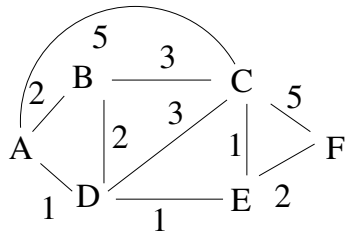
Algorithmique
Graphes – Parcours
(Correction)

Exercice 1 :

Exécuter l'algorithme de DIJKSTRA suivant sur le graphe d'en face avec $s = A$:

Entrées : Un graphe $G = (V, E)$ et une source s
Une fonction poids : $w : E \to \mathbb{R}_{\geq 0}$
Sorties : un vecteur distance $d = d(s, t)$ et
une fonction père $\pi : V \to V$.
Initialisation :
pour chaque sommet v de V **faire**
  $d[v] = +\infty$; $\pi[v] = \text{NULL}$;
fin pour
$d[s] = 0$; $\pi[s] = s$;
$Q = V$; $S = \emptyset$
Boucles :
tant que $Q \neq \emptyset$ **faire**
  $u = \min(Q, d)$
  $S = S - \{u\}$
  pour chaque sommet v voisin de u faire **faire**
    si $d[v] > d[u] + w(u, v)$ **alors**
      $d[v] = d[u] + w(u, v)$
      $\pi[v] = u$
    fin si
  fin pour
fin tant que
retourner (d, π).
▷



Iterat.	Q	A	B	C	D	E	F
init.	$\{ABCDEF\}$	$(0, \emptyset)$	$(\infty, \emptyset)$	$(\infty, \emptyset)$	$(\infty, \emptyset)$	$(\infty, \emptyset)$	$(\infty, \emptyset)$
1	$\{BCDEF\}$	$(0, A)$	$(2, A)$	$(5, A)$	$(1, A)$	$(\infty, \emptyset)$	$(\infty, \emptyset)$
2	$\{BCEF\}$	$(0, A)$	$(2, A)$	$(4, D)$	$(1, A)$	$(2, D)$	$(\infty, \emptyset)$
3	$\{CEF\}$	$(0, A)$	$(2, A)$	$(4, D)$	$(1, A)$	$(2, D)$	$(\infty, \emptyset)$
4	$\{CF\}$	$(0, A)$	$(2, A)$	$(3, E)$	$(1, A)$	$(2, D)$	$(4, E)$
5	$\{F\}$	$(0, A)$	$(2, A)$	$(3, E)$	$(1, A)$	$(2, D)$	$(4, E)$
6	$\emptyset$	$(0, A)$	$(2, A)$	$(3, E)$	$(1, A)$	$(2, D)$	$(4, E)$

Exercice 2 :

On note la monnaie d'un pays A par μ_A. Le taux de change entre un pays A et un second pays B est noté $a(A, B)$ et concrètement on échange 1 unité de μ_A en $a(A, B)$ μ_B (exemple $a(\text{USA, Europe}) = 0.82$ donc 1 US dollar vaut sur ce marché 0.82 euros).

1. Dans quelle(s) condition(s) peut-on espérer s'enrichir en ne faisant que des échanges et comment feriez-vous pour savoir si cela est possible ?

▷ *On peut supposer (rêve !) qu'un US dollar vaut* 0.82 *euro, qu'un euro vaut* 129.7*yen, qu'un yen vaut* 12 *livres turcs. Et qu'un livre turc vaut* 0.0008 *US dollar. Du coup, en échangeant* 1 *US dollar on peut avoir*

$$.82 \times 129.7 \times 12 \times 0.0008 \sim 1.02 \, US \ dollar.$$

(2% de bénéfice rien qu'avec l'algorithme qui cherche un cycle de poids < 0 *...)*

La solution c'est donc d'utiliser Bellman-Ford sur un graphe dirigé mais modélisé de manière astucieuse (pour détécter le cycle de poids négatif).

On remarque qu'une suite d'arcs amenant à un circuit

$$a(P_1, P_2).a(P_2, P_3). \cdots a(P_{k-1}, P_k).a(P_k, P_1) > 1$$

si et seulement si

$$\frac{1}{a(P_1, P_2)} \cdots \frac{1}{a(P_{k-1}, P_k)} \frac{1}{a(P_k, P_1)} < 1\,.$$

Donc

$$\log\left(\frac{1}{a(P_1, P_2)}\right) + \cdots + \log\left(\frac{1}{a(P_k, P_1)}\right) < 0\,.$$

On en conclut qu'il faut mettre sur l'arc $A \to B$ le poids

$$\log\left(\frac{1}{a(P_A, P_B)}\right) = -\log a(P_A, P_B)$$

et lancer l'algorithme de Bellman-Ford *avec l'appendice qui détecte les cycles négatifs (cf. cours).*

Exercice 3 :

Dans cet exercice, vous dévez organiser un tour en vélos pour vos clients. On a une carte de n villes connectées deux à deux par des routes cyclables directes. Une telle route entre deux villes u et v a une distance $d(u, v)$. De plus, rester une nuit dans une cité v coûte $c(v)$.
Un client a toujours ses exigences (ici 4). Le client présent veut (i) partir d'une ville s, (ii) arriver à une ville t en (iii) m étapes journalières en sachant que dans une étape ce client ne voudrait pas faire plus qu'une distance $u(k)$ avec $k \in [1, 2, \cdots, m]$.
Votre travail consiste à trouver un tour de m jours sans que le client ne reste jamais dans la même ville durant 2 nuits consécutives et sans que le client ne pédale pendant $u(k)$ le jour k. Il est à noter que le client peut voyager à vélo à travers différentes villes dans un même jour. Et vous devez minimiser les coûts des étapes du tour.

1. Donner un algorithme efficace pour ce problème, i.e. donnant explicitement un tour de longueur m :

$$s = v_0, v_1, \cdots, t = v_m$$

avec un coût minimum et telle que $d(v_{i-1}, v_i) \leq u(i)$.

▷ *Dans un premier temps, on calcule tous les chemins les plus cours entre v_i et v_j (pour tout couple i, j) en utilisant l'algorithme de* Floyd-Warshall. *On stocke donc les "poids" $W_{i,j}$ ainsi que les chemins entre v_i et v_j.*

Ensuite, on construit un graphe dirigé H avec $n \times (m+1)$ sommets où m est le m du sujet, i.e. la durée du tour. Chaque sommet de ce nouveau graph H sera étiqueté v_{ip} pour $i \in \{1, 2, \cdots, n\}$ et $p \in \{0, 1, \cdots, m\}$ et correspondra à l'option "rester dans la cité i le jour p".

Pour tout $i, j \in \{1, 2, \cdots, n\}$ avec $i \neq j$ ET $p \in \{1, \cdots, m\}$ rajouter l'arête $(v_{i(p-1)}, v_{jp})$ dans H si $W_{i,j} \leq u(p)$. Ceci signifie que le client peut rouler de la ville i à la ville j sans excèder la limite $u(p)$ pour le jour p. Pour chaque sommet v_{ip} de H, on mettra le poids $c_i = c(v_i)$ correspondant à la nuitée dans la ville.

Le tour le moins coûteux est simplement le chemin allant de v_{s0} à v_{tm} tel que le coût total par sommet soit minimum.

On transforme le problème en problème de plus court chemin :
pour chaque arc $(u \to v)$ nous pouvons lui assigner le coût de la nuitée en v. Ainsi le problème se résoud en utilisant un algorithme de plus court chemin sur les graphes dirigés : l'algorithme de Dijkstra *sur le graphe H à $n(m+1)$ sommets et $O(n^2m^2)$ arcs. Il a donc pour complexité $O((n^2m^2 \log(nm))$.*

Exercice 4 :

Un groupe de personnes peuvent s'envoyer des messages. Un message envoyé par une personne S est parfaitement reu par une personne T mais selon l'entente entre le couple S et T, T peut

- soit détruire immédiatement le message avec une probabilité p,
- soit le transmettre à une autre personne avec une probabilité $1 - p$.

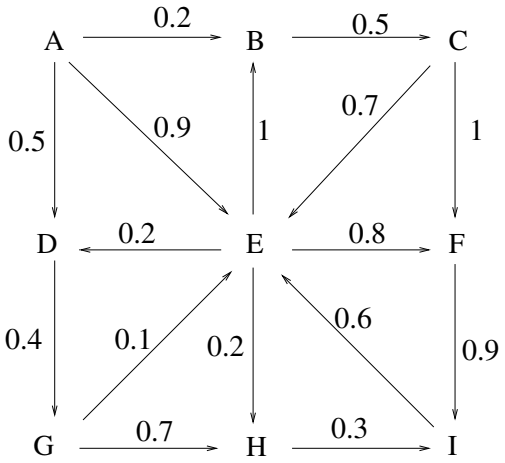
1. Modéliser l'envoi de message entre deux personnes.

▷ *On a un graphe dirigé. Entre les deux sommets S et T on met un arc de poids $1-p$: le poids complémentaire p reflète la destruction immédiate du message, tout se passe comme si dans ce cas on n'a rien envoyé.*

Les personnes sont (A)xel, (B)arbara, (C)hantal, (D)on, (E)l-Hadj, (F)ethi, (G)iulia, (H)oang, (I)gor. Un message envoyé de A à B a 80% de chance d'être détruit. De A à D, 50%. De A à E, 10%. De B à C, 50%. De C à E, 30%. De D à G, 60%. De E à D, 80%. De E à F, 20%. De E à H, 80%. De F à I, 10%. De G à E, 90%. De G à H, 30%. De H à I, 70%. De I à E, 40%.
Par ailleurs, un message de C à F ne sera jamais détruit. De même de E à B.

2. Dessiner le graphe qui reflète les probabilités qu'un message envoyé par une personne X soit retransmis par une personne Y.

▷



Malheureusement, Axel diffame sur Igor.

3. Montrer que la probabilité qu'Igor capte le message initié par Axel est d'au moins de 50%.

▷

$$A,E,F,I+A,B,C,F,I+A,B,C,E,F,I+A,D,G,H,I+A,D,G,E,F,I+A,D,G,E,H,I+\cdots > A,E,F,I = 0.648$$

4. Quel est le chemin le plus fiable que le message diffamatoire a suivi ?

▷ *Ici on peut encore calculer à la main : A, E – E, F – F, I pour trouver $.9 \times .8 \times .9 = 0.648$.*

5. Ecrire un algorithme adapté du cours pour trouver le chemin le plus fiable pour un message entre deux personnes quelconques.

▷ *On adapte Dijkstra. On cherche un chemin pour lequel le produit des probabilités soit maximal. Dans ce qui suit, S est l'ensemble des sommets.*

pour *chaque sommet q de S* ***faire***
  $d[q] := 0$
  $\pi[q] := NULL$
fin pour
$d[s] := 1$
$F := S$
tant que $F \neq \emptyset$ ***faire***
  soit $q_i \in F$ le sommet tel que $d[q_i]$ soit maximal
  $F := F - \{q_i\}$
  ($\star$ succ(s) est l'ens. des successeurs de s $\star$)
  pour *tout sommet $q_j \in succq_i$* ***faire***
    si $d[q_j] < d[q_i] \times proba(q_i, q_j)$ ***alors***
      $d[q_j] := d[q_i] \times proba(q_i, q_j)$
      $\pi[q_j] := \pi[q_i]$
    fin si
  fin pour
fin tant que

6. Ecrire un algorithme adapté du cours pour calculer la probabilité qu'un message se retrouve partagé entre deux personnes quelconques.

▷ *Soit $\mathcal{A}$ l'algorithme précédent débutant par un sommet s et dans lequel on a mis $d[s] := 1$. On remarque que pour un voisin v de s tel que "poids"$(s \to v) = p_{s\to v} > 0$ alors on peut relancer $\mathcal{A}$ en supprimant s en partant de v avec $d[v] := p_{s\to v}$. Cette astuce permet alors de calculer récursivement (avec une grande complexité certes) toutes les probabilités des chemins d'une source quelconque à un sommet quelconque.*