Université de Bordeaux
Mathématiques

4TMA801U – Master
Année 2019–2020

Devoir Surveillé, 11 mars 2020

Durée 1h30.

Certaines questions demandent d'écrire du code Sage. Écrivez ce code à la main sur votre copie, en veillant au respect de la syntaxe et de l'indentation.

Exercice 1 – [Algorithme d'Euclide classique]

Soit K un corps. On rappelle l'algorithme d'Euclide étendu classique pour le calcul du pgcd dans $K[X]$:

Algorithme 1. Algorithme d'Euclide étendu classique

Entrées: A, $B \in K[X]$
Sorties: $\gcd(A,B)$ et U, $V \in K[X]$ tels que $UA + VB = \gcd(A,B)$

$U_0 \leftarrow 1$, $V_0 \leftarrow 0$, $R_0 \leftarrow A$
$U_1 \leftarrow 0$, $V_1 \leftarrow 1$, $R_1 \leftarrow B$
$i \leftarrow 1$
tantque $R_i \neq 0$ **faire**
    $(Q_i, R_{i+1}) \leftarrow$ quotient et reste de la division de R_{i-1} par R_i
    $U_{i+1} \leftarrow U_{i-1} - Q_i U_i$, $V_{i+1} \leftarrow V_{i-1} - Q_i V_i$
    $i \leftarrow i + 1$
Retourner le dernier R_i non nul ainsi que les U_i et V_i correspondants

Soient $A, B \in K[X]$ tels que $A \neq 0$ et $\deg(B) \leqslant \deg(A)$. On note $n = \deg(A)$.

1) Écrire une fonction Sage nommée `EuclideClassique` qui suit l'algorithme 1. On pourra utiliser la fonction `divmod(Y,Z)` qui renvoie une paire contenant le quotient et le reste de la division euclidienne de Y par Z.

2) Rappeler la complexité algébrique de l'algorithme 1 en fonction de n. On ne demande pas de la recalculer.

3) Notons t la valeur de i lors du dernier passage dans la boucle (on a donc $R_{t+1} = 0$). Montrer que pour tout $i \in [\![0, t]\!]$,

$$\begin{pmatrix} U_i & V_i \\ U_{i+1} & V_{i+1} \end{pmatrix} \begin{pmatrix} A \\ B \end{pmatrix} = \begin{pmatrix} R_i \\ R_{i+1} \end{pmatrix} \quad \text{et}$$

$$\begin{pmatrix} U_i & V_i \\ U_{i+1} & V_{i+1} \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ 1 & -Q_i \end{pmatrix} \cdots \begin{pmatrix} 0 & 1 \\ 1 & -Q_1 \end{pmatrix}.$$

4) Montrer que la suite $(\deg(R_i))_{1 \leqslant i \leqslant t+1}$ est strictement décroissante (par convention, le degré du polynôme nul est $-\infty$). En déduire qu'il existe un unique entier $j \in [\![0, t]\!]$ tel que

$$\deg(R_j) \geqslant \frac{n}{2} \quad \text{et} \quad \deg(R_{j+1}) < \frac{n}{2}.$$

Exercice 2 – [Algorithme d'Euclide rapide]

Dans cet exercice, on suppose qu'il existe un algorithme `dpgcd`, appelé *demi-pgcd*, qui calcule directement à partir de A et B la matrice

$$M_{\text{dpgcd}}(A,B) = \begin{pmatrix} U_j & V_j \\ U_{j+1} & V_{j+1} \end{pmatrix},$$

où j est l'indice défini à la question 4 de l'exercice 1. On note également

$$M(A,B) = \begin{pmatrix} U_t & V_t \\ U_{t+1} & V_{t+1} \end{pmatrix}.$$

la matrice obtenue à la dernière étape.

L'algorithme récursif suivant calcule $M(A,B)$ à partir de A et B en utilisant l'algorithme du demi-pgcd.

Algorithme 2. Algorithme d'Euclide rapide

Entrées: A, $B \in K[X]$
Sorties: $M(A,B)$

1: $M_1 \leftarrow M_{\text{dpgcd}}(A,B)$ par un appel à `dpgcd`
2: Calculer R_j et R_{j+1} par l'égalité

$$\begin{pmatrix} R_j \\ R_{j+1} \end{pmatrix} = M_1 \begin{pmatrix} A \\ B \end{pmatrix}$$

3: Si $R_{j+1} = 0$, retourner M_1
4: $(Q, R_{j+2}) \leftarrow$ quotient et reste de la division euclidienne de R_j par R_{j+1}
5: $M_2 \leftarrow \begin{pmatrix} 0 & 1 \\ 1 & -Q \end{pmatrix}$
6: Si $R_{j+2} = 0$, retourner $M_2 M_1$
7: $M_3 \leftarrow M(R_{j+1}, R_{j+2})$ par un appel récursif
8: Retourner $M_3 M_2 M_1$.

---

1) Comment calculer $\text{pgcd}(A,B)$ à partir de $M(A,B)$?

2) Écrire une fonction Sage nommée `EuclideRapide` qui suit l'algorithme 2, en supposant qu'on dispose d'une fonction `dpgcd(A,B)` qui calcule $M_{\text{dpgcd}}(A,B)$.

Indication. Si `M1` est la matrice calculée à la ligne 1, alors l'appel

```
M1.parent()([Z1,Z2,Z3,Z4])
```

construit la matrice $\begin{pmatrix} Z_1 & Z_2 \\ Z_3 & Z_4 \end{pmatrix}$.

3) Montrer que les degrés de R_{j+1} et R_{j+2} sont strictement inférieurs à $n/2$.

4) Soit $E(n)$ le coût de l'algorithme d'Euclide rapide pour des entrées de degré au plus n. Établir une relation de récurrence reliant $E(n)$, $E(\lfloor n/2 \rfloor)$, le coût $H(n)$ de l'algorithme `dpgcd`, et le coût $T(n)$ de la multiplication. On admet que le coût de la division euclidienne est de $O(T(n))$.

5) On fait les hypothèses que $T(n) = O(H(n))$ et que $H(kn) \geqslant kH(n)$, pour tout entier k. Montrer que $E(n) = O(H(n))$.

6) Montrer que l'algorithme 2 est correct, c'est à dire qu'il calcule bien $M(A,B)$.

Remarque. On peut écrire un algorithme de demi-pgcd, basé sur une stratégie de type « diviser pour régner », en $O(T(n)\log(n))$. On obtient ainsi un algorithme de calcul de pgcd en $\widetilde{O}(n)$.