Sujet 1

Question de cours.

Question 1. Qu'est ce qu'un problème décidable ? Donner un exemple de problème indécidable.

Graphes orientés

Un graphe orienté est formé d'un ensemble S de sommets et d'un ensemble d'arcs $A \subset S \times S$. Chaque arc (x, y) est orienté de son origine x vers son extrémité.

Question 2. Qu'est ce qu'un algorithme de parcours en profondeur d'un graphe ? En écrire explicitement une version (on demande une description en pseudo-code du type "pour chaque arc entrant de x faire...").

Graphes orientés acycliques

Un circuit de $G = (S, A)$ est une suite $(x_1, \ldots, x_p)$ de sommets tels que $(x_i, x_{i+1}) \in A$ pour tout i et $(x_p, x_1) \in A$. Un graphe orienté est acyclique s'il ne contient pas de circuit.

Question 3. Donner un algorithme basé sur le parcours en profondeur qui construit une liste des sommets dans laquelle le sommet x apparait avant le sommet y si (x, y) est un arc de G.

Question 4. Donner un autre algorithme basé sur la remarque qu'un sommet sans arête entrante peut toujours être mis au début de la liste.

Question 5. Donner un algorithme qui calcule le plus long chemin orienté dans G.

Coloration

On veut colorier un graphe de façon que 2 sommets adjacents n'aient jamais la même couleur. Un graphe est k-coloriable si on peut le faire avec k couleurs.

Question 6. Que signifie l'affirmation : Le problème de savoir si un graphe est k-coloriable est NP-complet pour $k \geq 3$.

Question 7. Donner un algorithme pour tester si un graphe est 2-coloriable.

Question 8. Donner une condition nécessaire et suffisante sur la longueur des cycles pour qu'un graphe soit 2-coloriable.

Le degré $d(x)$ d'un sommet x est le nombre de ses voisins. On note $\Delta(G)$ le degré maximal d'une sommet de G.

Question 9. Montrer que le nombre de couleur utilisée par un algorithme glouton (qui traite les sommets l'un après l'autre et leur attribue la plus petite couleur possible) est au plus $\Delta(G) + 1$. Améliorer la borne en $\max_i \min(i, d(v_i) + 1)$, où v_i est le $i$eme sommet traité. Qu'est ce que ceci suggère de faire ?

Sujet 2

Arbre couvrant minimal

Un graphe non-orienté G est la donnée d'un couple (V, E) d'un ensemble fini V (les sommets) et d'un ensemble fini E de paires $\{v_0, v_1\}$ de sommets distincts (les arêtes). Un chemin d'extrémités v_0 et v_n est une suite alternée $v_0 e_1 v_1 e_2 \dots v_{n-1} e_n v_n$ de sommets et d'arêtes telles que $e_i = \{v_{i-1}, v_i\}$ pour tout i, et les arêtes e_i sont deux à deux distinctes. Un cycle est un chemin tel que $v_0 = v_n$. Un graphe est connexe s'il existe un chemin entre toute paire de sommets. Un arbre couvrant de G est un ensemble A d'arêtes tel que le graphe (V, A) soit connexe et sans cycle.

À chaque arête on associe un poids $p(e)$. Le poids d'une arête est la somme des poids de ses éléments : $p(A) = \sum_{v \in A} p(e)$.

On considère un algorithme qui construit progressivement un arbre en partant d'un sommet arbitraire et en ajoutant à chaque étape l'arc accessible (depuis la partie de l'arbre déjà constuite) de poids minimum.

Question 1. Illustrer l'algorithme sur un exemple.

Question 2. Soit X un sous-ensemble de l'ensemble des sommets V de G. Montrer qu'un arbre couvrant minimal contient forcément une arête joignant un sommet de X à un sommet de $V \setminus X$ de poids minimal parmi ces arêtes.

Question 3. Montrer que l'algorithme décrit ci-dessus construit un arbre couvrant de poids minimal.

Question 4. Proposer une implantation de cet algorithme (il ne s'agit pas de donner tous les détails du pseudo-code mais de préciser le type de structures de données utilisées et en particulier la façon de gérer les sommets accessibles, etc.). Quel est la complexité obtenue ?

Arbre de Steiner

On suppose maintenant les poids positifs. On s'intéresse à un sous ensemble S de l'ensemble des sommets et on cherche un arbre de Steiner, c'est-à-dire un arbre de coût minimum qui contienne tous les sommets de S et n'importe quel sous-ensemble des sommets de $V \setminus S$.

Question 5. Comment faut il comprendre l'assertion "la version décision du problème de l'arbre de Steiner est NP-complète" ?

On cherche un algorithme d'approximation. La cloture métrique C du graphe G est le graphe complet pondéré obtenu en ajoutant les arcs manquants à V en leur donnant comme poids le poids du chemin de poids minimal dans G entre leurs extrémités.

Question 6. Décrire un algorithme d'approximation pour le calcul du poids de l'arbre de Steiner qui se base sur un arbre couvrant minimal du sous-graphe C_S de C formé des sommets S et des arcs entre ces sommets.

Question 7. Construire un exemple de graphe pondéré où l'arbre couvrant de coût minimum n'est pas un arbre de Steiner.

Question 8. Montrer que le coût d'un arbre couvrant de poids minimum du graphe C_S est inférieur au double du coût de l'arbre de Steiner.

Sujet 3

Question de cours.

Question 1. Qu'est ce qu'un langage rationnel ? Énoncer une ou deux jolies propriétés de ces langages.

Minimum et maximum d'un tableau d'entier

On suppose qu'on travaille avec un ensemble d'entiers distincts stockés dans un tableau.

Question 2. Donner un algorithme pour trouver le maximum de l'ensemble. Quelle est la complexité en nombre de comparaisons ? Peut on faire mieux ?

Question 3. On veut maintenant chercher le maximum et le minimum. Proposer un algorithme plus efficace que d'appliquer deux fois le précédent.

On veut déterminer la complexité minimale d'un algorithme résolvant le problème précédent. On appelle "une information" le fait de savoir qu'un élément ne peut être le maximum, ou le fait de savoir qu'il ne peut être le minimum.

Question 4. Combien d'informations faut il pour garantir qu'on a trouvé le maximum et le minimum d'un tableau ?

Question 5. En fonction des informations qu'on connait déjà, combien d'informations nouvelles peut donner une comparaison ?

Question 6. L'algorithme que vous avez proposé à la question 2 est il optimal ? (on voudrait bien que oui...)

On s'intéresse maintenant à la recherche des 2 plus grands éléments.

Question 7. Proposer un algorithme meilleur que 2 recherches successives et analyser sa complexité.

Pour finir allons vers le milieu du tableau.

Question 8. Donner un algorithme pour la recherche de l'élément médian (tel que la moitié des éléments du tableau soient plus grands). On pourra éventuellement commencer par un algorithme probabiliste, auquel cas on donnera une estimation de sa complexité moyenne.