Algoritma *Greedy* (lanjutan)

5. Penjadwalan *Job* dengan Tenggang Waktu (*Job Schedulling with Deadlines*)

Persoalan:

- Ada *n* buah *job* yang akan dikerjakan oleh sebuah mesin;
- tiap *job* diproses oleh mesin selama 1 satuan waktu dan tenggat waktu (*deadline*) setiap *job i* adalah $d_i \geq 0$;
- *job i* akan memberikan keuntungan sebesar p_i jika dan hanya jika *job* tersebut diselesaikan tidak melebihi tenggat waktunya;

- Bagaimana memilih *job-job* yang akan dikerjakan oleh mesin sehingga keuntungan yang diperoleh dari pengerjaan itu maksimum?

- Fungsi obyektif persoalan ini:

  Maksimasi $F = \sum_{i \in J} p_i$

- Solusi layak: himpunan *J* yang berisi urutan *job* yang sedemikian sehingga setiap *job* di dalam *J* selesai dikerjakan sebelum tenggat waktunya.

- Solusi optimum ialah solusi layak yang memaksimumkan *F*.

Contoh 7. Misalkan A berisi 4 *job* ($n = 4$):

$(p_1, p_2, p_3, p_4) = (50, 10, 15, 30)$

$(d_1, d_2, d_3, d_4) = (2, 1, 2, 1)$

Mesin mulai bekerja jam 6.00 pagi.

Job	Tenggat (d_i)	Harus selesai sebelum pukul
1	2 jam	8.00
2	1 jam	7.00
3	2 jam	8.00
4	1 jam	7.00

Pemecahan Masalah dengan *Exhaustive Search*

Cari himpunan bagian (*subset*) *job* yang layak dan memberikan total keuntungan terbesar.

Contoh: $(p_1, p_2, p_3, p_4) = (50, 10, 15, 30)$
$(d_1, d_2, d_3, d_4) = (2, 1, 2, 1)$

Barisan *job*	Keuntungan (*F*)	Keterangan
{}	0	layak
{1}	50	layak
{2}	10	layak
{3}	15	layak
{4}	30	layak
{1, 2}	-	tidak layak
{1, 3}	65	layak
{1, 4}	-	tidak layak
{2, 1}	60	layak
{2, 3}	25	layak
{2, 4}	-	tidak layak
{3, 1}	65	layak
{3, 2}	-	tidak layak
{3, 4}	-	tidak layak
{4, 1}	**0**	**layak (Optimum!)**
{4, 2}	-	tidak layak
{4, 3}	45	layak

Solusi optimum: $J = \{4, 1\}$ dengan $F = 80$.

Kompleksitas algoritma *exhaustive search* : $O(n \cdot 2^n)$.

Pemecahan Masalah dengan Algoritma *Greedy*

- Strategi *greedy* untuk memilih *job*:

  Pada setiap langkah, pilih *job i* dengan p_i yang terbesar untuk menaikkan nilai fungsi obyektif *F*.

Contoh: $(p_1, p_2, p_3, p_4) = (50, 10, 15, 30)$

$(d_1, d_2, d_3, d_4) = (2, 1, 2, 1)$

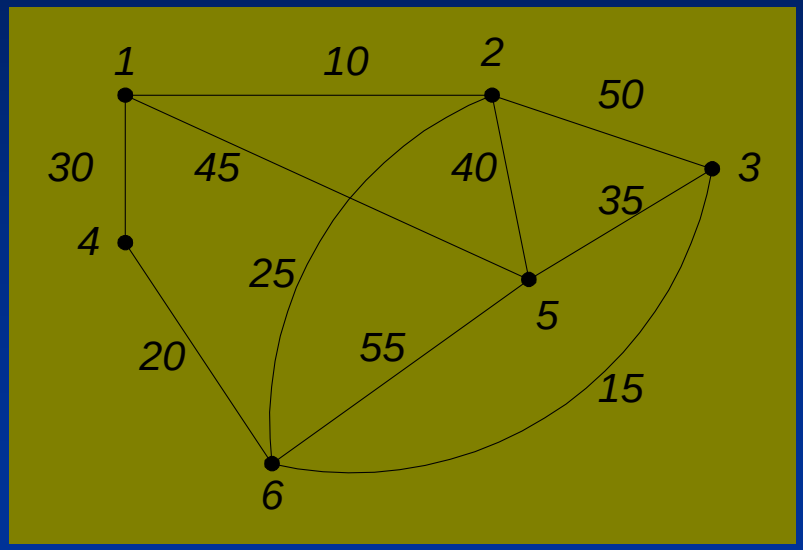
Langkah	J	$F = \sum p_i$	Keterangan
0	{}	0	-
1	{1}	50	layak
2	{4,1}	50 + 30 = 80	layak
3	{4, 1, 3}	-	tidak layak
4	{4, 1, 2}	-	tidak layak

Solusi optimal: J = {4, 1} dengan F = 80.

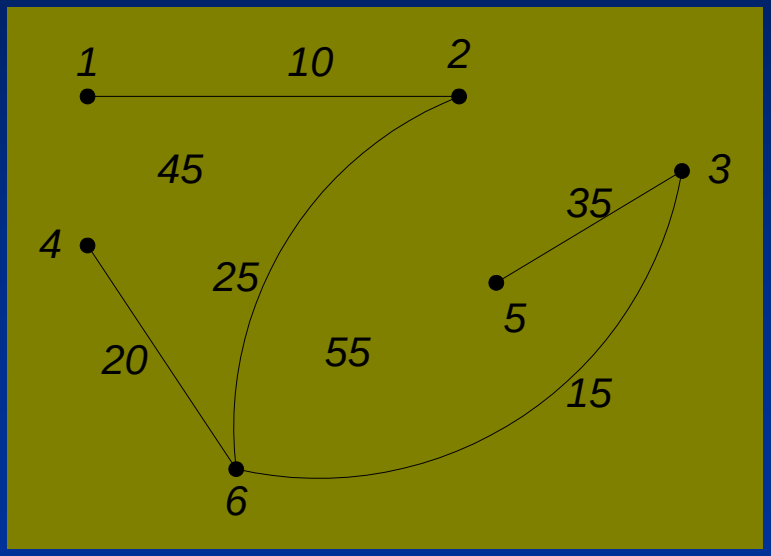
```
function JobSchedulling1(input C : himpunan_job) → himpunan_job
{ Menghasilkan barisan job yang akan diproses oleh mesin }

Deklarasi
  i : integer
  J : himpunan_job   { solusi }

Algoritma
  J ← {}
  while C ≠ {} do
     i ← job yang mempunyai p[i] terbesar
     C ← C - {i}
     if (semua job di dalam J ∪ {i} layak) then
        J ← J ∪ {i}
     endif
  endwhile
  { C = {} }
  return J
```

Kompleksitas algoritma *greedy* : $O(n^2)$.

6. Pohon Merentang Minimum



(a) Graf $G = (V, E)$



(b) Pohon merentang minimum

(a) Algoritma Prim

- Strategi *greedy* yang digunakan:

  Pada setiap langkah, pilih sisi *e* dari graf *G*(*V*, *E*) yang mempunyai bobot terkecil dan bersisian dengan simpul-simpul di *T* tetapi *e* tidak membentuk sirkuit di *T*.

- Komplesiats algoritma: $O(n^2)$

(a) Algoritma Kruskal

- Strategi *greedy* yang digunakan:

  Pada setiap langkah, pilih sisi *e* dari graf *G* yang mempunyai bobot minimum tetapi *e* tidak membentuk sirkuit di *T*.

Kompleksitas algoritma: $O(|E| \log |E|)$

7. Lintasan Terpendek (*Shortest Path)*

Beberapa macam persoalan lintasan terpendek:

- Lintasan terpendek antara dua buah simpul tertentu (*a pair shortest path*).
- Lintasan terpendek antara semua pasangan simpul (*all pairs shortest path*).
- Lintasan terpendek dari simpul tertentu ke semua simpul yang lain (*single-source shortest path*).
- Lintasan terpendek antara dua buah simpul yang melalui beberapa simpul tertentu (*intermediate shortest path*).

Persoalan:

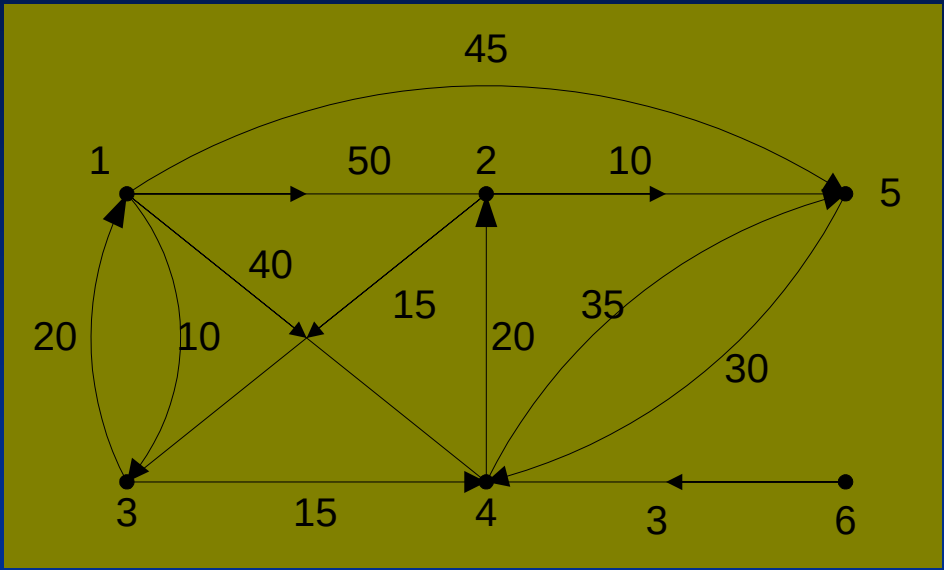
Diberikan graf berbobot $G = (V, E)$. Tentukan lintasan terpendek dari sebuah simpul asal *a* ke setiap simpul lainnya di *G*.

Asumsi yang kita buat adalah bahwa semua sisi berbobot positif.

Strategi *greedy*:

Lintasan dibentuk satu per satu. Lintasan berikutnya yang dibentuk ialah lintasan yang meminimumkan jumlah jaraknya.

Contoh 8.



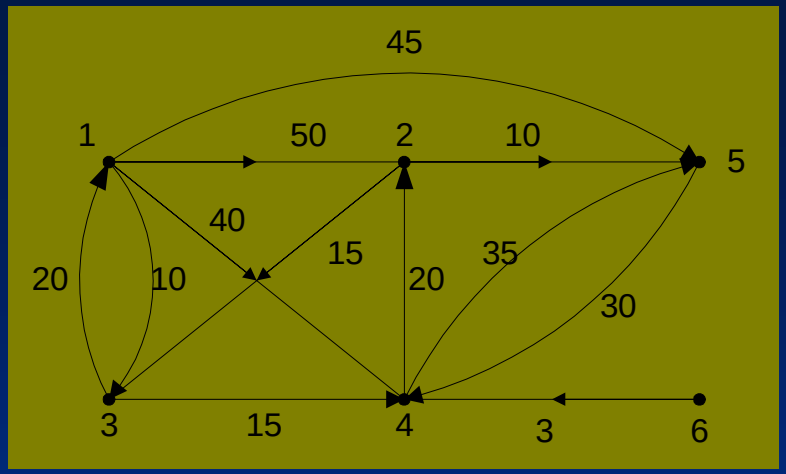
Simpul asal	Simpul tujuan	Lintasan terpendek	Jarak
1	3	$1 \rightarrow 3$	10
1	4	$1 \rightarrow 3 \rightarrow 4$	25
1	2	$1 \rightarrow 3 \rightarrow 4 \rightarrow 2$	45
1	5	$1 \rightarrow 5$	45
1	6	tidak ada	-

Algoritma Dijkstra

Strategi *greedy*:

Pada setiap langkah, ambil sisi yang berbobot minimum yang menghubungkan sebuah simpul yang sudah terpilih dengan sebuah simpul lain yang belum terpilih.

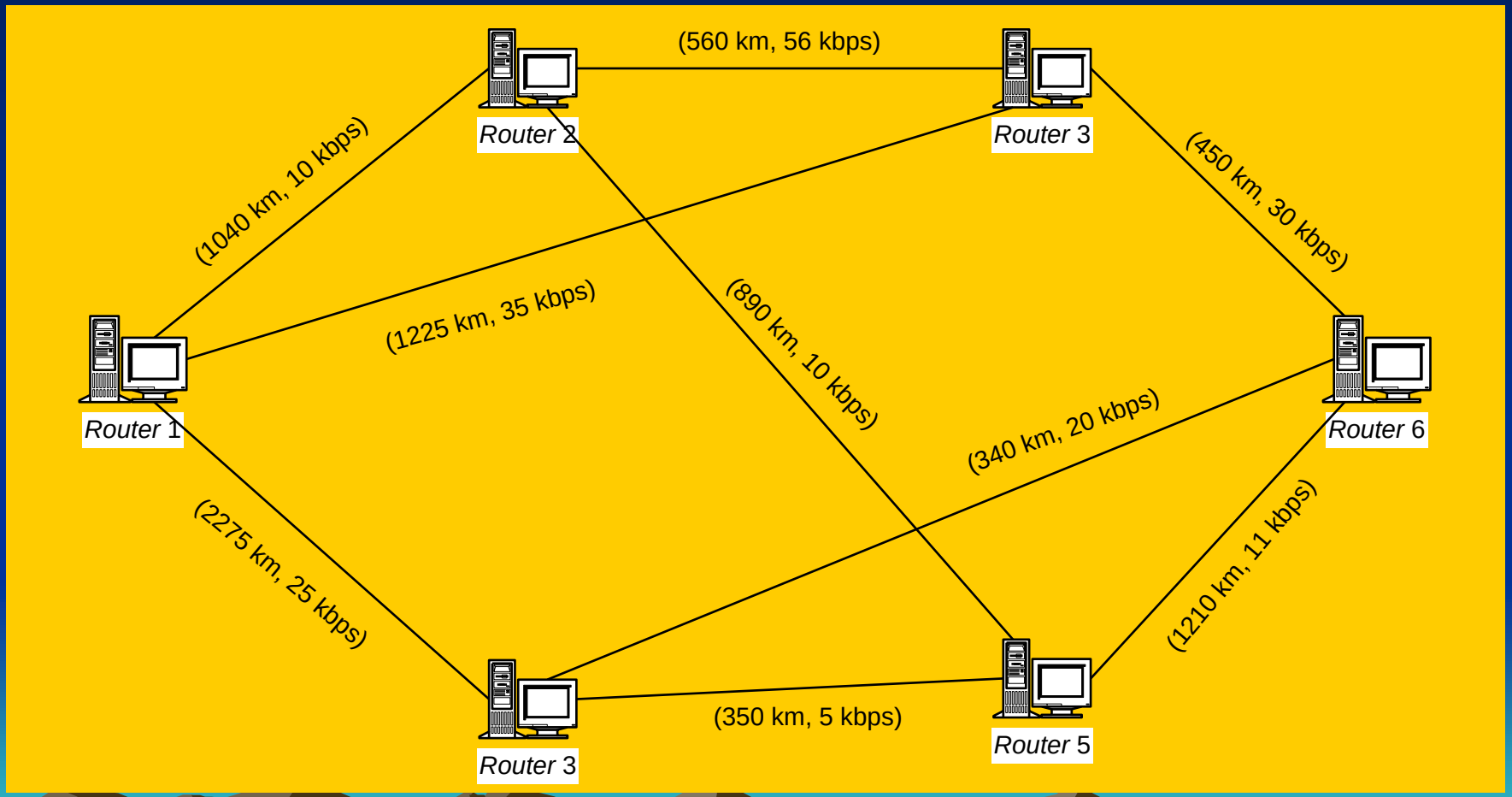
Lintasan dari simpul asal ke simpul yang baru haruslah merupakan lintasan yang terpendek diantara semua lintasannya ke simpul-simpul yang belum terpilih.



Lelaran	Simpul yang dipilih	Lintasan	*S*						*D*					
			1	2	3	4	5	6	1	2	3	4	5	6
Inisial	-	-	0	0	0	0	0	0	0	50 *(1,2)*	10 *(1,3)*	40 *(1,4)*	45 *(1,5)*	∞ *(1,6)*
1	1	1	1	0	0	0	0	0	∞	50 *(1,2)*	10 *(1,3)*	40 *(1,4)*	45 *(1,5)*	∞ *(1,6)*
2	3	1, 3	1	0	1	0	0	0	∞	50 *(1,2)*	10 *(1,3)*	25 *(1,3,4)*	45 *(1,5)*	∞ *(1,6)*
3	4	1, 3, 4	1	0	1	1	0	0	∞	45 *(1,3,4,2)*	10 *(1,3)*	25 *(1,3,4)*	45 *(1,5)*	∞ *(1,6)*
4	2	1, 3, 4, 2	1	1	1	1	0	0	∞	45 *(1,3,4,2)*	10 *(1,3)*	25 *(1,3,4)*	45 *(1,5)*	∞ *(1,6)*
5	5	1, 5	1	1	1	1	1	0	∞	45	10	25	45	∞

Aplikasi algoritma Dijkstra:

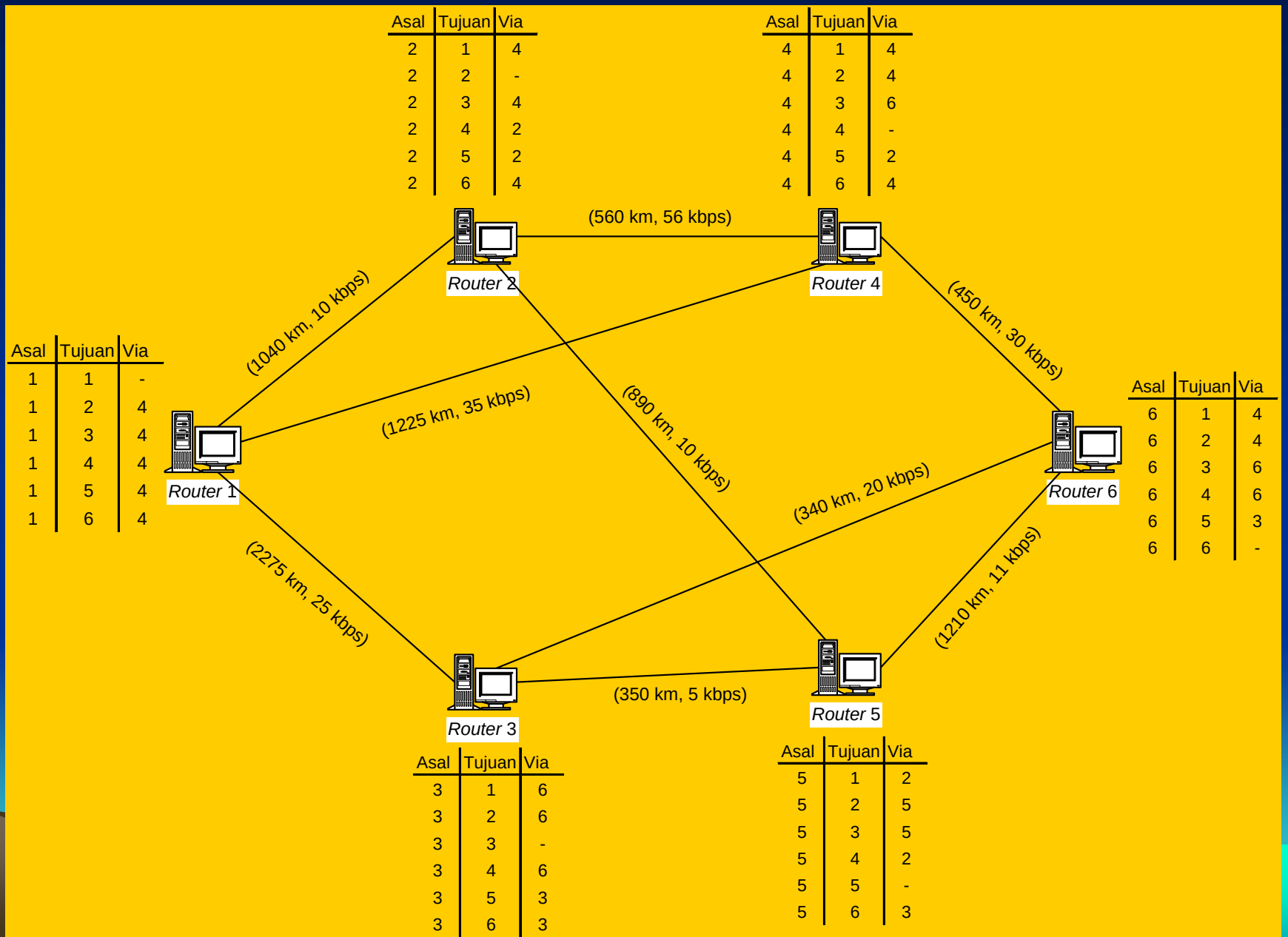
→*Routing* pada jaringan komputer



Lintasan terpendek (berdasarkan delai):

Router Asal	*Router* Tujuan	Lintasan Terpendek
1	1	-
	2	1, 4, 2
	3	1, 4, 6, 3
	4	1, 4
	5	1, 4, 2, 5
	6	1, 4, 6
2	1	2, 4, 1
	2	-
	3	2, 4, 6, 3
	4	2, 4
	5	2, 5
	6	2, 4, 6
3	1	3, 6, 4, 1
	2	3, 6, 4, 2
	3	-
	4	3, 6, 4
	5	3, 5
	6	3, 6
4	1	4, 1
	2	4, 2
	3	4, 6, 2
	4	4, 6, 3
	5	4, 2, 5
	6	4, 6

Router Asal	*Router* Tujuan	Lintasan Terpendek
5	1	5, 2, 4, 1
	2	5, 2
	3	5, 3
	4	5, 2, 4
	5	-
	6	5, 3, 6
6	1	6, 4, 1
	2	6, 4, 2
	3	6, 3
	4	6, 4
	5	6, 3, 5
	6	-


Asal Tujuan Via
2 1 4
2 2 -
2 3 4
2 4 2
2 5 2
2 6 4
Asal Tujuan Via
4 1 4
4 2 4
4 3 6
4 4 -
4 5 2
4 6 4
(560 km, 56 kbps)
Router 2
Router 4
(450 km, 30 kbps)
(1040 km, 10 kbps)
Asal Tujuan Via
1 1 -
1 2 4
1 3 4
1 4 4
1 5 4
1 6 4
(1225 km, 35 kbps)
(890 km, 10 kbps)
Asal Tujuan Via
6 1 4
6 2 4
6 3 6
6 4 6
6 5 3
6 6 -
Router 1
Router 6
(340 km, 20 kbps)
(2275 km, 25 kbps)
(1210 km, 11 kbps)
(350 km, 5 kbps)
Router 3
Router 5
Asal Tujuan Via
3 1 6
3 2 6
3 3 -
3 4 6
3 5 3
3 6 3
Asal Tujuan Via
5 1 2
5 2 5
5 3 5
5 4 2
5 5 -
5 6 3