Université des Sciences et de la Technologie Houari Boumediène
Faculté d'Electronique et d'Informatique
Département d'Informatique
LMD Master 1ère Année IL 2012/2013
Module ''Algorithmique Avancée et Complexité''
Date : 29/01/2013

Corrigé de l'examen de rattrapage

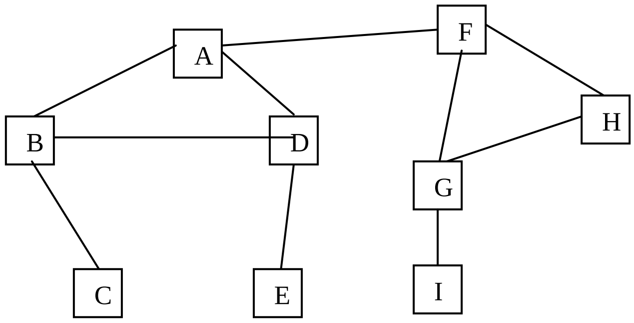
Exercice 1 <6 points=3+3> :
Si f(n) est une fonction en O(n), les affirmations suivantes sont-elles vraies ?
1. $(f(n))^2$=O(n^2)
2. $2^{f(n)} = O(2^n)$

Solution :
Soit donc une fonction f vérifiant f(n)=O(n).
1. f(n)=O(n) : il existe c>0 et $n_0 \geq 0$ tels que $\forall n \geq n_0$ f(n)≤c*n. Donc $\forall n \geq n_0$ $(f(n))^2 \leq c^2 * n^2$. Il existe donc $c_1=c^2$ et $n_1=n_0$ tels que $\forall n \geq n_1$ $(f(n))^2 \leq c_1 * n^2$. En conclusion, f(n)=O(n) implique bien $(f(n))^2$=O(n^2).
2. f(n)=O(n) n'implique pas (toujours) $2^{f(n)} = O(2^n)$. En voici un contre-exemple : f(n)=$\log_2 n$+n. On a bien f(n)=0(n). Par contre, $2^{f(n)}$=$n2^n \neq O(2^n)$. Pour le montrer, on procède par l'absurde. Supposons que $n2^n=O(2^n)$. Il existerait c>0 et $n_0 \geq 0$ tels que $\forall n \geq n_0$ $n2^n \leq c*2^n$. Ce qui donnerait $\forall n \geq n_0$ n≤c. La constante c recherchée n'existe pas : donc on a bien $n2^n \neq O(2^n)$.

Exercice 2 <5 points=2+2+1> :
Il vous est demandé d'écrire un algorithme polynomial d'addition de deux matrices carrées d'entiers, en procédant comme suit :
1) Donnez l'algorithme.
2) Calculez le nombre d'opérations élémentaires du pire cas de l'algorithme.
3) Montrez que l'algorithme est polynômial.

Solution :
Soient M et N deux matrices carrées n*n d'entiers, et P l'addition de M et N (matrice n*n).
1)
```
pour i=1 à n faire
        pour j=1 à n faire
                P[i,j]=M[i,j]+N[i,j]
        fait
fait
```
2) Le nombre d'opérations du pire cas de l'algorithme est clairement f(n)=$2n^2$ (n^2 fois deux opérations élémentaires consistant en une addition et une affectation).
3) Clairement f(n)=O(n^2) : il existe c>0 et $n_0 \geq 0$ tels que $\forall n \geq n_0$ f(n)≤c*n^2 (prendre c=2 et n_0=1). L'algorithme est donc bien polynômial.

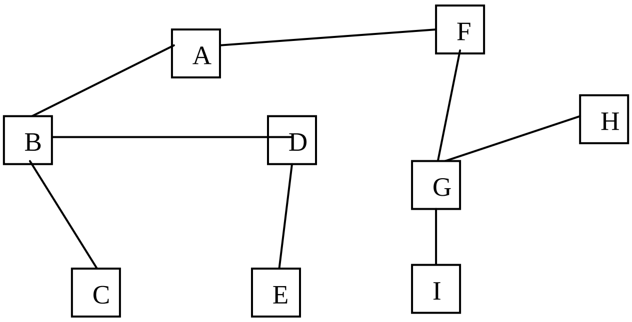
Exercice 3 <3 points=1,5+1,5> :
On considère le graphe non orienté suivant :



1. Donnez un arbre de recouvrement issu d'un parcours en profondeur d'abord du graphe
2. Donnez l'ordre dans lequel le parcours considéré a visité les sommets du graphe

Solution :

1)



2) ABCDEFGIH

Exercice 4 <6 points=1,5+1,5+2+1> :

On considère le problème de décision P suivant :

- **Description :** Un ensemble fini A et un ensemble B de triplets (a,b,c) d'éléments distincts de A.
- **Question :** Existe-t-il une bijection $f : A \rightarrow \{1,\ldots,|A|\}$ telle que pour tout triplet (a,b,c) de B, on ait $f(a)<f(b)<f(c)$ ou $f(c)<f(b)<f(a)$?

Le but de l'exercice est de trouver un algorithme polynômial de validation pour le problème P ci-dessus. Pour ce faire, il vous est demandé de procéder comme suit :

1. Donnez une structure de données permettant de représenter une instance du problème P. Expliquez
2. En quoi consiste un certificat d'une instance du problème P. Donnez une structure de données permettant la représentation d'un tel certificat. Expliquez
3. Donnez un algorithme de validation pour le problème P, que vous appellerez validation_P. L'algorithme, bien évidemment, doit être polynômial, la preuve de la polynômialité faisant l'objet de la question 4. Ecrivez l'algorithme sous forme d'une fonction booléenne dont il est important que vous expliquiez les paramètres.
4. Justifiez la polynômialité de l'algorithme validation_P.

Solution :

1. Une instance du problème P est constituée d'un ensemble fini A de n objets et d'un ensemble B de triplets d'éléments distincts de A. L'instance sera représentée par une structure donnée I=(n,A,B,m), n étant un entier positif, A un tableau de taille n d'objets, et B une matrice à m lignes et 3 colonnes dont les lignes sont toutes différentes et vérifient (B[i,1],B[i,2],B[i,3]) triplet d'éléments distincts de l'ensemble $\{1,\ldots,n\}$
2. Un certificat est une bijection $f : A \rightarrow \{1,\ldots,|A|\}$ qui est représentée par une structure de données consistant en un tableau c de permutation de taille n (les éléments de c sont tous distincts et appartiennent tous à $\{1,\ldots,n\}$)
3. booléen validation_P(n,A,B,m)

```
début
    pour i=1 à m faire
        si (c[B[i,1]]≥c[B[i,2]] ou c[B[i,2]]≥c[B[i,3]]) et
           (c[B[i,3]]≥c[B[i,2]] ou c[B[i,2]]≥c[B[i,1]]) alors retourner FAUX finsi
    fait
    retourner VRAI
fin
```

4. Le nombre d'opérations élémentaires de l'algorithme dans le pire des cas est clairement un polynôme de degré 1 en m. L'algorithme de validement est donc polynômial (linéaire).