TP4

**Graphes trois-colorables pour des preuves de connaissances**

**Description :**

Le but de ce TP est de vous faire implémenter une accréditation anonyme par une preuve de connaissance à divulgation nulle de type 3-coloriage de graphe.

Plus précisément, on est dans la situation où un utilisateur souhaite convaincre un vérificateur qu'il connaît une manière de colorier entièrement un graphe avec 3 couleurs de telle façon qu'aucun nœud du graphe n'ait la même couleur que l'un des nœuds voisins mais sans révéler aucune autre information à propos de ce coloriage.

Le TP lui-même se décompose en 3 parties.

**Matériel à rendre :**

Il vous est aussi demandé de réaliser un compte rendu contenant un programme qui sert à tester les différentes fonctions que vous aurez écrit durant ce TP. Vous devez remettre ce programme ainsi que vous avez réalisées (avec un minimum de commentaire dans le code).

**Partie 1 : génération du graphe et du coloriage**

On suppose que le graphe qu'on souhaite colorier se compose d’un nombre dynamique des nœuds à laquelle on peut ajouter/supprimer des sommets et des arrêtes, et on souhaite le représenter sous forme d’une matrice d’adjacences.

Pour cette première partie, vous devrez générer un graphe aléatoire qui est 3-coloriable (vert =1, bleu=2 et rouge=3).

Votre fonction s'intitulant *genererGraphe3Coloriable* doit produire en sortie la matrice d'adjacence du graphe ainsi que son coloriage sous forme d’un tableau TAB1.

**Partie 2 : mise en gage des couleurs**

On suppose que la structure du graphe est connue et publique mais que le coloriage constitue le secret. Lorsqu'un utilisateur souhaite prouver à un vérificateur distant qu'il connaît ce secret, c'est à dire une manière de 3-colorer le graphe, il va mettre en gage les différentes couleurs des nœuds puis les envoyer au vérificateur. Comme étape initiale avant la mise en gage, l'utilisateur commence par faire une permutation aléatoire sur les couleurs. Par exemple en transformant "rouge" en "vert", "bleu" en "rouge" et "vert" en "bleu".

Pour mettre en gage une couleur l'utilisateur tire une valeur aléatoire **a** (1<**a** <10) qu'il concatène à la couleur avant de passer le tout à travers une fonction de hachage.

Soit ci la couleur du ième nœud et ri le nombre aléatoire **a** choisit aléatoirement pour cette couleur et yi = h(ri || ci) la valeur mise en gage pour h une fonction de hachage (vous utiliserez l'opération modulo ‘’ mod le nombre des nœuds’’ pour obtenir la valeur de hachage à partir de l'entier fourni ). Pour la mise en gage, l'utilisateur envoie les différents yi (pour i allant de 1 au nombre des nœuds) dans un tableau TAB2.

Pour cela, il vous est demandé de réaliser une fonction qui s'appelle *miseEnGageColoriage* et qui prend deux arguments en entrée

1. TAB1 (contenant les couleurs d’une taille correspondante au nombre des nœuds).
2. un tableau contenant des valeurs aléatoires (1<**a**<10). Cette fonction doit produire en sortie un tableau correspondants au nombre des nœuds contenant les valeurs mises en gage.

**Partie 3 : preuve de connaissance à divulgation nulle d'un 3-coloriage**

Soit le protocole suivant de preuve de connaissance à divulgation nulle pour convaincre un vérificateur de la connaissance d'un 3-coloriage d'un graphe particulier:

1. L'utilisateur envoie une mise en gage d'un coloriage à un vérificateur en n’oubliant pas au préalable de faire une permutation aléatoire sur les couleurs des nœuds

(comme précisé dans la partie précédente).

1. Le vérificateur demande à voir les couleurs de deux nœuds i et j qui sont reliés par une arête (telle que défini par la matrice d'adjacence).
2. L'utilisateur envoie les (ri,ci) et (rj,cj) qui correspondent à ces deux nœuds.

Le vérificateur calcule ensuite si h(ri || ci) = yi et h(rj || cj) = yj et vérifie aussi que les deux couleurs soient différentes (soit ci ≠cj). Si ces trois conditions sont réunies le vérificateur accepte et sinon il refuse.

Ecrivez une fonction *preuveColoriage* qui prend en entrée la matrice d'adjacence d'un graphe, une mise en gage d'un coloriage et l'information nécessaire pour ouvrir les mises en gages de nœuds *i* et *j* et retourne vraie si l'étape 4 dans le protocole ci-dessus est réussie et faux sinon.

Cette procédure est ensuite répétée plusieurs fois (10 fois) et l'utilisateur est considéré comme authentifié seulement s'il passe tous les tests.

Il vous est aussi demandé d'écrire un programme (fonction main) qui implémente un scénario servant à tester les différentes fonctions que vous aurez écrit durant ce tp.