TD/TP Algorithmes élémentaires

Exercice 1 – Multiplication dans $\mathbf{Z}[X]$

Estimer le coût du calcul de la somme et du produit de deux polynômes $P, Q \in \mathbf{Z}[X]$ en fonction de leurs degrés et d'une borne uniforme M de la valeur absolue de leurs coefficients.

Exercice 2 – Algorithme de Karatsuba pour les polynômes

Soit A un anneau.

1. Calculer le coût (en nombre d'opérations dans A) de l'addition de deux polynômes de degré m et n à coefficients dans A.
2. Idem pour la multiplication de deux polynômes de degré m et n.
3. Soient $P, Q \in A[X]$ deux polynômes de degré inférieur ou égal à $2n-1$. En écrivant $P(X) = P_1(X) + P_2(X)X^n$ et $Q(X) = Q_1(X) + Q_2(X)X^n$ avec $P_1, P_2, Q_1, Q_2 \in A[X]$ de degrés inférieurs ou égaux à $n-1$, exprimer le produit PQ en fonction des polynômes P_1, P_2, Q_1 et Q_2.
4. Soit $C(n)$ le coût du calcul du produit de deux polynômes de degré au plus n, en nombre de multiplications dans A. Montrer qu'il existe un algorithme de multiplication avec un coût vérifiant $C(2n-1) \leq 3C(n)$.
5. En déduire un algorithme permettant de calculer le produit de deux polynômes de degré au plus n avec un coût de $O(n^{\log_2(3)})$ opérations.

Exercice 3 – Algorithme de Karatsuba pour les entiers

Soit $n = 2m$ un entier pair non nul et soient a, b deux entiers s'écrivant avec au plus n bits. Vérifier que l'algorithme suivant calcule le produit ab, puis déterminer son coût en fonction de n. Est-il plus efficace que l'algorithme classique ?

kara(a,b)
Ecrire a sous la forme $\alpha 2^m + \tilde{\alpha}$;
Ecrire b sous la forme $\beta 2^m + \tilde{\beta}$;
Poser $x := (\alpha + \tilde{\alpha}) \times (\beta + \tilde{\beta})$;
Poser $y := \alpha\beta$;
Poser $z := \tilde{\alpha}\tilde{\beta}$;
Renvoyer $y2^n + (x - y - z)2^m + z$.

Exercice 4 – Exponentiation rapide

Soient A un anneau, a un élément de A et $n \geq 1$ un entier. On rappelle l'algorithme d'exponention rapide permettant de calculer a^n :

expr(a,n)
Si $n = 1$, renvoyer a ;
Si n pair, renvoyer expr$(a, \frac{n}{2})^2$;
Si n impair, renvoyer $a \times$ expr$(a, \frac{n-1}{2})^2$.

1. Rappeler le coût de cet algorithme, en nombre de multiplications dans A.
2. Étant donnés des entiers $a \in \mathbf{Z}$ et $N \geq 1$, expliquer comment utiliser Sage pour calculer $a^n \mod N$ sans avoir à calculer a^n. Tester pour de grandes valeurs de n.

3. On suppose maintenant $A = \mathbf{Z}$. Estimer le coût $C(a, n)$ du calcul de a^n *en termes d'opérations élémentaires (sur les bits)*. Vérifier que l'essentiel du calcul est contenu dans la dernière multiplication, de sorte que l'on ne gagne pas grand chose à utiliser l'exponentiation rapide dans ce cas.

Exercice 5 – Suite de Fibonacci

On rappelle que la suite de Fibonacci $(F_n)_{n\geq 0}$ est définie par les relations suivantes : $F_0 = 0$, $F_1 = 1$ et, pour tout $n \geq 0$, $F_{n+2} := F_n + F_{n+1}$.

1. Exprimer F_n en fonction de n.
2. Ecrire un algorithme (récursif) naïf de calcul des nombres de Fibonacci et estimer son coût.
3. Ecrire un algorithme de calcul du n-ième nombre de Fibonacci ayant un coût en $O(n)$ opérations dans $\mathbf{Z}$.
   Indication : Penser à calculer la paire (F_{n-1}, F_n).
4. Ecrire un algorithme de calcul du n-ième nombre de Fibonacci ayant un coût en $O(\log_2(n))$ opérations dans $\mathbf{Z}$. Que dire du coût en nombre d'opérations sur les bits ?
   Indication : Penser à l'écriture matricielle, comme dans l'algorithme d'Euclide étendu.
5. Tester ces algorithmes avec le logiciel Sage.

Exercice 6 – Algorithme d'Euclide binaire

On se donne deux entiers $u, v \geq 1$, avec v impair. Vérifier que l'algorithme suivant donne le pgcd de u et v, et estimer son coût :

euclidebin(u,v)
Si $v = 1$ ou $u = v$, renvoyer v ;
Si u pair, renvoyer euclidebin$(u/2, v)$;
Si u impair $> v$, renvoyer euclidebin$(\frac{u-v}{2}, v)$;
Si u impair $< v$, renvoyer euclidebin$(\frac{v-u}{2}, u)$;

1. Implémenter les algorithmes d'Euclide classique et binaire binaire.
2. Comparer entre eux les temps d'exécution de ces deux algorithmes, puis les comparer au temps d'exécution de la commande incluse dans Sage.

Exercice 7 – Nombre de pas dans l'algorithme d'Euclide

Si m et n sont deux entiers ≥ 1, on note $t(m, n)$ le nombre de divisions euclidiennes nécessaires pour calculer le pgcd de m et n (on commence par diviser m par n et on s'arrête lorsque le reste est nul, ainsi $t(2, 1) = 1$ et $t(1, 2) = 2$).

1. Écrire une procédure calculant $t(m, n)$.
2. Si $(F_n)_{n\geq 0}$ est la suite de Fibonacci définie à l'exercice 5, vérifier $t(F_{n+2}, F_{n+1}) = n$ pour tout $1 \leq n \leq 50$.
3. Écrire une procédure qui permet de vérifier que le maximum de $t(m, n)$ pour des entiers $1 \leq m, n \leq N$ est obtenu pour le couple (F_r, F_{r+1}) où r est le plus grand entier tel que $F_{r+1} \leq N$.
4. On considère les fonctions T et τ définies par

$$T(n) = \frac{1}{n} \sum_{1\leq k\leq n} t(k, n) \qquad \tau(n) = \frac{1}{\varphi(n)} \sum_{\substack{1\leq k\leq n \\ (k,n)=1}} t(k, n)$$

   Tracer les graphes de T et τ sur un même graphique. Que constate-t-on ? Vérifier numériquement ou graphiquement qu'il existe des constantes $a, b > 0$ pour lesquelles $a \ln(n) + b$ est une bonne approximation de $\tau(n)$.

On considère la variante de l'algorithme d'Euclide dans laquelle le reste d'une division euclidienne $m = qn + r$ vérifie $|r| \leq n/2$. On note $t'(m, n)$ le nombre de pas nécessaires pour calculer le pgcd de m et n avec cette variante.

5. Déterminer $t'(F_{n+2}, F_{n+1})$.
6. Reprendre les questions 1, 3 et 4 pour la fonction t' (on notera T' et τ' les fonctions correspondantes).
7. Tracer sur un même graphique les fonctions τ et τ'. Que constate-t-on ?

Exercice 8 – Applications de l'algorithme d'Euclide étendu

Soient $a, b \geq 1$ deux entiers premiers entre eux.

1. Soit $(x, y) \in \mathbf{Z}/a\mathbf{Z} \times \mathbf{Z}/b\mathbf{Z}$. Montrer qu'il existe un unique élément $z \in \mathbf{Z}/ab\mathbf{Z}$ vérifiant $z \equiv x \bmod a$ et $z \equiv y \bmod b$, puis estimer le coût du calcul de z en fonction des tailles de a et b.
2. Montrer que toute fraction rationnelle de la forme $\frac{N}{ab}$ s'écrit de manière unique sous la forme $k + \frac{x}{a} + \frac{y}{b}$ avec $k \in \mathbf{Z}$ et $x, y \in \mathbf{N}$ vérifiant $0 \leq x < a$ et $0 \leq y < b$. Estimer le coût du calcul de k, x et y en fonction des tailles respectives de N, a et b.
3. Implémenter ces algorithmes.

Exercice 9 – Intégration numérique

1. Calculer numériquement avec Sage, avec 128 bits de précision, l'intégrale

$$I = \frac{1}{2\pi} \int_0^{2\pi} \sqrt{1 - 0,36 \sin^2(\theta)} d\theta.$$

2. Écrire une fonction qui prend en entrée un entier N et calcule une valeur approchée I_N de I en appliquant la règle des trapèzes avec une subdivision à N points.
3. Calculer $|I_N - I|$ pour quelques valeurs de N.
4. À partir de quelle valeur de N obtient-on la même valeur que I, à la précision donnée ?
5. Estimer expérimentalement l'ordre de grandeur de l'erreur $|I_N - I|$. La convergence est-elle typique d'une intégrale sur un segment ?

Méthode de Newton et applications

Exercice 10 – Méthode de Newton

Soit I un intervalle de $\mathbf{R}$ et $f : I \to \mathbf{R}$ une fonction de classe $\mathcal{C}^\infty$. Pour tout élément $x \in I$ n'annulant pas la dérivée f' de f, on pose $g(x) := x - \frac{f(x)}{f'(x)}$.

1. Etudier la régularité de g sur son ensemble de définition.
2. Comparer l'ensemble des zéros de f et l'ensemble des points fixes de g.
3. Supposons que a soit un zéro de f en lequel g est bien définie.
   (a) Calculer $g'(a)$.
   (b) Démontrer l'existence d'un voisinage U de a tel que pour tout élément $x_0 \in U$, la suite $(x_{n+1} := g(x_n))_{n \geq 0}$ est bien définie et converge vers a.
   (c) Supposons qu'il existe deux réels M et r strictement positifs tels que l'on ait $2Mr < 1$ et $|g''(x)| \leq M$ pour tout $x \in]a-r, a+r[$. Montrer que la suite $(x_n)_{n \geq 0}$ définie ci-avant vérifie $|x_n - a| \leq \left(\frac{1}{2}\right)^{2^n - 1} |x_0 - a|$ pour tout entier $n \geq 0$.

4. Que se passe-t-il lorsque a est un zéro commun à f et f' ?
5. Supposons que f est convexe et strictement croissante et considérons $x_0 \in I$ vérifiant $f(x_0) > 0$. Montrer que si f s'annule sur I, alors la suite $(x_n)_{n\geq 0}$ définie comme ci-dessus est bien définie et converge vers le point d'annulation de f dans I.

Exercice 11 – Le cas des polynômes

Soit $P \in \mathbf{R}[X]$ un polynôme non constant de coefficient dominant strictement positif. On suppose que P est scindé dans $\mathbf{R}$.

1. Montrer qu'il existe un réel $A > 0$ tel que pour tout $x_0 > A$, la suite de Newton $(x_n)_{n\geq 0}$ associée à x_0 et à P est bien définie et converge vers la plus grande racine réelle de P.
2. Expliciter une valeur possible de A à l'aide des coefficients de P.
3. Implémenter cet algorithme dans Sage, en prenant comme argument P et la précision en nombre de bits.

Exercice 12 – Le cas de l'inverse

Soit a un réel strictement positif dont on souhaite approcher l'inverse.

1. A quelle fonction f peut-on appliquer la méthode de Newton pour approcher a^{-1} ?
2. Choisissons $x_0 > 0$. Montrer que la suite de Newton associée à f et à x_0 est donnée par la relation de récurrence suivante : pour tout $n \geq 0$, $x_{n+1} = x_n(2 - ax_n)$.
3. Pour tout $n \geq 0$, posons $\varepsilon_n := ax_n - 1$.
   (a) Exprimer ε_{n+1} en fonction de ε_n.
   (b) En déduire que si $|\varepsilon_0| < 1$, alors la suite $(x_n)_{n\geq 0}$ converge vers a^{-1}.
4. Posons $I := [\frac{1}{2}, 1]$. Déterminer une paire de réels (λ, μ) permettant de minimiser la quantité $\sup_{a\in I}|\lambda - \mu a - a^{-1}|$.

Remarque : L'intérêt de la manœuvre est d'accélérer la convergence de la suite $(x_n)_{n\geq 0}$ lorsque a appartient à I en choisissant $x_0 := \lambda - \mu a$.

Exercice 13 – Le cas de la racine carrée

Soient a et x_0 deux réels strictement positifs.

1. Pour tout $n \geq 0$, on pose $x_{n+1} := \frac{1}{2}(x_n + \frac{a}{x_n})$. Montrer que la suite $(x_n)_{n\geq 0}$ est bien définie et converge vers $\sqrt{a}$.
2. Pour tout $n \geq 0$, posons $y_{n+1} := \frac{y_n}{2}(3 - ay_n^2)$ avec $y_0 := x_0$. Montrer que la suite $(y_n)_{n\geq 0}$ est bien définie et converge vers $\sqrt{a}^{-1}$.
3. Comparer les deux méthodes ainsi obtenues pour approcher $\sqrt{a}$: laquelle semble la plus efficace en temps de calcul ?