Pgcd de deux polynômes à paramètres : approche par la clôture constructible dynamique et généralisation de la méthode de S.A. Abramov, K.Yu. Kvashenko

Stéphane Dellière

INRIA

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

Pgcd de deux polynômes à paramètres : approche par la clôture constructible dynamique et généralisation de la méthode de S.A. Abramov, K.Yu. Kvashenko

Stéphane Dellière

N° 3882

Février 2000

THÈME 2

ISSN 0249-6399 ISRN INRIA/RR--3882--FR

INRIA
SOPHIA ANTIPOLIS

Pgcd de deux polynômes à paramètres : approche par la clôture constructible dynamique et généralisation de la méthode de S.A. Abramov, K.Yu. Kvashenko

Stéphane Dellière

Thème 2 — Génie logiciel
et calcul symbolique
Projet CAFÉ

Rapport de recherche n° 3882 — Février 2000 — 26 pages

Résumé : La première partie de ce papier est consacrée à l'étude de deux méthodes de calculs de pgcd de deux polynômes en une indéterminée et t paramètres. Ces deux algorithmes sont dus à T. Gómez-Díaz et l'auteur d'une part et à S.A. Abramov, K.Yu. Kvashenko d'autre part. Le domaine d'application du second algorithme est limité au cas $t = 1$. Dans la deuxième moitié de ce travail, celui-ci est alors généralisé à $t \geq 1$ et implanté en Axiom.

Mots-clés : Pgcd, polynôme, paramètre, sous-résultant, évaluation dynamique, Axiom.

Unité de recherche INRIA Sophia Antipolis
2004, route des Lucioles, B.P. 93, 06902 Sophia Antipolis Cedex (France)
Téléphone : 04 92 38 77 77 - International : +33 4 92 38 77 77 — Fax : 04 92 38 77 65 - International : +33 4 92 38 77 65

Gcd of two polynomials with parameters: the dynamical constructible closure approach and a generalisation of the method of S.A. Abramov, K.Yu. Kvashenko

Abstract: The first part of this paper compares two methods to compute the gcd of two polynomials with parameters. The first method is due to T. Gómez-Díaz and the author. The second is due to S.A. Abramov, K.Yu. Kvashenko and is restricted to the case where there is only one parameter. The second part presents a generalisation to several parameters of this latter method and its implementation in AXIOM.

Key-words: gcd, polynomial, parameter, subresultant, dynamical evaluation, Axiom

1 Introduction

Le problème du calcul de deux polynômes p et q à une indéterminée et à un paramètre intervient dans de nombreux domaines du calcul formel (se reporter à l'introduction de [2] pour de multiples références). On se propose d'étudier deux algorithmes. T. Gómez-Díaz est à l'origine du premier. C'est une application, parmi beaucoup d'autres [10, 11, 12], de ses programmes de la clôture constructible dynamique [9] implantés dans le logiciel de calcul formel Axiom [13]. Il est valable pour un nombre quelconque de paramètres. Le second, en revanche, est défini uniquement dans le cas d'un seul paramètre. Il est proposé par S.A. Abramov, K.Yu. Kvashenko dans [2].
Chacun des deux algorithmes est présenté et analysé respectivement dans les sections 2 et 3. En particulier, on montre que le second s'inscrit naturellement dans le cadre de l'évaluation dynamique. La section 4 est consacrée à la comparaison des deux méthodes. On propose ainsi dans la section 4.1 une condition nécessaire sur p et q pour que les deux algorithmes retournent un résultat identique. Différents exemples de calculs de pgcd illustrent ce résultat dans la section 4.2. Par ailleurs, on montre que les deux algorithmes ne diffèrent que par le traitement des coefficients dominants et des contenus des polynômes p et q. Il apparaît que cette différence penche à l'avantage de la méthode de S.A. Abramov, K.Yu. Kvashenko. De plus, on montre qu'il est conforme au formalisme de l'évaluation dynamique de [6, 9]. On effectue alors son implantation au sein des programmes de T. Gómez-Díaz. Cela aboutit à la création d'un package Axiom appelé ABRAMOV dont l'étude est menée dans la section 5. Cet algorithme généralise en particulier la méthode originale de S.A. Abramov, K.Yu. Kvashenko à un nombre quelconque de paramètres.

2 Méthode de la clôture constructible dynamique

Soient n paramètres $a_1,\ldots,a_n$ et un corps K. Les programmes de la clôture constructible dynamique de T. Gómez-Díaz permettent de calculer le pgcd de deux polynômes à une indéterminée et à coefficients dans $K[a_1,\ldots,a_n]$. Cette méthode repose sur une fonction appelée `gcd` implantée dans le domaine des polynômes dynamiques [6, 9]. Le but de cette section est d'analyser cette fonction. Son code (commenté ci-après) est le suivant :

```
gcd(p,q) ==
  p =$$ 0 => q
  q =$$ 0 => p
  p1:= reduce p
  dp:= maxIndex p1
  ++ dp := degre de p1
  dp = 0 => 1$$
  q1:= reduce q
  dq:= maxIndex q1
  ++ dq := degre de q1
  dq = 0 => 1$$
  if dp < dq then
    p2:= copy p1
    p1:= copy q1
    q1:= copy p2
  lp:= Suite_reste_ss_res(p1,q1)
  until r ^=$K 0 repeat
    p1:= lp.first
    lp:= lp.rest
    r:= p1.0
    ++ r := coefficient dominant de p1
  p1 / r
```

Remarque. L'étude menée dans cette section utilise la version des programmes de T. Gómez-Díaz qui bénéficie des implantations réalisées par l'auteur [6] de la condition de sans carré de D. Lazard [14] et d'un algorithme de calculs des sous-résultants basés sur les travaux de S.J. Berkowitz et de J. Abdeljaoued [1].

On se place désormais dans le cadre plus restreint de deux polynômes p et q *à un paramètre et à une indéterminée* notés respectivement a et X.

Afin d'utiliser cette fonction, il faut écrire un programme Axiom qui suit le schéma suivant :

```
K:= corps de base
CL:= DCCLOS K
Pol:= DynamicUnivariatePolynomial(CL)
X:Pol:= monomial(1,1)$Pol

CP:= DynamicConstructibleControlPackage(K,Pol)
dynamicGCD():Pol ==
  a:CL:= parameter('a)
  ++ definition des polynomes p et q
  ++ a coefficients dans K, en l'indeterminee X
  ++ et a un parametre a
  p:Pol:=
  q:Pol:=
  gcd(p,q)$Pol

allCases(dynamicGCD)$CP
```

En écrivant `CL:= DCCLOS K`, le domaine Axiom appelé `DynamicConstructibleClosure`, défini à partir du corps `K`, est désormais symbolisé par `CL`. C'est l'un des domaines principaux des programmes de la clôture constructible dynamique : il gère notamment l'introduction des paramètres et l'imposition des contraintes. De même, on note désormais `Pol` le domaine Axiom des polynômes dynamiques. En particulier, les polynômes p et q dont on va calculer les pgcd en fonction du paramètre a sont des éléments de `Pol`. L'indéterminée `X` est introduite à l'aide de l'instruction `X:Pol:= monomial(1,1)`. Enfin, le symbole `CP` renvoie au package Axiom `DynamicConstructibleControlPackage` dont le rôle est de fournir une fonction clé de ces programmes appelée `allCases` [9, section 4.1]. Son but est de donner la réponse *complète* d'un calcul.
Ainsi, afin d'appliquer la méthode de la clôture constructible dynamique sur l'exemple 1 page 14, on écrit le petit programme Axiom suivant (avec $K = \mathbb{Q}$) :

```
RN:= Fraction Integer
CL:= DynamicConstructibleClosure(RN)

Pol:= DynamicUnivariatePolynomial(CL)
X:Pol:= monomial(1,1)$Pol

CP:= DynamicConstructibleControlPackage(RN,Pol)

dynamicGCD():Pol ==
  a:CL:= parameter('a)
  p:Pol:= X**2+(a-1)*X+a
  q:Pol:= (a+1)*X**2+2*a
  gcd(p,q)$Pol

allCases(dynamicGCD)$CP
```

La méthode de la clôture constructible dynamique renvoie en sortie une liste de clauses [9]. Chacune d'entre elles est constituée :

- d'une valeur v du type : "value is $r(a,X)$";
- d'une loi $c(a)$.

Cela signifie que si le paramètre a vérifie la loi $c(a)$ alors le pgcd de p et q est $r(a,X)$. Cette loi peut être de trois types :

- *any* a. Dans ce cas, le paramètre a n'est soumis à aucune contrainte. Par exemple, le pgcd de $X + a$ et de $X + a$ est $X + a$ quel que soit la valeur de a. La clause est alors : "value is $X + a$ in case *any* a";
- $f_1(a) \neq 0, \ldots, f_k(a) \neq 0$ où $f_1, \ldots, f_k$ sont des polynômes unitaires de $K[a]$ de degrés strictement positifs, sans carré et premiers entre eux. Par définition, le paramètre a n'est pas un zéro de l'un des f_i $(1 \leq i \leq k)$. On dit alors que a est soumis à une contrainte de type *exception*;
- $f(a) = 0$ où f est un polynôme unitaire de $K[a]$ de degré strictement positif et sans carré. Cela signifie que a est un zéro de f. On dit alors que a est soumis à une contrainte de type *algébrique*.

Analysons la fonction `gcd`. On effectue tout d'abord la remarque suivante.

Remarque. Les polynômes dynamiques p,q,p_1 et q_1 de la fonction `gcd` peuvent s'interpréter comme des éléments de $L_1[X]$ [6, chapitre 4]. Dans le cadre de notre étude, l'anneau L_1 est égal à :

- $K[a]$ si a n'est soumis à aucune contrainte;
- $G^{-1}K[a]$ où G est la partie multiplicative de $K[a]$ engendrée par $f_1(a),\ldots,f_k(a)$ si a est soumis à la contrainte $f_1(a)\neq 0,\ldots,f_k(a)\neq 0$;
- $\dfrac{K[a]}{\langle f(a)\rangle}$ si a est soumis à la contrainte algébrique $f(a)=0$.

Soit $\widetilde{K}$ une clôture algébrique du corps K. Chacune de ces contraintes définit clairement un ensemble constructible de $\widetilde{K}$ noté par la suite Z_1. Il s'agit du sous-ensemble de $\widetilde{K}$ constitué des valeurs de a satisfaisant la contrainte à laquelle il est soumis.

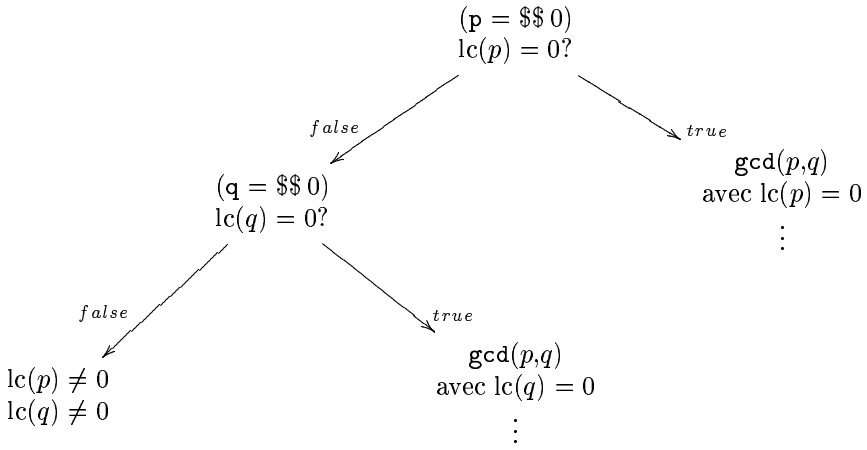
La première étape :

```
p =$$ 0 => q
q =$$ 0 => p
```

consiste à éliminer les cas triviaux où les polynômes p et q sont nuls dans la branche de calculs courante. Dans ce cas :

$$\begin{cases} (p=0)\Rightarrow(\text{pgcd}(p,q)=q);\\ (q=0)\Rightarrow(\text{pgcd}(p,q)=p). \end{cases}$$

Les deux opérations d'égalités (=$$) effectuées dans cette étape sont de nature dynamique. En effet, elles sont la source d'éventuels scindages. On peut schématiser leur fonctionnement de la manière suivante :



Ainsi, après cette étape, les deux polynômes p et q ont un coefficient dominant non nul (dans la branche de calculs courante). Autrement dit :

$$\forall\,\alpha\in Z_1,\text{lc}(p)(\alpha)\neq 0,\text{lc}(q)(\alpha)\neq 0.$$

Remarque. On peut noter que cette condition est relativement forte. En effet, l'algorithme des sous-résultants se spécialise bien dès que la condition ci-dessus est vérifiée seulement par l'un (au moins) des deux coefficients dominants des polynômes (se reporter par exemple à [8, corollaire 4.1 p.45]).
De plus, notons C_p et C_q les contenus des deux polynômes p et q. Par définition, ils vérifient en particulier :

$$\text{lc}(p)\in\langle C_p\rangle,\ \text{lc}(q)\in\langle C_q\rangle.$$

On en déduit alors que la condition précédente est aussi valable pour les contenus :

$$\forall\,\alpha\in Z_1,C_p(\alpha)\neq 0,C_q(\alpha)\neq 0.$$

La seconde étape :

```
p1:= reduce p
dp:= maxIndex p1
dp = 0 => 1$$
```

```
q1:= reduce q
dq:= maxIndex q1
dq = 0 => 1$$
```

a pour but de détecter, après réduction, les cas triviaux où p et q sont des polynômes constants non nuls. Elle trouve son origine dans le lemme suivant.

Lemme 2.1 *Après les tests d'égalité* `p =$$ 0` *et* `q=$$ 0`*, les coefficients dominants* $lc(p)$ *et* $lc(q)$ *de* p *et de* q *sont inversibles dans l'anneau* L_1*.*

Preuve. Les polynômes p et q peuvent s'interpréter comme des éléments de $L_1[X]$. De plus, après la première étape, on sait que les coefficients dominants de p et de q ne s'annulent pas sur Z_1. Il suffit alors d'appliquer [6, théorème 2.3.1] pour conclure.◇

Ainsi, si $\deg(p) = 0$ alors p est un élément de L_1 inversible. Il est donc proportionnel à 1. Par suite, l'élément 1 de L_1 constitue bien un pgcd de p et de q. Le cas $\deg(q) = 0$ étant similaire, on en déduit donc :

$$\begin{cases} (\deg(p) = 0) \Rightarrow (\text{pgcd}(p,q) = 1); \\ (\deg(q) = 0) \Rightarrow (\text{pgcd}(p,q) = 1). \end{cases}$$

Ce qui justifie bien les instructions ci-dessus.
La troisième étape :

```
if dp < dq then
  p2:= copy p1
  p1:= copy q1
  q1:= copy p2
```

est élémentaire. On souhaite appliquer l'algorithme des sous-résultants en s'assurant que :

$$\deg(p) \geq \deg(q) > 0.$$

Ainsi, si $\deg(p) < \deg(q)$, celle-ci consiste simplement à échanger les polynômes p et q. Cette étape trouve sa justification dans le résultat classique suivant. Etants donnés deux polynômes univariés f et g, les sous-résultants de f et de g sont égaux, au signe près, aux sous-résultants de g et de f [15, lemme 7.7.2 p.254].
Reste donc la dernière étape, le calcul des sous-résultants :

```
lp:= Suite_reste_ss_res(p1,q1)
until r ^=$K 0 repeat
  p1:= lp.first
  lp:= lp.rest
  r:= p1.0
p1 / r
```

Il ressort de ce qui précède que l'algorithme des sous-résultants est appliqué à deux polynômes p_1 et q_1 vérifiant :

1. $\deg(p_1) \geq \deg(q_1) > 0$;
2. $\forall \alpha \in Z_1, \text{lc}(p_1)(\alpha) \neq 0, \text{lc}(q_1)(\alpha) \neq 0$.

En particulier, *le second point garantit la bonne spécialisation de l'algorithme des sous-résultants* comme l'atteste [8, théorème 4.1 p.45] et *justifie son utilisation dans le calcul du pgcd de* p_1 *et* q_1 [8, théorème 4.3 p.48].

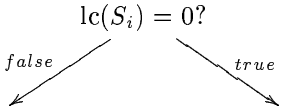
L'instruction :

```
lp:= Suite_reste_ss_res(p1,q1)
```

retourne une liste :

$$lp = [S_d, \ldots, S_{n-1}, S_n = q_1]$$

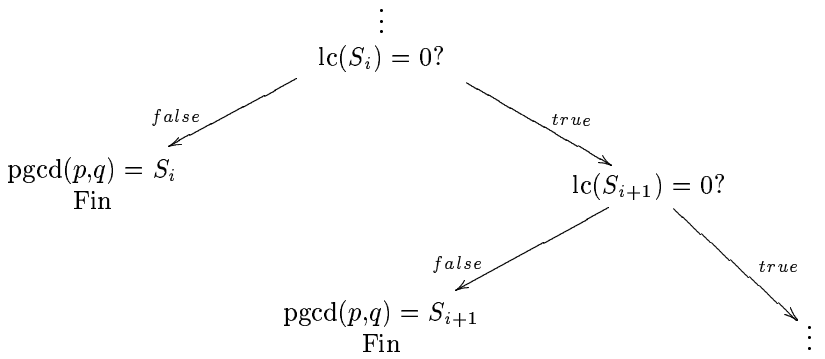
avec $d \in \mathbb{N}$ et où $(S_n, \ldots, S_d)$ constitue une suite de restes sous-résultants de p_1 et de q_1 [8, définition p.48] . Il s'agit d'une suite de polynômes de degrés strictement décroissants, proportionnels à la suite des restes de p_1 par q_1 et qui sont tous des sous-résultants de p_1 et q_1.
La boucle `until...repeat` suit alors un algorithme classique. Pour tout $d \leq i \leq n$, on considère le polynôme S_i. On distingue alors deux cas :



Si $\mathrm{lc}(S_i) \neq 0$ (dans la branche courante) alors [8, théorème 4.3 p.48] permet de conclure que :

$$\mathrm{pgcd}(p,q) = \mathrm{pgcd}(p_1,q_1) = S_i.$$

Dans le cas contraire ($\mathrm{lc}(S_i) = 0$ dans la branche courante) alors on réitère ce processus avec la nouvelle contrainte $\mathrm{lc}(S_i) = 0$. Pour tout $d \leq i \leq n-2$, il s'agit donc d'opérer le schéma suivant :



Le cas $i = n$ est simple. En effet, à ce stade, les coefficients dominants des sous-résultants d'indice $\leq n-1$ sont nuls. Par suite, on déduit de [8, théorème 4.3 p.48] qu'un pgcd de p et q est S_n, c'est-à-dire, le polynôme q_1.
Enfin, afin de collecter l'ensemble des solutions dans les différents cas, on utilise alors la fonction `allCases`. Elle permet de donner la réponse complète à la question "quel est le pgcd de $p(a,X)$ et de $q(a,X)$?".

3 Méthode de S.A. Abramov et de K.Yu. Kvashenko

Les auteurs de [2] proposent une méthode de calculs de pgcd de deux polynômes p et q à une indéterminée (notée X), à coefficients dans un corps K et à un paramètre a. Celui-ci peut être soumis à une contrainte $g(a) = 0$ où g est soit le polynôme nul, soit un polynôme sans carré de degré strictement positif.
Leur algorithme, appelé GCDpar, retourne des paires de la forme :

$$(d_1,G_1),\ldots,(d_k,G_k)$$

ou

$$(d_1,G_1),\ldots,(d_k,G_k),(0,G_{k+1})$$

où, pour tout $1 \leq i \leq k$, les d_i sont des polynômes de $K[a]$, sans carré et premiers entre eux. De plus, pour tout $1 \leq i \leq k$, les G_i sont des polynômes de $K[a,X]$ tels que :

$$(d_i(a) = 0) \Rightarrow (\mathrm{lc}(G_i)(a) \neq 0).$$

Les notations précédentes signifient que pour tout $1 \leq i \leq k$, si $d_i(a) = 0$ alors le pgcd de p et de q est G_i. Enfin, si l'algorithme retourne une paire $(0,G_{k+1})$, alors :

$$(d_1(a) \neq 0, d_2(a) \neq 0, \ldots, d_k(a) \neq 0) \Rightarrow (\mathrm{pgcd}(p,q) = G_{k+1}).$$

Leur algorithme est constitué de six étapes. On se propose d'analyser chacune d'entre elles. On met notamment en évidence un point important qui n'apparaît pas dans [2] : leur algorithme s'inscrit naturellement dans le cadre de l'évaluation dynamique.

Etape 1
Calculs préliminaires
$p := p \bmod g$
$q := q \bmod g$
Si $p = 0$ alors retourner (g,q)
Si $q = 0$ alors retourner (g,p)

Après une éventuelle réduction si a est soumis à une contrainte algébrique ($g(a) = 0$), il s'agit d'éliminer les cas où les polynômes p et q sont nuls. Bien entendu, on a alors :

$$\begin{cases} (p = 0) \Rightarrow (\text{pgcd}(p,q) = q); \\ (q = 0) \Rightarrow (\text{pgcd}(p,q) = p). \end{cases}$$

Etape 2
Etude des contenus
C_p:= contenu de p
C_q:= contenu de q
p:= partie primitive de p
q:= partie primitive de q
Si a est soumis à une contrainte algébrique $g(a) = 0$ alors
C_p:= pgcd(C_p,g)
C_q:= pgcd(C_q,g)
g:= $\dfrac{g}{\text{ppcm}(C_p,C_q)}$
Sinon
C_p:= partie sans carrée de C_p
C_q:= partie sans carrée de C_q
C:= pgcd(C_p,C_q)
result:= $(C,0),(\dfrac{C_p}{C},q),(\dfrac{C_q}{C},p)$

Le but de cette partie est clair. Il s'agit de rendre primitifs les polynômes p et q. Si $C_p \neq 0, C_q \neq 0$ alors on peut calculer les parties primitives $pp(p)$,$pp(q)$ de p et de q, respectivement égales à $\frac{p}{C_p}$ et $\frac{q}{C_q}$.

Notation. Etant donné un polynôme f dans $K[a]$, on note $\sqrt{f}$ la partie sans carrée de f.

On se place dans le cadre de l'évaluation dynamique. Ainsi, on utilise dans cette section indifféremment les termes "loi" et "contrainte". Si a n'est soumis à aucune contrainte alors, après cette étape, la nouvelle loi (à laquelle obéit le paramètre a) est naturellement :

$$\sqrt{C_p}(a) \neq 0, \sqrt{C_q}(a) \neq 0.$$

Dans le cas contraire, le paramètre a est soumis par hypothèse à une contrainte algébrique $g(a) = 0$. Le paramètre a est donc soumis aux contraintes :

$$\begin{cases} g(a) = 0 \\ C_p(a) \neq 0, C_q(a) \neq 0 \end{cases}$$

Puisque les polynômes g,C_p,C_q appartiennent à $K[a]$ et que g est sans carré alors, en se reportant à [9], on peut montrer (par des manipulations de pgcd et de ppcm) que la nouvelle loi est[1] :

$$g(a) = \left(\frac{g}{\text{ppcm}(\text{pgcd}(g,C_p),\text{pgcd}(g,C_q))}\right)(a) = 0.$$

Si $C_p\, C_q = 0$ alors il y a naturellement trois cas à étudier. Si $C_p = C_q = 0$ alors le pgcd de p et q est nul. Si $C_p = 0$ et $C_q \neq 0$ alors le pgcd est q. Dans ce cas, en s'appuyant toujours sur [9] et l'hypothèse de sans carré de g, alors :

- si a est soumis à une contrainte algébrique, la nouvelle loi est :

$$g(a) = \left(\frac{\text{pgcd}(g,C_p)}{\text{pgcd}(\text{pgcd}(g,C_p),\text{pgcd}(g,C_q))}\right)(a) = 0.$$

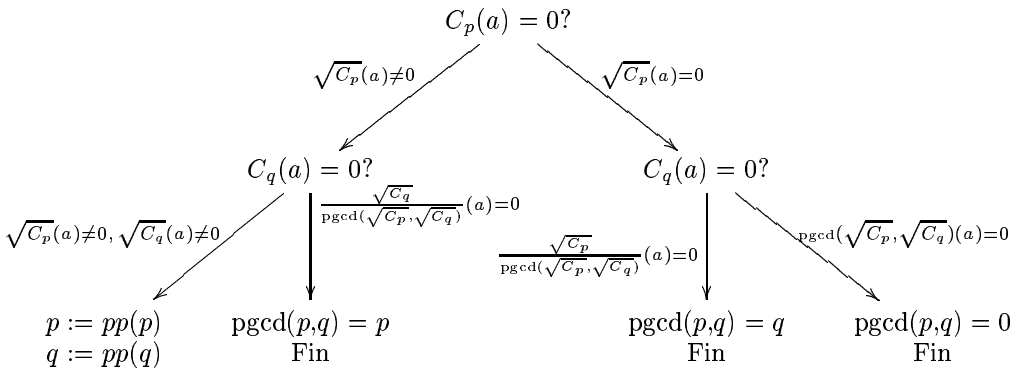
- sinon la nouvelle loi est :

$$g(a) = \left(\frac{\sqrt{C_p}}{\text{pgcd}(\sqrt{C_p},\sqrt{C_q})}\right)(a) = 0.$$

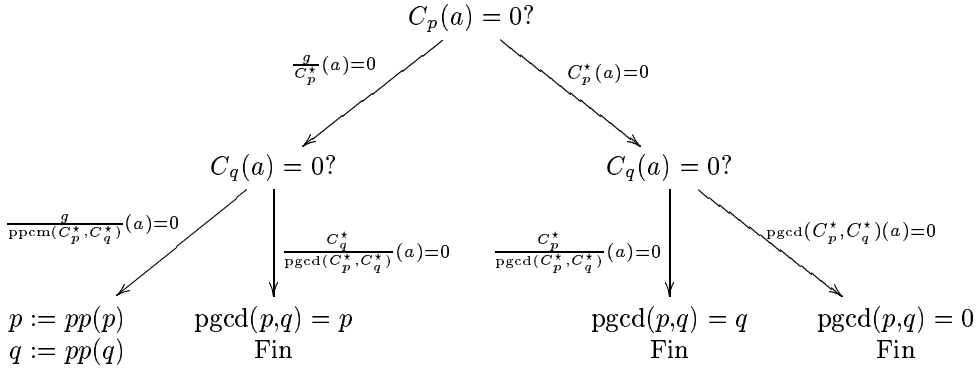
1. En utilisant ce même type d'arguments, on peut établir les différentes lois qui apparaissent tout au long de l'algorithme. Nous ne ferons pas les démonstrations. On renvoie pour cela à [9, théorèmes 3.3 p.73 et 3.5 p.75]

Le cas $C_p \neq 0$ et $C_q = 0$ est analogue.
Il faut noter que cette distinction de cas, effectuée dans le contexte de l'évaluation dynamique, justifie toutes les instructions opérées dans cette étape.
Les deux arbres suivants résument le processus dynamique réalisé. Sur chacune de leurs arêtes figure la contrainte subie par le paramètre a. Par ailleurs, pour chacun des nœuds, la branche de gauche renvoie à une réponse de type *false* et la branche de droite à une réponse de type *true*. Ce type d'arbre est à rapprocher de la notion d'arbre de scindages [9].

Si a n'est soumis à aucune contrainte :



Si a est soumis à une contrainte algébrique $g(a) = 0$:



avec $C_p^\star = \mathrm{pgcd}(g,C_p)$ et $C_q^\star = \mathrm{pgcd}(g,C_q)$.

Remarque. Pour des raisons de lisibilité, on note toujours p et q respectivement les polynômes $\frac{p}{C_p}$ et $\frac{q}{C_q}$.

Etape 3
Cas simples
Si $\deg(p) = 0$ ou $\deg(q) = 0$ alors retourner $(g,1)$

Autrement dit :

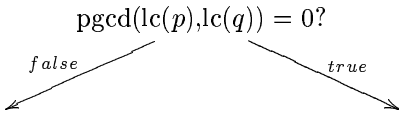
$$\begin{cases} (\deg(p) = 0) \Rightarrow (\mathrm{pgcd}(p,q) = 1); \\ (\deg(q) = 0) \Rightarrow (\mathrm{pgcd}(p,q) = 1). \end{cases}$$

Cette étape se justifie pleinement si on se place dans le cadre de l'évaluation dynamique. En effet, on peut alors considérer l'anneau L_1 qui, par construction, est égal à $G^{-1}K[a]$, où G est la partie multiplicative de $K[a]$ engendrée par C_p et C_q. De plus, les polynômes p et q peuvent s'interpréter comme des éléments de $L_1[X]$. En particulier, si $\deg(p) = 0$ alors le polynôme p est égal à son contenu C_p. On en déduit que p est inversible dans L_1 et donc que 1 constitue bien un pgcd de p et q. Le cas $\deg(q) = 0$ est analogue.

Etape 4
Etude des coefficients dominants
δ:= pgcd(lc(p),lc(q))
Si a est soumis à une contrainte algébrique $g(a) = 0$ alors
δ:= pgcd(δ,g)

Sinon
δ:= partie sans carrée de δ
Si $\deg(\delta) > 0$ alors
result:= GCDpar(p,q,δ), result
Si a est soumis à une loi algébrique $g(a) = 0$ alors $g := \frac{g}{\delta}$

Les auteurs de [2] rappellent tout d'abord un résultat classique. Si les coefficients dominants de p et q ne sont pas simultanément nuls alors l'algorithme des sous-résultants se spécialise bien. C'est évidemment le cas si pgcd(lc(p),lc(q)) est non nul. D'où la nécessité de distinguer les deux situations :



Dans la branche de gauche, on s'assure que lc(p) et lc(q) ne sont pas simultanément nuls. Il faut noter que si a n'est soumis à aucune contrainte alors la nouvelle loi est :

$$\sqrt{\text{pgcd}(\text{lc}(p),\text{lc}(q))}(a) \neq 0$$

Dans le cas contraire (a est soumis à la contrainte $g(a) = 0$), la nouvelle loi est :

$$\left(\frac{g}{\text{pgcd}(\text{pgcd}(\text{lc}(p),\text{lc}(q)),g)}\right)(a) = 0$$

D'où les instructions :
δ:= pgcd(lc(p),lc(q))
δ:= pgcd(δ,g)
$g := \frac{g}{\delta}$
Si pgcd(lc(p),lc(q)) = 0, il s'agit de relancer le calcul du pgcd de p et de q en tenant compte de cette contrainte supplémentaire. Ainsi, si a n'est soumis à aucune contrainte alors la nouvelle loi est :

$$\sqrt{\text{pgcd}(\text{lc}(p),\text{lc}(q))}(a) = 0$$

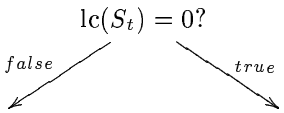
et dans le cas contraire (a est soumis à la contrainte $g(a) = 0$) :

$$\text{pgcd}(\text{pgcd}(\text{lc}(p),\text{lc}(q)),g)(a) = 0$$

Ce qui explique les instructions :
δ:= pgcd(lc(p),lc(q))
δ:= partie sans carrée de δ (si a n'est soumis à aucune contrainte)
δ:= pgcd(δ,g) (si a est soumis à la contrainte $g(a) = 0$)
result:= GCDpar(p,q,δ), result

Etape 5
Etude des sous-résultants
n:= deg(q)
Calcul de la suite des sous-résultants de p et q
d'indice $\leq n-1$: $S_0, \ldots, S_{n-1}$
S_t:= sous-résultant non identiquement nul de plus bas degré
r:= lc(S_t)
Si a est soumis à une contrainte algébrique $g(a) = 0$ alors
r:= pgcd(r,g)
g:= $\frac{g}{r}$
Sinon
r:= partie sans carrée de r
r:= $\frac{r}{\text{ppcm}(C_u,C_v,\delta)}$
result:= *result*,(g,S_t)

On rappelle que par hypothèse, les coefficents dominants de p et de q ne s'annulent pas simultanément. L'algorithme des sous-résultants se spécialise donc bien. On s'intéresse au sous-résultant S_t non identiquement nul de plus bas degré. Il s'agit alors de distinguer les deux cas :



Dans la branche de gauche, on déduit de [8, théorème 4.3 p.48] que $\text{pgcd}(p,q) = S_t$. Par ailleurs, si a est soumis à une contrainte algébrique $g(a) = 0$ alors la nouvelle loi est :

$$\left(\frac{g}{\text{pgcd}(\text{lc}(S_t),g)}\right)(a) = 0$$

Cela justifie les instructions :
r:= $\text{lc}(S_t)$
r:= $\text{pgcd}(r,g)$
g:= $\frac{g}{r}$
Ainsi, on a l'implication :

$$(g(a) = 0) \Rightarrow (\text{pgcd}(p,q) = S_t).$$

On peut noter que si a n'est soumis à aucune contrainte (i.e. g est le polynôme nul), cette implication reste vraie. Cela justifie la pertinence de la dernière instruction de cette étape.

Dans la branche de droite, on suppose désormais que $\text{lc}(S_t)$ est nul. Si a est soumis à une contrainte algébrique $g(a) = 0$ alors la nouvelle loi est :

$$\text{pgcd}(\text{lc}(S_t),g)(a) = 0.$$

Dans le cas contraire, il ne faut pas perdre de vue que C_p,C_q et $\delta = \text{pgcd}(\text{lc}(p),\text{lc}(q))$ sont supposés non nuls. Aussi, la nouvelle loi est :

$$\left(\frac{\sqrt{\text{lc}(S_t)}}{\text{pgcd}(\sqrt{\text{lc}(S_t)},\text{ppcm}(C_p,C_q,\delta))}\right)(a) = 0.$$

Ces deux remarques justifient respectivement les instructions :
r:= $\text{lc}(S_t)$
r:= $\text{pgcd}(r,g)$
et :
r:= $\text{lc}(S_t)$
r:= partie sans carrée de r
r:= $\frac{r}{\text{ppcm}(C_u,C_v,\delta)}$
Il faut noter que dans les deux cas, le polynôme qui définit la nouvelle contrainte algébrique est précisément la valeur de r à la fin de cette étape.

Notation. Etants donnés un anneau A et un élément a de A, on note $a \equiv 0$ si a est identiquement nul.

Etape 6
Pour $i = t+1$ à $n-1$ tel que $S_i \not\equiv 0$ faire
h:= $\text{pgcd}(r,\text{lc}(S_i))$
w:= $\frac{r}{h}$
r:= h
result:= *result*,(w,S_i)
result:= (r,q),*result*
return result

Il s'agit [2] de l'algorithme de calculs de pgcd de deux polynômes à partir de la suite de leurs sous-résultants. Le processus de la fin de l'étape précédente est itéré de nouveau pour chacun des sous-résultants non identiquement nuls sous l'hypothèse $\text{lc}(S_t) = 0$. On a montré qu'au début de la boucle, le paramètre a est soumis à la contrainte

2. Les auteurs de [2] appliquent cette étape à une *suite régulière* des sous-résultants de p et q. Il s'agit d'une sous-suite $\mathcal{S}$ de $(S_{t+1},\ldots,S_{n-1})$ où chacun des sous-résultants d'indice $i\,(t+1 \leq i \leq n-1)$ est de degré i. Il suffit alors de remplacer la première ligne par "Pour tout $S_i \in \mathcal{S}$ faire".

algébrique $r(a) = 0$. Supposons cela vrai à l'ordre i avec $t+1 \leq i \leq n-1$. On considère le coefficient dominant $\mathrm{lc}(S_i)$ de S_i. Si celui-ci est non nul alors le pgcd recherché est S_i. De plus, dans ce cas, la loi algébrique est :

$$\left(\frac{r}{\mathrm{pgcd}(r,\mathrm{lc}(S_i))}\right)(a) = 0.$$

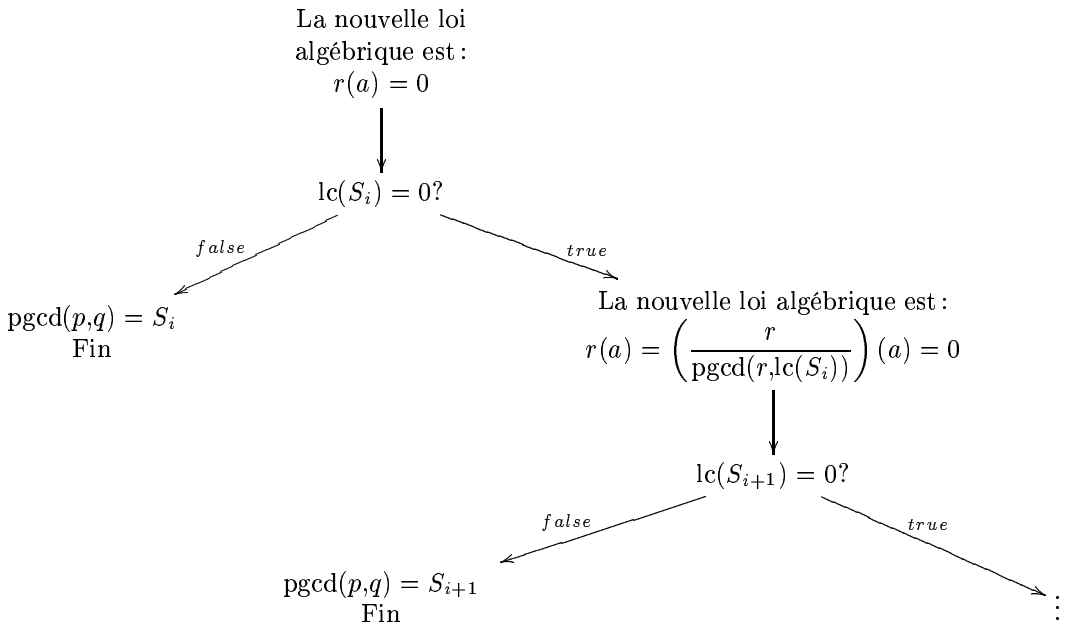
Ainsi, par définition de w, on a l'implication :

$$(w(a) = 0) \Rightarrow (\mathrm{pgcd}(p,q) = S_i).$$

Ce qui justifie les instructions qui figurent dans la boucle Pour.
Dans le cas contraire, la nouvelle loi est :

$$\mathrm{pgcd}(r,\mathrm{lc}(S_i))(a) = 0$$

c'est-à-dire $r(a) = 0$ par construction. On a donc montré que pour tout $t+1 \leq i \leq n-1$, si $\mathrm{lc}(S_i)$ est identiquement nul alors la nouvelle contrainte est $r(a) = 0$.
Pour tout $t+1 \leq i \leq n-2$, il s'agit donc de suivre le schéma suivant :



A la fin de la boucle Pour, tous les sous-résultants d'indice inférieur ou égal à $n-1$ sont nuls. Il ne reste qu'un seul sous-résultant à considérer : c'est le polynôme q lui-même. De plus, on sait que la loi algébrique est $r(a) = 0$. Ainsi, cela signifie que si $r(a)$ est nul alors le sous-résultant non identiquement nul de plus bas degré est le polynôme q. On en déduit alors que q est dans ce cas le pgcd recherché. D'où les instructions :
result:= (r,q),*result*
return result

4 Analyse et exemples

4.1 Analyse comparative des deux méthodes

On remarque tout d'abord que la méthode de la clôture constructible dynamique est plus générale que celle de S.A. Abramov et de K.Yu. Kvashenko pour deux raisons. Tout d'abord, elle est valable pour un nombre t de paramètres quelconque. De plus, si $t = 1$, elle reste aussi plus générale dans la mesure où le paramètre peut-être soumis initialement à une contrainte de type exception. Ce qui n'est pas possible avec l'algorithme de S.A. Abramov et de K.Yu. Kvashenko. Il faut noter que cette impossiblité tient seulement à la structure de leur algorithme. En effet, celui-ci s'inscrivant naturellement dans le contexte de l'évaluation dynamique, on pourrait envisager de généraliser leur algorithme de sorte qu'il accepte une contrainte initiale quelconque (i.e. de type algébrique ou exception). C'est en particulier l'objet de la section 5.

Par ailleurs, les étapes 1,3,5,6 de l'algorithme de S.A. Abramov et de K.Yu. Kvashenko se retrouvent sous une forme quasi-identique dans la méthode de la clôture constructible dynamique. Il n'y a donc aucune différence à ce niveau. En revanche, le traitement des coefficients dominants et des contenus des polynômes p et q diffère d'une méthode à l'autre (étapes 2 et 4 de l'algorithme de S.A. Abramov et de K.Yu. Kvashenko).
Plus précisément, la remarque page 5 a mis en évidence le fait que la méthode de la clôture constructible dynamique impose :

$$\left\{ \begin{array}{l} g(a) = 0 \\ \mathrm{lc}(p)(a) \neq 0, \mathrm{lc}(q)(a) \neq 0 \end{array} \right.$$

avec notamment pour conséquence :

$$\left\{ \begin{array}{l} g(a) = 0 \\ C_p(a) \neq 0, C_q(a) \neq 0 \end{array} \right.$$

Dans leur algorithme, S.A. Abramov et de K.Yu. Kvashenko imposent à ce niveau les conditions :

$$\left\{ \begin{array}{l} g(a) = 0 \\ \mathrm{pgcd}(\mathrm{lc}(\frac{p}{C_p}), \mathrm{lc}(\frac{q}{C_q}))(a) \neq 0, C_p(a) \neq 0, C_q(a) \neq 0 \end{array} \right.$$

Cette deuxième condition est clairement plus faible que la précédente. Ainsi, le traitement des coefficients dominants et des contenus des polynômes p et q est *plus fin* dans la méthode de S.A. Abramov et de K.Yu. Kvashenko.

La proposition suivante propose une condition nécessaire pour que les deux algorithmes donnent malgré tout un résultat identique. On démontre au prélable un petit lemme.

Lemme 4.1 *Soit A un anneau commutatif unitaire et u,v deux polynômes de $A[X]$ de degrés respectifs m et n avec $m \geq n > 0$. Pour tout $0 \leq i \leq n-1$, on note $SousRes_i(u,v)$ le sous-résultant de u et v d'indice i. On se donne deux éléments non nuls a et b de A. Alors pour tout $0 \leq i \leq n-1$:*

$$SousRes_i(a\,u, b\,v) = a^{n-i}\, b^{m-i}\, SousRes_i(u,v).$$

Preuve. On note que le résultat est immédiat pour $i = 0$ par la propriété classique de bilinéarité du déterminant. Soit un entier i compris entre 1 et $n-1$. On note Σ_i la matrice de Sylvester d'indice i associée à u et v et $\widetilde{\Sigma_i}$ la matrice de Sylvester d'indice i associée à $a\,u$ et $b\,v$ (voir par exemple [15, chapitre 7]). Ce sont toutes deux des matrices à $m+n-2i$ colonnes et $m+n-i$ lignes. Par définition, les sous-résultants $\mathrm{SousRes}_i(a\,u, b\,v)$ et $\mathrm{SousRes}_i(u,v)$ sont respectivement égaux aux déterminants polynomiaux des matrices $\widetilde{\Sigma_i}$ et Σ_i [6, définition 6.1.4 p.172]. Autrement dit, si pour tout k compris entre 0 et i, on note A_k et $\widetilde{A}_k$ les matrices carrées $(m+n-2i) \times (m+n-2i)$ constituées respectivement des $m+n-2i-1$ premières lignes et de la $m+n-i-k^{\text{ème}}$ ligne de Σ_i et des $m+n-2i-1$ premières lignes et de la $m+n-i-k^{\text{ème}}$ ligne de $\widetilde{\Sigma_i}$ alors :

$$\mathrm{SousRes}_i(u,v) = \sum_{j=0}^{i} det(A_j)\, T^j, \mathrm{SousRes}_i(a\,u, b\,v) = \sum_{j=0}^{i} det(\widetilde{A}_j)\, T^j.$$

Or, par construction, chaque matrice $\widetilde{A}_j$ $(0 \leq j \leq i)$ est constituée de $n-i$ colonnes de coefficients de $a\,u$ et de $m-i$ colonnes de coefficients de $b\,v$. Par suite, par bilinéarité du déterminant, on obtient :

$$\mathrm{SousRes}_i(a\,u, b\,v) = \sum_{j=0}^{i} a^{n-i}\, b^{m-i} det(A_j)\, T^j = a^{n-i}\, b^{m-i} \sum_{j=0}^{i} det(A_j)\, T^j.$$

D'où le résultat.◇

La preuve de la proposition suivante nécessite d'adopter quelques notations.

Notation. Soit $\widetilde{K}$ une clôture algébrique de K. On note $V(g)$ la variété affine de $\widetilde{K}$ définie par l'idéal engendré par g. De plus, étant donné un idéal $\mathcal{I}$ de $K[a]$, on désigne par $V_g(\mathcal{I})$ la sous-variété de $V(g)$ définie par l'image de l'idéal $\mathcal{I}$ dans l'anneau $\dfrac{K[a]}{\sqrt{\langle g \rangle}} = \dfrac{K[a]}{\langle g \rangle}$ (le polynôme g est sans carré par hypothèse).
Par ailleurs, étants donnés deux polynômes u,v à coefficients dans $K[a]$, on note $S(u,v)$ le sous-résultant de u et v non identiquement nul (modulo g) de plus bas degré.

Proposition 4.1 *Si les méthodes de la clôture constructible dynamique et de S.A. Abramov, K.Yu. Kvashenko déterminent les mêmes pgcd de p et de q en fonction de a alors :*

$$\langle \sqrt{lc(p)}\, \sqrt{lc(q)}\, \sqrt{lc(S(p,q))} \rangle = \langle \sqrt{pgcd(lc(\frac{p}{C_p}), lc(\frac{q}{C_q}))}\, \sqrt{lc(S(p,q))} \rangle \pmod{\langle g \rangle}.$$

Preuve. Notons Z_1 et Z_2 les ensembles contructibles que définissent les contraintes subies par le paramètre a respectivement dans les méthodes de la clôture constructible dynamique et de S.A. Abramov, K.Yu. Kvashenko en chaque nœud de la première branche du calcul. Avant l'étape des sous-résultants, il s'agit donc dans la première méthode de la branche où $\mathrm{lc}(p)$ et $\mathrm{lc}(q)$ sont supposés non nuls. De même, dans la seconde, il s'agit de la branche où C_p,C_q et $\mathrm{pgcd}(\mathrm{lc}(\frac{p}{C_p}),\mathrm{lc}(\frac{q}{C_q}))$ sont supposés non nuls. Ces deux ensembles sont donc égaux à :

$$Z_1 = V(g) - V(\mathrm{lc}(p)\,\mathrm{lc}(q)),\, Z_2 = V(g) - V(\mathrm{pgcd}((\mathrm{lc}(\frac{p}{C_p}),\mathrm{lc}(\frac{q}{C_q}))\,C_p\,C_q).$$

Par ailleurs, les sections 2 et 3 ont montré que le traitement des sous-résultants est identique dans les deux méthodes. D'autre part, il est clair que $S(\frac{p}{C_p},\frac{q}{C_q})$ et $S(p,q)$ sont deux sous-résultants de même indice. De plus, par construction de l'algorithme des sous-résultants, cette première branche se termine en imposant la contrainte $\mathrm{lc}(S(p,q)) \neq 0$ ou $\mathrm{lc}(S(\frac{p}{C_p},\frac{q}{C_q})) \neq 0$. Ainsi, à la fin de la première branche, on a :

$$Z_1 = V(g) - V(\mathrm{lc}(p)\,\mathrm{lc}(q)\,\mathrm{lc}(S(p,q)))$$

dans la méthode de la clôture constructible dynamique et

$$Z_2 = V(g) - V(\mathrm{pgcd}(\mathrm{lc}(\frac{p}{C_p}),\mathrm{lc}(\frac{q}{C_q}))\,C_p\,C_q\,\mathrm{lc}(S(\frac{p}{C_p},\frac{q}{C_q})))$$

dans l'algorithme de S.A. Abramov, K.Yu. Kvashenko. De plus, d'après le lemme précédent, il existe deux entiers naturels non nuls α,β tels que $S(p,q) = C_p^\alpha C_q^\beta S(\frac{p}{C_p},\frac{q}{C_q})$. D'où l'égalité :

$$V(C_p\,C_q\,\mathrm{lc}(S(\frac{p}{C_p},\frac{q}{C_q}))) = V(\mathrm{lc}(S(p,q))).$$

Or, si les deux méthodes renvoient en sortie un même résultat, cela implique en particulier que leur première branche est identique. Ainsi, la loi subie par le paramètre a est identique et définit donc dans chacune d'entre elles un même ensemble contructible. Autrement dit, les ensembles Z_1 et Z_2 doivent être égaux. Ce qui est équivalent à :

$$V_g(\mathrm{lc}(p)\,\mathrm{lc}(q)\,\mathrm{lc}(S(p,q))) = V_g(\mathrm{pgcd}(\mathrm{lc}(\frac{p}{C_p}),\mathrm{lc}(\frac{q}{C_q}))\,\mathrm{lc}(S(p,q))).$$

Il suffit alors d'appliquer le Nullstellensatz [3, théorème 5 p.237] pour conclure.◇

4.2 Exemples

On se propose de comparer les deux méthodes sur quatre exemples de calculs de pgcd (dont trois issus de [2]).

4.2.1 Exemple 1

Il s'agit de calculer le pgcd des polynômes $p = X^2 + (a-1)\,X + a$ et $q = (a+1)X^2 + 2\,a$.
La méthode de la clôture constructible dynamique donne le résultat suivant (où le symbole ? représente l'indéterminée X) :

```
                                        2
[value is 1 in case a = - 1, value is ?  + 1 in case a = 1,
                           2   2
 value is ? + 3a in case a  + - a = 0,
                             3
                                      2   2
 value is 1 in case a /= - 1 , a /= 1 , a  + - a /= 0]
                                        3
```

Ce qui signifie que :

- si $a = -1$ alors $\mathrm{pgcd}(p,q) = 1$;
- si $a = 1$ alors $\mathrm{pgcd}(p,q) = X^2 + 1$;
- si $a^2 + \frac{2}{3}\,a = 0$ alors $\mathrm{pgcd}(p,q) = X + 3\,a$;

– si $\left\{ \begin{array}{l} a \neq -1 \\ a \neq 1 \\ a^2 + \frac{2}{3}\,a \neq 0 \end{array} \right.$ alors $\mathrm{pgcd}(p,q) = 1$.

L'algorithme de S.A. Abramov, K.Yu. Kvashenko donne pour résultat :

– si $a = 1$ alors $\mathrm{pgcd}(p,q) = X^2 + 1$;

– si $a^2 + \frac{2}{3}\,a = 0$ alors $\mathrm{pgcd}(p,q) = X + 3\,a$;

– si $\left\{ \begin{array}{l} a \neq 1 \\ a^2 + \frac{2}{3}\,a \neq 0 \end{array} \right.$ alors $\mathrm{pgcd}(p,q) = 1$.

Ainsi, ils obtiennent un résultat différent (un cas de moins) de celui obtenu avec la méthode de la clôture constructible dynamique. Cette situation était-elle prévisible ?

Par définition, on a $\mathrm{lc}(p) = 1, \mathrm{lc}(q) = a + 1, C_p = 1$ et $C_q = 1$. De plus, le résultant de p et q est non nul, égal à $a(3a+2)(a-1)^2$. D'où :

$$\begin{array}{rcl} \langle \sqrt{\mathrm{lc}(p)}\,\sqrt{\mathrm{lc}(q)}\,\sqrt{\mathrm{lc}(S(p,q))} \rangle & = & \langle (a+1)a(3a+2)(a-1) \rangle \\ & \neq & \\ \langle \sqrt{\mathrm{pgcd}(\mathrm{lc}(\frac{p}{C_p}),\mathrm{lc}(\frac{q}{C_q}))}\,\sqrt{\mathrm{lc}(S(p,q))} \rangle & = & \langle a(3a+2)(a-1) \rangle. \end{array}$$

Le résultat obtenu est donc bien conforme à la proposition 4.1.

4.2.2 Exemple 2

Il s'agit de calculer le pgcd des polynômes $p = a\,X^3 + (a^3 - a + 1)\,X^2 + (a^2+2)\,X + 3\,a^2 - 3$ et $q = a\,X^3 + (a+1)\,X^2 + 4\,X + 3$.

La méthode de la clôture constructible dynamique donne le résultat suivant :

```
[value is ? + 3 in case a = 0,
           3    1        2          3            2
 value is ?  + (- a + 1)?  + 2a ? + - a in case a  - 2 = 0,
                2                   2
           2    1       3                    2
 value is ?  + --- ? + --- in case a /= 0 , a  - 2 /= 0]
              (a)     (a)
```

Ce qui signifie que :

– si $a = 0$ alors $\mathrm{pgcd}(p,q) = X + 3$;

– si $a^2 - 2 = 0$ alors $\mathrm{pgcd}(p,q) = X^3 + (\frac{1}{2}\,a + 1)\,X^2 + 2\,a\,X + \frac{3}{2}\,a$;

– si $\left\{ \begin{array}{l} a \neq 0 \\ a^2 - 2 \neq 0 \end{array} \right.$ alors $\mathrm{pgcd}(p,q) = X^2 + \frac{1}{a}\,X + \frac{3}{a}$.

L'algorithme de S.A. Abramov, K.Yu. Kvashenko retourne cette fois un résultat identique. Que donne la proposition 4.1 sur cet exemple ?

On a $\mathrm{lc}(p) = a, \mathrm{lc}(q) = a, C_p = 1$ et $C_q = 1$. Les sous-résultants de p et q d'indices 0,1 sont nuls et le sous-résultant d'indice 2 est égal à $-a(a^2-2)(aX^2 + X + 3)$. Ainsi :

$$\begin{array}{rcl} \langle \sqrt{\mathrm{lc}(p)}\,\sqrt{\mathrm{lc}(q)}\,\sqrt{\mathrm{lc}(S(p,q))} \rangle & = & \langle a(a^2-2) \rangle \\ & = & \\ \langle \sqrt{\mathrm{pgcd}(\mathrm{lc}(\frac{p}{C_p}),\mathrm{lc}(\frac{q}{C_q}))}\,\sqrt{\mathrm{lc}(S(p,q))} \rangle & = & \langle a(a^2-2) \rangle. \end{array}$$

La condition algébrique de la proposition 4.1 est donc bien vérifiée.

4.2.3 Exemple 3

Il s'agit de calculer le pgcd des polynômes $p = X^4 - a^2\,X^2 + (a^2+a)\,X - a^3 - a^2$ et $q = a\,X^3 + (-a^2+1)\,X^2 - a\,X$ sous la contrainte $g(a) = a^7 + a^6 + 3\,a^5 + 4\,a^4 + 2\,a^2 - 3 = 0$.

La méthode de la clôture constructible dynamique donne le résultat suivant :

```
           3
[value is ?  - ? in case a = - 1,
           2     3                        4
 value is ?  + (a  - a + 1)? - 1 in case a  + a - 1 = 0,
```

```
                                2
  value is ? - a in case a  + 3 = 0]
```

Ce qui signifie que :

- si $a = -1$ alors $\text{pgcd}(p,q) = X^3 - X$;
- si $a^4 + a - 1 = 0$ alors $\text{pgcd}(p,q) = X^2 + (a^3 - a + 1)\,X - 1$;
- si $a^2 + 3 = 0$ alors $\text{pgcd}(p,q) = X - a$.

Le résultat retourné par l'algorithme de S.A. Abramov, K.Yu. Kvashenko est, *après réduction*, identique. De plus, on a $\text{lc}(p) = 1$,$\text{lc}(q) = a$,$C_p = 1$ et $C_q = 1$. Le résultant de p et q est identiquement nul et le sous-résultant d'indice 1 est égal (modulo g) à :

$$S(p,q) = (a-3)(a+1)(a^4+a-1)X + 3(a^4+a-1)(a+1)^2.$$

D'autre part, les polynômes $\text{lc}(q)$ et g étant premiers entre eux, on en déduit que $\text{lc}(q)$ est inversible modulo g. Ce qui implique en particulier :

$$\begin{aligned} \langle \sqrt{\text{lc}(p)}\,\sqrt{\text{lc}(q)}\,\sqrt{\text{lc}(S(p,q))}\rangle &= \langle a(a-3)(a+1)(a^4+a-1)\rangle \\ &= \langle (a-3)(a+1)(a^4+a-1)\rangle \pmod{\langle g\rangle}. \end{aligned}$$

Par ailleurs, on a :

$$\langle \sqrt{\text{pgcd}(\text{lc}(\frac{p}{C_p}),\text{lc}(\frac{q}{C_q}))}\,\sqrt{\text{lc}(S(p,q))}\rangle = \langle (a-3)(a+1)(a^4+a-1)\rangle.$$

La condition algébrique de la proposition 4.1 est donc bien vérifiée.

4.2.4 Exemple 4

On termine cette section par un exemple qui montre que la proposition 4.1 donne une condition nécessaire mais non suffisante. Il s'agit de calculer le pgcd des polynômes $p = a(a-1)(a-2)\,X^2 + a(a-2)(a+1)\,X + a(a+1)(a+3)$ et $q = (a-2)(a+2)(a-1)\,X + (a-2)(a-1)(a+2)(a+3)$.
La méthode de la clôture constructible dynamique donne le résultat suivant :

```
[value is 4? + 12 in case a = 0, value is 30 in case a = 2,
                                                      2
 value is - 2? + 8 in case a = 1, value is - 24?  - 8? + 2

 in case a = - 2,
                                4     3     2
 value is ? + a + 3 in case a  + 2a  - 8a  - 6a + 27 = 0,

                                   2
  value is 1 in case a /= 0 , a  - 3a + 2 /= 0 , a /= - 2 ,
     4     3     2
    a  + 2a  - 8a  - 6a + 27 /= 0
 ]
```

Ce qui signifie que :

- si $a = 0$ alors $\text{pgcd}(p,q) = 4X + 12$;
- si $a = 2$ alors $\text{pgcd}(p,q) = 30$;
- si $a = 1$ alors $\text{pgcd}(p,q) = -2X + 8$;
- si $a = -2$ alors $\text{pgcd}(p,q) = -24X^2 - 8X + 2$;
- si $a^4 + 2a^3 - 8a^2 - 6a + 27 = 0$ alors $\text{pgcd}(p,q) = X + a + 3$;
- si $\begin{cases} a \neq 0 \\ a^2 - 3a + 2 \neq 0 \\ a \neq -2 \\ a^4 + 2a^3 - 8a^2 - 6a + 27 \neq 0 \end{cases}$ alors $\text{pgcd}(p,q) = 1$.

En revanche, si on applique (à la main) l'algorithme de S.A. Abramov, K.Yu. Kvashenko, on obtient le résultat suivant :

- si $a = 0$ alors $\text{pgcd}(p,q) = X + 3$;

- si $(a-2)(a+2)(a-1) = 0$ alors pgcd(p,q) $= p$. Plus précisement, après réduction, il apparaît que : pgcd(p,q) $= (-2a^2+6a-4)X^2+(2a-4)X+5a^2+7a-4$;
- si $a^4+2a^3-8a^2-6a+27=0$ alors pgcd(p,q) $= X+a+3$;
- si $\begin{cases} a \neq 0 \\ (a-2)(a+2)(a-1) \neq 0 \\ a^4+2a^3-8a^2-6a+27 \neq 0 \end{cases}$ alors pgcd(p,q) $= 1$.

On obtient donc deux cas de moins. Par ailleurs, on a lc(p) $= a(a-1)(a-2)$,lc(q) $= (a-2)(a+2)(a-1)$,$C_p = a$ et $C_q = (a-2)(a+2)(a-1)$. De plus, le résultant de p et q est non nul, égal à $a(a-2)^2(a+2)^2(a-1)^2(a^4+2a^3-8a^2-6a+27)$. Ce qui implique les égalités :

$$\langle\sqrt{\mathrm{lc}(p)}\,\sqrt{\mathrm{lc}(q)}\,\sqrt{\mathrm{lc}(S(p,q))}\rangle = \langle a(a-2)(a+2)(a-1)(a^4+2a^3-8a^2-6a+27)\rangle$$

et :

$$\langle\sqrt{\mathrm{pgcd}(\mathrm{lc}(\frac{p}{C_p}),\mathrm{lc}(\frac{q}{C_q}))}\,\sqrt{\mathrm{lc}(S(p,q))}\rangle = \langle a(a-2)(a+2)(a-1)(a^4+2a^3-8a^2-6a+27)\rangle.$$

Ainsi, bien qu'on obtienne deux résultats différents, le critère algébrique de la proposition 4.1 est vérifié. Celui-ci constitue donc seulement une condition nécessaire.

4.3 Une première conclusion

Etants donnés deux polynômes p,q à une indéterminée, à coefficients dans un corps et à un paramètre a, les deux algorithmes étudiés dans les sections précédentes déterminent le pgcd de p et q en fonction des valeurs de a. Le domaine d'application de la méthode de la clôture constructible dynamique est plus vaste. En effet, on peut l'utiliser avec un nombre quelconque de paramètres et avec une contrainte initiale de type exception. D'autre part, il ressort de cette étude que les deux algorithmes ne diffère que par le traitement des coefficients dominants et des contenus des polynômes. Or, il apparaît que cette opération est plus fine dans le second algorithme. Il est alors raisonnable de penser que celui-ci doit en pratique engendrer moins de scindages que le premier. On peut noter que les différents exemples testés abondent en ce sens.
Par ailleurs, la section 3 a montré que l'algorithme de S.A. Abramov, K.Yu. Kvashenko s'inscrit naturellement dans le cadre de l'évaluation dynamique. Ainsi, un travail intéressant serait d'implanter leur algorithme au sein des programmes de la clôture constructible dynamique. En effet, la méthode de S.A. Abramov, K.Yu. Kvashenko serait alors de fait applicable aussi avec une contrainte initiale de type exception. De plus, on peut envisager de tirer profit du contexte "dynamique" pour étendre leur algorithme à un nombre de paramètres quelconque. C'est l'objet de la section suivante.

5 Le package Axiom ABRAMOV

Cette section est consacrée à la généralisation et à l'implantation de la méthode de S.A. Abramov, K.Yu. Kvashenko au sein des programmes de T. Gómez-Díaz. On souhaite étendre ainsi leur méthode au calcul de pgcd de deux polynômes p et q à une indéterminée, à coefficients dans $K[a_1,\ldots,a_n]$ où K est un corps et $a_1,\ldots,a_n$ n paramètres. De plus, on veut que les différents paramètres puissent être soumis initialement à une contrainte de type *quelconque*.
On donne et commente tout d'abord le code du package Axiom ABRAMOV qui résulte de ce travail dans la première partie. On propose alors différents exemples (dont les quatre étudiés précédemment) d'utilisation de ce package dans la seconde moitié de cette section.

5.1 Implantation Axiom

Le code du package ABRAMOV (commenté dans les sections 5.1.1, 5.1.2 et 5.1.3) est le suivant :

```
)abbrev package ABRAMOV DynamicAbramovPackage

--% DynamicAbramovPackage

DynamicAbramovPackage(K): Exports ==  Implementation where

  K: Field
```

```
DB   ==> DynamicBuildField(K)
CL   ==> InternalDynamicConstructibleClosure DB
Pol ==> DynamicUnivariatePolynomial(CL)
Fra  ==> DynamicConstructibleFraction(CL)

Exports == with

  gcdCl: (CL,CL) -> CL
    ++ gcdCl(x1,x2) returns :
    ++ * 1 if x1 or x2 belongs to the field K;
    ++ * the gcd of the numerators of the dynamic fractions
    ++   associated with x1 and x2 otherwise.

  content: Pol -> CL
    ++ content(p) computes the content of the dynamic
    ++ polynomial p.

  gcdAbramov: (Pol,Pol) -> Pol
    ++ gcd(p,q) returns the gcd of p and q by an extended
    ++ version of S.A.~Abramov, K.Yu.~Kvashenko algorithm.
    ++ Notes:
    ++ * this gcd is computed by subresultants;
    ++ * gcd(0,0)= 0.

Implementation == add

  -- declaration

    p,q: Pol        -- input polynomials
    x1,x2: CL

    gcdCl(x1,x2) ==
      rawRetractable?(x1)$CL => 1::CL
      rawRetractable?(x2)$CL => 1::CL
      f:Pol:= numerator(constructibleFraction(x1)$CL)$Fra
      g:Pol:= numerator(constructibleFraction(x2)$CL)$Fra
      r:Pol:= gcdAbramov(f,g)
      construct(x1,r::Fra)$CL

    content(p) ==
      lp:List CL:= rawCoefficients(p)$Pol
      empty? lp => 0::CL
      #lp = 1 => first lp
      cp:CL:= first lp
      lp:= rest lp
      for i in 1..#lp repeat
        cp:= gcdCl(lp.i, cp)
        rawRetractable?(cp)$CL => return 1::CL
      cp

    gcdAbramov(p,q) ==
      p:= reduce p
      q:= reduce q
      rawZero?(p)$Pol => q
      rawZero?(q)$Pol => p
      Cp:CL:= content(p)
      Cq:CL:= content(q)
```

```
(Cp = 0)$CL =>
  (Cq = 0)$CL => 0::Pol
  (q / Cq)$Pol
p:= (p / Cp)$Pol
(Cq = 0)$CL => p
q:= (q / Cq)$Pol
p:= reduce p
q:= reduce q
rawDegree(p) = 0 => 1::Pol
rawDegree(q) = 0 => 1::Pol
lcp:CL:= rawLeadingCoefficient(p)$Pol
lcq:CL:= rawLeadingCoefficient(q)$Pol
delta:CL:= gcdCl(lcp,lcq)
(delta = 0)$CL => gcdAbramov(p,q)
p1:Pol:= reduce p
q1:Pol:= reduce q
dp:= rawDegree(p1)$Pol
dq:= rawDegree(q1)$Pol
if dp < dq then
  p2:Pol:= p1
  p1:= q1
  q1:= p2
lpol:= Suite_reste_ss_res(p1,q1)$Pol
until r ^=$CL 0 repeat
  p1:= lpol.first
  lpol:= lpol.rest
  r:CL:= rawLeadingCoefficient(p1)$Pol
(p1 / r)$Pol
```

Soient n paramètres notés $a_1,\ldots,a_n$. Le but de ce package est de déterminer le pgcd de deux polynômes p,q à une indéterminée X et à coefficients dans l'anneau $(K[a_1,\ldots,a_n])[X]$. Chacun des paramètres $a_i\,(1 \leq i \leq n)$ peut être soumis initialement à une loi de type quelconque. L'ensemble de ces lois forment une tour de T. Gómez-Díaz [6, section 4.1] à laquelle on peut associer un anneau L_n. Ainsi, les polynômes p et q peuvent s'interpréter comme deux éléments de l'anneau $L_n[X]$ sans dénominateur (i.e. les coefficients de p et q sont des éléments de L_n sans dénominateur).

Notation. On se donne désormais $n+1$ indéterminées $X_1,\ldots,X_n,X_{n+1}$ avec $X_{n+1} = X$.

Le lemme suivant sera utile dans la section 5.1.3 afin d'établir la pertinence de la fonction `gcdAbramov`.

Lemme 5.1 *Soient un entier $i\,(1 \leq i \leq n)$ et un polynôme $f \in L_i[X_{i+1}]$ sans dénominateur. Alors les coefficients de f sont des polynômes de $L_{i-1}[X_i]$ sans dénominateur.*

Preuve. On écrit f sous la forme $f = \sum_{k=0}^{d} r_k X_{i+1}^k$. Pour tout $0 \leq k \leq d$, il existe un entier $i_k\,(0 \leq i_k \leq i)$ tel que r_k soit un élément de l'anneau L_{i_k}. Puisque f est sans dénominateur, on déduit de [6, section 4.4.4] que chacun des numérateurs de r_k est un polynôme de $L_{i_k-1}[X_{i_k}]$. Or, d'après [6, proposition 1.2.3 p.43], chacun des anneaux L_{i_k-1} s'injecte dans L_{i-1}. Le résultat est alors immédiat.$\diamond$

5.1.1 La fonction `gcdCl`

La compréhension du code de cette fonction nécessite d'effectuer au préalable quelques rappels sur la représentation au sens d'Axiom des éléments du domaine `CL` [6, section 4.4.5 p.143],[9].
A chaque élément de la clôture constructible dynamique est associé un niveau (codé par un entier naturel). Ainsi, un élément x de `CL` est de niveau 0 si celui-ci est un élément du corps de base `K`. Il est alors représenté simplement comme un élément de ce corps. Il faut noter qu'il existe une fonction qui permet de savoir si tel est le cas. Il s'agit de la fonction `rawRetractable?` du domaine `CL` qui renvoie le booléen `true` si et seulement si elle est appliquée à un élément de niveau nul. Dans le cas contraire, l'élément x est de niveau strictement positif et est représenté par un enregistrement à deux champs :

$$[lev, fra]$$

où lev est le niveau de x et où fra est la fraction dynamique associée à x.

La méthode de S.A. Abramov, K.Yu. Kvashenko, précisément l'étape 3 de leur algorithme, nécessite l'implantation d'une fonction qui calcule le contenu d'un polynôme dynamique. Or, les coefficients d'un polynôme dynamique ne sont pas des éléments de `Pol` mais des éléments du domaine `CL`. Le problème est qu'il n'existe pas d'opération de pgcd dans le domaine de la clôture constructible dynamique des programmes de T. Gómez-Díaz. C'est le rôle de la fonction `gcdCl`. Plus précisément, étants donnés deux éléments x_1,x_2 du domaine `CL`, la fonction `gcdCl` retourne :

- l'élément 1 de `CL` si x_1 ou x_2 est de niveau nul, c'est-à-dire, si x_1 ou x_2 est un élément du corps de base K;
- le pgcd des numérateurs des fractions dynamiques associées à x_1 et x_2.

Ce qui renvoie respectivement aux instructions :

```
rawRetractable?(x1)$CL => 1::CL
rawRetractable?(x2)$CL => 1::CL
```

et :

```
f:Pol:= numerator(constructibleFraction(x1)$CL)$Fra
g:Pol:= numerator(constructibleFraction(x2)$CL)$Fra
r:Pol:= gcdAbramov(f,g)
```

Le premier cas est naturel. Par exemple, dans l'optique du calcul du contenu d'un polynôme p, il s'agit de retourner un contenu égal à 1 dès que l'un des coefficients de p appartient au corps de base K. Le second amène plusieurs commentaires. Il faut tout d'abord noter que x_1 et x_2 sont nécessairement de même niveau. En effet, la structure récursive de la représentation Axiom des polynômes dynamiques fait que leurs coefficients sont tous de même niveau. Par ailleurs, étant donné un élément x de `CL` de niveau > 0, l'instruction :

```
constructibleFraction(x)
```

retourne la fraction dynamique associée à x. Autrement dit, si x est codé par l'enregistrement $[lev,fra]$, alors cette fonction retourne fra. L'instruction :

```
numerator(constructibleFraction(x))
```

permet alors d'avoir accès au numérateur de cette fraction. Celui-ci est un élément du domaine `Pol`. Ainsi, le pgcd de deux coefficients de p représentés par des éléments de `CL` est ramené au calcul de pgcd de deux polynômes dynamiques (`f` et `g`). On calcule alors leur pgcd à l'aide de la fonction `gcdAbramov` (voir section 5.1.3). Le problème est que le résultat de `gcdAbramov` est un polynôme dynamique. Or, la fonction `gcdCl` doit retourner un élément de `CL`. Aussi, il faut coercer le polynôme `r` calculé par `gcdAbramov(f,g)` en un élément de `CL`, de même niveau que x_1 et x_2. C'est le rôle de l'instruction :

```
construct(x1,r::Fra)$CL
```

Remarque. Il faut noter que si l'algorithme de la fonction `gcdAbramov` est correct et termine en un nombre fini d'étapes alors il en est de même de la fonction `gcdCl`.

Par ailleurs, soient deux éléments x_1 et x_2 du domaine `CL` de niveau $0 < j \leq n$. Si on applique `gcdCl` au couple (x_1,x_2), cette fonction calcule le pgcd des numérateurs `f` et `g` des fractions dynamiques associées à x_1 et x_2 à l'aide de `gcdAbramov`. D'après [6, section 4.4.4], les polynômes `f` et `g` peuvent s'interpréter comme des éléments de $L_{j-1}[X_j]$. Or, on montre dans la section 5.1.3 que la fonction `gcdAbramov` s'applique à tout couple de polynômes (p,q) de $L_i[X_{i+1}] \times L_i[X_{i+1}]$ avec $0 \leq i \leq n$. Cela justifie alors son utilisation dans le calcul de pgcd de `f` et `g` et plus généralement, son utilisation au sein de `gcdCl`.

5.1.2 La fonction `content`

Etant donné un polynôme dynamique p, la fonction `content` calcule, comme son nom l'indique, son contenu qui est un élément du domaine `Cl`. L'instruction :

```
lp:List CL:= rawCoefficients(p)$Pol
```

retourne la liste des coefficients du polynôme p. Il s'agit donc d'une liste d'éléments de `CL`. La partie du code suivante :

```
empty? lp => 0::CL
#lp = 1 => first lp
```

est claire. La fonction `content` retourne 0 si p est identiquement nul et $\mathrm{lc}(p)$ si p est un polynôme constant non nul.
On détermine alors le pgcd des coefficients de p à l'aide d'une simple boucle `for` (et de la fonction `gcdCl` étudiée précédemment). Celle-ci s'arrête si le pgcd `cp` des $i+1$ premiers coefficients de p est un élément de K. Dans ce

cas, la fonction retourne l'élément 1 de `CL`. Dans le cas contraire, après la boucle `for`, le contenu recherché est `cp` :

```
cp:CL:= first lp
lp:= rest lp
for i in 1..#lp repeat
  cp:= gcdCl(lp.i, cp)
  rawRetractable?(cp)$CL => return 1::CL
cp
```

Remarque. On déduit de ce qui précède et de la remarque de la section précédente que si l'algorithme de la fonction `gcdAbramov` est correct et se termine en un nombre fini d'étapes alors il en est de même de la fonction `content`.

5.1.3 La fonction `gcdAbramov`

La fonction `gcdAbramov` constitue la partie principale de ce package. Il s'agit de l'implantation de la généralisation de l'algorithme étudié dans la section 3. Ainsi, l'étape 1 qui élimine les cas triviaux $p \equiv 0$, $q \equiv 0$ est implantée de la manière suivante :

```
p:= reduce p
q:= reduce q
rawZero?(p)$Pol => q
rawZero?(q)$Pol => p
```

On rappelle que la fonction `rawZero?` est un test d'égalité non dynamique. Elle renvoie le booléen `true` si et seulement si p est identiquement nul.
La seconde étape est implantée ainsi :

```
Cp:CL:= content(p)
Cq:CL:= content(q)
(Cp = 0)$CL =>
  (Cq = 0)$CL => 0::Pol
  (q / Cq)$Pol
p:= (p / Cp)$Pol
(Cq = 0)$CL => p
q:= (q / Cq)$Pol
p:= reduce p
q:= reduce q
```

Elle suit exactement le schéma des arbres de scindages proposés dans la section 3. La seule subtilité consiste à réduire les polynômes p et q après les deux tests d'égalité dynamique. Bien entendu, le calcul des nouvelles lois est effectué cette fois *automatiquement*.
Le code de la troisième étape qui gère les cas triviaux $\deg(p) = 0$, $\deg(q) = 0$ est :

```
rawDegree(p) = 0 => 1::Pol
rawDegree(q) = 0 => 1::Pol
```

Comme son nom l'indique, la fonction `rawDegree` permet de déterminer le degré d'un polynôme dynamique (sans effectuer de scindage). Par ailleurs, l'instruction `1::Pol` permet de retourner un polynôme dynamique égal à 1.
L'implantation de la quatrième étape suit elle aussi exactement le schéma proposé dans la section 3 :

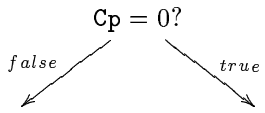
```
lcp:CL:= rawLeadingCoefficient(p)$Pol
lcq:CL:= rawLeadingCoefficient(q)$Pol
delta:CL:= gcdCl(lcp,lcq)
if (delta = 0)$CL then gcdAbramov(p,q)
p1:Pol:= reduce p
q1:Pol:= reduce q
```

Les coefficients dominants des polynômes p et q, symbolisés respectivement par `lcp` et `lcq`, sont calculés (sans effectuer de scindage) à l'aide d'une fonction appelée `rawLeadingCoefficient` du domaine des polynômes dynamiques `Pol`.
Enfin, l'implantation du calcul des sous-résultants reprend exactement la partie du code correspondante de la fonction `gcd`.

Ainsi, il ressort de ce qui précède que la fonction `gcdAbramov` *suit exactement* le schéma de l'algorithme de S.A. Abramov, K.Yu. Kvashenko. Il reste à montrer que c'est une version généralisée de leur algorithme qui se termine en un nombre fini d'étapes. C'est l'objet de la proposition suivante.

Proposition 5.1 *Soient un entier* $i\,(0 \leq i \leq n)$ *et deux polynômes* $p,q \in L_i[X_{i+1}]$ *sans dénominateur. La fonction* `gcdAbramov(p,q)` *retourne en un nombre fini d'étapes un pgcd de* p *et* q.

Preuve. Montrons tout d'abord que l'algorithme se termine en un nombre fini d'étapes. Il y a deux appels récursifs. Le premier est `(delta = 0)$CL =>gcdAbramov(p,q)`. Dans ce cas, le pgcd des coefficients dominants $\mathrm{lc}(p)$ et $\mathrm{lc}(q)$ de p et q est nul. Ce qui implique $\mathrm{lc}(p) = \mathrm{lc}(q) = 0$. Ainsi, on exécute de nouveau la fonction avec deux polynômes de degrés strictement inférieurs. Le second appel récursif est `dq1 = 0 =>gcdAbramov(p1,q1)`. On applique cette fois l'algorithme avec un polynôme q_1 de degré nul. Ainsi, le degré des polynômes décroît strictement à chaque appel récursif : l'algorithme se termine bien.
Montrons alors que celui-ci est correct. On raisonne par récurrence sur $i\,(0 \leq i \leq n)$. Le cas $i = 0$ est facile. En effet, les polynômes p et q sont alors à coefficients dans K. Par construction des fonctions `gcdCl` et `content`, il est facile de vérifier que `Cp`, `Cq` et `delta` sont égaux à 1. En l'absence de paramètres, les étapes de réduction sont obsolètes. Ainsi, la fonction `gcdAbramov` est un simple calcul de pgcd de deux polynômes en une indéterminée à coefficients dans un corps.
Supposons le résultat vrai jusqu'à l'ordre $i-1$. On se donne deux polynômes p et q de $L_i[X_{i+1}]$ sans dénominateur. D'après le lemme 5.1, les numérateurs des coefficients des polynômes p et q sont des éléments de $L_{i-1}[X_i]$ sans dénominateur. Par hypothèse de récurrence, la fonction `content` appliquée à p et à q retourne bien leur contenu respectif. De même, la fonction `gcdCl` appliquée à $\mathrm{lc}(p)$ et $\mathrm{lc}(q)$ retourne bien leur pgcd. Examinons alors le déroulement de l'algorithme. On effectue le test d'égalité :



Dans la branche de droite, on effectue alors le test d'égalité `Cq` $= 0$?. Ainsi, si `Cq` est nul, les deux polynômes sont nuls et la fonction renvoie 0. Si `Cq` est non nul alors la fonction renvoie q (puisque $p = 0$). Dans les deux cas, il s'agit bien d'un pgcd de p et de q.
Dans la branche de gauche (i.e. `Cp` $\neq 0$), on effectue le test d'égalité `Cq` $= 0$?. Le raisonnement est identique : si `Cq` est nul alors la fonction renvoie p qui est bien dans ce cas un pgcd de p et q.
On suppose donc que `Cp` et `Cq` sont non nuls (dans la branche courante). D'après [6, théorème 2.3.1 p.86], ce sont donc des éléments inversibles de L_{i-1}. Ainsi, si $\deg(p) = 0$ alors le polynôme p est inversible dans L_{i-1} (puisque p est égal à son contenu). Un pgcd de p et q est donc 1, ce qui est la valeur retournée par la fonction dans ce cas :

```
rawDegree(p) = 0 => 1::Pol
```

Le cas $\deg(q) = 0$ est analogue. Soit δ le pgcd de $\mathrm{lc}(p)$ et de $\mathrm{lc}(q)$. Si celui-ci est nul, on calcule de nouveau le pgcd de p et q avec cette contrainte supplémentaire. Dans le cas contraire, cela implique en particulier que $\mathrm{lc}(p)$ et $\mathrm{lc}(q)$ ne sont pas simultanément nuls. De plus, les deux polynômes p et q vérifient (à permutation près) $\deg(p) \geq \deg(q) > 0$. L'instruction :

```
Suite_reste_ss_res(p,q)
```

retourne alors par définition la liste de tous les sous-résultants d'indice $\leq \deg(q)$ de p et de q. Puisque $\delta \neq 0$, on déduit de [8, théorèmes 4.1 p.45] que l'algorithme des sous-résultants se spécialise bien. De plus [8, théorème 4.3 p.48] justifie son utilisation dans le calcul du pgcd de p et de q. La fonction `gcdAbramov` retourne donc bien le résultat souhaité.$\diamond$.

5.2 Exemples

Pour utiliser le package ABRAMOV, il suffit de suivre le schéma suivant :

```
K:= corps de base
DB:= DynamicBuildField(K)
CL:= InternalDynamicConstructibleClosure DB

AB:= DynamicAbramovPackage(K)
Pol:= DynamicUnivariatePolynomial(CL)
X:Pol:= monomial(1,1)$Pol
```

```
CP:= DynamicConstructibleControlPackage(K,Pol)

dynamicGCD():Pol ==
  ++ Introduction des parametres
  ++ a1,...,an et des contraintes initiales
  p:Pol:=
  q:Pol:=
  gcdAbramov(p,q)$AB

allCases(dynamicGCD)$CP
```

On reporte tout d'abord ci-dessous l'utilisation de ce package avec cinq exemples de calculs de pgcd de deux polynômes à *un* paramètre. Les quatre premiers sont exactement les quatre exemples étudiés dans la section précédente. On retrouve alors bien les résultats de [2] [3]. Le dernier calcul de pgcd (`exemple 5`) montre que l'on peut désormais utiliser l'algorithme de S.A. Abramov, K.Yu. Kvashenko avec une loi de type exception. Il s'agit de l'exemple 3 dans lequel on a remplacé la contrainte de type algébrique par une contrainte de type exception : $a - 1 \neq 0$.

```
-- exemple 1 --

              2                                                2   2
   [value is ?  + 1 in case a = 1, value is ? + 3a in case a  + - a = 0,
                                                                 3
                        2   2
    value is 1 in case a  + - a /= 0 , a /= 1]
                            3

-- exemple 2 --

   [value is ? + 3 in case a = 0,
              3    1        2          3            2
    value is ?  + (- a + 1)?  + 2a ? + - a in case a  - 2 = 0,
                   2                   2
              2    1       3                    2
    value is ?  + --- ? + --- in case a /= 0 , a  - 2 /= 0]
                  (a)     (a)

-- exemple 3 --

              3
   [value is ?  - ? in case a = - 1,
              2     3                        4
    value is ?  + (a  - a + 1)? - 1 in case a  + a - 1 = 0,
                            2
    value is ? - a in case a  + 3 = 0]

-- exemple 4 --

   [value is 4? + 12 in case a = 0,

                   2           2                  2
     value is (- 2a  + 6a - 4)?  + (2a - 4)? + 5a  + 7a - 4 in case
        3    2
       a  - a  - 4a + 4 = 0
     ,
```

3. On peut malgré tout remarquer que le pgcd retourné dans l'exemple 3 sous l'hypothèse $a^4 + a - 1 = 0$ est réduit contrairement au résultat présenté dans [2].

```
                                4     3     2
value is ? + a + 3 in case a  + 2a  - 8a  - 6a + 27 = 0,

                                  3    2
 value is 1 in case a /= 0 , a  - a  - 4a + 4 /= 0 ,
    4     3     2
   a  + 2a  - 8a  - 6a + 27 /= 0
 ]
```

Dans l'exemple suivant, il s'agit de calculer le pgcd des polynômes $p = X^4 - a^2\,X^2 + (a^2 + a)\,X - a^3 - a^2$ et $q = a\,X^3 + (-a^2 + 1)\,X^2 - a\,X$ sous la contrainte $g(a) = a + 1 \neq 0$. La méthode de la clôture constructible dynamique donne le résultat suivant :

```
            2
[value is ?  in case a = 0,
            2     3                          4
 value is ?  + (a  - a + 1)? - 1 in case a  + a - 1 = 0,
                                             4
 value is ? - a in case a /= - 1 , a /= 0 , a  + a - 1 /= 0]
```

Ainsi :

- si $a = 0$ alors pgcd$(p,q) = X^2$;
- si $a^4 + a - 1 = 0$ alors pgcd$(p,q) = X^2 + (a^3 - a + 1)X - 1$;
- si $\left\{ \begin{array}{l} a + 1 \neq 0 \\ a \neq 0 \\ a^4 + a - 1 \neq 0 \end{array} \right.$ alors pgcd$(p,q) = X - a$.

En revanche, notre implantation retourne :

```
-- exemple 5 --

            2     4    3      4               5    2
[value is ?  + (a  + a )? - a  - a in case a  + a  - a = 0,
                                      5    2
 value is ? - a in case a /= - 1 , a  + a  - a /= 0]
```

Ainsi :

- si $a^5 + a^2 - a = 0$ alors pgcd$(p,q) = X^2 + (a^4 + a^3)X - a^4 - a$;
- si $\left\{ \begin{array}{l} a + 1 \neq 0 \\ a^5 + a^2 - a \neq 0 \end{array} \right.$ alors pgcd$(p,q) = X - a$.

Elle retourne un cas de moins ($a = 0$) en ne factorisant pas le polynôme $a^5 + a^2 - a$.

On étudie alors deux exemples de pgcd de polynômes en une indéterminée et à *plusieurs* paramètres.
Ainsi, on souhaite tout d'abord déterminer le pgcd des polynômes $p = a\,X^2 + a\,X + 1$ et $q = b\,X^2 + b\,X + 1$ en fonction des deux paramètres a et b. La méthode de la clôture constructible dynamique donne le résultat suivant :

```
[value is 1 in case b = 0 and a = 0, value is 1 in case b /= 0 and a = 0,
 value is 1 in case b = 0 and a /= 0,
            2        1
 value is ?  + ? + --- in case b = a and a /= 0,
                  (a)
 value is 1 in case b /= 0 , b /= a and a /= 0]
```

c'est-à-dire :

- si $a = b = 0$ alors pgcd$(p,q) = 1$;
- si $a = 0$ et $b \neq 0$ alors pgcd$(p,q) = 1$;
- si $a \neq 0$ et $b = 0$ alors pgcd$(p,q) = 1$;
- si $a \neq 0$ et $b = a$ alors pgcd$(p,q) = X^2 + X + \frac{1}{a}$;
- si $\left\{ \begin{array}{l} b \neq 0, b \neq a \\ a \neq 0 \end{array} \right.$ alors pgcd$(p,q) = 1$.

En revanche, la méthode généralisée de S.A. Abramov, K.Yu. Kvashenko donne le résultat suivant :

```
[value is 1 in case any b and a = 0,
           2        1
 value is ?  + ? + --- in case b = a and a /= 0,
                  (a)
 value is 1 in case b /= a and a /= 0]
```

c'est-à-dire :

- si $a = 0$ alors pgcd$(p,q) = 1$;
- si $a \neq 0$ et $b = a$ alors pgcd$(p,q) = X^2 + X + \frac{1}{a}$;
- si $a \neq 0$ et $b \neq a$ alors pgcd$(p,q) = 1$.

Elle retourne donc deux cas de moins.
On s'intéresse enfin aux mêmes polynômes p et q mais cette fois sous la contrainte $a \neq 0$. On obtient les résultats suivants :

```
[value is 1 in case b = 0 and a /= 0,
           2        1
 value is ?  + ? + --- in case b = a and a /= 0,
                  (a)
 value is 1 in case b /= 0 , b /= a and a /= 0]
```

c'est-à-dire :

- si $a \neq 0$ et $b = 0$ alors pgcd$(p,q) = 1$;
- si $a \neq 0$ et $b = a$ alors pgcd$(p,q) = X^2 + X + \frac{1}{a}$;
- si $\left\{ \begin{array}{l} b \neq 0, b \neq a \\ a \neq 0 \end{array} \right.$ alors pgcd$(p,q) = 1$.

avec la méthode de la clôture constructible dynamique et :

```
           2        1
[value is ?  + ? + --- in case b = a and a /= 0,
                  (a)
 value is 1 in case b /= a and a /= 0]
```

c'est-à-dire :

- si $a \neq 0$ et $b = a$ alors pgcd$(p,q) = X^2 + X + \frac{1}{a}$;
- si $a \neq 0$ et $b \neq a$ alors pgcd$(p,q) = 1$.

avec la méthode généralisée de S.A. Abramov, K.Yu. Kvashenko. Cette dernière retourne donc un cas de moins.

6 Conclusion

L'étude menée dans les sections 3,4 amène naturellement à concevoir l'implantation de la méthode de S.A. Abramov, K.Yu. Kvashenko au sein des programmes de la clôture constructible dynamique. La réalisation de ce travail se présente sous la forme d'un package Axiom appelé ABRAMOV. Ses intérêts sont multiples. Tout d'abord, il élargit le domaine d'application de l'algorithme de S.A. Abramov, K.Yu. Kvashenko dans deux directions : le nombre de paramètres des polynômes et la nature des contraintes initiales. Ainsi, notre implantation de leur algorithme est valable pour un nombre de paramètre *quelconque* et chacun de ces paramètre peut être soumis à une contrainte de type *quelconque* : une équation, une ou plusieurs inéquation(s). De plus, les contraintes et les scindages sont cette fois gérés automatiquement par le dynamisme des programmes de T. Gómez-Díaz. Cela rend son implantation d'autant plus "agréable" dans ce cadre.

Remerciements. L'auteur tient à remercier l'équipe *Café* de l'INRIA Sophia-Antipolis pour leur accueil et plus particulièrement, E. Hubert pour ses conseils et l'intérêt qu'elle a manifesté tout le long de ce travail.

Références

[1] J. Abdeljaoued. - *Algorithmes rapides pour le calcul du polynôme caractéristique.* Thèse de doctorat de l'université de Franche-Comté (1997).

[2] S.A. Abramov, K.Yu. Kvashenko. -*On the greatest common divisor of polynomials which depend on a parameter.* Proceedings of the 1993 ISSAC. Manuel Bronstein, Editor.

[3] D. Cox, J. Little, D. O'Shea. - *Ideals, varieties and algorithms.* Springer-Verlag (1992).

[4] J. Della Dora, C. Dicrescenzo, D. Duval. - *About a new method for computing in algebraic number fields.* Eurocal'85, vol. 2, Springer Lecture Notes in Computer Science 204, ed. G. Goos, J. Hartmanis, p.289-290 (1985).

[5] C. Dicrescenzo, D. Duval. - *Algebraic extensions and algebraic closure in Scratchpad.* Symbolic and algebraic computation, Springer Lecture Notes in Computer Science 358, ed. P. Gianni, p.440-446 (1989).

[6] S. Dellière. - *Triangularisation des systèmes constructibles. Application à l'évaluation dynamique.* Thèse de doctorat de l'université de Limoges (1999).

[7] D. Duval. - *Diverses questions relatives au calcul formel avec des nombres algébriques.* Thèse d'Etat, Grenoble (1987).

[8] D. Duval. - *Autour de l'algorithme d'Euclide pour les polynômes.* Notes de cours (1996).

[9] T. Gómez-Díaz. - *Quelques applications de l'évaluation dynamique.* Thèse de doctorat de l'université de Limoges (1994).

[10] T. Gómez-Díaz. - *Examples of using dynamic constructible closure.* Proceedings of International Imacs Symposiumm on Symbolic Computation International Association for Mathematics and Computers in Simulation ed. G. Jacob, N.E. Oussous, S. Steiberg, p.17-21 (1993). Also in Mathematics and Computers in Simulation 42, p.375-383 (1996).

[11] T. Gómez-Díaz. - *Let T be a triangle, is it isosceles?* Actas del primer encuentro de àlgebra computacional y aplicaciones - EACA'95 (Santander) p.67-73 (1995).

[12] T. Gómez-Díaz. - *On the computation of Jordan forms with parameters.* (preprint 1997).

[13] R.D. Jenks, R.S. Sutor. - *Axiom, The Scientific Computation System.* NAG, Springer-Verlag (1992).

[14] D. Lazard. - *A new method for solving algebraic systems of positive dimension.* Discrete Applied Mathematics 33, p.147-160 (1991).

[15] B. Mishra. - *Algorithmic algebra.* New York : Springer-Verlag (1993).

Unité de recherche INRIA Sophia Antipolis
2004, route des Lucioles - B.P. 93 - 06902 Sophia Antipolis Cedex (France)

Unité de recherche INRIA Lorraine : Technopôle de Nancy-Brabois - Campus scientifique
615, rue du Jardin Botanique - B.P. 101 - 54602 Villers lès Nancy Cedex (France)
Unité de recherche INRIA Rennes : IRISA, Campus universitaire de Beaulieu - 35042 Rennes Cedex (France)
Unité de recherche INRIA Rhône-Alpes : 655, avenue de l'Europe - 38330 Montbonnot St Martin (France)
Unité de recherche INRIA Rocquencourt : Domaine de Voluceau - Rocquencourt - B.P. 105 - 78153 Le Chesnay Cedex (France)

Éditeur
INRIA - Domaine de Voluceau - Rocquencourt, B.P. 105 - 78153 Le Chesnay Cedex (France)
http://www.inria.fr
ISSN 0249-6399