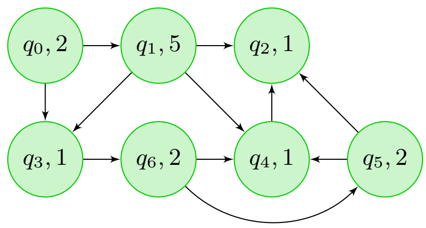
AL5
TD n° 11 : Choses vues

Exercice 1 : PCC et routage

On considère un problème de routage dans un réseau fixe décrit par un graphe non-orienté valué $G = (S, A, w)$ avec $w : A \to \mathbb{N}$ où $w(u, v)$ représente la longueur de l'arc (u, v) (c'est-à-dire le temps de transmission d'un message de u vers v). Chaque nœud représente un routeur qui doit dispatcher les messages qu'il reçoit vers le bon destinataire : lorsque le nœud x reçoit un message pour le nœud y, alors si $x = y$ c'est fini, et sinon, il doit l'envoyer à l'un de ses voisins z afin que z transmette à son tour le message à un voisin, *etc.* Bien sûr on souhaite utiliser des PCC : si x envoie le message pour y à son voisin z, c'est qu'il existe un PCC reliant x à y dont la première arête est (x, z).
Proposer un algorithme qui construit pour chaque nœud x de G, une table des distances $\mathsf{d}_x[-]$ donnant les distances des PCC de x aux autres nœuds (ainsi $\mathsf{d}_x[y]$ est égal à la distance d'un PCC de x à y) et une table de routage $\mathsf{r}_x[-]$ indiquant à quel voisin de x il faut envoyer les messages selon leur destinataire ($\mathsf{r}_x[y] = z$ dans notre exemple).
Donner la complexité de votre algorithme.

Exercice 2 : graphes pondérés

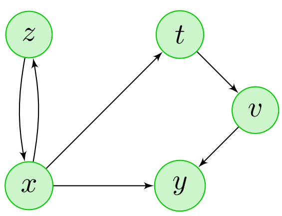
On dispose d'un graphe orienté valué $G = (S, A, w)$ où la fonction $w : S \to \mathbb{R}^+$ associe une valeur réelle positive aux **sommets** (et non aux arcs). La valeur $w(s)$ représente un coût à payer (ou une distance) lorsqu'on arrive dans le sommet s. Ainsi le chemin $s \to s' \to s''$ aura un coût égal à $w(s') + w(s'')$, et plus généralement un chemin $q_1 \to q_2 \to \ldots \to q_n$ aura un coût $\sum_{i=2\ldots n} w(q_i)$.



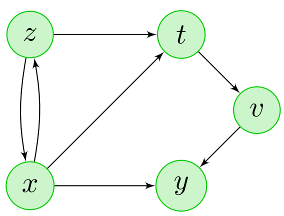
Proposer une méthode pour trouver les coûts minimum pour atteindre chaque sommet depuis un sommet de départ s donné. L'appliquer sur le graphe ci-contre depuis le sommet q_0.

Exercice 3 : toujours des parcours

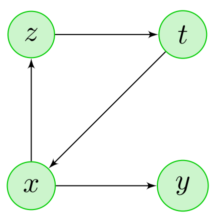
On considère un graphe orienté $G = (S, A)$. Écrire un algorithme $\mathsf{CheminPair}(G, x, y)$ qui renvoie vrai si et seulement s'il existe un chemin de longueur paire de x à y (ici la longueur désigne le nombre d'arcs du chemin).
NB : on considère des chemins quelconques (pas nécessairement simples).
Donner la complexité de votre algorithme.
Appliquer votre algorithme sur les trois graphes ci-dessous pour les sommets x et y.



G_1



G_2



G_3

Exercice 4 : chemins disjoints dans les graphes, théorèmes de Menger

Soit $G = (S, A)$ un graphe orienté et $s \neq t$ deux sommets de G. Deux chemins orientés P_1 et P_2 de s à t sont dits

- *arc-disjoints* s'ils ne partagent aucun arc,
- *sommet-disjoints* si les seuls sommets en commun de P_1 et P_2 sont s et t.

De plus un ensemble d'arcs A' (respectivement de sommets S') *déconnecte* s et t s'il n'existe aucun chemin orienté de s vers t dans le graphe obtenu à partir de G en supprimant tous les arcs de A' (resp. sommets de S').

1. Rappeler pourquoi l'algorithme de Ford Fulkerson fournit une preuve du théorème suivant :

Théorème 1. *Le nombre maximal de chemins deux à deux arc-disjoints de s vers t dans G est égal à la taille minimale d'un ensemble d'arcs qui déconnecte s et t.*

2. À partir de G, on construit un nouveau graphe orienté G' de la façon suivante :

- à chaque sommet $x \in S$ correspondent dans G' deux sommets x_- et x_+ ainsi que l'arc (x_-, x_+) ;
- à chaque arc $(x, y) \in A$ correspond dans G' l'arc (x_+, y_-).

Montrer comment cette définition permet de montrer le théorème suivant :

Théorème 2. *Le nombre maximal de chemins deux à deux sommet-disjoints dans G est égal à la taille minimale d'un ensemble de sommets qui déconnecte s et t.*

3. Montrer que les deux théorèmes précédents restent vrais dans le cas d'un graphe non orienté (avec les notions adhoc de chemins et d'ensembles d'arêtes ou de sommets déconnectants)