Ce texte a pour thème différents algorithmes de calcul de PGCD, utilisés pour améliorer l'efficacité de l'algorithme d'Euclide.

1 Le PGCD binaire pour les entiers

Il s'agit de calculer le PGCD de 2 entiers a et b de la manière la plus efficace possible sur une machine calculant en base 2. Le principe est le suivant :

- On rend a et b positifs
- on extrait la partie impaire de a et b par décalage vers la droite
- on effectue une première division euclidienne du plus grand par le plus petit, on extrait la partie impaire du reste
- on utilise la formule PGCD(a,b)=PGCD(min(a,b),abs$(a-b)$) et on extrait la partie impaire de la valeur absolue de la différence

L'algorithme standard d'Euclide (utilisant la formule PGCD(a,b)=PGCD$(b,a \pmod b)$) est plus lent que l'algorithme de PGCD binaire car on peut montrer (cf. Knuth section 4.5.3 Theorem E) que le quotient de l'algorithme d'Euclide est 1, 2 ou 3 dans 2/3 des cas, il est donc plus efficace de faire directement des soustractions. Toutefois, la première division de l'algorithme d'Euclide n'a aucune raison d'avoir un petit quotient et est donc effectuée.

Les deux algorithmes ont une complexité en $O(N^2)$ (N étant le nombre de digits du plus grand des deux entiers), mais on observe qu'en pratique le PGCD binaire est plus rapide que l'algorithme d'Euclide.

2 Les polynomes en une variable

Lorsqu'on travaille avec des polynomes à coefficients dans un corps fini, l'algorithme d'Euclide est bien adapté au calcul du PGCD de deux polynomes, sa complexité est en $O(n^2)$ si n désigne le plus grand des degrés des deux polynomes. Ce n'est plus le cas lorsque le corps de base est infini, par exemple les rationnels, car les opérations de base ne se font plus en temps constant et ce même si on s'apercoit in fine que les deux polynomes sont premiers entre eux, on utilise alors des algorithmes plus efficaces.

2.1 Les algorithmes de type sous-résultant

Cet algorithme peut être utilisé lorsque les coefficients sont dans un anneau euclidien contenu dans un corps de fractions, par exemple les entiers dans les rationnels, ou des polynomes (en d'autres variables) dans les fractions rationnelles, ... (plus généralement ces algorithmes s'appliquent si les coefficients sont dans un UFD - Unique Factorization Domain). On part de l'observation qu'on ne change pas le PGCD de P et Q si on multiplie P par un élément de l'anneau. Au lieu de faire une division euclidienne de P par Q, on effectue une pseudo-division de P par Q, c'est-à-dire qu'on multiplie P par le coefficient dominant de Q à la puissance différence des degrés plus un. Ceci assure que le reste est encore à coefficients dans l'anneau.

Cet algorithme a toutefois l'inconvénient de faire croitre de manière trop importante les coefficients des restes intermédiaires, on peut améliorer cela en divisant les restes par le PGCD des coefficients du reste. On obtient alors des restes intermédiaires de tailles raisonnables. L'inconvénient de cet algorithme est qu'il faut calculer le PGCD des coefficients, s'il s'agit eux-memes de polynomes, on a un appel récursif au PGCD de 2 polynomes qui s'avère couteux.

Il existe un algorithme qui évite ce dernier problème, au prix de coefficients des restes intermédiaires pas forcément premiers entre eux mais dont la taille croit de manière raisonnable, il s'agit de l'algorithme du sous-résultant, il effectue à chaque étape une pseudo-division et calcule un coefficient à priori (sans calculer le PGCD des coefficients du reste) par lequel on sait que les coefficients du reste seront divisibles.

Algorithme du sous-résultant Arguments : 2 polynômes P et Q primitifs. Valeur de retour : le pgcd de P et Q.

Pour calculer le coefficient à priori, on utilise 2 variables auxiliaires g et h initialisées a 1.

Boucle à effectuer tant que Q est non nul :

- on note δ =degre(P)-degre(Q) et q le coefficient dominant de Q
- on effectue la division euclidienne (sans fraction) de $q^{\delta+1}P$ par Q, soit R le reste
- Si R est constant, on sort de l'algorithme en renvoyant 1 comme pgcd
- on recopie Q dans P puis $R/(gh^{\delta})$ dans Q
- on recopie q dans g et $h^{1-\delta}q^{\delta}$ dans h.

Si on sort normalement de la boucle, Q est nul, on renvoie donc la partie primitive de P qui est le pgcd cherché.

Pour la justification, cf. Knuth.

2.2 Méthodes modulaires (cas des coefficients entiers)

Les méthodes ci-dessus améliorent l'algorithme d'Euclide, mais restent peu efficaces si les coefficients sont entiers, surtout lorsque le PGCD des 2 polynomes est 1 ou de petit degré. Les méthodes modulaires consistent à déterminer le PGCD de 2 polynomes à coefficients entiers à partir du PGCD de ces 2 polynomes vu comme à coefficients dans un $\mathbb{Z}/p\mathbb{Z}$ pour un ou plusieurs p premiers (l'intéret étant de calculer des PGCD de polynomes à coefficients dans un corps fini).

On montre d'abord que si deux polynomes sont primitifs, alors en multipliant un PGCD de P et Q dans $\mathbb{Q}[X]$ par le PGCD des dénominateurs, puis en rendant ce polynome primitif, on obtient un polynome divisant P et Q dans $\mathbb{Z}[X]$, et c'est ce polynome qu'on appelle "le" PGCD G de P et Q. En réduisant modulo p, on obtient un polynome qui divise P et Q modulo p, donc qui divise le pgcd de P et de Q modulo p. Si on chosit p ne divisant pas le PGCD des coefficients dominants de P et Q, alors p ne divise pas le coefficient dominant de G donc le degré de G ne change pas après réduction modulo p. On en déduit que le degré du pgcd de P et Q est inférieur ou égal au degré du pgcd de P et Q modulo p lorsque p ne divise pas le PGCD g des coefficients dominants de P et Q. En pratique il est presque toujours égal (il suffit que p ne divise pas les coefficients dominants des restes de l'algorithme d'Euclide avec pseudo-division euclidiennes).

Donc si on trouve P et Q premiers entre eux modulo p (pour p ne divisant pas g), alors P et Q sont premiers entre eux. Reste à étudier le cas de deux polynomes non premiers entre eux. L'idée consiste à reconstruire le pgcd de P et Q à partir de pgcd modulaire(s). Ce qui parait le plus simple est d'utiliser la représentation symétrique des modulo p et de choisir un p suffisamment grand pour que le pgcd modulaire de P et Q écrit en représentation symétrique ait forcément les coefficients du pgcd non modulaire de P et Q. On choisirait p en utilisant la borne de Landau-Mignotte des coefficients de G. Malheureusement, ce choix donnerait un p beaucoup trop grand. On préfère donc choisir plusieurs petits nombres premiers et reconstruire le pgcd non modulaire en appliquant le théorème des restes chinois.

Le dernier problème de reconstruction vient du fait que les pgcd modulaires sont toujours rendu unitaires (coefficient dominant égal à 1), alors que G n'a aucune raison de l'être, et pire on ne connait pas le coefficient dominant de G. Mais on sait qu'il divise g le pgcd des coefficients dominants de P et Q. On reconstruit alors non pas G mais le multiple entier de G dont le coefficient dominant est g.

Donc l'algorithme est le suivant :

- On rend P et Q primitifs par division par leur contenu
- Calcul de g
- Initialisation produit des p_i à 1, pgcd reconstruit H à rien
- Début de boucle
- Recherche du PGCD M de P et Q modulo un premiers p ne divisant pas g
- Si M est de degré nul, on renvoie 1 (plus précisément le pgcd des contenu de P et Q initiaux)
- Si M est de degré supérieur strict au pgcd reconstruit (s'il existe) on passe au premier suivant (retour à début de boucle)
- Si M est de degré inférieur strict au pgcd reconstruit (s'il existe), on réinitialise le produit des p_i à p et le pgcd reconstruit à M
- Si M est de même degré, on multiplie M par g et on remplace H par le polynome obtenu par restes chinois coefficient par coefficient du coefficient de H modulo le produit des p_i et du coefficient de M modulo p, on multiplie le produit des p_i par p, on écrit les coefficients en représentation symétrique (on utilise l'unique représentant dans l'intervalle $]-n/2, n/2]$ de l'entier modulo n)
- on teste si la partie primitive de H divise P et Q, si oui on renvoie la partie primitive de H (multipliée par le pgcd des contenus des P et Q originaux). En pratique, pour éviter de tester des H trop tot, on ne fait ce dernier test que si H n'a pas changé suite à l'ajout d'un nouveau nombre premier.

L'algorithme s'arrête forcément parce que le produit des nombres premiers dépassera la borne de Landau-Mignotte des coefficients du PGCD de P et Q multiplié par $2g$.

2.3 Le PGCD de plusieurs polynomes

Pour calculer le PGCD de plusieurs polynomes $P_1, ..., P_n$, plutot que de calculer le PGCD 2 à 2 successivement, on peut choisir des entiers aléatoires $k_2, ..., k_n$ et calculer le pgcd de P_1 avec $k_2P_2 + ... + k_nP_n$ et tester s'il divise P_2, ..., P_n. Si oui, c'est le PGCD cherché, sinon on change d'entiers aléatoires et on recommence.

3 Suggestions de développement.

- Présenter un algorithme de calcul de PGCD dans $\mathbb{Z}[i]$
- Présenter un algorithme effectif de calcul de l'identité de Bézout soit pour des entiers (par exemple l'algorithme multi-pas de Lehmer, Knuth section 4.5.2 algorithm L ou Cohen) soit pour des polynomes (par exemple méthode modulaire ou sous-résultant).
- Justifier l'utilisation de l'algorithme du PGCD binaire (théoriquement et en montrant sur des exemples aléatoires la proportion de quotient 1, 2 et 3)
- Illustration de l'intérêt des algorithmes de calcul de PGCD de polynomes présentés
- Autres algorithmes de calcul de PGCD.
- Applications du PGCD et de l'identité de Bézout (dans le cas des polynomes).
- Utilisation de méthodes modulaires pour des algorithmes sur les polynomes à coefficients entiers (par exemple Bézout, factorisation)
- Complexité d'un des algorithmes présenté : justification théorique et illustration par des exemples. Par exemple complexité de l'algorithme du PGCD modulaire (en $O(n^3)$ où n majore la taille des coefficients et les degrés, cf. J. Von zur Gathen and J. Gerhard)