Computing with Polynomials Given By Black Boxes for Their Evaluations: Greatest Common Divisors, Factorization, Separation of Numerators and Denominators [1]
de E. Kaltofen et B. Trager

Laurent Boyer

12 janvier 2007

1 Résumé de l'article

1.1 Représentation des polynômes par *boite noire*

Les auteurs utilisent une méthode de représentation des polynômes par *boite noire*. Cette représentation correspond à la notion intuitive en informatique de polynôme représenté par un programme qu'on peut interroger pour certaines valeurs d'entrée mais dont la méthode de calcul est inconnue. Formellement, on accède aux polynômes par appel à un oracle avec certaines valeurs en entrée : une boite noire est une fonction qui prend en entrée n éléments du corps K et retourne la valeur du polynôme en ce point (cf figure 1).

1.2 Résultats algorithmiques

Les auteurs fournissent des algorithmes permettant dans ce formalisme de répondre au problème de factorisation d'un polynôme, de trouver le PGCD d'une famille de polynômes, ainsi qu'un algorithme pour calculer le numérateur et le dénominateur d'une fraction rationnelle.

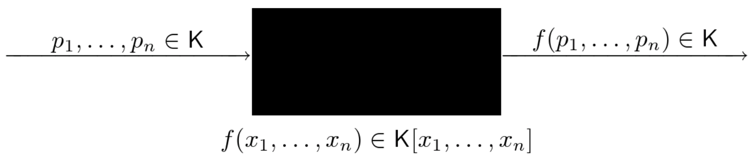


FIG. 1 – polynôme calculé par boite noire

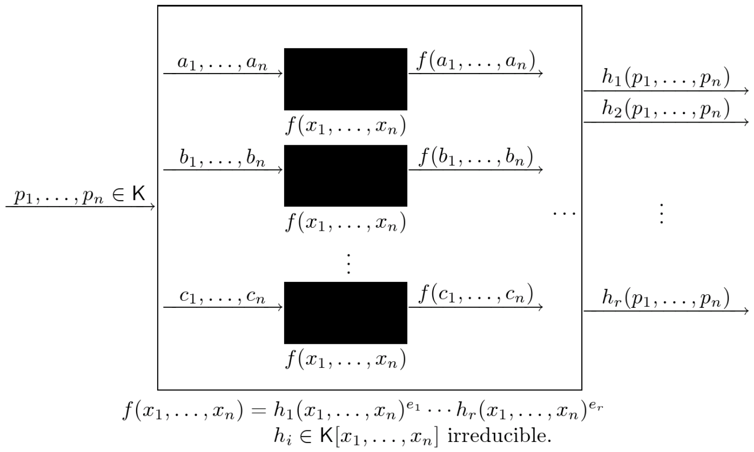


FIG. 2 – La procédure produite par l'algorithme de factorisation

Il faut faire attention à la notion d'algorithme dans le formalisme utilisé. Par exemple, un algorithme qui calcule les facteurs premier d'un polynôme est un algorithme qui prend en entrée une boite noire pour ce polynôme et qui retourne une procédure (qui peut être vue comme une boite noire) qui calcule les différents facteurs en utilisant les boites noires d'entrée (cf figure 2).

Le principe est le même dans les autres cas.

1.2.1 Algorithme de factorisation

Je vais présenter rapidement les grandes lignes du fonctionnement de l'algorithme de factorisation.
En écrivant la décomposition en facteurs premiers d'un polynôme

$$f(x_1, ..., x_n) = \prod_{i=1}^{r} h_i(x_1, ..., x_n)^{e_i}$$

on veut un algorithme effectuant le travail suivant :
- entrée : une boite noire calculant un polynôme $f(x_1, ..., x_n) \in K[x_1, ..., x_n]$, une probabilité d'erreur $\epsilon << 1$
- sortie (avec une probabilité au moins $1 - \epsilon$) : les r entiers e_i, $i \in [1..r]$, et une procédure (utilisant la boite noire d'entrée), qui peut être vue comme une boite noire, évaluant les fonctions $h_1, h_2, ...h_r$.

Par ailleurs pour que l'algorithme fonctionne on impose une hypothèse importante : on a besoin de disposer d'un algorithme pour factoriser un polynôme à une seule variable sur K. Il faut aussi signaler que le corps K est de caractéristique nulle.

Les étapes principales de l'algorithme sont les suivantes :

- On détermine d'abord *le motif des degrés des facteurs premiers de f* (c'est à dire r et les e_i).
  - on choisit aléatoirement des éléments de K : $a_2, ...a_n, b_1, ...b_n, c_3, ...c_n$.
  - on calcule par interpolation le polynôme $f_2(X_1, X_2) := f(X_1+c_1X_2+b_1, a_2X_1+X_2+b_2, ..., a_nX_1+c_nX_2+b_n)$ Si les variables aléatoires sont choisies dans un ensemble assez grand, ce polynôme a comme propriété que son motif des degrés des facteurs est avec grande probabilité le même que celui de f.
  - on factorise f_2 (on se ramène polynomialement au cas à une seule variable)
  - on peut retourner les e_i
- Il faut maintenant expliquer le fonctionnement de la boite noire qui retourne les facteurs évalués.
  - on calcule le polynôme $\overline{f}(X_1, Y) := f(X_1 + b_1, Y(p_2 - a_2(p_1 - b_1) - b_2) + a_2x_1, ..., Y(p_n - a_n(p_1 - b_1) - b_n) + a_nX_1 + b_n)$
  - on obtient efficacement les facteurs $\overline{g_i}$ de $\overline{f}(X_1, Y) = \prod \overline{g_i}(X_1, Y)^{e_i}$
  - On peut retourner les valeurs en évaluant les facteurs en $\{p_1 - b_1, 1\}$.

Quelques remarques

- Les similitudes entre la factorisation de f_2 et celle de f lorsque l'ensemble où on choisi les éléments est assez grand par rapport au degré de f sont expliqués par les résultats précédents des auteurs (en particulier voir [2]).
- Le calcul d'une factorisation de f_2 permet de garantir que la décomposition de f que l'on utilise à chaque appel à la boite noire est la même (Sinon il y a un risque d'obtenir des résultats à un facteur scalaire près, différent à chaque appel à la boite noire...).
  En effet, si on garde la factorisation de $f_2 = \prod g_{2,i}(X_1, X_2)^{e_i}$ *en dur* dans la boite noire, on peut lorqu'on effectue l'appel à la boite noire imposer que $g_{2,i}(X_1, 0) = \overline{g_i}(X_1, 0)$ ce qui fixe ces facteurs scalaires. C'est d'ailleurs l'intérêt de l'introduction de la variable supplémentaire Y. Le même type de remarque est valable pour les autres algorithmes.
- On a ici éludé deux aspects discutés dans l'article : la nécessité de connaître le degré du polynôme pour les interpolations et la raison de la necessité d'une projection sur $K[X_1, X_2]$ et pas directement sur $K[X]$.
- La réponse retournée par le programme est juste sauf si les factorisations ne sont pas conservées par la projection sur $K[X_1, X_2]$. Il suffit de rajouter deux tests pour garantir que la sortie est toujours juste.

Avec les quelques remarques précédentes, on a donc un algorithme polynomial en le nombre de variable et le degré de f, qui effectue un nombre polynomial d'appels à la boite noire d'entrée, et qui, si l'ensemble dans lesquel on choisit les variables aléatoires est assez grand (en fonction du degré de f), a une probabilité de terminer supérieure à $1 - \epsilon$

1.2.2 Calcul du PGCD

On veut un programme construisant à partir d'une famille de boites noires une boite noire qui prend en entrée une assignation des variables et retourne la valeur du pgcd des polynômes calculés par les boites noires précédentes en cette assignation.

Sur cet algorithme les auteurs de l'article passe très rapidement. Le principe de l'algorithme utilise les mêmes ingrédients de base que le précédent. En résumé les principales étapes sont à partir d'une famille $f_1, ...f_r$ de polynômes en entrée :

- on tire au hasard des éléments du corps $a_2, ...a_n, b_2, ...b_n, c_2, ...c_r$
- on calcule les polynômes projections sur deux variables :

$$\overline{f_1}(X, Y) \text{ et } \overline{f_0}(X, Y) = \sum_{i=2}^{r} c_i \overline{f_i}(X, Y)$$

- on calcule le pgcd de ces deux polynômes
- on retourne la valeur du pgcd calculé en le point voulu

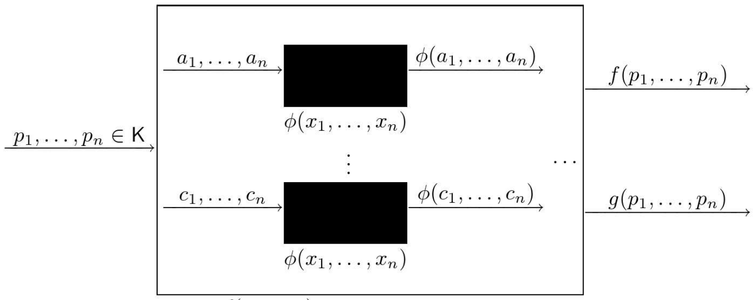
Quelques remarques

- D'après un travail précédent de E. Kaltofen (voir [3]), le pgcd qu'on utilise a de fortes chances d'être celui des $\overline{f_i}$ (on peut se ramener du pgcd de r polynômes à cellui de deux polynômes). Il faut toutefois remarquer que la référence citée ne dit pas exactement ce qu'on cherche. Ca doit être facile de s'y ramener avec un peu d'expérience dans la manipulation des polynômes.
- Par ailleurs le fait que $\overline{pgcd}(p_1, 1) = pgcd(p_1, ...p_n)$ n'est pas vraiment justifié.

Un article publié 5 ans après celui-ci ([5]) explique cet algorithme plus en détail et apporte un certain nombre d'améliorations.

1.2.3 Calcul du numérateur et du dénominateur

Le dernier algorithme présenté ici crée une boite noire qui évalue les numérateurs et dénominateurs d'une fraction rationelle donnée par boite noire.



$\phi(x_1, \ldots, x_n) = \frac{f(x_1,\ldots,x_n)}{g(x_1,\ldots,x_n)}, f, g \in \mathsf{K}[x_1, \ldots, x_n], \mathrm{GCD}(f, g) = 1.$

FIG. 3 – Le résultat de l'algorithme de calcul du dénominateur et numérateur

L'algorithme repose sur une procédure auxiliaire *Special Padé Approximation* telle que :

- entrée : la boite noire calculant f/g, des éléments de K $a_2, ..., a_n$ et $b_2, ...b_n$ et surtout des bornes d et e sur les degrés de f et g
- sortie : deux polynômes $\hat{f}(X)$ et $\hat{g}(X)$ tels que pour $d + e + 1$ valeurs x, on a
$$\frac{\hat{f}(x)}{\hat{g}(x)} = \frac{f}{g}(x, a_2x + b_2, ..., a_nx + b_n)$$
et tels que leurs degrés soient bornés par d et e

Pour construire ces deux polynômes l'idée est la suivante :

- on cherche $d+e+1$ entiers $x_i$pour lesquels $\frac{f}{g}(x_i, a_2x_i+b_2, ..., a_nx_n+b_n)$ est défini
- on calcule par interpolation un polynôme $h(X)$ égal en chacun des x_i à $\frac{f}{g}(X, a_2X + b_2, ..., a_nX + b_n)$
- par une méthode à la Euclide, on calcule deux polynômes $\hat{f}(X)$ et $\hat{g}(X)$ tels que $\hat{f}(X) \equiv \hat{g}(X)$ et tels que ces polynômes sont premies entre eux et de degrés bornés par e et d.

L'efficacité de cette procédure est discutée dans l'article de manière très étendue. Il faut ajouter que le résultat de cette procédure est particulier si les bornes d'entrée sont fausses (il détecte si les bornes ne sont pas cohérentes et retourne une valeur nulle pour les deux polynoems par exemple).

Ceci étant dit, l'algorithme de calcul du numérateur et du dénominateur reste compliqué :

- on choisit des $a_2, ..., a_n, b_2, ...b_n$ aléatoires
- on teste l'algorithme *Special Padé* pour des bornes $1, 2, ...$ sur les degrés et avec les a_i et b_i déja tirés jusqu'à obtenir un résultat. On obtient

les degrés de f et g : ce sont ceux de $\hat{f}(X)$ et $\hat{g}(X)$ avec une forte probabilité.
– Ensuite, étant donnée une entrée $p_1, ...p_n$, on peut calculer les polynômes

$$\overline{f}(X,Y) := f(X, a_2X+b_2+Y(p_2-a_2p_1-b_1), ..., a_nX+b_n+y(p_n-a_np_1-b_n))$$

et

$$\overline{g}(X,Y) := f(X, a_2X+b_2+Y(p_2-a_2p_1-b_1), ..., a_nX+b_n+y(p_n-a_np_1-b_n))$$

Pour celà :
  – on a déja calculé ces polynômes à l'étape précédente pour $Y = 0$.
  – On va utiliser l'algorithme *Special Padé* pour un certain nombre de valeurs de Y pour lesquelles c'est possible. Par interpolation, on peut ensuite calculer explicitement les polynômes $\overline{f}$ et $\overline{g}$.
– on peut ensuite évaluer $\overline{f}$ et $\overline{g}$. en $p_1, 1$ pour obtenir la valeur de f et g en $p_1, ...p_n$.

On a ici éludé de nombreuses difficultés afin de faire mettre l'accent sur l'idée générale. Les algorithmes sont bien mieux justifiés dans l'article. Chaque étape est détaillée d'une manière particulièrement convaincante pour ce dernier algorithme.

Pour ce qui est de la complexité, le théorème 3 de l'article donne une borne polynomiale en fonction de n, des degrés de f et de la borne sur le degré de g nécessaire au calcul. On a aussi besoin d'un nombre polynomiale de requêtes à la boite noire. On peut de plus exiger une correction avec un ϵ arbitraire à condition de pouvoir faire grandir arbitrairement la taille de l'ensemble dans lequel sont choisis les éléments aléatoires.

Pour démontrer ces bornes et en particulier l'espérance du nombre d'appel au *Special Padé*, ils utilisent des arguments visiblement classiques dans ce domaine par exemple la *résultante de Sylvester*...

1.3 Résultats théoriques

Il faut également signaler un résultat sur les classes de complexité induit par l'algorithme de factorisation. Il permet entre autres, en supposant que les appels à la boite noire ne cachent pas une complexité plus grande que NC, de ramener le problème du calcul de tous les polynômes creux à moins de t termes, au problème de la factorisation d'un polynôme à une seule variable à coefficient rationnel.

2 Discussion

La représentation par boite noire est plus générale que la représentation par SLP : on construit facilement une boite noire en interrogreant un SLP.

Par ailleurs la représentation par boite noire peut se ramener simplement à la représentation par polynômes creux (*sparse format representation*) grâce à des algorithmes d'interpolation. Ceci est évoqué par les auteurs dans l'introduction.

De manière plus générale, je crois qu'on peut dire que la représentation par boite noire est la plus générale qui soit. N'importe quelle représentation peut s'y ramener facilement pour peu qu'elle soit effective (c'est à dire qu'on puisse évaluer le polynôme représenté).

Du coup les algorithmes fournis ont une généralité inatteignable avec des SLP par exemple. En effet les algorithmes travaillants sur des SLP utilisent le plus souvent la structure même de la représentaion. En modifiant chaque étape du calcul par exemple pour les SLP.

Le revers de la médaille est dans doute à chercher dans l'efficacité des algorithmes fournis. En effet on imagine intuitivement assez bien que des algorithmes qui se contentent de regarder la représentation comme une boite noire ne peuvent pas profiter des atouts liés à une représentation particulière. Cependant, ici, toutes les considérations de complexité se bornent à des "*in polynomially many arithmetic step*". C'est le cas dans l'article présent mais aussi dans les articles considérant des SLP traitant des mêmes problèmes ([2] pour la factorisation par exemple).

Les auteurs ont une bibliographie personnelle impresionante sur les problèmes concernants les polynômes dans des corps variés ([2],[3]...). Ils se sont visiblement surtout interessés à d'autres représentation. Ici ils adaptent au modèle des boites noires une série de résultats deja obtenus pour des SLP par exemple.

Ils réutilisent donc un certain nombre de résultats déjà développés et auxquels les références sont souvent succinte. En fait ils renvoient souvent à un travail très important sur des résultats plus généraux. C'est le cas pour toutes les justifications du passage de n variables à 2 varialbes par exemple, de l'évocation du théorème de réduction effective de Hilbert...

3 Conclusion

La représentation des polynômes par boite noire permet une généralité bien utilisée par les auteurs mais qui n'empêche pas grâce à des outils d'interpolation de retourner à des modèles plus spécifique par des transformations polynomiales.

Dans cette article les auteurs prouvent dans cette représentation un certain nombre de résultats qui avaient fait l'objets de publications antérieurs pour d'autres représentations et réutilisent donc des résultats mathématiques, surtout probabilistes, déjà introduits et prouvés. Mais la qualité de rédaction de l'article fait que, à mon avis, quelqu'un d'un minimum familier au domaine peut s'interesser à l'article et en comprendre l'intérêt sans trop de difficultés.

Bibliographie

L'article étudié :

[1] E. Kaltofen et B. Trager.
Computing with polynomials given by black boxes for their evaluations : Greatest common divisors, factorization, separation of numerators and denominators.
J. Symbolic Comput., 9(3) :301-320, 1990.

Quelques références issues de la bibliographie de l'article

[2] E. Kaltofen.
Factorization of polynomials given by straight-line programs.
In S. Micali, editor, Randomness and Computation, volume 5 of Advances in Computing Research, pages 375-412. JAI Press Inc., Greenwhich, Connecticut, 1989.

[3] E. Kaltofen.
Greatest common divisors of polynomials given by straight-line programs.
J. ACM, 35(1) : 231-264, 1988.

[4] E. Kaltofen.
Polynomial-time reductions from multivariate to bi- and univariate integral polynomial factorization.
SIAM J. Comput., 14(2) : 469-489, 1985.

Un article plus récent sur le même sujet

[5] A. Díaz et E. Kaltofen.
On computing greatest common divisors with polynomials given by black boxes for their evaluation.
In Proc. 1995 Internat. Symp. Symbolic Algebraic Comput. (ISSAC'95), pages 232-239.