Difficulté du résultant et des grands déterminants

Bruno Grenet

Difficulté du résultant et des grands déterminants

Bruno Grenet
LIP*, ENS Lyon, Université de Lyon

12 novembre 2009

Résumé

Le résultant est un polynôme permettant de déterminer si plusieurs polynômes donnés ont une racine commune. Canny a pu donner un algorithme PSPACE calculant le résultant à l'aide de calculs de déterminants, mais pose la question de sa complexité exacte.

On s'intéresse ici à donner une estimation plus fine de cette complexité. D'une part, on montre que le résultant est dans AM, et qu'il est NP-difficile sous réduction probabiliste. D'autre part, les matrices en jeu étant descriptibles par des circuits de taille raisonnable, on montre que le calcul du déterminant pour de telles matrices est PSPACE-complet.

*UMR 5668 ENS Lyon – cnrs – UCBL – INRIA. Rapport de recherche RRLIP2009-32

Introduction

Le résultant de deux polynômes P et Q en une variable est un déterminant formé à partir des cœfficients des polynômes qui s'annule si et seulement s'ils ont une racine commune. Pour des polynômes de degrés p et q, la matrice construite est de taille $p+q$ et son déterminant peut donc être évalué efficacement. Il existe des généralisations en plusieurs variables, mais la taille des matrices en jeu est exponentielle en la taille du problème. Les déterminants de matrices de taille n étant calculables en espace polylogarithmique en n, le résultant multivarié peut être évalué en espace polynomial. Cette borne supérieure est la meilleure connue à ce jour pour l'évaluation du résultant.

L'objet de ce rapport est d'étudier plus finement la complexité du résultant, et en particulier le problème consistant à décider de sa nullité. Il est montré que ce problème appartient à la classe AM. On s'intéresse également à des bornes inférieures pour ce problème. D'une part, on montre que le problème est NP-difficile sous réduction probabiliste. D'autre part, on s'intéresse à un problème plus large, qui est le calcul du déterminant d'une matrice donnée sous forme succincte. Les matrices intervenant dans le calcul du résultant sont effectivement succinctement représentées, et la PSPACE-complétude du calcul de tels déterminants indique que cette approche ne peut servir à améliorer les bornes connues.

Dans la première partie, on situe le résultant dans la hiérarchie polynomiale. On montre en particulier l'appartenance à AM et la NP-difficulté sous réduction probabiliste. Ceci répond quasiment à la question de Canny qui s'interroge sur la complexité exacte du résultant, et affirme apparemment sans preuve que celui-ci est NP-difficile. Dans la seconde partie, on s'intéresse aux matrices sous forme succincte. Le résultat principal est la PSPACE-complétude du calcul du déterminant dans un tel cas. Enfin, la dernière partie revient sur le résultant à proprement parler. On donne la définition formelle de cet objet, ainsi que l'algorithme de Canny qui le calcule en espace polynomial. Ces explications nous permettent de démontrer l'intérêt des matrices succinctement décrites, puisqu'on prouve que les matrices de Macaulay, apparaissant dans la définition du résultant, sont effectivement descriptibles par des circuits de petite taille.

1 Difficulté du résultant multivarié

On s'intéresse dans cette première partie à la position occupée par le calcul du résultant dans la hiérarchie polynomiale. On donne une borne supérieure (classe AM) et une borne inférieure (NP-difficulté sous réduction probabiliste) pour décider de la nullité du résultant multivarié. Cela équivaut à décider de l'existence d'une solution non triviale à un système de n équations polynomiales homogènes à n inconnues. Plus de détails seront donnés dans la partie 3.

1.1 Définitions et borne supérieure

Le problème qui nous intéresse est de décider de la nullité ou non du résultant lorsque les polynômes considérés sont des polynômes sur $\mathbb{C}$. Il s'agit en fait de décider si un système de n équations polynomiales homogènes à n inconnues admet une solution non triviale. C'est un cas particulier du problème *Hilbert's Nullstellensatz* ($\mathtt{HN}_{\mathbb{C}}$) dans lequel on considère un nombre quelconque de polynômes, non nécessairement homogènes. On peut aussi s'intéresser au problème dans lequel on considère un nombre quelconque de polynômes, cette fois-ci homogènes.

Le problème $\mathtt{HN}_{\mathbb{C}}$ tient son nom du théorème du même nom qui stipule, sous sa forme faible, que s polynômes $f_1, \ldots, f_s$ à n variables n'admettent aucune racine commune dans $\mathbb{C}$ si et seulement s'il existe $g_1, \ldots, g_s \in \mathbb{C}[X_1, \ldots, X_n]$ tels que $\sum_i f_i g_i = 1$.

On donne maintenant une définition formelle de ces problèmes.

Définition 1.1. Les entrées des problèmes sont les suivantes :
- $\mathtt{HN}_{\mathbb{C}}$: $f_1, \dots, f_s \in \mathbb{C}[X_1, \dots, X_n]$;
- $\mathtt{H_2N}_{\mathbb{C}}$: $f_1, \dots, f_s \in \mathbb{C}[X_1, \dots, X_n]$, homogènes ;
- $\mathtt{H_2N}_{\mathbb{C}}^{\square}$: $f_1, \dots, f_n \in \mathbb{C}[X_1, \dots, X_n]$, homogènes ;

Pour chacun, la question est de savoir s'il existe $(x_1, \dots, x_n) \in \mathbb{C}^n$ tels que pour tout i, $f_i(\bar{x}) = 0$[1]. Dans les deux cas homogènes, on cherche une solution *non triviale*, le n-uplet $(0, 0, \dots, 0)$ étant toujours solution.

On définit les équivalents booléens HN, $\mathtt{H_2N}$ et $\mathtt{H_2N}^{\square}$ en prenant des polynômes à cœfficients dans $\mathbb{Z}$, et en se posant toujours la question de l'existence de racines complexes.

Les problèmes $\mathtt{H_2N}_{\mathbb{C}}^{\square}$ et $\mathtt{H_2N}^{\square}$ correspondent exactement à la nullité du résultant des polynômes en question. En effet, on verra dans la partie 3 que le résultant s'annule par définition si et seulement si les polynômes admettent une racine commune non triviale. Dans la suite, on ne considérera que les versions booléennes des problèmes, afin de pouvoir travailler sur leur complexité *classique* (et non *algébrique*).

On commence par montrer la borne supérieure annoncée.

Proposition 1.2. *Sous l'hypothèse de Riemann généralisée,* $\mathtt{H_2N}^{\square}$ *est dans la classe* AM.

Avant de prouver cette propriété, il faut en expliquer les termes. L'*Hypothèse de Riemann Généralisée* concerne la répartition des zéros d'un type particulier de fonctions, appelées *fonctions* L *de Dirichlet*. On définit ensuite informellement la classe AM. Pour une définition plus formelle, on se reportera à [1, chapitre 8].

Définition 1.3. La classe AM peut être définie par des protocoles interactifs probabilistes. On a ici deux joueurs, Arthur et Merlin. Le premier a une puissance de calcul polynomiale (*i.e.* tous les calculs qu'il peut effectuer sont dans P), tandis que le second a une puissance de calcul illimitée (*i.e.* il peut calculer toute fonction calculable). Pour un problème A, et une entrée x appartenant ou non à A, Merlin essaie de convaincre Arthur que $x \in A$, en lui fournissant une « preuve ». De son côté, Arthur essaie de vérifier la preuve. Il tire au hasard un mot z_1 et l'envoie à Merlin. Merlin calcule alors une preuve z_2 et la renvoie à Arthur. Ce dernier effectue un calcul sur x, z_1 et z_2 (en temps polynomial) pour décider s'il accepte ou non la preuve.

Le langage A est alors dans AM si avec ce protocole, quand $x \in A$, Merlin convainc toujours Arthur que c'est le cas, et quand $x \notin A$, Merlin convainc Arthur que $x \in A$ avec une probabilité au plus $1/4$.

Preuve de la proposition 1.2. Ceci provient de [29], qui démontre que c'est le cas du problème HN, toujours sous l'hypothèse de Riemann généralisée.

Partant d'un système $\mathcal{S}$ de n polynômes homogènes à n inconnues $x_1, \dots, x_n$, on peut construire une instance $\mathcal{T}$ de HN en ajoutant n nouvelles inconnues $y_1, \dots, y_n$ et l'équation (non homogène) $\sum_{i=1}^{n} x_i y_i = 1$. Le nouveau système admet une solution si et seulement si celui de départ en admettait une non triviale. Étant donnée une solution de $\mathcal{T}$, l'équation rajoutée assure que les x_i sont non tous nuls. Les équations de l'instance de départ étant dans le nouveau système, les x_i forment bien une solution non triviale de $\mathcal{S}$. Dans l'autre sens, partant d'une solution non triviale $x_1, \dots, x_n$ de $\mathcal{S}$ dont k inconnues ont une valeur non nulle, on fixe pour chaque inconnue x_i non nulle la valeur de y_i à $1/(kx_i)$. C'est bien une solution de $\mathcal{T}$.

Ceci nous montre donc que notre problème est moins dur que HN, et qu'il appartient à AM. □

[1] La notation $\bar{x}$ représente le uplet $(x_1, \dots, x_n)$.

1.2 Résultats intermédiaires

On va maintenant s'intéresser à la borne inférieure. Pour cela, on commence par énoncer trois résultats, sans liens, qui vont être utilisés dans la réduction. Tout d'abord, on montre qu'une variété définie par des polynômes homogènes peut être définie par un nombre restreint de tels polynômes. Ensuite, on s'intéresse à l'élimination des quantificateurs sur $\mathbb{C}$. Enfin, on exprime le lemme de Schwartz-Zippel [1] sur les racines d'un polynôme.

Lemme 1.4. *Soit K un corps algébriquement clos et de degré de transcendance infini , et V une variété projective définie par les polynômes homogènes $f_1, \ldots, f_s \in K[X_0, \ldots, X_n]$, de degrés identiques d. Alors V peut être définie par $n+1$ polynômes homogènes de degré d sur $K[X_0, \ldots, X_n]$.*

Preuve. On adapte ici la preuve de [30, Proposition 1], qui affirme qu'une variété algébrique définie par s polynômes (non homogènes) de $K[x_1, \ldots, x_n]$ peut-être définie par $n+1$ tels polynômes. La preuve construit ces polynômes comme des combinaisons linéaires des polynômes de départ.

Les coefficients des f_j $(1 \leqslant j \leqslant s)$ sont en nombre fini. Ils vivent donc dans un sous-corps k de K. On définit la matrice $(\alpha_{ij})_{1 \leqslant i \leqslant n+1, 1 \leqslant j \leqslant s} \in K^{s(n+1)}$ telle que les α_{ij} soient algébriquement indépendants sur k.

On définit alors les polynômes $g_1, \ldots, g_{n+1}$ par

$$g_i = \sum_{j=1}^{s} \alpha_{ij} f_j, 1 \leqslant i \leqslant n+1.$$

Remarquons que les f_j étant tous homogènes de même degré d, alors les g_i le sont également. Nous affirmons maintenant que les g_i définissent la variété V.

Tout d'abord, si tous les f_j s'annulent en $(x_0, x_1, \ldots, x_n)$, alors il est clair que les g_i s'annulent également en ce point. Réciproquement, supposons que tous les g_i s'annulent en $(x_0, \ldots, x_n)$. On distingue alors deux cas, dépendant de la nullité de x_0.

Si $x_0 = 0$, alors on définit les $\tilde{g}_i$ et les $\tilde{f}_j$ par

$$\tilde{g}_i(X_1, \ldots, X_n) = g_i(0, X_1, \ldots, X_n) \in K[X_1, \ldots, X_n],$$

$$\tilde{f}_j(X_1, \ldots, X_n) = f_j(0, X_1, \ldots, X_n) \in K[X_1, \ldots, X_n].$$

On remarque alors que pour $1 \leqslant i \leqslant n+1$, $\tilde{g}_i = \sum_{j=1}^{s} \alpha_{ij} \tilde{f}_j$. Les α_{ij} sont encore algébriquement indépendants sur $k[0] = k$, et on peut donc appliquer le résultat de [30] pour déduire que

$$(\forall i, \tilde{g}_i(x_1, \ldots, x_n) = 0) \implies (\forall j, \tilde{f}_j(x_1, \ldots, x_n) = 0).$$

Or dire que les $\tilde{f}_j$ s'annulent sur $(x_1, \ldots, x_n)$ revient exactement à dire que les f_j s'annulent sur $(0, x_1, \ldots, x_n)$.

Supposons maintenant que $x_0 \neq 0$. Comme les g_i sont des polynômes homogènes, s'ils s'annulent sur $(x_0, \ldots, x_n)$, ils s'annulent également sur $(1, x_1/x_0, \ldots, x_n/x_0)$. De même que précédemment on définit les $\tilde{f}_j$ et les $\tilde{g}_i$, en spécialisant la variable X_0 non plus à 0 mais à 1. Les α_{ij} sont encore algébriquement indépendants sur $k[1] = k$. On utilise à nouveau [30] pour déduire de la nullité des $\tilde{g}_i$ sur $(x_1/x_0, \ldots, x_n/x_0)$ celle des $\tilde{f}_j$. Donc pour $1 \leqslant j \leqslant s$, $\tilde{f}_j(x_1/x_0, \ldots, x_n/x_0) = 0$. Ce qui équivaut à $f_j(1, x_1/x_0, \ldots, x_n/x_0) = 0$, et par homogénéité, on obtient le résultat souhaité. □

On va se servir dans la suite de l'élimination des quantificateurs sur $\mathbb{C}$. Pour utiliser cette élimination de manière effective, il nous faut des bornes sur les formules produites. C'est l'objet du lemme suivant, qui est un cas particulier de [23, Theorem 2].

Lemme 1.5. *Soit φ une formule du premier ordre sur le langage $(\mathbb{C}, 0, 1, +, -, \times, =)$ de la forme $\forall \bar{x} \psi(\bar{x})$ où $\bar{x}$ est un n-uplet de variables et ψ une formule sans quantificateurs dans laquelle s polynômes apparaissent. Supposons de plus que le degré des polynômes et leurs cœfficients sont bornés par des constantes.*

Alors il existe un polynôme $p(n, \log s)$, indépendant de φ, tel que φ est équivalente à une formule sans quantificateurs sous forme normale disjonctive dans laquelle interviennent au plus $2^{p(n,\log s)}$ polynômes, dont les degrés sont bornés par $2^{p(n,\log s)}$.

Enfin, le troisième lemme a trait à la probabilité pour un n-uplet aléatoire d'être racine d'un polynôme donné.

Lemme 1.6 (Schwartz-Zippel [1])**.** *Soit p un polynôme à n variables à cœfficients entiers. On suppose que p n'est pas identiquement nul, et que son degré est au plus d. Alors pour un n-uplet $(a_1, \dots, a_n)$ choisi aléatoirement suivant la distribution uniforme dans $\{1, \dots, q\}^n$, la probabilité que $(a_1, \dots, a_n)$ soit racine de p est*

$$\Pr[p(a_1, \dots, a_n) = 0] \leqslant \frac{d}{q}.$$

Preuve. La preuve se fait par induction sur n. Le cas $n = 1$ est trivial, et repose simplement sur le fait qu'un polynôme de degré d a au plus d racines. Supposons alors le résultat pour un nombre $n-1$ de variables. En écrivant p sous la forme $\sum_{i=0}^{d} X_1^i p_i(X_2, \dots, X_n)$, chaque p_i a un degré au plus $d - i \leqslant d$. De plus, p n'étant pas identiquement nul, c'est également le cas de l'un des p_i au moins. Notons $k = \max\{i : p_i \neq 0\}$. Alors p_k est un polynôme à $n-1$ variables, de degré au plus d, et non identiquement nul.

Par hypothèse d'induction, $\Pr[p_k(a_2, \dots, a_n) = 0] \leqslant (d-k)/q$. Maintenant, pour $(a_2, \dots, a_n)$ fixé tel que $p_k(a_2, \dots, a_n) \neq 0$, le polynôme à une variable $p(x_1, a_2, \dots, a_n)$ n'est pas identiquement nul. De plus, il est de degré k. La probabilité de l'annuler est donc au plus k/q. Finalement, pour annuler p en $(a_1, \dots, a_n)$, il est nécessaire d'annuler p_k en $(a_2, \dots, a_n)$ ou $p(x_1, a_2, \dots, a_n)$ en a_1. Donc

$$\Pr[p(a_1, \dots, a_n) = 0] \leqslant \frac{d-k}{q} + \frac{k}{q} = \frac{d}{q}.$$

Ce qui achève la preuve. □

1.3 Réduction

On commence maintenant la réduction proprement dite. À cette fin, on va utiliser plusieurs problèmes intermédiaires. Commençons par énoncer le résultat principal.

Théorème 1.7. *$H_2M^\square$ est* NP*-difficile sous réduction probabiliste.*

Pour prouver le théorème, on effectue une réduction depuis le problème `Boolsys`, de la même façon que dans la preuve de la NP-difficulté de $\mathtt{H_2N}$ [31]. Il faut modifier un peu la réduction, afin de pouvoir utiliser le lemme 1.4. De plus, le passage au cas homogène nous conduit à effectuer une réduction probabiliste, rendue possible par les lemmes 1.5 et 1.6.

Une instance de `Boolsys` est la donnée d'un système d'équations en les variables booléennes $X_1, \dots, X_n$. Les équations peuvent prendre trois formes, $X_i = 1$, $X_i = \neg X_j$ ou $X_i = X_j \vee X_k$.

Lemme 1.8. *`Boolsys` est* NP*-complet.*

Preuve. On réduit facilement `3-SAT` à `Boolsys` en remplaçant chaque clause $l_1 \vee l_2 \vee l_3$ par au plus six équations. Pour $l_i = \neg x$ (où x est une variable), on crée l'équation $x' = \neg x$ avec x' une variable

fraîche. On peut alors réécrire $l_1 \vee l_2 \vee l_3$ avec des variables non niées (en prenant éventuellement x'). Cette clause est alors équivalente à l'ensemble d'équations $X = l_1 \vee l_2$, $Y = X \vee l_3$ et $Y = 1$, avec X et Y deux variables fraîches. Notre clause de départ est donc équivalente à ces trois dernières équations auxquelles on ajoute éventuellement au plus trois équations du type $x' = \neg x$ pour se ramener à une clause sans variables niées.

L'appartenance du problème à NP est claire. □

Lemme 1.9. *H₂N, restreint à un système de polynômes homogènes de degré 2, est* NP*-difficile.*

Preuve. Partant d'une instance de `Boolsys` à n variables $X_1, \ldots, X_n$, on crée une instance de $\mathtt{H_2N}$ à $n+1$ variables $x_0, x_1, \ldots, x_n$, dont tous les polynômes ont degré 2. Les x_i correspondent aux X_i pour $i \geqslant 1$, et x_0 est une nouvelle variable. Les équations sont séparées en deux groupes. Premièrement, on définit les n équations $x_i^2 = x_0^2$ $(1 \leqslant i \leqslant n)$. Ensuite, pour chaque équation de la forme $X_i = 1$, on crée l'équation $(x_i + x_0)^2 = 0$. Pour $X_i = \neg X_j$, on crée $(x_i + x_j)^2 = 0$. Enfin, pour $X_i = X_j \vee X_k$, on crée $4x_i x_0 = (x_j + x_k)^2 + 2x_0(x_j + x_k) - 4x_0^2$.

Supposons que l'instance de `Boolsys` soit satisfaite par $(X_1, \ldots, X_n)$. Alors pour tout $x_0 \neq 0$, en posant $x_i = -x_0$ si X_i est vrai et $x_i = x_0$ dans le cas inverse, on peut aisément vérifier que le $(n+1)$-uplet $(x_0, x_1, \ldots, x_n)$ est solution de notre instance de $\mathtt{H_2N}$ construite. Réciproquement, si $(x_0, \ldots, x_n)$ est solution, on remarque que x_0 ne peut être nul, et que chaque x_i vaut $\pm x_0$. De la même façon, on construit un assignement satisfaisant des variables en posant X_i vrai si et seulement si $x_i = -x_0$. □

La réduction au problème $\mathtt{H_2N}^{\square}$ est effectuée à partir de ce dernier problème dont on a prouvé la NP-difficulté. C'est cette ultime réduction qui possède un caractère probabiliste.

Preuve du théorème 1.7. On a donc construit une instance de $\mathtt{H_2N}$ avec s polynômes de degré 2, équivalente à notre instance de `3-SAT` de départ. On va maintenant construire une instance de $\mathtt{H_2N}^{\square}$ à partir de celle de $\mathtt{H_2N}$. Notre ensemble de polynômes satisfait les conditions du lemme 1.4. On peut donc les remplacer par les g_i définis par $g_i = \sum_{j=1}^{s} \alpha_{ij} f_j$. La difficulté consiste à trouver les α_{ij} qui conviennent. La preuve du lemme 1.4 utilise des éléments algébriquement indépendants. On ne peut pas le faire ici car on décrit les polynômes par la liste de leurs cœfficients entiers, et qu'on ne peut utiliser d'éléments transcendants. Pour utiliser des transcendants, il faudrait changer la description utilisée, et donc le problème. On pourrait utiliser des grands entiers, en suivant [32], mais ça ne fonctionne toujours pas ici car les entiers utilisés sont trop grands pour garder le caractère polynomial de la réduction. On va donc avoir recours à des α_{ij} aléatoires, ce qui impliquera le caractère probabiliste de la réduction.

Étant donnée la définition des g_i, ils s'annulent tous dès que tous les f_j s'annulent. On cherche maintenant à assurer la réciproque, qui s'écrit formellement sous la forme

$$\Phi(\bar{\alpha}) \equiv \forall x_0 \cdots \forall x_n \left(\bigwedge_{j=1}^{s} f_j(\bar{x}) = 0 \right) \vee \left(\bigvee_{i=0}^{n} \sum_{j=1}^{s} \alpha_{ij} f_j(\bar{x}) \neq 0 \right).$$

L'idée va consister à éliminer les quantificateurs dans la formule $\Phi(\bar{\alpha})$[2]. On obtient alors une formule équivalente $\Psi(\bar{\alpha})$ de la forme

$$\Psi(\bar{\alpha}) \equiv \bigvee_{k} \left(\bigwedge_{l} P_{kl}(\bar{\alpha}) = 0 \wedge \bigwedge_{m} Q_{km}(\bar{\alpha}) \neq 0 \right),$$

[2]La notation $\bar{\alpha}$ représente le uplet des α_{ij}.

où les P_{kl} et les Q_{kl} sont des polynômes en les α_{ij}.

On sait que la formule Φ (et donc Ψ) est satisfaite par tout uplet de α_{ij} algébriquement indépendants sur $\mathbb{Q}$. L'ensemble des points qui satisfont Ψ est donc dense dans $\mathbb{C}^{s(n+1)}$. Ainsi, aucun sous-espace de dimension strictement inférieure à $s(n+1)$, ni union de tels sous-espaces, ne peuvent contenir tous les $s(n+1)$-uplets satisfaisant Ψ. Ce qui prouve qu'il existe une des clauses de Ψ dans laquelle n'apparaît aucune équation de la forme $P_{kl}(\bar{\alpha}) = 0$. Pour satisfaire Ψ, il suffit donc de trouver des α_{ij} satisfaisant une formule $\bigwedge_k Q_k(\bar{\alpha}) \neq 0$, ou encore de trouver des α_{ij} n'étant pas racine d'un polynôme donné (de la forme $\prod_k Q_k(\bar{\alpha})$).

On fait maintenant appel au lemme 1.5. Partant de notre formule Φ, on peut construire une formule Ψ équivalente, sous forme normale et sans quantificateurs, dans laquelle il y a au plus $2^{p(n,\log(s+n))}$ polynômes de degrés au plus $2^{p(n,\log(s+n))}$. Le polynôme dont on cherche à éviter les racines a donc un degré au plus $2^{2p(n,\log(s+n))}$. Par le lemme 1.6, il suffit de prendre les α_{ij} aléatoirement entre 0 et $2^{2p(n,\log(s+n))+2}$ pour assurer qu'avec probabilité au moins 3/4, $\Psi(\bar{\alpha})$ et donc $\Phi(\bar{\alpha})$ sont vérifiées. □

2 Déterminant de matrices succinctement représentées

On étudie ici le calcul du déterminant de matrices données par circuit. Ce formalisme est une adaptation aux matrices de celui des *représentations succinctes*. Il a été d'abord développé pour les graphes [24] sous la forme des *Small Circuit Representations* (SCR). Il a été montré que pour certains problèmes classiques sur les graphes, la donnée sous forme de SCR augmentait la complexité exponentiellement. Une classe de tels problèmes a été caractérisée dans [41]. Des extensions du formalisme précédent ont été proposées [37, 2] dans lesquelles on a accès au voisinage complet des sommets. De même, une caractérisation de problèmes dont la complexité croit exponentiellement y est donnée, basée sur les réductions de [14]. Une description des graphes sous forme d'OBDD (*Ordered Binary Decision Diagram*) mène à des croissances similaires de la complexité [22]. D'autres approches sont développées dans [46, 3], ce qui permet de traiter différentes sortes de problèmes combinatoires. Enfin, [43] s'intéresse aux versions succinctes de certains problèmes de comptages, tandis que [10] s'intéresse à des descriptions succinctes dans le modèle BSS de machine sur les réels [7, 6]. D'autre part, la construction du graphe des transitions d'une machine de Turing proposée dans [14, 10] pourrait permettre de donner une autre preuve de la PSPACE-difficulté de l'accessibilité dans un graphe (lemme 2.7).

On définit d'abord ce que l'on entend par une représentation par circuit, puis on explique pourquoi le déterminant d'une matrice donnée sous cette forme est PSPACE-complet.

2.1 Définitions

Définition 2.1. Une *représentation par circuit* d'un graphe $G = (V, E)$ est un circuit C_G tel que

1. C_G a deux entrées de $\lceil \log |V| \rceil$ bits chacune ;
2. la sortie de C_G est définie par $C_G(i, j) = 1$ si $(v_i, v_j) \in E$ et $C_G(v_i, v_j) = 0$ si $(v_i, v_j) \notin E$ (le comportement n'est pas défini si l'un des deux numéros i ou j ne correspond à aucun sommet).

On définit la *taille* d'une représentation comme étant le nombre de portes du circuit.

On définit maintenant l'analogue de cette définition adapté au cas des matrices.

Définition 2.2. Une *représentation par circuit* d'une matrice M de taille $n \times m$ est un circuit C_M tel que

1. C_M a deux entrées de $\lceil \log n \rceil$ et $\lceil \log m \rceil$ bits respectivement ;

2. la sortie de C_M, constituée d'un ou plusieurs bits, est définie par $C_M(i,j) = M_{ij}$ (le comportement n'est pas défini si $i > n$ ou $j > m$).

Dans la suite, on s'intéresse à la complexité de problèmes sur des graphes ou des matrices représentés par circuits. La complexité peut augmenter avec cette représentation. En effet, un circuit de taille n peut définir un graphe ou une matrice de taille exponentielle en n. C'est par exemple le cas des matrices de Macaulay apparaissant dans la définition du résultant (*cf* partie 3). Sur ces entrées, construire explicitement le graphe ou la matrice pour ensuite appliquer les algorithmes habituels fait exploser la complexité. La question que l'on se pose ici est de savoir s'il y a une meilleure méthode, en tenant compte du fait que les graphes et les matrices ainsi décrits ne sont pas totalement quelconques.

2.2 Résultat principal

Le théorème suivant représente une augmentation exponentielle de la complexité spatiale par rapport au cas classique.

Théorème 2.3. *Décider si le déterminant d'une matrice donnée par circuit est nul est* PSPACE*-complet.*

Preuve (appartenance à PSPACE*).* L'appartenance du problème à PSPACE peut être déduite de [8, 5, 39, 19] qui étudient le calcul du rang et du déterminant d'une matrice en parallèle. Leurs bornes sur la profondeur d'un circuit ou le temps parallèle nécessaire (polylogarithmique) suffisent pour conclure que le calcul de ce déterminant est dans PSPACE, ce qui est affirmé dans [11].

En effet, on peut simuler un circuit de profondeur p en espace linéaire en p sur une machine de Turing. Notons $s(p)$ l'espace nécessaire à simuler un circuit de profondeur p, *i.e.* l'espace nécessaire à déterminer la valeur prise par le nœud S de sortie. Notons S_1 et S_2 les deux prédécesseurs de S, dont la valeur détermine la valeur de S. Leur valeur est déterminée par un circuit de profondeur au plus $p-1$. Pour déterminer la valeur de S, on procède alors comme suit. On détermine la valeur de S_1 (en espace $s_1 \leqslant s(p-1)$), on retient cette valeur (ce qui occupe une case mémoire), et on détermine la valeur de S_2 (en espace $s_2 \leqslant s(p-1)$). L'espace total est donc $s(p) = \max(s_1, 1+s_2) \leqslant 1+s(p-1)$. Pour un circuit de profondeur 1, l'espace nécessaire est 1, ce qui donne un espace nécessaire inférieur ou égal à p.

Ici, l'entrée de la matrice est de taille logarithmique en sa dimension, donc la profondeur polylogarithmique des circuits correspond à un espace polynomial pour les simuler, d'où l'appartenance à PSPACE. □

On montre ensuite que ce problème est PSPACE-difficile. Pour cela, on réduit tout problème de PSPACE à une formule booléenne quantifiée. En se servant de la forme spéciale des formules trouvées, on les transforme alors en graphe, puis suivant la construction de [44], on transforme ce graphe en matrice. La non nullité du déterminant correspondant alors à l'existence d'un cycle non trivial dans le graphe, lui-même correspondant à la vérité de la formule. On détaille ci-après les différentes étapes.

Définition 2.4. On note $\exists! x\, \varphi(x)$ la formule $\exists x\, (\varphi(x) \wedge \forall y\, (\varphi(y) \implies x = y))$.

On appelle *formule booléenne quantifiée* une formule booléenne sous forme normale conjonctive dont chaque variable est quantifiée soit existentiellement, soit universellement. On impose également que la formule soit prénexe.

On appelle `UQBF` (pour *Unique*-`QBF`) l'ensemble des formules booléennes quantifiées vraies qui sont équivalentes à la formule obtenue en remplaçant les quantifications existentielles $\exists x \varphi(x)$ par la formule $\exists! x \varphi(x)$.

Lemme 2.5. *UQBF est* PSPACE*-complet.*

Preuve. On remarque d'abord que UQBF est dans PSPACE pour la même raison que QBF. On construit facilement un circuit qui prend en entrée une instance de QBF et dont la sortie est 1 si et seulement si cette instance est positive (la formule est vraie) [40]. Ce circuit peut être légèrement modifié pour reconnaître les instances de UQBF à la place de QBF. Pour une formule $\psi = \forall x_1 \exists x_2 \forall x_3 \dots Q_n x_n\ \varphi(\bar{x})$, on définit un arbre binaire complet de profondeur n. Le sous-arbre gauche contient les assignements de $(x_1, \dots, x_n)$ avec $x_1 = 1$ tandis que le sous-arbre droit contient ceux avec $x_1 = 0$. On effectue le même branchement pour x_2 à profondeur 1, et de la même façon pour toutes les variables. On transforme alors ce graphe en circuit. Les branchements correspondant à des quantifications universelles sont remplacés par des portes ET. Ceux correspondant à des quantifications existentielles sont remplacés par un sous-circuit calculant un XOR ($x \oplus y \equiv (x \wedge \neg y) \vee (\neg x \wedge y)$). Cette modification n'augmente que linéairement la profondeur du circuit, et on peut montrer avec les mêmes arguments que dans la preuve du théorème 2.3 que son évaluation est en espace polynomial.

Il reste maintenant à montrer la PSPACE-difficulté. On utilise pour cela la preuve classique de la PSPACE-complétude de QBF, en modifiant la réduction afin de construire des formules booléennes quantifiées appartenant à UQBF.

Soit donc L un langage de PSPACE, et $x \in L$. Alors on sait qu'il existe une machine de Turing déterministe M et un chemin entre la configuration initiale q_0x de M et sa configuration acceptante c_{accept} qui utilise un espace au plus $s(|x|)$, où s est un polynôme. La longueur de ce chemin est donc au plus $2^{as(n)}$, où $n = |x|$ et a est une constante ne dépendant que de M. On va construire une formule booléenne quantifiée qui exprime ce fait.

On construit φ_t telle que $\varphi_t(c, c')$ signifie que l'on passe de la configuration c à c' en *au plus* t étapes de calcul si c' est la configuration acceptante, et en *exactement* t étapes de calcul si c' n'est pas acceptante.

On définit

$$\varphi_1(c, c') \equiv (c = c' = c_{\text{accept}}) \vee (\text{« } c' \text{ est obtenue en un pas de calcul depuis } c \text{ »}).$$

Puis, par induction $\varphi_{2^{t+1}}(c, c') \equiv \exists c''(\varphi_{2^t}(c, c'') \wedge \varphi_{2^t}(c'', c'))$. De même que dans la preuve de PSPACE-complétude de QBF, il faut modifier un peu cette formule pour ne pas faire exploser sa taille. On a alors la formule équivalente

$$\varphi_{2^{t+1}}(c, c') \equiv \exists c'' \forall d, d' \left((d = c \wedge d' = c'') \vee (d = c'' \wedge d' = c')\right) \to \varphi_{2^t}(d, d').$$

Enfin, la formule recherchée est $\varphi_{2^{as(n)}}(q_0x, c_{\text{accept}})$.

L'existence d'un assignement satisfaisant la formule est assurée par le fait qu'un langage reconnu en espace $s(n)$ est reconnu en temps au plus $2^{as(n)}$ pour une constante a ne dépendant que de la machine utilisée. Le déterminisme de M, ajouté au fait que $\varphi_1(c, c)$ n'est vraie que pour $c = c_{\text{accept}}$, assure l'appartenance de la formule à UQBF. Enfin, la formule construite est bien de taille polynomiale en $|x|$, et sa construction se fait en temps polynomial. □

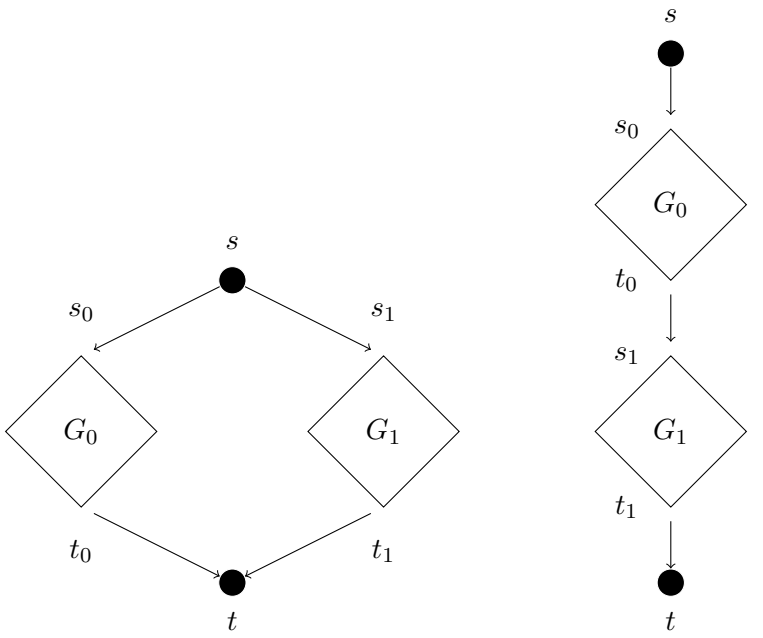
Remarque 2.6. *Nous avons découvert après avoir effectué cette preuve que le problème UQBF avait été défini d'une manière légèrement différente, mais équivalente, dans [18]. Ce papier, introuvable mais dont on a pu lire une partie du contenu sous forme d'un rapport de stage, contient apparemment une preuve, relativement similaire à la nôtre, de la* PSPACE*-difficulté du problème.*

Pour la suite de la preuve, il nous faut introduire le problème d'*accessibilité dans un graphe*. Étant donné un graphe G et deux sommets s et t de ce graphe, la question est de savoir s'il existe un chemin de s à t dans le graphe.

Lemme 2.7. *L'accessibilité dans un graphe donné par circuit est* PSPACE*-difficile.*

Preuve. Soit ψ une formule de QBF : $\psi = \forall x_1 \exists x_2 \forall x_3 \dots Q_n x_n \ \varphi(\bar{x})$ où φ est sans quantificateur, et $Q_n \in \{\forall, \exists\}$. On va transformer cette formule en un graphe G_ψ. Il faut alors s'assurer de plusieurs choses. L'accessibilité de s à t dans G_ψ doit être un problème équivalent à la vérité de la formule de départ, et le graphe doit bien admettre une représentation par circuit au sens de la définition 2.1, de taille polynomiale en $|\psi|$.

On définit le graphe récursivement. Pour une formule sans quantificateur vraie, on construit le graphe consistant en deux sommets s et t (la *source* et le *puits*) reliés par un arc $s \to t$. De même, pour une formule sans quantificateur fausse, le graphe consiste en une source et un puits, cette fois-ci non reliés. On veut maintenant construire le graphe G représentant une formule $Qx\,\varphi(x)$, où $Q \in \{\forall, \exists\}$. Soit G_0 et G_1 les graphes représentant $\varphi(0)$ et $\varphi(1)$ respectivement. On note s_0, s_1, t_0, t_1 les sources et les puits respectifs de G_0 et G_1. Alors le graphe G possède une source s et un puits t, ainsi qu'une copie de G_0 et G_1. Les arcs entre s, s_0, s_1, t, t_0 et t_1 dépendent de Q. Si $Q = \exists$, on crée les arcs $s \to s_0$, $s \to s_1$, $t_0 \to t$ et $t_1 \to t$. Si $Q = \forall$, les arcs sont $s \to s_0$, $t_0 \to s_1$, et $t_1 \to t$. Cette construction est illustrée par la figure 1.



(a) Quantificateur existentiel

(b) Quantificateur universel

FIG. 1: Constructions pour les quantificateurs existentiel (à gauche) et universel (à droite).

Pour achever la construction du graphe G_ψ, il suffit de répéter récursivement l'opération précédente pour toutes les variables de la formule ψ. Il est clair que l'existence d'un chemin de s à t dans le graphe est équivalente à la vérité de la formule $\psi\cdot$ En effet, la construction effectuée pour une quantification existentielle montre qu'il existe un chemin si et seulement si la formule φ est vraie pour $x = 0$ ou $x = 1$. Dans le cas d'une quantification universelle, il existe un chemin de s à t s'il en existe un qui passe par G_0 et un qui passe par G_1, ce qui revient à dire qu'il existe un chemin de s à t si $\varphi(0)$ est vraie et si $\varphi(1)$ l'est.

Pour s'assurer que ce graphe admet une représentation par circuit, il faut indiquer une numérotation des sommets qui permette de retrouver facilement les liens existant dans le graphe construit. La numérotation des sommets est construite récursivement, avec les mêmes étapes que pour la construction du graphe. On suppose que dans le graphe initial à deux sommets, s prend le numéro 10 et t le numéro 11. Les numéros des sommets vont évoluer à chaque construction effectuée pour un quantificateur. Les numéros des sommets dans la copie de G_0 sont calculés de la façon suivante : si un sommet a le numéro v dans G_0, alors on lui attribue le numéro $v0$[3] dans la copie. Le sommet s_0 a donc le numéro 100 et t_0 le numéro 110. Pour G_1, on fait la même chose en concaténant 1 à la place de 0. La nouvelle source et le nouveau puits créés prennent respectivement les numéros 10 et 11. En répétant ça pour chaque quantificateur, on obtient la numérotation finale du graphe, la source portant le numéro 10 et le puits le 11. On se convainc facilement que la numérotation utilise tous les entiers entre 2 et $2^{n+2} - 1$, que l'on peut ramener entre 1 et $2^{n+2} - 2$ si nécessaire.

Maintenant, pour décider de l'existence ou non d'un arc entre deux sommets u et v du graphe, il suffit de lire leurs numéros. Les deux premiers bits nous indiquent le type de sommets (s ou t) auxquels on s'intéresse. On note τ_u et τ_v les types respectifs de u et v, et on effectue une distinction de cas. On note b_u et b_v les chaînes booléennes respectives constituées par le numéro de u et v desquels on a supprimé les deux premiers bits. Les cas suivants sont ceux pour lesquels il existe un arc $u \to v$, et ce sont les seuls. On renvoie entre parenthèses à l'arc correspondant dans la figure 1.

1. $\tau_u = \tau_v = s$ et $b_v = b_u 0$ $(s \to s_0)$;
2. $\tau_u = \tau_v = s$ et $b_v = b_u 1$ et la taille de b_u indique qu'on est dans le cas d'une quantification existentielle $(s \to s_1)$;
3. $\tau_u = s$, $\tau_v = t$, $b_u = b_v$ et φ est évaluée à vraie avec les valeurs mises aux valeurs indiquées par b_u ;
4. $\tau_u = t$, $\tau_v = s$, $b_u = b0$ et $b_v = b1$ pour un certain b et la longueur de b indique qu'on est dans le cas d'une quantification universelle $(t_0 \to s_1)$;
5. $\tau_u = \tau_v = t$ et $b_u = b_v 0$ et la taille de b_u indique qu'on est dans le cas d'une quantification existentielle $(t_0 \to t)$;
6. $\tau_u = \tau_v = t$ et $b_u = b_v 1$ $(t_1 \to t)$.

Pour achever la preuve, il suffit de remarquer que l'algorithme décrit est polynomial et peut donc être implémenté par un circuit de taille polynomiale en la longueur de la formule ψ. □

Corollaire 2.8. *L'accessibilité* unique *dans un graphe donné par circuit est* PSPACE*-difficile.*

Preuve. Le problème consiste à décider de l'existence d'un chemin *unique* entre s et t. On utilise la réduction précédente, mais en partant d'une instance d'`UQBF`, et non plus de `QBF`. Si on a deux chemins distincts, ils se séparent nécessairement au niveau d'une quantification existentielle. On a donc une formule de la forme $\exists x\, \varphi(x)$ telle que $\varphi(0)$ et $\varphi(1)$ sont tous deux vrais. Ce qui contredit l'appartenance de la formule de départ à `UQBF`. □

Le dernier ingrédient nécessaire à la preuve du théorème est de savoir réduire l'accessibilité dans le graphe au calcul d'un déterminant. On introduit à cette fin la notion de couverture par cycles.

Définition 2.9. Soit $G = (V, E)$ un graphe (orienté ou non). Une *couverture par cycles* de G est une partition de V en sous-ensembles tels que chaque sous-ensemble forme un cycle dans G.

Le lemme suivant qui relie les couvertures par cycles et le déterminant de la matrice d'adjacence est une adaptation de [44].

[3] On concatène un 0 à la fin du numéro v.

Lemme 2.10. *Soit G un graphe orienté dont tous les cycles sont de longueur impaire. Alors le nombre de couvertures par cycles de G est égal au déterminant de sa matrice d'adjacence.*

Preuve. Soit M la matrice d'adjacence de G. Le déterminant de $(M_{ij})_{1\leqslant i,j\leqslant N}$ est défini par

$$\det(M) = \sum_{\sigma} (-1)^{\mathrm{sgn}(\sigma)} \prod_{i=1}^{N} M_{i,\sigma(i)},$$

où la somme est effectuée sur les $N!$ permutations de $\{1,\ldots,N\}$. Chaque permutation peut se décomposer en cycles[4]. Les seules permutations qui contribuent à la somme sont celles pour lesquelles le produit $\prod_{i=1}^{N} M_{i,\sigma(i)}$ est non nul, et qui correspondent donc aux cycles apparaissant dans le graphe. Ainsi, chaque permutation de contribution non nulle correspond exactement à une couverture par cycles. On a supposé qu'une telle couverture n'était constituée que de cycles de longueur impaire. Or la signature d'une permutation produit de cycles de longueur impaire est toujours 0, si bien que le terme $(-1)^{\mathrm{sgn}(\sigma)}$ peut être omis ici. Ceci montre que le nombre de couvertures par cycles du graphe est égale au déterminant de sa matrice d'adjacence. □

On peut maintenant prouver le théorème énoncé initialement, à savoir que le calcul du déterminant d'une matrice donné par circuit est PSPACE-difficile.

Preuve du théorème 2.3. On va montrer que l'accessibilité dans le graphe construit précédemment se réduit au calcul d'un déterminant. Pour cela, on utilise le lemme 2.10. On modifie notre graphe en identifiant la source et le puits. On ajoute aussi une boucle sur chaque sommet, excepté la source-puits $s = t$. Il est clair que ces deux modifications ne nécessitent pas de beaucoup modifier le circuit calculant, étant donnés deux numéros de sommet u et v, la présence ou non d'un arc de u vers v. On s'intéresse alors à la couverture par cycles de notre graphe. Si s est relié à t dans le graphe original, il existe au minimum une couverture par cycles : elle comprend le chemin de s à t et les boucles sur les sommets non utilisés. Sinon, il n'y a pas de couverture par cycles. En effet, le sommet source-puits n'ayant pas de boucle, il ne peut être couvert que par un cycle non trivial. Or un cycle non trivial passant par la source-puits est un chemin de s à t.

Pour appliquer le lemme 2.10, il faut vérifier que tous les cycles de notre graphe sont de longueur impaire. Les constructions pour les quantificateurs montrent qu'un quantificateur existentiel allonge le chemin d'une longueur 2, et qu'un quantificateur universel le double et ajoute 3. Si on s'intéresse à la parité des longueurs de chemins, on remarque que le quantificateur existentiel ne la change pas, tandis que le quantificateur universel rend la longueur du chemin toujours impaire. Il est donc très facile de connaître la parité de la longueur des chemins dans le graphe : s'il y a au moins un quantificateur universel, les chemins sont de longueur impaire. On peut supposer que c'est le cas, sans perte de généralité.

Enfin, le déterminant de la matrice d'adjacence est non nul si et seulement s'il existe une couverture par cycle, si et seulement s'il existe un chemin de s à t. On a vu que le graphe modifié admet toujours une représentation par circuit du même ordre de taille, et c'est donc aussi le cas de sa matrice d'adjacence (le circuit est le même dans les deux cas). L'accessibilité dans un graphe donné par circuit étant PSPACE-difficile, c'est également le cas du calcul du déterminant d'une matrice donnée par circuit. □

Corollaire 2.11. *Pour tout entier $p \geqslant 2$, décider si le déterminant d'une matrice donnée par circuit est nul* modulo p *est* PSPACE*-complet.*

[4]On impose que tous les éléments apparaissent dans la décomposition, en utilisant des cycles de longueur 1.

Preuve. Si on réduit depuis le problème d'accessibilité unique, le déterminant ne peut prendre que deux valeurs, 0 ou 1. Ceci suffit à prouver que le même calcul *modulo* un entier quelconque reste PSPACE-complet. □

2.3 Autres résultats

On regarde dans cette partie deux variations du problème original que l'on prouve être de mêmes difficultés. La première d'entre elles montre que la difficulté est équivalente pour des matrices creuses. On s'intéresse ensuite aux matrices symétriques.

Corollaire 2.12. *Décider de la nullité du déterminant d'une matrice ayant au plus trois cœfficients non nuls par ligne et par colonne et donnée par circuit est* PSPACE*-complet.*

Preuve. On remarque que les matrices construites dans la réduction ont cette propriété. □

Théorème 2.13. *Décider de la nullité du déterminant d'une matrice symétrique donnée par circuit est* PSPACE*-complet.*

Preuve. On va continuer la réduction effectuée depuis `UQBF` et exploiter ainsi la forme des matrices créées. Remarquons d'abord que grâce au corollaire 2.11, on peut se restreindre à des matrices ayant comme déterminant 0 ou 1. On sait alors que pour une telle matrice M, $\det(MM^t) = \det(M)^2 = \det(M)$. Si on peut prouver que pour les matrices construites dans notre réduction, la matrice MM^t admet également une représentation par circuit de taille raisonnable, alors on pourra conclure.

Le cœfficient $(MM^t)_{ij}$ est défini par $\sum_{k=1}^{n} M_{ik}M_{jk}$. On va à nouveau s'intéresser aux graphes correspondant à M et MM^t (ce dernier étant pondéré). Pour cela, on définit le numéro (binaire) du sommet source-puits à 11, *i.e.* on lui donne le numéro de l'ancien puits. On a donc des sommets numérotés entre 3 et $2^{n+2} - 1$. On suppose dans la suite que les lignes et les colonnes des matrices considérées sont également numérotées de 3 à $2^{n+2} - 1$ (la première ligne sera donc la ligne 3).

Si on note G le graphe correspondant à M et G' le graphe (pondéré) associé à MM^t, on voit qu'il y a un arc $u \to v$ dans G' si u et v pointaient tous deux vers un même sommet w. Plus précisément, s'il existe k sommets pointés à la fois par u et v sans G, alors le graphe G' contient un arc de poids k de u vers v. Bien entendu, les rôles de u et v sont symétriques, et il existe le même arc de v vers u. Ceci correspond au fait que MM^t est une matrice symétrique.

Les boucles sur chaque sommet (sauf la source-puits) indiquent que les arcs existant dans G existent également dans G', ainsi que leurs arcs contraires (s'il y avait un arc $u \to v$ dans G, il y a les arcs $u \to v$ et $v \to u$ dans G'). De plus, on peut remarquer que dans la construction réalisée, deux sommets u et v reliés ne pointent jamais vers un même autre sommet w. Les arcs $u \leftrightarrow v$ dans G' sont donc de poids 1. Il y a une exception pour les arcs (ou l'arc si la dernière quantification est universelle) incidents à la source-puits. Puisqu'il n'y a pas de boucle sur ce sommet, ces deux arcs (ou cet arc) doivent être supprimés.

Pour les boucles sur un sommet u, il suffit de compter le nombre d'arcs sortants de u (y compris la boucle) dans G. Ce nombre est le poids de la boucle dans G'.

Enfin, pour chaque quantification existentielle, on a créé des deux arcs $t_0 \to t$ et $t_1 \to t$ (cf figure 1). Il faut donc rajouter les arcs $t_0 \leftrightarrow t_1$ dans G'.

On peut résumer ceci avec l'algorithme suivant pour déterminer, étant donnés i et j, la valeur $(MM^t)_{ij}$. On suppose ici que i et j varient entre 3 et $2^{n+2} - 1$, et on s'intéresse à leur écriture binaire. On note par analogie avec la preuve du lemme 2.7, τ_i (respectivement τ_j) les deux premiers bits de i (respectivement j), et b_i (respectivement b_j) les autres bits. Le cas A correspond au cas particulier de la source-puits. Les six cas du point B sont les arcs existants dans G, avec leurs arcs

contraires (et correspondent aux cas 1 à 6 de la preuve du lemme 2.7). Le cas C correspond aux arcs $t_0 \leftrightarrow t_1$, tandis que le cas D traite des boucles.

A. $i = 11$, $\tau_j = 10$: si $b_j = 0$, ou si $b_j = 1$ et que la quantification est existentielle, alors $(MM^t)_{ij} = (MM^t)_{ji} = 1$; si $i = j$, alors $(MM^t)_{ii}$ vaut 1 ou 2 selon que la quantification est universelle ou existentielle ; dans tous les autres cas où $i = 11$, $(MM^t)_{ij} = (MM^t)_{ji} = 0$;

B. Dans les cas suivants, $(MM^t)_{ij} = (MM^t)_{ji} = 1$:

   1. $\tau_i = \tau_j = 10$ et $b_j = b_i 0$;
   2. $\tau_i = \tau_j = 10$ et $b_j = b_i 1$ et la taille de b_i indique qu'on est dans le cas d'une quantification existentielle ;
   3. $\tau_i = 10$, $\tau_j = 11$, $b_i = b_j$ et φ est évaluée à vraie avec les valeurs mises aux valeurs indiquées par b_i ;
   4. $\tau_i = 11$, $\tau_j = 10$, $b_i = b0$ et $b_j = b1$ pour un certain b et la longueur de b indique qu'on est dans le cas d'une quantification universelle ;
   5. $\tau_i = \tau_j = 11$ et $b_i = b_j 0$ et la taille de b_i indique qu'on est dans le cas d'une quantification existentielle ;
   6. $\tau_i = \tau_j = 11$ et $b_i = b_j 1$;

C. $\tau_i = \tau_j = 11$, $b_i = b0$ et $b_j = b1$ pour un certain b et la longueur de b indique qu'on est dans le cas d'une quantification existentielle : $(MM^t)_{ij} = (MM^t)_{ji} = 1$;

D. $i = j$, alors $(MM^t)_{ii} = 2$, sauf si $\tau_i = 10$ et qu'on est dans le cas d'une quantification existentielle, la valeur étant alors 3 ;

E. dans tous les autres cas, $(MM^t)_{ij} = (MM^t)_{ji} = 0$.

Cet algorithme étant clairement polynomial, il peut être implémenté par un circuit de taille polynomiale. □

3 Le résultant

Cette dernière partie présente le résultant. On donne un survol très rapide de la théorie de l'élimination [45, Chapter XI] dont il est issu. On s'intéresse également à un algorithme pour le calculer, qui fonctionne en espace polynomial [12, Chapter 3]. Enfin, pour motiver l'étude faite dans la partie 2, on montre que les matrices intervenants dans le calcul du résultant sont descriptibles par des circuits de taille polynomiale (en le nombre et le degré des polynômes considérés).

3.1 Définitions

3.1.1 Résultant de deux polynômes

Soit P et Q deux polynômes en la variable X, à cœfficients sur un corps K. On cherche à déterminer dans quels cas ces deux polynômes admettent une racine commune dans la clôture algébrique $\bar{K}$ de K. On note $P(X) = a_p X^P + \cdots + a_0$ et $Q(X) = b_q X^q + \cdots + b_0$.

Définition 3.1. La *matrice de Sylvester* de P et Q est la matrice carrée de taille $p+q$ donnée par

$$\begin{pmatrix}
a_p & a_{p-1} & \dots & \dots & a_0 & 0 & & \\
 & a_p & a_{p-1} & \dots & \dots & a_0 & & \\
 & & \ddots & \ddots & & & \ddots & \\
 & & & a_p & a_{p-1} & \dots & \dots & a_0 \\
b_q & b_{q-1} & \dots & b_0 & & & & \\
 & b_q & b_{q-1} & \dots & b_0 & & & \\
 & & \ddots & \ddots & & \ddots & & \\
 & & & \ddots & \ddots & & \ddots & \\
 & & & & b_q & b_{q-1} & \dots & b_0
\end{pmatrix}.$$

où les espaces laissés blancs doivent être remplis par des zéros.

Le *résultant* de P et Q est alors défini comme le déterminant de la matrice de Sylvester.

On peut montrer que le résultant de P et Q est nul si et seulement si P et Q admettent une racine en commun dans $\bar{K}$ [35]. On peut alors s'intéresser à des généralisations de cette notion. En premier lieu, on peut regarder le cas où les deux polynômes sont multivariés. Dans ce cas, on peut se ramener au résultant classique, et le calcul se fait récursivement [17]. Deux autres généralisations possibles sont d'augmenter le nombre de polynômes, en les gardant univariés ou en passant au cas multivarié. Dans ces cas-ci, on ne sait pas trouver un polynôme unique permettant de décider de l'existence ou non de racines communes, mais un ensemble de polynômes qui admettent des racines en commun si nos polynômes originaux en admettaient [45]. Un autre cas intéressant consiste à considérer des polynômes multivariés homogènes. On cherche dans ce cas à décider de l'existence de racines communes non triviales, le uplet $(0, 0, \dots, 0)$ étant toujours racine d'un polynôme homogène. L'étude est en fait assez différente car l'homogénéité des polynômes facilite le travail, mais le fait que l'on cherche une racine non triviale élimine certains algorithmes traitant le cas non homogène. Dans le cas général de s polynômes à n variables, on est dans le même cas que pour les polynômes non homogènes, il faut considérer plusieurs polynômes pour décider de l'existence de racines communes.

Dans la suite, on considérera le cas de n polynômes homogènes $f_1, \dots, f_n$ à n variables. On va voir ci-après que dans ce cas là, la situation est un peu moins désespérée. Cette situation sera la seule considérée. Ainsi on parlera de *résultant multivarié* ou du *cas multivarié* en sous-entendant les autres hypothèses que l'on vient d'exprimer.

3.1.2 Résultant multivarié

On étend la définition 3.1 au cas de n polynômes homogènes à n variables. On cherche alors un polynôme R en les cœfficients des f_i tel que sur une valeur donnée de ces cœfficients, R s'annule si et seulement si les f_i ont une racine commune non triviale. On le cherche de plus de degré minimal.

Il se trouve que dans le cas multivarié, on ne sait pas toujours exprimer le résultant sous forme d'un déterminant. Il existe de nombreuses formulations qui ne donnent parfois qu'un multiple du résultant (pouvant s'annuler sans que les f_i aient une racine commune) [38, 21, 36, 13, 42, 28, 27, 20]. Une formulation intéressante qui va nous servir ici est celle de Macaulay [38] qui exprime le résultant sous la forme d'un quotient de deux déterminants. Cette formulation donne la valeur exacte du résultant si les cœfficients des f_i sont génériques (*i.e.* ce sont des indéterminées sans relation entre elles). La définition que l'on donne ici se base sur les matrices de Macaulay mais est un peu plus complexe. Elle a cependant deux avantages non négligeables, celui de toujours donner le bon résultat et celui d'être à la base de l'algorithme de Canny. Pour une présentation plus détaillée, on se reportera à [45].

On commence avec quelques notations. Soit $f_1, \ldots, f_n$ des polynômes homogènes en les variables $x_1, \ldots, x_n$. Notons $d_1, \ldots, d_n$ leurs degrés respectifs, et $d = 1 + \sum_{i=1}^{n}(d_i - 1)$. Pour $\alpha = (\alpha_1, \alpha_2, \ldots, \alpha_n)$, soit $\bar{x}^\alpha = x_1^{\alpha_1} x_2^{\alpha_2} \cdots x_n^{\alpha_n}$. Soit $X^d = \{\bar{x}^\alpha : \alpha_1 + \alpha_2 + \cdots + \alpha_n = d\}$ l'ensemble des monômes de degré d. Le cardinal de X^d est $N = \binom{d+n-1}{d}$.

Soit M une matrice à N lignes et N colonnes, indexées par les éléments de X^d. La ligne de M ayant comme indice le monôme $\bar{x}^\alpha$ représente le polynôme homogène de degré d

$$\frac{\bar{x}^\alpha}{x_i^{d_i}} f_i, \text{ où } i = \min\{j : x_j^{d_j} \text{ divise } \bar{x}^\alpha\}. \tag{1}$$

L'ensemble $\{j : x_j^{d_j}$ divise $\bar{x}^\alpha\}$ considéré est non vide grâce au choix effectué pour d. La matrice M est appelée *matrice de Macaulay*. Pour des raisons qui vont apparaître claires immédiatement, on note cette matrice Mac^n.

La formule (1) particularise l'ordre $x_1, \ldots, x_n$ sur les variables. On peut définir d'autres matrices de Macaulay en changeant l'ordre d'apparition des variables. Ainsi, en réordonnant les variables $x_1, \ldots, x_{k-1}, x_{k+1}, \ldots, x_n, x_k$, et en utilisant ce nouvel ordre pour le calcul du minimum dans la formule (1) (*i.e.* $1 \prec 2 \prec \cdots \prec k-1 \prec k+1 \prec \cdots \prec n \prec k$), on définit de la même manière la matrice Mac^k.

On peut alors montrer que le PGCD des déterminants des Mac^k *en tant que polynômes en les cœfficients des* f_i satisfait les conditions requises, à savoir de s'annuler si et seulement si les f_i ont une racine commune non triviale et être de degré minimal [45]. En effet, chaque Mac^k est un multiple du résultant (*i.e.* s'annule dès que les f_i ont une racine commune non triviale), et une considération sur les degrés permet de conclure.

3.2 Le résultant est dans PSPACE

On explique ici l'algorithme de Canny [12] montrant que le calcul du résultant est faisable en espace polynomial. D'autres algorithmes pour ce calcul existent qui utilisent les bases de Gröbner [4] ou une matrice de Sylvester [17], mais ces algorithmes sont en temps doublement exponentiel.

L'algorithme utilise de manière centrale les matrices de Macaulay. On explique dans un premier temps comment transformer la formulation du résultant en tant que PGCD des matrices de Macaulay en une définition récursive du résultant. On montre dans un deuxième temps comment cette définition récursive peut être transformée en un algorithme efficace, *i.e.* calculant le résultant en espace polynomial. On donne seulement l'idée de l'algorithme, sans prouver formellement ni sa correction, ni sa complexité. Les détails peuvent être trouvés dans le texte original.

3.2.1 Définition récursive du résultant

L'idée principale pour définir récursivement le résultant consiste à se ramener à des polynômes unitaires, en introduisant une nouvelle indéterminée.

On définit pour tout i le polynôme

$$\hat{f}_i(x_1, \ldots, x_n) = f_i(x_1, \ldots, x_n) + u_i x_i^{d_i},$$

où les u_i sont de nouvelles indéterminées. Les $\hat{f}_i$ sont considérés comme des éléments de $\mathbb{C}[u_1, \ldots, u_n][x_1, \ldots, x_n]$, *i.e.* des polynômes homogènes à cœfficients dans $\mathbb{C}[u_1, \ldots, u_n]$. On construit de la même façon que précédemment les matrices de Macaulay, mais à partir des $\hat{f}_i$ à la place des f_i. Pour tout k, on note a^k le déterminant de la matrice de Macaulay Mac^k, qui est un polynôme

en les u_i. On peut montrer que le résultant R a le même degré en u_k que a^k. Or a^k est un multiple de R, donc

$$a^k(u_1, \dots, u_n) = b^k(u_1, \dots, u_{k-1}, u_{k+1}, \dots, u_n)R(u_1, \dots, u_n), \tag{2}$$

où b^k est un polynôme en tous les u_i sauf u_k. On remarque également que R, a^k et b^k sont des polynômes unitaires et *rectangulaires*, *i.e.* dont le terme de plus haut degré (globalement) est le terme de plus haut degré en chaque u_i. La relation (2) va être à la base de l'algorithme.

Notons $R_j(u_1, \dots, u_j)$ le cœfficient dominant de R en tant que polynôme en $u_{j+1}, \dots, u_n$. On définit de la même façon $a^k_j(u_1, \dots, u_j)$ et $b^k_j(u_1, \dots, u_j)$. Comme R, a^k et b^k sont rectangulaires et unitaires, c'est aussi le cas des R_j, a^k_j et b^k_j. Comme b^k est indépendant de u_k, il vient que $b^k_k = b^k_{k-1}$. Ce qui conduit, pour tout k entre 1 et n, aux relations

$$\begin{cases} R_{k-1} \cdot b^k_{k-1} &= a^k_{k-1}, \\ R_k \quad \cdot b^k_{k-1} &= a^k_k \quad . \end{cases} \tag{3}$$

Ce système d'équation est proche de celui que l'on va utiliser dans l'algorithme. Avant cela, on va considérer de nouveaux polynômes, en une seule indéterminée. On note respectivement $\bar{R}_j(u)$, $\bar{a}^k_j(u)$ et $\bar{b}^k_j(u)$ les polynômes $R_j(u, \dots, u)$, $a^k_j(u, \dots, u)$ et $b^k_j(u, \dots, u)$. On obtient alors les équations suivantes, valables pour $k = 1, \dots, n$,

$$\begin{cases} \bar{b}^k_{k-1}(u) &= \frac{\bar{a}^k_{k-1}(u)}{\bar{R}_{k-1}(u)}, \\ \bar{R}_k(u) &= \frac{\bar{a}^k_k(u)}{\bar{b}^k_{k-1}(u)} \, . \end{cases} \tag{4}$$

Comme $\bar{R}_0(u) = 1$ (car R est unitaire), si l'on sait calculer les $\bar{a}^k_j(u)$, les deux relations de récurrence (4) permettent le calcul de $\bar{R}_n(u) = \bar{R}(u)$. Le cœfficient constant de ce polynôme est le résultant recherché. Il reste à voir comment calculer les $\bar{a}^k_j(u)$ et effectuer efficacement les divisions qui interviennent dans la récurrence.

3.2.2 Algorithme

Comme annoncé, on commence par expliquer le calcul des $a^k_j(u)$. On définit la matrice Mac^k_j comme étant la sous-matrice de Mac^k (construite ici à partir des f_i et non des $\hat{f}_i$, *i.e.* ne contenant pas les variables u_i) dont on a supprimé les lignes et les colonnes contenant le cœfficient $x_i^{d_i}$ de f_i, pour tout $i > j$. Ces lignes et colonnes sont celles qui auraient contenu les variables $u_{j+1}, \dots, u_n$ si on avait construit Mac^k à partir des $\hat{f}_i$. On peut alors se convaincre que $\bar{a}^k_j(u)$ est le polynôme caractéristique de Mac^k_j [12], qui se calcule en espace polynomial [19].

Étant capables de calculer les $a^k_j(u)$, on peut utiliser la récurrence (4) pour calculer le résultant. Les divisions effectuées sont des divisions par un polynôme unitaire. Elles s'effectuent donc sans réaliser de division sur le corps de base, et peuvent d'ailleurs être vues comme des calculs de déterminant de matrices de taille exponentielle [12]. Grâce aux méthodes de [19], le calcul des déterminants s'effectue en espace polynomial.

Ainsi, le calcul du résultant ne fait intervenir que des calculs de déterminants. La taille des matrices assure que les calculs se font en espace polynomial.

3.3 Résultant et circuits

Le but de cette partie est de montrer que les matrices de Macaulay d'un système de n polynômes homogènes à n inconnues sont descriptibles par un circuit de taille polynomiale en n. On s'intéresse à la matrice Mac^n pour fixer les idées, le cas de Mac^k pour k quelconque s'en déduisant très facilement.

Notons $A_n^d = \{(\alpha_1, \ldots, \alpha_n) : \alpha_1 + \cdots + \alpha_n = d\}$. On a donc $X_d = \{\bar{x}^\alpha : \alpha \in A_n^d\}$. On ordonne A_n^d dans l'ordre lexicographique. Si de même on note $A_k^d = \{(\alpha_1, \ldots, \alpha_k, \alpha_{k+1}, \ldots, \alpha_n) : \alpha_1 + \cdots + \alpha_k = d \wedge \alpha_{k+1} = \cdots \alpha_n = 0\}$, alors

$$\left|A_k^d\right| = \binom{d+k-1}{d} = \left|A_{k-1}^d\right| \cdot \frac{n+d-1}{n-1} = \left|A_k^{d-1}\right| \cdot \frac{d+n-1}{d}.$$

Or, dans l'ordre lexicographique, on énumère les A_k^d les uns après les autres, pour k croissant. Pour connaître le i^{e} élément de l'énumération, il suffit alors de calculer les $\left|A_k^d\right|$ pour k à partir de 1 jusqu'à avoir $\left|A_k^d\right| \geqslant i$. Ainsi, on peut déterminer que $\alpha_{k+1} = \cdots = \alpha_n = 0$. Puis en calculant les $\left|A_k^p\right|$ à partir de $p = 0$ jusqu'à avoir $\left|A_k^d\right| - \left|A_k^p\right| \leqslant i$, on peut connaître la valeur de α_k. On peut alors recommencer avec des valeurs de n et d différentes, jusqu'à connaître le n-uplet α apparaissant en i^{e} position dans l'énumération. Il suffit pour cela de s'arrêter lorsque l'une des deux inégalités précédentes est en fait une égalité. La première étape consiste en au plus n multiplications, tandis que la deuxième en nécessite au plus d. Ces deux étapes étant effectuées au plus n fois, retrouver le i^{e} élément de l'énumération nécessite de l'ordre de $n(n+d)$ multiplications.

Étant donné un couple (i, j), on cherche à déterminer le cœfficient Mac_{ij}^n. On commence par calculer les n-uplet α et β apparaissant respectivement en i^{e} et j^{e} positions dans l'énumération de A_n^d. La j^{e} ligne de la matrice, correspondant au monôme $\bar{x}^\beta$, représente le polynôme

$$\frac{\bar{x}^\beta}{x_i^{d_i}} f_i, \text{ où } i = \min\{j : x_j^{d_j} \text{ divise } \bar{x}^\beta\}.$$

On cherche donc le cœfficient en $\bar{x}^\alpha$ de ce polynôme. Or $\min\{j : x_j^{d_j} \text{ divise } \bar{x}^\beta\} = \min\{j : d_j \leqslant \beta_j\}$. A partir de cela, on calcule très aisément $\bar{x}^\beta / x_i^{d_i}$. Puis de même, il suffit de diviser $\bar{x}^\alpha$ par ce dernier monôme et de retourner le cœfficient de f_i correspondant au monôme obtenu. En effet, le cœfficient du monôme $\bar{x}^\alpha / (\bar{x}^\beta / x_i^{d_i})$ dans f_i est bien celui du monôme $\bar{x}^\alpha$ dans $(\bar{x}^\beta / x_i^{d_i}) \cdot f_i$.

L'algorithme décrit précédemment est de complexité polynomiale, il existe donc un circuit de taille polynomiale pour décrire chaque matrice de Macaulay. Ceci justifie l'étude du calcul des déterminants de matrices données sous forme de circuits. En effet, la difficulté de ce calcul signifie que l'on ne peut se contenter, pour calculer le résultant efficacement, de construire les circuits correspondant aux matrices de Macaulay pour ensuite calculer leurs déterminants. Le théorème 2.3 affirme en effet qu'une telle stratégie est PSPACE-complète.

Conclusion

Nous avons situé dans ce rapport la complexité du résultant de manière plus précise que ce qui était déjà connu, ainsi que donné des indices quant aux méthodes à utiliser pour affiner encore les bornes. Pour le calcul du résultant proprement dit, la borne PSPACE de Canny reste la meilleure connue. Cependant, le résultant est intéressant surtout dans sa version de décision, puisqu'il donne l'existence ou non d'une solution à un système polynomial homogène. Pour ce problème, on a prouvé des bornes inférieure et supérieure. On a en effet montré que le problème appartient à la classe AM, qui est en quelque sorte *à peine plus grande* que la classe NP (on a $\mathsf{NP} \subseteq \mathsf{AM} \subseteq \mathsf{RP}^{\mathsf{NP}} \subseteq \Pi_2$). On sait également que $\mathsf{AM} = \mathsf{BP}\cdot\mathsf{NP}$ [1], qui est la classe des problèmes qui se réduise de manière probabiliste à SAT. De plus, on a montré que ce problème était NP-difficile sous réduction probabiliste, c'est-à-dire que SAT se réduit de manière probabiliste au résultant. Ces deux bornes répondent quasiment à la question soulevée par Canny dans [12] de connaître la complexité exacte du résultant. Une borne inférieure de NP-difficulté est donnée dans ce papier, mais semble-t-il sans preuve.

D'autre part, nous avons montré que le calcul du déterminant de matrices données par circuit est PSPACE-complet, et que les matrices de Macaulay apparaissant dans le calcul du résultant peuvent être données par un circuit de taille raisonnable. Cela montre que pour espérer améliorer la borne supérieure, on ne peut pas se contenter du caractère *facilement descriptible* des matrices en jeu, mais qu'il faut étudier de plus près leur structure.

La borne inférieure de NP-difficulté sous réduction probabiliste n'est pas pleinement satisfaisante de par son caractère probabiliste. On s'interroge donc sur une méthode pour obtenir le même résultat sous une réduction déterministe, par oracle ou *many-one*. Une idée consiste, partant d'un système de s polynômes à n variables à ajouter $n - s$ variables et modifier un peu les polynômes. Les résultats de [33, 16] pourraient permettre d'assurer que le nouveau système est équivalent à celui de départ. Pour s'en assurer, on pourrait aussi travailler sur la forme spéciale qu'a la matrice jacobienne du système initial, celui-ci étant issu d'une instance de `Boolsys` et donc de `3-SAT`.

D'autre part, les matrices de grande taille peuvent jouer un rôle dans différents domaines. Notre formalisme de matrices succinctement décrites, et le premier résultat obtenu, pourraient être étendus à d'autres cas. En effet, [26, 25] s'intéressent à la simulation de la physique quantique et à sa complexité (quantique). Il se trouve que ces simulations font apparaître des matrices de grande taille que l'on peut facilement décrire. Notre approche pourrait permettre d'estimer la complexité classique de telles simulations. Il existe également d'autres domaines d'applications que le calcul du résultant en algèbre linéaire. On peut citer [15] qui s'intéresse à l'algèbre linéaire (et à la résolution de systèmes linéaires) en grande dimension pour trouver les racines de polynômes cyclotomiques. Une question laissée ouverte dans ce papier porte justement sur des polynômes représentés par circuits. Mentionnons enfin l'*Effective Nullstellensatz* qui stipule qu'un système polynomial $f_1, \dots, f_s$ n'admet aucune solution si et seulement s'il existe des polynômes $g_1, \dots, g_s$ tels que $\sum_i f_i g_i = 1$. Les bornes sur les degrés des g_i [9, 34] indiquent qu'on peut résoudre ce problème à l'aide d'un système linéaire de taille exponentielle.

Références

1. S. Arora et B. Barak : *Computational Complexity : A Modern Approach*. Cambridge University Press, 2009.
2. J. Balcázar : The complexity of searching implicit graphs. *Artificial Intelligence*, 86(1):171–188, 1996.
3. J. Balcázar, A. Lozano et J. Torán : The complexity of algorithmic problems on succinct instances. *Computer Science : Research and Applications (R. Baeza-Yates and U. Manber, Eds.)*, p. 351–377, 1992.
4. S. Basu, R. Pollack et M. Roy : *Algorithms in Real Algebraic Geometry*. Springer, 2003.
5. S. Berkowitz : On computing the determinant in small parallel time using a small number of processors. *Information Processing Letters*, 18(3):147–150, 1984.
6. L. Blum, M. Shub, F. Cucker et S. Smale : *Complexity and Real Computation*. Springer Verlag, 1998.
7. L. Blum, M. Shub et S. Smale : On a theory of computation and complexity over the real numbers : NP-completeness, recursive functions and universal machines. *Bull. Am. Math. Soc.*, 21(1):1–46, 1989.
8. A. Borodin, J. von zur Gathen et J. Hopcroft : Fast parallel matrix and GCD computations. *Information and control*, 52(3):241–256, 1982.

9. W. D. Brownawell : Bounds for the degrees in the Nullstellensatz. *The Annals of Math.*, 126(3):577–591, 1987.
10. P. Bürgisser, F. Cucker et P. De Naurois : The complexity of semilinear problems in succinct representation. *Computational Complexity*, 15(3):197–235, 2006.
11. J. F. Canny : A new algebraic method for robot motion planning and real geometry. *In Proc. FOCS'87*, p. 39–48, 1987.
12. J. F. Canny : *The complexity of robot motion planning*, vol. 1987 de *ACM Doctoral Dissertation Award*. MIT Press, 1988.
13. J. F. Canny, E. Kaltofen et L. Yagati : Solving systems of nonlinear polynomial equations faster. *In Proc. SIGSAM'89*, p. 121–128, 1989.
14. A. Chandra, L. Stockmeyer et U. Vishkin : Constant depth reducibility. *SIAM J. on Comput.*, 13:423, 1984.
15. Q. Cheng, S. P. Tarasov et M. N. Vyalyi : Efficient algorithms for sparse cyclotomic integer zero testing. *Theory of Comput. Syst.*, 2009. To appear.
16. A. Chistov, H. Fournier, L. Gurvits et P. Koiran : Vandermonde matrices, NP-completeness, and transversal subspaces. *Foundations of Computational Mathematics*, 3(4):421–427, 2003.
17. G. Collins : The calculation of multivariate polynomial resultants. *In Proc. 2nd ACM Symp. on Symbolic and Algebraic Manipulation*, p. 212–222, 1971.
18. M. Crasmaru : PSPACE problems with unique solutions. *IEIC Technical Report*, 100(705):129–136, 2001.
19. L. Csanky : Fast parallel matrix inversion algorithms. *SIAM J. on Comput.*, 5:618, 1976.
20. C. D'Andrea et A. Dickenstein : Explicit formulas for the multivariate resultant. *J. Pure Appl. Algebra*, 164(1-2):59–86, 2001.
21. A. Dixon : The eliminant of three quantics in two independent variables. *Proc. London Math. Soc.*, 6:468–478, 1908.
22. J. Feigenbaum, S. Kannan, M. Vardi et M. Viswanathan : Complexity of problems on graphs represented as OBDDs (extended abstract). *In Proc. STACS'98*, p. 216, 1998.
23. N. Fitchas, A. Galligo et J. Morgenstern : Precise sequential and parallel complexity bounds for quantifier elimination over algebraically closed fields. *J. Pure Appl. Algebra*, 67(1):1–14, 1990.
24. H. Galperin et A. Wigderson : Succinct representations of graphs. *Information and Control*, 56(3):183–198, 1984.
25. S. P. Jordan et P. J. Love : QMA-complete problems for stoquastic Hamiltonians and Markov matrices, 2009. arXiv :0904 :4755v1.
26. S. P. Jordan et P. Wocjan : Efficient quantum circuits for arbitrary sparse unitaries, 2009. arXiv :0904.2211.
27. D. Kapur et T. Saxena : Comparison of various multivariate resultant formulations. *In Proc. ISSAC'95*, p. 187–194, 1995.
28. D. Kapur, T. Saxena et L. Yang : Algebraic and geometric reasoning using Dixon resultants. *In Proc. ISSAC'94*, p. 99–107, 1994.
29. P. Koiran : Hilbert's Nullstellensatz is in the polynomial hierarchy. *Journal of Complexity*, 12(4):273–286, 1996.

30. P. Koiran : Circuits versus trees in algebraic complexity. *In Proc. STACS'00*, p. 35–54, 2000.
31. P. Koiran : The complexity of local dimensions for constructible sets. *Journal of Complexity*, 16(1):311–323, 2000.
32. P. Koiran et S. Perifel : VPSPACE and a transfer theorem over the complex field. *In Proc. MFCS'07*, p. 359–370, 2007.
33. P. Koiran, N. Portier et G. Villard : A rank theorem for Vandermonde matrices. *Linear Algebra and its Applications*, 378:99–108, 2004.
34. J. Kollár : Sharp effective Nullstellensatz. *J. Am. Math. Soc.*, 1(4):963–975, 1988.
35. S. Lang : *Algebra*. Springer, 2002.
36. D. Lazard : Résolution des systèmes d'équations algébriques. *Theoretical Computer Science*, 15(1):77 – 110, 1981.
37. A. Lozano et J. Balcázar : The complexity of graph problems for succinctly represented graphs. *In Proc. 15th Int. Workshop WG'89*, p. 277, 1990.
38. F. Macaulay : Some formulae in elimination. *Proc. London Math. Soc.*, 1(1):3, 1902.
39. K. Mulmuley : A fast parallel algorithm to compute the rank of a matrix over an arbitrary field. *Combinatorica*, 7(1):101–104, 1987.
40. C. Papadimitriou : *Computational complexity*. Addison-Wesley Reading, Mass, 1994.
41. C. Papadimitriou et M. Yannakakis : A note on succinct representations of graphs. *Information and Control*, 71(3):181–185, 1986.
42. B. Sturmfels : Sparse elimination theory. *In Proc. Computat. Algebraic Geom. and Commut. Algebra*. D. Eusenbud and L. Robbiano, eds., 1991.
43. J. Torán : Succinct representations of counting problems. *In Proc. 6th Int. Conf. Appl. Algebra, Algebraic Algo. and Error-Correcting Codes*, p. 415–426, 1988.
44. L. G. Valiant : Completeness classes in algebra. *In Proc. 11th ACM Symp. on Theory of Comput.*, p. 249–261, 1979.
45. B. L. van der Waerden : *Modern Algebra*. (third ed.) F. Ungar Publishing Co., New York, 1950.
46. K. Wagner : The complexity of combinatorial problems with succinct input representation. *Acta Informatica*, 23(3):325–356, 1986.

Juin 2009

Bruno Grenet, Université de Lyon, LIP – ÉNS Lyon, 46 allée d'Italie, 69364 Lyon Cedex 07, France
Courriel : `Bruno.Grenet@ens-lyon.fr` • *Url :* `http://perso.ens-lyon.fr/bruno.grenet/`