Projet EURÉCA

Preuve, calcul symbolique et logique

Localisation : *Nancy*[1]

Mots-clés : lambda-calcul, logique, réécriture, expression régulière, preuve de programme, calcul symbolique, structure décomposable, génération aléatoire, jeu de test.

1 Composition de l'équipe

Responsable scientifique

Pierre Lescanne, directeur de recherche CNRS

Secrétaire

Céline Cordier

Personnel INRIA

Fabrice Rouillier, chargé de recherche
Grégory Kucherov, chargé de recherche
Paul Zimmermann, chargé de recherche

Personnel CNRS

Nicolas Hermann, chargé de recherche
Jean-Luc Rémy, chargé de recherche

Personnel Université

Zine-El-Abidine Benaissa, Attaché Temporaire d'Enseignement et de Recherche, Université Henri Poincaré Nancy 1
Boutheïna Chetali, Attachée Temporaire d'Enseignement et de Recherche, Université Henri Poincaré Nancy 1
Daniel Briaud, Attaché Temporaire d'Enseignement et de Recherche, Université Henri Poincaré Nancy 1
Adam Cichon, Professeur, Université Henri Poincaré Nancy 1
Hélène Touzet, Attachée Temporaire d'Enseignement et de Recherche, Université Henri Poincaré Nancy 1

[1] Projet commun à l'INRIA et au CRIN, URA 262 du CNRS et des universités Henri Poincaré Nancy 1, Nancy 2 et INPL.

Chercheurs invités

Sapphorain Pétermann, Université de Genève, Suisse, du 1er septembre au 31 octobre 1996
Kristoffer Høgsbro Rose, Basic Research In Computer Science, Department of Computer Science, Danemark, du 1er mars au 31 mai 1996

Chercheurs doctorants

Otmane Aït-Mohamed, boursier du gouvernement algérien
François Bertault, boursier Centre Charles Hermite
Guillaume Bonfante, allocataire MESR (à partir de novembre 1996)
Djamila Bouzid, boursière INRIA
Vladimir Grébinski, boursier INRIA
Barbara Heyd, allocataire MESR
Laurent Juban, allocataire MESR (à partir de novembre 1996)
Frédéric Lang, allocataire MESR
Sohame Selhab

2 Présentation du projet

Le projet EURÉCA possède deux thématiques scientifiques complémentaires qui ont un intérêt commun pour le calcul symbolique, les structures algébriques qu'il sous-tend et les algorithmes qu'il met en œuvre.

L'algorithmique : cette composante s'intéresse à l'étude et à la mise en œuvre d'algorithmes pour le *calcul symbolique*. L'intérêt se porte aussi bien sur les études théoriques des propriétés et de la complexité des algorithmes que sur les applications au génome, au dessin de structures combinatoires et à la géométrie algorithmique.

La preuve[2] donne lieu à deux types de recherches, l'une sur la mise en œuvre de notre savoir-faire pour la *vérification formelle de systèmes informatiques* et l'autre sur les problèmes fondamentaux de la *démonstration automatique*. Le projet EURÉCA a en particulier étudié la correction de divers protocoles, notamment des protocoles de communication, et passe progressivement des études de cas à l'analyse de protocoles plus réalistes. Il a aussi développé des environnements de preuves pour UNITY. Sur le plan théorique, la théorie de la démonstration, les ordinaux, la complexité des problèmes de comptage en filtrage et unification, le calcul des substitutions explicites constituent les principaux axes de recherche.

3 Actions de recherche

3.1 Algorithmique

3.1.1 Combinatoire

Participants : François Bertault, Vladimir Grébinski, Grégory Kucherov, Paul Zimmermann

Mots-clés : expression régulière, structure décomposable, génération aléatoire, jeu de test.

[2] Le substantif « preuve » est habituellement utilisé par les philosophes et les policiers quand il s'agit d'apporter l'évidence d'un fait ; l'influence anglo-saxonne le fait de plus en plus employer dans le sens de l'objet mathématique qui réalise la démonstration automatique ou mécanique par ordinateur d'une propriété logique.

Génération aléatoire de structures combinatoires

L'étude de la génération aléatoire de structures combinatoires trouve des applications variées, dont un exemple significatif est la réalisation de jeux de tests pour la validation expérimentale ou l'étalonnage d'algorithmes. La théorie des structures décomposables permet de définir de façon générique, à partir d'un faible nombre de constructeurs, une infinité de classes de structures. On pourra par exemple décrire des structures de données aussi variées que les permutations, les arbres généraux ou encore les graphes fonctionnels. Le logiciel `combstruct` développé en commun par les projets ALGO et EURÉCA, et intégré au système de calcul formel Maple, est aujourd'hui le système de référence en génération aléatoire uniforme de structures combinatoires décomposables. En pratique, il devient cependant difficile de générer des structures de grande taille pour des raisons d'occupation mémoire et de temps de calcul. Nous travaillons actuellement à la réalisation d'une extension de `combstruct` permettant de générer directement du code `C` compilable. Les premiers résultats, basés sur l'utilisation de bibliothèques d'entiers longs, font déjà apparaître des gains en performance très importants.

Représentation graphique de structures combinatoires

La représentation graphique de structures combinatoires est un outil permettant d'aider à la compréhension et de donner l'intuition de certaines propriétés mathématiques. Nous avons étendu le champ d'application du logiciel `Adocs`, dédié aux tracés de structures générées de façon aléatoire par `combstruct`, en réalisant un outil plus général, appelé `CGraph`. Ce système complet repousse les limitations des systèmes de tracé de graphes actuels en permettant la représentation d'une large classe de graphes possédant à la fois des relations d'adjacence et d'inclusion entre les nœuds, et les graphes composés. Il est possible de créer de façon interactive et de dessiner automatiquement des structures imbriquées, que l'on retrouve dans des domaines d'application variés (ingénierie, sciences sociales, chimie). `CGraph` est conçu dans ce but pour permettre son utilisation comme interface pour la représentation des structures de données imbriquées d'autres programmes.

Combinatoire des mots

En collaboration avec R. Kolpakov de l'Institut Liapunov à Moscou, nous avons obtenu un résultat intéressant dans un sujet classique de la combinatoire des mots, à savoir les propriétés des répétitions. Depuis les travaux de Thue du début du siècle, il est connu qu'il existe des mots infinis « sans carré » sur un alphabet de trois lettres et des mots infinis « sans cube » sur un alphabet de deux lettres. Les mots sans carré et sans cube sont ceux qui ne contiennent pas de facteur de la forme uu et uuu respectivement. Plus généralement, on parle d'un mot sans répétition de puissance n s'il ne contient pas de facteur u^n. La question que nous nous sommes posée est la suivante : quel est le taux minimal limite d'une lettre dans un mot infini sans répétition de puissance n ? Dans l'article [129] soumis à TAPSOFT'97 nous avons prouvé que ce taux est de $1/n$. Ce travail a ouvert une perspective de recherche intéressante que nous comptons explorer dans un proche avenir.

Applications de la recherche combinatoire à l'analyse du génome

Depuis l'arrivée de V. Grébinski en automne 1995, nous travaillons sur des problèmes combinatoires motivés par la cartographie du génome - le problème central dans l'analyse des séquences génomiques. Ce travail se fait en collaboration avec des chercheurs du Laboratoire de Génétique Microbienne de l'INRA (Jouy-en-Josas). Nous avons réussi à développer et à analyser quelques modèles mathématiques pour une méthode de la cartographie du génome pratiquée dans ce laboratoire. Ces modèles s'expriment comme des problèmes de recherche combinatoire (*group testing* en anglais), plus précisément comme des problèmes d'ordonnancement de points sur un cercle à l'aide de questions sur le voisinage. En termes de graphes, le problème consiste à retrouver dans un graphe complet un chemin hamiltonien fixé en se servant de tests sur des sous-graphes complets. Ici, les tests correspondent aux réactions chimiques et donc l'aspect pratique de cette étude consiste à développer des algorithmes efficaces permettant

aux biologistes de diminuer le nombre de réactions chimiques nécessaires pour construire une carte physique du génome. Nous avons développé de tels algorithmes et analysé leur complexité qui, dans certains cas, s'est avérée asymptotiquement optimale. Le papier décrivant ce travail est soumis à la revue Discrete Applied Mathematics [128]. Nous avons implanté certains de ces algorithmes en Maple dans le logiciel appelé `Designer` et nous comptons continuer ce développement pour rendre ces algorithmes utilisables par les biologistes.

Sur le plan théorique, plusieurs nouveaux résultats ont été obtenus dont les principaux concernent le modèle dit additif ou quantitatif (modèle comportant la possibilité de comptage). En particulier, nous avons proposé des algorithmes efficaces (voire optimaux) pour retrouver dans un graphe complet un graphe fixé d'une classe spécifiée, par exemple un chemin hamiltonien, un couplage (*matching*) ou un graphe biparti.

3.1.2 Arithmétique

Participants : François Bertault, Sapphorain Pétermann, Jean-Luc Rémy, Paul Zimmermann

Mots-clés : conjecture de Waring, suite de Golomb.

Sommes de cubes

Dans le cadre de l'action « Calcul symbolique » du Centre Charles Hermite, Olivier Ramaré (IECN) nous avait demandé de vérifier que tout entier $1290740 < n < B$ s'écrit comme somme de 5 cubes, avec B aussi grand que possible, par exemple $B = 10^{14}$. À l'aide d'un lemme d'O. Ramaré, un tel résultat permet de montrer que tout entier n vérifiant la propriété

$$455 \leq n \leq \left(\frac{1}{3}(B/3 - 430247)^{3/2} + 427566\right)^{3/2} + 8043$$

est somme de 7 cubes, soit $455 \leq n \leq 5.13\,10^{29}$ pour $B = 10^{14}$. À l'aide d'un programme écrit en C et parallélisé par F. Bertault pour le Power Challenge Array, nous avons vérifié que tout entier $1290740 < n < 3.375\,10^{12}$ est somme de 5 cubes.[3] Notre programme calculant une décomposition $n = a^3 + b^3 + c^3 + d^3 + e^3$ où a est le plus grand possible, nous avons obtenu en plus des statistiques sur le comportement de $\lfloor n^{1/3} \rfloor - a$, qui ont fait apparaître des oscillations encore inexpliquées.

Le résultat obtenu avec O. Ramaré a depuis été amélioré par Bernard Landreau (Université Bordeaux I), qui a montré avec Jean-Marc Deshouillers et François Hennecart que tout entier $1290740 < n < 10^{16}$ est somme de 5 cubes, ce qui permet d'aller jusqu'à 10^{34} pour les sommes de 7 cubes. La preuve de B. Landreau est basée sur l'analyse des sommes de 4 cubes dans les classes modulo 63 jusqu'à 10^{10} environ, puis sur un lemme permettant de passer des sommes de 4 cubes aux sommes de 5 cubes.

Ce sujet est loin d'être clos, puisque le meilleur résultat effectif connu est celui de McCurley qui a prouvé en 1984 que tout entier supérieur à $\exp(1077334)$ est somme de 7 cubes. Il reste donc un immense espace entre 10^{34} et $\exp(1077334)$, qui vaut environ 10^{467880} !

Un autre problème ouvert est celui de la densité des sommes de 3 cubes. B. Landreau a montré que si l'on suppose que les sommes de 3 cubes suivent le modèle probabiliste d'Erdös et Rényi, alors leur densité est positive et vaut $0.099919 < \delta < 0.099963$. Cela semble concorder avec les résultats expérimentaux qui donnent une densité de 0.099902 jusqu'à $2\,10^{12}$.

Évaluation asymptotique de fonctions arithmétiques

Nous avons répondu à une première partie d'une conjecture concernant le terme reste du développement asymptotique d'une suite « auto-construite » d'entiers. Nous avons montré pour cela que la borne supérieure trouvée par d'autres chercheurs était la meilleure possible. Notre preuve allie raisonnements

[3] Cf. `http://www.loria.fr/~zimmerma/records/cubes.html`

formels et calculs numériques sur des valeurs de plus en plus grandes de la suite. Nous avons fait appel pour ces calculs à un système de calcul formel.

Nous avons ensuite répondu à la deuxième partie de cette conjecture, faisant apparaître le comportement périodique d'une fonction intervenant dans l'expression du terme reste. Un article à soumettre dans une revue est en cours de finition.

Factorisation d'entiers

Au sein de l'opération « Calcul symbolique » du Centre Charles Hermite, et dans le cadre du groupe NFSNET, constitué d'une quinzaine de personnes, nous avons participé à la factorisation de plusieurs nombres des tables de Cunningham, dont en particulier $3^{331}-1$, qui fut factorisé sur le *Power Challenge Array* du Centre Charles Hermite à l'occasion de son inauguration. Les autres nombres de Cunningham factorisés par NFSNET en 1996 sont $2^{524}+1, 2^{536}+1, 2^{934}+1, 2^{974}+1, 2^{559}-1, 5^{226}+1, 5^{505}-1, 5^{515}-1, 6^{206}+1, 6^{199}-1, 7^{181}+1, 7^{187}-1, 10^{158}+1, 10^{163}+1, 10^{167}-1, 11^{142}+1, 11^{146}+1, 12^{137}-1, 12^{139}-1$.

3.1.3 Calcul symbolique

Participants : Fabrice Rouillier, Paul Zimmermann

Mots-clés : équation différentielle ordinaire, factorisation, système algébrique.

Résolution d'équations différentielles ordinaires

Avec Frank Postel (Université de Paderborn), nous avons comparé les sept grands systèmes de calcul formel actuels (Axiom, Derive, Maple, Mathematica, Macsyma, MuPAD et Reduce) pour la résolution des équations différentielles ordinaires. Sur la base d'une cinquantaine de problèmes, nous avons mis en évidence les forces et faiblesses de chaque système, et exhibé plusieurs erreurs, dont certaines ont d'ailleurs déjà été corrigées dans de nouvelles versions [125].

Factorisation de polynômes

En ce qui concerne la factorisation dans $\mathbf{Z}_p[x]$, avec Laurent Bernardin (ETH Zürich) et Michael Monagan (Simon Fraser University), nous avons factorisé en Maple et MuPAD tous les polynômes f_n de Von zur Gathen et F_n de Shoup jusqu'au degré 500[4]. Ces résultats, présentés à la conférence ISSAC en juillet [130], ont montré l'efficacité de l'algorithme de Shoup pour la factorisation dans $\mathbf{Z}_p[x]$.

Nous avons ensuite étudié la factorisation de polynômes à coefficients entiers ($\mathbf{Z}[x]$). C'est en effet un outil de base pour la résolution des systèmes polynomiaux, permettant de scinder les ensembles triangulaires obtenus après calcul de bases de Gröbner. Avec une certaine surprise, nous avons constaté que les systèmes de calcul formel tels que Maple ou Mathematica ne peuvent pas factoriser de polynômes véritablement "gros". Nous avons ensuite optimisé l'implantation de l'algorithme de Zassenhaus en MuPAD, ce qui en fait avec MAGMA le système le plus performant à l'heure actuelle pour la factorisation de polynômes (cf. `http://www.loria.fr/~zimmerma/mupad`).

Systèmes polynomiaux

L'étude des zéros réels des systèmes polynomiaux constitue un lien direct entre le calcul formel et les applications. Dans le cadre du projet Européen Frisco LTR 21.042, notre but est de fournir des outils performants en vue d'une utilisation industrielle. Ce travail comporte plusieurs aspects :

[4]Cf. `http://www.loria.fr/~zimmerma/papers`

- Un aspect théorique en géométrie réelle pour l'élaboration d'algorithmes effectifs : méthodes de comptage avec ou sans contraintes, simplification des systèmes, isolation des racines, traitement des hypersurfaces.
- Un aspect algorithmique pour l'optimisation pratique des méthodes effectives : diminution du nombre d'opérations arithmétiques, contrôle de la taille des données en cours de calcul.
- Un aspect informatique pour l'implantation efficace de ces méthodes sur un grand nombre de plateformes.

Nous avons mis un accent particulier sur l'étude des zéros réels des systèmes admettant un nombre fini de solutions complexes. L'outil de base est la Représentation Univariée Rationnelle (RUR) permettant de ramener tout problème à l'étude d'un unique polynôme en une variable. Notre bibliothèque C++ (RealSolving) a atteint un niveau de performance élevé et l'accent est mis maintenant sur le calcul de la RUR ainsi que sur le traitement de la sortie de cet algorithme, pour augmenter la classe des exemples que l'on peut traiter. Plusieurs stratégies sont à l'étude :

- Parallélisation des calculs, pour une implantation sur les machines du centre Charles Hermite,
- Utilisation d'arithmétiques hybrides (calculs simultanés modulaires et flottants multi-précision) pour éviter la croissance des coefficients,
- Stratégies de type multi-modulaire dans le cas des systèmes à coefficients entiers ou polynomiaux.

Quelques applications pratiques ont vu le jour, notamment en compression d'image par le calcul de nouveaux filtres (collaboration avec le CNET) ou en robotique (étude de robots parallèles - collaboration avec le projet PRISME).

3.2 Preuve

3.2.1 Application de la déduction automatique aux protocoles de communication

Participants : Otmane Aït Mohamed, Djamila Bouzid, Boutheïna Chetali, Barbara Heyd, Pierre Lescanne

Mots-clés : π-calcul, parallélisme, bisimulation, vérification de programme, preuve de programme, UNITY, protocoles.

Les travaux sur la preuve portent sur la valorisation d'un savoir-faire autour de problèmes concrets, notamment autour des réseaux, mais aussi sur des problèmes plus fondamentaux de complexité, de théorie de la démonstration et d'applications du lambda-calcul à la programmation fonctionnelle et à la démonstration automatique. Pour le projet Euréca, la recherche fondamentale en logique constitue le complément nécessaire des développements applicatifs ; en effet, la maîtrise de logiciels comme HOL, COQ ou LP requièrent des compétences en logique conduisant à des interactions fortes entre chercheurs.

Nous avons acquis au fil des ans un savoir-faire dans le domaine de la déduction automatique que l'on applique actuellement à la preuve de propriétés de systèmes informatiques. Nous nous sommes intéressés plus particulièrement aux protocoles de communication. En effet, la vérification formelle des protocoles de communication est une tâche complexe et sujette à des erreurs. Cette complexité est due au fait que les protocoles eux-mêmes sont complexes et exhibent des comportements compliqués. L'utilisation d'un outil de preuve pour vérifier la correction des preuves est donc nécessaire pour le succès d'une telle tâche.

Notre collaboration avec Bull dans le cadre du GIE Dyade et avec le CNET nous a amené à considérer un protocole fonctionnant sur Internet et un autre sur ATM (*Asynchronous Transfert Mode*).

Coq-Unity

Dans la lignée du travail réalisé au cours des années précédentes, nous avons continué notre investigation dans le cadre de la vérification formelle des programmes concurrents spécifiés en UNITY. Cette recherche se divise toujours en deux parties :

La première partie concerne le codage de la logique UNITY en COQ. Cette logique a été formalisée suivant le modèle état-transition proposé par Chandy et Misra. Cependant B. Sanders a démontré que la logique d'UNITY avec l'axiome de substitution n'est pas consistante et que sans cet axiome, elle est incomplète. En s'appuyant sur ses travaux, sur ceux de Misra et en collaboration avec Pierre Crégut (CNET-Lannion), nous avons proposé une nouvelle logique UNITY. Cette nouvelle logique introduit la notion d'environnement de programmes et définit de nouveaux opérateurs tenant compte de cet environnement. La complétude et la correction de cette nouvelle logique sont démontrées dans [120] et ces résultats ont été présentés à la conférence TPHOL (*Theorem Proving in Higher Order Logic*), qui s'est tenue en Finlande en août 1996.

La seconde partie s'intéresse à l'application à divers exemples de l'outil formel développé. Après des exemples d'école comme la division euclidienne ou le problème des lecteurs/rédacteurs, la vérification d'un contrôleur d'ascenseur a été abordée. Cet exemple étant déjà démontré en HOL-UNITY, l'intérêt de l'exercice a été de déterminer la méthodologie des preuves mais aussi d'en améliorer l'automatisation par la création de tactiques. De plus, voulant mettre à profit la notion d'environnement et les nouveaux opérateurs de la logique, l'expérience s'est prolongée pour la composition de deux ascenseurs. Par ailleurs, nous avons étudié avec P. Crégut la vérification formelle d'un protocole développé au CNET, l'*ABT/DT ATM Block Transfert/Delayed Transmission*, et nous avons participé à l'élaboration de la spécification du protocole. L'article [119], présenté à la conférence ICNP (*International Conference Network Protocol*), qui s'est tenue aux États-Unis du 29 octobre au 1er novembre 1996, donne les points importants de ce protocole. Ce protocole est en cours de normalisation à l'UIT-T (Union Internationale des Télécommunications, secteur des Télécommunications, ex Comité Consultatif International pour le Télégraphe et le Téléphone). Le codage du protocole dans l'outil formel étant achevé, la démonstration des propriétés est en cours.

Vérification formelle à l'aide de LP

Cette année nous a permis de faire la synthèse [110] de nos travaux sur la vérification des programmes parallèles décrits en UNITY à l'aide de LP (démonstrateur automatique du LARCH). En effet, nous avons poursuivi la formalisation de la logique d'UNITY et de son système de preuve. Pour cela, nous avons achevé la construction d'une base de faits représentant les données de base sur lesquelles porte le raisonnement. Cette base de faits est utilisée dans tous les programmes UNITY et plus généralement, elle est indépendante du modèle UNITY. Par la suite, nous avons effectué des améliorations en introduisant le quantificateur existentiel, qui permet une formalisation plus intuitive et plus proche de la philosophie UNITY. Ceci nous a permis d'améliorer la formalisation des relations de progrès et en particulier celle de l'opérateur temporel *leads_to* et notamment de prouver les règles d'inférence associées, à l'aide d'un schéma d'induction basé sur une relation nœthérienne. Ce travail nous a permis d'illustrer l'apport de la quantification existentielle pour l'expressivité du langage, de montrer que les formalisations ne sont pas figées et enfin d'étudier la mécanisation des preuves manipulant les quantificateurs existentiels. Enfin, nous avons proposé une méthodologie de spécification et de vérification à l'aide de LP, basée sur notre expérience et sur les preuves effectuées. En particulier, nous avons proposé une méthode incrémentale pour la spécification d'une base de travail ainsi que des principes de gestion et d'organisation des preuves complexes. Par ailleurs, nous poursuivons notre collaboration avec D. Bolignano, amorcée en 1993, où l'intérêt porté aux méthodes formelles se concentre plutôt sur les preuves concises et lisibles que sur l'automatisation des preuves. Cette collaboration s'effectue dans le cadre du GIE DYADE, en particulier dans le groupe VIP, où l'objectif serait l'utilisation du démonstrateur COQ pour la vérification protocoles cryptographiques.

Internet et VIP

Le projet EURÉCA a participé à l'opération VIP (*Verified Internet Protocols*) du GIE DYADE. La tâche qui lui a été confiée consistait à examiner les aspects « preuves » des protocoles Internet IPv4 et IPv6. L'étude s'est portée sur les aspects *fragmentation* et *réassemblage* de ces protocoles et une spécification de ces fonctionnalités a été fournie.

Outil de preuve pour le π-calcul

Notre travail [109] pour cette année s'est focalisé sur l'étude et la vérification du protocole de communication du "*Handover*" en utilisant notre outil de preuve PIC. Le protocole du "*Handover*" est proposé par l'Institut Européen de Télécommunications (ETSI) pour être utilisé dans le GSM *Public Land Mobile Network* (PLMN). Ce protocole est spécifié dans le π-calcul par un ensemble d'équations récursives et mutuellement récursives. Pour le représenter dans PIC, il a fallu simuler ces équations récursives et mutuellement récursives par le seul opérateur **Rec** disponible dans la théorie du π-calcul formalisée dans HOL. La preuve que nous avons développée interactivement dans PIC consistait à montrer qu'une certaine relation donnée est une bisimulation à une équivalence forte près. Une analyse détaillée du protocole permet de déterminer cette relation. Ensuite, l'environnement de preuve nous assiste dans la construction de la preuve souhaitée. La technique de bisimulation à équivalence près est très puissante et permet un raisonnement sur des relations *a priori* infinies.

3.2.2 Études des substitutions explicites et de leurs applications

Participants : Zine-El-Abidine Benaissa, Daniel Briaud, Frédéric Lang, Pierre Lescanne, Kristoffer Rose

Mots-clés : lambda-calcul, réécriture.

Les calculs de *substitutions explicites* sont des calculs qui intègrent l'opération fondamentale du lambda-calcul qu'est la substitution à l'intérieur du lambda-calcul lui-même. Ce sont des outils puissants pour la compréhension du lambda-calcul et de la *programmation fonctionnelle*. Après avoir investi dans les domaines fondamentaux des substitutions explicites[111, 123, 124], nous avons, cette année, étudié leurs applications, notamment dans le cadre de l'unification d'ordre supérieur et de la description d'implantations.

Unification d'ordre supérieur

Cette partie concerne l'utilisation de $\lambda\upsilon$ (lire *lambda upsilon*) pour analyser l'unification d'ordre supérieur. $\lambda\upsilon$ est un système de réécriture à la fois terminant et confluent sur les termes clos simplement typés. Ces deux propriétés conduisent à employer la surréduction pour résoudre des équations dans la théorie engendrée par $\lambda\upsilon$, autrement dit, pour résoudre des problèmes de $\lambda\upsilon$-unification. Précisément, il s'agit d'une surréduction typée dont la cohérence et la complétude se démontrent de manière usuelle.

Disposant d'une méthode élémentaire de $\lambda\upsilon$-unification, l'étape suivante consiste à coder un problème d'unification d'ordre supérieur en un problème de $\lambda\upsilon$-unification, d'une part, et à traduire les $\lambda\upsilon$-unifieurs fournis par surréduction typée, en unifieurs d'ordre supérieur, d'autre part. La difficulté réside dans cette dernière phase. Nous avons défini des procédures de codage et de décodage assez simples et montré leur cohérence ainsi que leur complétude. Ce travail a été présenté au *workshop Unif'96* [117].

Afin de finaliser cette recherche, il reste à tirer parti de cette analyse de l'unification d'ordre supérieur. Cela signifie affiner la surréduction typée en vue d'obtenir une procédure de pré-unification. Ce travail est en cours.

Description d'implantations

En utilisant les calculs de substitutions explicites, nous avons élaboré un cadre formel pour décrire certains aspects des implantations du lambda-calcul et en particulier les calculs faibles qui sont à la base des compilateurs de langages fonctionnels. Ceci nous a permis de raisonner sur ces implantations en prouvant par exemple leur correction ou leur bon comportement vis-à-vis de l'espace mémoire occupé par ces programmes. Grâce à cette modélisation nous avons synthétisé la plupart des compilateurs existants. De plus, ce modèle permet d'étudier la complexité des programmes fonctionnels qui est peu abordée par la communauté des chercheurs.

Substitutions explicites et réécriture d'ordre supérieur

Kristoffer Rose, en collaboration avec Roel Bloo, de l'Université Technique d'Eindhoven, a étudié la généralisation des substitutions explicites à la réécriture d'ordre supérieur et a décrit comment les « systèmes de réduction combinatoire » de Klop sont transformables automatiquement en systèmes de réécriture d'ordre supérieur munis d'une substitution « explicite ». Ils ont prouvé que les systèmes de réduction combinatoire qui satisfont une condition de préservation des structures préservent aussi les normalisations fortes [116].

3.2.3 Complexité de la déduction automatique

Participant : Nicolas Hermann, Laurent Juban

Complexité de comptage en satisfaisabilité généralisée

Les preuves de borne inférieure pour les différents problèmes fonctionnels, c'est-à-dire les problèmes dont la réponse est une chaîne de caractères ou un entier naturel au lieu d'une valeur booléenne sont en général des preuves par réduction « parcimonieuse » à partir d'un problème de satisfaisabilité généralisée. Dans les problèmes de satisfaisabilité généralisée, on considère les formules propositionnelles en forme normale conjonctive, dont les clauses peuvent être formées par une relation logique quelconque. La classe de complexité #P contient les fonctions qui calculent le nombre de solutions des problèmes dans la classe NP. N. Hermann et N. Creignou ont démontré un théorème dichotomique pour la classe #P [113]. En effet, d'après ce théorème, la connaissance du nombre de valuations satisfaisables d'une formule propositionnelle généralisée peut être déterminée en temps polynomial déterministe uniquement si toutes les relations logiques utilisées sont affines, plus précisément si elles font partie d'un anneau booléen. Dans ce cas, on peut utiliser la transformation de Gauss d'un système d'équations pour déterminer le nombre de solutions. Dans tous les autres cas, le problème de comptage est #P-complet. Un corollaire direct de ce résultat s'énonce ainsi « Si un problème de décision en satisfaisabilité est NP-complet alors le problème correspondant de comptage est #P-complet ». Le résultat principal de ce travail a déjà été utilisé, entre autres, pour prouver les bornes inférieures de problèmes de comptage en filtrage équationnel dans l'article [114].

Impossibilité d'une méthode de combinaison polynomiale pour les algorithmes d'unification

L'unification équationnelle, c'est-à-dire la résolution d'équations en présence d'égalités, est omniprésente en déduction automatique. Or, il existe des familles de théories pour lesquelles on connaît des algorithmes d'unification, sans que l'on connaisse un algorithme d'unification pour leur réunion (mélange) alors que les signatures des théories sont disjointes, ce qui pourrait faire espérer que la combinaison des algorithmes d'unification soit facile. Comme ce cas est relativement fréquent, des méthodes pratiques de combinaison d'algorithmes d'unifications ont été présentées à la fin des années 80 et au début des années 90. Toutes ces méthodes sont de complexité exponentielle. La question se posait alors

de savoir si le problème de la combinaison d'algorithmes d'unification est de complexité intrinsèquement non polynomiale. N. Hermann et P. G. Kolaitis ont démontré [122] qu'une méthode générale de combinaison en temps polynomial est impossible.

Nous nous sommes intéressés à l'étude de l'élimination des coupures dans le calcul classique des séquents en nous appuyant sur le théorème de Gentzen (1935).

L'approche que nous avons adoptée consiste à utiliser des techniques standards de réécriture. Nous avons représenté les preuves par des termes, puis nous avons construit un système de réécriture réalisant la normalisation des preuves. La preuve de terminaison de ce système de réécriture établit alors la propriété de normalisation forte pour le calcul classique des séquents. Cette nouvelle approche a produit une démonstration générique applicable à différentes logiques. En effet, nous avons étendu ce résultat aux calculs intuitionniste ($LJ1$) et linéaire ($MALL1$) des séquents en suivant la même approche. Ce travail a été accepté pour être publié dans *Theoretical Computer Science* [112].

Nous avons utilisé l'approche décrite ci-dessus pour produire une preuve d'élimination forte des coupures pour la logique linéaire (LL), partant de la preuve de D. Roorda.

Ce travail ayant été achevé cette année, nous avons commencé l'étude de la classe de fonctions numériques du lambda-calcul polymorphe (système F) ou les fonctions prouvablement totales dans $PA2$, un système d'arithmétique de second ordre, afin d'obtenir des caractérisations de complexité.

3.2.4 Terminaison, théorie des ordinaux

Participants : Adam Cichon, Hélène Touzet, Guillaume Bonfante

Mots-clés : réécriture, logique, ordinaux.

Pour établir la terminaison d'un système de réécriture, une méthode générale consiste à définir des fonctions d'interprétation. Le problème de cette approche sémantique est qu'il n'existe pas de méthode déterministe pour produire ces interprétations.

Dans cette perspective, en nous basant sur des travaux précédents, afin de mieux exploiter les hiérarchies sous-récursives de fonctions indexées par des ordinaux pour déterminer de telles fonctions, nous avons défini un système de *termes ordinaux*, définis sur la signature $\{1, s, +, *, \exp\}$ en gérant la construction des ordinaux limites par un opérateur unaire Δ.

Établir la terminaison de certains systèmes de réécriture revient alors à résoudre une équation ordinale générée directement à partir des règles du système. Cela s'apparente à la résolution d'une équation récurrente sur les entiers.

Ce travail a été publié dans les actes de la conférence *CAAP* en avril 1996 [118].

Notre but, avec Guillaume Bonfante – récemment arrivé dans l'équipe, est d'étendre ces méthodes pour pouvoir les appliquer dans le domaine des logiques formelles, notamment la logique linéaire, afin d'améliorer nos connaissances sur la récursivité et le raisonnement par récurrence.

4 Actions industrielles

4.1 Collaboration avec le CNET sur les formalisations et la preuve de protocoles

Participante : Barbara Heyd

Le protocole ABT/DT a été étudié dans le cadre d'une collaboration avec le CNET-Lannion décrite au paragraphe 3.2.1.

4.2 VIP et DYADE

Participants : Djamila Bouzid, Pierre Lescanne, Nicolas Hermann

Dans le cadre de l'opération VIP *(Verified Internet Protocols)* du GIE DYADE, nous avons participé à la formalisation de certains aspects du protocole Internet IP, pour en étudier la vérification par preuve (voir paragraphe 3.2.1).

5 Actions nationales et internationales

5.1 Actions nationales

P. Lescanne est directeur du *Centre Charles Hermite* et a été directeur du *GDR Programmation* jusqu'en juin 96.

P. Zimmermann a participé à l'organisation des journées scientifiques du Centre Charles Hermite.

G. Kucherov est membre du comité de rédaction de la lettre de l'AFIT (Association Française d'Informatique Théorique).

N. Hermann a collaboré avec N. Creignou, maître de conférences à l'Université de Caen, sur le sujet de la complexité des problèmes de comptage.

5.2 Actions internationales

P. Lescanne est membre du comité éditorial du journal *Applicable Algebra in Engineering, Communication and Computation* édité par Springer-Verlag, et membre du comité de programme de la conférence *Rewriting Techniques and Applications 96*. Il est aussi membre de projet européen BASIC RESEARCH appelé COMPASS qui étudie la spécification de systèmes informatiques.

5.2.1 Europe

N. Hermann a collaboré avec l'équipe du professeur Alexandre Leitsch au Département d'Informatique à l'Université Technique de Vienne, en Autriche. En particulier, il a poursuivi sa recherche sur les schématisations avec Gernot Salzer et a donné un cours à l'Université Technique de Vienne pour les étudiants de maîtrise et de DEA. Il a été invité à donner un séminaire sur « l'impossibilité d'une méthode polynomiale de combinaison des algorithmes d'unification équationnelle » à l'Université Technique de Vienne, dans le cadre du Séminaire Kurt Gödel. Il a par ailleurs exposé ce travail au LIP à l'ENS de Lyon, ainsi qu'au Département d'Informatique à l'Université de Bratislava, en Slovaquie.

Z. Benaissa a été invité pour un séjour d'une semaine au BRICS (Basic Research in Computer Science, Danemark) où il a présenté le séminaire « Non-determinism, Explicit Substitutions and Sharing ».

G. Kucherov continue sa collaboration avec ses collègues allemands, D. Hofbauer de l'Université Technique de Berlin et M. Huber de l'Université de Kassel. Une visite à Berlin est prévue au mois de décembre.

P. Zimmermann a déposé un projet d'action intégrée PROCOPE avec l'Université de Paderborn (Allemagne) qui a été accepté.

P. Lescanne a donné une conférence invitée à APPIA-GULP-PRODE'96 qui rassemble les communautés latines (espagnole, italienne et portugaise) en programmation fonctionnelle et logique.

Le Projet Européen Frisco LTR 21.042. Le projet Frisco (*a FRamework for Integrating Symbolic/numeric COmputation*) a pour but de chercher et de développer de nouvelles technologies pour élaborer des outils logiciels puissants pour le traitement et la résolution des systèmes polynomiaux.

Les principaux contractants sont le CNRS (IRMAR-Rennes), l'INRIA (Sophia-Antipolis), NAG (Angleterre), l'université de Pise (Italie) et l'université de Santander (Espagne). Le projet EURÉCA est en passe de devenir sous-contractant.

F. Rouillier et P. Zimmermann participent au projet Frisco dans les parties *algorithmique*, *implantations*, *évaluation des logiciels* et *besoins de l'industrie*.

À ce titre, F. Rouillier a présenté un nouveau résultat pour la synthèse de bancs de filtres (travail effectué en coopération avec J.-C. Faugère, CNRS-Paris VI, et Francois Moreau, CCETT-CNET) lors du *workshops Frisco: Needs of industry* à Barcelone en octobre 1996.

G. Kucherov collabore avec des chercheurs de l'Université de Moscou (A. Ugolnikov, R. Kolpakov) dans le cadre de l'Institut Liapunov franco-russe d'informatique et de mathématiques appliquées (voir section 3.1.1). La constitution d'un projet de l'Institut Liapunov est envisagée.

5.2.2 Afrique

P. Lescanne a participé à la *Conférence Africaine sur la Recherche en Informatique* CARI à Libreville au Gabon et y a donné une communication.

5.2.3 Amérique

N. Hermann a collaboré avec le professeur P. G. Kolaitis à l'Université de Californie à Santa Cruz sur le sujet de la complexité en déduction automatique.

5.3 Actions d'enseignement

N. Hermann a assuré un cours d'un semestre (printemps 1996) sur l'unification, la réécriture et la déduction automatique au Département d'Informatique à l'Université Technique de Vienne en Autriche.

A. Cichon a assuré la partie logique du cours de « Mathématiques pour l'Informatique » de la Licence d'informatique de l'Université Henri Poincaré Nancy 1, et de la première année de l'École Supérieure d'Informatique et Applications de Lorraine (ESIAL), des travaux dirigés de programmation logique en DESS, des travaux dirigés de spécification en DESS troisième année de l'ESIAL et le cours *Logique 2* du DEA d'informatique de l'UHP. Il est en outre responsable du DESS Informatique de l'Université Henri Poincaré Nancy 1.

G. Kucherov a enseigné le cours de DEA d'Informatique *Complexité et analyse d'algorithmes* (module de spécialisation) à l'Université de Nancy 1.

5.4 Participation à des colloques

Des membres de l'équipe ont participé à des conférences et *workshops* ; on se reportera à la bibliographie pour en avoir la liste.

GDR-PRC AMI. Les membres du projet participent à plusieurs groupes de travail du GDR-PRC AMI : Z. Benaissa et P. Lescanne ont participé aux journées de Dijon, ainsi qu'au groupe de travail λ-calcul et réécriture à Paris, et au colloque logique et modèles de calcul à Marseille. P. Zimmermann a participé aux journées ALÉA à Biviers, où F. Bertault a présenté le logiciel `CGraph`, et V. Grébinski et G. Kucherov leurs travaux sur la combinatoire pour le génome, qu'ils ont également exposés au groupe de travail « Informatique et Génome ».

N. Hermann, G. Kucherov et P. Lescanne ont participé à la réunion de conférences FLOC (CADE+CAV+LICS+RTA) où N. Hermann a présenté l'article [122] à la conférence CADE'96.

Z. Benaissa a présenté l'article [121] à la conférence PLILP-ALP à Aachen (Allemagne).

Les travaux menés par B. Heyd en collaboration avec P. Crégut du CNET Lannion ont été présentés lors des conférences *Theorem Proving in Higher Order Logic* en Finlande et *International Conference Network Protocol* aux États-Unis.

F. Bertault a participé au concours de tracé de graphes associé à la conférence Graph Drawing'96, qui s'est tenue à Berkeley, Californie, et s'est vu attribuer la mention honorable dans deux catégories, pour des graphes réalisés avec le logiciel `CGraph`. Ces graphes figureront dans les comptes-rendus de la conférence.

P. Zimmermann a présenté une comparaison des systèmes de calcul formel actuels pour la résolution d'équations différentielles ordinaires au *5th Rhine Workshop on Computer Algebra* qui s'est tenu à Saint-Louis début avril [125]. En juillet, il a également participé à la conférence *Applications of Computer Algebra* à Linz, où il a donné son point de vue sur les systèmes de calcul formel commerciaux et domaine public, puis à la conférence ISSAC à Zürich, où il a présenté les résultats obtenus sur la factorisation de polynômes [130].

A. Cichon a donné un séminaire au Laboratoire de Logique, Algorithmique et Informatique de Clermont 1, en mars 1996.

S. Selhab a présenté un séminaire dans le cadre du projet PROCOPE, issu d'une coopération avec un groupe de recherche du département d'Informatique de l'Université Technique de Berlin, dirigé par D. Siefkes.

5.5 Conférences invitées, tutoriels, cours, etc.

F. Bertault a réalisé un support de cours pour l'enseignement de Maple en classes préparatoires scientifiques [126].

P. Zimmermann a été invité par M. Mignotte (Strasbourg) à faire un exposé sur le système de calcul formel MUPAD. Il a aussi fait un exposé sur les « mathématiques inverses » dans le groupe de J.-L. Nicolas à Lyon, et au séminaire de théorie des nombres de G. Ténenbaum à l'IECN.

5.6 Animations scientifiques

P. Lescanne a publié un article sur l'histoire de l'informatique en Lorraine dans l'Encyclopédie Illustrée de la Lorraine, Histoire des Sciences et des Techniques, Volume *Sciences exactes* [115].

5.7 Diffusion de logiciels

Les logiciels `CGraph`, `grappe` et `Designer` ont été mis dans le domaine public (accessibles sur la page WWW du projet[5]). `Cgraph` est un logiciel de tracé de graphes. `grappe` est un logiciel de recherche de motifs généraux dans les textes (voir rapport d'activités 1995). `Designer` est un programme Maple implantant un algorithme combinatoire de couverture d'ensembles finis (voir section 3.1.1).

6 Publications

Thèses

[109] O. AÏT MOHAMED, *La théorie du π-calcul dans le système HOL*, thèse de doctorat, Université Henri Poincaré Nancy I, juillet 1996.

[110] B. CHETALI, *Vérification Formelle des Systèmes Parallèles décrits en UNITY à l'aide d'un outil de Démonstration Automatique*, thèse de doctorat, Université Henri Poincaré Nancy I, mai 1996.

[5] `http://www.loria.fr/exterieur/equipe/eureca/Logiciels/logiciels.html`

Articles et chapitres de livre

[111] Z. BENAISSA, D. BRIAUD, P. LESCANNE, J. ROUYER-DEGLI, «$\lambda\upsilon$, a calculus of explicit substitutions which preserves strong normalisation», *Journal of Functional Programming 6*, 5, septembre 1996, (paru aussi comme rapport de recherche Inria n°2477).

[112] E. CICHON, M. RUSINOWITCH, S. SELHAB, «Cut Elimination and Rewriting: Termination proofs», *Theoretical Computer Science B*, à paraître.

[113] N. CREIGNOU, M. HERMANN, «Complexity of generalized satisfiability counting problems», *Information and Computation 125*, 1, 1996, p. 1–12.

[114] M. HERMANN, P. KOLAITIS, «The complexity of counting problems in equational matching», *Journal of Symbolic Computation 20*, 3, 1995, p. 343–362.

[115] P. LESCANNE, *Encyclopédie Illustrée de la Lorraine*, *Sciences exactes*, Éditions Serpenoises, 1996, ch. La science informatique en Lorraine.

Communications à des congrès, colloques, etc.

[116] R. BLOO, K. H. ROSE, «Combinatory Reduction Systems with Explicit Substitution that Preserve Strong Normalisation», *in : 7th RTA—Rewriting Techniques and Applications*, H. Ganzinger (réd.), *Lecture Notes in Computer Science*, 1103, Rutgers University, Springer-Verlag, p. 169–183, New Brunswick, New Jersey, juillet 1996, http://www.brics.dk/~krisrose/PAPERS/bloo+rose-rta96.ps.gz.

[117] D. BRIAUD, «Higher Order Unification as a Typed Narrowing», *in : Proceedings of UNIF'96*, K. Schulz, S. Kepser (réd.), CIS-Universität München, p. 131–138, 1996.

[118] A. CICHON, H. TOUZET, «An Ordinal Calculus for Proving Termination in Term Rewriting», *in : Int. CAAP-Coll.on Trees in Algebra and Programming*, H. Kirchner (réd.), *Lecture Notes in Computer Science*, *1059*, Springer-Verlag, p. 226–240, 1996.

[119] P. CRÉGUT, F. GUILLEMIN, B. HEYD, «A Protocol for Supporting the ATM Block Transfer with Delayed Transmission Capability», *in : International Conference on Network Protocols'96. sujet : ATM Protocols*, 1996.

[120] P. CRÉGUT, B. HEYD, «A modular coding of Unity in Coq», *in : Theorem Proving in Higher Order Logic*, J. von Wright, J. Grundy, J. Harrison (réd.), *Lecture Notes in Computer Science*, *1125*, Springer-Verlag, p. 251–266, Turku, Finland, août 1996.

[121] Z. E. A. BENAISSA, K. H. ROSE, P. LESCANNE, «Modeling Sharing and Recursion for Weak Reduction Strategies using Explicit Substitution», *in : 8th PLILP—Symposium on Programming Language Implementation and Logic Programming*, H. Kuchen, D. Swierstra (réd.), *Lecture Notes in Computer Science*, 1140, Springer-Verlag, p. 393–407, Aachen, Germany, septembre 1996, http://www.loria.fr/~kris/benaissa+lescanne+rose-96.ps.gz.

[122] M. HERMANN, P. KOLAITIS, «Unification algorithms cannot be combined in polynomial time», *in : Proceedings 13th International Conference on Automated Deduction, New Brunswick NY (USA)*, M. McRobbie, J. Slaney (réd.), *Lecture Notes in Artificial Intelligence*, *1104*, Springer-Verlag, p. 246–260, juillet/août 1996.

[123] P. LESCANNE, Z.-E.-A. BENAISSA, D. BRIAUD, «Calculi of explicit substitutions: new results», *in : Proceedings of the conference APPIA-GULP-PRODE'96*, p. 309–326, juillet 1996.

[124] P. LESCANNE, «Programmation fonctionnelle et substitutions explicites», *in : Proc. 3e Colloque Africaine de Recherche en Informatique, CARI'96*, P. Mokeli (réd.), Ortsom Éditions, octobre 1996.

[125] F. POSTEL, P. ZIMMERMANN, «A review of the ODE solvers of Axiom, Derive, Maple, Mathematica, Macsyma, MuPAD and Reduce», *in : Proceedings of the 5th RHINE workshop on computer algebra*, A. C. et L. R. Oudin (réd.), p. 2.1–2.10, Saint-Louis (France), avril 1996, http://www.loria.fr/~zimmerma/papers.

Rapports de recherche et publications internes

[126] F. BERTAULT, «Maple : résumé de cours et exercices en sciences physiques», *rapport de recherche n°RR-2936*, Inria Lorraine, 1996.

Divers

[127] P. DEVIENNE, P. LESCANNE, «Rapport d'activités du GDR-PRC Programmation», janvier 1996.

[128] V. GRÉBINSKI, G. KUCHEROV, «Reconstructing a Hamiltonian circuit by queuering the graph: Application to DNA physical mapping», Soumis à Discrete Applied Mathematics, 1996.

[129] R. KOLPAKOV, G. KUCHEROV, «Minimal letter frequence in n-th power-free binary words», Soumis au Colloquium on Trees in Algebra and Programming (TAPSOFT'97), 1996.

[130] P. ZIMMERMANN, L. BERNARDIN, M. MONAGAN, «Polynomial Factorization Challenges», Poster à ISSAC'96, juillet 1996, http://www.loria.fr/~zimmerma/papers.

7 Abstract

Research in Computer Science in Nancy is conducted jointly under the auspices of two institutions: (1) the CRIN (Centre de Recherche en Informatique de Nancy), a laboratory associated with the CNRS and comprising Nancy's main university teaching establishments, and (2) the INRIA (Institut Nationale de Recherche en Informatique et Automatique). Researchers work in formal groups. *Eureca* is the group headed by Pierre Lescanne, and comprises research and teaching staff employed, in particular, by the CNRS, the University and the INRIA-Lorraine. In 1996 *Eureca* comprised :–

Group Secretary: Céline Cordier

CNRS: Pierre Lescanne [head of *Eureca*], Nicolas Hermann, Jean-Luc Rémy

INRIA - Lorraine: Grégory Kucherov, Paul Zimmermann, Fabrice Rouillier

University Henri Poincaré, Nancy 1: Adam Cichon, Jocelyne Rouyer, Joseph Rouyer, Zine-El-Abidine Benaissa, Daniel Briaud, Boutheina Chetali, Hélène Touzet

Visiting members of Eureca: Sapphorain Pfermann, Kristoffer Høgsbro Rose

Doctoral students: François Bertault, Guillaume Bonfante, Djamilla Bouzid, Vladimir Grebinski, Barbara Heyd, Frédéric Lang, Otmane Aït Mohamed, Sohame Selhab

The group's research activities come under the banner : *Proof, Symbolic Computation and Logic*. There are two main lines of research:

1. Algorithmics

This concerns the study and creation of algorithms for symbolic computation. Research is conducted into both the theoretical properties and complexity of algorithms and applications to the study of genetic codes.

2. Proof and Logic

Here there are two sub-divisions into the field of *formal verification of systems* and *automated deduction*. Research has been conducted into the soundness of protocols and into the developement of proof environments for UNITY. Theoretical activity includes research in Proof Theory, the λ-calculus and complexity related to filtering and unification.