INRIA

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

Projet LEMME

Logiciels et Mathématiques

Sophia Antipolis

THÈME 2A

Table des matières

1 Composition de l'équipe **3**

2 Présentation et objectifs généraux **4**

3 Fondements scientifiques **4**
3.1 Environnements de preuves 4
3.2 Théorie des types et formalisation de théories mathématiques 4
3.3 Implémentations certifiées d'algorithmes de calcul scientifique 4
3.4 Sémantique des langages de programmation 4

4 Domaines d'applications **5**
4.1 Cartes à puces 5
4.2 Algorithmes certifiés 5
4.3 Web, MathML, XML 5

5 Logiciels **6**
5.1 Pcoq 6
5.2 Aïoli et Figue 7
5.3 Algorithme de Stålmarck 7

6 Résultats nouveaux **7**
6.1 Outils pour les environnements de preuve 7
6.1.1 Evolution du système Pcoq 7
6.1.2 Affichage de formules mathématiques 8
6.1.3 Visualisateur de théories mathématiques Coq en SVG. 9
6.2 Théorie des types et formalisation de théories mathématiques 9
6.2.1 Fonctions récursives 9
6.2.2 Changements de structure de données et réutilisation de démonstrations 9
6.2.3 Logique 10
6.2.4 Quotients 10
6.3 Certification d'algorithmes 10
6.3.1 Élimination des quantificateurs dans les réels, et tactique en Coq 10
6.3.2 Intégration d'un vérificateur de tautologie dans le système Coq. 12
6.3.3 Extension de la formalisation des flottants dans Coq. 12
6.3.4 Certification d'algorithmes d'enveloppe convexe 12
6.3.5 Algorithmes arithmétiques sur les grands entiers 12
6.3.6 Représentation des nombres rationnels 13
6.4 Sémantique des langages de programmation 13
6.4.1 Description sémantique des langages 13
6.4.2 Langages de spécification 13
6.4.3 Le langage JavaCard 14
6.4.4 Environnement de vérification pour JavaCard 14
6.4.5 Vérification du code mobile et embarqué 15

7 Contrats industriels (nationaux, européens et internationaux) 15
7.1 Dassault Aviation . . . 15
7.2 Mowgli . . . 15
7.3 Verificard . . . 15

8 Actions régionales, nationales et internationales 15
8.1 Actions internationales . . . 15
8.1.1 Collaboration avec l'ORCCA . . . 15
8.1.2 Collaboration avec ICM . . . 16
8.1.3 Divers . . . 16
8.2 Actions nationales . . . 16
8.2.1 Action de Recherche Coopérative AOC. . . . 16
8.2.2 Action de Recherche Coopérative S-Java. . . . 16
8.2.3 Divers . . . 17
8.3 Actions européennes . . . 17
8.3.1 Réseau Types . . . 17

9 Diffusion de résultats 17
9.1 Manifestations scientifiques, missions . . . 17
9.2 Animation de la communauté scientifique . . . 18
9.3 Divers . . . 18
9.4 Visites . . . 18
9.5 Direction de thèses . . . 19
9.6 Jury de thèses . . . 19
9.7 Encadrement de stagiaires . . . 19
9.8 Enseignement . . . 19

10 Bibliographie 20

1 Composition de l'équipe

Responsable scientifique
Loïc Pottier [Chargé de recherche INRIA]
Responsable permanent
Yves Bertot [Chargé de recherche INRIA, HDR]
Assistante de projet
Nathalie Bellesso
Personnel Inria
Gilles Barthe [Chargé de recherche INRIA]
Marieke Huisman [Chargée de recherche INRIA depuis octobre, post-doctorante avant]
Francis Montagnac [Ingénieur de Recherche INRIA]
Laurence Rideau [Chargée de recherche INRIA]
Laurent Théry [Chargé de recherche INRIA]
Fonctionnaires en délégation
Philippe Audebaud [Maître de conférence, ENS Lyon, depuis décembre]
Solange Coupet-Grimal [Maître de conférence, Marseille, depuis octobre]
Chercheurs post-doctorants
Didier Bondyfalat [action AOC]
Venanzio Capretta [depuis novembre, bourse CEE]
Pierre Courtieu [depuis septembre, bourse JavaCard]
Sorin Stratulat [depuis octobre, bourse JavaCard]
Collaborateurs extérieurs
Frédérique Guilhot [Professeur agrégé]
André Hirschowitz [Professeur UNSA, Laboratoire J.A.Dieudonné]
Roger Marlin [Professeur UNSA, Laboratoire J.A.Dieudonné]
Monica Nesi [Maître de Conférences, Université de L'Aquila, Italie]
Olivier Pons [Maître de Conférence, Université d'Evry]
Ingénieur en poste d'accueil-jeune
Ahmed Amerkad
Chercheurs doctorants
Antonia Balaa [ATER UNSA]
Néstor Cataño [bourse INRIA, depuis septembre]
Laurent Chicli [ATER UNSA]
Kuntal Das Barman [bourse INRIA, depuis mai]
Guillaume Dufay [bourse MESR]
Kwong-Cheong Wong [bourse INRIA, depuis août]
Hanane Naciri [bourse MESR]
Nicolas Magaud [bourse MESR]
Daniel Perovich [INCO, Montevideo, depuis septembre]
Simão Melo de Sousa [Boursier gouvernement Portuguais]
Kerry Trentelman [ANU, Canberra, depuis Septembre]
Stagiaires
Assia Mahboubi [ENS Lyon juillet-août]

David Pichardie [ENS Cachan/Rennes, juillet-août]
Joachim Habib [2ème année Essi juillet-août]

2 Présentation et objectifs généraux

On se reportera au rapport d'activité 2001 pour une présentation détaillée des thèmes du projet, qui ont peu changé cette année, excepté le thème «sémantique des langages» qui est monté en puissance avec l'arrivée dans le projet de Gilles Barthe et de ses étudiants, par le recrutement de Marieke Huisman comme chargée de recherche INRIA, et par le début du projet européen Verificard.

3 Fondements scientifiques

3.1 Environnements de preuves

Mots clés : preuve, environnement, interface homme-machine.

Le but de ce thème est d'étudier les outils mécaniques de recherche et de vérification de preuves pour faciliter leur utilisation par des ingénieurs et des mathématiciens dans la production de logiciels et de théories mathématiques formelles.

3.2 Théorie des types et formalisation de théories mathématiques

Mots clés : formalisation, mathématiques, théorie des types.

Les buts de ce thème sont de développer la théorie des types, et d'étudier comment des théories mathématiques où interviennent de nombreux types d'objets peuvent être représentées dans le calcul des constructions inductives (CCI en abrégé), de façon à être aussi lisibles et utilisables que possible par quelqu'un qui en a une connaissance scolaire ou académique.

3.3 Implémentations certifiées d'algorithmes de calcul scientifique

Mots clés : algorithme, certification.

Pour obtenir des programmes certifiés, nous proposons une approche inverse de celle généralement utilisée : plutôt que de chercher à prouver des propriétés d'un programme existant (en formalisant sa sémantique), nous proposons de produire des programmes dont la correction découle de celle de leur processus de création.

3.4 Sémantique des langages de programmation

Mots clés : sémantique, langages, programmation, Java, JavaCard, Coq.

Les algorithmes intervenant dans l'implantation des langages de programmation font également partie de notre champ d'investigation. Pour ces algorithmes, on se repose généralement sur la description sémantique d'un langage, et les propriétés que l'on cherche à établir pour un

algorithme sont soit qu'il préserve la sémantique des programmes (s'il s'agit d'un algorithme de transformation ou d'optimisation) soit que les programmes qu'il produit sont exempts de certains comportements indésirables (s'il s'agit d'un compilateur ou d'un vérificateur de programmes). Pour classifier ce type d'algorithmes et de vérification, nous parlons de preuves en sémantique des langages de programmation.

4 Domaines d'applications

4.1 Cartes à puces

Les langages utilisés dans les cartes à puces, ainsi que les programmes écrits dans ces langages peuvent faire l'objet de traitements formels. Dans le cas particulier de l'entreprise Gem+, nous avons étudié la preuve d'un mini-vérificateur de byte-code pour un langage abstrait concernant l'initialisation, ce qui a conduit à un vérificateur de byte-code certifié qui peut être traduit en un programme `Ocaml` grâce au mécanisme d'extraction de Coq. Dans le cadre du projet européen VERIFICARD, nous avons travaillé sur la formalisation du langage Javacard et le développement d'outils qui leurs sont associés (voir les paragraphes sur ce sujet dans la suite). Ces travaux se poursuivent avec une collaboration avec Gem+, dans le cadre du projet européen VERIFICARD.

4.2 Algorithmes certifiés

Dans certains domaines, il est crucial que les algorithmes et les programmes mis en œuvre soient absolument sûrs. C'est le cas par exemple dans les environnements permettant de faire de la preuve formelle. Dans ce domaine, à la suite de contacts avec la société Prover technology, l'implémentation certifiée de l'algorithme de Stålmarck [1] développée dans le projet a été améliorée pour pouvoir traiter des cas significatifs. C'est aussi le cas pour l'arithmétique flottante ; dans le cadre de l'action coopérative AOC, nous avons formalisé la norme IEEE754, et, avec Paul Zimmermann, effectué la preuve d'un algorithme de calcul de racine carrée et d'un algorithme de division sur les grands entiers.

4.3 Web, MathML, XML

Nos travaux autour de XML et MathML dans le cadre de Figue, ont deux principales retombées applicatives :

- d'une part ces travaux devraient nous permettre de partager des preuves (script Coq) sur le Web en générant depuis notre interface PCoq, des sources XML+MathML qui pourront être ensuite affichés (et imprimés) avec de "vraies" formules mathématiques par des navigateurs acceptant du MathML, et ceci indépendamment de Pcoq.
- d'autre part, dans la suite des travaux de Loïc Pottier sur Wims (WWW Interactive Mathematics Server, developpé par XIAO Gang à l'université de Nice), nous voulons expérimenter l'interaction pour les formules mathématiques à travers le Web, afin de construire directement les preuves sur le Web.

[1] Satisfiabilité de formules booléennes

Cette migration vers le Web devrait, à terme, élargir la visibilité de nos travaux, et nous permettre d'atteindre une plus grande communauté d'utilisateurs afin de valider nos outils. Le contrat européen LTR Mowgli, récemment accepté va nous y aider fortement.

5 Logiciels

5.1 Pcoq

Participants : Ahmed Amerkad, Yves Bertot [correspondant], Loïc Pottier, Laurence Rideau.

La version 1.2 de Pcoq a été rendue disponible en avril : `http://www-sop.inria.fr/lemme/pcoq/pcoq-fra.html`. Pcoq fournit un environnement de travail pour le système de preuve Coq. Il a été développé en suivant une approche générale pour la construction d'interfaces utilisateurs pour les assistants de preuves. Il réunit les caractéristiques suivantes :

- Interface graphique : des polices de caractères multiples et des couleurs sont utilisées pour afficher les formules mathématiques et les commandes.
- Séparation entre l'interface et le système de preuve : l'interface graphique et Coq sont deux processus indépendants. Les utilisateurs peuvent choisir de faire tourner l'un des processus sur une autre machine accessible sur le réseau.
- Des mécanismes d'édition et de présentation structurées : l'environnement fournit des moyens pour éditer les formules en respectant leur structure. De nouvelles notations peuvent être ajoutées facilement.
- *Proof by pointing* : l'environnement utilise la structure des formules logiques pour aider l'utilisateur à effectuer les étapes du raisonnement en les désignant à la souris.
- Écrit en Java et en `Ocaml`, Pcoq bénéficie de leurs portabilités combinées.
- Langue naturelle : les preuves en cours de développement peuvent être visualisées sous forme d'un texte en français ou en anglais, sur lequel les mécanismes de *proof-by-pointing* fonctionnent.

Pcoq continue à être développé, en particulier grâce au contrat sur poste d'accueil-jeune d'Ahmed Amerkad, dont le travail est décrit dans la suite.

Prise en charge de Pcoq : avant de prendre en charge le côté java de Pcoq, on a dû passer du temps pour comprendreun travail de compréhension a dû être fait pour sur l'environnement du travail vu les dimensions (multi-langage, concurrent, basé sur la communication de plusieurs programmes, multi-plateforme) de Pcoq et sa complexité.

L'efficacité : ce qui a été fait à ce stade, est le changement du protocole de communication avec le programme externe Coq, le portage de Pcoq sous jdk1.3 et l'élaboration des tests avec OptimizeIt comme profiler java.

L'ergonomie : ce point concerne un certain nombre d'éléments ergonomiques et interactifs visant à mieux guider les utilisateurs du système (génération automatique des menus, témoin de l'état de Pcoq, structuration de script de commandes).

Gestion de documents : cela permet à l'utilisateur de gérer des documents au cours d'une session (historique des fichiers ouverts, historique des buffers traités au cous d'une session,

historique des derniers résultats de recherche, recherche par chaîne de caractères, recherche par motif d'arbre).

Fonctionnement de Pcoq : il s'agit d'assurer certaines fonctionnalités qui permettent aux utilisateurs de faire des retours en arrière et des allers en avant dans leurs preuves. Il est à noter aussi que le protocole de communication avec le parser est changé pour ne plus perdre la connexion une fois établie. Il est aussi possible maintenant, de développer plusieurs preuves à la fois dans une seule fenêtre de preuve.

Configuration de Pcoq : il s'agit de permettre aux utilisateurs de configurer les propriétés de Pcoq au cours d'une session (changement du langage utilisé par l'interface, changement des propriétés ppml d'un opérateur, ajout de nouveaux chemins(PATH)) et de leur offrir des options supplémentaires au lancement de Pcoq.

5.2 Aïoli et Figue

Participants : Hanane Naciri, Laurence Rideau, Laurent Théry [correspondant].

Aïoli (`http://www-sop.inria.fr/croap/aioli/doc/aioli.html`) et Figue (`http://www-sop.inria.fr/croap/figue/`) sont des composants de base de Pcoq, et à ce titre ont reçu des améliorations : sélections multiples, nouveaux combinateurs 2D (matrices, crochets, fractions, racines n-ièmes).

5.3 Algorithme de Stålmarck

Participants : Pierre Letouzey, Laurent Théry [correspondant].

À l'adresse (`http://www-sop.inria.fr/lemme/stalmarck/index.html`), on peut tester une implémentation certifiée de l'algorithme de Stålmarck (détection de tautologies dans les formules booléennes).

6 Résultats nouveaux

6.1 Outils pour les environnements de preuve

6.1.1 Evolution du système Pcoq

Participants : Ahmed Amerkad, Yves Bertot, Hanane Naciri, Laurence Rideau, Loïc Pottier, Laurent Théry.

Une nouvelle version de l'interface graphique pour le système de preuve Coq a été distribuée en cours d'année. Parmi les améliorations présentes dans cette version, soulignons en particulier un système d'affichage bi-dimensionnel bien plus performant, une plus grande facilité de configuration et de meilleures performances en consommation mémoire. Un article décrivant les apports de cette interface pour la présentation des démonstrations a été présenté au colloque PTP (Proof Transformation and Presentation'01) qui s'est tenu en marge de la conférence ETAPS'2001 à Gênes.

6.1.2 Affichage de formules mathématiques

Participants : Hanane Naciri, Laurence Rideau.

Ce travail se place dans le cadre du développement d'outils pour l'interaction homme machine dans les environnements de démonstrations mathématiques. Nous continuons à étendre et à améliorer notre outil d'affichage bidimensionnel, incrémental, et interactif FIGUE. Cet outil, permet de manipuler dynamiquement les objets structurés représentant des documents comme des programmes ou des formules mathématiques. Notre but est de présenter les objets structurés et en particulier les formules mathématiques de façon conviviale et d'offrir des interactions variées à la souris comme la sélection de sous-expressions afin de pouvoir les manipuler dynamiquement (évaluation, simplification, modification, génération de code, etc...). Nous étudions le problème lié à la diversité des objets à manipuler : du texte simple, des formules mathématiques pour les systèmes de preuves ou de calcul formel, des images. Des problèmes particuliers se posent pour les formules mathématiques qui ont une structure complexe et bidimensionnelle (matrices, intégrales, indices, ...).

Pour nous conformer au standard des mathématiques sur Internet MathML dans le cadre des environnements de preuve, nous nous sommes intéressées au lien entre FIGUE et MathML. Nous avons implémenté un module en FIGUE permettant d'interpréter des documents structurés XML incluant du MathML (MathML présentation et contenu), de les afficher et de les manipuler par la suite. FIGUE peut être utilisé comme composante d'affichage de MathML par d'autres applications. Inversement, il est possible de générer du MathML à partir des objets mathématiques affichés par FIGUE. Tout ceci fait que MathML peut être utilisé par FIGUE comme format d'importation et d'exportation de données et aussi comme format d'échange avec d'autres logiciels mathématiques. Cette implémentation permet ainsi l'affichage des preuves issues de Pcoq par tout navigateur Web, supportant du MathML.

Nous travaillons aussi sur l'affichage bidirectionnel des formules mathématiques dans différentes langues, en particulier l'arabe et l'hébreu (où le texte s'écrit de droite à gauche et les formules de gauche à droite). Notre travail a pour but de déterminer les règles et l'algorithme d'affichage bidirectionnel des formules mathématiques pour gérer le changement de direction d'affichage. L'affichage bidirectionnel des formules mathématiques est plus complexe que celui du texte simple. Cette complexité est due à la nature bidimensionnelle des formules mathématiques et à l'importance des relations spatiales dans les notations mathématiques. Nous expérimentons les résultats de notre algorithme bidirectionnel dans notre interface Pcoq : il est possible dans Pcoq d'avoir des explications des preuves mathématiques en langue naturelle, en français et en anglais et nous expérimentons l'implémentation des explications en arabe, ce qui nous permet de tester notre algorithme d'affichage bidirectionnel dans FIGUE pour écrire de droite à gauche et même mélanger l'affichage droite-gauche et l'affichage gauche-droite (formules mathématiques).

6.1.3 Visualisateur de théories mathématiques Coq en SVG.

Mots clés : formules mathématiques, Web, SVG.

Participant : Joachim Habib.

SVG (Scalable Vector Graphics) est une proposition du W3C pour l'échange et la visualisation de documents contenant des graphiques. Nous avons étudié comment il serait possible d'utiliser SVG pour visualiser les théories mathématiques développées en Coq. Un prototype a été développé qui utilise SVG pour représenter une théorie comme un graphe interactif contenant les différents fichiers de cette théorie avec leurs dépendances. Sélectionner un nœud du graphe permet de visualiser avec les notations mathématiques usuelles les définitions et théorèmes contenus dans le fichier selectionné.

6.2 Théorie des types et formalisation de théories mathématiques

6.2.1 Fonctions récursives

Participants : Antonia Balaa, Gilles Barthe, Yves Bertot, Pierre Courtieu.

Gilles Barthe et ses collègues de l'Université du Minho ont développé une nouvelle théorie dans laquelle la terminaison des fonctions récursives est garantie par le typage, ce qui permet d'obtenir des langages plus robustes et plus faciles à implanter correctement.

Antonia Balaa et Yves Bertot ont effectué cette année une avancée marquante dans notre travail sur les fonctions récursives générales. En effet, ils ont mis au point une technique pour obtenir la démonstration de l'équation de point fixe bien plus rapide et facile à mettre en œuvre que le résultat présenté l'année précédente. Pour l'utilisateur d'un système de preuve comme Coq, ceci se traduit par une meilleure intégration du style de programmation "à la ML" dans le langage de programmation du système de preuve. Un article présentant ce résultat a été soumis pour publication. La thèse d'Antonia Balaa est également en cours de rédaction.

Pierre Courtieu, Gilles Barthe et Yves Bertot ont développé une méthode de démonstration de propriétés de fonctions récursives qui passe par l'établissement d'un théorème décrivant la structure habituelle des preuves concernant cette fonction. Intuitivement, ce théorème prend en argument des propriétés à prouver pour chacun des cas de définition de la fonction et chacune de ces propriétés comporte une hypothèse de récurrence pour chaque appel récursif de la fonction. Cette méthode se concrétise par la construction d'un outil automatique qui construit le théorème à partir de la définition de la fonction. Une extension à la récursion générale est également envisagée.

6.2.2 Changements de structure de données et réutilisation de démonstrations

Participants : Nicolas Magaud, Gilles Barthe, Yves Bertot.

Le travail de Nicolas Magaud et Yves Bertot sur les traductions de démonstrations d'une structure à une autre s'est continué par une comparaison avec les travaux de Connor Mc Bride sur les intéractions entre structures inductives et récursion. Les résultats préliminaires ont été

publiés dans une conférence francophone [18] et une version plus étoffée est également en cours de publication dans les actes d'une conférence internationale.

Gilles Barthe et Olivier Pons ont étudié une nouvelle approche pour les coercions implicites : cette approche simplifie le développement théorique sous-jacent sans affecter l'expressivité de la théorie des types [11].

6.2.3 Logique

Participant : Gilles Barthe.

Gilles Barthe a travaillé :
- sur les traductions en continuation-passing style (CPS) pour les types inductifs et coinductifs (collaboration avec T. Uustalu) [12].
- sur les propriétés méta-théoriques des systèmes de types purs, en particulier à la vérification du typage (collaboration avec B. Ruiz Jiménez) et les points fixes (collaboration avec T. Coquand [29])
- la formalisation des ensembles en théorie des types (collaboration avec V. Capretta et O. Pons) [28] ;

6.2.4 Quotients

Participants : Laurent Chicli, Loïc Pottier.

Nous nous sommes intéressé au problème des types quotients dans *Coq*. L'un des défauts majeur du *CCI* lorsqu'on s'y intéresse pour formaliser les mathématiques est l'absence de type quotient. En effet ceci oblige l'utilisateur, s'il veut pouvoir parler d'ensembles quotients dans *Coq*, à utiliser la structure de sétoïde (la donnée d'un type 'support' et d'une relation d'équivalence qui sert d'égalité sur ce type) dont l'expérience montre la lourdeur d'utilisation à long terme. Loïc Pottier, en se basant sur des travaux de G.Barthe, S.Boutin et M.Hofman, a proposé un système d'axiomes pour les types quotients qui semble aussi expressif que les quotients mathématiques [23]. Laurent Chicli et Loïc Pottier ont développé des commandes et tactiques pour *Coq* pour utiliser ces axiomes et faire en sorte que leur utilisation soit des plus naturelles possible et ressemble à celle qu'en ont les mathématiciens.

6.3 Certification d'algorithmes

6.3.1 Élimination des quantificateurs dans les réels, et tactique en Coq

Participants : Assia Mahboubi, Loïc Pottier.

Le système Coq[2] permet maintenant de faire des preuves formelles en analyse réelle. Mais les preuves dans ce domaine sont encore très pénibles, car très peu automatisées, en particulier celles qui manipulent les inégalités. Dans le cas des équations et inéquations linéaires, la tactique Fourier permet au système de faire lui-même les preuves, mais cette tactique ne s'applique pas dans le cas de systèmes d'équations et d'inéquations polynômiales. Pourtant il

[2]Projet Logical, `http://pauillac.inria.fr/coq/coq-fra.html`

existe pour ce cas des procédures de décision efficaces et simples, au moins en degré faible et avec un nombre de variables ne dépassant pas la dizaine.

Le but du travail qu'on présente ici[3] est d'implémenter en Ocaml [4] une telle procédure sous la forme d'une tactique de Coq. Nous pensons qu'elle pourra être très utile pour les utilisateurs de Coq qui travailleront avec des nombres réels. On peut pour s'en convaincre constater que de très nombreuses preuves faites dans la librairie Reals de Coq reviennent à montrer qu'un système d'inéquations polynômiales n'a pas de solution (par exemple dans les calculs de limites, de continuité).

La méthode qu'on a choisie est une méthode d'élimination des quantificateurs dans les corps réels clos, basée sur le principe de Tarski-Seidenberg,telle qu'elle est exposée dans le livre de J. Bochniak, M. Coste, et M-F. Roy : *Géométrie Algébrique Réelle*, Springer-Verlag (1986), pages 15-19. Basée sur une démonstration d'Hormander, elle a l'avantage d'être assez simple à programmer, relativement efficace, et de ne faire intervenir que des théorèmes d'analyse réelle élémentaire (essentiellement le théorème des valeurs intermédiaires, prouvé en Coq par deux stagiaires l'année passée).

En pratique la tactique suit les modèles des tactiques Omega et Fourier. On commence par traduire les objets mathématiques de Coq qui interviennent dans les hypothèses et dans le but qu'on veut prouver dans des types de données de Ocaml, puis on effectue le calcul de la procédure de décision en Ocaml. Si celle-ci réussit, on construit alors à partir d'une trace des calculs effectués une preuve dans le formalisme de Coq, qui est ensuite vérifiée par le système.

Si cette tactique s'avère utile, on pourra ensuite envisager d'utiliser la technique de la réflexion pour programmer et vérifier dans Coq lui-même la procédure de décision utilisée. Mais ce n'est pas encore à l'ordre du jour, d'autant que la programmation de la méthode de J. Bochniak, M. Coste, et M-F. Roy a fait apparaître de nombreuses améliorations algorithmiques possibles : l'algorithme n'est de fait pas encore stabilisé.

Actuellement, les choses en sont au stade suivant :

- L'algorithme d'élimination des quantificateurs est écrit en Ocaml et fonctionne sur des exemples simples mais non triviaux, comme l'équation du troisième degré (voir en annexe). Il fonctionne aussi sur des exemples tirés des preuves sur les réels de la librairie de Coq. Enfin il fonctionne sur l'équation du quatrième degré, mais échoue à reconstruire la trace complète des calculs par manque de place mémoire (>1 Giga).
- La construction de la preuve Coq dans le cas d'une variable est à 90% effectuée. Reste à prouver en Coq les théorèmes sur les réels utilisés.
- Un prototype de tactique, appelée «Tarski» fonctionne en Coq : cette tactique appelle l'algorithme en Ocaml, et, s'il répond par un succès, prouve sauvagement le but à l'aide d'un axiome `(P :Prop)P`. Evidemment le but est de vérifier la faisabilité de l'intégration du code caml avec Coq, et aussi de pouvoir tester facilement et confortablement des exemples variés de preuves sur les réels.

L'extension de la construction de la preuve Coq au cas de plusieurs variables devrait demander beaucoup moins de travail que ce qui a été nécessaire au cas d'une variable, où les

[3] Ce travail a en grande partie été effectué lors du stage de deuxième année de l'ENS Lyon d'Assia Mahboubi, dans le projet Lemme de l'INRIA Sophia Antipolis, encadré par Loïc Pottier et Marie-Françoise Roy (IRMAR Rennes I)

[4] `http://caml.inria.fr/ocaml/`

nombreux cas d'application du théorème des valeurs intermédiaires sont assez fastidieux.

Ce travail est décrit dans [20].

6.3.2 Intégration d'un vérificateur de tautologie dans le système Coq.

Mots clés : formules booléennes, réflexion, extraction, trace, Coq.

Participant : Laurent Théry.

Suite à la formalisation de l'algorithme de Stålmarck dans Coq, nous avons étudié comment il est possible de l'intégrer afin d'obtenir une procédure automatique de preuve. Différentes approches ont été essayées et comparées :

- Extraction : on ajoute au code de Coq une implantation certifiée de l'algorithme.
- Réflexion : on utilise la capacité calculatoire de la logique pour exprimer l'algorithme en Coq
- Trace : on sépare la recherche de la preuve de sa vérification. La recherche se fait à l'exterieur de Coq et la vérification par réflexion.

Ce travail fera l'objet d'une publication comme rapport de recherche en fin d'année.

6.3.3 Extension de la formalisation des flottants dans Coq.

Mots clés : nombres flottants, norme IEEE754, certification.

Participants : Laurent Théry, Laurence Rideau.

Dans le cadre de l'action de recherche coopérative AOC (Arithmétique sur Ordinateur Certifiée), nous avons continué notre travail de formalisation des nombres flottants en Coq. Notre approche, qui se veut la plus générique possible, nous a permis de prouver la correction d'un programme qui détecte la base de l'arithmétique de l'ordinateur sur lequel il tourne. Ce travail a fait l'objet d'une publication [15] et notre bibliothèque fait maintenant partie de la distribution de Coq.

6.3.4 Certification d'algorithmes d'enveloppe convexe

Participants : David Pichardie, Yves Bertot.

Le travail sur les algorithmes certifiés en Coq pour le calcul d'enveloppe convexe a été complété dans deux directions. D'une part nous avons formalisé une méthode de résolution des dégénerescences par perturbation, suggérée par O. Devillers du projet Prisme. D'autre part, nous avons montré que nos algorithmes satisfaisaient une spécification plus abstraite que la spécification originale, inadaptée pour valider la méthode de résolution par perturbations. Un article a également été rédigé et publié dans la conférence TPHOLs'2001 [21].

6.3.5 Algorithmes arithmétiques sur les grands entiers

Participants : Didier Bondyfalat, Nicolas Magaud, Yves Bertot, Paul Zimmerman

(projet Spaces).

Dans le cadre de l'action de recherche concertée AOC, nous avons travaillé sur la formalisation en Coq d'algorithmes sur les grands entiers, en nous concentrant sur un algorithme de division et un algorithme de calcul de racine carrée, tous deux basés sur la méthode "diviser pour règner". Nous avons étudié ces algorithmes à plusieurs niveaux d'abstraction : le niveau le plus abstrait considère des nombres entiers sans se préoccuper de la quantité de mémoire nécessaire pour les représenter, un niveau intermédiaire considère ces nombres en vérifiant leur taille par rapport aux exposants d'une base donnée (ce niveau d'abstraction n'a été utilisé que pour l'algorithme de division). Le troisième niveau manipule directement la mémoire comme un tableau dont chaque cellule ne peut contenir qu'un fragment inférieur à la base, en utilisant des techniques de programmation impérative. La vérification se fait en utilisant des techniques "à la Hoare" grâce aux travaux de J.-C. Filliâtre de l'équipe Démon de l'Université de Paris-Sud (ce niveau d'abstraction n'a été utilisé que pour la racine carrée). Ces expériences pourront mener à une approche systématique de la réutilisation des preuves lors des changements de niveau d'abstraction.

6.3.6 Représentation des nombres rationnels

Participant : Yves Bertot.

Nous avons mis au point une structure de donnée extrèmement simple (une algèbre libre à trois constructeurs) qui peut être mise en bijection avec l'ensemble des nombres rationnels strictement positif. Les propriétés de base de cette structure de donnée ont été vérifiées formellement en Coq. Un rapport de recherche est en cours de préparation.

6.4 Sémantique des langages de programmation

6.4.1 Description sémantique des langages

Participants : Kuntal Das Barman, Yves Bertot.

Les techniques de descriptions pour les fonctions récursives générales étudiées dans la thèse d'Antonia Balaa devraient également s'appliquer pour la description fonctionnelle d'un langage de programmation issu de la sémantique dénotationnelle. Nous approfondissons actuellement cette idée en espérant l'appliquer aux descriptions formelles de la machine virtuelle de JavaCard et du langage JavaCard.

6.4.2 Langages de spécification

Participants : Nestor Cataño Collazos, Marieke Huisman, Kerry Trentelman (ANU, Australie).

Pour pouvoir vérifier des programmes (en Javacard), on doit d'abord spécifier ses propriétés. Nous étudions les langages de spécification existants et leurs outils associés, comme JML et ESC/Java, pour voir s'ils sont capables de spécifier ce que nous demandons, et pour proposer des extensions facilitant nos besoins.

6.4.3 Le langage JavaCard

Participants : Gilles Barthe, Yves Bertot, Guillaume Dufay, Line Jacubiec, Simão Melo de Sousa.

Le langage JavaCard est une variante du langage Java adaptée au fonctionnement sur les cartes à puces, qui se compile en un langage assembleur pour une machine virtuelle appelée JCVM. Le bytecode verifier (BCV) est une des principales fonctions de sécurité de la plateforme JavaCard : il exprime une discipline de typage permettant de rejeter les programmes qui effectuent des opérations illicites du point de vue des types de données (comme appliquer à un objet d'une certaine classe une méthode utilisable uniquement pour des objets d'une autre classe).

Gilles Barthe, Guillaume Dufay, Line Jakubiec et Simão Melo de Sousa ont fourni une description formelle complète de cette machine virtuelle et utilisé cette description formelle pour dériver un BCV qu'ils ont ensuite certifié [8]. Yves Bertot a étudié une discipline similaire pour le problème de l'initialisation des objets [13].

6.4.4 Environnement de vérification pour JavaCard

Participants : Gilles Barthe, Pierre Courtieu, Guillaume Dufay, Marieke Huisman, Simão Melo de Sousa, Sorin Stratulat.

La vérification de bytecode permet de s'assurer de l'absence d'erreurs de typage, initialisation, etc. à l'éxécution d'un programme mais n'apporte aucune garantie pour d'autres erreurs/anomalies (comme une consommation abusive des ressources, qui mènerait à une attaque dite de deni de service). En conséquence, il est important de développer de nouveaux vérifieurs de bytecode ayant la capacité d'effectuer des vérifications supplémentaires.

Dans le cadre de nos travaux autour de JavaCard, nous avons conçu un prototype appelé Jakarta [7], dont le but est de faciliter la construction et la certification de tels vérifieurs. Il s'agit en particulier de produire, à partir d'une machine virtuelle défensive (c.à.d. une machine effectuant toutes les vérifications à l'éxécution) :

- une machine virtuelle abstraite qui est à son tour utilisée pour construire un vérifieur de bytecode ;
- une machine offensive (c.à.d. une machine optimisée effectuant le plus petit nombre de vérifications à l'éxécution) ;
- une preuve de correction du vérifieur de bytecode, montrant que les machines coincident sur les programmes ayant réussi la vérification de bytecode.

Nous avons déjà obtenu des résultats prometteurs sur les deux premiers points, et poursuivrons nos travaux sur le prototype, qui devrait à terme devenir une véritable boîte à outils. Nous nous attacherons en particulier à l'automatisation des preuves en nous appuyant sur des techniques de réécriture (nos expériences avec le BCV suggèrent que la plus grosse partie de la preuve de correction de la machine virtuelle abstraite, à partir de laquelle il est facile de déduire la correction du BCV, consiste en de la réécriture, et peut être automatisée).

D'autre part, nous utiliserons Jakarta dans nos travaux sur la sémantique de JavaCard, et aussi pour valider nos travaux sur les systèmes de type pour le code mobile, que nous décrivons

plus bas.

6.4.5 Vérification du code mobile et embarqué

Participants : Gilles Barthe, Dilian Gurov (SICS, Suède), Marieke Huisman, Daniel Perovich, Christophe Sprenger (SICS, Suède puis INRIA à partir de Septembre 2002), Kwong-Cheong Wong.

La sécurité du code mobile et embarqué passe bien sûr par la correction de la plate-forme d'éxécution, mais aussi par le bon comportement des programmes. Ainsi il est nécessaire de définir pour chaque application une politique de sécurité et de démontrer sa conformité vis-à-vis de celle-ci. Dans cette perspective, nous avons introduit une logique temporelle pour la vérification compositionelle et étudié ses applications à la confidentialité [9, 30].

7 Contrats industriels (nationaux, européens et internationaux)

7.1 Dassault Aviation

Ce contrat finance la thèse d'Antonia Balaa, sur l'étude des fonctions récursives à terminaison non structurelle en théorie des types.

7.2 Mowgli

La proposition européenne Mowgli (LTR) a été acceptée, et démarre à la fin 2001. Le sujet concerne les mathématiques formelles sur le Web. Participants : Universités de Bologne, de Berlin, de Nijmgen et Eindhoven, DFKI Saarbrücken, Max Planck Institute, et la société Trusted Logic.

7.3 Verificard

Ce contrat, démarré en 2001 pour une durée de 3 ans, finance les thèses de Kuntal das Barman, de Nestor Cataño Collazos et de Kwong Cheong Wong, et les stages post-doctoraux de Sorin Stratulat et Pierre Courtieu. Il concerne la modélisation et la vérification de la plate-forme et des programmes Javacard (participants : GemPlus, Bull, Universités de Nimegue, Munich, Hagen, Sics). Le travail du projet dans ce cadre est décrit dans la partie «Sémantique des langages de programmation».

8 Actions régionales, nationales et internationales

8.1 Actions internationales

8.1.1 Collaboration avec l'ORCCA

Nous collaborons avec le professeur Stephen Watt sur des sujets portant sur MathML (dialecte de XML pour les mathématiques `http://www.w3.org/Math/`). Actuellement, nous

coopérons plus particulièrement avec l'ORCCA (Ontario Research Centre for Computer Algebra) sur l'affichage bidirectionnel de mathématiques dans certains scripts (l'aspect bidirectionnel étant nécessaire dans certaines langues comme l'arabe ou l'hébreu) : notre but étant de générer et d'afficher des explications de preuves mathématiques en langue naturelle (en l'occurrence l'arabe), ce qui implique un mélange d'affichage droite-gauche (le texte arabe), et gauche-droite (les formules mathématiques). Ces travaux devraient permettre de définir de nouvelles recommandations de la norme MathML (Stephen Watt est expert invité du groupe de travail MathML) et seront inclus dans notre interface Pcoq (`http://www-sop.inria.fr/lemme/pcoq/index.html`) pour le système de preuve Coq.

8.1.2 Collaboration avec ICM

Nous collaborons aussi avec le professeur Paul Wang du département informatique de l'université de Kent State aux Etats Unis. Nous travaillons avec son équipe sur le projet IAMC (Internet Accessible Mathematical Computation) : utilisation de notre composante d'affichage interactif FIGUE dans leur interface pour afficher et manipuler des formules mathématiques encodées en MathML. Cette collaboration nous permet d'utiliser et de tester FIGUE comme composante d'affichage supportant du MathML.

8.1.3 Divers

L'équipe est impliquée dans deux collaborations bilatérales :

- collaboration ICCTI-INRIA avec l'Université de Minho, Portugal. La collaboration porte sur les systèmes de types pour les assistants de preuve et les langages de programmation.
- collaboration INRIA-Cono Sur avec des universités argentines et uruguayennes. La collaboration porte sur l'utilisation des méthodes formelles pour la sécurité des cartes à puce.

Nous collaborons également avec l'Automated Reasoning Group de l'Australian National University, Canberra, Australie, sur les extensions de JML avec la logique temporelle.

8.2 Actions nationales

8.2.1 Action de Recherche Coopérative AOC.

L'année 2001 est la deuxième et dernière année de l'action de recherche coopérative AOC (Arithmétique sur Ordinateur Certifiée). Cette action regroupe les équipes Arénaire (Rhones-Alpes), Lemme (Sophia) et Spaces (Nancy). Elle a pour but d'appliquer les méthodes formelles au développement et à la vérification d'opérateurs arithmétiques. La troisième réunion s'est tenue en juin à Sophia et la quatrième en décembre à Nancy. L'activité de cette action de recherche est présentée à l'adresse suivante : `http://www-sop.inria.fr/lemme/AOC/`

8.2.2 Action de Recherche Coopérative S-Java.

L'année 2001 est la deuxième et dernière année de l'action de recherche coopérative SJava, qui regroupe les équipes Coq, Lande, Lemme, Mimosa et Oasis, l'équipe Interprétation Abstraite et Sémantique de l'ENS, ainsi que Gemplus et Trusted Logic. Le but de l'action est

l'utilisation des méthodes formelles pour la vérification de propriétés de sécurité des programmes Java et JavaCard. L'activité de cette action de recherche est présentée à l'adresse suivante :`http://www-sop.inria.fr/oasis/SJava`

8.2.3 Divers

Gilles Barthe et Yves Bertot ont participé à une collaboration locale de recherche (COLOR) avec les équipes de GemPlus et l'Université de Marseille, sur l'étude du langage JavaCard. A l'issue de cette collaboration Solange Coupet-Grimal, maitre de conférence à l'université de Marseille a demandé et obtenu une délégation à l'INRIA Sophia-Antipolis.

8.3 Actions européennes

8.3.1 Réseau Types

Lemme participe activement au réseau Types reconduit en 2000, qui regroupe les équipes européennes travaillant sur la théorie des types.

9 Diffusion de résultats

9.1 Manifestations scientifiques, missions

Gilles Barthe a participé aux conférences APPSEM'01, ETAPS'01, DTP'01, JSLC'01. Il a donné un exposé invité aux journées Plate-forme Systèmes et Logiciels Critiques, Grenoble, Novembre 2001. Exposé sur la vérification de la plate-forme et des applications JavaCard ("Tool-Assisted Reasoning about JavaCard").

Nicolas Magaud, Laurent Chicli, Loïc Pottier et Laurence Rideau ont participé aux JFLA'2001 à Pontarlier, France les 29 et 30 Janvier 2001.

Yves Bertot a présenté ses travaux sur la vérification de byte-code à la conférence CAV'01 (Computer Aided Verification) à Paris en Juin 2001. Il a présenté les travaux sur la géométrie algorithmique certifiée effectués avec David Pichardie à la conférence TPHOLs2001 (Theorem Proving in Higher Order Logics) à Edimbourg en Septembre 2001.

Didier Bondyfalat s'est rendu à l'INRIA Nancy dans le cadre de l'action de recherche concertée AOC en Septembre 2001.

Guillaume Dufay a présenté un exposé aux conférences à ETAPS/ESOP à Gênes. Il a participé à l'école d'été EJC 2001 à Cargèse en Corse.

Marieke Huisman : a présenté un exposé aux conférences E-Smart 2001 (Cannes), ECOOP (Budapest), aux séminaires des projets Lande et Logical, à l'Université de Nijmegen (Informatics for Technical Applications) et à SICS, Suède. Elle a donné un exposé aux réunions WP4 VerifiCard et SJava.

Hanane Naciri a participé au Workshop IAMC'2001 (Internet Accessible Mathematical Computation) et la conférence ISSAC'2001 (International Symposium on Symbolic and Algebraic Computation), qui ont eu lieu en Juillet 2001 à London Ontario au Canada.

Elle a participé à la conférence asiatique ASCM'2001 (Asian Symposium on Computer Mathematics), du 26 au 28 septembre 2001 à Matsuyama, Japan.

Nicolas Magaud a présenté ses travaux sur les changements de structures de données à la conférence JFLA'2001 (Journées Francophones sur les langages applicatifs) qui s'est tenue à Pontarlier en Janvier 2001. Il s'est rendu à Paris en Novembre 2001 pour y présenter les travaux effectués sur l'algorithme de racine carré dans le cadre d'un séminaire du projet Spaces.

Loïc Pottier a présenté des communications aux conférences JFLA2001 à Pontarlier, TPHOL2001 à Edimbourg, et PTP à Sienne.

Simao Melo de Sousa a participé aux conférences APPSEM'01 et E-SMART'01.

Laurent Théry s'est rendu à l'université de Eindhoven (Pays-Bas) début février, à la conférence TPHOL à Edimbourg (Écosse) début septembre, et à plusieurs occasions à l'université de l'Aquila (Italie).

9.2 Animation de la communauté scientifique

Gilles Barthe est membre du comité de direction et représentant de l'INRIA (équipes Lemme, Lande et Logical) au sein du projet VerifiCard. Il a co-organisé la rencontre "Dependent Type Theory Meets Programming Practice", Dagstuhl, Allemagne, 19.08.2001-24.08.2001 avec P. Dybjer (Chalmers, Suède) et P. Thiemann (Freiburg, Allemagne). Il est éditeur d'ún numéro spécial de JFP, memebre des comités de programmes des conférences ICALP'01, SAIG'01, et organisé le séminaire Dagstuhl 01341.

Nicolas Magaud, Marieke Huisman et Kuntal Das Barman organisent le séminaire Lemme.

Laurence Rideau est présidente du comité de programme des JFLA'2002

Laurent Théry a organisé la troisième réunion du groupe de travail AOC à Sophia en juin.

9.3 Divers

Gilles Barthe est membre suppléant de la commission de spécialistes 27e section de l'Université de Marseille.

Yves Bertot est membre de la commission de spécialistes 27e section à l'université de Marseille II, et membre suppléant de la commission de spécialistes 27e section à l'école normale supérieure de Lyon.

Loïc Pottier est membre nommé du CNU 27ème section, membre suppléant des commissions de spécialistes 27ème section de l'UNSA et l'université de Perpignan.

9.4 Visites

Sylvie Boldo, étudiante à l'ENS Lyon, nous a rendu visite une semaine en février.

Nous avons accueilli de nombreux visiteurs étrangers, en particulier à l'occasion du séminaire du projet (cf `http://www-sop.inria.fr/lemme/Nicolas.Magaud/seminaire/index.html`)

9.5 Direction de thèses

Gilles Barthe dirige les thèses de Maria João Frade (Université du Minho, Portugal), Simão Melo de Sousa, Guillaume Dufay et Kwong Cheong Wong.

Yves Bertot dirige les thèses d'Antonia Balaa, Nicolas Magaud et Kuntal Das Barman.

Marieke Huisman dirige la thèse de Nestor Cataño Collazos.

Loïc Pottier codirige avec André Hirschowitz la thèse de Laurent Chicli.

Laurence Rideau dirige la thèse de Hanane Naciri.

9.6 Jury de thèses

Yves Bertot est rapporteur pour la thèse de David Delahaye et co-rapporteur (avec Laurent Théry, non-habilité) pour la thèse de Micaela Mayero.

Gilles Barthe a participé au jury de thèse de Guillaume Gillard, Université Paris 7, Juin 2001.

Loïc Pottier a participé au jury de thèse de Claire Finot.

9.7 Encadrement de stagiaires

Gilles Barthe dirige un étudiant uruguayen (Daniel Perovich) entre Septembre 2001 et Mars 2002. Ses travaux portent sur JavaCard.

Yves Bertot : David Pichardie, Ens Cachan Bretagne, 2 mois.

Marieke Huisman : un stagiaire de DEA, Néstor Cataño Collazos sur «La clause modifiable de JML : Sémantique, Vérification et Application»

Loïc Pottier : Assia Mahboubi, deuxième année ENS Lyon une tactique d'élimination des quantificateur dans les réels en Coq. Deux étudiants de la maîtrise MIM de l'UNSA, Eric Wolszytnski et Ramzi Redjeb, sur la conversion de preuves Coq en langue naturelle avec le logiciel GF.

Laurence Rideau, Laurent Théry : Joachim Habib 2eme année Essi du 1/07 au 31/08 "Visualisation d'expressions mathématiques en SVG"

9.8 Enseignement

Antonia Balaa : algorithmique Java en Deug (39 heures)

Gilles Barthe a donné un cours sur les systèmes d'aide à la preuve, leurs principes et applications à l' Universidad de la Républica, Montevideo, Uruguay, Aout 2001. Il a été responsable du module "Sécurité : réseaux, systèmes, applications" à l'Université de Sophia-Antipolis, DESS Télécommunications.

Yves Bertot : Sémantique des langages de programmation en maîtrise (24 heures), Introduction aux méthodes formelles en DESS (24 heures), logique en licence (10 heures).

Didier Bondyfalat : Programmation C en maîtrise (45 heures).

Laurent Chicli : Travaux dirigés en Licence de Mathématiques à l'UNSA : Algorithmique et programmation (L5) (26 heures). Travaux dirigés en DEUG MI1 à l'UNSA : Algèbre

et Analyse (52 heures). Encadrements de projets maple en première année d'ingénieur à l'ESSI (24 heures). Cours et travaux dirigés de mathématiques générales en première année d'ingénieur à l'ESSI (72 heures).

Guillaume Dufay a assuré des TD/TP de Java à l'université de Nice, 39 heures en DEUG MASS 2 et 21 heures en DEUG MI-1.

Marieke Huisman cours «Java program verification in higher order logic with PVS and Isabelle» en Uruguay.

Nicolas Magaud : algorithmique Java en Deug (39 heures)

Hanane Naciri Travaux pratiques du cours JAVA en première année ESSI (22 heures), encadrement de projet de programmation en première année ESSI, d'une semaine à plein temps.

Loïc Pottier : 52h de cours d'algorithmique et logique en maîtrise Math-Info à l'UNSA.

Laurent Théry a donné des cours au DEA de Marseille, a donné un cours sur la mécanisation des preuves en arithmétique flottante à l'École de printemps d'informatique théorique en mars à Prapoutel-Les-Sept-Laux.

10 Bibliographie

Thèses et habilitations à diriger des recherches

[1] M. Huisman, *Java program verification in higher order logic with PVS and Isabelle*, thèse de doctorat, University of Nijmegen, 2001.

Articles et chapitres de livre

[2] G. Barthe, T. Coquand, *Dependent Type Theory*, *Lecture Notes in Computer Science*, Springer-Verlag, 2002, To appear.

[3] G. Barthe, J. Hatcliff, M. Sørensen, « An induction principle for Pure Type Systems », *Theoretical Computer Science 266*, septembre 2001, p. 773–818.

[4] G. Barthe, J. Hatcliff, M. Sørensen, « Weak Normalization implies Strong Normalization in Generalized Non-Dependent Pure Type Systems », *Theoretical Computer Science 269*, octobre 2001, p. 317–361.

[5] L. Théry, « A Machine-Checked Implementation of Buchberger's Algorithm », *Journal of Automated Reasoning 26*, 2001, p. 107–137.

Communications à des congrès, colloques, etc.

[6] A. Amerkad, Y. Bertot, L. Rideau, , L. Pottier, « Mathematics and proof presentation in Pcoq. », *in : Workshop Proof Transformation and Presentation and Proof Complexities (PTP'01)*, juin 2001.

[7] G. Barthe, G. Dufay, M. Huisman, S. M. de Sousa, « Jakarta : a toolset to reason about the JavaCard platform », *in : Proceedings of e-SMART'01*, I. Attali, T. Jensen (éditeurs), *Lecture Notes in Computer Science, 2140*, Springer-Verlag, p. 2–18, 2001.

[8] G. Barthe, G. Dufay, L. Jakubiec, B. Serpette, S. M. de Sousa, « A Formal Executable Semantics of the JavaCard Platform », *in : Proceedings of ESOP'01*, D. Sands (éditeur), *Lecture Notes in Computer Science, 2028*, Springer-Verlag, p. 302–319, 2001.

[9] G. Barthe, D. Gurov, M. Huisman, « Compositional specification and verification of control flow based security properties of multi-application programs », *in : Proceedings of FtfJP'01*, S. D. et al (éditeur), 2001.

[10] G. Barthe, B. R. Jiménez, « Principal types and semi-full closure for Extended Pure Type Systems (In Spanish) », *in : Proceedings of AGP'01*, L. M. Pereira, P. Quaresma (éditeurs), 2001.

[11] G. Barthe, O. Pons, « Type Isomorphisms and Proof Reuse in Dependent Type Theory », *in : Proceedings of FOSSACS'01*, F. Honsell, M. Miculan (éditeurs), *Lecture Notes in Computer Science, 2030*, Springer-Verlag, p. 57–71, 2001.

[12] G. Barthe, T. Uustalu, « CPS Translating Inductive and Coinductive Types », *in : Proceedings of PEPM'02*, P. Thiemann (éditeur), ACM Press.

[13] Y. Bertot, « Formalizing a JVML verifier for initialization in a theorem prover », *in : Computer Aided Verification (CAV'01)*, *LNCS*, 2102, Springer-Verlag, Paris, France, juillet 2001.

[14] L. Chicli, « Une formalisation des faisceaux et des schémas affines en théorie des types avec Coq », *in : Journées Francophones des Langages Applicatifs*, INRIA, Pontarlier, France, 2001.

[15] M. Daumas, L. Rideau, L. Théry, « A Generic Library for Floating-Point Numbers and its Application to Exact Computing », *in : Theorem Proving in Higher Order Logics : 14th International Conference, TPHOLs'01*, *LNCS*, 2152, Springer-Verlag, Edimbourg, Écosse, septembre 2001.

[16] H.Naciri, L.Rideau, « FIGUE : Mathematical Formula Layout with Interaction and MathML Support. », *in : The Fifth Asian Symposium on Computer Mathematics ASCM'2001*, September 2001.

[17] H.Naciri, L.Rideau, « The Mariage of MathML and Theorem Proving. », *in : Internet Accessible Mathematical Computation Workshop 2001 (IAMC'2001)*, juillet 2001.

[18] N. Magaud, Y. Bertot, « Changement de représentation des structures de données en Coq : le cas des entiers naturels », *in : JFLA'2001*, 2001. Also available as INRIA Research Report RR-4039, `ftp://ftp-sop.inria.fr/lemme/Nicolas.Magaud/JFLA2001.ps.gz`.

[19] N. Magaud, Y. Bertot, « Changing data structures in type theory : a study of natural numbers », *in : TYPES'2000*, 2001.

[20] A. Mahboubi, L. Pottier, « Elimination des quantificateurs sur les réels en Coq », *in : Proceedings of JFLA'02*, L. Rideau (éditeur), 2002. To appear.

[21] D. Pichardie, Y. Bertot, « Formalizing Convex Hull Algorithms », *in : Theorem Proving in Higher Order Logics : 14th International Conference, TPHOLs'01*, *LNCS*, 2152, Springer-Verlag, Edimbourg, Écosse, septembre 2001.

[22] L. Pottier, « Extraction dans le CCI », *in : Journées Francophones des Langages Applicatifs*, INRIA, Pontarlier, France, 2001.

[23] L. Pottier, « Quotients in the CIC », *in : Theorem Proving in Higher Order Logics : 14th International Conference, TPHOLs'01*, University of Edinburgh, Edimbourg, Écosse, septembre 2001.

Rapports de recherche et publications internes

[24] A. Amerkad, Y. Bertot, L. Rideau, , L. Pottier, « Mathematics and proof presentation in Pcoq. », *Rapport de Recherche n°4313*, INRIA Sophia Antipolis, 2001, `http://www.inria.fr/rrrt/rr-4313`.

[25] L. Chicli, « Formalisation des faisceaux et des schémas affines en théorie des types avec Coq », *Rapport de Recherche n°4216*, INRIA Sophia Antipolis, 2001, `http://www.inria.fr/rrrt/rr-4216`.

[26] M. Daumas, C. Moreau-Finot, L. Théry, « Computer Validated Proofs of a Toolset for Adaptable Arithmetic », *Rapport de Recherche n°4095*, INRIA Rhones-Alpes, janvier 2001, `http://www.inria.fr/rrrt/rr-4095.html`.

[27] H.Naciri, L.Rideau, « Affichage et manipulation interactive de formules mathématiques dans les documents structurés. », *Rapport de Recherche n°4140*, Inria, janvier 2001, `http://www.inria.fr/rrrt/rr-4140.html`.

Divers

[28] G. Barthe, V. Capretta, O. Pons, « Setoids in Type Theory », Submitted, 2000.

[29] G. Barthe, T. Coquand, « On the equational theory of non-normalizing Pure Type Systems », Manuscript, 2001.

[30] G. Barthe, D. Gurov, M. Huisman, « Compositional Verification of Secure Applet Interactions », Manuscript, 2001.

[31] N. Cataño Collazos, M. Huisman, « Formal specification of Gemplus' electronic purse case study », Manuscript, 2001.

[32] L.Théry, « Note sur la mécanisation des preuves en arithmétique flottante », Note de cours, 2001, `ftp://ftp-sop.inria.fr/lemme/Laurent.Thery/prapoutel/paper2.ps.gz`.