Introduction à l'optimisation et la recherche opérationnelle (2018–2019)

Professeur : Michel Bierlaire, Assistants responsables : Nikola Obrenovic et Nourelhouda Dougui

Plus court chemin - corrigé (9 novembre 2018)

Question 1:

1. Il s'agit ici de résoudre le probème du plus court chemin dans un graphe. Ce problème étant un cas particulier du problème de transbordemant, on peut le résoudre avec l'algorithme du simplexe. Cependant, cet algorithme n'est pas le plus efficace pour ce problème en particulier car il n'exploite pas la structure du problème.

   D'un autre côté, l'algorithme générique du plus court chemin est un algorithme dédiée pour ce problème. Il peut résoudre le problème du plus court chemin sur n'importe quel graphe qui ne contient pas de cycle négatif.

   Comme les coûts de tous les arcs de ce graphe sont positifs, nous pouvons appliquer l'algorithme de Dijkstra qui est l'algorithme le plus efficace pour retrouver les plus courts chemins. En effet, Dijkstra traite chaque noeud du graphe une seul fois. Il faut souligner qu'on ne peut appliquer cet algorithme que si tous les arcs du graphe sont positifs. En effet, Dijkstra n'a pas de mécanisme pour detecter un cycle négatif et il aurait un nombre infini d'itération si un tel cycle existait dans le graphe.

2. Toutes les itérations de l'algorithme sont présentées dans le tableau ci-dessous.

Itération	S	i	λ_1	λ_2	λ_3	λ_4	λ_5	λ_6	λ_7	λ_8
0	$\{1\}$	1	0	∞	∞	∞	∞	∞	∞	∞
1	$\{2,3\}$	2	0	10	12	∞	∞	∞	∞	∞
2	$\{3,4,5\}$	3	0	10	12	19	17	∞	∞	∞
3	$\{4,5\}$	5	0	10	12	19	15	∞	∞	∞
4	$\{4,6,7\}$	4	0	10	12	19	15	26	20	∞
5	$\{6,7\}$	7	0	10	12	19	15	26	20	∞
6	$\{6,8\}$	8	0	10	12	19	15	26	20	23
7	$\{6\}$	6	0	10	12	19	15	26	20	23
8	$\{\}$		0	10	12	19	15	26	20	23

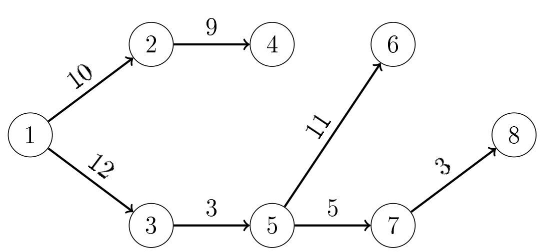
   La table ci-dessous montre les valeurs de π à chaque itération de l'algorithme de Dijkstra.

Iteration	π_1	π_2	π_3	π_4	π_5	π_6	π_7	π_8
0	-1	-1	-1	-1	-1	-1	-1	-1
1	-1	1	1	-1	-1	-1	-1	-1
2	-1	1	1	2	2	-1	-1	-1
3	-1	1	1	2	3	-1	-1	-1
4	-1	1	1	2	3	5	5	-1
5	-1	1	1	2	3	5	5	-1
6	-1	1	1	2	3	5	5	7
7	-1	1	1	2	3	5	5	7
8	-1	1	1	2	3	5	5	7

3. La valeur final du vecteur π nous permet de retrouver les plus courts chemins en utilisant une procédure de retour en arrière. Par exemple, pour retrouver le plus court chemin du noeud 1 au noeud 7, on utilise $\pi_7 = 5$. Ainsi, le prédécesseur de 7 est 5. Comme $\pi_5 = 3$, alors le prédécesseur de 5 est 3 et ainsi de suite. On obtient le chemin

$$1 \rightarrow 3 \rightarrow 5 \rightarrow 7.$$

En appliquant cette procédure à tous les noeuds du graphe, on obtient l'arbre suivant qui est l'arbre des plus courts chemins du graphe.



4. Comme le graphe comprend des arcs avec des coûts négatifs, on applique l'algorithme générique du plus court chemin. Les itérations de l'algorithme sont présentées dans le tableau ci-dessous.

Iteration	S	i	λ_1	λ_2	λ_3	λ_4	λ_5	λ_6
0	$\{1\}$	1	0	∞	∞	∞	∞	∞
1	$\{2,3\}$	2	0	-1	3	∞	∞	∞
2	$\{3,4,5\}$	3	0	-1	3	6	4	∞
3	$\{2,4,5\}$	2	0	-6	3	6	4	∞
4	$\{4,5\}$	5	0	-6	3	1	-1	∞
5	$\{3,4,6\}$	3	0	-6	-8	1	-1	-3
6	$\{2,4,6\}$	2	0	-17	-8	1	-1	-3
7	$\{4,5,6\}$	5	0	-17	-8	-10	-12	-3
8	$\{3,4,6\}$	3	0	-17	-19	-10	-12	-14
9	$\{2,4,6\}$	2	0	-28	-19	-10	-12	-14
10	$\{4,5,6\}$	5	0	-28	-19	-21	-23	-14
11	$\{3,4,6\}$	3	0	-28	-30	-21	-23	-25
11	$\{2,4,6\}$	2	0	-39	-30	-21	-23	-25
12	$\{4,5,6\}$	5	0	-39	-30	-32	-34	-25
13	$\{3,4,6\}$	3	0	-39	-41	-32	-34	-36
14	$\{2,4,6\}$	-	0	-50	-41	-32	-34	-36

A l'itération 14 de l'algorithme, on obtient $\lambda_2 = -50$: on a donc

$$\lambda_2 < 0$$

et

$$\lambda_2 < (m-1) \min_{(i,j)\in\mathcal{A}} c_{ij} = 5 \cdot (-9) = -45.$$

avec m le nombre de noeuds dans le graphe. Ainsi, un cycle négatif est detecté dans le graphe. L'algorithme s'arrête et on ne peut pas retrouver le plus court chemin simple.

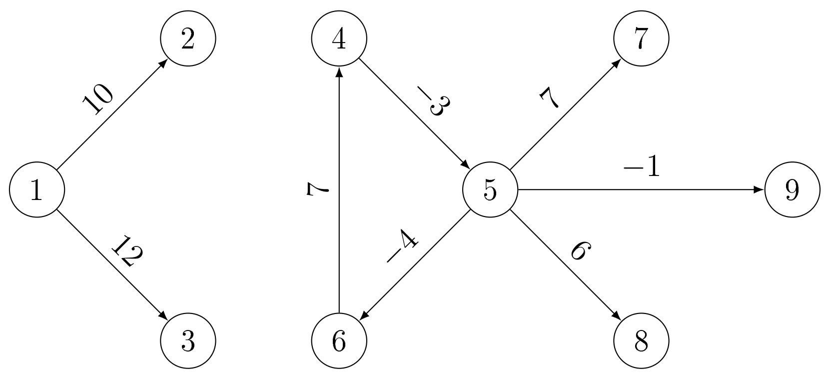
Question 2:

1. As the graph $(\mathcal{N}, \mathcal{A})$ contains arcs with negatif costs, we apply the shortest path algorithm in order to find the shortest paths from node 1 to all other nodes. The iterations of the algorithm are presented in the following table :

   Here, the shortest path algorithm finds the shortest paths from node 1 to all other nodes. Indeed, this algorithm can solve any problem that does not involve a negative costs cycle.

Iteration	S	i	λ_1	λ_2	λ_3	λ_4	λ_5	λ_6	λ_7	λ_8	λ_9
0	$\{1\}$	1	0	∞	∞	∞	∞	∞	∞	∞	∞
1	$\{2,3\}$	2	0	10	12	∞	∞	∞	∞	∞	∞
2	$\{3,4,6\}$	4	0	10	12	-2	∞	14	∞	∞	∞
3	$\{3,5,6,7\}$	5	0	10	12	-2	-5	14	14	∞	∞
4	$\{3,6,7,8,9\}$	6	0	10	12	-2	-5	-9	2	1	-6
5	$\{3,7,8,9\}$	9	0	10	12	-2	-5	-9	2	1	-6
6	$\{3,7,8\}$	8	0	10	12	-2	-5	-9	2	1	-6
7	$\{3,7\}$	7	0	10	12	-2	-5	-9	2	1	-6
8	$\{3\}$	3	0	10	12	-2	-5	-9	2	1	-6
9	$\{\}$	-	0	10	12	-2	-5	-9	2	1	-6

2. In order to find the Bellman's subnetwork of the graph, we consider the subgraph $(\mathcal{N}, \mathcal{A}')$ where $\mathcal{N}$ is the set of all nodes of the network and $\mathcal{A}'$ is generated as follows. For each node j different from the origin, we select one arc (i, j) such that $\lambda_j = \min_{(i,j)\in\mathcal{A}}(\lambda_i + c_{ij})$. If several arcs verify the equation, then we arbitrarily select one of them. We obtain the following subnetwork :



By construction, the number of arcs in Bellman's subnetwork is $m-1$ where m is the number of nodes. Therefore, if the subnetwork does not contain any cycle, then it is a spanning tree. Actually, if all cycles in a network have positive lengths(> 0), the Bellman's network is a tree (Theorem 23.12). However, this network has a cycle of zero length $4 \to 5 \to 6 \to 4$, so that the sufficient condition of theorem 23.12 is not verified. And thus, the Bellman's subnetwork we found is not a tree and contains the cycle $4 \to 5 \to 6 \to 4$.
Note, that if Bellman's subnetwork is not a tree, it isn't connected.

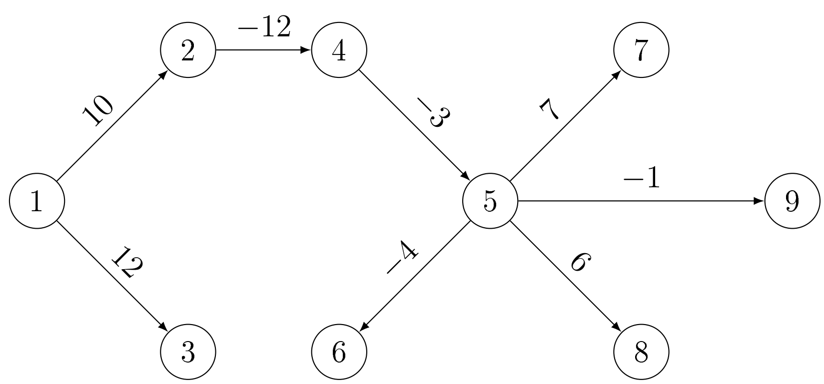
3. We are going to show that there exists a spanning tree satisfying Bellman's equations.

   As the Bellman's subnetwork we found contains a cycle and isn't connected, in order to find another Bellman's subnetwork that is a tree, we need to replace one arc in the cycle (here $(6,4)$) by another arc verifying Bellman's equation that connects the subnetwork (here $(2,4)$).

   Indeed, we observe that

$$\lambda_4 = -2 = \lambda_2(10) + c_{24}(-12) = \lambda_6(-9) + c_{64}(7) = \min_{(i,j)\in\mathcal{A}}(\lambda_i + c_{ij}).$$

   Thus, if we remove $(6,4)$ and add $(2,4)$, then Bellman's equation are going to still be satisfied. We end up with the following shortest path spanning tree :



Question 3:

The optimization problem form suggests a transhipment problem.

Each constraint corresponds to a node : there are 6 constraints and thus 6 nodes belong to the set of nodes $\mathcal{N}$.

Each variable corresponds to an arc of the network. For example, the variable x_{12} is an arc from node 1 to node 2. There are then 8 arcs which form the set $\mathcal{A}$ of arcs.

The objective function is of the form

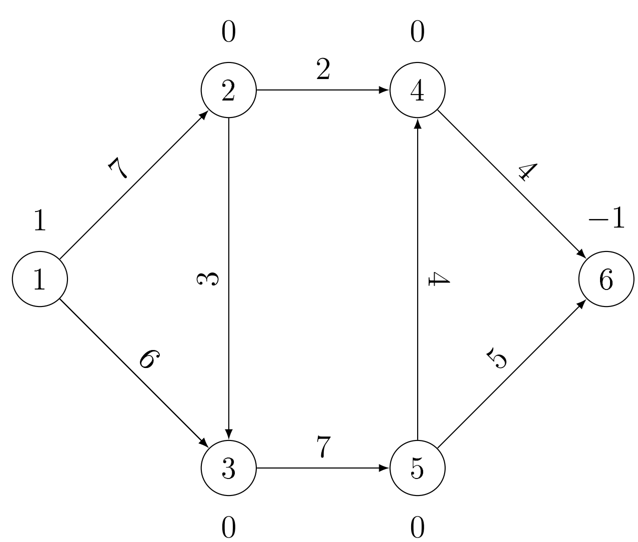
$$\min \sum_{(i,j)\in\mathcal{A}} c_{ij}x_{ij}.$$

Thus, we associate to each arc a cost which is the corresponding coefficient in the objective function.

Each constraint associates the divergence of the node to the supply and demand of that node. Here, node 1 is a supply node which emits 1 flow unit. Node 6 is a destination node which receives 1 flow unit. Each other node is a transit node which has a divergence of 0.

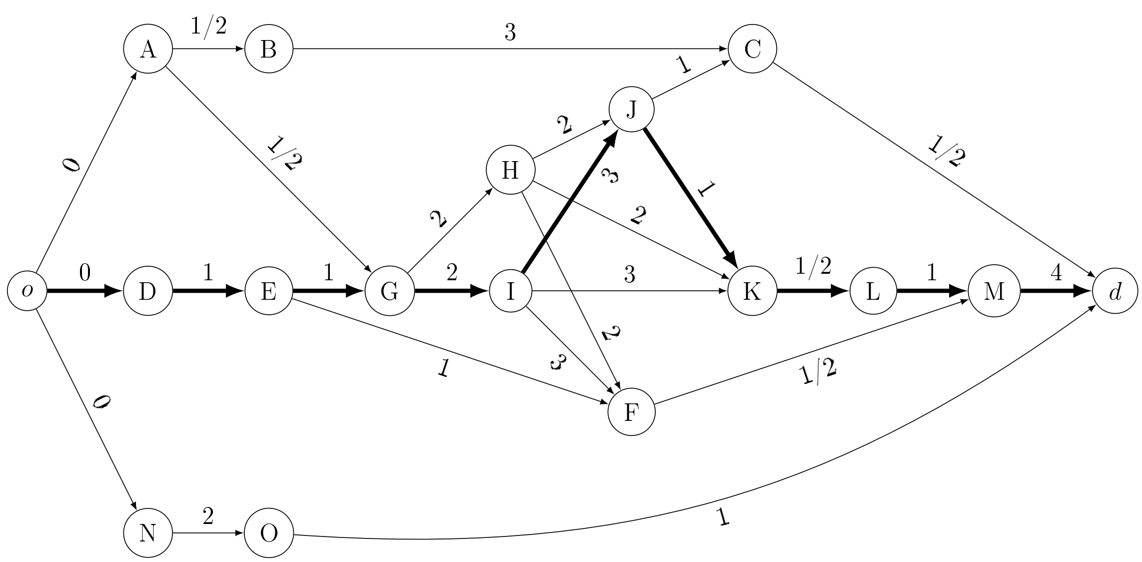
These are the characteristics of a specific transhipment problem which is the shortest path problem.

The resulting network is shown in the Figure below.



Question 4:

1. We want to model the problem as a mathematical network $(\mathcal{N}, \mathcal{A})$.
   First we define the set of nodes $\mathcal{N}$:
   — A node is associated with each task.
   — We create an artificial node o representing the beginning of living room renovation,
   — and an artificial node d representing its ending.
   The set of arcs is defined as :
   — For each task j, we add an arc (i, j) for each predecessor i.
   — For each task j without predecessor, we define an arc (o, j).
   — For each task i without successor, we define an arc (i, d).
   To finish, a cost is associated with each arc (i, j) which is the duration of task i. The cost of each arc of the form (o, j) is zero.
   Thus, we get the following network :



2. To answer this question we must solve a longest path problem. To find the longest path from o to d, we apply the shortest path algorithm over the network with the opposite sign of the costs.
   Each iteration of the algorithm is presented in the Table 1 below.

It.	S	i	λ_o	λ_A	λ_B	λ_C	λ_D	λ_E	λ_F	λ_G	λ_H	λ_I	λ_J	λ_K	λ_L	λ_M	λ_N	λ_O	λ_d
0	$\{o\}$	o	0	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞
1	$\{A, D, N\}$	D	0	0	∞	∞	0	∞	∞	∞	∞	∞	∞	∞	∞	∞	0	∞	∞
2	$\{A, E, N\}$	E	0	0	∞	∞	0	-1	∞	∞	∞	∞	∞	∞	∞	∞	0	∞	∞
3	$\{A, F, G, N\}$	G	0	0	∞	∞	0	-1	-2	-2	∞	∞	∞	∞	∞	∞	0	∞	∞
4	$\{A, F, H, I, N\}$	I	0	0	∞	∞	0	-1	-2	-2	-4	-4	∞	∞	∞	∞	0	∞	∞
5	$\{A, F, H, J, K, N\}$	J	0	0	∞	∞	0	-1	-2	-2	-4	-4	-7	-7	∞	∞	0	∞	∞
6	$\{A, C, F, H, K, N\}$	K	0	0	∞	-8	0	-1	-2	-2	-4	-4	-7	-8	∞	∞	0	∞	∞
7	$\{A, C, F, H, L, N\}$	L	0	0	∞	-8	0	-1	-2	-2	-4	-4	-7	-8	-8.5	∞	0	∞	∞
8	$\{A, C, F, H, M, N\}$	M	0	0	∞	-8	0	-1	-2	-2	-4	-4	-7	-8	-8.5	-9.5	0	∞	∞
9	$\{A, C, F, H, N\}$	C	0	0	∞	-8	0	-1	-2	-2	-4	-4	-7	-8	-8.5	-9.5	0	∞	-13.5
10	$\{A, F, H, N\}$	F	0	0	∞	-8	0	-1	-2	-2	-4	-4	-7	-8	-8.5	-9.5	0	∞	-13.5
11	$\{A, H, N\}$	H	0	0	∞	-8	0	-1	-2	-2	-4	-4	-7	-8	-8.5	-9.5	0	∞	-13.5
12	$\{A, N\}$	A	0	0	-1/2	-8	0	-1	-2	-7	-4	-4	-7	-8	-8.5	-9.5	0	∞	-13.5
13	$\{B, N\}$	B	0	0	-1/2	-8	0	-1	-2	-7	-4	-4	-7	-8	-8.5	-9.5	0	∞	-13.5
14	$\{N\}$	N	0	0	-1/2	-8	0	-1	-2	-7	-4	-4	-7	-8	-8.5	-9.5	0	-2	-13.5
15	$\{O\}$	O	0	0	-1/2	-8	0	-1	-2	-7	-4	-4	-7	-8	-8.5	-9.5	0	-2	-13.5
16	$\{\}$	-	0	0	-1/2	-8	0	-1	-2	-7	-4	-4	-7	-8	-8.5	-9.5	0	-2	-13.5

TABLE 1 – Iteration of the shortest path algorithm to find the longest path from o to d.

According to each of the steps of shortest path algorithm we have the vector π

$$\pi_o = -1, \quad \pi_A = o, \quad \pi_B = A, \quad \pi_D = o, \quad \pi_E = D, \quad \pi_F = H,$$

$$\pi_G = E, \quad \pi_H = G, \quad \pi_I = G, \quad \pi_J = I, \quad \pi_K = J, \quad \pi_L = K,$$

$$\pi_M = L, \quad \pi_N = o, \quad \pi_O = N \text{ and } \pi_d = M.$$

The longest path from o to d is represented in bold in the above network. Each node among that path corresponds to a critical tasks. The critical tasks are therefore D, E, G, I, J, K, L and M. The minimal duration of the work is 13.5 days.

3. The optimal labels of the algorithm are the earliest start for each tasks (column δ_i in the Table below with sign "-").

   In order to obtain the latest start for each task, we have to invert the direction and the sign of the cost for each arc. Then, we apply the shortest path algorithm starting from node d. Now, the label of node i represent the opposite of the minimum time required between the beginning of task i and the end of task d. It is then sufficient to subtract this minimum time from the minimal duration of the work (which is here 13.5 days).

   For example, the new label of node C is $-\frac{1}{2}$. The latest start of task C is then $13.5 - 0.5 = 13$. The latest start of the each task is given in column φ_i in the Table below.

   Note that for all critical tasks, the earliest start corresponds to the latest start.

Ranking	Task	Duration	Predecessor	Successor	Earliest start δ_i	Latest start φ_i
1	o	0	-	A, D, N	0	0
2	A	1/2	o	B, G	0	1.5
3	D	1	o	E	0	0
4	N	2	o	O	0	10.5
5	B	3	A	C	1/2	10
6	E	1	D	G, F	1	1
7	O	1	N	d	2	12.5
8	G	2	A, E	H, I	2	2
9	H	2	G	F, J, K	4	5
10	I	3	G	F, J, K	4	4
11	J	1	H, I	C, K	7	7
12	F	1/2	E, H, I	M	7	9
13	C	1/2	B, J	d	8	13
14	K	1/2	H, I, J	L	8	8
15	L	1	K	M	8.5	8.5
16	M	4	L, F	d	9.5	9.5
17	d	0	C,M,O	-	13.5	13.5