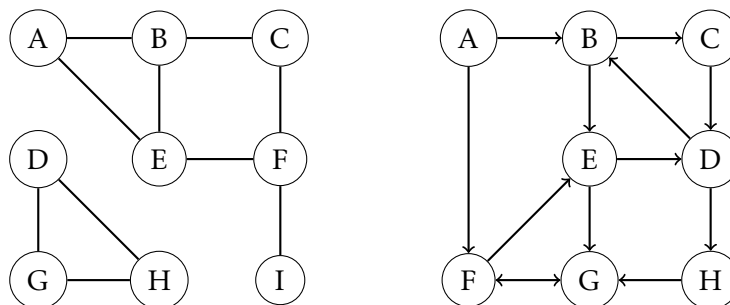


TD 4 & 5 Graphes : parcours

Exercice 1 (Parcours).

1. Appliquez l'algorithme du parcours en largeur sur les deux exemples suivants. Vous marquez les sommets selon leur ordre de visite et leur distance au sommet initial. Quand plusieurs choix s'offrent à vous, vous choisirez les sommets dans l'ordre alphabétique. Donnez les nombres pre et post associés à chaque sommet.



2. Même question avec le parcours en profondeur.

Exercice 2 (Plus courts chemins entre ensembles). Écrivez un algorithme qui prend en entrée un graphe orienté $G = (V, E)$ et deux ensembles de sommets $V_1, V_2 \subseteq V$, et renvoie un chemin de longueur minimale entre V_1 et V_2 .

Exercice 3 (Cycles). Dans un graphe $G = (V, E)$, et pour un sommet $v \in V$, on appelle *cycle issu de v* un chemin $v = v_0 \rightarrow v_1 \rightarrow \dots \rightarrow v_n = v$ dans G tel que les sommets v_1, \dots, v_{n-1} sont distincts deux à deux.

1. Écrivez un algorithme qui prend en entrée un graphe non orienté $G = (V, E)$ et un sommet $v \in V$, et renvoie, s'il existe, un cycle de longueur minimale passant par v ; ou None sinon. La complexité de l'algorithme devra être $O(|V| + |E|)$.
2. Écrivez un algorithme qui prend en entrée un graphe non orienté $G = (V, E)$, et renvoie True s'il existe un cycle dans G , et False sinon.
3. (Bonus) Écrivez un algorithme qui prend en entrée un graphe orienté $G = (V, E)$ et renvoie True s'il existe un cycle dans G , et False sinon.

Exercice 4 (Le loup, la chèvre et le chou (xkcd.com/1134)). Il était une fois une fermière qui alla au marché et acheta un loup, une chèvre et un chou. Pour rentrer chez elle, la fermière loua une barque pour traverser une rivière. Mais la barque ne pouvait contenir, à chaque traversée, que la fermière et l'un de ses achats : le loup, la chèvre ou le chou. S'ils étaient laissés sans surveillance ensembles, le loup dévorerait la chèvre, et la chèvre mangerait le chou.

La fermière traversa la rivière avec l'ensemble de ses achats intacts. Comment a-t-elle fait ?

Exercice 5 (Connexité). Dans un graphe non orienté, une *composante connexe* est un sous-ensemble maximal de sommets qui sont tous reliés par des chemins les uns aux autres.

1. Écrivez un algorithme prenant en entrée un graphe non orienté $G = (V, E)$ et renvoie la liste de ses composantes connexes. La complexité devra être en $O(|V| + |E|)$.
2. Écrivez un algorithme prenant en entrée un graphe non orienté $G = (V, E)$ ainsi qu'un sommet $v \in V$, et renvoie le nombre de sommets de la composante connexe contenant v .
3. Écrivez un algorithme prenant en entrée un graphe non orienté $G = (V, E)$ et renvoie la liste des sommets de sa plus grande composante connexe.

Exercice 6 (Distances). Soit $G = (V, E)$ un graphe connexe non orienté. On note $d(u, v)$ la distance du plus court chemin entre deux sommets u et v .

1. Montrer que pour tout $u, v, w \in V$, $d(u, v) \leq d(u, w) + d(w, v)$.
2. Montrer que pour tout $u, v_1, v_2 \in V$, si $(v_1, v_2) \in E$, alors $|d(u, v_1) - d(u, v_2)| \leq 1$.
3. Montrer que G contient un cycle de longueur impaire si et seulement si, quelque soit $u \in V$, il existe $v_1, v_2 \in V$ tels que $(v_1, v_2) \in E$ et $d(u, v_1) = d(u, v_2)$.

Exercice 7 (Graphes bipartis). Un graphe non orienté $G = (V, E)$ est *biparti* s'il existe une partition (V_1, V_2) de V telle que pour toute arête $(u, v) \in E$, u et v ne sont pas dans la même partition (i.e. on a soit $u \in V_1$ et $v \in V_2$, soit $v \in V_2$ et $u \in V_1$). On dit aussi que G est *2-coloriable*.

1. Montrer que si G est *biparti*, alors il ne contient pas de cycle de longueur impaire.
2. Montrer que si G ne contient que des cycles de longueur paire, alors G est biparti.
3. En utilisant les questions et l'exercice précédents, écrivez un algorithme qui prend en entrée un graphe non orienté $G = (V, E)$ et renvoie `True` si G est biparti, `False` sinon. La complexité de l'algorithme sera $O(|V| + |E|)$.
4. Écrivez un algorithme qui prend en entrée un graphe non orienté $G = (V, E)$ et renvoie, s'il existe, un 2-coloriage de G : autrement dit, renvoie une fonction $c : V \mapsto \{1, 2\}$ telle que $(u, v) \in E \implies c(u) \neq c(v)$.

Exercice 8 (Forte connexité). Un graphe orienté $G = (V, E)$ est dit *fortement connexe* si pour toute paire de sommets $u, v \in V$, v est accessible par un chemin depuis u .

1. Étant donné un graphe orienté $G = (V, E)$, on définit le graphe transposé $G^T = (V, E^T)$ par $E^T = \{(v, u) \in V^2 \mid (u, v) \in E\}$ (autrement dit, on renverse les arêtes de G).
Pour $G = (V, E)$ un graphe orienté, et $u \in V$ un sommet arbitraire, montrez que G est fortement connexe si et seulement si, pour tout sommet $v \in V$, il existe un chemin de u à v à la fois dans G et dans G^T .
2. À partir de la question précédente, écrivez un algorithme qui prend en entrée un graphe orienté $G = (V, E)$, et renvoie `True` si G est fortement connexe, `False` sinon. La complexité de l'algorithme sera $O(|V| + |E|)$.

Remarque : Cette dernière question est très similaire à l'algorithme de Kosaraju, qui permet de déterminer les composantes « fortement connexes » d'un graphe orienté en temps $O(|V| + |E|)$. Vous étudierez ces questions plus précisément au second semestre.

Exercice 9 (Plus courte durée). Quatre vaillantes aventurières doivent traverser un pont délabré de nuit. Alex effectue la traversée en une minute, Clover en deux minutes, Sam en cinq minutes et Britney en huit minutes. Elles ne possèdent qu'une seule torche, sachant qu'il est impossible de traverser le pont sans torche. Et le pont ne peut supporter que le poids de deux personnes à la fois.

1. Comment peuvent-elles passer le pont si la torche ne dure que quinze minutes ? (Indication : on pourra modéliser le problème par un graphe *valué*, i.e. un graphe $G = (V, E, w)$ où $w : E \mapsto \mathbb{N}$ est une fonction de *valuation* qui attribue une durée à chaque arête.)
2. Proposez un algorithme qui calcule la solution minimisant la durée de la traversée.

Remarque : Pour un algorithme plus optimal, vous pouvez chercher les algorithmes dits « de plus courts chemins » (Dijkstra, Bellman-Ford, A^ ...).*