Devoir 1

Yuchen Hui 20150470, Yan Zhuang 20146367

19 février 2022

1 Q1

1.1 a)

Faux.

Démonstration. Voici un contre-exemple. Soit

$$f_1 = 3x, f_2 = 2x, g_1 = x^2 + 1, g_2 = x^2.$$

On voit bien que

$$3x \in \mathcal{O}\left(x^2 + 1\right) \text{ et } 2x \in \mathcal{O}\left(x^2\right).$$

Cependant nous avons

$$f_1 - f_2 = x \notin \mathcal{O}\left(x^2 + 1 - x^2\right) = \mathcal{O}(1),$$

ce qui contredit l'énoncé $f_1 \in \mathcal{O}(g_1) \wedge f_2 \in \mathcal{O}(g_2) \implies f_1 - f_2 \in \mathcal{O}(g_1 - g_2)$

□

1.2 b)

Vrai.

Démonstration.

$$\begin{aligned}
&\exists k \in \mathbb{N}^*, \forall n > k\,[f_1(n) \leq f_2(n)] \\
&\implies \forall n > k, [f_1(n) + f_2(n) \leq 2f_2(n)] \\
&\implies \exists c \in \mathbb{R}^+, k \in \mathbb{N}^*, \forall n > k, [f_1(n) + f_2(n) \leq cf_2(n)].
\end{aligned}$$

Donc d'après la définition de la notation big O, nous concluons que $f_1(n) + f_2(n) \in \mathcal{O}(f_2(n))$. □

2 Q2

2.1 a)

$$\begin{aligned}\frac{an^k}{\log_3 n} =& a \cdot \frac{n^k}{\log_3 n} \\ \leq& a \cdot \frac{n^k}{\log_3 n} \cdot \log_3 n (\forall n > 1) \\ \leq& a \cdot n^k \\ \in& O(n^k)\end{aligned}$$

On doit toutefois prouver que $\frac{an^k}{\log_3 n} \notin \Omega(n^k)$ afin d'arriver à la conclusion que $\frac{an^k}{\log_3 n} \notin \Theta(n^k)$
Supposons que $\frac{an^k}{\log_3 n} \in \Omega(n^k)$, cela veut dire qu'on peut trouver un constant $c > 0$ que $\frac{an^k}{\log_3 n} \geq c \cdot n^k$

$$\begin{aligned}\frac{an^k}{\log_3 n} \geq& c \cdot n^k \\ \frac{a}{\log_3 n} \geq& c \\ \frac{1}{\log_3 n} \geq& \frac{c}{a}\end{aligned}$$

Si c'est vrai, l'équation devrait être vrai pour tout n. Mais c'est évident ici, vu que $\frac{c}{a}$ est un constant, il exist un n qui viole cette contrainte. Donc, on arrive à une contradiction. Cela veut dire que $\frac{an^k}{\log_3 n} \notin \Omega(n^k)$, qui veut donc dire $\frac{an^k}{\log_3 n} \notin \Theta(n^k)$

2.2 b)

$$\begin{aligned}2^{n+a} =& 2^n \cdot 2^a \\ \leq& c \cdot 2^n (\forall c | c \geq 2^a) \\ \in& O(2^n)\end{aligned}$$

2.3 c)

Supposons, par contradiction, que $2^{2n+1} \in O(2^n)$, selon la définition, on peut trouver un $c > 0 | 2^{2n+1} \leq c \cdot 2^n$. Toutefois, on a les transformations suivantes

$$\begin{aligned}2^{2n} \cdot 2 \leq& c \cdot 2^n \\ 2^{2n} \leq& c \cdot 2^{n-1} \\ \frac{2^{2n}}{2^{n-1}} \leq& c \\ 2^{n+1} \leq& c\end{aligned}$$

Si notre hypothèse est correcte, alors une fois que l'on fixe un c, cette équation devrait être correct pour n'importe quel n. Toutefois, on voit que ce n'est clairement pas le cas. On peut toujours trouver un n qui va violer cette équation (Avec un c fixé)
Par conséquent, on arrive à une contradiction. Cela veut dire que $2^{2n+1} \in O(2^n)$

2.4 d)

Celui-ci est plus facile à prouver avec une preuve par induction.
Afin de prouver que $2^n \in O(n!)$, cela veut dire que l'on peut trouver un constant c tel que $2^n \leq c \cdot n!$ pour $\forall n, n \geq n_0$. Pour faciliter la preuve, pour le cas de base, on choisit $n_0 = 1, c = 2, n = n_0$
Pour le cas de base, on a donc :

$$2^1 \leq 2 \cdot 1$$

On voit clairement que cette équation est bien valide. Cela veut dire que l'hypothèse est vrai pour $n_0 = 1$. Toutefois, afin de pouvoir conclure que $2^n \in O(n!)$, il faut que l'hypothèse est vrai pour tout $n | n \geq n_0$ pour un c fixé.

On suppose que c'est vrai pour tout $n, n \geq n_0$, on doit prouver que $n+1$ est vrai aussi. On a donc :

$$2^{n+1} \leq c \cdot (n+1)!$$

On voit que à la gauche, on multiple l'équation par 2. Toutefois, à la droite, on multiple l'équation par $(n+1)$ qui est, par notre hypothèse, au moins 2. Donc cette inégalité est toujours vrai. Donc, c'est vrait pour tout $n \geq n_0$ avec un c fixé.

On peut donc conclure maintenant que $2^n \in O(n!)$.

2.5 e)

Si on suppose que $n! \in O(2^n)$, cela veut dire que $n! \leq c \cdot 2^n$ pour un c fixé avec $n \geq n_0$

Toutefois, on ne peut pas trouver un tel c. On a :

$$\begin{aligned} n! &\leq c \cdot 2^n \\ \frac{n!}{2^n} &\leq c \end{aligned}$$

Sans utiliser la règle de limite, on doit montrer qu'il existe pas une borne supérieur pour $\frac{n!}{2^n}$. On a :

$$\begin{aligned} \frac{n!}{2^n} &= \frac{n}{2} \cdot \frac{n-1}{2} \cdot \cdots \cdot \frac{2}{2} \cdot \frac{1}{2} \\ &\geq \frac{n}{2} \cdot \frac{1}{2} \\ &= \frac{n}{4} \end{aligned}$$

On voit que $\frac{n}{4}$ n'est jamais décroissant (non borné), et vu qu'on a $\frac{n!}{2^n} \geq \frac{n}{4}$, $\frac{n!}{2^n}$ est aussi non borné. Cela veut dire que l'on ne peut jamais trouver un c fixé pour que $n! \leq c \cdot 2^n$ pour tout $n \geq n_0$.

Cela veut dire aussi que $n! \notin O(2^n)$

3 Q3

3.1 a)

On a $f(n) = \frac{n^2}{ln(n)}, g(n) = n \cdot \sqrt{n}$ On applique la règle de la limite

$$\begin{aligned}
\lim_{n->\infty} \frac{\frac{n^2}{\ln(n)}}{n \cdot \sqrt{n}} &= \lim_{n->\infty} \frac{n^2}{\ln(n) \cdot n \cdot \sqrt{n}} \\
&= \lim_{n->\infty} \frac{n^2}{n^{\frac{3}{2}} \cdot \ln(n)} \\
&= \lim_{n->\infty} \frac{n^{\frac{1}{2}}}{\ln(n)} \\
&= \lim_{n->\infty} \frac{1}{2} n^{\frac{1}{2}} \text{ (L.H)} \\
&= \infty
\end{aligned}$$

En utilisant la règle de limite, on arrive à un résultat de ∞. Donc, on peut conclure que $g(n) \in O(f(n))$

3.2 b)

On a $f(n) = ln(\sqrt{n}), g(n) = ln(n)$. On applique la règle de limite

$$\begin{aligned}
\lim_{n->\infty} \frac{\ln(\sqrt{n})}{\ln(n)} &= \frac{1}{2} \lim_{n->\infty} \frac{\frac{1}{n}}{\frac{1}{n}} \\
&= 1
\end{aligned}$$

On arrive à un résultat d'un entier constant, on peut conclure que $f(n) \in \Theta(g(n))$

3.3 c)

On a $f(n) = n^a, g(n) = 2^{\sqrt{ln(n)}}$. On applique la règle de limite

$$\begin{aligned}
y &= \lim_{n->\infty} \frac{n^a}{2^{\sqrt{ln(n)}}} \\
\ln(y) &= \lim_{n->\infty} \ln\left(\frac{n^a}{2^{\sqrt{ln(n)}}}\right) \\
\ln(y) &= \lim_{n->\infty} \left(\ln(n^a) - \ln(2^{\sqrt{ln(n)}})\right) \\
\ln(y) &= \lim_{n->\infty} \left(a\ln(n) - \sqrt{\ln(n)}\ln(2)\right) \\
\ln(y) &= \lim_{n->\infty} \left(\frac{a^2 \cdot \ln^2(n) - \ln(n) \cdot \ln^2(2)}{a\ln(n) + \sqrt{\ln(n)}\ln 2}\right) \\
\ln(y) &= \lim_{n->\infty} \left(\frac{2a^2 \cdot \ln(n) \cdot \frac{1}{n} - \ln^2(2) \cdot \frac{1}{n}}{a \cdot \frac{1}{n} + \frac{1}{2} \cdot \frac{1}{\sqrt{\ln(n)}} \cdot \frac{1}{n} \ln 2}\right) \text{ (L.H)} \\
\ln(y) &= \lim_{n->\infty} \left(\frac{2a^2 \cdot \ln(n) - \ln^2(2)}{a + \frac{1}{2} \cdot \frac{1}{\sqrt{\ln(n)}} \cdot \ln 2}\right) \\
\ln(y) &= \frac{2a^2 \cdot \ln(\infty) - \ln^2(2)}{a + \frac{1}{2} \cdot \frac{1}{\sqrt{\ln(\infty)}} \cdot \ln 2} \\
\ln(y) &= \frac{\infty}{a} \\
\ln(y) &= \infty \\
y &= e^{\infty} \\
y &= \infty
\end{aligned}$$

On arrive à un résultat d'une infinité, on peut conclure que $g(n) \in O(f(n))$

4 Q4

4.1 a)

Pour faciliter le calcul, on post $t_i = T(2^i)$ par remplaçer n par 2^i. On a donc une formule de récurrence :

$$t_i = \frac{3}{2}t_{i-1} - \frac{1}{2}t_{i-2} + 2^i, i \neq 1$$

On a la transformation suivante :

$$t_i - \frac{3}{2}t_{i-1} + \frac{1}{2}t_{i-2} = 2^i$$

Ici, $b = 2, p(n) = 1$ de degré de 0. En utilisant la formule (4.10) du livre, on obtient la polynôme caractéristique suivante :

$$(x^2 - \frac{3}{2}x + \frac{1}{2})(x-2),$$

soit

$$(x - \frac{1}{2})(x-1)(x-2).$$

Par conséquent, les racines sont $x_1 = \frac{1}{2}, x_2 = 1, x_3 = 2$, dont la multiplicite sont tous 1. Donc on a une formule de récurrence ayant la forme suivante :

$$t_i = c_1\left(\frac{1}{2}\right)^i + c_2 + c_3 2^i.$$

En appliquant $n = 2^i$, on obtient

$$T(n) = c_1\frac{1}{n}1 + c_2 + c_3 n.$$

On peut utiliser les conditions initiales pour trouver les constants

$$T(1) = 1, T(2) = \frac{3}{2}T(4) = \frac{3}{2}$$

Donc, on a quatre équations, assez pour déterminer chaque constant

$$\begin{aligned} c_3 + c_4 &= 1 \\ c_1 + c_2 + c_3 + \frac{1}{2}c_4 &= \frac{3}{2} \\ 4c_1 + 2c_2 + c_3 + \frac{1}{4}c_4 &= \frac{23}{4} \\ 9c_1 + 3c_2 + c_3 + \frac{1}{8}c_4 &= \frac{127}{8} \end{aligned}$$

Après la résolution du syèteme on obtient :

$$t_i = 4i^2 - 12i + 18 - 17 \cdot \left(\frac{1}{2}\right)^i$$

$$\begin{aligned} 2^i &= n \iff i = \log_2 n \\ T(n) &= 4\log_2^2 n - 12\log_2 n - 17 \cdot \left(\frac{1}{2}\right)^{\log_2 n} + 18 \\ &= 4\log_2^2 n - 12\log_2 n - 17 \cdot \frac{1}{n} + 18 \\ &\leq 4\log_2^2 n + 18\log_2^2 n \\ &= 22\log_2^2 n \\ T(n) &\in O(\log_2^2 n | n \text{ est une puissance de } 2) \end{aligned}$$

4.2 b)

Pour arriver à la conslusion que $T(n) \in O(\log_2^2 n) \forall n$, on doit d'abord prouver que $T(n)$ est e.n.d. On peut analyser le dérivée pour voir s'il peut être non-décroissant à partir d'une certaine indice.

$$\begin{aligned}\frac{d}{dn}\left(4\log_2^2 n\right) - \frac{d}{dn}\left(12\log_2 n\right) - \frac{d}{dn}\left(17 \cdot \frac{1}{n}\right) &= \frac{8\log_2 n}{n\ln(2)} - \frac{12}{n\ln(2)} + \frac{17}{n^2} \\ &= \frac{8\log_2 n - 12}{n\ln(2)} + \frac{17}{n^2}\end{aligned}$$

On voit clairement que tant que c'est possible de trouver une indice à partir de laquelle, le dérivé de $T(n)$ sera strictement positive cela implique que la fonction $T(n)$ est e.n.d

On doit aussi prouver que $\log_2^2 n$ est une fonction lisse. On essaie de prouver que $f(2n) \in O(\log_2^2 n)$

$$\begin{aligned}f(2n) &= \log_2^2(2n) \\ &= \log_2^2 2 + \log_2^2 n \\ &= 1 + \log_2^2 n \\ &\leq \log_2^2 n + \log_2^2 n \\ &= 2\log_2^2 n \\ &\in O(\log_2^2 n)\end{aligned}$$

Vu qu'on a prouvé que $T(n)$ est e.n.d et que $\log_2^2 n$ est une fonction lisse, on peut arriver à la conslusion que $T(n) \in O(\log_2^2 n) \forall n$

5 Q5

5.1 a)

Cette approche gloutonne n'est pas optimale.Voici un contre-exemple :

Soit e_1, e_2, e_3 les trois événements à considérer, et la liste déjà triée L_E = [e_1 (d = 5 Janvier, f = 8 Janvier), e_2 (d = 1 Janvier, f = 6 Janvier), e_3 t(d = 7 Janvier, f = 6 May)] . Après l'éxecution de l'algorithme, on obtiendra S = $\{e_1\}$ et R = $\{e_2, e_3\}$, car e_1 est l'événement dont la durée est la moindre, puis malheuresement il existe des chevauchements des périodes entre e_1,e_2 et e_3. Cependant on constate que la solution optimale dans notre cas sera S^* = $\{e_2, e_3\}$, dont la cardinalité est plus grande que celle de S qui est choisi par notre algorithme a. Donc l'approche n'est pas optimale.

5.2 b)

Cette approche gloutonne n'est pas optimale.Voici un contre-exemple assez extrême :

Soit e_1, e_2, e_3 trois événements à considérer, et la liste déjà triée L_E = [e_1 (d = 1 Janvier, f = 31 Decembre), e_2 (d = 2 Janvier, f = 4 Janvier), e_3 t(d = 5 Janvier, f = 6 Janvier t)] . Après l'éxecution de l'algorithme, on obtiendra S = $\{e_1\}$ et R = $\{e_2, e_3\}$, tout simplement à cause du fait que e_1 commence le premier et il est en conflit avec tant e_1 que e_2. Cependant on constate que la solution optimale dans notre cas sera S^* = $\{e_2, e_3\}$, dont la cardinalité est plus grande que celle de S qui est choisi par notre algorithme b. Donc l'approche n'est pas optimale.

5.3 c)

Cette approche gloutonne est bien optimale. Voici une preuve par induction :

Démonstration. Nous allons montrer le prédicat

$P(n) : \forall n \in \mathbb{N}^*$, si n événements sont choisis par algorithme c, alors ces n événements forment une solution optimale.

par induction sur n.

Base d'induction. n = 1. Supposons qu'il existe m événements à considérer, alors l'algorithme $\underline{c}$ a choisi l'événement qui finit le plus tôt. Dénotons cet événement par e_1. On a que tout événement autre que e_1 est rejeté par l'algorithme à cause des conflits d'horaire, donc leurs dates de début sont tous plus tôt que f_{e_1}. En même temps, leurs dates de fin sont tous plus tard que f_{e_1}. En conséquence, nous concluons qu'au jour f_{e_1}, tout événement autre que e_1 se produira. Finalement, ces m événements sont en conflit l'un avec l'autre, donc nous ne pouvons choisir qu'un événement pendant l'année en tout cas, ce qui implique que P(1) est vrai.

Hypothèse d'induction. Supposons que $P(k)$ est vrai.

Étape d'induction Considérons maintenant le cas $k+1$. Maintenant, notre algorithme $\underline{c}$ a déjà choisi $k+1$ événements, dont l'événement e_1 fait partie (e_1 est l'événement qui finit le plus tôt). L'idée de preuve sera de voir le processus de sélection de ces $k+1$ événements en deux étapes : 1) choisir e_1, 2) choisir k événements en fonction de notre algorithme c. L'étape 2) est déjà optimale grâce à l'hypothèse d'induction.

Maintenant, le problème est, si nous pouvons toujours obtenir une solution optimale en choisissant e_1 à la première étape. Si nous pouvons prouver la proposition suivante :

Il existe sûrement une solution optimale dans laquelle e_1 est le premier événement, (1)

alors la question de chercher une solution optimale sera réduite à : 1) choisir e_1 2) Arranger le plus d'événements possibles après le jour f_{e_1}. Si nous appliquons algorithme c à étape 2), l'optimalité pourrait être garantie et nous obtienderons une solution optimale. Dans ce cas, nous trouverons que la solution trouvée par les deux étapes combinées est belle et bien les $k+1$ événements obtenus par l'algorithme c. Comme cela nous pourrions conclure que $P(k+1)$ est vrai.

Pour voir la véracité de la proposition 1, considérons une arbitraire solution optimale $S^{'}$ qui n'include pas e_1, nous allons voir nous pourrions toujours la transformer en une solution (aussi optimale) commançant par e_1. La méthode consiste à remplaçer le premier événement $e_1^{'}$ de $S^{'}$ par e_1. Ce changement ne causera aucun conflit avec d'autres événements dans $S^{'}$, parce que tout autre événement se produit après $f_{e_1'}$ et $f_{e_1'} \geq f_{e_1}$, donc après e_1. Maintenant, nous avons obtenu une solution optimale commançant par e_1.

□