Informatique I

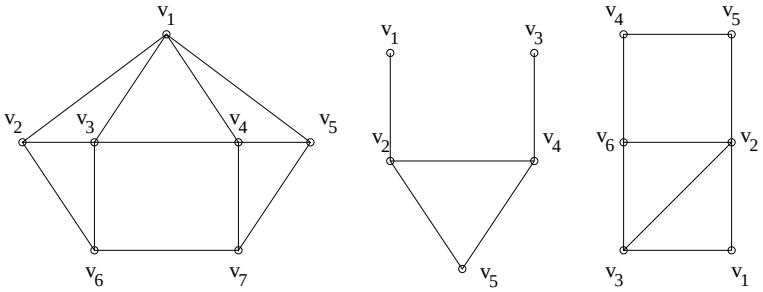
Le problème porte sur le problème de coloriage de graphes, et une application à la sécurité informatique.

Notations et définitions

La notation $\hat{=}$ désigne l'*égalité par définition*, pour la distinguer de l'égalité $=$.

Dans tout le problème, un *graphe* désigne un graphe non orienté, c'est-à-dire un couple $G\hat{=}(V, E)$, où V est un ensemble fini de *sommets*, et E est un ensemble de paires $\{v_1, v_2\}$ de sommets appelées *arêtes*. Si $\{v_1, v_2\} \in E$, on dit que v_1 et v_2 sont *adjacents*, et qu'ils sont les *extrémités* de l'arête $\{v_1, v_2\}$. Le *degré* $d_G(v)$ d'un sommet v de G est le nombre de sommets adjacents à v dans G. Le *degré* $d(G)$ du graphe G est $\max_{v \in V} d_G(v)$.

On notera souvent les graphes par des dessins, par exemple :



[1] [2] [3]

Pour tout entier $k \in \mathbb{N}$, un *k-coloriage* de G est une application c de V vers $[1, k]$ telle que $c(v_1) \neq c(v_2)$ pour tous sommets adjacents v_1, v_2. De façon imagée, on appelle $c(v)$ la *couleur* du sommet v de G dans c. Si G a un k-coloriage, on dit que G est *k-colorable*. Le plus petit entier k tel que G est k-colorable est le *nombre chromatique* de G.

Pour tout ensemble fini S, $|S|$ désigne le cardinal de S.

1 Algorithmes de coloriage

1. Montrer que le nombre chromatique du graphe [1] est exactement 4.

2. On considère l'algorithme glouton suivant, où les sommets de $G \hat{=} (V, E)$ sont supposés numérotés de 1 à $|V|$, et $V[i]$ est la liste des sommets adjacents au sommet i, et où c est un tableau de $|V|$ entrées :

```
1 fonction COL (V, E) {
2         couleur := 0; coloriés := 0;
3         pour i = 1..|V| { c[i] := 0; }
4         tant que coloriés < |V| {
5               couleur := couleur+1;
6               pour i = 1..|V| {
7                     si c[i] = 0 et pour tout j ∈ V[i] c[j] ≠ couleur
8                        alors { c[i] :=couleur; coloriés := coloriés +1; }
9                     }
10              }
11        retourner (couleur, c);
12        }
```

   L'idée est que pour tout graphe (V, E), COL (V, E) retourne un couple (k, c) tel que c est un k-coloriage de (V, E). Pour chacun des graphes (V, E) dessinés plus haut, [1], [2] et [3], calculer COL (V, E). On supposera que les sommets sont listés dans l'ordre v_1, v_2, ..., comme indiqué sur la figure.

3. Montrer que COL (G) termine en temps polynomial en la taille $|V| + |E|$ de $G \hat{=} (V, E)$, et retourne effectivement une paire (k, c) où c est un k-coloriage de G.

4. En s'aidant de la question 2, montrer que COL n'est pas optimal, autrement dit il existe un graphe G, un entier k et un k-coloriage c de G tels que COL $(G) = (k, c)$ et k est strictement supérieur au nombre chromatique de G.

5. Montrer cependant qu'il existe toujours une permutation π de $\{1, \ldots, |V|\}$ tel que COL retourne un résultat optimal sur le graphe permuté $\pi(G)$. (Formellement, $\pi(G)$ est le graphe dont les sommets sont ceux de G, et dont les arêtes sont les $\{\pi(v), \pi(v')\}$ lorsque $\{v, v'\}$ parcourt les arêtes de G.)

6. Écrire un algorithme énumérant toutes les permutations sur $\{1, \ldots, |V|\}$ sans répétition. À titre d'indication, on montrera que toute permutation π sur $\{1, \ldots, N\}$ s'écrit de façon unique comme la composée $(N, i_N) \circ (N-1, i_{N-1}) \circ \ldots \circ (2, i_2) \circ (1, i_1)$, où $1 \leq i_1 \leq N$, $2 \leq i_2 \leq N$, ..., $N \leq i_n \leq N$, et où la notation (i, j) dénote la transposition qui échange i et j si $i \neq j$, et la fonction identité sinon.

7. En déduire un algorithme en temps exponentiel COLORIAGE qui prend en entrée un graphe G, et retourne son nombre chromatique k et un k-coloriage de G.

8. Montrer que le nombre chromatique d'un graphe G quelconque est $\leq d(G)+1$. On exhibera un algorithme COLDEG de k-coloriage en temps polynomial de tout graphe tel que $d(G)+1 \leq k$. (Indication : combien de couleurs sont utilisées au maximum par les sommets adjacents à un sommet fixé v ?)

9. Montrer que le fait qu'un graphe G soit k-colorable ou non reste inchangé si on supprime de G un sommet v quelconque de degré $\leq k-1$, autrement dit G est k-colorable si et seulement si $G-\{v\}$ est k-colorable. On montrera d'autre part comment obtenir un k-coloriage de G à partir d'un k-coloriage quelconque de $G-\{v\}$.

10. En déduire qu'il existe un algorithme en temps polynomial qui prend en entrée un graphe G, et retourne un sous-graphe G' dont tous les sommets sont de degré $\geq k$, qui est k-colorable si et seulement si G l'est, et tel que l'on peut reconstruire en temps polynomial un k-coloriage de G à partir de tout k-coloriage de G'.

2 Complexité du coloriage

On rappelle que le problème 3-SAT de la satisfiabilité d'ensembles de 3-clauses est NP-complet. Un *littéral* L est soit une variable propositionnelle x soit une négation $\overline{x}$ d'une variable propositionnelle x. Une 3-*clause* est une disjonction de trois littéraux, non nécessairement distincts. Un *ensemble de 3-clauses* S est vu comme une conjonction de ses 3-clauses. Une *valuation* ρ est une application qui à chaque variable x associe un booléen, VRAI ou FAUX; ρ *satisfait* une 3-clause si elle contient x tel que $\rho(x)=$VRAI ou $\overline{x}$ tel que $\rho(x)=$FAUX; ρ *satisfait* un ensemble S si ρ satisfait toute 3-clause de S. S est *satisfiable* si et seulement si S est satisfait par au moins une valuation ρ.

1. Soit G un graphe 2-colorable. Soit n la couleur du sommet v. Quelles sont les couleurs des sommets adjacents à v ?

2. Montrer que G est k-colorable si et seulement si toutes ses composantes connexes sont k-colorables. On rappelle qu'une composante connexe de G est un sous-graphe maximal de G dans lequel il existe un chemin entre v et v' pour tous sommets v et v'.

3. Déduire des questions précédentes que le problème de la 2-colorabilité d'un graphe G donné en entrée est décidable en temps polynomial. On donnera un algorithme 2COL qui prend en entrée un graphe G et qui retourne en sortie soit un 2-coloriage de G dans un tableau c, soit la réponse NON-2-COLORABLE si G n'est pas 2-colorable.

4. Montrer que le graphe [2] a la propriété suivante : dans tout 3-coloriage de ce graphe, si v_1 et v_3 sont de couleurs autres que 3, et si v_5 est de couleur 1, alors v_1 ou v_3 est de couleur 1. (Le graphe [2], intuitivement, code donc une disjonction.)

5. Trouver un graphe ayant au moins deux sommets v_1 et v_2, tel que dans tout 3-coloriage, si v_1 et v_2 sont de couleurs différentes de 3, alors v_2 est de couleur 1 si et seulement si v_1 n'est pas de couleur 1. (Ceci code une négation.)

6. Si un graphe a deux sommets A de couleur 3 et B de couleur 2, dans un 3-coloriage donné, quelles sont les couleurs des sommets adjacents à la fois à A et à B ? (Si on code les formules logiques par des sommets, et que "être de couleur 1" signifie "être vrai", ceci code une conjonction.)

7. Déduire des trois questions précédentes qu'il existe un algorithme en temps polynomial qui transforme tout ensemble S de 3-clauses en un graphe G qui est 3-colorable si et seulement si S est satisfiable. (Indication : on créera deux sommets distingués A et B dans G, adjacents; par symétrie, on supposera que A est de couleur 3 et B de couleur 2, et on connectera les sous-graphes des questions précédentes à l'un ou l'autre de ces sommets ou même aux deux.)

8. Que peut-on conclure de la question précédente ?

3 Mots de passe à connaissance nulle

On construit un système de mots de passe permettant à un utilisateur U de s'authentifier sans jamais divulguer son mot de passe, même à l'ordinateur O sur lequel il se connecte. Soit t_0 une durée quelconque dépassant le durée prévue d'utilisation du système (par exemple, 1000 ans).

On suppose pour ceci que l'on dispose d'un générateur de bits aléatoires, d'un mécanisme de chiffrement f, tel que pour tout suite M d'au moins N_0 bits (le *message en clair*), pour toute suite K d'au moins N_1 bits (la *clé*), $f(M,K)$ est une suite de $\geq N_0$ bits. On suppose que dans ces conditions, il existe une unique suite K' de bits (la *clé inverse*), contenant au moins N_1 bits, telle que $f(f(M,K),K') = M$. La fonction $M \mapsto f(M,K)$ est donc injective. On suppose aussi que l'image inverse M de $M' = f(M,K)$ n'est pas calculable en temps inférieur à t_0 en fonction de M' seul (en particulier, sans connaissance de K'). On supposera finalement que l'on sait produire des paires de clés (K,K') aléatoires, avec K inverse de K'.

On supposera d'autre part qu'il est impossible de trouver un 3-coloriage d'un graphe ayant au moins N_2 arêtes, où N_2 est une constante suffisamment grande, en un temps inférieur à t_0. (Ceci est une simplification, en réalité ce n'est vrai qu'avec probabilité proche de 1.)

Soit $\overline{i}$ la suite de deux bits codant l'entier $i \in \{0,1,2,3\}$ en binaire; réciproquement, $\underline{r}$, où r est une suite de deux bits, est l'entier représenté par r en binaire. On note ϵ la suite vide, et $s \cdot s'$ la concaténation des suites s et s'.

L'utilisateur U s'authentifie en utilisant comme nom d'utilisateur un graphe G aléatoire 3-colorable ayant au moins αN_2 arêtes, où α est une constante > 1, et comme mot de passe un 3-coloriage c de G. Quant à O, il maintient un ensemble fini $\mathcal{G}$ de noms d'utilisateurs autorisés, tous de taille $\geq N_2$. Soit n_0 un entier fixé. Une session d'authentification se déroule comme suit :

a. U fournit à O son nom G (dont les sommets sont $1, \ldots, N$).

b. O pose $n := 0$. Si G n'est pas dans $\mathcal{G}$, alors O répond NON et la session se termine.

c. Si $n > n_0$, alors O répond OUI et la session se termine. Sinon :

d. U tire une permutation aléatoire π des couleurs $\{1,2,3\}$, et envoie à O les suites de bits $f(\overline{\pi(c(i))} \cdot R_i, K_i)$, pour chaque sommet $i \in \{1,\ldots,N\}$, où R_i est une chaîne de bits aléatoires de longueur $\geq N_0 - 2$, et (K_i, K_i') est une paire de clés aléatoires de tailles $\geq N_1$, K_i' étant l'inverse de K_i.

e. Maintenant, O possède une description de G, et pour tout sommet i de G, une suite de bits M_i. O tire aléatoirement deux sommets i et j, $1 \leq i,j \leq N$, $i \neq j$, qui sont adjacents dans G, et fournit i et j à U.

f. U fournit les clés inverses K_i' et K_j' à O.

g. O calcule $\underline{r_i}$ et $\underline{r_j}$, où r_i est la suite formée des deux premiers bits de $f(M_i, K_i')$ et r_j de même pour $f(M_j, K_j')$. Si $\underline{r_i} \neq \underline{r_j}$ et $\underline{r_i} \neq 0$, $\underline{r_j} \neq 0$, alors O incrémente n et retourne à l'étape c. Sinon, O répond NON et la session se termine.

1. Montrer que si U possède effectivement un 3-coloriage c de G et l'utilise effectivement comme il est spécifié, alors O retourne OUI, mais que si U est un *intrus* et fournit un graphe G en-dehors de $\mathcal{G}$ ou n'en connaît pas de 3-coloriage, alors O retourne NON avec une probabilité tendant rapidement vers 1 lorsque n_0 tend vers $+\infty$, que l'on déterminera. (On pourra supposer que n_0 est négligeable devant la taille de G.)

2. Montrer que le système ne permet à O d'obtenir aucune information sur le 3-coloriage c de G que U possède. Plus précisément, supposons que O est un *intrus* qui implémente un algorithme arbitraire plutôt que celui décrit plus haut, et retournant aussi soit OUI soit NON après un nombre fini de tours n_0. Considérons deux utilisateurs U et U', possédant le même nom G, et deux 3-coloriages différents c et c'.

   Montrer que O ne peut pas faire la différence entre U et U', au sens où la distribution des réponses de O en présence de l'un ou l'autre est la même, avec probabilité très proche de 1; on déterminera quels paramètres doivent tendre vers l'infini pour que cette probabilité tende vers 1.