IFT 2121 H99 **Pierre McKenzie**
Introduction à l'algorithmique

EXAMEN INTRA

11:30–13:20, salle N-615, pavillon principal

Directives:

- Aucune documentation n'est permise.
- Répondez <u>sur le questionnaire</u>, dans l'espace libre qui suit chaque question.
- L'énoncé de notre "théorème 0.3" sur les récurrences asymptotiques est reproduit au bas de la dernière page de l'examen. Lorsque vous l'utilisez, <u>dites quel cas s'applique</u>.

1. ______ /18
2. ______ /12
3. ______ /36
4. ______ /20
5. ______ /14

Total: ______ /100

Nom: ________________________________ **Code permanent:** ________________

1. **(18 points)** Notations asymptotiques. Soient $f, g : \mathbb{N} \to \mathbb{R}^{\geq 0}$.

   (a) (6 pts) Donnez la définition formelle complète de

      i. $O(f(n))$:

      ii. $\Omega(g(n))$:

   (b) (6 pts) Prouvez rigoureusement, à partir des définitions, que

$$\forall f \forall g,\ [\ O(f) \neq O(g)\ ] \Longrightarrow [f \notin O(g) \ \text{ ou } \ g \notin O(f)].$$

   (c) (3 pts) Est-il vrai pour tout f et g que $g \in \Omega(f)$ implique $g \notin O(f)$? (Justifiez.)

(d) (3 pts) Est-il possible qu'un algorithme A dont le temps d'exécution en pire cas est dans $\Theta(n^2)$ fonctionne plus lentement, sur chaque exemplaire, qu'un autre algorithme B dont le temps en pire cas est dans $O(n^3)$? (Justifiez.)

2. **(12 points)** Récurrences. Résolvez exactement la récurrence suivante:

$$\begin{aligned} f_0 &= 0 \\ f_1 &= 1 \\ f_n &= 3f_{n-1} - 2f_{n-2} + 2 \qquad \text{si } n \geq 2. \end{aligned}$$

3. **(36 points)** Diviser-pour-régner.
Chaque sous-question concerne un tableau $S[1..n]$ d'entiers positifs distincts.

(a) Considérez la fonction suivante:

fonction *Hiha* $(S[1..n])$
{ retourne un tableau d'entiers }
si $n \leq 1$ **alors** retourner (S)
sinon
$V \leftarrow ...$
$PP \leftarrow$ tableau formé de $\{ S[i] \mid 1 \leq i \leq n$ et $S[i] < V \}$
$PG \leftarrow$ tableau formé de $\{ S[i] \mid 1 \leq i \leq n$ et $S[i] > V \}$
retourner (concaténer(*Hiha*(PP), V, *Hiha*(PG)))

Posez une récurrence asymptotique raisonnable décrivant le temps d'exécution $T(n)$ de la fonction *Hiha* en pire cas et donnez explicitement $O(T(n))$, si

i. (4 pts) $V \leftarrow$ plus petit élément de S,

ii. (4 pts) $V \leftarrow$ médiane de S,

iii. (4 pts) $V \leftarrow S[1]$.

(b) Nous avons beaucoup accéléré la multiplication de grands entiers en divisant récursivement le problème en trois parties plutôt qu'en deux. Nous voulons appliquer cette idée au tri par fusion, qui consiste à trier récursivement chaque moitié d'un tableau, et à fusionner les deux sous-tableaux triés en un tableau trié complet.

  i. (4 pts) Donnez la récurrence asymptotique qui décrirait raisonnablement le temps d'exécution $t(n)$ d'un tri par fusion ainsi modifié:

  ii. (4 pts) Justifiez brièvement que votre récurrence est la bonne.

  iii. (4 pts) Donnez explicitement $O(t(n))$, et dites en quoi (le cas échéant) le tri par fusion modifié procure un avantage sur le tri par fusion basé sur la subdivision en deux parties.

(c) Supposons que les entiers du tableau $S[1..n]$ sont des entiers de n bits. Nous voulons calculer les n bits les moins significatifs de $S[1] \times S[2] \times \ldots \times S[n]$. Nous connaissons une fonction

Produit(i un entier de m bits, j un entier de m bits): entier de m bits,

dont le temps d'exécution est dans $\Theta(m^{\log_2 3})$, qui calcule les m bits les moins significatifs de $i \times j$.

i. (4 pts) Donnez avec brève justification une fonction $g(n)$, la plus simple possible, telle que le temps d'exécution d'un algorithme naïf calculant à l'aide de *Produit* les n bits les moins significatifs de $\prod_{i=1}^{n} S[i]$ est dans $\Theta(g(n))$.

ii. (4 pts) Quel est l'ordre exact, i.e. le Θ, d'un $t(n)$ vérifiant la récurrence

$$t(n) \in t(\lfloor n/2 \rfloor) + t(\lceil n/2 \rceil) + \Theta(n^{\log_2 3}) \quad ?$$

iii. (4 pts) Dites, en justifiant brièvement, si la récurrence ci-dessus décrit bien le temps d'exécution de l'appel *Produit_itéré*($S[1..n]$,n) à la fonction

fonction *Produit_itéré*($S[1..n], m$) { $S[1..n]$ est un tableau d'entiers de m bits }
{ Calcule un entier formé des m bits les moins significatifs de $\prod_{i=1}^{n} S[i]$ }
  si $n = 1$ **alors retourner** $S[1]$
  sinon
    HAUT ← *Produit_itéré*($S[1..\lfloor n/2 \rfloor], m$)
    BAS ← *Produit_itéré*($S[\lfloor n/2 \rfloor + 1..n], m$)
    retourner *Produit*(HAUT, BAS)

4. **(20 points)** Algorithmes voraces.
Définissons le *poids* d'une colonne d'une matrice comme la somme des entrées de cette colonne. Complétez la description d'un algorithme vorace résolvant le problème suivant:

   Donnée: matrice M de petits entiers positifs, de dimension $n \times n$,
   Calculer: un ensemble de colonnes de M, linéairement indépendantes, choisies de manière à maximiser la somme des poids des colonnes de l'ensemble.

   Vous pouvez faire appel à une fonction de tri sur un tableau de couples (*poids, numéro de colonne*), et à une fonction *indép*$(T[1..n, 1..n], k)$ qui retourne VRAI ou FAUX selon que les k premières colonnes du tableau T sont linéairement indépendantes ou non.

   (a) (10 pts) Votre algorithme:
   fonction *Maximal*$(T[1..n, 1..n], k)$
   $\{$En entrée: $k = n$, et T contient la matrice d'entiers positifs $M\}$
   $\{$En sortie: les premières colonnes de T sont les colonnes choisies et k est leur nombre.$\}$

(b) (5 pts) En faisant le lien avec la matière du cours, donnez une bonne raison pour laquelle on peut avoir confiance en l'exactitude de $Maximal(T[1..n, 1..n], k)$.

(c) (5 pts) En supposant l'addition de deux entiers à coût unitaire, et un coût de $f(n) \in \Omega(n)$ pour chaque appel à $ind\acute{e}p(T[1..n, 1..n], k)$, estimez raisonnablement l'ordre du temps d'exécution de votre fonction $Maximal(T[1..n, 1..n], n)$. (Justifiez très brièvement.)

5. **(14 points)** Permutations.
Soient $g = (13)(24)$ et $h = (3546)$ deux permutations de l'ensemble $\{1, 2, 3, 4, 5, 6\}$.

(a) (3 pts) Quelle est la permutation h^{-1}?

(b) (3 pts) Quelle est la permutation $g * h * g$?

(c) (8 pts) Un tortionnaire pressé vous menace des pires supplices à moins que le nombre de permutations qui s'expriment sous forme d'un quelconque produit des permutations g et h soit supérieur à 30. Coincidence heureuse, votre tortionnaire connaît bien le principe de l'algorithme basé sur le tamisage, que vous connaissez également. Pouvez-vous *rapidement* convaincre votre bourreau qu'il aurait tort de s'en prendre à vous, c'est à dire que le nombre de permutations exprimables est en effet supérieur à 30? (Justifiez.)

À votre service...theorème 0.3: Soient $a_1, a_2 \in \mathbb{R}^{\geq 0}$ tels que $a = a_1 + a_2 \geq 1$. Soit $b \geq 2$ un entier. Soit la fonction $t : \mathbb{N} \longrightarrow \mathbb{R}^{\geq 0}$ dont on ne connaît que la récurrence asymptotique

$$t(n) \in a_1 t(\lceil n/b \rceil) + a_2 t(\lfloor n/b \rfloor) + O(h(n)),$$

où $h : \mathbb{N} \longrightarrow \mathbb{R}^{\geq 0}$. Soit $\varepsilon \in \mathbb{R}^{\geq 0}$.

1. Si $\varepsilon > 0$ et $(\log_b a - \varepsilon) \geq 0$ et $h(n) \in O(n^{\log_b a \, - \, \varepsilon})$ alors $t(n) \in O(n^{\log_b a})$.
2. Si $\varepsilon > 0$ et $(\log_b a + \varepsilon) \geq 0$ et $h(n) \in O(n^{\log_b a \, + \, \varepsilon})$ alors $t(n) \in O(n^{\log_b a \, + \, \varepsilon})$.
3. Si $h(n) \in O(n^{\log_b a} (\log n)^{\varepsilon})$ alors $t(n) \in O(n^{\log_b a} (\log n)^{\varepsilon + 1})$.

S'applique aussi verbatim si $(\forall n \geq 1)[t(n) > 0]$ et *tous* les "O" sont remplacés par des "Ω".