

Travaux dirigés 2

Rappels de cours Un algorithme récursif (paradigme "diviser pour régner") a un temps d'exécution

$$T(n) = aT(n/b) + f(n)$$

où n est la taille des entrées, a est le nombre de sous-problèmes, n/b ($=\lfloor n/b \rfloor$ ou $\lceil n/b \rceil$) est la taille des sous-problèmes, et $f(n)$ est le temps requis pour diviser et combiner.

- 1) Si $f(n) = O(n^{\log_b a - \varepsilon})$ pour une constante $\varepsilon > 0$, alors $T(n) = \Theta(n^{\log_b a})$;
- 2) Si $f(n) = \Theta(n^{\log_b a} \log^k n)$ pour une constante $k \in \mathbb{N}$, alors $T(n) = \Theta(n^{\log_b a} \log^{k+1} n)$;
- 3) Si $f(n) = \Omega(n^{\log_b a + \varepsilon})$ pour une constante $\varepsilon > 0$, et si, pour n assez grand, $af(n/b) \leq cf(n)$ pour une constante $c < 1$, alors $T(n) = \Theta(f(n))$.

Exercice 1 Donner $g(n)$ telle que $T(n) = \Theta(g(n))$ où $T(n)$ est définie récursivement par :

- 1) $T(n) = 8T(n/2) + n^3$.
- 2) $T(n) = 16T(n/4) + n$.
- 3) $T(n) = 8T(n/2) + n^2$.
- 4) $T(n) = 17T(n/16) + n \log n$.
- 5) $T(n) = 8T(n/3) + n^2$.
- 6) $T(n) = 9T(n/3) + n^2$.
- 7) $T(n) = 8T(n/2) + n^4 \log n$.
- 8) $T(n) = 2T(n/2) + n$.

Exercice 2 Soit l'algorithme de tri suivant :

Algorithme 1 : TRI-FUSION(A, i, k)

si $i < k$ alors

$j \leftarrow \lfloor \frac{i+k}{2} \rfloor$

TRI-FUSION(A, i, j)

TRI-FUSION($A, j+1, k$)

FUSIONNER (A, i, j, k)

fin

- 1) Appliquer l'algorithme tri-fusion (**Algorithme ??**) au tableau A suivant :

9	8	12	3	5	14	6
---	---	----	---	---	----	---

- 2) Sachant que la fusion de deux tableaux triés dont la somme des longueurs est n s'effectue en $O(n)$, donner la complexité du tri-fusion.
- 3) Démontrer sa validité.

Exercice 3 Somme des éléments d'un tableau

Soit A un tableau de $n \geq 1$ entiers.

- 1) Ecrire en Python un algorithme itératif calculant la somme des éléments de A et démontrer sa validité.
- 2) Déterminer sa complexité.
- 3) Réécrire l'algorithme pour qu'il soit récursif est satisfasse $T(n) = 2T(n/2) + O(1)$.
- 4) A-t-on ainsi amélioré la complexité de l'algorithme ?

Exercice 4 *Fonctions mystères*

Soit la fonction F (dépendant d'un entier n) suivante :

Algorithme 2 : $F(n)$

```
si  $n = 0$  alors  
    retourner 2  
sinon  
    retourner  $F(n - 1) * F(n - 1)$   
fin
```

- 1) Que calcule F ? Le démontrer.
- 2) Donner le nombre $m(n)$ de multiplications effectuées par $F(n)$.
- 3) Déterminer la complexité de F et montrer comment l'améliorer.

Soit la fonction G (dépendant d'un entier n) suivante :

Algorithme 3 : $G(n)$

```

 $R \leftarrow 2$ 
pour  $i = 1$  à  $n$  faire
     $R \leftarrow R * R$ 
fin
retourner  $R$ 

```

- 4) Que calcule G ? Le démontrer.
- 5) Déterminer la complexité de G .

Exercice 5 ★ Multiplication de matrices

- 1) Ecrire en pseudo-code l'algorithme classique de multiplication de deux matrices. Donner la complexité de cette opération.
- 2) L'algorithme de Strassen recherche le produit $C = AB$ de deux matrices carrées A et B de taille 2^k , en divisant les trois matrices A, B et C en matrices par blocs de taille égale :

$$\begin{pmatrix} C_{11} & C_{12} \\ C_{21} & C_{22} \end{pmatrix} = \begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix} \begin{pmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{pmatrix}$$

où X_{ij} est la sous-matrice carrée de X formée des 2^{k-1} premières (resp. dernières) lignes si $i = 1$ (resp. si $i = 2$) et des 2^{k-1} premières (resp. dernières) colonnes si $j = 1$ (resp. si $j = 2$).

La matrice C est alors déterminée en calculant :

$$\begin{aligned} P_1 &= (A_{11} + A_{22})(B_{11} + B_{22}) \\ P_2 &= (A_{21} + A_{22})(B_{11}) \\ P_3 &= (A_{11})(B_{12} - B_{22}) \\ P_4 &= (A_{22})(B_{21} - B_{11}) \\ P_5 &= (A_{11} + A_{12})(B_{22}) \\ P_6 &= (A_{21} - A_{11})(B_{11} + B_{12}) \\ P_7 &= (A_{12} - A_{22})(B_{21} + B_{22}) \end{aligned}$$

et en remarquant que :

$$\begin{aligned} C_{11} &= P_1 + P_4 - P_5 + P_7 \\ C_{12} &= P_3 + P_5 \\ C_{21} &= P_2 + P_4 \\ C_{22} &= P_1 - P_2 + P_3 + P_6 \end{aligned}$$

2) Détaillez les itérations de Strassen pour le calcul de

$$\begin{pmatrix} 1 & 3 \\ 5 & 7 \end{pmatrix} \begin{pmatrix} 8 & 4 \\ 6 & 2 \end{pmatrix}$$

3) Soit $T(n)$ le temps requis pour la multiplication de deux matrices de taille n . Quelle est l'équation de récurrence satisfaite par $T(n)$? En déduire la complexité de l'algorithme de Strassen.