IPS 14 décembre 2015
Examen terminal Durée: 3h

HMIN320
Examen 11 janvier 2015

Les documents sont autorisés. La note prendra en compte la clarté des explications. Il est conseillé d'effectuer les exercices dans l'ordre.

Exercice 1

On dispose d'une pile de n factures de téléphone et d'une pile de p chèques destinés à payer les factures et qui portent chacun le numéro de téléphone correspondant. On veut savoir quelles factures n'ont pas été payées. On envisage trois algorithmes :

1. On prend les chèques dans l'ordre où ils se présentent et, pour chacun d'eux, on cherche la facture qui lui correspond ; on enlève chèque et facture des listes donnés.
2. On trie les factures par ordre de numéros de téléphone croissants. Pour chaque chèque, on fait une recherche dichotomique de la facture correspondante.
3. On trie séparément factures et chèques, puis, pour les i de 1 à p, on compare le $i$ème chèque avec les factures restantes. Quand on a trouvé la facture, on l'ôte ainsi que toutes celles de numéro inférieur.

Donner la complexité au pire des cas de chaque algorithme en nombre de comparaisons. Comparer ces complexités en envisageant les grandeurs relatives de n et p.

□

Exercice 2

On considère le graphe orienté valué donné par la figure 3. Déterminer une arborescence des plus courts chemins depuis le sommet s en utilisant l'algorithme de Dijkstra. On donnera à chaque étape l'arborescence obtenue.

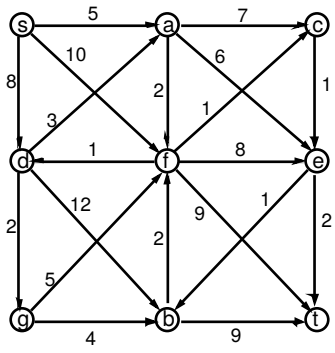

FIGURE 1 – *Graphe orienté*.

□

Exercice 3

On représente un graphe orienté par sa matrice d'adjacence $n \times n$. Vous donnerez la complexité des algorithms suivants :

1. Écrire un algorithme pour déterminer si le graphe est sans boucle.
2. Écrire un algorithme pour faire afficher tous les successeurs d'un sommet donné.
3. Écrire un algorithme pour faire afficher tous les prédécesseurs d'un sommet donné.

□

Exercice 4

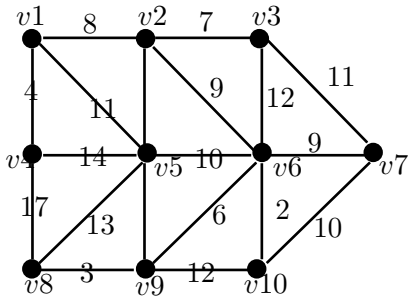

FIGURE 2 – *Graphe G non orienté.*

Donner un arbre couvrant de poids minimal pour le graphe G de la figure 1, en utilisant les algorithmes de Prim et de Kruskal. On indiquera quelle arête entre à chaque étape dans l'arbre (ou la forêt) couvrant(e). □

Exercice 5

Calculer en fonction de n la complexité en nombre d'additions des fonctions suivantes :

```
fragment_1
    for i<- 0 to n do
          sum <-sum+1
  return sum
```

```
fragment_2
   for i<-0 to n do
    for j<-0 to n do
       sum<-sum+1
  return sum
```

```
fragment_3
    for i<- 0 to n do
          sum <-sum+1
     for j<-0 to n do
          sum <-sum+1
 return sum
```

```
fragment_4
  for i<-0 to n do
   for j<-0 to n^2 do
     sum<-sum+1
 return sum
```

```
fragment_5
```

```
fragment_6
```

```
   for i<-0 to n do
       for j<-0 to i do
           sum<-sum+1
return sum
```

```
 for i<-0 to n do
   for j<-i to n do
      for k<-0 to n do
         sum<-sum+1
return sum
```

□

Exercice 6

1. Si je prouve que la complexité dans le pire des cas d'un algorithme est en $O(n^2)$, est-il possible qu'il soit en $O(n)$ sur certaines données ?
2. Si je prouve que la complexité dans le pire des cas d'un algorithme est en $O(n^2)$, est-il possible qu'il soit en $O(n)$ sur toutes les données ?
3. Si je prouve que la complexité dans le pire des cas d'un algorithme est en $\theta(n^2)$, est-il possible qu'il soit en $O(n)$ sur certaines données ?
4. Si je prouve que la complexité dans le pire des cas d'un algorithme est en $\theta(n^2)$, est-il possible qu'il soit en $O(n)$ sur toutes les données ?
5. A et B sont deux algorihtmes pour le même problème. La complexité dans le pire des cas de l'algorithme A est en $O(n \log n)$, celle de B est en $O(n^2)$. Peut-on dire que l'algorithme A est préférable à B ? Avez-vous un exemple de telles situations ?
6. Un algorihme de tri prend une seconde sur votre machine pour trier 1000 éléments. Combien de temps prendra-t-il pour trier 10000 si
   - vous supposez que son temps d'exécution est proportionnel à n^2.
   - vous supposerez que son temps d'exécution est proportionnel à $n \log n$.
7. Un algorithme prend une seconde sur votre machine pour traiter une donnée de taille n_0. Vous échangez votre machine contre une machine quatre fois plus rapide. Quelle est la taille maximale d'une donnée pouvant maintenant être traitée en une seconde si :
   - vous supposez que le temps d'exécution est proportionnel à n.
   - vous supposez que le temps d'exécution est proportionnel à n^2.
   - vous supposez que le temps d'exécution est proportionnel à $\log n$.
   - vous supposez que le temps d'exécution est proportionnel à 2^n.

□