TP 2 : Algorithme d'Euclide et multiplication de Karatsuba

1 Algorithme d'Euclide pour les entiers

On rappelle que l'algorithme d'Euclide consiste, partant de deux entiers a et b, à faire les opérations suivantes tant que $b \neq 0$: noter r le reste de la division euclidienne de a par b, poser $a = b$ et $b = r$. À la fin, a contient le pgcd de a et b.

- Étudier la différence entre les commandes `mod(a,m)` et `a%m` qui donnent la classe de `a` modulo `m`. Calculer alors le reste et le quotient de la division euclidienne de 53 par 22. On pourra utiliser les commandes `%` et `\\`.
- Calculer le pgcd d de 15132 et de 103196 en utilisant la commande `gcd` et déterminer deux entiers u et v tels que $d = 15132u + 103196v$. Retrouver la valeur de d en utilisant la commande `factor`.
- Implémenter l'algorithme d'Euclide en itératif et en récursif. Comparer avec la version originelle d'Euclide, qui utilisait simplement que
$$\text{pgcd}(a, b) = \text{pgcd}(\min(a, b), \max(a, b) - \min(a, b)),$$
que l'on implémentera également en itératif et en récursif.
- Faire renvoyer à votre fonction non seulement le pgcd, mais aussi le nombre d'étapes de calcul.
- Déterminer expérimentalement le nombre d'étapes nécessaires au calcul du pgcd de F_{n+2} et de F_{n+1} en affichant ce nombre pour $n \in \{1, \dots, 100\}$, où $(F_n)_{n\in\mathbb{N}}$ est la suite de Fibonacci. Observer une règle générale et la démontrer.
- Étendre votre implémentation de l'algorithme d'Euclide de façon à renvoyer non seulement le pgcd de a et b, mais aussi des entiers u et v tels que $au + bv = \text{pgcd}(a, b)$. Pour cela, on pourra introduire une matrice M carrée de taille 2 telle que si on note a et b les arguments de la fonction et a' et b' les nouvelles valeurs de ces variables au cours de l'exécution de l'algorithme, on ait $\begin{pmatrix} a' \\ b' \end{pmatrix} = M \begin{pmatrix} a \\ b \end{pmatrix}$.
- Écrire une fonction `inverse` prenant deux arguments a et n, et renvoyant un entier représentant l'inverse de a dans $\mathbf{Z}/n\mathbf{Z}$ si la classe de a est inversible, et déclenchant une exception sinon. On pourra utiliser la syntaxe `raise ZeroDivisionError` ou afficher un message d'erreur à la main à l'aide de la commande `print()`.
- Implémenter une fonction `chinois2` prenant quatre arguments a, b, m et n qui déclenche une exception si m et n ne sont pas premiers entre eux et sinon renvoie un entier congru à a modulo m et à b modulo n.
- Implémenter une fonction `chinois` prenant en argument deux listes d'entiers de même taille[1] $[a_1, \dots, a_k]$ et $[n_1, \dots, n_k]$, qui déclenche une exception si les n_i ne sont pas premiers entre eux deux à deux, et renvoie un entier congru à a_i modulo n_i pour tout $i \in \{1, \dots, k\}$ sinon. Donner dans le cas général lorsque les n_i ne sont pas nécessairement premiers entre eux une condition nécessaire et suffisante pour que le système
$$\begin{cases} x & \equiv & a_1 \pmod{n_1} \\ & \vdots & \\ x & \equiv & a_k \pmod{n_k} \end{cases}$$
admette une solution. Proposer un algorithme de résolution et l'implémenter.

1. On rappelle que la fonction `len` permet de déterminer le nombre d'éléments d'une liste.

2 Algorithme d'Euclide pour les polynômes

Une façon de manipuler les polynômes en x consiste à définir l'anneau $R = \mathbf{Q}[x]$ de la façon suivante :

```
R.<x> = PolynomialRing(QQ)
```

- Étudier les commandes `P%Q` qui donnent la classe de `P` modulo `Q`. Calculer alors le reste et le quotient de la division euclidienne de `x^3+2*x+1` par `x-2`. On pourra utiliser les commandes `%` et `\\`.
- Calculer le pgcd D de `-x^8+x^7+x^5-x^3-x+1` et de `x^7-x^6-x+1`, que l'on notera respectivement P et Q dans la suite, en utilisant la commande `gcd` et déterminer deux polynômes U et V tels que $D = PU + QV$. Retrouver la valeur de D en utilisant la commande `factor`.
- Tester les fonctions de la partie **1** sur des polynômes. Implémenter l'algorithme d'Euclide et l'algorithme d'Euclide étendu pour des polynômes en x. On renverra l'unique pgcd unitaire D et deux polynômes tels que $PU+QV = D$ pour D ce pgcd unitaire.
- Faire renvoyer à votre fonction non seulement le pgcd, mais aussi le nombre d'étapes de calcul. Faire également afficher la suite des restes dans le cas de `x^8+x^6-3x^4-3x^3+8x^2+2x-5` et de `3x^6+5x^4-4x^2-9x+21`. Que constatez-vous ? Suggérer des pistes d'amélioration !
- Définir une fonction qui prenne en argument un entier $n \geqslant 0$ et renvoie un couple de polynômes $(P, Q) \in \mathbf{Q}[x]^2$ avec $\deg P = n$ et $\deg P > \deg Q$ tel que le nombre d'étapes du calcul de l'algorithme d'Euclide pour (P, Q) soit n.
- Sans recours à la factorisation dans $\mathbf{Q}[x]$, écrire une fonction `SansCarre(P)` donnant la partie sans facteur carré d'un polynôme $P \in \mathbf{Q}[x]$. On rappelle que la *partie sans facteur carré* d'un polynôme est le produit des facteurs irréductibles 2 à 2 distincts de P.
- Soit $\zeta_{15} = e^{\frac{2i\pi}{15}} \in \mathbf{C}$. Écrire l'inverse de $1 + \zeta_{15}$ comme polynôme en ζ_{15} à coefficients rationnels.

3 Karatsuba

On représentera les entiers comme une suite de 0 et de 1. La suite $[a_0, \ldots, a_n]$ correspondant à l'entier $a_0 + 2a_1 + 2^2 a_2 + \cdots + 2^n a_n$.

- Écrire des fonctions permettant de passer de la représentation usuelle dans `Sage` à celle-ci et inversement.
- Implémenter l'addition des entiers représentés sous cette forme.
- Implémenter la multiplication en utilisant d'une part l'algorithme naïf et d'autre part celui de Karatsuba.

On cherche alors à comparer les temps d'exécution des trois algorithmes quand n augmente et à les comparer aux complexités théoriques du cours.

- Écrire une fonction `mesure(fun, nb_chiffres)` qui étant donnée une opération `fun` retourne le temps de calcul de `fun` sur deux entiers ayant `nb_chiffres` chiffres.
- Créer une liste `tailles` contenant la liste des nombres de chiffres des chiffres que l'on va utiliser. On partira de nombres à 10 chiffres jusqu'à des nombres à 2560 chiffres en multipliant le nombre de chiffres par 2 à chaque fois.
- Mesurer les temps de calcul de l'addition et des deux multiplications (naïve et Karatsuba), que l'on notera respectivement $\text{temps}_{\text{add}}$, $\text{temps}_{\text{naïf}}$ et $\text{temps}_{\text{Kara}}$. Que concluez-vous quant à la complexité de l'addition ?
- Déterminer une fonction f telle que $f(\text{temps}_{\text{naïf}}(n))$ et $f(\text{temps}_{\text{Kara}}(n))$ soient en théorie linéaires. Effectuer les tracés correspondants.
- À l'aide de la librairie python `scipy.optimize`, déterminer $C_{\text{naïf}}$, C_{Kara} ainsi que α_1, α_2 tels que

$$\text{temps}_{\text{naïf}}(n) \approx C_{\text{naïf}} n^{\alpha_1} \quad \text{et} \quad \text{temps}_{\text{Kara}}(n) \approx C_{\text{Kara}} n^{\alpha_2}.$$

  Ces résultats sont-ils en accord avec la théorie ?

4 Bonus

Si vous avez fini et réussi tout le reste, bravo ! Voici une liste de questions supplémentaires pour vous occupez jusqu'à la fin du TP !

- Utiliser la partie 1 pour implémenter une fonction qui calcule le pgcd et le ppcm de $a_1, \dots, a_k$ avec $k \in \mathbb{N}^*$. Améliorez votre algorithme de façon à ce qu'il retourne également le nombre d'étapes et des entiers $(u_1, \dots, u_k)$ tels que $\text{pgcd}(a_1, \dots, a_k) = a_1u_1 + \cdots + a_ku_k$. Estimez le nombre d'opérations de l'algorithme et essayez de déterminer le pire cas.
- Implémenter un algorithme qui prend en argument deux entiers a et b et renvoie (q, r) où q et r sont respectivement le quotient et le reste de la division euclidienne de a par b. Étudiez la complexité de votre algorithme ainsi que celle de l'algorithme d'Euclide théoriquement et expérimentalement. Renseignez-vous sur les options alternatives pour coder une telle division euclidienne.
- Quelle propriété fondamentale de l'anneau $\mathbf{Z}$ est nécessaire pour implémenter un algorithme d'Euclide ? Implémenter un calcul de pgcd dans $\mathbf{Z}[i]$.
- Pour a et $b \neq 0$ deux entiers, soient $(q_0, \dots, q_N)$ la liste des quotients obtenus à chaque étape de l'algorithme d'Euclide. Écrire une fonction qui prend en argument a et b et qui renvoie la fraction continue

$$q_0 + \cfrac{1}{q_1 + \cfrac{1}{q_2 + \cfrac{1}{\ddots + \cfrac{1}{q_N}}}}.$$

  Que conjecturez-vous ? Le démontrer.
- Écrire une fonction qui prend en argument un triplet d'entiers (a, b, c) et qui renvoie l'ensemble des solutions (x, y) entières de l'équation diophantienne $ax + by = c$. Implémenter une fonction qui les représente graphiquement.
- L'anneau des polynômes en plusieurs variables est-il euclidien ? Factoriel ? Proposer un algorithme pour calculer le pgcd de polynômes en plusieurs variables en se ramenant à l'algorithme d'Euclide en une variable. On pourra commencer par le cas de deux variables.