TD Complexité de Problèmes- Les classes P et NP

Exercice 1 : La mise en sachets ($BINPACKING$).

Le problème de la mise en sachets est défini par:
Donnée:
n –un nb d'objets
$x_1, \cdots, x_n$, les poids des objets
c –la capacité d'un sac
k –le nombre de sacs avec $k \leq n$
Sortie: oui, si il existe une mise en sachets possibles, i.e.:
$aff : [1..n] \rightarrow [1..k]$ tq $\sum_{i/aff(i)=j} x_i \leq c$, pour tout numéro de sac j, $1 \leq j \leq k$.
Q 1. Montrer que la propriété est NP.

Exercice 2 : Sac à dos 0-1

Soit le problème de décision défini par:
Donnée:
$p_1, \cdots, p_n$ – n entiers positifs.
$v_1, \cdots, v_n$ – n entiers positifs.
c et v –deux entiers
Sortie: Oui, si il existe un remplissage du sac telle que la valeur du chargement soit au moins v, les objets ne pouvant pas être fractionnés.
Q 1. Montrer qu'il est NP.
Q 2. On a vu dans une feuille précédente de TD qu'il existe un algorithme en $0(n * c)$ qui résoud le problème. Pourquoi ne peut-on pas en déduire que le problème est P?

Exercice 3 : La plus courte super-suite commune.

Donnée:
$u_1, ..., u_n$ n mots –l'alphabet est fixé
k un entier, $k < |u_1| + ... + |u_n|$
Sortie: oui, si il existe une supersuite de longueur k des n mots u_i (i.e. un mot u de longueur k tel que les u_i soient sous-mots de u)
Q 1. Montrer que le problème est NP.
Q 2. Dans le cas où $n = 2$, proposer un algorithme polynomial Supersuite(u,v) qui retourne une Supersuite commune à u et v de longueur minimale.
Q 3. Soit l'algorithme suivant:

```
Super:=u_1;
pour i de 2 à n
 Super=SuperSuite(Super,u_i)
```

Cet algorithme produit-il bien une supersuite des u_i? Produit-il toujours la supersuite la plus courte?

Exercice 4 : Le problème de recouvrement (SET_COVERING)

Donnée: un ensemble E de cardinal n
p sous-ensembles de E, $(E_i)_{i=1}^{i=p}$, qui forment un recouvrement de E (i.e. $E = \bigcup_1^p E_i$).
un entier k
Sortie: oui, si il existe un recouvrement de cardinal k ($J \subset [1..p]$ de cardinal k tq $\bigcup_{i \in J} E_i = E$), non, sinon.

Q 1. Soit $E = [1..8], E_1 = \{1,3,5,7\}, E_2 = \{3,5,8\}, E_3 = \{1,2,3\}, E_4 = \{6,7\}, E_5 = \{4,5\}$. Y-a-t-il une solution pour k=4? Pour k=3?
Q 2. Montrer que la propriété est NP. Quelle serait la complexité d'une méthode par recherche exhaustive?

Exercice 5 : Le Sudoku

Soit le Sudoku généralisé: une grille est un carré de côté k^2 pour un certain k, qu'on peut diviser en k^2 carrés de k^2 cases, le but étant de remplir avec les entiers de 1 à k^2 avec les règles suivantes: un entier doit apparaître une et une seule fois dans chaque ligne, dans chaque colonne, dans chaque carré.
Q 1. Soit le problème de décision:
Donnée: un entier k et une grille $k^2 * k^2$ partiellement remplie par des entiers de 1 à k^2.
Sortie: Oui, Ssi la grille peut être complétée en respectant les règles.
Q 2. Que pensez-vous de la complexité de ce problème de décision:
Donnée: un entier k et une grille $k^2 * k^2$ partiellement remplie par des entiers de 1 à k^2.
Sortie: Oui, Ssi il existe **au plus** une complétion correcte de la grille.

Exercice 6 : Propriétés de clôture

Q 1. Montrer que la classe P est close par union, intersection et complémentaire.
Q 2. Montrer que la classe P est close par concaténation.
Q 3. *(un peu plus dur...)* Montrer que la classe P est close par étoile.
Q 4. Montrer que la classe NP est close par union, intersection.
Q 5. Montrer que si NP n'est pas close par complémentaire, $NP \neq P$.
Q 6. Montrer que la classe NP est close par concaténation et étoile.

Exercice 7 : Autour de SAT

Q 1. Quelle est l'erreur dans le raisonnement suivant: Toute formule booléenne peut être mise sous forme disjonctive; or tester la satisfiabilité d'une formule booléenne sous forme disjonctive est polynomial. Donc tester la satisfiabilité d'une formule booléenne est polynomial.
Q 2. Que pensez-vous de la complexité de $2 - CNF - SAT$?

Exercice 8 : Noir et Blanc

On a vu en cours que le 3-coloriage de graphes est NP. Que pensez-vous du 2-coloriage de graphes?

Exercice 9 : Test de Primalité

Soit l'algorithme:

```
//n est un entier naturel >=2
Booléen Composé(Entier n)
Pour i de 2 à racine-carree(n)
 si i divise n alors retourne Vrai
fin Pour;
retourne Faux
```

Q 1. Cet algorithme est-il correct, i.e. retourne-t-il Vrai Ssi n est composé? Est-il polynomial?
Q 2. Montrer que la propriété " être composé" est NP?
Q 3. Le test de primalité d'un entier a été montré P en 2002 après de longues années de recherche. Qu'en déduire pour la propriété "être composé"?

Pour les curieux: Jusque 2002, certains tests utilisés étaient supposés être polynomiaux mais ce n'avait pas été prouvé. Le nouveau test est appelé le test de primalité d'Agarwal-Kayal-Saxena -"AKS primality test"-, sa complexité étant en $O((log\ n)^{12+\epsilon})$. Des améliorations ont été proposées pour faire passer l'exposant de 12 à 6. (http://www.utm.edu/research/primes).

Avant 2002, on savait déjà que "être premier" était NP, donc était dans $co - NP \cap NP$. La notion de certificat est un peu plus difficile à trouver que pour "être composé"; elle est basée sur le fait que n est premier Ssi il existe a tel $a^{n-1} \equiv 1(mod\ n)$ mais $a^m \not\equiv 1(mod\ n)$ pour tout m, $1 \leq m < n$.