Commencé le	Monday 11 December 2023, 09:02
État	Terminé
Terminé le	Monday 11 December 2023, 09:19
Temps mis	17 min 21 s
Points	0,14/10,00
Note	**0,29** sur 20,00 (**1,43**%)

Description

Théorème général.

Pour une équation de récurrence de la forme $T(n) = aT(n/b \pm \mathrm{O}(1)) + f(n)$ avec $a \geq 1,\ b > 1$:

- si $f(n) = \mathrm{O}(n^{(\log_b a)-\varepsilon})$ pour un $\varepsilon > 0$, alors $T(n) = \Theta(n^{\log_b a})$;
- si $f(n) = \Theta(n^{\log_b a})$, alors $T(n) = \Theta(n^{\log_b a} \log n)$;
- si $f(n) = \Omega(n^{(\log_b a)+\varepsilon})$ pour nu $\varepsilon > 0$, **et** par ailleurs $af(n/b) \leq cf(n)$ pour un $c < 1$ et des grandes valeurs de n , alors $T(n) = \Theta(f(n))$.

Question **1**

Incorrect

Note de 0,00 sur 1,00

Quelles sont les classes de complexités compatibles avec l'équation suivante?

$T(n) = 3T(n/3) + \Theta(1).$

Veuillez choisir au moins une réponse.

- [ ] $T(n) = O(\log n)$
- [ ] $T(n) = O(n)$
- [ ] $T(n) = O(n \log n)$
- [x] $T(n) = \Theta(1)$ ✖
- [ ] $T(n) = \Theta(\log n)$
- [ ] $T(n) = \Theta(n^2)$
- [ ] $T(n) = \Theta(n)$
- [ ] $T(n) = O(n^2)$
- [ ] $T(n) = \Theta(n \log n)$

Votre réponse est incorrecte.

Apply the master theorem with $a = 3$, $b = 3$, $n^{\log_3(3)} = n$. Compare $\Theta(1)$ to $\begin{cases} O(n^{(1-\varepsilon)}) \\ \Theta(n) \\ \Omega(n^{1+\varepsilon}) \end{cases}$. We are in the first case with for instance $\varepsilon = 1$: $\Theta(1) \in O(n^0) = O(1)$.

As a conclusion: $T(n) = \Theta(n^{log_3(3)}) = \Theta(n)$. But $\Theta(n)$ is also included in $O(n)$, $O(n \log n)$ and $O(n^2)$

Les réponses correctes sont :

$T(n) = \Theta(n)$

,

$T(n) = O(n)$

,

$T(n) = O(n \log n)$

,

$T(n) = O(n^2)$

Question **2**

Incorrect

Note de 0,00 sur 1,00

Quelles sont les classes de complexités compatiables avec l'équation suivante?

$V(n) = V(n/2) + n + 2.$

Veuillez choisir au moins une réponse.

- [ ] $V(n) = O(n \log n)$
- [ ] $V(n) = O(\log n)$
- [ ] $V(n) = \Theta(1)$
- [x] $V(n) = \Theta(n \log n)$ ✖
- [ ] $V(n) = \Theta(n^2)$
- [ ] $V(n) = \Theta(\log n)$
- [ ] $V(n) = \Theta(n)$
- [ ] $V(n) = O(n^2)$
- [ ] $V(n) = O(n)$

Votre réponse est incorrecte.

$a = 1, b = 2, n^{\log_2(1)} = n^0 = 1$

We must compare $f(n) = n + 2$ to $\begin{cases} O(n^{0-\varepsilon}) \\ \Theta(1) \\ \Omega(n^{0+\varepsilon}) \end{cases}$

Here, it looks like we should be in the third case of the theorem, with for instance $\varepsilon = 1/2$: then $(n+2) \in \Omega(n^{1/2}) = \Omega(\sqrt{n})$. (Note that $\varepsilon = 1$ would also work, we have many choices.)

Since we appears to fall into the third case, we need to find some $c < 1$ such that $af(n/b) \leq cf(n)$ for large values of n. In other words, we look for $c < 1$ such that $\frac{n}{2} + 2 \leq c(n+2)$ when $n \to \infty$. We can see that any value of c such that $\frac{1}{2} < c < 1$ works. Therefore the third case actually applies.

Conclusion: $V(n) = \Theta(f(n)) = \Theta(n)$. However $\Theta(n)$ is also included in $O(n)$, $O(n \log n)$, and $O(n^2)$

Les réponses correctes sont :

$V(n) = \Theta(n)$

,

$V(n) = O(n)$

,

$V(n) = O(n \log n)$

,

$V(n) = O(n^2)$

Question **3**

Partiellement correct

Note de 0,14 sur 1,00

Quelles sont les classes de complexités compatiables avec l'équation suivante?

$U(n) = 2U(n/3) + \Theta(1).$

À toutes fins utiles: $\log_3(2) \approx 0.63$, $\log_2(3) \approx 1.58$.

Veuillez choisir au moins une réponse.

- [ ] $U(n) = \Theta(n^{\log_2(3)})$
- [x] $U(n) = \Theta(n^{\log_3(2)})$ ✔
- [ ] $U(n) = O(n^{\log_2(3)} \log n)$
- [ ] $U(n) = \Theta(\log n)$
- [ ] $U(n) = O(\log n)$
- [ ] $U(n) = \Theta(n \log n)$
- [ ] $U(n) = \Theta(n^2)$
- [ ] $U(n) = \Theta(n^{\log_3(2)} \log n)$
- [ ] $U(n) = \Theta(1)$
- [ ] $U(n) = \Theta(n^{\log_2(3)} \log n)$
- [ ] $U(n) = O(n^2)$
- [ ] $U(n) = O(n \log n)$
- [ ] $U(n) = O(n^{\log_3(2)})$
- [ ] $U(n) = O(n^{\log_2(3)})$
- [ ] $U(n) = O(n^{\log_3(2)} \log n)$

Votre réponse est partiellement correcte.

Vous en avez sélectionné correctement 1.

$a = 2, b = 3, n^{\log_3(2)} = n^{0,63\ldots}$.

We must compare $\Theta(1)$ to $\begin{cases} O(n^{(\log_3(2)-\varepsilon)}) \\ \Theta(n^{\log_3 2}) \\ \Omega(n^{\log_3(2)+\varepsilon}) \end{cases}$

We are the first case, with for instance $\varepsilon = \log_3(2)$: $\Theta(1) \in O(n^0) = O(1)$.

Conclusion: $U(n) = \Theta(n^{\log_3(2)})$. But $\Theta(n^{\log_3(2)})$\) is also included in $O(\Theta(n^{\log_3(2)})$, $O(n^{\log_3(2)} \log n)$, $O(\Theta(n^{\log_2(3)})$, $O(n^{\log_2(3)} \log n)$, $O(n \log n)$ and $O(\Theta(n^2)$.

Les réponses correctes sont :
$U(n) = \Theta(n^{\log_3(2)})$

,
$U(n) = O(n^{\log_3(2)})$

,
$U(n) = O(n^{\log_3(2)} \log n)$

,
$U(n) = O(n^{\log_2(3)} \log n)$

,

$U(n) = O(n^{\log_2(3)})$

,

$U(n) = O(n \log n)$

,

$U(n) = O(n^2)$

Question **4**

Incorrect

Note de 0,00 sur 1,00

Quelles sont les classes de complexités compatiables avec l'équation suivante?

$X(n) = 2X(n/2) + \Theta(n \log n).$

Veuillez choisir au moins une réponse.

- [ ] $X(n) = O(n(\log n)(\log n))$
- [ ] $X(n) = O(n)$
- [ ] $X(n) = \Theta(n(\log n)(\log n))$
- [ ] $X(n) = O(n(\log n))$
- [ ] $X(n) = O(n^2)$
- [x] $X(n) = \Theta(n \log n)$ ✖
- [ ] $X(n) = \Theta(1)$
- [ ] $X(n) = \Theta(\log n)$
- [ ] $X(n) = \Theta(n^2)$
- [ ] $X(n) = \Theta(n)$

Votre réponse est incorrecte.

The master theorem does not apply here. Go rewatch video 21 if you did not recognize this equation.

The tight solution $\Theta(n \log n \log n)$ also belong to $O(n \log n \log n)$ and $\backslash O(n^2)$.

Les réponses correctes sont :

$X(n) = \Theta(n(\log n)(\log n))$

,

$X(n) = O(n(\log n)(\log n))$

,

$X(n) = O(n^2)$

Question **5**

Incorrect

Note de 0,00 sur 1,00

Quelles sont les classes de complexités compatiables avec l'équation suivante?

$W(n) = W(\lfloor n/2 \rfloor) + W(\lceil n/2 \rceil) + \Theta(\log n)$.

Veuillez choisir au moins une réponse.

- [ ] $W(n) = \Theta(n \log n \log n)$
- [ ] $W(n) = O(n \log n)$
- [ ] $W(n) = \Theta(1)$
- [ ] $W(n) = \Theta(\log n)$

- [ ] $W(n) = \Theta(n^2)$
- [ ] $W(n) = O(n^2)$
- [ ] $W(n) = O(\log n)$
- [x] $W(n) = \Theta(n \log n)$ ✖
- [ ] $W(n) = O(n \log n \log n)$
- [ ] $W(n) = \Theta(n)$
- [ ] $W(n) = O(n)$

Votre réponse est incorrecte.

The $\lfloor \cdot \rfloor$ and $\lceil \cdot \rceil$ were only here to frighten you.

This equation has the shape $W(n) = 2W(n/2 + O(1)) + \Theta(\log n)$ where the master theorem can be applied.

We have $a = 2$, $b = 2$, $n^{\log_2(2)} = n^1 = n$, so we must compare $\Theta(\log n)$ to $\begin{cases} O(n^{1-\varepsilon)}) \\ \Theta(n) \\ \Omega(n^{1+\varepsilon}) \end{cases}$ $.

This is the first case with for instance $\varepsilon = 0.9$ (any value such that $0 < \varepsilon < 1$ works because $\log n$ is dominated by n^p for any $p > 0$).

Conclusion: $W(n) = \Theta(n^{\log_2(2)}) = \Theta(n)$. However $\Theta(n)$ is also included in $O(n)$, $O(n \log n)$, $O(n \log n \log n)$, and $O(n^2)$.

Les réponses correctes sont : $W(n) = \Theta(n)$

,

$W(n) = O(n)$

,

$W(n) = O(n \log n)$

,

$W(n) = O(n \log n \log n)$

,

$W(n) = O(n^2)$

Description

Le code Python qui suit, produit par ChatGPT 3.5 puis corrigé à la main, est l'implémentation d'un algorithme connu de multiplication matricielle, mais que vous n'avez pas besoin de savoir expliquer pour répondre aux questions. On suppose que les arguments A et B sont des matrices carrées de taille \(n\times n\) où \(n\) est une puissance de 2. On suppose de plus que tous les coefficients des matrices sont des entiers.

```
def Mult(A, B):
    if len(A) == 1:
        return [[A[0][0] * B[0][0]]]

    # Matrix dimension
    n = len(A)

    # Divide matrices into quarters
    A11 = [row[:n // 2] for row in A[:n // 2]]
    A12 = [row[n // 2:] for row in A[:n // 2]]
    A21 = [row[:n // 2] for row in A[n // 2:]]
    A22 = [row[n // 2:] for row in A[n // 2:]]

    B11 = [row[:n // 2] for row in B[:n // 2]]
    B12 = [row[n // 2:] for row in B[:n // 2]]
    B21 = [row[:n // 2] for row in B[n // 2:]]
    B22 = [row[n // 2:] for row in B[n // 2:]]

    # Recursive calls
    P1 = Mult(A11, subtract_matrices(B12, B22))
    P2 = Mult(add_matrices(A11, A12), B22)
    P3 = Mult(add_matrices(A21, A22), B11)
    P4 = Mult(A22, subtract_matrices(B21, B11))
    P5 = Mult(add_matrices(A11, A22), add_matrices(B11, B22))
    P6 = Mult(subtract_matrices(A12, A22), add_matrices(B21, B22))
    P7 = Mult(subtract_matrices(A11, A21), add_matrices(B11, B12))

    # Combine results
    C11 = add_matrices(subtract_matrices(add_matrices(P5, P4), P2), P6)
    C12 = add_matrices(P1, P2)
    C21 = add_matrices(P3, P4)
    C22 = subtract_matrices(subtract_matrices(P5, P3), subtract_matrices(P7, P1))

    # Concatenate result matrices
    result_matrix = [C11[i] + C12[i] for i in range(len(C11))]
    result_matrix += [C21[i] + C22[i] for i in range(len(C21))]
    return result_matrix
```

```
def add_matrices(A, B):
    return [[A[i][j] + B[i][j] for j in range(len(A))] for i in range(len(A))]

def subtract_matrices(A, B):
    return [[A[i][j] - B[i][j] for j in range(len(A))] for i in range(len(A))]
```

Question **6**

Incorrect

Note de 0,00 sur 1,00

Soit $A(n)$ le nombre d'appels à la fonction `add_matrices` pendant la multiplication de deux matrices $n\times n$. $A(n)$ doit satisfaire une équation récursive de la forme:

$\begin{cases}A(1)=0\\A(n)=\alpha A(n/\beta)+\gamma\end{cases}$

où α,β,γ sont des entiers.

Trouvez α,β,γ et entrez la valeur du produit $\alpha\times\beta\times \gamma$

Réponse : 24 ✖

$A(n)=7A(n/2)+10$

La réponse correcte est : 140

Question **7**

Incorrect

Note de 0,00 sur 1,00

Quel est le véritable nom de l'algorithme implémenté ci-dessus? (La réponse attendue ne contient qu'un seul mot. Les majuscules ne sont pas importantes.)

Réponse : kleen ✖

La réponse correcte est : Strassen

Question **8**

Incorrect

Note de 0,00 sur 1,00

Notions \(T(n)\) la complexité temporelle de la fonction Mult lorsqu'on l'applique sur deux matrices de taille \(n\times n\). Lesquelles des égalités suivantes sont vraies?

Veuillez choisir au moins une réponse.

- [ ] \(T(n)=\Theta(n\log n)\)
- [ ] \(T(n)=\Theta(1)\)
- [ ] \(T(n)=O(n^{\log_2 7})\)
- [ ] \(T(n)=O(n^3)\)
- [ ] \(T(n)=O(n\log n)\)
- [ ] \(T(n)=O(n)\)
- [ ] \(T(n)=O(n^2)\)
- [ ] \(T(n)=\Theta(\log n)\)
- [ ] \(T(n)=\Theta(n^{\log_2 7})\)
- [ ] \(T(n)=O(n^{\log_7 2})\)
- [ ] \(T(n)=\Theta(n^{\log_7 2})\)
- [ ] \(T(n)=\Theta(n^3)\)
- [ ] \(T(n)=O(\log n)\)
- [x] \(T(n)=\Theta(n^2)\) ✖
- [ ] \(T(n)=\Theta(n)\)

Your answer is incorrect.

Les réponses correctes sont :
\(T(n)=\Theta(n^{\log_2 7})\)

,
\(T(n)=O(n^{\log_2 7})\)

,
\(T(n)=O(n^3)\)

Question **9**

Incorrect

Note de 0,00 sur 1,00

L'algorithme de maintenance de la médiane est utilisé pour calculer efficacement la médiane au fur et à mesure que les valeurs d'un flux de nombres sont lues. L'idée de base est de maintenir deux tas : un tas-max pour stocker la moitié inférieure des nombres et un tas-min pour stocker la moitié supérieure. Cela permet un accès rapide à la médiane courrante à toute étape de la lecture du flux.

Voici une description étape par étape de l'algorithme :

1. **Initialisation des tas :**
   - Créer un tas-max vide (appelé `maxHeap`) pour stocker la moitié inférieure des nombres.
   - Créer un tas-min vide (appelé `minHeap`) pour stocker la moitié supérieure des nombres.
2. **Traitement des nombres :**
   - Pour chaque nombre entrant dans le flux :
     - Le comparer avec la médiane actuelle. Si le nombre est plus petit que la médiane, l'insérer dans `maxHeap`; sinon, l'insérer dans `minHeap`.
     - S'assurer que la différence de taille entre `maxHeap` et `minHeap` est au plus égale à 1. Si la différence devient plus grande, déplacer la racine du tas le plus grand vers le tas le plus petit.
3. **Calcul de la médiane :**
   - Si le nombre total d'éléments est impair, la médiane est la racine du tas le plus grand (quel que soit le tas qui a le plus d'éléments).
   - Si le nombre total d'éléments est pair, la médiane est la moyenne des racines des deux tas.

Notons $T(n)$ ne nombre total d'opérations effectuées pour traîter n nombres. Lesquelles des égalités suivantes sont vraies?

Veuillez choisir au moins une réponse.

- [ ] $T(n)=O(n^2)$
- [ ] $T(n)=\Omega(n^2)$
- [ ] $T(n)=O(\log n)$
- [ ] $T(n)=\Theta(1)$
- [ ] $T(n)=\Omega(n)$
- [x] $T(n)=\Theta(n^2)$ ✖
- [ ] $T(n)=\Theta(n)$
- [ ] $T(n)=O(n)$
- [ ] $T(n)=\Omega(n\log n)$
- [ ] $T(n)=\Theta(\log n)$
- [ ] $T(n)=O(1)$
- [ ] $T(n)=\Theta(n\log n)$
- [ ] $T(n)=\Omega(\log n)$
- [ ] $T(n)=O(n\log n)$

Votre réponse est incorrecte.

Initialisation is constant time.

Processing one number takes between $\Theta(1)$ and $\Theta(\log m)$ where m is the number of values read so far. As a consequence processing n numbers will take at least $\Theta(n)$ operations and up to $\sum_{m=1}^n\Theta(\log m)= \Theta(n\log n)$.

Les réponses correctes sont :
$T(n)=\Theta(n\log n)$

,

\(T(n)=O(n\log n)\)

,

\(T(n)=O(n^2)\)

,

\(T(n)=\Omega(\log n)\)

,

\(T(n)=\Omega(n)\)

,

\(T(n)=\Omega(n\log n)\)

Question **10**

Incorrect

Note de 0,00 sur 1,00

On considère le tas “max” correspondant au tableau suivant:

9	8	6	5	7	4	3	1	2

.

Donnez l’état de ce tas après les suppressions successives de ses trois plus grandes valeurs. (Saissiez les chiffres les uns après les autres, dans l’ordre où ils apparaissent dans le tableau, comme s’il s’agissait d’un nombre. Par exemple si vous pensez que le tableau contient encore les 6 plus petites valeurs l’ordre croissant, saisissez 123456.)

Réponse : 645312 ✖

La réponse correcte est : 654123

◄ CPXA_30.10.2023 (English)

Aller à…