Binome II - Exercice 2.10

17 janvier 2021

0.1 Motivations

La théorie de *la complexité* est le domaine des mathématiques, et plus précisément de l'informatique théorique, qui étudie formellement la quantité de ressources (temps, espace mémoire, etc.) dont a besoin un algorithme pour résoudre un problème algorithmique.

Excercice 2.10

Dans cet exercice on s'intéresse au calcul du nombre d'opérations, *du temps*, nécessaire á l'exécution d'un algorithme. La complexité étudiée est donc *temporelle.*

Soit u une fonction complexe et a-périodique. La T.F.D calcule les coefficients de Fourier de u à partir du N-échantillon $u(\frac{ka}{N}), k = 0, ..., N-1$.
En posant $w_N = e^{\frac{2i\pi}{N}}$ et $u_k = u(\frac{ka}{N}), k = 0, ..., N-1$ on a :

$$\widetilde{u}_n = \frac{1}{N}\sum_{l=0}^{N-1} u_l w_N^{-nl} = \frac{1}{N}\sum_{l=0}^{N-1} u_l e^{\frac{-2i\pi nl}{N}}.$$

Donc le calcul des $\widetilde{u}_n$, $n = 0, ..., N-1$. revient á évaluer le polynôme $P(X) = \sum_{k=0}^{N-1} a_k X^k$ en $e^{\frac{2i\pi k}{N}}$, les racines N-ième de l'unitée, avec $a_k = u_k$.
Par l'algorithme de Horner, on a, pour chaque $n = 0, ..., N-1$, un nombre d'opérations pour calculer les $\widetilde{u}_n$ d'ordre $\mathcal{O}(N)$, donc la complexité du calcul de la Transformation de Fourier Discrète est d'ordre $\mathcal{O}(N^2)$.

Le but de cet exercice est de diminuer cette complexité à un $\mathcal{O}(N\log(N))$.

0.2 L'algorithme de T.F.R de *Cooley-Tuckey*

On appelle $T(N)$ le nombre d'opérations arithmétiques requis pour évaluer $P(X) = \sum_{k=0}^{N-1} a_k X^k$ en les racines N-ième de l'unitée.

Pour des raisons de simplification, l'étude sera réduite sur l'ensemble $P_2 = \{2^n, n \in \mathbb{N}^*\}$.
On pose :

$$N = 2^n \ , \ \ Q(X) = \sum_{k=0}^{\frac{N}{2}-1} a_{2k}X^k, \quad \text{et} \quad R(X) = \sum_{k=0}^{\frac{N}{2}-1} a_{2k+1}X^k.$$

Pour k dans $\{0, ..., N-1\}$, on a :

$$\begin{aligned}
P(w_N^k) &= \sum_{l=0}^{N-1} a_l (w_N^k)^l \\
&= \sum_{l=0}^{\frac{N}{2}-1} a_{2l}(w_N^k)^{2l} + \sum_{l=0}^{\frac{N}{2}-1} a_{2l+1}(w_N^k)^{2l+1} \\
&= \sum_{l=0}^{\frac{N}{2}-1} a_{2l}((w_N^k)^2)^l + w_N^k \sum_{l=0}^{\frac{N}{2}-1} a_{2l+1}((w_N^k)^2)^l \\
&= Q((w_N^k)^2) + w_N^k R((w_N^k)^2).
\end{aligned}$$

Donc, pour évaluer P aux racines N-ième de l'unitée, il suffit d'évaluer Q et R aux $(w_N^k)^2, k = 0, ..., N-1$.. N étant pair, alors $(w_N^k)^2, k = 0, ..., N-1$ sont des racines $\frac{N}{2}$-ième de l'unitée.
L'évaluation de R et Q aux $(w_N^k)^2, k = 0, ..., \frac{N}{2}-1$ nécessite $T(\frac{N}{2})$ opérations et pour l'évaluation aux $(w_N^k)^2$ pour $k = \frac{N}{2}, ..., N-1$, remarquons que $(w_N^k)^2 = (w_N^{k-\frac{N}{2}})^2$ et $k-\frac{N}{2} = 0, ..., \frac{N}{2}-1$ donc le calcul est déjá fait. Enfin, puisque on effectue une multiplication et une somme pour atteindre

$$P(w_N^k) = Q((w_N^k)^2) + w_N^k R((w_N^k)^2)$$

, On obtient,

$$T(N) = 2N + 2T(\frac{N}{2}).$$

Montrons que : $T(N) = \mathcal{O}(N\log(N))$**.**

Démonstration. $N = 2^n, n \geq 1$.
On a :

$$\begin{aligned} T(2^{n+1}) = 2.2^{n+1} + 2T(\frac{2^{n+1}}{2}) \;\Rightarrow\; & \frac{T(2^{n+1})}{2^{n+1}} = 2 + \frac{T(2^n)}{2^n}, \text{ donc } (\frac{T(2^n)}{2^n}) \text{ est arithmétique de raison 2} \\ \Rightarrow\; & \frac{T(2^n)}{2^n} = \frac{T(2)}{2} + 2(n-1). \\ \Rightarrow\; & \frac{T(N)}{N} = \frac{T(2)}{2} - 2 + 2\log_2(N) \\ \Rightarrow\; & T(N) = N(\frac{T(2)}{2} - 2) + 2N\log_2(N) \\ \Rightarrow\; & T(N) = \mathcal{O}(N\log(N)). \end{aligned}$$

□