INRIA

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

Projet spaces

Systèmes Polynomiaux, Arithmétiques, Calculs Efficaces et Sûrs

Lorraine

THÈME 2B

Table des matières

1. Composition de l'équipe

SPACES *est un projet bilocalisé entre le* LORIA *(Nancy) et le* LIP6 *(Laboratoire d'Informatique de Paris 6).*

Responsable scientifique
Daniel Lazard [professeur, Université Pierre et Marie Curie (Paris 6) (UPMC), en délégation au LORIA]

Responsable permanent
Paul Zimmermann [directeur de recherche, INRIA]

Assistants de projet
David Massot [UPMC, temps partiel]
Hélène Zganic [LORIA, temps partiel]

Personnel CNRS
Jean-Charles Faugère [CR]
Dongming Wang [CR]

Personnel INRIA
Guillaume Hanrot [CR]
Vincent Lefèvre [CR]
Fabrice Rouillier [CR]

Enseignants-Chercheurs
Philippe Aubry [MC UPMC]
Luc Rolland [ATER Université Nancy 1]
Mohab Safey El Din [MC UPMC]

Chercheurs doctorants
Gwenolé Ars [DGA, co-encadré avec Marie-Françoise Roy, soutenance prévue en 2004]
Magali Bardet [allocataire moniteur, co-encadrée avec Daniel Augot, soutenance prévue en 2004]
Abdolali Basiri [bourse Sfere, soutenance prévue en 2003]
Jean-Paul Cerri [co-encadré avec Ch. Bachoc, soutenance prévue en 2005]
Solen Corvez [BDI CNRS, co-encadrée avec Marie-Françoise Roy, soutenance prévue en 2004]
Nicolas Gurel [LIX, co-encadré avec François Morain, soutenance prévue en 2003]
Amir Hashemi [bourse Sfere, soutenance prévue en 2005]
Philippe Trébuchet [allocataire moniteur, co-encadré avec Bernard Mourrain, soutenance le 16 décembre 2002]

Chercheur post-doctorant
Bill Allombert [ACI Cryptologie, depuis septembre 2002]

Chercheur invité
Yves Pétermann [août et octobre 2002]

Ingénieur SeDRE
Étienne Petitjean [depuis mai 2002]

Membres extérieurs
François Boulier [MC, Université de Lille 1]
Marie-Françoise Roy [professeur, Université de Rennes 1]

2. Présentation et objectifs généraux

2.1. Introduction

Le thème de notre projet est parfaitement résumé par le développement de l'acronyme SPACES en français et en anglais :

L'objectif principal est la résolution des systèmes d'équations et inégalités polynomiales (*Systèmes Polynomiaux*). Nous privilégions les méthodes algébriques (*Algebraic Computation*), qui sont plus robustes et souvent plus efficaces que les méthodes purement numériques.

En raison de la complexité élevée des problèmes abordés, pour obtenir des logiciels efficaces (*Efficient Software*), il faut associer aux algorithmes théoriques des méthodes d'implantation fines, et en particulier des *Arithmétiques* adaptées et efficaces, notamment les entiers et les rationnels de précision arbitraire, les flottants de grande précision, l'arithmétique d'intervalles, mais aussi les arithmétiques modulaires et p-adiques ou celle des nombres infinitésimaux.

Les systèmes polynomiaux ont des applications dans des domaines variés, aussi bien industriels qu'académiques. Cependant, un travail important est nécessaire pour définir des spécifications de sortie des logiciels qui soient adaptées aux problèmes posés, et aussi pour transcrire les problèmes du langage des applications vers une forme adaptée à la résolution. Ainsi, la résolution de problèmes (*Solving Problems*) est une part essentielle de notre activité de développement logiciel.

La variété même de ces applications rend nécessaire que les logiciels soient robustes (*Calculs Efficaces et Sûrs*) : d'une part, l'instabilité numérique est presque la règle dans les problèmes abordés, et une garantie sur les erreurs d'arrondis est indispensable ; d'autre part, il est certain que les hypothèses simplificatrices courantes nécessitées par certains algorithmes, telle la régularité, ne sont pas toujours satisfaites, et doivent donc pouvoir être évitées ou à tout le moins testées.

2.2. Aperçu historique

La résolution des équations et systèmes polynomiaux constitue depuis longtemps un problème fondamental. En sont témoins, par exemple, les efforts consacrés à la résolution par radicaux, jusqu'à ce que Galois résolve le problème vers 1830, mais aussi le fait que le théorème de d'Alembert, qui affirme l'existence de solutions complexes, soit appelé « *fundamental theorem of algebra* » par les anglo-saxons.

Ce n'est qu'à la fin du XIX^{e} siècle que sont apparus les premiers algorithmes généraux pour résoudre les systèmes polynomiaux, dans les travaux de Bézout, Sylvester, Kronecker et surtout Macaulay. Mais la complexité du problème rendait les calculs totalement impraticables, ce qui a conduit les mathématiciens à abandonner cette approche effective pour mettre l'accent sur des résultats qualitatifs et non effectifs. C'est ainsi qu'est apparue la géométrie algébrique moderne, avec son niveau d'abstraction qui la rend souvent presque ésotérique. Cette évolution est illustrée par le célèbre traité d'algèbre de van der Waerden [59] dans lequel les méthodes effectives ont été supprimées à partir de la quatrième édition ; en particulier le chapitre sur la théorie de l'élimination, qui traitait de la résolution effective des systèmes polynomiaux, a totalement disparu.

Avec l'invention des ordinateurs et l'apparition des systèmes de calcul formel, le problème de la résolution effective des systèmes polynomiaux est redevenu d'actualité, dès que, dans les années 1970, les calculs de PGCD et de factorisation des polynômes ont reçu des solutions satisfaisantes. Mais la complexité et la difficulté du problème font qu'il a fallu attendre la deuxième moitié des années 1980 pour commencer à pouvoir résoudre par logiciel des problèmes inaccessibles au calcul manuel.

2.3. Systèmes de dimension zéro

Stricto sensu, un système d'équations polynomiales est une formule

$$P_1 = 0 \text{ et } P_2 = 0 \text{ et } \ldots \text{ et } P_k = 0, \tag{1}$$

où les P_i sont des polynômes à plusieurs variables et à coefficients dans un corps K. Un tel système est généralement représenté par l'ensemble des polynômes P_i. Résoudre un tel système consiste à déterminer les valeurs des variables qui satisfont cette formule. Ces valeurs sont recherchées dans un corps algébriquement clos contenant K, ou dans le corps des réels si K est un corps réel[1]. Quand K est un corps fini à q éléments,

[1] Si $\mathbb{Z}$ est l'anneau des nombres entiers, l'existence de solutions dans $\mathbb{Z}$ est un problème indécidable.

ce qui est le cas en cryptographie, en ajoutant, une équation $x^q - x = 0$ pour chaque variable x on obtient un système dont toutes les solutions sont dans K et qui a les mêmes solutions dans K que le système de départ.

Un système est dit « de dimension zéro » si l'ensemble de ses solutions dans un corps algébriquement clos est fini. Dans ce cas, les solutions ne dépendent pas du corps algébriquement clos choisi. C'est la seule situation accessible au calcul purement numérique. Encore faut-il que le nombre d'équations soit égal au nombre de variables, et, même dans ce cas, les performances des solveurs numériques sont le plus souvent médiocres en raison des instabilités dues aux approximations.

Les techniques algébriques subdivisent le problème de la résolution en deux étapes : la première consiste à transformer le système en un ou plusieurs systèmes équivalents mais mieux adaptés et qui constituent ce que l'on peut appeler une *solution algébrique*. La deuxième étape consiste, dans le cas où K est un sous-corps des complexes, à calculer les valeurs numériques des solutions à partir de la solution algébrique.

Le calcul de la solution algébrique se décompose elle-même généralement en plusieurs étapes. La première est le plus souvent le calcul d'une base de Gröbner ; aussi de nombreux chercheurs ont assimilé, pendant plusieurs années, le calcul des bases de Gröbner à la résolution des systèmes polynomiaux.

La solution algébrique peut prendre diverses formes ayant chacune ses avantages et ses inconvénients. La forme la mieux adaptée au calcul des solutions numériques est la RUR (représentation univariée rationnelle [9]) qui consiste en une équation en une variable $f(t) = 0$ (où t est souvent une variable auxiliaire) et en l'expression des autres variables comme fractions rationnelles en t (quotients de deux polynômes).

À partir de la RUR, le calcul des valeurs numériques des solutions revient à calculer les racines d'un polynôme univarié. Ce n'est toutefois pas aussi simple qu'il n'y paraît car c'est généralement un polynôme de degré élevé ayant de très grands coefficients. En outre, dans le cas très fréquent où l'on s'intéresse aux solutions réelles satisfaisant certaines conditions de signe, il faut garantir une précision suffisante pour pouvoir déterminer le signe d'un certain nombre d'expressions dépendant d'une des racines du polynôme univarié.

Les algorithmes et les implantations développés par J.-C. Faugère et F. Rouillier sont très largement les plus performants existant actuellement. Ils permettent couramment de résoudre des systèmes ayant de l'ordre d'un millier de solutions. Néanmoins, un travail important est encore nécessaire pour optimiser encore ces algorithmes, mais surtout, à très court terme, pour rendre les développements récents accessibles à la communauté, sous forme de logiciels robustes, utilisables par des non spécialistes.

Il faut noter qu'en raison de la taille des systèmes dont la résolution est maintenant accessible, le facteur limitant provient souvent de l'efficacité de l'arithmétique utilisée.

2.4. Systèmes de dimension positive

Pour les systèmes ayant une infinité de solutions, on ne peut pas se limiter aux systèmes de la forme (1) : il faut souvent considérer des inéquations ($P_i \neq 0$) et, dans le cas réel, des inégalités ($P_i > 0$, ou encore $P_i \geq 0$)[2]. En outre, beaucoup de problèmes ne se limitent pas à une conjonction d'équations, inéquations et inégalités, mais se présentent sous la forme d'une expression avec quantificateurs (formule de la logique du premier ordre dont les objets atomiques sont des équations, inéquations ou inégalités).

Pour de tels systèmes généraux, la résolution se décompose *a priori* en trois étapes : en premier lieu, on connaît des algorithmes permettant d'associer à un tel système général un système sans quantificateur (formule du calcul propositionnel) ayant les mêmes solutions dans un corps algébriquement clos ou, s'il y a des inégalités, dans le corps des réels. Étant donné un tel système sans quantificateur, on connaît également des algorithmes permettant de décomposer l'ensemble des solutions en composantes irréductibles et/ou en composantes connexes (définies par de nouveaux systèmes). Enfin, étant donnée une telle composante, se pose le problème de l'étudier (la dessiner, déterminer ses singularités, sa topologie, etc.).

Un tel programme se heurte à des difficultés qui sont très loin d'être résolues. En premier lieu, les algorithmes résolvant les deux premières étapes ont une complexité élevée, généralement exponentielle en le nombre de variables et doublement exponentielle en le nombre d'alternances de quantificateurs. Cette

[2] Il en est de même en dimension zéro, mais, dans ce cas, les inéquations et inégalités peuvent n'être considérées qu'à la fin du calcul, pour éliminer certaines solutions.

complexité est asymptotiquement optimale, à une constante près *située en exposant*. Mais ces algorithmes ont une efficacité pratique si mauvaise qu'il n'y a aucun intérêt à essayer de les implanter.

Maintenant que nous disposons d'algorithmes efficaces en dimension zéro, un des objectifs principaux du projet est de mettre au point et d'implanter des algorithmes ayant une efficacité pratique acceptable en dimension positive, quitte à restreindre la classe des problèmes considérés.

Une autre catégorie de difficultés vient de ce que l'« étude » mentionnée plus haut n'a pas de spécification générale. Il y a donc un vaste travail nécessaire pour déterminer quelles spécifications de sortie sont utiles pour les applications, tout en étant accessibles au calcul.

Comme exemples de telles spécifications partielles, on peut citer certains des travaux récents du projet : un algorithme efficace pour produire au moins un point par composante connexe de l'ensemble des solutions réelles d'un système de la forme (1) ; un algorithme pour discuter le nombre de solutions réelles satisfaisant certaines inégalités pour un système de dimension zéro dépendant de paramètres (cet algorithme a été développé dans le cadre de la résolution d'un problème de mécanique céleste et a été utilisé en géométrie plane et en robotique) ; enfin un algorithme pour résoudre des systèmes surdéterminés dépendant de paramètres approchés, car issus de mesures.

2.5. Calculs efficaces et arithmétique

Pour résoudre un système polynomial, il est vain d'utiliser un algorithme de bonne complexité arithmétique, c'est-à-dire effectuant peu d'opérations sur les coefficients des polynômes, si les opérations sur ces coefficients sont elles-mêmes lentes. En effet, ces coefficients peuvent couramment atteindre des tailles de l'ordre du millier de chiffres. Il est donc primordial pour obtenir des logiciels efficaces (*Efficient Software*), ce qui constitue notre but final, de diminuer tous les facteurs intervenant dans le coût réel des algorithmes (*bit complexity*). En langage courant, on dirait « tous les coups sont permis ». On peut distinguer *grosso modo* deux classes d'améliorations possibles : celles concernant les coefficients des polynômes considérés, et toutes les autres (protocoles, ramasse-miettes, etc.).

D'autre part, les algorithmes de calcul formel nécessitent des développements spécifiques pour leur implantation. En effet, une des particularités du calcul exact est de manipuler un nombre important d'objets de grande taille variant fortement et de façon souvent mal contrôlée dans le temps. Il en découle toute une série de problèmes informatiques délicats (gestionnaire de mémoire, échanges de données, etc.) ne trouvant pas de solution satisfaisante parmi les outils standards.

2.5.1. Arithmétiques

On s'intéresse ici aux algorithmes et méthodes permettant de rendre plus efficaces les calculs sur les coefficients des polynômes du système à résoudre. Les coefficients de départ sont le plus souvent des entiers ou rationnels ($\mathbb{Z}$ ou $\mathbb{Q}$) ou des entiers modulaires ($\mathbb{F}_p$). Cependant, plusieurs autres arithmétiques auxiliaires peuvent être utilisées efficacement : les nombres flottants à précision fixée, tout en bornant la taille des calculs, permettent dans nombre de cas d'obtenir des résultats suffisants ; les nombres p-adiques sont souvent un intermédiaire très utile entre nombres entiers et modulaires (par exemple dans l'algorithme de factorisation de polynômes de Berlekamp-Zassenhaus) ; les nombres algébriques sont indispensables pour la résolution de certains systèmes avec paramètres ; enfin les infinitésimaux permettent l'étude de systèmes légèrement « déformés ».

On peut répartir ces arithmétiques en deux classes : les arithmétiques « exactes » (entiers, nombres algébriques) et les arithmétiques « approchées » (flottants, nombres p-adiques, infinitésimaux). La plupart des algorithmes sur les polynômes (resp. séries) s'appliquent aux objets de la première (resp. deuxième) classe. Si les meilleurs algorithmes connus demeurent les mêmes depuis longtemps pour les arithmétiques exactes, curieusement il n'en est pas de même pour les arithmétiques approchées. Citons pour preuve les travaux récents de Mulders [70] qui a montré qu'un gain d'environ 20% pouvait être obtenu sur un produit flottant sur n bits par rapport au produit entier[3], lorsque ce dernier est calculé via l'algorithme en $O(n^{1.59})$ de Karatsuba.

[3] Rappelons que le produit flottant sur n bits est le produit d'entiers tronqué aux n bits de poids fort.

2.5.2. Infrastructures logicielles

2.5.2.1. L'environnement de programmation.

En calcul formel, l'utilisateur est, pour l'heure, obligé de jongler avec différents logiciels. Les logiciels généraux tels Maple, MuPAD ou Mathematica possèdent de nombreuses fonctionnalités mais sont très limités lorsqu'il s'agit de faire du calcul intensif. D'un autre côté, il y a des logiciels spécialisés et langages de base (C/C++), rustiques d'utilisation mais permettant de faire certaines parties de calcul beaucoup plus efficacement. Pour utiliser de façon optimale les ressources dont on dispose, il est essentiel de recourir à un environnement de développement efficace. La diversité des structures de données utilisées en calcul formel rend impérative la définition de protocoles d'échanges assez complexes mais surtout souples et efficaces comme le protocole UDX (*Universal Data Exchange*) développé par les membres du projet et objet d'un brevet.

2.5.2.2. La gestion de mémoire.

L'implantation des algorithmes est une tâche primordiale à laquelle les membres du projet prennent une part active : en effet une mauvaise implantation peut rendre totalement inefficace un algorithme potentiellement bon. Un objectif du projet est d'implanter dans des langages de bas niveau (C/C++) toutes les parties critiques des nouveaux algorithmes. Il est à noter que la vitesse d'exécution d'un algorithme n'est pas le seul critère à prendre en considération ; bien souvent c'est la taille des données en mémoire qui limite l'utilisation d'un programme de calcul formel. Les membres du projet ont développé un algorithme de gestion de la mémoire original particulièrement adapté à ce type de données (dont la taille varie beaucoup en cours de calcul) et facile à utiliser dans un langage de bas niveau. Il faut que ce mécanisme de gestion de la mémoire reste efficace même lorsqu'on utilise plus de mémoire que la mémoire physiquement disponible (*swap*) ; de plus, ce mécanisme est bien adapté au calcul distribué et aux systèmes de cache des ordinateurs actuels.

2.6. Applications

Les applications sont fondamentales pour l'activité du projet pour plusieurs raisons, et, de ce fait, font pleinement partie de notre thématique de recherche.

La première de ces raisons est que les applications constituent des tests indispensables pour vérifier et valider l'efficacité de nos algorithmes. En effet, la complexité de la résolution d'un système polynomial dépend très irrégulièrement du problème posé, et la difficulté d'un problème tiré au hasard ne donne aucune indication sur celle d'un problème réel de taille comparable.

Une deuxième raison a été mentionnée plus haut : les applications sont nécessaires pour déterminer quels algorithmes sont à développer ou à optimiser en priorité. Inversement, c'est souvent en essayant de comprendre pourquoi une application particulière résistait à nos logiciels que de nouveaux algorithmes, beaucoup plus efficaces, ont été découverts.

Enfin, la résolution de problèmes inaccessibles aux autres approches existantes est la meilleure justification de l'importance de la résolution des systèmes polynomiaux et de la qualité de nos travaux.

Mais il y a là une difficulté spécifique : les problèmes susceptibles d'être résolus avec nos méthodes peuvent être modélisés de manières très variées. Avec l'idée reçue que les systèmes polynomiaux sont numériquement insolubles, la plupart des chercheurs et ingénieurs présentent leurs problèmes sous une forme telle, qu'il n'est pas immédiat qu'il s'agisse de systèmes polynomiaux. Même quand c'est apparent, ces systèmes sont présentés sous une forme « simplifiée » voire linéarisée qui augmente la difficulté du problème au point de le rendre insoluble.

Ainsi, il n'est pas possible d'utiliser nos logiciels comme une boîte noire, et il faut souvent remonter aux sources des problèmes (et donc comprendre le langage lié à l'application), afin de trouver une formulation susceptible d'être résolue. Ceci aboutit à ce qu'une part notable de nos publications relève de domaines scientifiques extérieurs au nôtre.

Ainsi, nous avons déjà publié ou avons des articles en préparation en robotique (robots parallèles et robots cuspidaux), traitement du signal (brevet en compression d'image), mécanique céleste (configurations stables

de n corps), biophysique (positions stables de protéines coniques) [18], codes correcteurs (décodage des codes cycliques au-delà de la distance minimale) et statistique (mélange de lois gaussiennes).

Il faut ajouter la cryptologie, qui, pour nous, est essentiellement un domaine d'application. Cependant, le temps de travail que nous y consacrons et les résultats obtenus en font une action de recherche à part entière.

3. Fondements scientifiques

3.1. Résolution algébrique

Le problème de la résolution des systèmes polynomiaux (non linéaires) ne représente qu'une partie des questions abordées par le calcul formel ; cependant ce problème fondamental pourrait bien être le meilleur exemple d'application où le calcul formel apporte le plus par rapport aux méthodes numériques. En ce domaine les avantages du calcul formel sont nombreux : exactitude des résultats puisque tous les calculs intermédiaires sont exacts ; garantie de trouver *toutes* les solutions (même si le nombre de solutions est infini) ; efficacité des algorithmes et des implantations de ces méthodes y compris pour des problèmes provenant d'applications industrielles.

Les membres du projet et les logiciels qu'ils ont écrits (FGb/RS représentant environ 200 000 lignes de code C/C++) sont déjà reconnus comme les leaders mondiaux pour le calcul des racines réelles et complexes des systèmes d'équations algébriques dans le cas particulier (important) où le nombre de solutions est fini. Ces logiciels ont déjà permis de résoudre plusieurs applications provenant du milieu académique ou industriel (traitement du signal, robotique, mécanique céleste, biophysique [18], cryptologie, codes correcteurs).

Le cas particulier de la résolution des systèmes algébriques où le nombre de solutions est fini est maintenant bien maîtrisé en théorie et en pratique et ceci en grande partie grâce aux travaux des membres du projet. L'outil de base pour résoudre ces systèmes est le calcul de bases de Gröbner. Plusieurs algorithmes parmi les plus efficaces ont été proposés par J.-C. Faugère et sont maintenant des algorithmes standards implantés dans tous les systèmes de calcul formel (Maple, Mathematica, Axiom, Magma, etc.). Cette étape permet de calculer toutes les racines complexes du problème. Dans la pratique les solutions cherchées vérifient d'autres contraintes : les racines doivent être réelles, réelles positives ou vérifier un certain nombre d'inégalités. Pour extraire ces racines réelles, la méthode de [9] consiste à ramener l'étude d'un problème à plusieurs variables à l'étude d'*un* polynôme en une variable, en transformant le résultat de l'étape précédente (base de Gröbner) en une liste de fractions rationnelles en une variable, exprimant les solutions en fonction des racines d'un polynôme (forme RUR). Des algorithmes [20] permettent ensuite de calculer très rapidement les racines réelles d'un polynôme univarié de façon exacte et avec une précision aussi grande que voulue. Tous ces algorithmes sont implantés dans les logiciels FGb et RS et constituent un solveur complet permettant de résoudre des problèmes ayant plusieurs milliers de solutions. Ces logiciels sont de plusieurs ordres de grandeur plus efficaces ou fiables que leurs concurrents.

Notre but est donc de faire avancer significativement en théorie et en pratique le problème de la résolution des systèmes d'équations algébriques *avec paramètres*. Lorsque le nombre de solutions est infini, les systèmes d'équations en nombres réels constituent des problèmes très mal résolus, en raison à la fois de la complexité sur-exponentielle des problèmes et de l'inefficacité des algorithmes actuellement implantés. Pourtant les applications potentielles de ces méthodes sont nombreuses : c'est la raison pour laquelle ce problème constitue un défi actuel du calcul formel.

Certains algorithmes pour calculer les racines réelles ont une très bonne complexité théorique mais sont inefficaces en pratique et très difficiles à implanter en machine. De nouveaux algorithmes (dont certains développés par des membres du projet) radicalement différents, visant à réduire le coût des calculs par rapport à la CAD [54] (*Cylindric Algebraic Decomposition*) utilisent fortement la possibilité de décomposer les idéaux (en particulier la décomposition en idéaux premiers) : plusieurs membres du projet ont proposé des algorithmes efficaces pour le calcul de ces décompositions. Un objectif du projet est d'implanter dans des langages de bas niveau (C/C++) et de façon très efficace (vitesse et gestion mémoire) ce type d'algorithmes sophistiqués et d'en faire une évaluation précise.

La spécification mathématique du résultat d'un calcul lorsque le nombre de solutions est infini est déjà un problème difficile en lui-même [1][2]. Dans ce cas, plusieurs expressions formelles des solutions sont envisageables mais il apparaît que les outils algébriques nécessaires pour calculer les zéros réels ou complexes peuvent se résumer à :

(Z) la résolution des systèmes sans paramètre (problème bien maîtrisé) ;
(Rad) le calcul d'un système de générateurs du radical d'un idéal ;
(Dec) la décomposition équidimensionnelle d'un idéal (c'est-à-dire séparer les points isolés des courbes et des surfaces dans l'ensemble des solutions) ;
(Prim) la décomposition d'un idéal en idéaux premiers, appelée décomposition première, qui consiste à compléter l'action de (Dec) par la séparation entre eux des différents points isolés, des différentes courbes, etc.

Il faut noter que le radical d'un idéal se déduit facilement d'une décomposition équidimensionnelle qui, elle-même, s'obtient très facilement à partir d'une décomposition en idéaux premiers. On pourrait donc s'attendre à ce que le calcul du seul radical soit moins coûteux que la décomposition en idéaux premiers, pourtant l'expérience montre que l'algorithme F_7 réalise ces tâches de façon efficace et à peu près dans le même temps. De plus, même si le temps pour calculer la décomposition en idéaux premiers est légèrement supérieur à celui du calcul du radical, ce sur-coût est largement compensé par le fait que ce calcul a permis de casser le problème initial en sous-problèmes indépendants de plus petite taille, sur lesquels le calcul de toutes les racines peut se poursuivre indépendamment et, éventuellement, en parallèle.

L'outil des bases de Gröbner, qui était le cœur de l'algorithmique pour les systèmes sans paramètres, n'est plus adapté pour calculer des décompositions, car il ne permet généralement de les détecter que tout à la fin d'un calcul difficile, ce qui empêche de prendre avantage de ces décompositions pour simplifier les calculs. De plus, même dans le cas des idéaux premiers, le nombre d'éléments d'une base de Gröbner n'est pas borné par une fonction du nombre des variables, contrairement aux ensembles triangulaires présentés plus bas.

Deux voies sont explorées par les membres de l'équipe pour contourner les bases de Gröbner. Plusieurs membres du projet ont proposé des algorithmes pour calculer une décomposition équidimensionnelle : ces algorithmes sont basés sur les ensembles triangulaires. Ces méthodes sont potentiellement meilleures que les bases de Gröbner puisqu'elles se placent d'emblée dans le radical de l'idéal (méthode ensembliste) et parce qu'elles effectuent des scindages dès le début de l'algorithme. Un autre avantage des ensembles triangulaires est que le nombre de générateurs est borné par le nombre de variables. L'étude des idéaux représentés par un ensemble triangulaire nous a permis de dégager des propriétés plus fines, trouvant des applications dans différents domaines. Nous avons ainsi nettement amélioré l'efficacité d'algorithmes que nous avons proposés en géométrie réelle effective. Notons que les ensembles triangulaires sont un outil essentiel en algèbre différentielle et que le travail mené sur ce sujet par les membres du projet bénéficie directement à la résolution des systèmes algébriques différentiels. Par ailleurs, les décompositions triangulaires sont bien adaptées aux problèmes de raisonnement automatique en géométrie où on a besoin de considérer les variétés plutôt que les idéaux, et de prendre en compte les cas de dégénérescence, qui nécessitent généralement un traitement séparé, aussi bien pendant le calcul que dans la présentation des résultats. Des progrès spectaculaires ont déjà été accomplis dans le domaine de l'efficacité de ces méthodes. Toutefois il reste encore un travail important d'optimisation et d'implantation fine pour traiter les applications réelles, et notamment les problèmes industriels.

Faugère a proposé un nouvel algorithme (F_7) qui calcule une décomposition en idéaux premiers, algorithme lui-même basé sur l'algorithme F_5. L'algorithme F_5 [25] est un algorithme récent et très efficace qui permet de calculer des bases de Gröbner sans calcul intermédiaire inutile en utilisant de façon intensive l'algèbre linéaire creuse. L'algorithme F_7 essaye de calculer une base de Gröbner traditionnelle, mais tente de détecter les matrices singulières qui apparaissent dans F_5 ; lorsque c'est le cas un scindage est effectué permettant de casser prématurément le problème. L'algorithme F_7 utilise également dans sa phase terminale les travaux récents sur la factorisation rapide des polynômes. Un premier prototype d'implantation de cet algorithme a été réalisé

et montre que bien souvent non seulement le calcul est plus rapide qu'un simple calcul de base de Gröbner mais, de plus, la qualité du résultat est fortement améliorée : le résultat est unique et mathématiquement bien spécifié ; la taille du résultat plus petite (parfois d'un facteur 1000) ; pour chaque composante la structure du résultat est plus simple ; celui-ci peut être présenté, au choix, sous la forme d'une base de Gröbner ou d'un ensemble triangulaire [2].

Toutefois de nombreuses questions restent ouvertes quant à la meilleure façon de représenter mathématiquement et informatiquement une composante première : nous comptons sur le retour d'expériences provenant des applications afin de déterminer la meilleure façon d'exprimer les solutions en fonction des besoins de l'algorithme sur les zéros réels.

Le couplage des algorithmes décrits dans ce paragraphe et des algorithmes issus de la partie réelle constituera, à terme, le noyau informatique d'un solveur complet pour les systèmes paramétrés.

3.2. Solutions réelles

Mots clés : *zéro réel*, *isolation*, *système algébrique*, *ensemble semi-algébrique*.

Glossaire

CAD Cylindrical Algebraic Decomposition

L'étude des zéros réels des systèmes algébriques est un sujet plutôt mal cerné d'un point de vue algorithmique. Si l'on sait désormais résoudre correctement les systèmes de dimension zéro (*i.e.*, admettant un nombre fini de solutions complexes), il n'en est pas de même pour les systèmes de dimension positive.

En géométrie réelle effective, l'objectif à long terme de beaucoup d'équipes est la résolution du problème général d'élimination des quantificateurs. La résolution de ce problème en géométrie réelle s'appuie principalement, d'un point de vue algorithmique, sur la résolution de systèmes généraux d'égalités et d'inégalités polynomiales (par résolution, on entend décider du vide et/ou fournir un point par composante connexe).

L'état de l'art sur l'implantation de méthodes résolvant ce problème est facile à établir.

D'un côté, nous avons l'algorithme de décomposition cylindrique algébrique (CAD) [54] qui est la seule méthode viable en pratique mais dont la complexité est désastreuse tant en ce qui concerne les calculs qu'en ce qui concerne la taille des résultats (beaucoup trop de points calculés, codage complexe des résultats intermédiaires : polynômes à coefficients dans des tours d'extensions). Cette méthode est basée sur un traitement récursif en le nombre de variables des polynômes considérés.

D'un autre côté, nous avons un certain nombre de méthodes théoriques de bonne complexité (au moins en ce qui concerne la taille de la sortie) essentiellement basées sur l'étude des points critiques de fonctions bien choisies (projection selon un axe, fonction distance à un point, etc.). Grigoriev et Vorobjov puis différents auteurs comme en particulier Basu, Pollack et Roy ou encore Traverso travaillent sur des méthodes basées sur l'étude des points critiques de fonctions bien choisies. Pour illustrer la philosophie générale, on sait par exemple que l'ensemble des points critiques de la fonction de projection par rapport à une coordonnée intersecte chaque composante connexe de la variété étudiée lorsqu'elle est lisse et bornée. De plus, on sait facilement modéliser ces points lorsque la variété est définie par une unique équation. Deux avantages importants sont à mettre au crédit des méthodes basées sur ce type de résultats :

- on se ramène en une étape (et non récursivement) à la résolution de problèmes *simples* (systèmes zéro-dimensionnels),
- la taille de la sortie est bien contrôlée.

Les différents auteurs ramènent le cas général des hypersurfaces définies par une unique équation à celui d'une hypersurface lisse et bornée, au moyen de déformations infinitésimales (une ou deux selon les versions). Pour généraliser ensuite ces techniques au cas des systèmes d'équations quelconques, des méthodes standards sont employées (prendre la somme des carrés des équations par exemple) et le passage au cas de systèmes d'inégalités se fait par une nouvelle série de déformations et/ou de mises en position générale.

Un rapport détaillé dû à Hoon-Hong [63] montre que l'implantation naïve de ces *méthodes théoriquement bonnes* est elle aussi désastreuse (en estimant simplement le temps d'affichage de certains résultats intermédiaires, il constate que sur des problèmes où la CAD prend moins d'une seconde, il faudrait plusieurs milliers d'années aux algorithmes dits théoriquement bons pour effectuer le calcul). Les mêmes observations peuvent être faites sur les propositions d'algorithmes les plus récentes.

En analysant la situation précisément, on peut voir que les méthodes *de bonne complexité théorique* proposées souffrent d'une mauvaise conception au sens informatique du terme. En effet, la façon dont elles ont été pensées est le reflet d'une démarche fréquente en mathématiques : on résout le problème dans une situation particulière où l'on peut soigneusement éviter tous les ennuis (par exemple en supposant les variétés lisses ou bornées). On se ramène ultérieurement à cette situation par une suite de transformations ayant peu d'influence sur la complexité théorique, mais catastrophiques en pratique : sommes de carrés (l'élévation au carré ne multipliant les degrés que par un facteur $O(1)$), déformations infinitésimales (cela ne change que le coût des opérations de base, bien souvent ignoré dans les calculs de complexité théorique), utilisation de changements de variables dits *génériques* qui tuent en général la structure des équations initiales si bien que l'on se retrouve systématiquement dans le *pire des cas*.

À court et moyen terme, nous avons deux objectifs principaux. En premier lieu, fournir des algorithmes efficaces pour le calcul d'au moins un point par composante connexe pour les variétés algébriques réelles. Ce type d'outil permettra alors de résoudre ponctuellement bon nombre de problèmes et, à plus long terme, servira de base pour la résolution des systèmes d'égalités puis, plus généralement, pour l'élaboration de méthodes générales pour la résolution du problème d'élimination des quantificateurs. L'idée est de préserver la philosophie des méthodes théoriquement bonnes (étude de points critiques de fonctions bien choisies), assurant une bonne complexité pour la taille de la sortie, tout en rendant possible le calcul effectif dans les applications réelles.

En second lieu, de nombreuses questions relevant de l'élimination des quantificateurs peuvent être réduites par des opérations purement algébriques (projection, décomposition d'idéaux) à l'étude de la topologie des solutions de systèmes dépendant d'un nombre réduit de variables. Ces opérations algébriques sont coûteuses, mais les algorithmes récents sont beaucoup plus efficaces que les méthodes générales d'élimination des quantificateurs. De telles réductions sont, en particulier, possibles pour les systèmes dépendant de paramètres, pour lesquels elles permettent de se ramener à l'étude de composantes connexes dans l'espace des paramètres.

Ainsi, dans les deux cas, notre point de vue est d'étudier géométriquement les variétés, dans leur globalité, en limitant les calculs, autant que faire se peut, à ce qui est indispensable à la résolution du problème considéré ; au contraire, les méthodes générales introduisent une explosion combinatoire liée à la recherche d'informations qui seront inutiles *in fine*. De nombreuses études ont déjà été effectuées dans ce sens et quelques résultats théoriques intéressants sont disponibles. Ils sont encore parcellaires, mais semblent converger vers une méthodologie cohérente, qui a un grand caractère de généralité.

3.3. Arithmétiques

Dans la plupart des arithmétiques étudiées, l'essentiel des problèmes se rencontre lors de l'étude des algorithmes de multiplication, division et racine carrée, ces deux dernières opérations pouvant elles-mêmes se ramener à la multiplication. Dans le cas des arithmétiques exactes, selon la taille des objets manipulés (notée n par la suite), on utilise généralement soit une méthode naïve de complexité $O(n^2)$, soit une méthode à la Karatsuba de complexité $O(n^{1.59})$, pour laquelle les variantes de type Toom-Cook donnent une complexité $O(n^{\log(2r+1)/\log(r+1)})$, soit des méthodes basées sur la transformée de Fourier rapide de complexité $O(n^{1+\varepsilon})$. Le cas des arithmétiques approchées est similaire, excepté que la prise en compte des spécificités du problème conduit usuellement à de meilleures constantes multiplicatives.

Si relativement peu de résultats nouveaux ont été obtenus ces dix dernières années pour les calculs entiers[4], plusieurs algorithmes originaux ont été inventés pour les calculs flottants (astuce de Karp et Markstein pour

[4]Citons cependant l'algorithme de division rapide à la Karatsuba inventé par Jebelean et clarifié par Burnikel et Ziegler (rapport 98-1-022, MPI Saarbrücken, 1998).

la division et la racine carrée via l'itération de Newton en 1997 [64], produit et division « courts » de Mulders [70], et « produit médian » de Hanrot-Quercia-Zimmermann en 2000 [62]), ce qui laisse penser que de nombreux autres résultats sont à attendre.

Le domaine d'application de ces algorithmes est très vaste ; par exemple, l'itération de Newton est utilisée jusque dans les microprocesseurs actuels pour effectuer les divisions et racines carrées, même en simple précision !

Sur le plan des implantations, d'excellents outils d'arithmétique exacte sont largement répandus. Citons par exemple la bibliothèque GNU MP, développée par T. Granlund, qui semble à l'heure actuelle offrir les meilleures performances sur une large palette d'architectures courantes.

La situation des arithmétiques approchées est bien plus complexe. Le premier problème qui se pose pour l'utilisateur (mais auquel le développeur n'a hélas pas toujours pensé) est celui de la sémantique des opérations (une définition précise de cette sémantique est indispensable pour garantir un comportement déterministe des programmes). Prenons par exemple le cas de la multiplication de deux nombres flottants de n bits ; seuls les n premiers bits du résultat auront un sens, et les méthodes de calcul tenteront souvent d'exploiter le fait que seuls n bits seront renvoyés ; toutefois, certaines bibliothèques ignorent par là-même complètement toute retenue pouvant surgir de la partie « non significative », entachant ainsi d'erreur les derniers bits du résultat retourné.

La définition d'une sémantique précise est particulièrement critique dans le cas d'algorithmes hybrides, où l'utilisation d'algorithmes approchés doit quand même rester dans un cadre où la rigueur est suffisante pour s'apercevoir le cas échéant que les calculs ont perdu toute signification.

Historiquement, ce problème a déjà été rencontré pour les nombres flottants double précision, et a donné lieu, à l'issue de longues tergiversations[5], à la définition de la norme IEEE-754.

Plus récemment, on a vu fleurir diverses tentatives de généralisations partielles de la norme IEEE-754 en précision arbitraire, avec des implantations associées. Aucune n'a à ce jour réussi à s'imposer comme standard. On peut citer Arithmos (Université d'Anvers), Mathematica, etc. Les membres de SPACES ont eux-mêmes implanté une généralisation de la norme IEEE-754, au-dessus de la bibliothèque GMP précédemment mentionnée. Cette bibliothèque, baptisée MPFR (pour *Multiple Precision Floating-Point Reliable*) est distribuée avec GMP depuis la version 3.1 d'août 2000.

L'arithmétique des corps premiers finis a été bien étudiée, et on sait bien comment l'optimiser en fonction de la caractéristique, en utilisant selon la taille du corps, des techniques spécifiques ou l'arithmétique des entiers. Mais le problème est plus compliqué pour les extensions de corps, qui non seulement dépendent de deux paramètres (la caractéristique et le degré de l'extension), mais dépendent aussi du choix d'un générateur, qui peut avoir une grande influence sur la complexité arithmétique. Il en résulte qu'il y a de nombreuses situations (notamment les corps de caractéristique 2 et de taille moyenne qui interviennent en cryptologie) où l'on ne sait pas optimiser correctement l'arithmétique.

Nos objectifs incluent, le lecteur l'aura compris, l'étude (de la conception à l'implantation *efficace*) d'algorithmes pour diverses arithmétiques exactes ou approchées. Ces algorithmes seront, bien entendu, pensés en étroite relation avec leur cadre d'utilisation, à savoir certaines des méthodes hybrides mentionnées dans la section 3.4.

L'arithmétique peut aussi offrir des problèmes d'intérêt intrinsèque. Par exemple, autour de V. Lefèvre, nous visons, en collaboration avec le projet Arénaire (INRIA Rhône-Alpes et ENS Lyon) la détermination efficace des pires cas des fonctions élémentaires. V. Lefèvre a mis au point dans sa thèse [7] un algorithme performant à base d'approximations polynomiales hiérarchiques, mais qui nécessite de « descendre » jusqu'à une approximation de degré 1, dont la plage de validité est limitée à de l'ordre de 2^{16} points, ce qui rend la double précision (2^{53} points) faisable mais difficile, et la quadruple précision (2^{113} points) impossible.

Des travaux récents de Coppersmith et Elkies [56][57][55][60] nous ont permis d'élaborer un algorithme basé sur la réduction des réseaux, utilisant des approximations de degré 2 ou plus [49]. Si cet algorithme peut être implanté de manière suffisamment efficace, cela pourrait permettre de traiter la double précision plus rapidement et pourquoi pas d'approcher la quadruple précision. Notons par ailleurs que l'algorithme LLL est

[5] http://www.cs.berkeley.edu/~wkahan/ieee754status/754story.html.

un exemple typique où les méthodes hybrides permettent d'améliorer grandement les performances, même si les problèmes de stabilité numérique ne sont peut-être pas encore tous compris. Cela pourrait donner lieu à une étude approfondie.

À l'issue de l'étude des pires cas pour les fonctions élémentaires, on pourrait envisager - toujours en collaboration avec le projet Arénaire - la réalisation d'une bibliothèque garantissant l'arrondi exact sur les fonctions élémentaires (exp, log, sin, cos, etc.) en double précision.

Par ailleurs, avec le développement de nos applications en cryptologie, il faudra envisager aussi la réalisation de bibliothèques pour l'arithmétique des différents corps finis intervenant dans ce domaine. Cette tâche a commencé en 2002 avec le recrutement de Bill Allombert dans l'équipe comme post-doctorant.

3.4. Méthodes hybrides

Ces dernières années ont vu l'émergence de nombreux algorithmes basés sur des arithmétiques « hybrides ». Une arithmétique hybride consiste en l'utilisation conjointe de plusieurs arithmétiques de base comme celles mentionnées dans la section 2.5.1. Par exemple, on remplace dans un algorithme les calculs entiers par des calculs flottants, tout en pouvant décider de la plupart des signes des expressions rencontrées - à condition que l'arithmétique flottante ait une sémantique précise - et on n'effectue les calculs entiers que dans les rares cas où l'arithmétique flottante échoue. On parle alors d'algorithme hybride symbolique-numérique. Ces algorithmes, lorsqu'ils sont bien conçus, permettent de combiner les avantages d'une arithmétique exacte (permettant par exemple de tester l'égalité à zéro du résultat) et d'une arithmétique approchée (principalement la rapidité).

Les algorithmes hybrides constituent un sujet de recherche particulièrement actif ; notons en particulier que le comité d'organisation permanent de la conférence ISSAC a décidé d'encourager les recherches autour de ce type d'algorithmes, avec la création en 2001 de divers prix pour des travaux dans ce domaine.

On peut également utiliser des arithmétiques hybrides lors d'une pré-exécution pour produire des « traces de calcul ». Ainsi un premier calcul de base de Gröbner avec des coefficients modulaires va produire une « trace » des paires critiques ayant conduit ou non à des polynômes nuls. Cette trace est ensuite réutilisée pour « guider » les calculs - beaucoup plus coûteux - avec des coefficients entiers, et donc éliminer la plupart des calculs « inutiles ». Cette technique est pertinente pour tous les algorithmes opérant sur des ensembles, où l'ordre des opérations est *a priori* indifférent, mais joue un rôle important et imprévisible en pratique.

Notre objectif principal est de poursuivre l'effort entrepris depuis quelques années pour mettre au point des algorithmes hybrides ayant une meilleure efficacité à la fois théorique et pratique. Cet effort porte à la fois sur les arithmétiques elles-mêmes (voir section 3.3) et leur utilisation pour la résolution de systèmes polynomiaux. Citons pour cette seconde partie un nouvel algorithme hybride d'isolation des racines réelles d'un polynôme de $\mathbb{Z}[x]$ [20]. Pour les algorithmes hybrides entiers-flottants, nous basons nos implantations sur la bibliothèque MPFR (http://www.mpfr.org). Nous essayons également d'étendre nos investigations à des arithmétiques hybrides non-classiques, par exemple l'utilisation conjointe de flottants et de nombres modulaires, de nombres p-adiques, etc.

4. Domaines d'application

4.1. Introduction

Dans la section 2.6, nous avons décrit le rôle et l'importance des applications pour notre projet, ainsi que la variété des domaines scientifiques concernés. Dans cette section, nous nous limitons aux applications les plus importantes, auxquelles des membres du projet consacrent une part appréciable de leur activité.

Il s'agit de la simulation et du contrôle des robots parallèles (section 4.2), de la conception des robots série (section 4.3), du raisonnement géométrique (section 4.4), de la cryptologie (section 4.5) et de la mécanique céleste (section 4.6).

4.2. Robots parallèles

Mots clés : *robot parallèle*, *planification de trajectoire*, *étalonnage*, *calcul exact*.

Participants : Solen Corvez, Jean-Charles Faugère, Luc Rolland, Fabrice Rouillier.

Glossaire

MGD Modèle Géométrique Direct

Les manipulateurs que nous étudions sont des robots parallèles généraux : les hexapodes sont des mécanismes complexes constitués de six chaînes cinématiques souvent identiques, d'une base (corps rigide fixe comprenant six joints ou articulations) et d'une nacelle (corps rigide mobile contenant six autres joints).

La conception et l'étude de robots parallèles nécessitent l'établissement et la résolution de modèles géométriques directs (calcul des coordonnées absolues des points d'attache de la plateforme connaissant la position et la géométrie de la base, la géométrie de la plateforme ainsi que les distances entre les points d'attache des chaînes cinématiques à la base et à la nacelle) et inverses (distances entre les points d'attache des chaînes cinématiques à la base et à la nacelle connaissant la position absolue de la base et de la nacelle). Le modèle géométrique inverse se résout facilement. C'est par conséquent sur la résolution du modèle géométrique direct que portent nos efforts.

L'étude du modèle géométrique direct pour les robots parallèles est une activité récurrente de plusieurs membres du projet. On peut dire que les progrès effectués dans ce domaine illustrent parfaitement l'évolution des méthodes de résolution des systèmes algébriques. L'intérêt porté à ce sujet est ancien. Les premiers travaux auxquels ont participé des membres du projet [6][5] ont essentiellement porté sur l'étude du nombre de solutions (complexes) du problème. Ils ont souvent été illustrés par des calculs de bases de Gröbner effectués avec le logiciel GB (voir section 5.2). Un des points remarquables de cette étude est certainement la classification proposée dans [3]. Les efforts suivants ont porté sur les racines réelles et le calcul effectif des solutions [8]. Les études se sont ensuite poursuivies au gré des divers progrès algorithmiques réalisés, jusqu'à ce que les outils développés permettent d'appréhender des problèmes non académiques. C'est alors (1999) que les divers efforts se sont concrétisés par un contrat industriel avec la PME vosgienne CMW (Constructions Mécaniques des Vosges Marioni) pour l'élaboration d'un robot dédié aux machines-outils. De nouveaux problèmes apparaissent désormais et concernent deux de nos directions de recherche :

- le calcul en temps réel du modèle géométrique direct. Ceci pourrait se faire en particulier en générant des programmes de calcul numérique à partir d'un calcul exact suffisamment générique (base de Gröbner).
- l'étude de systèmes de dimension positive pour, par exemple, pouvoir traiter des informations dépendant du temps (dimension 1) ou encore permettre de relâcher certains paramètres du problème pour mieux les étudier.

4.3. Robots série

Mots clés : *robot série*, *classification*, *cuspidalité*, *calcul exact*.

Participants : Solen Corvez, Jean-Charles Faugère, Fabrice Rouillier, Mohab Safey El Din.

Les mécanismes articulés des robots que l'on rencontre dans l'industrie ont tous une géométrie standard. Or, la diversité des tâches à effectuer mène à l'étude d'autres types de robots dont les paramètres de conception diffèrent des paramètres habituels et qui pourraient présenter de nouvelles qualités, telles que par exemple la stabilité ou la possibilité d'effectuer certains types de trajectoires.

Cependant une difficulté importante freine l'utilisation industrielle de ces nouveaux robots. Des études récentes (Ph. Wenger [73][75][74] et J. El Omri [71]) ont montré que ceux-ci peuvent présenter un comportement qualitativement différent de celui des robots actuellement utilisés dans l'industrie et permettent d'autres changements de posture. Ces robots, appelés cuspidaux, ne peuvent pas être pilotés comme les autres. La

majorité des robots étant en fait cuspidaux, les robots industriels actuellement sur le marché forment une sous-classe très restreinte de l'ensemble des robots possibles. Un répertoire systématique de l'ensemble des robots cuspidaux serait d'un grand intérêt pour le concepteur et l'utilisateur.

Ce projet s'insère dans une tendance actuelle en robotique qui consiste soit à concevoir le robot pour que ses performances soient optimales pour une application donnée tout en préservant la possibilité de l'utiliser pour une autre tâche, soit à le spécialiser au maximum pour une application afin d'en réduire le coût et d'augmenter sa sécurité de fonctionnement...

L'étude du comportement d'un robot lors d'un changement de posture se ramène d'un point de vue mathématique à l'étude de l'existence d'une racine triple d'un certain polynôme de degré 4 dépendant des paramètres.

4.4. Raisonnement géométrique

Mots clés : *preuve de théorème*, *découverte automatique de théorème*, *géométrie différentielle*, *dessin automatique*, *équation implicite de surface*, *classification de nombre de solutions réelles*, *interpolation de Birkhoff*.

Participants : Philippe Aubry, Daniel Lazard, Fabrice Rouillier, Mohab Safey El Din, Dongming Wang.

Glossaire

CAGD *Computer-Aided Geometric Design*

Des méthodes et logiciels développés dans notre équipe ont été améliorés et étendus pour l'application au raisonnement géométrique. En particulier, nous avons découvert et prouvé plusieurs théorèmes en géométrie différentielle, montré comment dessiner des diagrammes automatiquement et comment calculer effectivement l'équation d'une surface rationnelle en présence de points base, résolu un problème géométrique soulevé par deux chercheurs chinois, ainsi que le problème d'interpolation de Birkhoff.

Le raisonnement géométrique est un sujet de recherche fondamental où la résolution algébrique (base de Gröbner, ensemble triangulaire, résolution réelle) a trouvé une application importante. Il a de nombreuses applications dans le domaine de la géométrie informatique, en particulier en CAGD (résolution des contraintes géométriques, mise sous forme implicite des courbes et surfaces rationnelles, etc.) et en vision par ordinateur (ajustement de modèles, dérivation des propriétés géométriques, etc.). Le problème du raisonnement géométrique consiste à étudier et établir automatiquement des relations entre objets géométriques, par exemple à prouver un théorème ou découvrir une relation géométrique inconnue. Il s'agit d'attaquer ce problème par des méthodes algébriques en traduisant des relations géométriques en expressions algébriques.

Par exemple, nous avons développé des méthodes et outils d'élimination (différentielle) pour prouver des théorèmes et dériver des relations inconnues en géométrie élémentaire et dans la théorie locale des surfaces en géométrie différentielle. Plusieurs nouveaux théorèmes ont été établis au moyen de nos méthodes et expérimentations. Une application de l'ensemble des méthodes et techniques développées récemment dans notre équipe permet de résoudre systématiquement un autre problème géométrique (la détermination du nombre de solutions réelles d'un système dépendant de paramètres en fonction de ces paramètres) soulevé par deux chercheurs chinois [65]. Nous avons également réussi à résoudre le problème d'interpolation de Birkhoff par la méthode des points critiques [72]. Plus récemment, nous avons conçu une approche pour le dessin automatique de diagrammes géométriques et une méthode nouvelle et simple pour la mise sous forme implicite des courbes et surfaces paramétriques rationnelles (voir la section 6.5).

4.5. Cryptologie

Mots clés : *cryptologie*, *système à clé publique*, *cryptanalyse algébrique*, *jacobienne de courbe algébrique*.

Participants : Magali Bardet, Abdolali Basiri, Jean-Charles Faugère, Nicolas Gurel, Guillaume Hanrot, Gwenolé Ars.

Glossaire

HFE *Hidden Field Equation*

La cryptologie est le domaine de l'informatique qui s'intéresse aux problèmes de sécurité de l'information sous un sens général. Lorsqu'un nouveau cryptosystème est proposé, plusieurs outils d'analyse sont développés pour s'assurer de sa robustesse (cryptanalyse linéaire, cryptanalyse différentielle). Dans le cadre de SPACES, nous proposons une façon supplémentaire d'analyser un cryptosystème : la cryptanalyse algébrique. Nous montrons que le fait de poser le problème de manière algébrique et d'utiliser les algorithmes/résultats sur les bases de Gröbner permet d'améliorer significativement les attaques connues pour plusieurs cas : HFE et les registres filtrés.

Les travaux dans ce domaine d'application se situent dans le cadre de l'action PolyCrypt de l'Action Concertée Incitative (ACI) Cryptologie (voir 8.1).

Comme toutes les applications traitées par SPACES, la première étape de la cryptanalyse algébrique consiste à mettre en équation le problème. C'est parfois une étape très difficile : c'est le cas en particulier du système à chiffrement AES (standard de chiffrement retenu par l'administration américaine). Diverses mises en équation du problème (ou d'une version simplifiée du cryptosystème) sont ainsi en cours d'étude. Certaines mises en équation font intervenir plusieurs équations en plusieurs milliers de variables ; d'autres mises en équation font intervenir beaucoup moins de variables, mais sont de degré plus élevé et/ou sur un corps de plus grande taille. Le cryptosystème AES constitue donc un des défis principaux à plus long terme pour les techniques de SPACES, même s'il semble hors de portée des algorithmes existants. Une fois la mise en équation effectuée, la résolution se fait par un calcul de base de Gröbner. En première approximation, il suffit juste de changer l'arithmétique sur les coefficients dans les algorithmes généraux comme F_5 [25]. Cependant, pour le cas très particulier de GF(2), il a été nécessaire de refondre complètement l'algorithme et surtout l'implantation afin de tenir compte de l'action de l'automorphisme de Frobenius ($f^2 = f$) qui rajoute des relations triviales ; de plus, une nouvelle bibliothèque d'algèbre linéaire spécifique à GF(2) a été créée ; cette version, que nous nommons $F_5/2$, est plus rapide d'un facteur environ 100 que la meilleure implantation de F_4 [10].

Un autre problème fondamental auquel on est confronté dans les problèmes issus de la cryptographie est de reconnaître rapidement si un système algébrique est « aléatoire » ou non. Pour cela, nous avons été amenés à établir de nouvelles bornes de complexité théorique très précises pour les systèmes algébriques aléatoires dans un corps fini et les systèmes surdéterminés. Ceci est un travail en collaboration avec Bruno Salvy (projet Algo). Nous donnons en particulier l'équivalent de la borne de Macaulay $1 + \sum_i (d_i - 1)$ (Lazard 83). Ces bornes sont valides y compris lorsque le nombre d'équations m est plus grand que le nombre de variables n. Le calcul d'un développement asymptotique se fait en calculant les séries génératrices associées, puis en utilisant la méthode des points cols coalescents. Par exemple, le degré maximal d'un calcul de base de Gröbner pour n équations de degré 2 sur le corps GF(2) est :

$$d_n = \lambda_0^{-1} n - \lambda_0^{-4/3} \lambda_1 n^{1/3} + O(n^{-1/3})$$

avec

$$\lambda_0 = 3/2\ \sqrt{3} + 5/2 + 1/2\ \sqrt{72 + 42\ \sqrt{3}} \approx 11.11$$

et $\lambda_1 \approx 1.0034$ s'exprime à l'aide du plus grand zéro de la fonction Ai d'Airy.

Cette borne est quasi exacte dès que $n \geq 3$ et permet de calculer exactement les tailles des matrices qui apparaissent dans l'algorithme F_5 [25]. Ces bornes théoriques sont particulièrement utiles pour faire la distinction en pratique entre un système aléatoire (difficile) et un système provenant d'un cryptosystème particulier (comme HFE).

Lorsque les ressources sont limitées ou bruitées (téléphone GSM) et qu'on désire un débit élevé, on utilise des techniques de chiffrement différentes : le chiffrement par flot (nommé encore chiffrement à la volée). Dans ce contexte, les registres filtrés (LFSR) sont des systèmes performants pour générer des séquences aléatoires de très grande période à partir de clés courtes (128 bits) ; ils sont de ce fait très utilisés dans les communications militaires. L'utilisation des techniques algébriques et en particulier de l'algorithme $F_5/2$ semble très concluante puisque des exemples réalistes de longueur 128 bits peuvent être résolus dès à présent (ces exemples sont largement hors de portée des autres techniques). Un contrat (fin 2002) avec le Celar (DGA, Rennes) vise à étudier précisément les tailles de registres attaquables par les bases de Gröbner en fonction des divers paramètres du registre (nature de la fonction filtrante, position des pattes, taille du registre, etc.). Ce travail fait l'objet d'une collaboration avec D. Augot et A. Canteaut (attaque par corrélation) du projet CODES. Une application directe de l'étude théorique de complexité sur les systèmes aléatoires permet d'énoncer immédiatement : pour un registre filtré de longueur n, étant donnés $n \log n$ échantillons, on peut retrouver en temps sous-exponentiel l'état initial (secret) du registre. Un travail en commun avec la société Thalès devrait également débuter en 2003 sur ce sujet.

HFE (Hidden Fields Equations) est un cryptosystème à clé publique n'utilisant pas la théorie des nombres (comme RSA), mais des opérations sur les polynômes à coefficients dans un corps fini. Ce cryptosystème a été proposé par Jacques Patarin à Eurocrypt 96 en améliorant les idées de Matsumoto et Imai. Ce cryptosystème semble très prometteur car il peut servir à générer des signatures très courtes : 128, 100 ou même 80 bits. L'idée de HFE est de prendre un polynôme univarié secret à coefficients dans $\mathrm{GF}(2^n)$ puis d'exprimer ce polynôme sur $\mathrm{GF}(2)$. On obtient ainsi un système algébrique en n variables (la clé publique).

Retrouver le message original connaissant la clé secrète est « facile » puisque cela revient à résoudre un problème univarié. En revanche, avec seulement la clé publique, cela devient un problème très difficile puisqu'il s'agit de résoudre un système algébrique (complexité exponentielle dans le cas des systèmes aléatoires). Ce cryptosystème a été étudié par plusieurs cryptographes (Shamir, Courtois, etc.).

Les avancés algorithmiques (F_5 [25]) permettent maintenant de :

- casser assez facilement le *Challenge* proposé par J. Patarin (il s'agit d'un exemple réaliste en taille 80 bits). Cette taille de système (80 équations denses de degré 2) était totalement encore hors de portée il y a quelques mois ;
- mener une étude de complexité expérimentale pour les systèmes HFE. Par exemple, si le degré du polynôme secret est inférieur à 129, la complexité du calcul d'une base de Gröbner est seulement $O(n^8)$.

L'importance cryptographique croissante prise par les jacobiennes de courbes algébriques sur les corps finis [13], au moins dans une logique prospective, rend de plus en plus fréquent d'avoir affaire avec des idéaux dans les corps de fonctions des courbes ou de leurs jacobiennes. Calculer dans de tels corps de fonctions peut se faire en manipulant des idéaux d'algèbres de polynômes. Des travaux en préparation améliorent les meilleurs algorithmes connus dans le cas de certaines familles particulières de courbes.

En outre, les avancées sur le calcul du nombre de points de ces courbes en grande caractéristique, indispensable si l'on souhaite s'assurer que le système ne présente pas une faille grossière, semblent actuellement passer par les généralisations de l'algorithme de Schoof (utiliser l'action de l'endomorphisme sur le module de Tate $T_l(J)$), ce qui impose de savoir calculer efficacement modulo les idéaux de l-torsion, qui sont des idéaux d'une algèbre de polynômes en plusieurs variables. Ce problème d'efficacité, actuellement non résolu, est très limitant.

Un autre problème traité par les techniques de SPACES est le problème du décodage des codes cycliques au-delà de la distance minimale [43]. *A priori* la correction des codes ne relève pas de la cryptographie, mais les techniques algébriques sont identiques : en effet, le problème du décodage des codes cycliques peut se récrire en un système algébrique sur un corps fini de caractéristique 2 (comme AES) dont les solutions sont étroitement liées à l'erreur qui s'est produite. Les travaux précédents ont montré que le calcul d'une base de Gröbner de ce système algébrique permet de décoder jusqu'à la distance minimale du code. Le calcul de base de Gröbner peut se faire, soit en prétraitement (décodage formel), les paramètres étant considérés comme des variables, soit pour chaque mot à décoder (décodage en ligne), en calculant pour chaque mot les paramètres et en les substituant dans le système. Dans le cas du décodage formel, il a été montré qu'il est possible d'obtenir, à partir de la base de Gröbner formelle, des formules de décodage pour les coefficients du polynôme localisateur. Malheureusement, il devient rapidement impossible de calculer cette base de Gröbner formelle, même pour des codes de petite longueur.

Motivés par le problème du décodage des codes à résidus quadratiques (codes RQ), pour lesquels il n'existe actuellement aucun algorithme général de décodage, nous améliorons les résultats sur plusieurs points. D'abord, nous introduisons des systèmes modifiés, sans équation de degré élevé, pour lesquels le calcul de la base de Gröbner est plus facile. Ceci permet de calculer la base de Gröbner formelle pour des codes de plus grande longueur.

Nous montrons sur l'exemple du code RQ de type [41,21,9] que les formules deviennent rapidement de grande taille, et sont donc inutilisables pour le décodage. Cela indique que les efforts développés pour le décodage algébrique des codes cycliques par formules ont peu de chance d'aboutir. L'autre approche (décodage en ligne) est plus efficace d'un point de vue informatique. En utilisant des méthodes générales de compilation pour des systèmes avec paramètres, nous améliorons l'efficacité des calculs. Nous donnons de nombreux exemples (codes BCH de longueur 75, 511, codes RQ de longueur 41, 73, 89, 113, et un code de longueur 75 qui n'appartient à aucune classe connue de codes cycliques).

Cette technique de décodage des codes cycliques avec des bases de Gröbner s'applique à n'importe quel code, est entièrement automatique, et permet de décoder au-delà de la capacité de correction du code.

4.6. Mécanique céleste

Mots clés : *mécanique céleste*, *configuration centrale*.

Participants : Jean-Charles Faugère, Daniel Lazard.

Glossaire

configuration centrale Configuration de n corps gravitationnels, telle que toutes les accélérations sont dirigées vers le centre de masse et proportionnelles à la distance à ce centre. Une configuration centrale plane reste semblable à elle-même au cours du temps.

L'étude du mouvement de plusieurs masses ponctuelles soumises à l'attraction newtonienne est un problème très ancien [61]. Devant la difficulté du problème (dès que l'on a plus de trois corps, le système différentiel décrivant le mouvement est non intégrable analytiquement) il apparaît naturel de se restreindre aux solutions « simples » de ces équations différentielles. Ainsi la recherche des configurations centrales permet de se ramener à l'étude d'un problème algébrique. Toutefois l'étude de ces systèmes algébriques est particulièrement difficile en particulier à cause du degré élevé des équations de départ. Les membres de l'équipe collaborent sur ce sujet, depuis plusieurs années, avec des chercheurs du Bureau des Longitudes de l'Observatoire de Paris (autour d'Alain Albouy). Cette collaboration a donné lieu à une thèse co-encadrée (Ilias Kotsireas).

Plus récemment, les progrès algorithmiques du projet ont permis de faire progresser sensiblement cette question : Nous avons complètement déterminé, en fonction des différentes masses, le nombre des configurations centrales de 4 corps ayant un axe de symétrie, de masses $m_1, m_2, m_3 = m_4 = 1$. Il y a une, trois ou cinq configurations possibles, suivant la position du point (m_1, m_2) par rapport à une courbe délimitant des régions du plan où la nature et le nombre des solutions changent. Cette courbe est définie par un polynôme à deux

variables de degré 424 de plus de 50 000 termes, et qui occupe 10 Mo en mémoire. Le tracé de cette courbe est un des problèmes qui nous ont conduits à mettre au point le logiciel TCI (section 5.7). C'est également la résolution de cet exemple qui nous a amenés à mettre au point un algorithme général de résolution de systèmes dépendant de paramètres (section 6.2).

Cette application est au centre de l'ACI « jeunes chercheurs » dont l'équipe est bénéficiaire (section 8.1). Elle a fait l'objet de plusieurs conférences invitées en 2002 [32][31].

Dans le cas du potentiel logarithmique (problème des tourbillons en hydrodynamique) une mise en équation utilisant les symétries et l'application de l'algorithme F_4 ont permis de réaliser une classification complète jusqu'à 7 corps de masses égales et de trouver de nouvelles familles infinies de solutions particulières (généralisant ainsi les solutions alignées obtenues à partir des polynômes de Hermite) [28].

5. Logiciels

5.1. MPFR/MPFI

MPFR *et* MPFI *sont des logiciels fondamentaux de l'équipe.*
Mots clés : *arithmétique d'intervalle*, *précision arbitraire*, *arrondi exact*.

Participants : Guillaume Hanrot, Vincent Lefèvre, Yves Pétermann, Jean-Luc Rémy [avant-projet Adage], Fabrice Rouillier, Paul Zimmermann.

Glossaire

GMP GNU Multi-Precision library
MPFR Multi-Precision Floating point Reliable arithmetic
MPFI Multi-Precision Floating point Interval arithmetic

Le développement de MPFR s'est poursuivi sur les points suivants : meilleure spécification (par exemple, concernant les cas particuliers comme le traitement des zéros signés, ou encore la sémantique de l'*underflow*), meilleur support des dépassements de capacité (*overflow, underflow*), écriture de nouvelles fonctions, réécriture de certaines fonctions (pour optimisation et/ou parce que les spécifications ont évolué), optimisations diverses... La portabilité de MPFR a été accrue pour sa diffusion sur le CD-ROM logiciels libres de l'INRIA en avril 2002, et la diffusion de la version 2.0.1 le 15 avril, version distribuée avec GNU MP 4.1, sorti en mai.

En août et en octobre 2002 Y.-F.S. Pétermann, chercheur de l'Université de Genève invité par l'UHP (août) et par l'INRIA (octobre), a travaillé avec J.-L. Rémy (avant-projet Adage) sur l'implantation de la fonction zêta de Riemann en MPFR, accompagnée d'une analyse d'erreur rigoureuse. Ils ont achevé le travail concernant le cas réel, dont ils avaient déjà réalisé une bonne partie en octobre 2001, et se sont attaqués au cas général complexe. Voir le paragraphe 6.8 ci-dessous concernant les résultats nouveaux en arithmétique.

MPFI est une bibliothèque d'arithmétique d'intervalles, écrite en C (environ 1000 lignes), basée sur MPFR. Initialement, MPFI a été développée pour les besoins d'un nouvel algorithme hybride pour l'isolation de zéros réels de polynômes. MPFI contient le même nombre d'opérations et de fonctions que MPFR [58], le code est disponible et documenté.

5.2. Gb/FGb

Les logiciels GB/FGB *sont des logiciels fondamentaux de l'équipe.*
Mots clés : *Gröbner*, *interface*, *logiciel*.

Participant : Jean-Charles Faugère.

GB est un logiciel pour le calcul de bases de Gröbner développé en C++ (100 000 lignes). Ce logiciel est très stable et fait l'objet d'une diffusion par le Web. Il est interfacé avec plusieurs logiciels de calcul formel (Maple, MuPAD et Axiom) et avec les logiciels REALSOLVING et RS. Le logiciel et les interfaces sont disponibles sur le Web http://calfor.lip6.fr/~jcf. FGB est un nouveau logiciel expérimental (91 000 lignes de C) dans lequel

sont implantés les nouveaux algorithmes F_4, F_5 ([25]) et F_7. Il est interfacé avec Maple et utilisable via le Web. Toutefois son aspect expérimental n'en permet pas encore une diffusion générale.

5.3. RS/RealSolving

Mots clés : *zéro réel*, *système de dimension zéro*, *polynôme en une variable*.

Participant : Fabrice Rouillier.

RS est un logiciel dédié à l'étude des zéros réels des systèmes algébriques. Il est entièrement développé en C (100 000 lignes environ) et succède à REALSOLVING développé lors des projets européens PoSSo et FRISCO. RS contient principalement des fonctions pour le comptage et l'isolation des zéros réels de systèmes algébriques admettant un nombre fini de racines complexes. Les interfaces utilisateur de RS sont entièrement compatibles avec celles de GB/FGB (ASCII, MuPAD, Maple). RS est utilisé dans le projet depuis plusieurs mois et une version de développement a été mise à disposition de diverses équipes. La version actuelle est suffisamment stable pour être diffusée.

5.4. Epsilon

Mots clés : *ensemble caractéristique*, *ensemble triangulaire*, *décomposition de système polynomial*, *décomposition primaire*, *factorisation algébrique*, *preuve de théorème*, *dessin automatique*, *résultant*, *forme implicite*.

Participant : Dongming Wang.

Glossaire

mot-clé bibliothèque Epsilon

La bibliothèque Epsilon est une collection de fonctions pour l'élimination et le calcul avec des systèmes de polynômes à plusieurs variables et leurs applications géométriques.

La première version de la bibliothèque Epsilon en Maple 8 est prête à distribuer. Cette bibliothèque contient huit modules : `epsilon`, `charsets`, `dcharsets`, `trisys`, `dtrisys`, `sisys`, `geother`, et `miscel`, dont le module `charsets` est une implantation complète de la méthode des ensembles caractéristiques, déjà distribués mondialement depuis 1991 pour des utilisations académiques (voir http://www.mapleapps.com/maplelinks/share/charsets.html et http://calfor.lip6.fr/~wang/charsets.html). L'environnement `geother` (htmladdnormallinkurlhttp ://calfor.lip6.fr/ wang/GEOTHER/http ://calfor.lip6.fr/ wang/GEOTHER/) conçu et implanté pour la manipulation et la preuve de théorèmes géométriques est déjà connu dans la communauté de la déduction automatique en géométrie.

Conjointement avec les modules `trisys`, `dtrisys` et `sisys` qui implantent les deux méthodes d'ensembles triangulaires proposées par D. Wang, et le module `miscel` constitué de fonctions diverses, Epsilon permet de

- trianguliariser des systèmes quelconques de polynômes à plusieurs variables,
- décomposer de tels systèmes en systèmes triangulaires de différentes sortes (réguliers, simples, irréductibles, ou munis de propriétés de projection), et donc les résoudre,
- décomposer des variétés algébriques en composantes irréductibles ou équidimensionnelles,
- décomposer des idéaux polynomiaux en composantes primaires,
- factoriser des polynômes sur des extensions algébriques successives de corps,
- prouver des théorèmes géométriques et dessiner des diagrammes automatiquement,
- traduire des spécifications géométriques en expressions algébriques et en langue naturelle (anglais et chinois) automatiquement,
- calculer des résultants (de Bézout et de Macaulay) et sous-résultants,
- mettre sous forme implicite des surfaces paramétriques rationnelles,
- accomplir une partie des tâches ci-dessus pour des systèmes de polynômes différentiels ordinaires.

Epsilon comprend plus de 70 fonctions avec des aides en ligne et a environ 20 000 lignes de code Maple et Java. Une description d'Epsilon a été publiée dans les actes du congrès ICMS 2002 [41] ainsi qu'une présentation de l'environnement `geother` à la conférence ADG 2002 [42]. Un livre qui présente la bibliothèque et ses applications sera publié en 2003.

5.5. UDX

Participants : Fabrice Rouillier [correspondant], Jean-Charles Faugère.

Glossaire

UDX Universal Data eXchange
XDR eXternal Data Representation

UDX est un logiciel pour l'échange de données binaires. Il a été implanté à l'origine pour montrer la puissance d'un nouveau protocole, objet d'un dépôt de brevet par l'INRIA et l'UPMC (numéro de dépôt 99 08172, 1999). Le code résultant, écrit en C ANSI, est très portable et d'une efficacité certaine, même lorsque le protocole breveté n'est pas utilisé. UDX est composé de cinq modules indépendants :

- base : système optimisé de tampons et de synchronisation des entrées et sorties ;
- supports : opérations de lecture/écriture sur des supports variés (*sockets*, fichiers, mémoire partagée, etc.) ;
- protocoles : protocoles d'échanges variés (protocole breveté, XDR, etc.) ;
- échanges de types composites : flottants simple et double précision, entiers, rationnels, flottants multiprécision ;
- interfaces : interfaces utilisateur de haut niveau faisant appel aux quatre autres modules.

UDX est utilisé dans le projet pour interfacer les différents composants logiciels entre eux ou avec des outils de calcul extérieurs (MuPAD par exemple).

UDX est utilisé pour certaines interfaces développées dans le projet (GB/RS/MuPAD) mais également dans des complexes logiciels extérieurs au projet (SYNAPS et ROXANE, en collaboration avec le projet GALAAD [24], interface MuPAD/Scilab diffusée avec MuPAD 2.5.1).

5.6. Interfaces

Participants : Jean-Charles Faugère, Fabrice Rouillier.

Afin de rendre aisée l'utilisation des divers logiciels développés dans le projet, des conventions d'échanges de fichiers ASCII et binaires ont déjà été mises au point et permettent une utilisation souple des serveurs de calcul GB/FGB/RS.

Pour rendre transparente leur utilisation à partir de logiciels généralistes tels que Maple ou MuPAD, nous avons consolidé et homogénéisé les prototypes d'interfaces existantes et nous proposons actuellement une distribution commune pour GB/FGB/RS incluant des serveurs de calculs, des interfaces Maple/MuPAD, un mode d'emploi illustré par des exemples concrets et un processus d'installation simple.

5.7. Tracé de courbes implicites (TCI)

Mots clés : *courbe implicite*, *tracé certifié*.

Participants : Jean-Charles Faugère, Daniel Lazard, Fabrice Rouillier.

Dès qu'il s'agit d'applications réelles, les polynômes issus de procédés d'élimination (bases de Gröbner, ensembles triangulaires) sont très souvent trop gros pour pouvoir être étudiés par les outils généraux de calcul formel. Dans le cas de polynômes en 2 variables, un tracé certifié suffit dans beaucoup de cas pour *résoudre* le problème étudié (c'est le cas notamment pour certaines applications en mécanique céleste). Ce type de tracé est maintenant possible grâce aux différents outils développés dans le projet.

Deux composants sont actuellement en cours de développement : la routine de calcul (prenant en entrée la fonction polynomiale et retournant un ensemble de points) est stable et peut être utilisée comme boîte noire seule ou depuis Maple, la routine de tracé est, elle, en cours d'étude.

5.8. Simulateur de planification de trajectoires (SPT)

Participants : Fabrice Rouillier [correspondant], Luc Rolland.

Le projet SPT a pour but d'homogénéiser et d'exporter l'ensemble des outils développés dans le cadre de nos applications en robotique parallèle. Les composants logiciels de SPT sont essentiellement des implantations en C suivant les standards du logiciel RS, des algorithmes écrits en MuPAD ou Maple utilisant les prototypes d'interfaces pour GB et RS. L'encapsulation de tous ces composants dans une distribution unique est en cours de test.

Quelques prototypes de composants pour la résolution temps réel de certains types de systèmes utilisent désormais la bibliothèque ALIAS développée dans le projet COPRIN.

5.9. Test des fonctions élémentaires

Participant : Vincent Lefèvre.

Les tests des fonctions élémentaires pour trouver des pires cas sont en cours de réécriture, en s'orientant vers les points suivants :

- concevoir et utiliser de nouveaux algorithmes, comme celui basé sur LLL ou bien des variantes plus rapides de l'algorithme de minoration de la distance d'un segment de droite aux points à coordonnées entières, utilisé jusqu'à présent ;
- avoir du code portable tout en étant aussi rapide, grâce à la couche bas niveau de GMP ;
- avoir des programmes fiables : le but ultime est de prouver le maximum, et lorsque ce n'est pas possible (à cause de l'utilisation de logiciels externes, comme Maple, pour des tâches complexes), effectuer des calculs redondants à l'aide de différents logiciels ;
- pouvoir pleinement réutiliser le code (écrit de manière beaucoup plus modulaire) dans diverses précisions à tester, avec de nouveaux algorithmes ou pour d'autres problèmes que la recherche des pires cas.

6. Résultats nouveaux

6.1. Bases de Gröbner

Participants : Magali Bardet, Jean-Charles Faugère.

L'algorithme F_5 a été présenté [25] et permet de gagner au moins un ordre de grandeur dans les calculs. Cet algorithme a fait l'objet de plusieurs conférences invitées [27][26] et sa version spécialisée au corps fini $\mathbb{F}_2$ a permis de casser [29] le Challenge proposé par Patarin en 1996. L'application au problème HFE de cet algorithme a fait l'objet de plusieurs présentations (10) en France en 2002.

Sur le plan théorique, la simplicité de l'algorithme F_5 a permis de donner la complexité du calcul d'une base de Gröbner d'un système aléatoire dans le cas des corps finis ou des systèmes surdéterminés (travail en commun avec B. Salvy).

6.2. Systèmes dépendant de paramètres

Participants : Jean-Charles Faugère, Daniel Lazard, Fabrice Rouillier, Mohab Safey el Din.

La plupart des applications récentes que nous avons traitées (mécanique céleste, robots cuspidaux, statistiques, etc.) nécessitent l'étude de systèmes semi-algébriques dépendant de paramètres. Bien que nous ayons traité ces

sujets de manière indépendante, il se dégage actuellement de ces expériences quelques algorithmes généraux pour la résolution de ce type de systèmes.

La philosophie générale consiste à étudier les solutions *génériques* en se plaçant en dehors de sous-variétés algébriques (que nous appelons désormais variétés discriminantes) de dimension plus faible que la variété semi-algébrique considérée. Le traitement des variétés ainsi exclues peut se faire séparément pour obtenir une réponse complète au problème, ou est simplement négligé si l'on ne s'intéresse qu'aux solutions *génériques*, ce qui est le cas dans certaines applications.

Plusieurs stratégies de calcul ont été élaborées. Une décomposition *a priori* en composantes équidimensionnelles définies comme zéros d'idéaux radicaux permet de simplifier le calcul et le traitement des variétés discriminantes. Ce calcul préliminaire s'avérant toutefois parfois très coûteux, nous travaillons également sur des solutions adaptatives ne décomposant le problème (à la volée) qu'en cas de nécessité.

Une fois les sous-variétés discriminantes calculées, plusieurs possibilités s'offrent à nous. Dans le cas de problèmes de petite dimension, on peut se passer assez facilement de l'étude des variétés discriminantes en déduisant leur topologie de l'étude des solutions génériques du système (c'est ce qui a été fait pour le problème de mécanique céleste par exemple). Dans le cas des problèmes de plus grande dimension (par exemple l'application sur les robots cuspidaux), on peut calculer une décomposition cellulaire *partielle* de l'espace des paramètres selon une propriété fixée (par exemple nombre de solutions réelles constant).

Il est apparu, tout récemment, que ces techniques s'appliquent également à la résolution des systèmes surdéterminés dépendant de paramètres approchés (valeurs mesurées). Elles ont permis, dans un problème issu de la statistique, de montrer l'unicité de la solution pour toute valeur admissible des paramètres, d'une manière qui permette la mise au point d'un algorithme de résolution de ces systèmes surdéterminés (il n'en existe pas à notre connaissance) qui soit stable dans les régions du domaine des paramètres où c'est possible, et qui détermine approximativement ces régions.

6.3. Zéros réels des systèmes de dimension positive

Participants : Philippe Aubry, Fabrice Rouillier, Mohab Safey El Din.

Bien que les problèmes généraux en dimension positive puissent être traités comme des problèmes paramétrés après changement de coordonnées, nous travaillons sur une famille de méthodes permettant de calculer un point par composante connexe sur une variété algébrique réelle, proposant ainsi une alternative à la décomposition cylindrique algébrique.

Nous travaillons essentiellement sur des méthodes dites de *points critiques* : on calcule les points critiques d'une fonction bien choisie de manière à se ramener à l'étude d'un système zéro-dimensionnel.

Dans le cas d'une variété compacte sans singularité, n'importe quelle fonction polynomiale convient. On préfère alors choisir une fonction linéaire (modélisant une projection sur un axe), le coût du calcul des points critiques étant moindre dans ce cas précis. La difficulté consiste à étendre cette méthode aux cas plus généraux de variétés non compactes et/ou contenant des points singuliers.

Dans un premier temps, nous nous sommes concentrés sur la recherche d'une méthode permettant de gérer efficacement la présence éventuelle de singularités en choisissant une fonction dont la restriction à la variété étudiée est propre, afin de se soustraire provisoirement au problème de la compacité. En effet, dans ce cas, la fonction atteint ses extrema sur chaque composante connexe de la variété.

Pour cela, nous considérons dans [12] la fonction *distance* à un point générique. Le principe consiste toujours à calculer les points critiques de la fonction distance à un point, mais lorsque des singularités sont présentes dans la variété de départ, celles-ci sont étudiées comme une variété algébrique à part entière, de dimension strictement inférieure à celle de la variété initiale. Cet algorithme effectue donc une induction sur la dimension des variétés intermédiaires apparaissant en cours de calcul. Ce principe permet de gérer en pratique efficacement la présence éventuelle de singularités.

Dans un deuxième temps, en collaboration avec E. Schost (Laboratoire GAGE, École Polytechnique), nous avons considéré dans [48] des fonctions de projection pour calculer au moins un point par composante connexe sur une variété. Dans ce cas la difficulté essentielle est que ces fonctions n'atteignent pas nécessairement leurs

extrema. Comme précédemment, l'algorithme obtenu effectue une induction sur la dimension de variétés intermédiaires dont on calcule :

- le lieu critique de la projection sur un sous-espace affine générique de même dimension,
- le lieu singulier de la variété,
- l'intersection de la variété avec l'image réciproque par la projection considérée d'au moins un point du sous-espace affine dans chaque composante connexe du lieu de non propreté de cette projection.

À chaque étape, on obtient des variétés algébriques de dimension strictement inférieure à celle de la variété courante. L'algorithme est alors appliqué récursivement sur ces nouvelles variétés.

Les deux algorithmes mentionnés ci-dessus ont été implantés dans un paquetage Maple utilisant GB, RS et l'interface GB/Maple. Ce paquetage a été mis à la disposition des membres du projet. Pour l'instant, il n'est doté que des fonctionnalités de décomposition en composantes équidimensionnelles et radicales d'un système d'équations polynomiales et du calcul d'un point par composante connexe de la variété algébrique réelle définie par le système.

Sur un jeu d'exemples, de référence pour la communauté, le second algorithme est plus efficace en pratique et retourne moins de points que le premier, ceci étant probablement dû au fait que le premier algorithme calcule des points critiques d'une fonction quadratique, alors que le second calcule des points critiques de fonctions linéaires.

Par ailleurs, il est intéressant de noter la convergence entre ces développements et ceux relatifs aux systèmes paramétrés, les objets géométriques considérés dans le dernier algorithme étant identiques à ceux étudiés dans le cas des systèmes paramétrés.

6.4. Polynômes univariés

Participants : Guillaume Hanrot, Daniel Lazard, Fabrice Rouillier, Paul Zimmermann.

La résolution en terme de radicaux d'une équation de degré cinq est connue comme impossible depuis les travaux d'Abel et de Galois. Ce dernier avait en outre montré que ce problème est décidable. Mais le test de résolubilité par radicaux, et surtout le calcul de la solution, quand elle existe, était hors de portée des moyens de l'époque. Ce problème avait déjà été abordé par l'équipe l'an dernier [4]. À l'occasion du bicentenaire d'Abel, nous avons publié un algorithme calculant effectivement cette résolution par radicaux, quand elle est possible. Nous montrons en outre que l'extension de corps dans laquelle la solution est décrite est la plus petite possible. Toutefois, la complexité du résultat de cet algorithme fait que son seul intérêt est d'être un excellent test des logiciels de manipulation des extensions algébriques [19].

L'article décrivant la version « mémoire constante » de l'algorithme d'Uspensky pour l'isolation des racines réelles d'un polynôme univarié va enfin paraître [20]. Guillaume Hanrot a fait une implantation en GNU MP de cet algorithme (http://www.loria.fr/~hanrot/progs/usp.c).

6.5. Raisonnement géométrique

Participant : Dongming Wang.

Mots clés : *dessin automatique*, *diagramme géométrique*, *surface implicite*.

Le dessin automatique des diagrammes est désirable pendant la manipulation et la preuve de théorèmes géométriques ainsi que pour la conception géométrique par ordinateur, mais c'est un problème difficile en général. Nous avons proposé et implanté une approche qui permet de dessiner les diagrammes automatiquement et effectivement pour des théorèmes en géométrie plane euclidienne [22]. Cette approche utilise Maple pour le calcul algébrique et Java pour le dessin graphique. Les diagrammes dessinés peuvent être modifiés et animés avec la souris.

Les méthodes bien connues pour mettre sous forme implicite des courbes et surfaces paramétriques rationnelles sont fondées sur le calcul de bases de Gröbner et de résultants, ou sur la méthode de Sederberg.

La méthode utilisant les bases de Gröbner est quelquefois inefficace (par exemple en Maple), tandis que des autres méthodes plus efficaces ont été validées en l'absence de points de base seulement. Nous avons proposé une méthode nouvelle et simple [21] qui marche bien en général et beaucoup mieux en présence des points de base. Notre méthode utilise le principe des coefficients indéterminés pour transformer la recherche d'une forme implicite en résolution de systèmes linéaires aléatoires et partiellement triangulaires.

6.6. Géométrie informatique

Participant : Daniel Lazard.

Mots clés : *géométrie*, *quadrique*, *intersection*.

La collaboration avec le projet ISA (Sylvain Lazard et Sylvain Petitjean) entamée en 2001 s'est poursuivie.

Pour l'essentiel, elle a porté sur la paramétrisation des surfaces quadriques et de leurs intersections, en vue du calcul des configurations d'arcs de courbes définis par une famille de quadriques. Ces problèmes se rencontrent dans la représentation des solides par les surfaces qui les délimitent.

Nous avons mis au point un algorithme robuste pour paramétriser ces quadriques et leurs intersections [17], et nous avons montré que, dans tous les cas, il est optimal du point de vue du nombre nécessaire d'extractions de racines carrées. Ce résultat d'optimalité a fait l'objet d'un exposé invité à la conférence de clôture des ARC CoSTIC et Visi3D [66].

6.7. Robots cuspidaux

Participants : Fabrice Rouillier, Solen Corvez.

Dans [23] (une version étendue [45] est soumise pour publication), nous présentons une classification d'une famille de robots série à 3 degrés de liberté selon leur comportement cuspidal. Il a été montré, dans des travaux précédents, que tester le caractère cuspidal d'un robot de ce type revient exactement à décider si un polynôme de degré quatre, dépendant des paramètres de conception, admet ou non une racine réelle triple.

Nous avons calculé une partition de l'espace des paramètres telle que au dessus de chaque composante connexe de cette partition, le nombre de points triples du polynôme soit constant, produisant ainsi une description complète des configurations cuspidales.

6.8. Arithmétique

Participants : Guillaume Hanrot, Vincent Lefèvre, Yves Pétermann, Jean-Luc Rémy [avant-projet Adage], Paul Zimmermann.

Dans le cadre de la constitution de la bibliothèque MPFR, Y.-F.S. Pétermann et J.-L. Rémy ont terminé l'analyse d'erreur pour le calcul de $\zeta(s)$, la fonction zêta de Riemann, dans le cas où l'argument est un nombre réel $s \geq 1/2$, et ont commencé à traiter le cas général où s est un nombre complexe. L'algorithme exploité est dû à H. Cohen et M. Olivier [53], et a déjà été utilisé pour implanter le calcul de $\zeta(s)$ dans le système PARI, mais aucune analyse d'erreur complète n'avait été faite jusqu'ici. Dans le cas réel, si $P \geq 11$ est la précision voulue (au sens classique, sans qu'on se préoccupe pour l'instant de réaliser un arrondi exact), pour laquelle le nombre calculé $\zeta(s)^*$ satisfait $|\zeta(s) - \zeta(s)^*| < 2^{-P}|\zeta(s)|$, ce travail [51] établit qu'une précision interne de calcul $D = P + \lceil C \log P + c \rceil$ suffit, où les constantes C et c sont données explicitement (pour autant bien sûr que l'argument s puisse s'écrire avec au plus D chiffres en base 2). Concrètement, si par exemple $P = 1000$, $D = 1018$ suffit. On ne peut malheureusement pas s'attendre à obtenir un résultat aussi miraculeusement bon dans le cas général complexe. Par ailleurs il semble que l'utilisation d'une vieille conjecture de R.P. Brent concernant le calcul des nombres de Bernoulli, très plausible mais jusqu'ici non démontrée, soit nécessaire pour espérer déduire la suffisance d'une précision interne de calcul raisonnable, c'est-à-dire pas trop gigantesque. L'état d'avancement des travaux est décrit en deuxième partie du rapport de recherche [50], qui reproduit en première partie le travail [51] soumis à publication.

G. Hanrot et P. Zimmermann ont étudié la multiplication des séries formelles ; par rapport à la multiplication de polynômes, celle-ci a la particularité que seule une partie du résultat est significative. L'opération

correspondante est donc le produit court, ou produit tronqué, de polynômes. Mulders a proposé il y a peu une méthode permettant d'effectuer ce produit tronqué plus vite qu'un produit complet, mais sa méthode reposait sur le choix d'un point de coupure optimal qui n'était déterminé qu'heuristiquement. Dans le travail [46], le point de coupure est déterminé de façon précise, et une variante de l'idée de Mulders plus efficace asymptotiquement est présentée.

En outre, une étude systématique du produit médian menée avec Michel Quercia a conduit à des améliorations des algorithmes pour l'arithmétique des polynômes (division avec et sans reste, racine carrée, carré court). Des équivalents pour les flottants multiprécision sont à l'étude.

En collaboration avec Richard Brent (Oxford) et Samuli Larvala (Helsinki), P. Zimmermann a publié les résultats obtenus pour la recherche de trinômes primitifs de degré 3021377 sur $\mathrm{GF}(2)$ [15]. À l'aide des ressources mises à disposition par le CINES, la recherche d'un trinôme primitif de degré 6972593 a été poursuivie, et couronnée de succès, puisqu'un premier trinôme primitif a été découvert fin août :

$$x^{6972593} + x^{3037958} + 1.$$

Un article pour la gazette du CINES est en préparation.

Un article de Yann Bugeaud, Guillaume Hanrot et Maurice Mignotte [16] sur l'équation diophantienne $(x^n - 1)/(x - 1) = y^q$, qui résout complètement cette équation pour $q \leq 17$, vient de paraître.

Yves Bertot, Nicolas Magaud (tous deux du projet Lemme à INRIA Sophia Antipolis) et P. Zimmermann ont prouvé [44] à l'aide de l'assistant de preuve Coq un algorithme de calcul de racine carrée d'entier en précision arbitraire [76]. L'originalité de ce travail est que la preuve ne considère pas seulement la description abstraite de l'algorithme, mais également son implantation, en l'occurrence dans la bibliothèque GNU MP. Cette preuve paraîtra dans un numéro spécial de la revue *Journal of Automated Reasoning* en l'honneur de N. G. de Bruijn.

V. Lefèvre a continué son travail sur la multiplication par une constante, mais cette fois-ci pour obtenir des résultats de complexité : majorants de la taille des décalages nécessaire dans un algorithme optimal et minorants (en moyenne et au pire) de la longueur d'un code de multiplication par une constante [47].

On suppose que la taille des décalages utilisés pour multiplier un entier par une constante n sur m bits est majorée par une fonction $S(m)$. On impose que $S(m) \geq m$ de façon à pouvoir multiplier par $2^m - 1$ en une seule opération élémentaire (décalage de m bits couplé à une soustraction). Pour les heuristiques utilisées en pratique, on peut poser $S(m) = m$, *i.e.* tous les décalages intervenant dans la multiplication par une constante sur m bits ont une taille inférieure ou égale à m. Mais que peut-on dire si on s'intéresse au code optimal pour multiplier par une constante sur m bits ? Les deux résultats suivants ont été obtenus :

- Parmi différents codes optimaux (i.e. contenant un nombre minimal d'opérations élémentaires), certains peuvent nécessiter des décalages plus importants que d'autres, et pour certains codes optimaux, la borne $S(m) = m$ peut être dépassée. En effet, pour multiplier par le nombre de $6h + 1$ bits $n = (1 + 2^h)(1 + 2^{2h})(1 + 2^{4h}) - 2^{7h}$, 4 opérations élémentaires sont nécessaires et il existe un code optimal utilisant un décalage de $7h$ bits, alors qu'il existe un autre code optimal avec des décalages ne dépassant pas les $6h + 1$ bits. Le choix du code optimal peut donc être déterminant pour pouvoir diminuer la valeur de $S(m)$.
- Pour n'importe quel code optimal, la taille des décalages est inférieure ou égale à $S(m) = 2^{\lfloor m/2 \rfloor - 2}(m + 1)$ pour $m \geq 4$. Malheureusement, cette borne n'est d'aucune utilité en pratique, mais c'est déjà un premier pas.

Pour obtenir des minorants de la taille du code associé à une constante n (nombre d'opérations élémentaires), on doit limiter $S(m)$ à des valeurs raisonnables. On suppose qu'il existe $\alpha \geq 1$ tel que pour toute constante n de m bits, il existe un code optimal de multiplication par n tel que les décalages sont majorés par $S(m) = \alpha m$; cette propriété est très probablement vérifiée. Avec cette contrainte, on obtient un minorant de la forme

$$\frac{m}{3\log_2 m}(1 + o(1))$$

pour le nombre moyen d'opérations élémentaires (pour la complexité au pire, on obtient la même borne asymptotique).

6.9. Arithmétique d'intervalles en précision arbitraire

Participants : Nathalie Revol [projet Arénaire, INRIA Rhône-Alpes], Fabrice Rouillier.

Mots clés : *arithmétique par intervalles*, *précision arbitraire*, *calcul certifié*, *adaptation automatique de la précision*.

La bibliothèque MPFI (voir §5.1) d'arithmétique par intervalles en précision arbitraire et ses justifications théoriques ont été présentées à la conférence Validated Computing [34].

Le travail sur MPFI s'est poursuivi : les fonctions trigonométriques et toutes les fonctions disponibles dans MPFR [58] ont été implantées de façon à rendre le résultat le plus précis possible. Un test comparatif a été mené sur l'élimination de Gauss appliquée à des H-matrices de dimensions 300×300 [35] : comparé avec `fi_lib` [69], le surcoût lié à l'utilisation d'une arithmétique flottante multiprécision est de 5 pour des intervalles en double précision et de seulement 12 pour 10 fois plus de précision.

6.10. Test de la qualité des fonctions mathématiques sur ordinateur

Participants : Vincent Lefèvre, Nathalie Revol [projet Arénaire, INRIA Rhône-Alpes], Paul Zimmermann.

Mots clés : *bibliothèque mathématique*, *fonction élémentaire*, *qualité*, *arrondi*, *pire cas*, *LLL*.

Le point de départ de ce travail est la collaboration avec HP-Intel sur le processeur IA-64 : HP-Intel nous a fourni deux IA-64 pour que, entre autres, nous testions la qualité des fonctions mathématiques développées pour ce processeur. Dans un premier temps, nous avons testé la précision de ces fonctions : sur des échantillons de nombres aléatoires (typiquement de 1 à 10 millions), nous avons mesuré les erreurs moyenne et maximale entre la valeur exacte et la valeur calculée, pour les fonctions mathématiques usuelles (log, exp, sin, atan, cosh, asinh, etc.), pour les quatre modes d'arrondi et pour la double précision. Nous avons ensuite étendu ces tests pour prendre en compte les différentes précisions (double étendue et quadruple) et d'autres critères, à savoir le respect de l'ensemble d'arrivée ($]-\frac{\pi}{2}, \frac{\pi}{2}[$ pour l'arc-tangente par exemple), la monotonie et la parité de la fonction s'il y a lieu.

En collaboration avec Damien Stehlé (ENS Paris), nous avons mis au point un nouvel algorithme pour trouver dans un intervalle donné tous les « pires cas » d'une fonction analytique régulière [49]. Cet algorithme est basé sur la méthode de Coppersmith pour trouver de petites racines d'une équation polynomiale modulaire [56][57][55], qui lui-même utilise la réduction des réseaux [68]. Le nouvel algorithme trouve en $N^{3/5+\varepsilon}$ opérations les pires cas sur un intervalle de N valeurs, contre $N^{2/3+\varepsilon}$ pour le meilleur algorithme connu auparavant [7]. Un exemple de pire cas de la fonction 2^x en quadruple précision, obtenu avec ce nouvel algorithme, est le suivant :

```
> N:=2^113: x:=-1/2+248114878814265552206O/N:
> evalf(2*N*2^x, 60);
                                                                    35
      .146860332744167700291607993713133989999999999999999999996252 10
```

7. Contrats industriels

7.1. Interface MuPAD-Scilab

Mots clés : *calcul formel*, *calcul numérique*, *interface*.

Participants : Fabrice Rouillier, Paul Zimmermann.

En novembre 2001 a été signé entre la société SciFace (qui distribue le logiciel de calcul formel MuPAD) et l'INRIA Lorraine (représentant l'Université Pierre et Marie Curie) un accord de coopération autour du programme UDXF d'échange de données binaires. UDXF prend en paramètre un protocole de communication, par exemple celui du brevet UDX, mais ce peut être aussi un autre protocole. Dans le cadre de cet accord, SciFace a le droit d'utiliser UDXF pour l'interface MuPAD-Scilab. En échange, SciFace participe au développement et à l'amélioration de UDX. La version 2.5.1 de MuPAD, distribuée depuis septembre 2002, intègre cette interface.

7.2. Robots parallèles

Mots clés : *calcul formel*, *systèmes algébriques*, *robots parallèles*.

Participants : Fabrice Rouillier [correspondant du projet SPACES], Jean-Charles Faugère, Jean-Pierre Merlet [correspondant du projet COPRIN], Luc Rolland.

Depuis 1997, nous travaillons avec la société CMW (Constructions Mécaniques de Vosges Marioni) sur la conception d'un robot parallèle pour l'usinage grande vitesse. L'année 2002 a été réservée à la recherche de partenaires pour constituer une chaîne complète (logiciels et composants électroniques) visant à proposer un nouveau système de commande pour ce type de manipulateurs. Avec l'aide de la Direction des Ressources Industrielles de l'INRIA, un contrat de coopération avec une société française spécialisée dans la modélisation et la commande de procédés et de systèmes devrait être établi prochainement.

8. Actions régionales, nationales et internationales

8.1. Actions nationales

8.1.1. Action Spécifique CNRS « Robots Cuspidaux »

Mots clés : *calcul formel*, *géométrie réelle*, *robot série*.

Participants : Fabrice Rouillier [correspondant], Solen Corvez, Jean-Charles Faugère, Marie-Françoise Roy, Mohab Safey El Din.

Le projet *Robots Cuspidaux et Racines Triples* se déroule dans le cadre d'une action spécifique CNRS *Maths-Stic*. Deux équipes CNRS (équipe CMAO et Productique de l'IRCCyN à Nantes, équipe de Géométrie Réelle et Calcul Formel de l'IRMAR à Rennes) et deux projets INRIA (COPRIN, SPACES) y participent.

Ce projet a débuté en juin 2002 pour une durée d'un an.

8.1.2. Action Spécifique CNRS « Arithmétique des Ordinateurs »

Mots clés : *arithmétique virgule fixe*, *arithmétique entière*, *implantation matérielle*, *algorithme*, *preuve*, *précision*.

Participants : Guillaume Hanrot, Vincent Lefèvre, Paul Zimmermann.

Cette action s'intéresse à tout ce qui concerne l'implantation des opérations arithmétiques et des fonctions usuelles. À ce titre, nous étudions les systèmes de numération, les algorithmes arithmétiques, l'implantation matérielle, logicielle et hybride des opérations et fonctions, ainsi que les problèmes d'estimation et de maîtrise de l'erreur numérique.

Outre SPACES, cette action regroupe le projet Arénaire (INRIA Rhône-Alpes, coordinateur), l'équipe CHPV du LIP6 (contact J.-M. Chesneaux), l'équipe « Modèles discrets pour l'arithmétique des ordinateurs »

du LIAFA (contact Christiane Frougny), le projet « Arithmétique des Ordinateurs » du LIRMM (contact Jean-Claude Bajard), l'équipe de Philippe Langlois du MANO (Perpignan), et le projet Lemme (INRIA Sophia-Antipolis).

8.1.3. Action concertée incitative « Jeunes Chercheurs » - Résolution des systèmes algébriques avec paramètres

Participants : Jean-Charles Faugère [chef de projet], Fabrice Rouillier.

Cette action d'une durée de 3 ans a débuté en janvier 2001. Le but est de mettre au point des outils performants pour la résolution des systèmes algébriques à paramètres. Un certain nombre de travaux en cours dans le projet (zéros réels des systèmes paramétrés ou de dimension positive, tracé de courbes implicites) sont directement liés à cette action. Initialement portée par deux personnes cette action s'étoffe puisque M. Bardet, S. Corvez, Ph. Aubry, D. Lazard et M. Safey El Din contribuent activement aux travaux de recherche.

8.1.4. Action concertée incitative Cryptologie « PolyCrypt »

Participants : Guillaume Hanrot [chef de projet], Bill Allombert, Gwenolé Ars, Magali Bardet, Abdolali Basiri, Jean-Charles Faugère, Nicolas Gurel.

L'action « PolyCrypt », menée dans le cadre de l'ACI Cryptologie pilotée par le ministère, regroupe les chercheurs de SPACES listés ci-dessus ainsi que D. Augot (projet CODES), N. Brisebarre (Université de St-Étienne) et Ph. Gaborit (Université de Limoges). Cette action a été acceptée en juillet 2001 pour une durée de 3 ans.

Il s'agit d'une action d'interface, étudiant des applications à la cryptologie de domaines situés en amont de celle-ci, principalement ceux de la compétence du projet, à savoir l'arithmétique (dans le cas des corps finis) et les systèmes polynomiaux.

Nos premiers objectifs sont d'étudier l'algorithmique des corps finis et d'en offrir une implantation la plus efficace et générique possible (au-delà des applications cryptographiques, il serait souhaitable d'avoir une arithmétique pour des valeurs moyennes du degré et de la caractéristique), et d'étudier les problèmes de systèmes polynomiaux quand le corps de base est spécifiquement fini ; dans ce cas, on dispose d'équations supplémentaires du type $x^N = x$ sur les variables, et on peut se demander de quelle façon les exploiter dans la procédure d'élimination.

Les applications à la cryptographie sont décrites dans la section 4.5.

8.2. Actions européennes

8.2.1. Réseau RAAG

Mots clés : *calcul formel*, *géométrie algébrique réelle*.

Participants : Fabrice Rouillier [correspondant], Marie-Françoise Roy, Mohab Safey El Din.

Le projet SPACES participe au projet européeen RAAG (*Real Algebraic and Analytic Geometry*), F. Rouillier étant co-responsable de la partie *Applications et liaison avec l'industrie*.

RAAG est un réseau de formation par la recherche (*Research Training Network*) du programme « *Human Potential* » de la Commission Européenne, subventionné pour une durée de 48 mois à compter de mars 2002. Le réseau est géré par l'Université de Passau (Allemagne), la coordination du travail de l'équipe française étant assurée par l'équipe de géométrie réelle et calcul formel de l'Université de Rennes I.

Le but premier de ce projet est d'accroître les liens entre les différents domaines de recherches balayés par les thèmes du projet (géométrie algébrique réelle, géométrie analytique, complexité, calcul formel, applications etc.) par le biais de conférences, d'écoles et d'échanges de jeunes chercheurs. Il regroupe un très grand nombre d'équipes européennes et en particulier la plupart des équipes françaises travaillant dans les domaines s'inscrivant dans les thèmes du projet.

8.3. Visites et invitations de chercheurs

Yves Pétermann (Université de Genève) a été invité deux mois (août et octobre) pour travailler avec P. Zimmermann et Jean-Luc Rémy (avant-projet Adage) sur le calcul de la fonction zêta de Riemann dans le plan complexe, et son implantation.

Allan Steel, l'un des développeurs du logiciel Magma, a été invité une semaine en mai 2002 au LIP6. Allan Steel a présenté une extension et une implantion en Magma de l'algorithme F_4 sur un anneau euclidien et a réalisé une première implantation de l'algorithme F_5 [25] en Magma.

P. Zimmermann a rendu visite à Richard Brent à Oxford (Angleterre) fin août ; avec Samuli Larvala, ils ont poursuivi un travail en commun sur la recherche de trinômes primitifs de grand degré. P. Zimmermann a également rendu visite au groupe MuPAD à Paderborn (Allemagne) fin novembre, où il a discuté de la possibilité de développer un noyau libre pour le langage MuPAD.

9. Diffusion des résultats

9.1. Articles et conférences de synthèse

F. Rouillier a donné plusieurs exposés de synthèse sur la résolution des systèmes algébriques en général (journées nationales de Géométrie Algorithmique, session spéciale en marge de la conférence ISSAC) et sur la résolution des systèmes paramétrés ou en dimension positive en particulier (Colloque *Mathématiquess effectives* à Poitiers, *Workshop on Positive Polynomials* à Oberwolfach, Allemagne). Un article de synthèse résumant une partie de ces interventions a par ailleurs été accepté pour publication [36].

P. Zimmermann a rédigé un article sur l'algorithme ECM [52] (factorisation d'entier à l'aide de courbes elliptiques) pour l'Encyclopédie sur la Sécurité de l'Information, ouvrage collectif de la communauté internationale (à paraître).

D. Wang a écrit un livre [11] en chinois sur des méthodes d'élimination avec applications. Ce livre est basé sur les matériaux présentés dans son ouvrage *Elimination Methods* (Springer, 2001) et sa thèse d'habilitation (INPG, janvier 1999) avec améliorations structurales.

9.2. Exposés invités

J.C. Faugère a donné des exposés invités à Cambridge (Newton Institute) [30], dans les conférences *RWCA 2002* (Manheim, Allemagne) [27], *Application of Commutative Algebra* (Catania, Italie, avril 2002) [26], *YACC 2002* (Porquerolles, France, juin 2002) [29] *Application of Computer Algebra* (Volos, Grèce, juin 2002) [28].

D. Lazard a donné des conférences invitées au colloque de clôture des ARC CoSTIC et Visi3D (Paris, décembre 2001) [66], dans les conférences *Application of Commutative Algebra* (Catania, Italie, avril 2002) [32], *Application of Computer Algebra* (Volos, Grèce, juin 2002) [31] et *Colloque de Mathématiques effectives* (Poitiers, septembre 2002) [33]. La traduction en anglais d'un article de D. Lazard [67] datant de 1981 est une forme de reconnaissance difficilement classable, mais du même ordre que les exposés invités.

V. Lefèvre a donné un exposé de vulgarisation à l'Université des Sciences et Technologies de Lille dans le cadre d'un cycle de conférences sur l'erreur (mars 2002) ; suite à cet exposé, un article, co-écrit avec J.-M. Muller, a été proposé à *Pour la Science*.

V. Lefèvre a présenté la bibliothèque MPFR lors du groupe de travail *Computability and Complexity in Analysis* (CCA 2002, Málaga, Espagne, juillet 2002).

F. Rouillier a donné des exposés invités à Oberwolfach (*Workshop on Positive Polynomials*, février 2002 [39]), Santander (*Workshop on Computations in Real Algebraic Geometry and Applications*, juin 2002 [40]), Poitiers (Colloque *Mathématiques effectives*, septembre 2002 [37], Obernai (*Journées Nationales de Géométrie Algorithmique*, octobre 2002 [38]).

D. Wang a donné des exposés invités à l'Université d'Ontario de l'Ouest (Canada, janvier 2002), au Séminaire de Kolchin sur l'Algèbre Différentielle à l'Université de la Ville de New York et à l'Université d'État du Kent (États-Unis, mars 2001).

P. Zimmermann a donné un exposé invité à la conférence SCAN'2002 (Paris, sept. 2002).

9.3. Organisation de conférences et journées

J.-C. Faugère et F. Rouillier ont participé à l'organisation de l'école d'été *Outils de Calcul Symbolique Numérique Collaboratif* à Giens en septembre 2002. J.-C. Faugère et F. Rouillier ont participé à l'édition d'un CD-ROM sur les logiciels libres.

G. Hanrot et F. Rouillier participent à l'organisation des journées nationales de calcul formel qui auront lieu à Luminy en janvier 2003.

F. Rouillier a co-organisé le *Workshop on Computations in Real Algebraic Geometry and Applications* (Santander, Juin 2002) dans le cadre du projet européen RAAG.

F. Rouillier et P. Zimmermann ont participé à l'organisation des journées sur le calcul formel et les logiciels libres à Lyon (*Workshop on Open Source Computer Algebra*, mai 2002).

D. Wang a co-organisé le *Seminar on Geometric Computation* (Hefei, Chine, Avril 2002) et organisé la session *Polynomial Algebra* à l'*International Congress of Mathematical Software* (ICMS) (Pékin, août 2002).

9.4. Comités scientifiques, de programme et de rédaction

D. Lazard a été membre du comité de programme de ISSAC 2002 (juillet, Lille). Il fait partie du *Advisory Board* de la série de colloques MEGA (*Effective Methods in Algebraic Geometry*). Il fait partie du comité de quatre personnes chargé de proposer les membres du bureau du SIGSAM (*Special Interest Group in Symbolic and Algebraic Manipulation*) de l'ACM (*Association for Computing Machinery*).

F. Rouillier fait partie du comité scientifique de l'action spécifique *Calcul Formel* du département STIC du CNRS.

D. Wang est membre du comité de rédaction du *Journal of Symbolic Computation*, de la serie d'ouvrages sur la mécanisation des mathématiques (publiés en chinois par Science Press, Pékin), et des comités de programme pour :

- AISC 2002 (*International Conference on Artificial Intelligence and Symbolic Computation*) (Marseille, France, 1-5 juillet 2002),
- ICMS 2002 (Pékin, Chine, 17-19 août 2002),
- ADG 2002 (*International Workshop on Automated Deduction in Geometry*) (Linz, Autriche, 4-6 septembre 2002),
- ATCM 2002 (*Asian Technology Conference in Mathematics*) (Université Multimédia, Malaisie, 17-21 décembre 2002),
- ASCM 2003 (*Asian Symposium on Computer Mathematics*) (Pékin, Chine, 17-19 avril 2003).

P. Zimmermann fait partie du comité de programme de la conférence Arith'16 qui aura lieu à Saint-Jacques de Compostelle (Espagne) en juin 2003. Il est également membre du comité de direction du GDR ALP, et responsable du pôle « Algorithmique et Calcul Formel » de ce GDR.

9.5. Enseignement

F. Rouillier participe à diverses actions pour l'enseignement du calcul formel à partir du premier cycle universitaire (avec l'Université de Lille et l'Université Technologique de Compiègne). À ce titre, il a donné 4 exposés introductifs à l'université de Lille sur la résolution des systèmes algébriques.

G. Hanrot a assuré le cours de DEA « Codes correcteurs d'erreurs » du DEA de Nancy, et la moitié du cours de « Théorie algorithmique des nombres » du DEA Algorithmique (ENS, ENS Cachan, ENST, Universités Paris 6, 7, 11, École Polytechnique), où J.-C. Faugère, D. Lazard et F. Rouillier ont assuré deux cours dans la filière « Géométrie Calcul formel ».

D. Wang a donné un cours court sur le Calcul Formel et Géométrique (sous invitation) à l'Université des Sciences et Technologies de Chine (avril 2002).

10. Bibliographie

Bibliographie de référence

[1] P. AUBRY. *Ensembles triangulaires de polynômes et résolution de systèmes algébriques.* thèse de doctorat, Université Paris 6, 1999.

[2] P. AUBRY, D. LAZARD, M. MORENO MAZA. *On the Theories of Triangular Sets.* in « Journal of Symbolic Computation, Special Issue on Polynomial Elimination », volume 28, 1999, pages 105-124.

[3] J. FAUGÈRE, D. LAZARD. *The Combinatorial Classes of Parallel Manipulators.* in « Mechanism and Machine Theory », volume 30, 1995, pages 765-776.

[4] G. HANROT, F. MORAIN. *Solvability by Radicals From an Algorithmic Point of View.* in « International Symposium on Symbolic and Algebraic Computation - ISSAC'2001, London, Ontario, Canada », ACM, éditeurs B. MOURRAIN., 2001.

[5] D. LAZARD. *Stewart Platforms and Gröbner Basis.* in « Proceedings of Advances in Robotics Kinematics », pages 136-142, 1992.

[6] D. LAZARD, J.-P. MERLET. *The (True) Stewart Platform has 12 Configurations.* in « Proc. of IEEE Conference on Robotics and Vision, San Diego », 1994.

[7] V. LEFÈVRE. *Moyens arithmétiques pour un calcul fiable.* Thèse de doctorat, École Normale Supérieure de Lyon, 2000.

[8] F. ROUILLIER. *Real Root Counting For Some Robotics Problems.* in « Solid Mechanics and its Applications, Kluwer Academic Publishers », volume 40, 1995, pages 73-82.

[9] F. ROUILLIER. *Solving Zero-Dimensional Systems Through the Rational Univariate Representation.* in « Journal of Applicable Algebra in Engineering, Communication and Computing », numéro 5, volume 9, 1999, pages 433-461.

[10] J.-C. FAUGÈRE.. *A New Efficient Algorithm for Computing Gröbner Bases (F4).*-. in « Journal of Pure and Applied Algebra », numéro 1-3, volume 139, 1999, pages 61-88.

Livres et monographies

[11] D. WANG. *Méthodes d'élimination avec applications.* Science Press, Beijing, 2002, en chinois.

Articles et chapitres de livre

[12] P. AUBRY, F. ROUILLIER, M. SAFEY EL DIN. *Real Solving for Positive Dimensional Systems.* in « Journal of Symbolic Computation », numéro 6, volume 34, 2002, pages 543-560.

[13] A. BASIRI, A. ENGE, J.-C. FAUGERE, N. GUREL. *The Arithmetic of Jacobian Groups of Superelliptic Cubics.* in « Math Comp », 2002, http://www.inria.fr/rrrt/rr-4618.html, Existe en Rapport INRIA RR-4618.

[14] M. BENITO, W. CREYAUFMÜLLER, J. L. VARONA, P. ZIMMERMANN. *Aliquot Sequence 3630 Ends After Reaching 100 Digits.* in « Experimental Mathematics », numéro 2, volume 11, 2002, pages 201-206.

[15] R. P. BRENT, S. LARVALA, P. ZIMMERMANN. *A Fast Algorithm for Testing Reducibility of Trinomials mod 2 and Some New Primitive Trinomials of Degree 3021377.* in « Mathematics of Computation », 2002, à paraître.

[16] Y. BUGEAUD, G. HANROT, M. MIGNOTTE. *Sur l'équation diophantienne* $(x^n - 1)/(x - 1) = y^q$*, III.* in « Proc. London Math. Soc. », numéro 3, volume 84, 2002, pages 59-78.

[17] L. DUPONT, D. LAZARD, S. LAZARD, S. PETITJEAN. *Towards the Robust Intersection of Implicit Quadrics.* éditeurs J. WINKLER, M. NIRANJAN., in « Uncertainty in Geometric Computations », série International Series in Engineering and Computer Science, volume 704, Kluwer Academic Publishers, 2002, chapitre 5, pages 59-68.

[18] J.-C. FAUGERE, M. HERING, J. PHAN. *The Membrane Inclusions Curvature Equations.* in « Advances in Applied Mathematics », 2002.

[19] D. LAZARD. *Solving Quintics by Radicals.* in « The Legacy of Niels Henrik Abel », Springer, 2002.

[20] F. ROUILLIER, P. ZIMMERMANN. *Efficient Isolation of a Polynomial Real Roots.* in « Journal of Computational and Applied Mathematics », 2002, à paraître.

[21] D. WANG. *A Simple Method for Implicitizing Rational Curves and Surfaces.* in « Journal of Symbolic Computation », 2002, submitted for publication.

[22] D. WANG. *Automated Generation of Diagrams with Maple and Java.* éditeurs M. JOSWIG, N. TAKAYAMA., in « Algebra, Geometry, and Software Systems », Springer-Verlag, Berlin Heidelberg, 2002, pages 277-287, to appear.

Communications à des congrès, colloques, etc.

[23] S. CORVEZ, F. ROUILLIER. *Using Computer Algebra Tools to Classify Serial Manipulators.* in « Fourth International Workshop on Automated Deduction in Geometry, ADG 2002, Hagenberg, Austria », 2002.

[24] G. DOS REIS, B. MOURRAIN, P. TRÉBUCHET, F. ROUILLIER. *An Environment for Symbolic and Numeric Computation.* in « International Congress of Mathematical Software - ICMS'2002, Beijing, China », 2002.

[25] J.-C. FAUGERE. *A New Efficient Algorithm for Computing Gröbner Bases Without Reduction to Zero F5.* in « ISSAC 2002, Villeneuve d'Ascq, France », 2002.

[26] J.-C. FAUGERE. *A New Efficient Algorithm for Computing Gröbner Bases Without Reduction to Zero.* in « Workshop on application of Groebner Bases 2002, Catania, Spain », 2002.

[27] J.-C. FAUGERE. *A New Efficient Algorithm for Computing Gröbner Bases Without Reduction to Zero.* in « RWCA 2002 Manheim, Mannheim, Germany », 2002.

[28] J.-C. FAUGERE. *Classification of All Planar Central Configurations of N Bodies With Equal Masses in the Case of the Logarithmic Potential and $N < 8$.* in « ACA 2002 Volos (Grece), Volos, Grèce », 2002.

[29] J.-C. FAUGERE. *Gröbner Bases and Application to HFE.* in « YACC 2002 , Porquerolles, France », 2002.

[30] J.-C. FAUGERE. *Solving Polynomial Systems. Algorithms and Applications..* in « Computer Algebra in Applications to Integrable Systems. Cambridge, Cambridge, England », 2002.

[31] D. LAZARD. *Central Configurations of Four Gravitational Masses With an Axis of Symmetry.* in « 8th International Conference on Applications of Computer Algebra (ACA 2002), Volos, Grèce », 2002.

[32] D. LAZARD. *Central Configurations of Four Gravitational Masses with an Axys of Symmetry.* in « Applications of Computer Algebra, Catania, Italie », 2002.

[33] D. LAZARD. *Discussion du nombre de solutions réelles d'un système dépendant de paramètres..* in « Mathématiques effectives, Poitiers », 2002.

[34] N. REVOL, F. ROUILLIER. *Motivations for an Arbitrary Precision Interval Arithmetic and the MPFI Library.* in « Validated Computing (Toronto, Canada) », 2002.

[35] N. REVOL, F. ROUILLIER. *MPFI : a Library for Arbitrary Precision Interval Arithmetic.* in « SCAN (Paris, France) », 2002.

[36] F. ROUILLIER. *Efficient Algorithms Based on Critical Points Method.* in « Workshop on Algorithmic and Quantitative Aspects of Real Algebraic Geometry in Mathematics and Computer Science - revised papers », 2002.

[37] F. ROUILLIER. *Outils pour l'étude des zéros réels de systèmes algébriques.* in « Colloque de Mathématiques effectives, Poitiers, France », 2002.

[38] F. ROUILLIER. *Outils pour l'Étude des Zéros Réels de Systèmes Algébriques.* in « Journées de Géométrie Algorithmique, Obernay, France », 2002.

[39] F. ROUILLIER. *Computational Problems Related to Positive Polynomials.* in « Workshop on positive polynomials, Oberwolfach, Oberwolfach, Germany », 2002.

[40] F. ROUILLIER. *Real Solving and Parallel Robots.* in « Workshop on Computations and Applications in RAAG », 2002.

[41] D. WANG. *EPSILON : A Library of Software Tools for Polynomial Elimination.* in « Mathematical Software - Proceedings of the First International Congress of Mathematical Software », World Scientific, Singapore New Jersey, éditeurs A. M. COHEN, X.-S. GAO, N. TAKAYAMA., pages 379-389, 2002.

[42] D. WANG. *GEOTHER 1.1 : Handling and Proving Geometric Theorems Automatically.* in « Automated Deduction in Geometry - ADG 2002 Proceedings », Springer-Verlag, Berlin Heidelberg, éditeurs F. WINKLER., 2002, To appear (paper presented at ADG 2002 and submitted for publication).

Rapports de recherche et publications internes

[43] D. AUGOT, M. BARDET, J.-C. FAUGERE. *Efficient Decoding of (Binary) Cyclic Codes Beyond the Correction Capacity of the Code Using Gröbner Bases.* Rapport de recherche, LORIA, 2002, http://www.inria.fr/rrrt/rr-4652.html, Version partielle soumise a ISIT 2003 Japan.

[44] Y. BERTOT, N. MAGAUD, P. ZIMMERMANN. *A Proof of GMP Square Root Using the Coq Assistant.* rapport technique, numéro RR-4475, Institut National de Recherche en Informatique et en Automatique, 2002, http://www.inria.fr/rrrt/rr-4475.html.

[45] S. CORVEZ, F. ROUILLIER. *Using Computer Algebra Tools to Classify Serial Cuspidal Manipulators.* Rapport de recherche, INRIA, 2002, http://www.inria.fr/rrrt/rr-4653.html.

[46] G. HANROT, P. ZIMMERMANN. *A Long Note on Mulders' Short Product.* rapport technique, numéro RR-4654, Institut National de Recherche en Informatique et en Automatique, 2002, http://www.inria.fr/rrrt/rr-4654.html.

[47] V. LEFÈVRE. *Multiplication by an Integer Constant : Lower Bounds on the Code Length.* Rapport de recherche, INRIA, 2002, http://www.inria.fr/rrrt/rr-4493.html.

[48] M. SAFEY EL DIN, E. SCHOST. *Properness Defects of Projections and Computation of One Point in Each Connected Component of a Real Algebraic Set.* Rapport de recherche, 2002, http://www.inria.fr/rrrt/rr-4598.html.

[49] D. STEHLÉ, V. LEFÈVRE, P. ZIMMERMANN. *Worst Cases and Lattice Reduction.* Rapport de recherche, numéro 4586, INRIA, 2002, http://www.inria.fr/rrrt/rr-4586.html.

Divers

[50] Y.-F. PÉTERMANN, J.-L. RÉMY. *Error Analysis for an Arbitrary Precision in Computing $\zeta(s)$ With the Cohen-Olivier Algorithm : Complete Description of the Real Case and Preliminary Report on the General Case.* 2002.

[51] Y.-F. PÉTERMANN, J.-L. RÉMY. *On the Cohen-Olivier Algorithm for Computing $\zeta(s)$: Error Analysis in the Real Case for an Arbitrary Precision.* 2002.

[52] P. ZIMMERMANN. *The Elliptic Curve Method.* Entry in the Encyclopedia of Information Security, éditeurs B. KALISKI., Kluwer, 2002, à paraître.

Bibliographie générale

[53] H. COHEN, M. OLIVIER. *Calcul des valeurs de la fonction zêta de Riemann en multiprécision.* in « Comptes-Rendus de l'Académie des Sciences », volume 314, 1992, pages 427-430.

[54] G. COLLINS. *Quantifier Elimination for Real Closed Fields by Cylindrical Algebraic Decomposition.* in « Springer Lecture Notes in Computer Science », volume 33, 1975, pages 515-532.

[55] D. COPPERSMITH. *Finding Small Solutions to Small Degree Polynomials.* in « Proceedings of CALC'01 », série Lecture Notes in Computer Science, numéro 2146, pages 20-31, 2001.

[56] D. COPPERSMITH. *Finding a Small Root of a Bivariate Integer Equation ; Factoring with High Bits Known.* in « Proceedings of Eurocrypt'96 », série Lecture Notes in Computer Science, volume 1070, Springer-Verlag, éditeurs U. MAURER., pages 178-189, 1996.

[57] D. COPPERSMITH. *Finding a Small Root of a Univariate Modular Equation.* in « Proceedings of Eurocrypt'96 », série Lecture Notes in Computer Science, numéro 1070, Springer-Verlag, éditeurs U. MAURER., pages 155-165, 1996.

[58] D. DANEY, G. HANROT, V. LEFÈVRE, F. ROUILLIER, P. ZIMMERMANN. *The MPFR Library.* http://www.mpfr.org, 2001.

[59] V. DER WAERDEN. *Moderne Algebra.* Springer-Verlag, 1955.

[60] N. ELKIES. *Rational Points Near Curves and Small Nonzero $|x^3 - y^2|$ via Lattice Reduction.* in « Proceedings of ANTS-IV », série Lecture Notes in Computer Science, volume 1838, Springer-Verlag, éditeurs W. BOSMA., pages 33-63, 2000.

[61] L. EULERUS. *Opera omnia. Series secunda (Opera mechanica et astronomica), Vol. XXV : Commentationes astronomicae ad theoriam perturbationum pertinentes, Vol. primum.* Auctoritate et impensis Societatis Scientiarum Naturalium Helveticae. Orell Füssli, Zurich, 1960, chapitre Considerationes de motu corpurum coelestium, version originale en 1762.

[62] G. HANROT, M. QUERCIA, P. ZIMMERMANN. *Speeding up the Division and Square Root of Power Series.* Rapport de recherche, numéro 3973, Institut National de Recherche en Informatique et en Automatique, 2000, http://www.inria.fr/RRRT/RR-3973.html.

[63] H. HONG. *Comparison of Several Decision Algorithms for the Existential Theory of the Reals.* rapport technique, RISC-Linz, Johannes Kepler University, 1991.

[64] A. H. KARP, P. MARKSTEIN. *High-Precision Division and Square Root.* in « ACM Transactions on Mathematical Software », numéro 4, volume 23, 1997, pages 561-589.

[65] D. LAZARD. *On the Specification for Solvers of Polynomial Systems.* in « 5th Asian Symposium on Computer Mathematics - ASCM 2001, Matsuyama, Japon », série Lecture Notes Series on Computing, volume 9, World Scientific, pages 66-75, 2001.

[66] D. LAZARD. *Optimality of the Parameterization of Quadrics and Their Intersections.* in « Journées de cloture Visi3D et CoSTIC, Paris, France », 2001, http://www.loria.fr/publications/2001/A01-R-441/A01-R-441.ps.

[67] D. LAZARD. *Solving Systems of Algebraic Equations.* in « SIGSAM Bulletin », numéro 3, volume 35, 2001, pages 11-37.

[68] A. K. LENSTRA, H. W. LENSTRA, L. LOVÁSZ. *Factoring Polynomials with Rational Coefficients.* in « Mathematische Annalen », volume 261, 1982, pages 515-534.

[69] M. LERCH, G. TISCHLER, J. WOLFF VON GUDENBERG, W. HOFSCHUSTER, W. KRÄMER. *The Interval Library filib++ 2.0 - Design, Features and Sample Programs.* Research Report, numéro 2001/4, Universität Wuppertal, Germany, 2001.

[70] T. MULDERS. *On Short Multiplications and Divisions.* in « AAECC », numéro 1, volume 11, 2000, pages 69-88.

[71] J. E. OMRI. *Analyse géométrique et cinématique des mécanismes de type manipulateur.* thèse de doctorat, Université de Nantes, 1996.

[72] F. ROUILLIER, M. SAFEY EL DIN, E. SCHOST. *Solving the Birkhoff Interpolation Problem Via the Critical Point Method : an Experimental Study.* in « Automated Deduction in Geometry, Zurich, Switzerland », 2000.

[73] P. WENGER, J. E. OMRI. *Changing Posture for Cuspidal Robot Manipulators.* in « Proceeding of the 1996 IEEE Int. Conf on Robotics and Automation », pages 3173-3178, 1996.

[74] P. WENGER. *Some Guidelines for the Kinematic Design of New Manipulators.* in « Mechanism and Machine Theory », volume 35, 2000, pages 437-449.

[75] P. WENGER. *Classification of 3R Positioning Manipulators.* in « Journal of Mechanical Design », volume 120, 1998, pages 327-332.

[76] P. ZIMMERMANN. *Karatsuba Square Root.* Research Report, numéro 3805, INRIA, 1999, http://www.inria.fr/rrrt/rr-3805.html.