DS N° 3 — CORRIGÉ (PARTIE À RÉDIGER)

I Questions de cours

Question 3 [4pts]

II Exercices

II.2) [1pt] La complexité vérifie l'équation $T(n) = T(n-1) + \Theta(1)$ dans le pire des cas (celui où l'élément n'est pas dans la liste) donc la complexité est linéaire par rapport à la taille n de la liste dans le pire des cas.

II.4) [1pt] L'équation de complexité est dans le pire des cas (celui où l'on a aucun doublon)

$$T(n) = \underbrace{\Theta(n-1)}_{\text{appel de }\texttt{appartient}} + \underbrace{T(n-1)}_{\text{appel récursif}}$$

On a ici une complexité quadratique.

II.6) [1pt] `supprimer` a aussi une complexité linéaire (dans tous les cas) donc on retrouve dans le pire des cas (celui où la liste est sans doublon) l'équation de complexité

$$T(n) = \underbrace{\Theta(n-1)}_{\text{appel de }\texttt{elimine}} + \underbrace{T(n-1)}_{\text{appel récursif}}$$

On a encore une complexité quadratique comme à la question II.4).

III Multiplication des polynômes

III.2) [1pt] On choisit naturellement le tableau vide `[||]` comme représentation adaptée du polynôme nul.

III.7) [1pt] On ajoute successivement tous les n termes de la représentation ce qui nécessite n additions.

III.9) [2pts] On veut effectuer en machine le calcul naïf

$$\left(\sum_{i=0}^{n-1} a_i X^i\right) \times \left(\sum_{j=0}^{n-1} b_j X^j\right) = \sum_{i=0}^{n-1}\sum_{j=0}^{n-1} a_i b_j X^{i+j}$$

Il y a donc exactement n^2 produits $a_i b_j$ à effectuer et $(n-1)^2$ additions entre ces produits [1].

III.10) [1pt] $PQ = (X^m P_1 + P_2)(X^m Q_1 + Q_2) = X^n P_1 Q_1 + X^m(P_1 Q_2 + P_2 Q_1) + P_2 Q_2$.

1. Dans l'implémentation proposée on effectue en fait n^2 additions également.

III.11) [2pts] Il s'agit du paradigme *diviser pour régner*. On va chercher à calculer le produit de deux polynômes de degrés au plus $n-1$ en calculant récursivement des produits de polynômes de degrés au plus $n/2-1$ et en combinant ces résultats par la formule précédente.

III.12) [2pts] $M(n) = \underbrace{M(n/2)}_{\text{calcul de } P_1Q_1} + \underbrace{M(n/2)}_{\text{calcul de } P_1Q_2} + \underbrace{M(n/2)}_{\text{calcul de } P_2Q_1} + \underbrace{M(n/2)}_{\text{calcul de } P_2Q_2} + \Theta(n)$. En dehors des appelles récursifs on compte des extractions de sous-tableaux dans des tableaux de taille au plus $2n$, un nombre constant d'additions de polynômes de taille au plus $2n$, ce qui se réalise avec une complexité linéaire en n et justifie le terme en $\Theta(n)$.

III.13) [3pts] On a $\log_2(4) > 1$ et le théorème du cours donne une complexité en $\Theta(n^2)$. Cette méthode *diviser pour régner* n'a pas permis d'amélioration de la complexité.

III.14) [2pts] On a $P_1Q_2 + P_2Q_1 = R_2 - R_1 - R_3$.

III.15) [1pt] On a alors $PQ = X^n R_1 + X^m(R_2 - R_1 - R_3) + R_3$.

III.16) [1pt] La relation de récurrence devient $M(n) = 3M(n/2) + \Theta(n)$.

III.17) [3pts] On a $\log_2(3) \approx 1.58 > 1$ donc le résultat du cours donne $M(n) = \Theta(n^{\log_2(3)})$. Ceci est cette fois-ci une amélioration substantielle de la complexité !

III.19) [3pts] La complexité spatiale vérifie aussi l'équation de récurrence $S(n) = 3S(n/2) + \Theta(n)$, donc $S(n) = \Theta(n^{\log_2(3)})$.

IV Nombre d'inversions dans une liste

IV.1) [1pt] $(1, 3)$; $(2, 3)$; $(4, 5)$ sont les inversions de cette suite finie.

IV.3) [2pts] La complexité est quadratique car elle vérifie l'équation

$$T(n) = \underbrace{T(n-1)}_{\text{appel récursif}} + \underbrace{\Theta(n)}_{\text{appel de } \texttt{nb\_inversions\_tete}}$$

IV.4) [1pt] Récursivement !

IV.5) [2pt] Il faut que la recherche des inversions à cheval ait une complexité en $\Theta(n^\alpha)$ où $\alpha < 2$ si on veut une complexité meilleure que pour la méthode naïve, vu que l'on a deux appels récursifs et $\log_2(2) = 1$.

IV.6) [2pts] Supposons que l'on ait des listes d'éléments distincts ℓ_1 et ℓ_2 triées dans l'ordre croissant, de même taille à une unité près et qu'on ait accès à taille(ℓ_1).

En posant $\ell_1 = a_1 :: s_1$ et $\ell_2 = a_2 :: s_2$, on voit que

- soit $a_1 < a_2$ et dans ce cas a_1 ne peut intervenir dans aucune inversion vu que les listes sont triées, donc le nombre d'inversions à cheval sont celles de (s_1, ℓ_2) ;
- soit $a_1 > a_2$ et dans ce cas tous les éléments de ℓ_1 sont strictement supérieurs à a_2, donc il y a taille(ℓ_1) d'inversions à cheval plus celles de (ℓ_1, s_2).

IV.9 [3pts] Le calcul des inversions se fait en temps linéaire $O(n)$ et on obtient donc, toujours d'après le cours, une complexité en $O(n \log n)$.