TP

Le jeu de Snort

Le jeu de Snort (du nom de son inventeur Simon Norton) se joue sur un graphe non orienté G : initialement, tous les nœuds de ce graphe sont incolores ; à tour de rôle, deux joueurs colorient l'un en bleu et l'autre en rouge un sommet de leur choix, avec la contrainte qu'un sommet bleu ne peut jamais être voisin d'un sommet rouge. Le premier des deux joueurs qui ne peut plus colorer de sommet est le perdant.

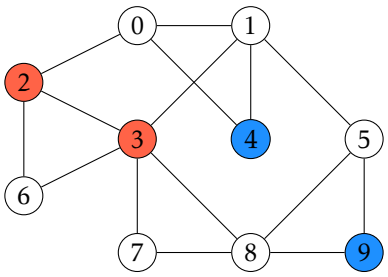


FIGURE 1 – Si c'est au joueur rouge de jouer, ce dernier peut colorer le sommet 6 ou le sommet 7. Si c'est au tour du joueur bleu, seul le sommet 5 est coloriable.

Génération pseudo-aléatoire d'un graphe

On définit une suite (u_k) par la donnée de la condition initiale $u_0 = 42$ et la relation de récurrence

$$\forall k \in \mathbb{N}, \quad u_{k+1} = 19\,999\,999 \times u_k \bmod 19\,999\,981$$

Cette suite est utilisée pour générer des graphes pseudo-aléatoires. Pour que la création de ces derniers ne soit pas trop longue, il est indispensable de stocker dans une liste les différentes valeurs de u_n pour $0 \leqslant n < 1\,000$.

Par la suite, étant donné deux entiers n et k, $G_{n,k}$ désignera le graphe non orienté dont les sommets sont les entiers $0, 1, 2, \ldots, n-1$ et tel que les sommets p et q (avec $p \neq q$) sont voisins si et seulement si $u_{p+q} \bmod (10000 + n) < k$.

Question 1. Définir une fonction `genereGraphe(n, k)` qui renvoie le graphe $G_{n,k}$ représenté par la liste de ses listes d'adjacences.

Calculer le nombre d'arêtes du graphe $G_{n,k}$ pour les valeurs suivantes :

a) $(n,k) = (100, 1\,000)$ b) $(n,k) = (100, 5000)$

Quelques fonctions utilitaires

Pour représenter la coloration partielle d'un graphe dont les sommets sont étiquetées de 0 à $n-1$, on utilise une liste `couleur` de taille n avec la convention :

$$\text{pour tout sommet } s,\ \texttt{couleur[s]} = \begin{cases} 0 & \text{si le sommet } s \text{ est incolore ;} \\ 1 & \text{si le sommet } s \text{ est de couleur rouge ;} \\ 2 & \text{si le sommet } s \text{ est de couleur bleue.} \end{cases}$$

Question 2. Rédiger une fonction `estValide(G, couleur)` qui renvoie `True` lorsqu'il n'existe pas de sommets voisins de couleurs différentes, et `False` sinon.

Pour les valeurs de n suivantes, indiquer la valeur minimale de $k \in \mathbb{N}^*$ pour laquelle la coloration $[0, 1, 2, 0, 1, 2, 0, 1, 2, 0, \ldots]$ n'est pas une coloration valide du graphe $G_{n,k}$:

a) $n = 50$ b) $n = 100$

Question 3. Rédiger une fonction `coupsPossiblesRouge(G, couleur)` qui renvoie la liste des sommets incolores qu'il est possible de colorer en rouge en respectant les contraintes du jeu. On supposera que `couleur` est une coloration valide du graphe.
Rédiger de même la fonction `coupsPossiblesBleu(G, couleur)`.

Pour les valeurs de n et de k suivantes, donner le nombre de coups possibles pour le joueur bleu pour la coloration $[0,1,2,0,1,2,0,1,2,0,\dots]$:

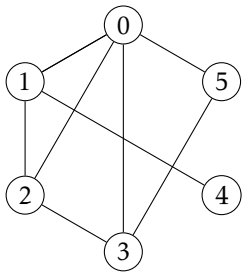
a) $(n,k) = (50,150)$ b) $(n,k) = (100,25)$

Question 4. La liste `couleur` $= [c_0, c_1, \dots, c_{n-1}]$ peut être interprétée comme la décomposition en base 3 de $m = \sum_{i=0}^{n-1} c_i 3^i$.

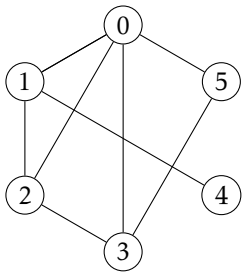
a. Rédiger une fonction `encode(couleur)` qui calcule cet entier m.

b. Rédiger une fonction `decode(m)` qui, partant de l'entier m, renvoie la liste `couleur` qui lui est associée. On rappelle que si $m = \sum_{i=0}^{n-1} c_i 3^i$ alors $m \bmod 3 = c_0$ et $\left\lfloor \frac{m}{3} \right\rfloor = \sum_{i=1}^{n-1} c_i 3^{i-1}$.

c. Le graphe représenté ci-dessous correspond au couple $(n,k) = (6,5000)$. Colorer les sommets associés aux codages 624 et 495.



$n = 624$



$n = 163$

Dans les deux cas, c'est au joueur rouge de jouer. S'agit-il d'une position gagnante ou perdante pour ce joueur ?

Simulation d'une partie

Dans cette partie, chaque joueur adopte la stratégie suivante : lorsqu'il doit jouer, le joueur calcule l'ensemble des coups qui lui sont possibles et s'il en existe, choisit le sommet le plus petit. Par convention, le joueur rouge commence.

Question 5. Rédiger une fonction `partie(G)` qui prend en argument un graphe et renvoie la liste des coups joués ainsi que le gagnant, lorsque les deux joueurs adoptent la stratégie explicitée ci-dessus.
Donner la liste des coups joués et le vainqueur pour les valeurs de n et de k suivantes :

a) $(n,k) = (10,3\,000)$ b) $(n,k) = (100,6\,000)$

Recherche des positions gagnantes

D'après le cours nous savons que les colorations valides d'un graphe peuvent être séparées en deux groupes :

- les *positions perdantes* (le noyau), qui sont caractérisées par le fait que tous les coups possibles sont des positions gagnantes (ce qui inclus le cas où aucun coup n'est possible) ;
- les *positions gagnantes*, qui sont caractérisées par l'existence d'un coup possible menant à une position perdante.

Nous avons vu en cours un algorithme itératif de calcul du noyau ; nous allons cette fois mettre en place un algorithme récursif de manière à obtenir un dictionnaire qui à chaque coloration valide associe le booléen `True` si la position est gagnante pour le joueur rouge, et `False` sinon.
Sachant que les clefs d'un dictionnaire ne peuvent être de type mutable (ce qui exclus les listes), nous utiliserons le codage d'une coloration (défini à la question 4) en guise de clef.

Question 6. On suppose ce dictionnaire créé et nommé `PositionsGagnantes`.
Rédiger deux fonctions mutuellement récursives `rougeGagne(G, couleur)` et `bleuGagne(G, couleur)` qui renvoient le booléen `True` ou `False` suivant que la position `couleur` est gagnante ou pas pour le joueur concerné. On utilisera une mémoïsation des résultats dans le dictionnaire `PositionsGagnantes`.

Remarque. Noter que le dictionnaire est relatif au joueur rouge, et qu'une position gagnante pour le joueur bleu est une position perdante pour le joueur rouge.

On suppose que le joueur rouge commence. Donner le nom du vainqueur lorsque les deux joueurs adoptent une stratégie optimale pour les valeurs de n et de k suivantes :

a) $(n,k) = (8, 5\,000)$ b) $(n,k) = (12, 1\,000)$ c) $(n,k) = (20, 5\,000)$

Les réponses attendues

Question 1.

a) pour $(n,k) = (100, 1\,000)$: **346**

b) pour $(n,k) = (100, 5\,000)$: **2 216**

Question 2.

a) pour $n = 50$: **166**

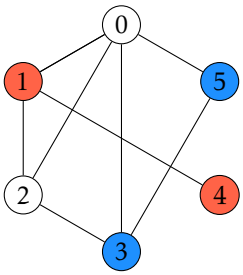
b) pour $n = 100$: **46**

Question 3.

a) pour $(n,k) = (50, 150)$: **14**

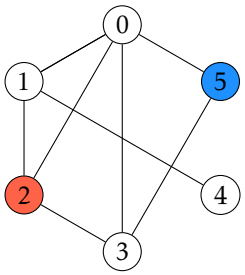
b) pour $(n,k) = (100, 25)$: **18**

Question 4.



$n = 624$

bleu gagne



$n = 495$

rouge gagne (en jouant 1 ou 4)

Question 5.

a) pour $(n,k) = (10, 3\,000)$: **bleu**

$[0, 2, 1, 3]$

b) pour $(n,k) = (100, 6\,000)$: **rouge**

$[0, 8, 6, 15, 19, 55, 26, 70, 66]$

Question 6.

a) pour $(n,k) = (8, 5\,000)$: **bleu**

b) pour $(n,k) = (12, 1\,000)$: **rouge**

c) pour $(n,k) = (20, 5\,000)$: **rouge**