Mini-projet numéro 2 - Algorithmes et Complexité.

# Algorithmes pour le Jeu Rouge Bleu

Langages autorisées : Java, Python et C.

Date limite pour envoyer votre projet : déterminée prochainement.

Lien pour déposer le projet :
`https://www.dropbox.com/request/KPs8fjvAEII567gMhtmp`

Projet à réaliser en groupe de 4 (pas nécessairement d'un même groupe de TD).

Groupe à constituer au plus tard le vendredi 3 décembre 2021 (18 heures) en envoyant un courrier électronique à Johny Bond. Un groupe sera considéré FISA, uniquement si tous les membres sont FISA. Les groupes de moins de 4 seront pénalisés, sauf autorisation de Johny Bond (le nombre d'inscrits n'est pas forcément multiple de 4). Si difficulté dans la constitution des groupes, merci de contacter Johny Bond dès que possible.

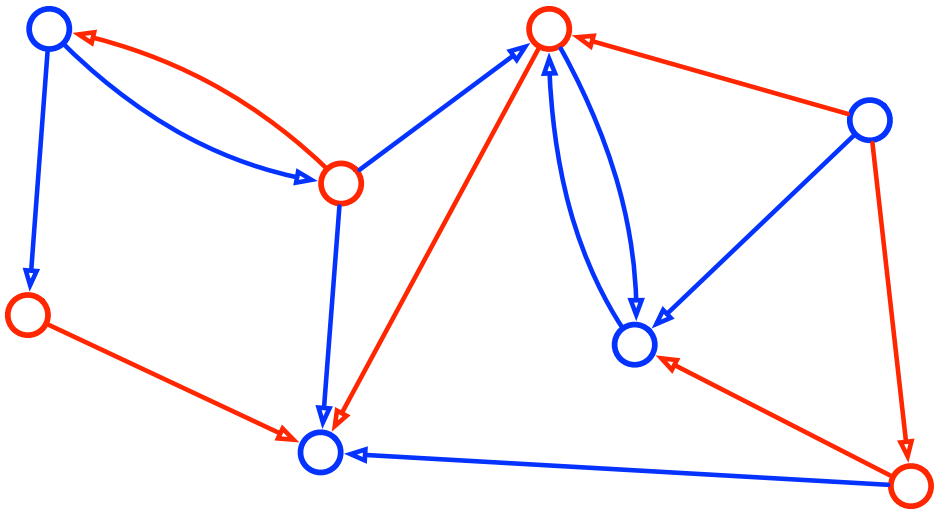
Question sur le projet ? dorian.mazauric@inria.fr

**Vous devez répondre pour chacune des questions (dans un fichier au format PDF obligatoire) et nous faire parvenir le code pour les questions 5 et 6.**

Johny Bond, François Doré, Dorian Mazauric, Christophe Papazian

2021 - 2022

Un graphe orienté bi-coloré $G = (V, A, c)$ est tel que de tout sommet $u \in V$ à tout sommet $v \in V$, il y a au plus un arc et la couleur $c(v)$ de tout sommet $v \in V$ et la couleur $c(a)$ de tout arc $a \in A$, est *rouge* ou *bleu*. Un exemple de graphe bi-coloré est visible ci-dessous.

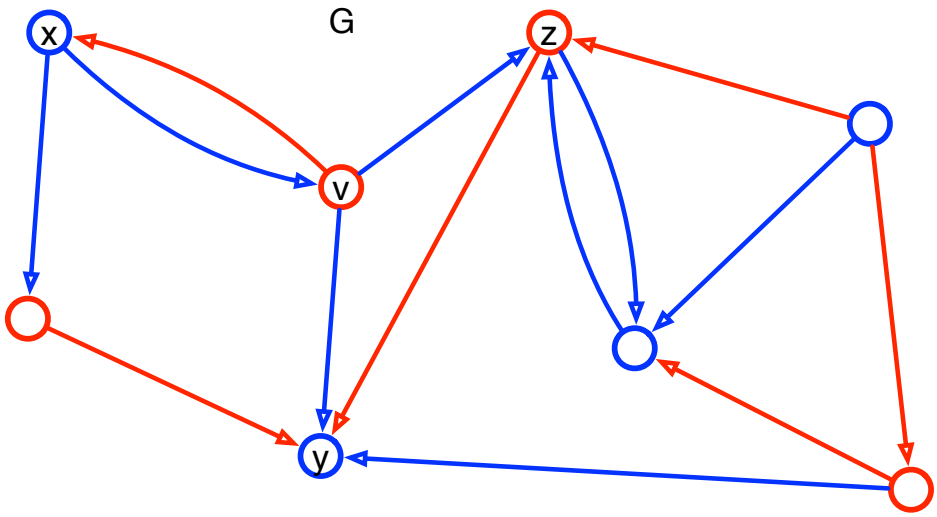


L'ensemble $N_G^-(u)$ est l'ensemble des voisins entrants d'un sommet $u \in V$. L'ensemble $N_G^+(u)$ est l'ensemble des voisins sortants d'un sommet $u \in V$.

Étant donné $G = (V, A, c)$ et $v \in V$, nous définissons le graphe orienté bi-coloré $G' = (V', A', c') = f(G, v)$ comme suit :

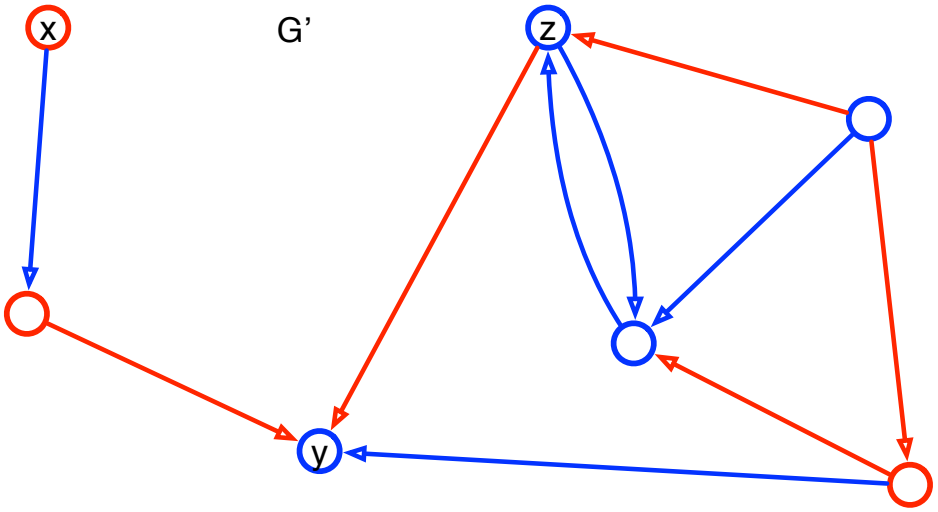
— $V' = V \setminus \{v\}$ ;
— $A' = A \setminus (\{(u, v) \in A \mid u \in V\} \cup \{(v, u) \in A \mid u \in V\})$ ;
— $c'(a) = c(a)$ pour tout $a \in A'$ ;
— $c'(u) = c(e)$ pour tout $u \in N_G^+(v)$ avec $e = (v, u)$ ; et $c'(u) = c(u)$ pour tout $u \in V' \setminus N_G^+(v)$.

Autrement dit, $G'$ est une copie de $G$ dans laquelle tout arc sortant de $v$ vers $v'$ donne sa couleur à $v'$ et en supprimant ensuite le sommet $v$. Reprenons notre exemple avec le sommet $v$ indiqué ci-dessous.




Nous construisons $G'$ en modifiant la couleur de deux des trois voisins sortants de $v$, c'est-à-dire
— le sommet $x$ devient rouge car l'arc de $v$ vers $x$ est rouge,

— le sommet $z$ devient bleu car l'arc de $v$ vers $z$ est bleu,
— le sommet $y$ reste bleu car l'arc de $v$ vers $y$ est bleu.

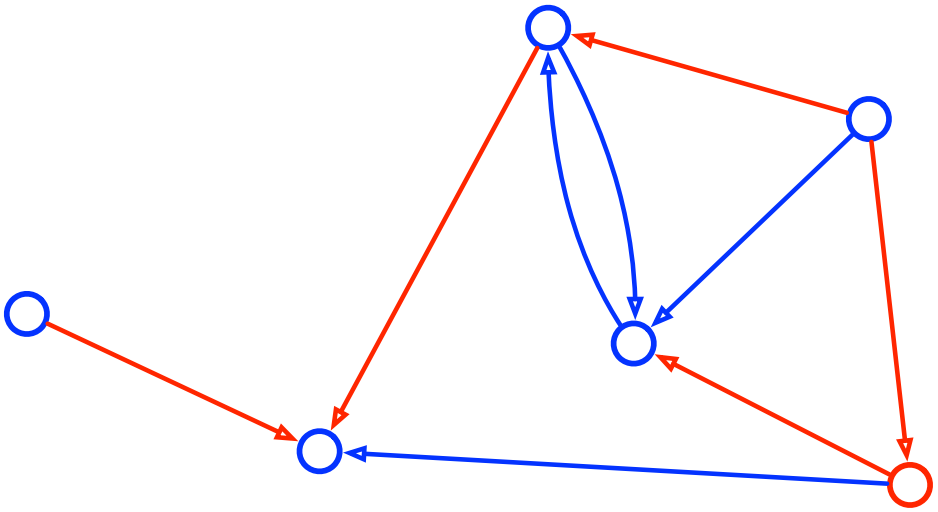


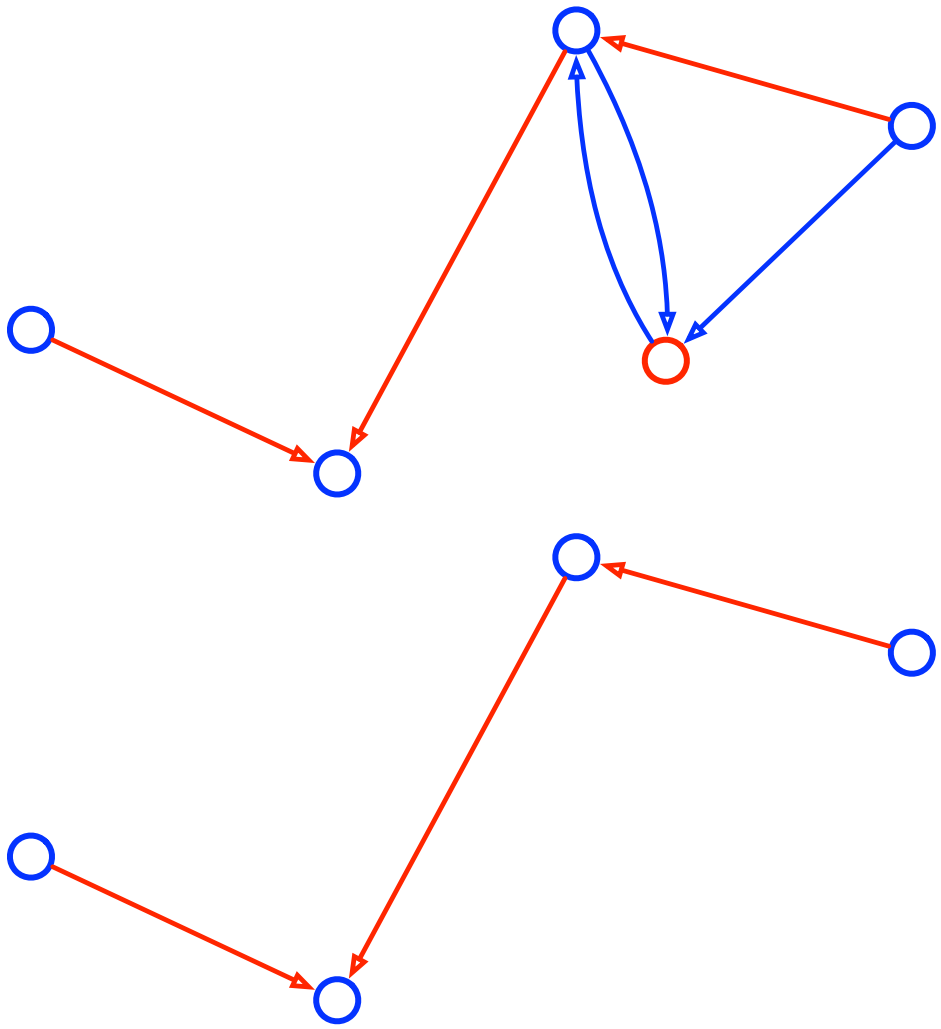

Définissions $G_0 = G$. Une *séquence rouge* de sommets de $G$ est un vecteur $(v_0, \ldots, v_{t-1})$ qui satisfait $G_{i+1} = f(G_i, v_i)$ pour tout $i \in [\![0, t-1]\!]$ et $c_i(v_i)$ est rouge pour tout $i \in [\![0, t-1]\!]$, avec $c_i$ la coloration dans $G_i$.

Autrement dit, une *séquence rouge* est une suite de suppressions de sommets rouges dans les graphes obtenus après ces suppressions successives.

**Problème rouge-bleu.** Étant donné un graphe orienté bi-coloré $G = (V, A, c)$ et un entier $k \geq 1$, le problème rouge-bleu consiste à décider s'il existe une *séquence rouge* de sommets de $G$ de taille au moins $k$.

Dans la suite, nous montrons que la réponse est **oui** pour notre graphe exemple et pour $k = 4$. Vous trouverez ci-dessous la suite de la séquence (c'est-à-dire les trois suppressions supplémentaires)





**Question 1.** Considérons l'exemple précédent. Existe-t-il une solution pour le problème rouge-bleu avec $k = 5$ ? $k = 6$ ? $k = 7$ ? Dans chacun des cas, donner la séquence associée ou montrer son inexistence.

**Question 2.** Montrer que le problème rouge-bleu est NP-complet. Pour cela vous pouvez utiliser une des problèmes vus en cours.

Considérons la classe d'instances suivantes. L'ensemble des $n$ sommets est $V = \{v_0, \ldots, v_{n-1}\}$. Il y a $n-1$ arcs, à savoir exactement un arc entre $v_i$ et $v_{i+1}$ pour tout $i \in [\![0, n-2]\!]$ (soit de $v_i$ à $v_{i+1}$, soit de $v_{i+1}$ à $v_i$). Un exemple de graphe est décrit ci-dessous.

Considérons à présent la version maximisation du problème rouge-bleu qui consiste à trouver une *séquence rouge* de plus grande taille. Autrement dit de minimiser le nombre de sommets du graphe obtenu après la séquence.

**Question 4.** Décrire précisément deux algorithmes polynomiaux pour le problème rouge-bleu (version maximisation) pour les classe d'instances décrite précédemment et évaluer leurs complexités Vous donnerez ensuite quelques éléments qui permettront de comprendre pourquoi vos algorithmes devraient être efficaces.
Les alternants proposeront un seul algorithme.

**Question 5.** Programmer la classe d'instances précédente de la manière suivante. Nous fixons le nombre de sommets à 100. Programmer une fonction qui, étant donné deux réels $p, q \in (0, 1)$, construit un graphe $G$

- $p$ la probabilité pour un sommet d'être rouge,
- $q$ la probabilité pour un arc d'être rouge
- $r$ la probabilité pour un arc d'être "vers la gauche".

**Question 6.** Programmer les algorithmes de la Question 4.

**Question 7.** Le but de cette question est dévaluer l'efficacité de vos deux algorithmes décrits dans la Question 4 en utilisant la fonction qui génère des instances décrites dans la Question 5 en prenant $r = 0.5$. Vous remplirez deux tableaux (un par algorithme) de la manière suivante. Chaque case $f(p, q)$ sera la longueur moyenne des *séquences rouges* obtenues par vos algorithmes sur 100 graphes générés aléatoirement. Est-ce qu'un des deux algorithmes est significativement plus efficace pour certaines valeurs de $p$ et de $q$ ?

| p q | 0 | 0.1 | 0.2 | 0.3 | 0.4 | 0.5 | 0.6 | 0.7 | 0.8 | 0.9 | 1 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | f(0,0) | f(0,0.1) | f(0,0.2) | f(0,0.3) | f(0,0.4) | f(0,0.5) | f(0,0.6) | f(0,0.7) | f(0,0.8) | f(0,0.9) | f(0,1) |
| 0.1 | f(0.1,0) | ... | ... | ... | ... | ... | ... | ... | ... | | |
| 0.2 | f(0.2,0) | ... | | | | | | | | | |
| 0.3 | f(0.3,0) | ... | | | | | | | | | |
| 0.4 | f(0.4,0) | ... | | | | | | | | | |
| 0.5 | f(0.5,0) | ... | | | | f(p,q) | | | | | |
| 0.6 | f(0.6,0) | ... | | | | | | | | | |
| 0.7 | f(0.7,0) | ... | | | | | | | | | |
| 0.8 | f(0.8,0) | ... | | | | | | | | | |
| 0.9 | f(0.9,0) | ... | | | | | | | | | |
| 1 | f(1,0) | ... | | | | | | | | | |

**Question 8 (bonus).** Prouver un algorithme polynomial exact pour la classe d'instances décrite précédemment.