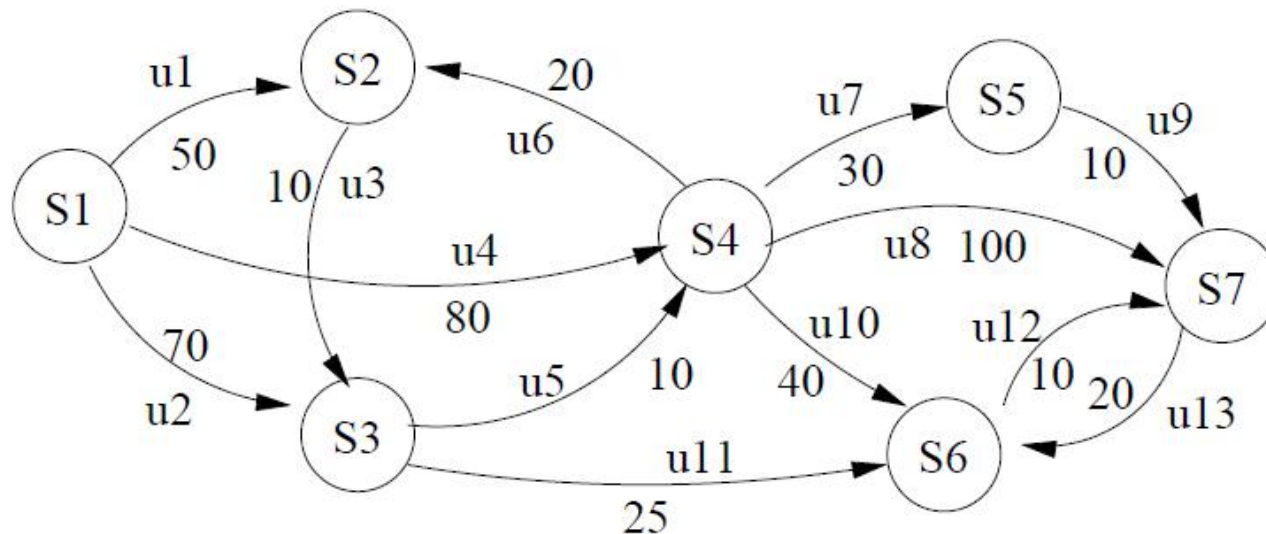


AFG TD 3 corrigé

# Q1

## 1 Dijkstra

**Q1:** A quels types de graphe s'applique l'algorithme de Dijkstra? Appliquez-le au graphe ci-dessous, à partir du sommet S1.



L'algo s'applique si il n'y a pas de poids d'arcs négatifs dans le graphe. Tableau d'évolution de  $D[]$  comme p 16 partie 2 du cours sur slide suivant

# Q1 Dijkstra

		D						
#	Choisi	D[1]	D[2]	D[3]	D[4]	D[5]	D[6]	D[7]
Init	-	0	$\infty$	$\infty$	$\infty$	$\infty$	$\infty$	$\infty$
1	$S_1$	0	50	70	80	$\infty$	$\infty$	$\infty$
2	$S_2$	0	50	60	80	$\infty$	$\infty$	$\infty$
3	$S_3$	0	50	60	70	$\infty$	85	$\infty$
4	$S_4$	0	50	60	70	100	85	170
5	$S_6$	0	50	60	70	100	85	95
6	$S_7$	0	50	60	70	100	85	95
7	$S_5$	0	50	60	70	100	85	95

MAJ

Tableau des prédecesseurs  
(voir p 15)

C						
C[1]	C[2]	C[3]	C[4]	C[5]	C[6]	C[7]
—	$S_1$	$S_2$	$S_3$	$S_4$	$S_3$	$S_6$

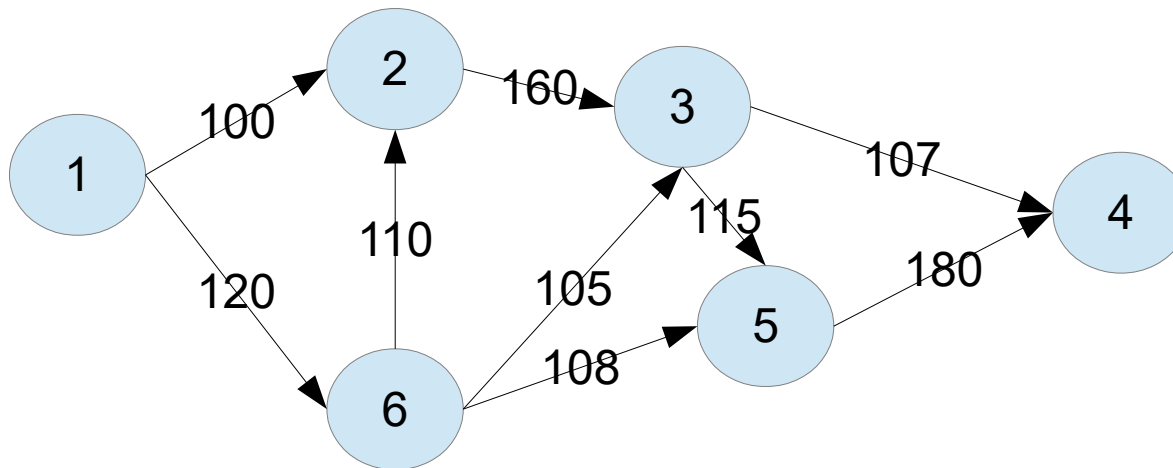
# Q2

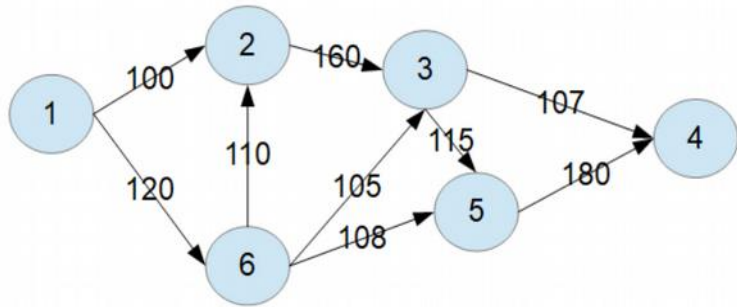
23

**Q2:** Quel est la solution de l'algorithme de capacité maximum sur le graphe de la question précédente ?

D – capacités maxi						
D[1]	D[2]	D[3]	D[4]	D[5]	D[6]	D[7]
$\infty$	50	70	80	30	40	80

NON FAIT en détail – capa max sur le graphe ci dessous à la place – réponse page suivante





# Q2

		D - capa max					
#	Choisi	D[1]	D[2]	D[3]	D[4]	D[5]	D[6]
Init	-	$\infty$	0	0	0	0	0
1	$S_1$	$\infty$	100	0	0	0	120
2	$S_6$	$\infty$	110	105	0	108	120
3	$S_2$	$\infty$	110	110	0	108	120
4	$S_3$	$\infty$	110	110	107	110	120
5	$S_5$	$\infty$	110	110	110	110	120
6	$S_4$	$\infty$	110	110	110	110	120

**MAJ**

C - capa max					
C[1]	C[2]	C[3]	C[4]	C[5]	C[6]
—	$S_6$	$S_2$	$S_5$	$S_3$	$S_1$

Tableau des prédecesseurs (voir p 15)

# Q3

## 2 Les autos

On cherche à déterminer une politique de remplacement d'une voiture (ou d'une moto) sur 5 ans. La voiture coûte 15.000 euros, elle est achetée à la date 0. Deux facteurs sont pris en compte, pour le coût global de l'auto : le coût d'entretien annuel, qui varie en fonction de l'âge de la voiture, et le prix de revente, également en fonction de l'âge (voir tableau).

**Q3:** Trouvez la règle qui donne le coût d'une voiture achetée en début d'année  $i$  et revendue en fin d'année  $j$ .

Le cout total comprend :

- Achat = 15.000
- Somme des entretiens année  $i$  (cout[1]) + année  $i+1$  (cout[2]) + ... + année  $j-1$  (cout[j - i])
- - valeur de reprise d'une voiture qui a  $(j-i)$  ans : reprise[j-i]

exemple : achat au 1/1/20 et revente au 1/1/22 ( $\rightarrow$  voiture gardée 2 ans)

cout total = 15.000 + 2.000 + 4.000 – 6.000 = 15.000

voiture gardée 1 an : 8.000

voiture gardée 3 ans : 24.000

voiture gardée 4 ans : 34.000

voiture gardée 5 ans : 47.000

# Q4

**Q4:** Modélisez le problème comme un problème de plus court chemin. Trouvez la solution en appliquant l'algorithme de Dijkstra.

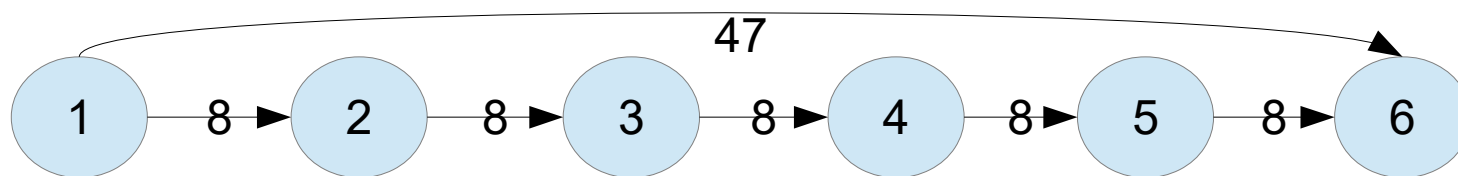
Si on change chaque année de voiture, pendant 5 ans :  $8 + 8 + 8 + 8 + 8 = 40$   
(5 x le cout global d'une voiture gardée un an)

Si on garde la même voiture 5 ans : 47 (voir calcul page precedente)  
→ il vaut mieux changer de voiture tous les ans

En terme de graphe :

- 1 sommet par date :
  - date 1 = début des 5 ans,
  - date 6 debut des 6 ans == fin des 5 ans
- 1 arc de i à j si achat à date i et revente à date j. poids de l'arc = cout pour durée entre i et j

→ pour les deux solutions évoquées ci dessus :

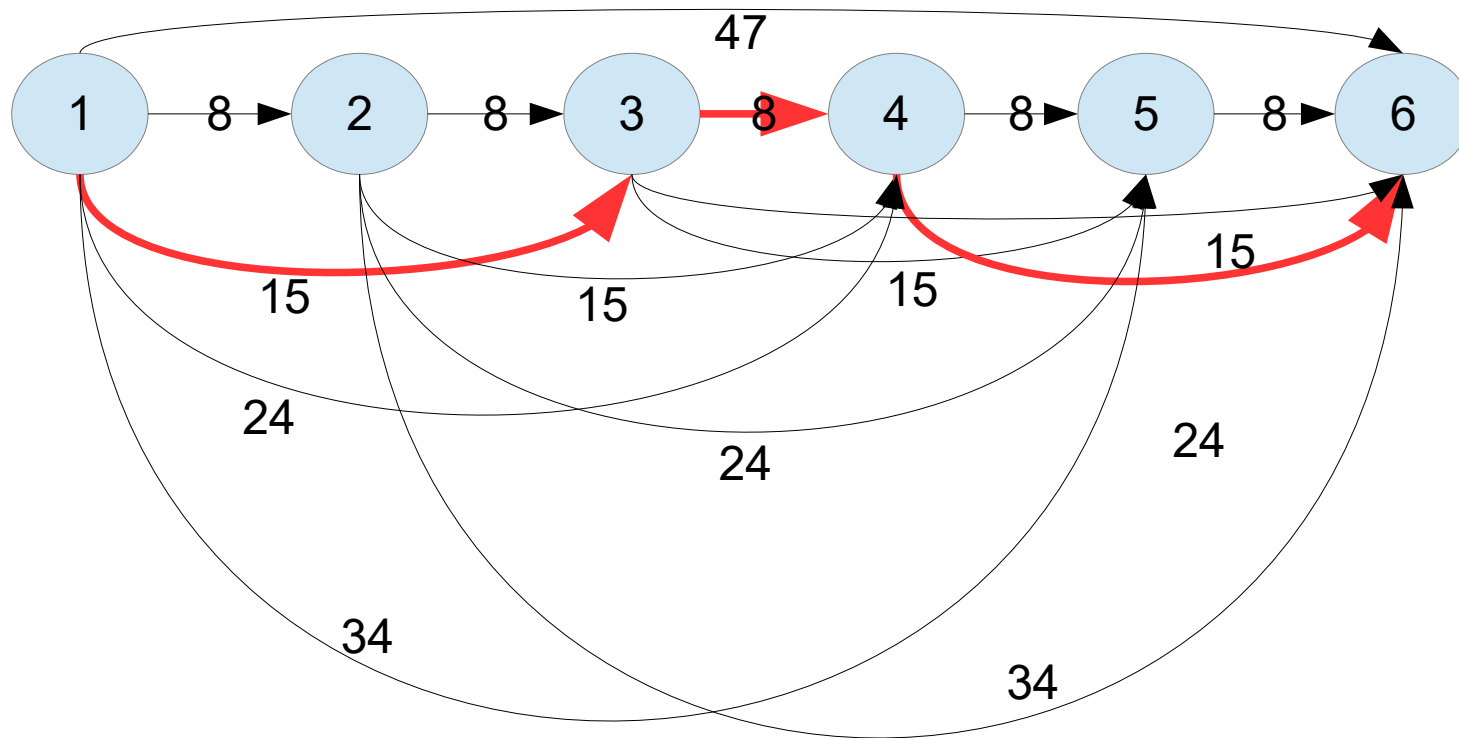


Et le plus court chemin du 1 au 6, trouvé par Dijkstra est bien celui qui passe par tous les sommets et vaut 40

# Q4

**Q4:** Modélisez le problème comme un problème de plus court chemin. Trouvez la solution en appliquant l'algorithme de Dijkstra.

Il faut compléter le graphe avec toutes les possibilités d'achat revente



Dijkstra à partir de 1.

$D[6] = 38$

Il existe 3 chemins de la longueur optimale, tous combinaisons de 2 ans, 2 ans et 1 an voiture gardée 1-2-2 ou 2-1-2 ou 2-2-1, avec comme cout sur 5 ans de  $15 + 15 + 8 = 38.000$  euros. Une des solutions optimale en rouge

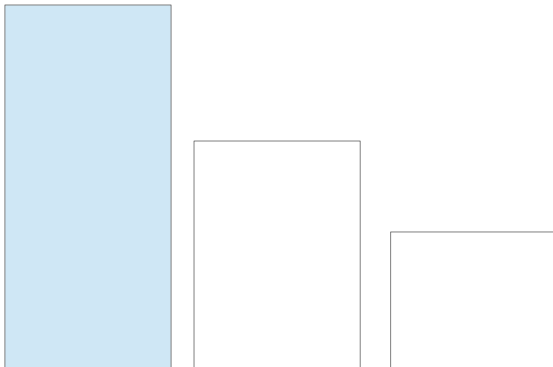


# Q5

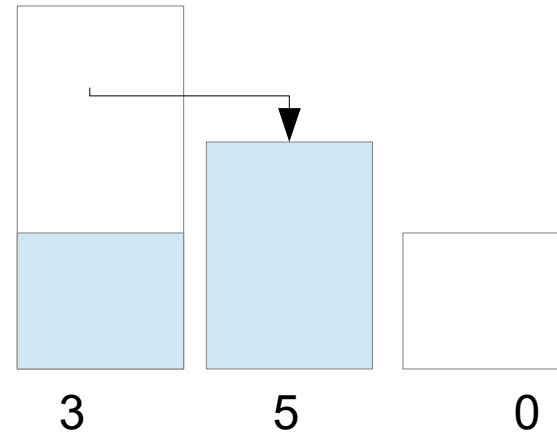
## 3 Les flacons

On cherche à résoudre le problème suivant. A partir de 3 flacons, de contenances respectives 8, 5 et 3 litres, on désire répartir le liquide de manière équitable entre les deux premiers flacons. Au départ le facon de 8 litres est rempli à ras bord, et les deux autres facons sont vides. Seules les opérations de transvasement complet sont autorisées : le contenu d'un flacon est versé dans un second flacon jusqu'à ce que celui-ci soit plein, ou que le flacon d'origine soit vide.

On a au départ :  
8L/ 8, 0L/5, 0L/3,  
Noté « 800 »



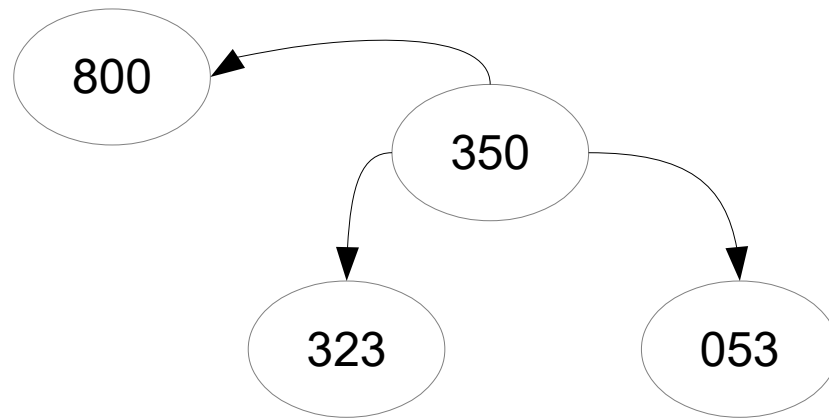
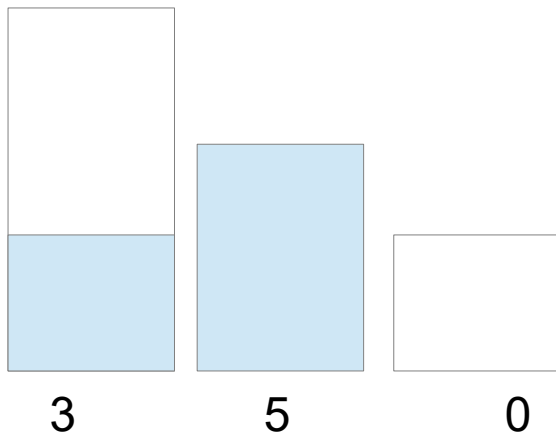
Si on verse du grand vers celui du milieu, on s'arrete quand le 5L est plein, et il en reste 3L dans le grand. → **état** 350



# Q5

A partir d'un état donné (avec combien il y a dans chaque flacon), on peut faire au plus 6 opérations (certaines n'étant pas possibles. Exemple pour état 350) :

- Grand vers milieu 350 → impossible, car milieu plein
- Grand vers petit 350 → 053
- Milieu vers grand 350 → 800
- Milieu vers petit 350 → 323
- Petit vers grand 350 → impossible car petit vide ET milieu plein
- Petit vers milieu 350 → impossible car petit vide



C'est le début d'un automate d'états fini (état du système de flacons, transitions possibles entre états)

# Q5

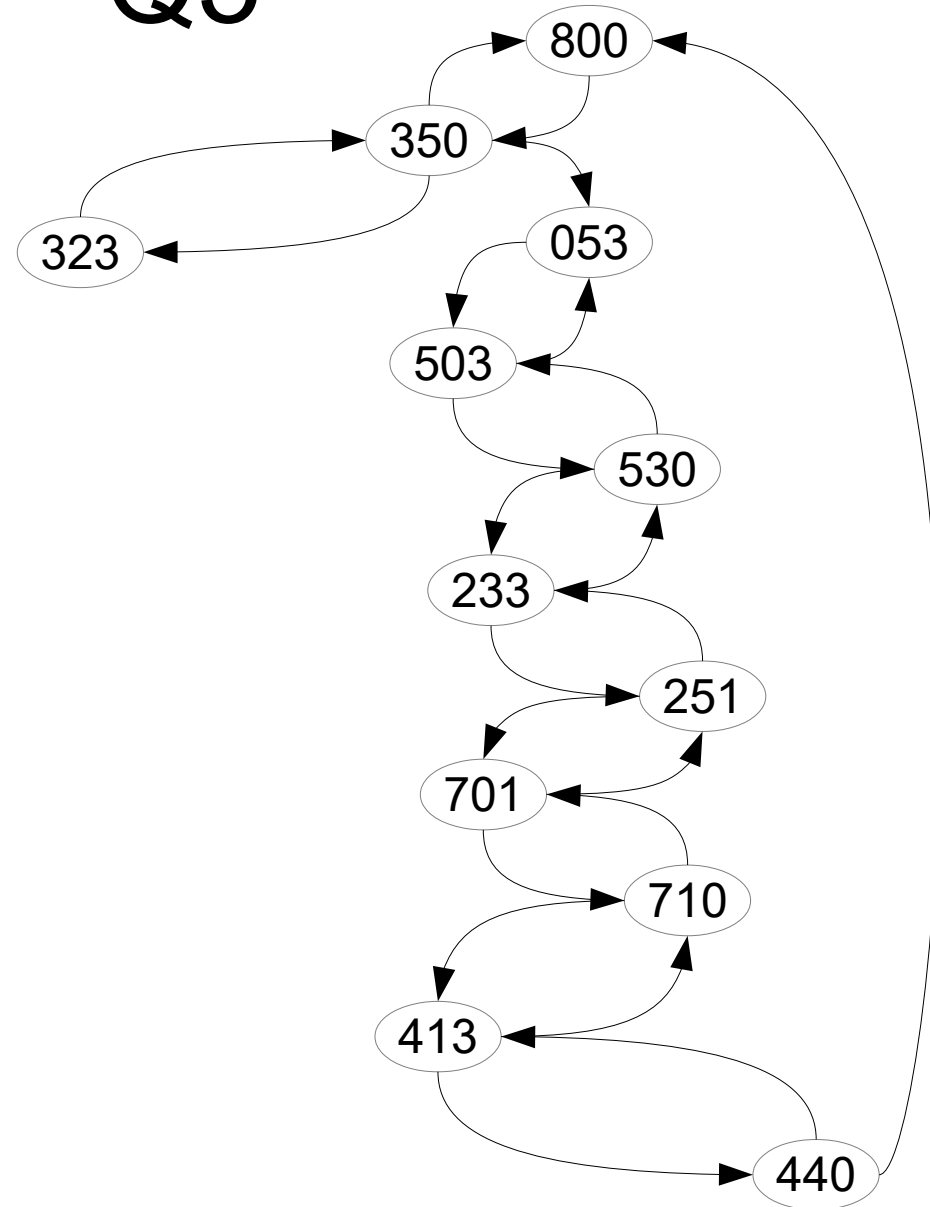
On cherche en fait l'état 440  
On peut faire une recherche à la main de la solution.

Ci contre un graphe partiel, qui montre une solution.

Notez que on peut revenir à l'état de départ directement !

Si on a le graphe complet, et que on donne un poids de 1 à chaque arc, alors un plus court chemin (PCC avec Dijkstra par exemple) de 800 à 440 donnera le nombre minimum de transitions à effectuer.

→ comment obtenir le graphe ?



# Q6

Une fois que on a le graphe complet :

**Q6:** A quel problème sur le graphe correspondent les problèmes suivants :

- trouver une série de transvasements pour passer de l'état initial à l'état final. → parcours
- minimiser le nombre de transvasements nécessaires. → PCC avec poids de 1 partout
- minimiser le volume de liquide déplacé. → PCC avec Volume déplacé à chaque transition

# Q7

**Q7:** Résolvez le problème de la minimisation du nombre de transvasements nécessaires et du nombre de litres déplacés. Pas FAIT : en fait correspond à la génération du graphe

Principe de la génération du graphe :

- Partir de l'état de départ 800
- Pour chaque état  $e$ , trouver les états voisins possibles
- Pour chaque état voisin  $v$ 
  - Si c'est un nouvel état alors l'ajouter au graphe
  - Dans tous les cas, ajouter l'arc  $e \rightarrow v$

```
genGraphe(e0, transitions() )      // pour nous : G = genGraphe(800, transvasements() )
    ensemble aTraiter = { e0 }
    graphe G = (X, U). ensembles X = { e0 }, U = { } (sommets, arcs)
    Tant que aTraiter <> { }
        e = retirerUnElementDe(aTraiter)
        voisins = transitions(e) // la fonction qui calcule les états atteignables
        pour chaque élément e' de voisins
            si appartient(e', X) == FAUX // nouvel état découvert
                ajoutElement(e', X)      // ajout au graphe
                ajoutElement(e', aTraiter) // il faudra trouver ses voisins
        fsi
        ajoutElement(arc e->e', U)
    finpour
    retourner G
```

Q13 à Q16