_2(x_1)$			Solusi Optimum	
	2	3	4	$f_1(s)$	x_1^*
1	13	11	11	11	3 atau 4

Solusi optimum dapat dibaca pada tabel di bawah ini:

	x_1	x_2	x_3	x_4	Panjang Lintasan Terpendek
1	3	5	8	10	11
		5	8	10	11
4		6	9	10	11

Jadi ada tiga lintasan terpendek dari 1 ke 10, yaitu

$$1 \rightarrow 3 \rightarrow 5 \rightarrow 8 \rightarrow 10$$

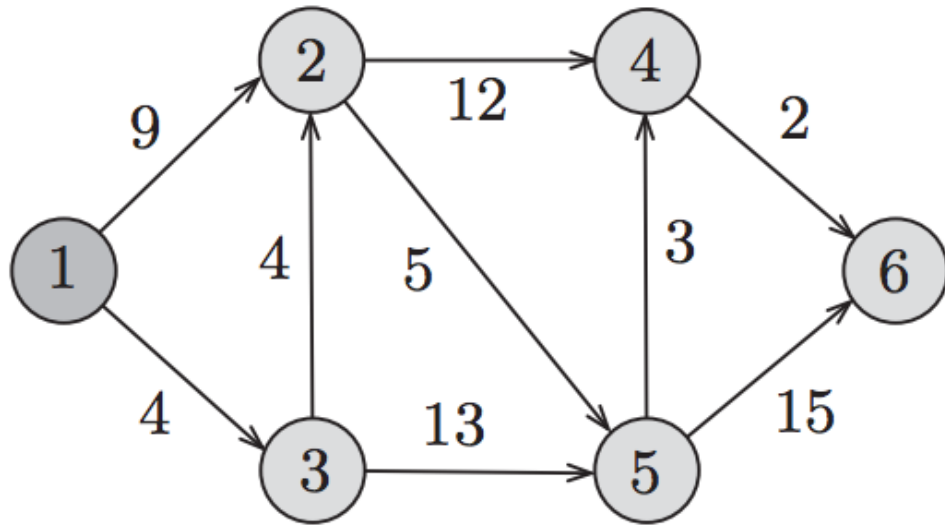
$$1 \rightarrow 4 \rightarrow 5 \rightarrow 8 \rightarrow 10$$

$$1 \rightarrow 4 \rightarrow 6 \rightarrow 9 \rightarrow 10$$

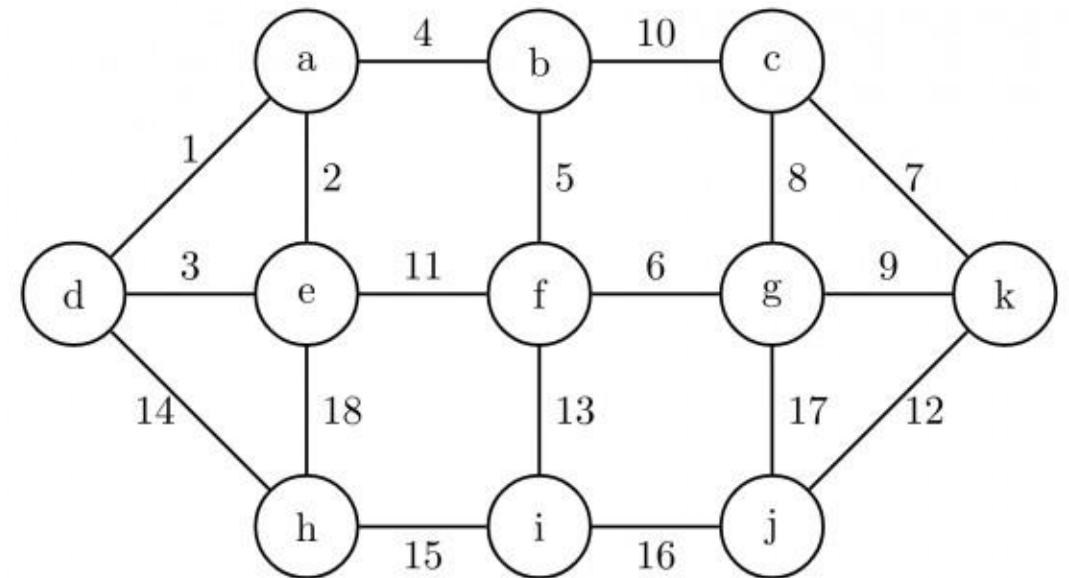
yang mana panjang ketiga lintasan tersebut sama, yaitu 11.

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*

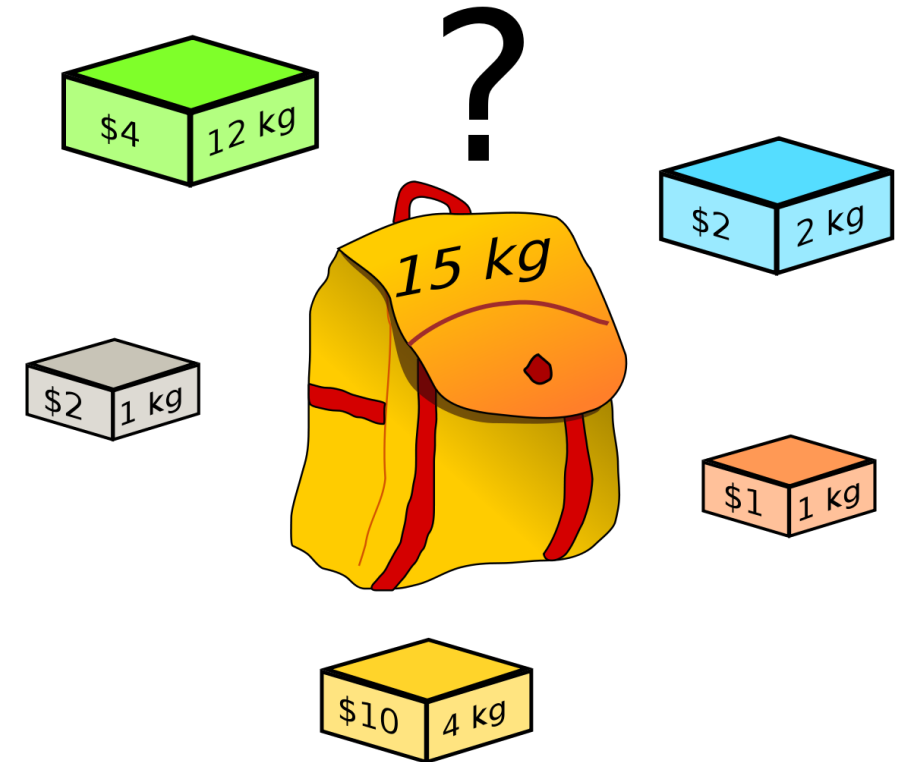
Diberikan sebuah *knapsack* dengan kapasitas M . Terdapat n buah objek, setiap objek memiliki bobot w_i dan keuntungan p_i . Bagaimana cara memilih objek-objek yang dimasukkan ke dalam *knapsack* sehingga total keuntungan yang diperoleh maksimal.

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

dengan kendala (*constraint*)

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

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



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 tahap 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

$$f_0(y) = 0, \quad y = 0, 1, 2, \dots, M \quad (\text{basis})$$

$$f_k(y) = -\infty, \quad y < 0 \quad (\text{basis})$$

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

$$k = 1, 2, \dots, n$$

- $f_k(y)$ 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	$80 + 0 = \mathbf{80}$	80	(0, 1, 0)
4	65	$80 + 0 = \mathbf{80}$	80	(0, 1, 0)
5	65	$80 + 65 = \mathbf{145}$	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	Solusi Optimum			
	$f_2(y)$	$30 + f_2(y - 1)$	$f_3(y)$	(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	$30 + 65 = \mathbf{95}$	95	(1, 0, 1)
4	80	$30 + 80 = \mathbf{110}$	110	(0, 1, 1)
5	145	$30 + 80 = 110$	145	(1, 1, 0)

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

Latihan

(Coin exchange problem) Tinjau kembali persoalan penukaran uang yang telah dibahas di dalam algoritma greedy. Misalkan terdapat sejumlah koin (dalam jumlah yang banyak) dengan nilai 1, 3, dan 4. Berapa jumlah minimal koin yang digunakan untuk menukar uang senilai x ? Sebagai contoh, jika $x = 10$, maka $10 = 3 + 3 + 4$, jadi diperlukan tiga koin untuk menukar uang senilai 10.

Penyelesaian:

Sifat penting dari fungsi f adalah bahwa nilainya dapat dihitung secara rekursif dari nilai-nilai yang lebih kecil. Idennya adalah untuk fokus pada koin pertama yang kita pilih untuk jumlahnya. Misalnya, dalam skenario di atas, koin pertama bisa berupa 1, 3, atau 4. Jika kita pertama kali memilih koin 1, tugas yang tersisa adalah membentuk jumlah 9 menggunakan jumlah koin minimum, yang merupakan submasalah dari masalah aslinya. Hal yang sama berlaku untuk koin 3 dan 4.

Formula rekursif berikut untuk menghitung jumlah koin minimum:

$$f(x) = \min\{f(x - 1) + 1, f(x - 3) + 1, f(x - 4) + 1\}$$

Kasus basis adalah $f(0) = 0$ karena tidak ada penukaran koin untuk $x = 0$, dan $f(x) = \infty$ untuk $x < 0$

Jadi, jumlah minimal koin yang diperlukan untuk menukar uang senilai x adalah:

$$f(x) = \begin{cases} \infty & , x < 0 \\ 0 & , x = 0 \\ \min_{c \in \{koin\}} f(x - c) + 1 & , x > 0 \end{cases}$$

Contoh: Koin 1, 3, dan 4, $x = 10$, maka

$$f(x) = \begin{cases} \infty & , x < 0 \\ 0 & , x = 0 \\ \min \{f(x - 1) + 1, f(x - 3) + 1, f(x - 4) + 1\} & , x > 0 \end{cases}$$

Perhitungan dengan program dinamis maju adalah sebagai berikut:

$$f(x) = \begin{cases} \infty & , x < 0 \\ 0 & , x = 0 \\ \min \{f(x - 1) + 1, f(x - 3) + 1, f(x - 4) + 1\} & , x > 0 \end{cases}$$

- $f(0) = 0$
- $f(1) = \min\{f(0)+1, f(-2)+1, f(-3)+1\} = \min\{0+1, \infty, \infty\} = \min\{1, \infty, \infty\} = 1$ (koin yang dipilih: 1)
- $f(2) = \min\{f(1)+1, f(-1)+1, f(-2)+1\} = \min\{1+1, \infty, \infty\} = \min\{2, \infty, \infty\} = 2$ (koin yang dipilih: 1)
- $f(3) = \min\{f(2)+1, f(0)+1, f(-1)+1\} = \min\{2+1, 0+1, \infty\} = \min\{3, 1, \infty\} = 1$ (koin yang dipilih: 3)
- $f(4) = \min\{f(3)+1, f(1)+1, f(0)+1\} = \min\{1+1, 1+1, 0+1\} = \min\{2, 2, 1\} = 1$ (koin yang dipilih: 4)
- $f(5) = \min\{f(4)+1, f(2)+1, f(1)+1\} = \min\{1+1, 2+1, 1+1\} = \min\{2, 3, 2\} = 2$ (koin yang dipilih: 4 atau 1)
- $f(6) = \min\{f(5)+1, f(3)+1, f(2)+1\} = \min\{2+1, 1+1, 2+1\} = \min\{3, 2, 3\} = 2$ (koin yang dipilih : 3)
- $f(7) = \min\{f(6)+1, f(4)+1, f(3)+1\} = \min\{2+1, 1+1, 1+1\} = \min\{3, 2, 2\} = 2$ (koin yang dipilih : 3 atau 4)
- $f(8) = \min\{f(7)+1, f(5)+1, f(4)+1\} = \min\{2+1, 2+1, 1+1\} = \min\{3, 3, 2\} = 2$ (koin yang dipilih : 4)
- $f(9) = \min\{f(8)+1, f(6)+1, f(5)+1\} = \min\{2+1, 2+1, 2+1\} = \min\{3, 3, 3\} = 3$ (koin yang dipilih : 1,3 atau 4)
- $f(10) = \min\{f(9)+1, f(7)+1, f(6)+1\} = \min\{3+1, 2+1, 2+1\} = \min\{4, 3, 3\} = 3$ (koin yang dipilih : 3 atau 4)

Rekonstruksi solusi dilakukan sebagai berikut:

$f(x)$ = jumlah minimum koin, $last[x]$ = koin terakhir yang dipilih

x	Perhitungan	f(x)	last[x]
0	basis	0	-
1	$\min\{f(0) + 1\}=1$	1	1
2	$\min\{f(1) + 1\}=2$	2	1
3	$\min\{f(2) + 1, f(0) + 1\}=1$	1	3
4	$\min\{f(3) + 1, f(1) + 1, f(0) + 1\}=1$	1	4
5	$\min\{f(4) + 1, f(2) + 1, f(1) + 1\}=2$	2	1 atau 4
6	$\min\{f(5) + 1, f(3) + 1, f(2) + 1\}=2$	2	3
7	$\min\{f(6) + 1, f(4) + 1, f(3) + 1\}=2$	2	3 atau 4
8	$\min\{f(7) + 1, f(5) + 1, f(4) + 1\}=2$	2	4
9	$\min\{f(8) + 1, f(6) + 1, f(5) + 1\}=3$	3	1 atau 3
10	$\min\{f(9) + 1, f(7) + 1, f(6) \downarrow\}=3$	3	4 atau 3

Dari $f(10)$, koin yang dipilih: 4

$$10 - 4 = 6$$

Dari $f(6)$, koin yang dipilih: 3

$$6 - 3 = 3$$

Dari $f(3)$, koin yang dipilih: 3

$$3 - 3 = 0$$

Kombinasi koin: 4, 3, 3

atau

Dari $f(10)$, koin yang dipilih: 3

$$10 - 3 = 7$$

Dari $f(7)$, koin yang dipilih: 3

$$7 - 3 = 4$$

Dari $f(4)$, koin yang dipilih: 4

$$3 - 3 = 0$$

Kombinasi koin: 3, 3, 4

Kombinasi lainnya: 3, 4, 3

Bersambung ke Bagian 2