Université Bordeaux 1 — **N1MA8W02 – Master**
Mathématiques — Année 2020–2021

Devoir Surveillé, 10 mars 2021
Durée 1h30.

Exercice 1 – [ALGORITHME MYSTÈRE, FFT]

Algorithme 1. Mystère

Entrées: n : élément de $\mathbb{N} \setminus \{0,1\}$, G : groupe cyclique d'ordre n d'élément neutre 1, g : élément de G, $\mathcal{P}$: liste des diviseurs premiers de n
Sorties: 0 ou 1

1: $t = 1$, $i = 0$, $r = \text{taille}(\mathcal{P})$ *commentaire : l'indice de $\mathcal{P}$ va donc de 0 à $r-1$*
2: Tant que $i < r$ et $t = 1$:
3: $\quad h = g^{n/\mathcal{P}[i]}$
4: $\quad i = i + 1$
5: $\quad$ Si $h = 1$:
6: $\quad\quad t = 0$
7: Sortir t

---

1) Exécuter à la main cet algorithme avec les données : $n = 12$, $G = \mathbb{F}_{13}^*$ (c'est-à-dire le groupe multiplicatif de $(\mathbb{Z}/13\mathbb{Z}, +, .)$), $g = [2]_{13}$ (la classe de 2 dans $\mathbb{F}_{13}$) $\mathcal{P} = [2, 3]$. Quel est l'ordre de $[2]_{13}$ dans $\mathbb{F}_{13}$? En déduire une racine primitive quatrième de 1 dans $\mathbb{F}_{13}$, que l'on notera ω.
On revient maintenant au cas général.

2) Quand `Mystère`$(n, G, g, \mathcal{P})$ rend 1, que peut-on en déduire ? Justifier.

3) Montrer que $n \geqslant 2^r$. En déduire que $r \leqslant \log n$.

4) En déduire que la complexité algébrique de `Mystère` est en $O\left((\log n)^2\right)$.

5) Exécuter à la main `FFT(4,`ω`,P)` où $P = x^3 - 2x^2 + 2x + 3 \in \mathbb{F}_{13}[x]$ et où ω a été calculé à la question 1 (`FFT` est rappelé en fin d'énoncé).
Pour qui n'aurait pas calculé ω dans $\mathbb{F}_{13}$ (et uniquement dans ce cas), faire la question 5) en considérant P dans $\mathbb{C}[x]$, avec $\omega = i$.

Exercice 2 – [INVERSION RAPIDE MODULO x^n]
Soit K un corps. Soient $P \in K[x]$ tel que $P(0) = 1$ et $n \in \mathbb{N} \setminus \{0\}$.

1) Montrer que x ne divise pas P. En déduire que la classe $[P]_{x^n}$ de P dans $K[x]/(x^n)$ est inversible. Cet exercice porte sur un algorithme rapide pour calculer l'inverse de cette classe, en utilisant un algorithme de multiplication rapide.

On rappelle que la division euclidienne d'un polynôme par x^k consiste à tronquer le polynôme. Ce n'est pas pris en compte dans la complexité algébrique.

Soit (A_i) la suite définie de la manière suivante.

$$A_0 = 1 \quad , \quad A_{i+1} = 2A_i - PA_i^2 \quad \forall i \geqslant 0.$$

2) Montrer que $PA_i \equiv 1 \mod x^{2^i}$ pour tout $i \geqslant 0$.

3) Pour tout polynôme Q et tout entier r, on note $\mathrm{Rem}(Q, x^r)$ le reste de la division de Q par x^r (cela tronque le polynôme Q). Soit l'algorithme

---

Algorithme 2. Inverse modulo x^n

Entrées: n un entier naturel, P un polynôme de $K[x]$ tel que $P(0) = 1$

Sorties: le représentant de degré strictement inférieur à n de $[P]_{x^n}^{-1}$

1: $A = 1$, $k = 1$
2: Tant que $k < n$:
3: $\quad k = 2k$
4: $\quad P_0 = \mathrm{Rem}(P, x^k)$
5: $\quad A_2 = \mathrm{Rem}(A^2, x^k)$
6: $\quad A = 2A - \mathrm{Rem}(A_2 P_0, x^k)$
7: Sortir $\mathrm{Rem}(A, x^n)$

On suppose que la complexité algébrique de la multiplication de deux polynômes de $K[x]$ de degrés inférieurs à un entier N est en $O(N \log N)$.

Au i-ème passage dans la boucle "tant que", montrer qu'après le pas 3, $k = 2^i$. En déduire que ce i-ème passage se fait en $O(i2^i)$ opérations dans K.

4) Soient $C(n)$ la complexité algébrique de `Inverse(n,P)` et $r = \lceil \log n \rceil$ le plus petit entier supérieur ou égal à $\log n$ (alors $2^{r-1} < n \leqslant 2^r$). Montrer que $C(n)$ est en $O(r2^r)$.

5) En déduire que $C(n)$ est en $O(n \log n)$.

Rappel

Algorithme 3. FFT

Entrées: n : puissance de 2, ω : racine primitive n-ème de 1 dans un corps K, P : un polynôme de $K[x]$ de degré strictement inférieur à n (codé par la liste de longueur n de ses coefficients)

Sorties: $\mathcal{F}_\omega(P) = (P(1), P(\omega), \ldots, P(\omega^{n-1}))$

1: Si $n = 1$, alors sortir P
2: $m = n/2$
3: $P_0 = [P[2i]$ pour i dans $[0, m-1]]$
4: $P_1 = [P[2i+1]$ pour i dans $[0, m-1]]$
5: $T_0 = \mathtt{FFT}(m, \omega^2, P_0)$
6: $T_1 = \mathtt{FFT}(m, \omega^2, P_1)$
7: Pour k de 0 à $m-1$:
8: $\quad R[k] = T_0[k] + \omega^k T_1[k]$
9: $\quad R[k+m] = T_0[k] - \omega^k T_1[k]$
10: Sortir R
