INRIA

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

Projet ARENAIRE

Arithmétique des Ordinateurs

Rhône-Alpes

THÈME 2B

Table des matières

1. Composition de l'équipe

Le projet ARÉNAIRE *est un projet commun au* CNRS*, à l'École Normale Supérieure de Lyon et à l'*INRIA*. Il fait partie du Laboratoire de l'Informatique du Parallélisme (*LIP*,* UMR CNRS–ENS *de Lyon–*INRIA *5668) de l'École Normale Supérieure de Lyon. Ce projet est localisé à Lyon dans les locaux de l'*ENS *de Lyon.*

Responsable scientifique
Jean-Michel Muller [Directeur de Recherche au CNRS]

Assistante de projet
Sylvie Boyer [Adjointe Technique de Recherche, 20% sur le projet]

Personnel Inria
Claude-Pierre Jeannerod [Chargé de Recherche, arrive en février 02]
Philippe Langlois [Délégation sur un poste de Chargé de Recherche, jusqu'au 30/09/01]
Nathalie Revol [Délégation sur un poste de Chargé de Recherche]
Arnaud Tisserand [Chargé de Recherche]

Personnel Cnrs
Marc Daumas [Chargé de Recherche]
Gilles Villard [Chargé de Recherche]

Personnel Ens de Lyon
Florent Dupont de Dinechin [Maître de Conférences à l'ENS de Lyon]

Chercheurs doctorants
Sylvie Boldo [Élève de 4ème année de l'ENS de Lyon, début de thèse au 01/10/01]
Nicolas Boullis [Élève de 4ème année de l'ENS de Lyon, début de thèse au 01/10/01]
David Defour [Allocataire MENRT, deuxième année de thèse]
Claire Finot-Moreau [Allocataire et Moniteur MENRT, thèse soutenue le 10/07/01]
Pascal Giorgi [Allocataire MENRT, début de thèse au 01/10/01]

2. Présentation et objectifs généraux du projet

2.1. (Sans titre)

Mots clés : *arithmétique des ordinateurs, circuits VLSI, circuits FPGA, opérateurs à faible consommation d'énergie, calcul entier, calcul approché, nombres à virgule flottante, fonctions élémentaires, fiabilité numérique, précision multiple, arithmétique d'intervalles, calcul formel, corps finis, algèbre linéaire*.

Le projet ARÉNAIRE contribue à l'élaboration et à la consolidation des connaissances dans le domaine de l'arithmétique des ordinateurs. Chaque étude et chaque solution que nous proposons utilise et enrichit ce socle de connaissances tout en dépendant fortement des cibles choisies. On comptera principalement l'implantation matérielle de l'arithmétique (circuits intégrés ou FPGA), le calcul numérique (virgule flottante) ou le calcul très précis voire exact (précision multiple, arithmétique d'intervalles et précision illimitée). Le choix d'une cible fixe souvent la problématique et les contraintes. Fiabilité, précision et rapidité sont les principaux objectifs que l'on retrouve dans les études du projet. Le projet participe à l'amélioration de l'arithmétique disponible pour les utilisateurs, que ce soit au niveau logiciel ou matériel, sur les calculateurs, les processeurs, les circuits dédiés ou enfouis, **etc**. Calculer mieux ne veut pas uniquement dire calculer plus vite ou plus précisément. Nos études portent aussi par exemple sur la fiabilité de codes numériques dans leur ensemble ou sur la prise en compte de paramètres plus technologiques tels que la consommation d'énergie dans les circuits intégrés.

Historiquement, les fabricants de machines ont toujours eu besoin d'intégrer au niveau matériel ou logiciel les fonctions arithmétiques de base pour une précision « moyenne », c'est-à-dire des mots de 8 à 64 bits. L'addition ou la multiplication ont été très étudiées, mais leurs performances sont encore critiques en surface (pour la multiplication) ou en vitesse (pour les deux). La division et la racine carrée sont moins critiques, mais

il reste probablement plus de possibilités pour de nouvelles améliorations. Pour les fonctions élémentaires (sinus, cosinus, arc-tangente, exponentielle, logarithme, **etc**.), les concepteurs ont plutôt travaillé jusqu'à ce jour sur la vitesse ou l'économie des ressources. La recherche sur les algorithmes et les architectures pour la multiplication, pour la division et pour le calcul des fonctions élémentaires ou spéciales est toujours très active. Nous donnerons juste un exemple avec les algorithmes de division des processeurs HP PA-7100, HP PA-8000, AMD 29050, UltraSparc, IBM RS/6000, Itanium qui sont **tous** différents[SL97]. Ceci montre à l'évidence que les solutions sont encore très loin d'être figées. Les membres du projet ont acquis une solide réputation dans ce domaine et ont l'intention de poursuivre leurs travaux dans cette voie.

Concevoir un opérateur matériel n'est pas uniquement une oeuvre d'assemblage. Il faut intégrer à la démarche de nombreuses données ou contraintes technologiques. Avec l'évolution rapide des technologies, il est nécessaire, par exemple, de bien maîtriser les outils pour le placement et le routage afin de réaliser des circuits performants. La consommation d'énergie d'un circuit intégré dépend, entre autres, de son activité et celle-ci dépend de la valeur de ses entrées. Il faut donc se donner des modèles réalistes sur les données pour faire des simulations pertinentes. Le mode de représentation des nombres est ici très important. Certains codages sont utilisés par les algorithmes performants, d'autres permettent de limiter la consommation du circuit.

Les contraintes pour la conception d'opérateurs arithmétiques sur FPGA sont un peu différentes du fait de la granularité de ces circuits. Par exemple, les blocs logiques de base implantent des petites tables permettant de réaliser des fonctions à quatre entrées. Cette spécificité va permettre d'utiliser la base 4 avec un codage de bas niveau assez efficace. Ici encore, les techniques classiques d'optimisation ne sont pas les bonnes.

Pour se convaincre du bon fonctionnement de l'opérateur réalisé, on effectue des tests, exhaustifs quand cela est possible ou on construit la preuve du bon fonctionnement de l'opérateur. Ces approches, relativement bien adaptées à la conception d'opérateurs entiers, ne sont plus recommandables pour la validation d'opérateurs à virgule flottante. Le niveau de détail devient tel que seule une approche très rigoureuse de la preuve peut garantir que l'opérateur n'a pas de faille. Cela nous a amenés à utiliser l'assistant de preuve COQ pour la vérification automatique des preuves formelles de bon fonctionnement de nos algorithmes.

Il faut apporter un soin tout particulier à la spécification formelle des opérateurs dans l'assistant pour qu'elle reflète fidèlement la sémantique usuellement associée. Cela n'est pas toujours simple parce que les opérateurs, déjà complexes dans leur fonctionnement « normal », doivent de plus gérer correctement des situations exceptionnelles (division par zéro, **etc**). Cependant, la sémantique des opérations est désormais bien définie.

En effet, l'arithmétique des ordinateurs a connu un changement majeur en 1985 avec l'apparition de la norme IEEE-754, qui spécifie les formats de représentation des nombres et les opérations arithmétiques en virgule flottante. Cette norme s'est rapidement imposée. Un nombre est représenté par un triplet (s, n, e) où s est le bit de signe, n est appelé la mantisse ou la fraction selon les cas, et e est appelé l'exposant. La valeur représentée est :

$$(-1)^s \times n \times 2^e.$$

La somme, le produit, le quotient de deux nombres qui s'écrivent exactement avec la notation à virgule flottante ne sont pas, en général, représentables exactement avec cette même notation si on limite la longueur des champs utilisés pour la mantisse et l'exposant. Il faut les **arrondir**. La norme IEEE-754 spécifie que l'utilisateur choisit parmi quatre modes d'arrondi : au plus près, vers zéro, vers le haut, et enfin, vers le bas. L'arrondi au plus près est le mode par défaut. Les deux derniers modes servent de support matériel à l'arithmétique d'intervalles.

Si a et b sont les entrées de l'opérateur $\star$, une quelconque des opérations normalisées, la sortie de $a \star b$ est le nombre à virgule flottante que l'on obtient en faisant le calcul exactement avant de l'arrondir avec le mode d'arrondi demandé par l'utilisateur. Cette exigence est appelée « arrondi correct ». Il n'est pas nécessaire de faire un calcul exact pour obtenir l'arrondi correct pour l'addition, la multiplication, la division et la

racine carrée. Coonen[Coo78] utilise seulement trois bits supplémentaires par rapport à la taille des nombres manipulés.

Avant l'adoption de la norme IEEE, il était assez difficile de savoir exactement ce que faisait une machine. Maintenant, un algorithme peut être analysé jusqu'au dernier bit de précision du résultat. Les nombres à virgule flottante ne sont plus seulement des représentations approchées des réels, mais un type complet avec une sémantique précise. Kahan[Kah96] et ses élèves à l'Université de Californie à Berkeley donnent des algorithmes et des preuves de correction qui utilisent cette spécification. Comme la plupart des ordinateurs suivent la norme, ces algorithmes sont de plus portables d'une machine à une autre.

L'exemple suivant illustre les avantages que l'on peut retirer de cette démarche. On prouve que, en l'absence de dépassement de capacité, si x est exactement représentable et si z est une approximation au dernier bit près de $1/x$, alors la suite d'opérations

$$\begin{aligned} \epsilon &= \circ(1 - xz) \\ z' &= \circ(z + \epsilon z) \end{aligned}$$

donne une valeur z' égale à l'arrondi au plus près de $1/x$[CHGM99]. La notation $\circ(a + bc)$ signifie que le calcul $a + bc$ est fait en arrondi au plus près en utilisant un opérateur fusionné de multiplication-accumulation (FMAC). On en déduit un algorithme de division qui calcule l'arrondi correct sans un seul bit de précision supplémentaire pour les calculs intermédiaires. Il est utilisé sur l'architecture IA64, développée conjointement par Intel et HP.

Avec uniquement des opérateurs normalisés à virgule flottante, nous construisons des bibliothèques de calcul en précision étendue et des opérateurs performants (précision, vitesse, **etc**) pour le produit scalaire, les fonctions élémentaires et spéciales, qui ne sont pas inclus dans la norme et ne sont donc pas toujours bien implantés. Dans le cadre de l'ARC « AOC » (Arithmétique des Ordinateurs Certifiée) de l'INRIA, nous travaillons avec des membres des projets LEMME (Nice—Sophia–Antipolis) et SPACES (Lorraine) à la preuve de propriétés sur les nombres à virgule flottante et à la preuve de bon fonctionnement de nos algorithmes.

Notre développement d'une spécification formelle de l'arithmétique à virgule flottante en COQ nous permet de valider nos preuves. La spécification généralise les idées présentées par la norme IEEE et y ajoute certains fonctionnements non normalisés (opérateur FMAC, mode **flush to zero**). Dans le cadre d'un contrat entre l'INRIA et ST Microelectronics, nous avons proposé des algorithmes de division par une constante utilisant les opérateurs disponibles sur un circuit DSP construit par ST.

Quelle que soit la cible, matérielle ou logicielle, le choix du système de numération est important. L'exemple caricatural est celui du système logarithmique où l'on représente un nombre par son logarithme en base 2. Dans ce système, la multiplication et la division sont deux opérations faciles et qui n'introduisent pas d'erreur d'arrondi. L'addition est par contre très difficile à réaliser. Un exemple plus classique est celui des systèmes de numération **redondants (carry-save, borrow-save, etc)** qui sont utilisés à l'intérieur de nombreux multiplieurs et diviseurs. Les entrées comme les sorties des opérateurs sont représentées dans un système usuel, seuls les calculs internes ont recours à ces systèmes « redondants ». Pour un microprocesseur, l'arithmétique à virgule flottante semble incontournable, même si on peut certainement en améliorer les implantations existantes, mais sur des systèmes dédiés à des applications particulières, de nombreux modes de représentation des nombres peuvent s'avérer mieux adaptés. Un troisième exemple est celui du calcul en ligne. C'est un calcul où l'opérateur commence à travailler et à produire des chiffres du résultat sans connaître exactement les données, mais seulement les chiffres les plus significatifs. Le calcul en ligne permet un comportement adaptatif, c'est-à-dire que l'on continue un calcul qui devient de plus en plus précis jusqu'à ce que l'utilisateur soit satisfait de la précision atteinte. Pour faire du calcul en ligne, nous utilisons une notation redondante en numération de position (en matériel) ou une expansion redondante en nombres à virgule flottante (en logiciel).

La maîtrise des erreurs d'arrondi dans les calculs (et de manière plus générale, la fiabilité numérique des systèmes) est un sujet de plus en plus important. On effectue des calculs considérablement plus volumineux que dans les années 1970, alors que les formats de représentation des nombres, et par conséquent la précision

de chaque opération prise individuellement, n'ont presque pas changé. En conséquence, des problèmes de précision apparaissent de plus en plus au niveau de l'application dans son ensemble. Dans de nombreux domaines, l'imprécision de l'arithmétique flottante peut avoir des conséquences tragiques. L'amélioration automatique de la précision des résultats complète la maîtrise de la fiabilité numérique. Les utilisateurs qui rencontrent des problèmes de précision numérique n'ont souvent ni le temps ni le métier pour répondre à ces problèmes difficiles. Ce constat incite à proposer des approches automatiques qui permettent de contrôler et d'améliorer la qualité numérique d'un programme vu comme une « boîte noire ».

Au-delà de la virgule flottante — en précision simple ou étendue — quand il s'agit de s'affranchir complètement des erreurs de calcul, on fait appel à d'autres types d'arithmétiques. Nous nous intéressons d'une part, en collaboration avec le projet SPACES de l'INRIA Lorraine, à l'arithmétique d'intervalle en précision multiple qui permet d'obtenir des encadrements certifiés et précis de solutions. De tels intervalles fournissent des informations de nature exacte, par exemple, quand la question est de majorer ou de minorer le résultat d'un calcul (optimisation globale). Nous nous intéressons d'autre part à des arithmétiques exactes en calcul formel qui permettent de calculer dans des domaines algébriques tels que les corps finis, les entiers à précision illimitée et les polynômes (cryptologie, calculs mathématiques).

À travers le développement d'algorithmes et de bibliothèques de programmes, nous cherchons à appréhender l'impact de ces arithmétiques sur les méthodes de résolution de problèmes. Dans le but de les améliorer, nous voulons cerner leurs possibilités respectives, leurs contraintes associées et leurs limites inhérentes.

3. Fondements scientifiques

3.1. Implantations matérielles de l'arithmétique

3.1.1. Méthodes à base de tables

Lorsqu'une fonction est difficile à calculer mais que son domaine (discret) compte un petit nombre de valeurs, on peut envisager de pré-calculer une fois pour toutes l'ensemble des résultats pour les stocker dans une table. Par la suite, le calcul pourra être remplacé par une lecture de la table. C'est une solution en général rapide mais consommatrice de ressources matérielles. Les algorithmes font donc un usage parcimonieux des tables pour trouver le bon compromis entre la taille des tables et la vitesse d'exécution.

À l'un des extrêmes de ce compromis, on trouve quelques algorithmes récents, dus à Matula et Das Sarma[DSM95], Schulte et Stine[SS99b], Wong et Goto[WG95] et Muller[Mul99a], qui calculent des fonctions arbitraires d'un argument en n'utilisant que des tables et des additions. Leur avantage est la vitesse : le temps de calcul est celui de quelques lectures de tables et quelques additions. La recherche sur ce type d'algorithmes est loin d'être terminée et leur pertinence augmente avec l'intégration des circuits.

3.1.2. Opérateurs arithmétiques pour Fpga

Ces quinze dernières années ont vu l'apparition d'un nouveau type de circuit intégré qui peut concurrencer les microprocesseurs dans certaines applications. Il s'agit des réseaux de cellules reconfigurables (**field programmable gate arrays** ou FPGA). Ces composants sont des matrices formées d'un très grand nombre de cellules élémentaires identiques, chaque cellule pouvant être programmée séparément pour se comporter comme quelques portes logiques et quelques bits de mémoire. Les cellules sont reliées par des connexions également configurables dans un ensemble de topologies possibles.

Un composant peut ainsi réaliser un circuit logique arbitraire avec des performances proches de celles qu'aurait un circuit intégré spécifique. Le coût en est cependant bien moindre puisque le FPGA est un circuit produit en grande série et programmer un FPGA est bien plus simple que concevoir un circuit intégré. On peut reconfigurer le FPGA un nombre quelconque de fois, soit pour corriger ou améliorer sa configuration, soit pour qu'un seul FPGA remplace successivement plusieurs circuits spécifiques au cours de la vie d'une application.

La programmation des FPGA, pour des raisons historiques, s'apparente actuellement plus à de la synthèse de circuits VLSI qu'à de la programmation au sens usuel. Les bibliothèques arithmétiques pour FPGA sont très rudimentaires. Les implantations publiées ou vendues utilisent pour l'essentiel des algorithmes développés

pour le VLSI, alors que les métriques de temps et de surface sont très différentes dans les FPGA. De plus, la souplesse supplémentaire des FPGA (reconfigurabilité et adaptabilité à l'application) est rarement utilisée pour optimiser les opérateurs arithmétiques.

De manière plus pratique, les langages utilisés sont également hérités directement du monde du VLSI. Des langages plus adaptés font leur apparition comme JBits[1] chez Xilinx. Un ensemble de classes Java permet l'intégration simple de composants implantés sur des FPGA et de composants logiciels classiques. À plus long terme, on aimerait que la surcharge d'un opérateur ou d'une fonction dans un programme classique permette de déléguer son exécution à un composant spécifique implanté dans un FPGA.

3.1.3. Opérateurs arithmétiques asynchrones

La quasi-totalité des circuits intégrés numériques actuels sont cadencés par un signal d'horloge distribué sur toute la surface du circuit. La mise en oeuvre de cette synchronisation globale est de plus en plus complexe. Avec l'augmentation des fréquences et la réduction des technologies, des problèmes comme le routage des signaux d'horloge, les marges de bruit, les pics de courant aux fronts d'horloge... deviennent critiques. Une alternative est l'utilisation de circuits asynchrones. Dans ces circuits, les échanges de données sont cadencés localement par un protocole de communication implanté au plus bas niveau. Il n'y a donc plus d'horloge globale.

Les circuits asynchrones offrent une caractéristique particulièrement intéressante en arithmétique des ordinateurs : c'est la possibilité de calculer en temps moyen. En effet, chaque échange de données étant cadencé localement, il est possible de concevoir des opérateurs dont le temps de calcul effectif est totalement dépendant des valeurs des entrées. On caractérise alors le temps de calcul de l'opérateur par son temps moyen de calcul ainsi que certaines autres caractéristiques statistiques relatives à sa distribution. La notion d'optimisation des cas les plus probables permet d'envisager de nouvelles solutions à tous les niveaux.

Les circuits asynchrones offrent d'autres caractéristiques intéressantes comme la mise en veille naturelle, une meilleure répartition du spectre d'émission électromagnétique, moins de bruits sur les signaux d'alimentation. Enfin, dans les technologies fortement sub-microniques, les délais dans les interconnexions voient leur importance croître, ce qui se traduit par une difficulté accrue dans la conception des horloges. L'asynchronisme n'a pas ce problème.

3.2. Arithmétique à virgule flottante

Comme nous l'avons vu, un nombre à virgule flottante est représenté par un triplet (s, n, e) associé à la valeur

$$(-1)^s \times n \times \beta^e,$$

où β est la base du système. Dans la pratique, $\beta = 2$, mais étudier le système indépendamment de la valeur de β permet de mieux comprendre son fonctionnement. Un opérateur arithmétique manipulant des nombres à virgule flottante est plus complexe que le même opérateur restreint aux seuls nombres entiers. Il faut calculer correctement l'arrondi spécifié par l'utilisateur, manipuler à la fois les mantisses et les exposants des opérandes, traiter les divers cas d'exception (infinis, nombres « dénormalisés », **etc**).

Sur de nombreux processeurs, l'unité de calcul à virgule flottante est plus puissante que l'unité de calcul entier. En effet, il s'agit de la « vitrine » que les constructeurs développent et mettent en avant. Il est ainsi important de bien en maîtriser le fonctionnement pour y intégrer des algorithmes ou pour améliorer les performances des applications. Une bonne connaissance des spécificités des normes actuelles peut permettre d'obtenir de nouveaux développements très efficaces en terme de vitesse ou de précision.

3.2.1. Spécifications et preuves formelles

Des « plantages » retentissants ont montré que l'arithmétique à virgule flottante est parfois difficile à manier ou à implanter. Peu d'outils permettent de faire des preuves rigoureuses sur ce type de données. Pourtant, grâce à la norme IEEE-754, les opérations arithmétiques sont complètement spécifiées ce qui permet de construire

[1] http://www.xilinx.com/products/jbits/index.htm.

des preuves d'algorithmes et de propriétés. Mais il est difficile de présenter synthétiquement une preuve face à l'avalanche de cas particuliers générés par ces calculs. La formalisation des preuves, dans le cadre de notre collaboration avec les projets LEMME et SPACES (ARC AOC), permet d'utiliser un assistant de preuve tel que COQ[HKPM97] pour garantir que chaque cas particulier a été envisagé et traité correctement.

Les systèmes tels que COQ permettent de définir de nouveaux objets et de dériver des conséquences formelles de ces définitions. Grâce à la logique d'ordre supérieur, on peut établir des propriétés sous une forme très générale. Par exemple, nous avons utilisé des quantificateurs universels pour établir des propriétés indépendamment de la base d'écriture des nombres à virgule flottante ou pour un mode d'arrondi arbitraire. Les preuves sont construites de façon interactive en guidant l'assistant avec des tactiques de haut niveau. À la fin de chaque preuve, COQ construit un objet interne qui contient tous les détails des dérivations et garantit que le théorème est valide.

3.2.2. Arrondi correct

Dans la norme IEEE-754, l'« arrondi correct » signifie que le résultat d'une opération dont les entrées sont des nombres à virgule flottante représentés exactement en machine doit être celui qu'on aurait si on avait tout d'abord effectué le calcul « exactement » pour ensuite l'arrondir suivant le mode d'arrondi spécifié par l'utilisateur. Cette exigence n'apparaît que pour les quatre opérations arithmétiques et la racine carrée. Les fonctions élémentaires ou spéciales n'ont pas besoin d'être correctement arrondies parce qu'on a longtemps cru qu'une telle exigence était impossible à satisfaire.

Lorsque l'on cherche à évaluer une fonction élémentaire (sinus, cosinus, logarithme, exponentielle, **etc**.), on calcule en fait une approximation très précise de la valeur exacte sur une arithmétique à précision finie. La précision de l'approximation et du format intermédiaire doit être supérieure à la précision qui correspond au format du résultat final. On arrondit ensuite l'approximation vers le format final, en suivant le mode d'arrondi spécifié par l'utilisateur. La question qui se pose à la personne qui veut utiliser cette méthode est « Quelle est la précision du calcul intermédiaire suffisante pour que l'arrondi de l'approximation coïncide toujours avec l'arrondi du résultat exact ? ». Résoudre ce problème permettrait de fournir des environnements où toutes les primitives numériques, y compris les fonctions élémentaires, sont complètement spécifiées.

Nous avons des bornes raisonnables sur la précision intermédiaire pour certaines fonctions algébriques. Nous avons trouvé les « pires cas », c'est-à-dire les valeurs demandant la plus grande précision lors des calculs intermédiaires, pour certaines fonctions élémentaires en arithmétique à virgule flottante en double précision. Notre travail consiste maintenant à mettre au point des bibliothèques de calcul de ces fonctions avec arrondi correct.

3.2.3. Contrôle et correction des erreurs d'arrondi

Les erreurs d'arrondi introduites par chaque opération arithmétique modifient le comportement numérique de certains algorithmes. Elles altèrent la précision du résultat final allant parfois même jusqu'à l'invalider. Elles peuvent dégrader la stabilité de certains algorithmes pourtant démontrés stables en arithmétique exacte. Elles peuvent également faire prendre un « mauvais branchement » lors de l'exécution d'un test.

L'erreur globale sur le résultat final peut être approchée en linéarisant l'influence des erreurs d'arrondi élémentaires. La correction linéaire automatique de CENA étend cette technique en **calculant** la linéarisation par différentiation automatique et en évaluant chacune des erreurs d'arrondi élémentaires. On dispose ainsi d'un terme correctif qui, ajouté au résultat final, en améliore la précision.

Cette amélioration n'est pas équivalente à celle qui résulterait d'un passage en précision supérieure. Elle s'applique automatiquement au résultat final, mais pour certains algorithmes on peut aussi choisir de l'appliquer à certaines variables intermédiaires. On change ainsi le comportement numérique de l'algorithme pour avoir par exemple une meilleure stabilité.

Au lieu de mettre en oeuvre des outils automatiques qui travaillent sur un code existant, on peut aussi isoler des procédures clés sur lesquelles nous travaillons pour fournir une analyse très fine et des améliorations **ad-hoc**. Le produit scalaire, l'évaluation d'un polynôme, l'ensemble des routines définies par les BLAS ou les filtres numériques constituent des routines de ce type.

3.2.4. Bibliothèques de fonctions élémentaires

Plusieurs bibliothèques de calcul des fonctions élémentaires sont actuellement disponibles. Des constructeurs, tels Sun ou Compaq, dont les processeurs n'intègrent pas ces fonctions au niveau matériel les proposent avec leurs compilateurs. Les développeurs de Linux ont eux aussi proposé plusieurs implantations de la bibliothèque mathématique qui contient les fonctions élémentaires. Aucune de ces fonctions n'est assez précise pour fournir toujours l'arrondi correct. Elles respectent toutefois certaines propriétés telles que la monotonie de la fonction implantée, sa symétrie par rapport à 0, s'il y a lieu, et parfois son domaine d'arrivée.

Nous venons de voir que l'implantation d'une bibliothèque correctement arrondie de fonctions élémentaires en double précision demande le calcul d'une approximation plus précise que la double précision. Plusieurs bibliothèques seraient capables de faire ces calculs. On trouve les deux bibliothèques en précision double-double de K. Briggs[2] et de D. Bailey[3] ainsi que de nombreuses bibliothèques en précision paramétrable, dont la plus fiable semble être MPFR dont nous reparlerons plus loin.

L'implantation d'une fonction élémentaire se décompose en trois phases : la réduction d'argument, le calcul d'une approximation et la reconstruction du résultat. Nos travaux nous permettent d'attaquer chacun de ces problèmes séparément ou dans leur ensemble pour proposer de nouvelles bibliothèques de calcul des fonctions élémentaires ou une bibliothèque de calcul de ces fonctions correctement arrondies en double précision.

3.2.5. Algèbre polynomiale en précision finie

La puissance de calcul et l'omniprésence de l'arithmétique à virgule flottante favorisent l'exécution en arithmétique finie d'algorithmes intrinsèquement exacts. Un problème classique est celui du PGCD de deux polynômes. L'arrondi de représentation sur les coefficients des polynômes semble déjà à lui seul détruire toutes les propriétés attendues. En fait, on définit une nouvelle notion de PGCD adaptée à la précision finie.

Supposons par exemple que le polynôme p divise le polynôme q dans $\mathbb{R}$. Une perturbation arbitrairement petite d'un coefficient de p produit presque certainement un polynôme $\hat{p}$ premier avec q. Deux problèmes similaires, **a priori** moins difficiles, sont la divisibilité et la primalité de deux polynômes.

La détermination du PGCD de deux polynômes est donc un problème mal posé qu'il faut régulariser. Essayons de formuler correctement le problème à résoudre. Chercher le PGCD « de l'arithmétique exacte » n'a pas de sens car la machine ne connaît même pas les polynômes p et q initiaux. Si $\hat{p}$ et $\hat{q}$ sont la donnée du problème informatique, on cherchera un polynôme de degré maximum qui divise deux polynômes voisins $\tilde{p}$ et $\tilde{q}$ obtenus par une perturbation acceptable sur chacun des coefficients des polynômes d'origine $\hat{p}$ et $\hat{q}$. On parle alors d' $\epsilon-$PGCD. D'autres questions découlent du problème ainsi posé, comme le choix des critères de proximité (coefficients ou racines), les éventuelles restrictions de perturbations, le conditionnement de ces problèmes...

Plusieurs notions d' $\epsilon-$PGCD, d' $\epsilon-$divisibilité ou d' $\epsilon-$primalité sont proposées par la communauté du calcul symbolique, certaines associées à des algorithmes de détermination[KL98][BL98]. Ces notions répondent-elles au problème posé en précision finie ? Quelle est la fiabilité de ces algorithmes en précision finie ? Nous proposons d'étudier ce type de questions en privilégiant une approche plus numérique qui utilise l'analyse inverse et ses outils (conditionnement, erreur inverse, régularité, pseudo-solutions).

3.3. Algorithmes et arithmétiques

Les besoins en arithmétiques des ordinateurs pour le calcul scientifique sont variés et dépassent largement le cadre de l'arithmétique flottante en simple ou multiple précision. Pour des calculs garantis et l'encadrement sûr de solutions on utilise des arithmétiques d'**intervalles**. On manipule des données imprécises issues de mesures physiques, on cherche à optimiser des critères ou à résoudre un ensemble de contraintes... Quand on attend une solution essentiellement exacte, on utilise le calcul formel et ses arithmétiques à **précision « illimitée »** ou **symboliques** (cryptologie, combinatoire, géométrie...).

[2] http://www.btexact.com/people/briggsk2/doubledouble.html.
[3] `http://www.nersc.gov/ dhbailey/mpdist/qd/qd.html`.

Les grands systèmes généralistes tels que MATLAB ou MAPLE offrent depuis quelques années la possibilité de « jongler » avec plusieurs de ces modes de calcul. Le fonctionnement de ces systèmes est convaincant et ceux-ci ont un impact substantiel sur la qualité des résultats obtenus. On atteint cependant leurs limites quand il s'agit de développer des modules logiciels hautes-performances facilement réutilisables comme composants d'autres ensembles ou quand il s'agit aussi de tirer parti d'outils externes existants. C'est dans cette démarche de développement de bibliothèque et de réutilisation de code que nous nous inscrivons.

L'utilisation de l'algèbre linéaire s'est généralisée dans de nombreuses branches scientifiques. En particulier dans le domaine du calcul exact, de nombreux progrès reposent de manière cruciale sur la facilité d'emploi, l'efficacité et la réutilisation de fonctionnalités de base sur les matrices. D'un point de vue applicatif, les techniques modernes, par exemple en cryptologie[Odl00], en résolution d'équations polynomiales[Mou98][Fau99] ou en topologie algébrique [DSV01] (formes normales), se fondent sur des représentations matricielles de grandes dimensions et structurées (matrices creuses, faible rang de déplacement, résultants...). Ce sont ces aspects qui nous intéressent principalement. Une facette « plus théorique » concerne la complexité et la recherche de nouveaux algorithmes. Une facette « plus pratique » veut diffuser les acquis algorithmiques et améliorer leurs implications logicielles par le développement de bibliothèques. Dans ce contexte ce sont les arithmétiques dans des corps finis, des arithmétiques de nombres rationnels ou algébriques et des arithmétiques de polynômes qui sont mises en jeu.

Nos collaborations principales sont avec les projets INRIA APACHE (développement C++), SPACES (arithmétique multi-précision) et des contacts avec GALAAD (algèbre linéaire). Au plan international nous participons à l'action LINBOX. Tout en menant des recherches algorithmiques dans les domaines privilégiés que sont l'**algèbre linéaire** exacte et l'**optimisation globale** avec ou sans contraintes, nous nous intéressons à fournir des fonctionnalités arithmétiques spécifiques (mixtes intervalles/multi-précision, corps finis...), à l'utilisation et à l'impact d'arithmétiques de diverses provenances.

3.3.1. Bibliothèques pour l'arithmétique d'intervalles en précision multiple

Les calculs actuels impliquent de plus en plus d'opérations et donc d'erreurs. On atteint par exemple les limites de validité théorique de certains algorithmes numériques. Pour retarder l'occurrence de ces problèmes numériques, on peut calculer en multi-précision, mais le nombre de chiffres corrects des résultats reste indéterminé. Avec une arithmétique par intervalles, les résultats sont des encadrements garantis des solutions au sens où elles ne peuvent pas prendre de valeurs à l'extérieur de ces intervalles. Cependant les encadrements fournis peuvent être trop larges pour que l'information calculée soit pertinente, et ce, même si la largeur des intervalles en entrée (qui correspond à l'imprécision sur les données) est la précision machine. Une solution pour obvier à ce problème consiste à augmenter la précision des calculs.

Le développement d'une bibliothèque efficace pour l'arithmétique d'intervalles en précision multiple, ainsi que d'une bibliothèque d'algorithmes basés sur cette arithmétique, est l'un des volets de nos travaux. Il faut pour cela disposer d'une arithmétique multi-précision avec arrondis dirigés telle qu'Arithmos[4] ou MPFR développées par l'Université d'Anvers pour la première et par le projet SPACES de l'INRIA Lorraine pour la seconde.

Pour utiliser une telle arithmétique, il faut que les algorithmes implantés puissent tirer parti de la précision adaptable : se pose alors le problème du choix de la précision adéquate. En effet, comme les calculs en grande précision sont coûteux, il faut n'augmenter la précision de calcul que quand cela est nécessaire. Une solution classique consiste, quand la précision s'avère insuffisante, à reprendre tout le calcul avec une précision supérieure[Mül00b]. Les caractéristiques de certains algorithmes permettent de ne recommencer que les derniers calculs. Typiquement, le caractère auto-correcteur de l'algorithme de Newton pour la résolution des systèmes non linéaires autorise un accroissement de la précision au fur et à mesure des calculs. Ce type de propriété est à rechercher pour les algorithmes développés.

[4] http://win-www.uia.ac.be/u/cant/arithmos/.

3.3.2. Algorithmes numériques en arithmétique d'intervalles

Non seulement les algorithmes de résolution de systèmes linéaires ou non linéaires en arithmétique d'intervalles sont différents des algorithmes numériques classiques, mais de nouveaux problèmes peuvent être résolus avec cette arithmétique qui calcule sur des ensembles et non pas sur des nombres. En particulier, le calcul par intervalles permet d'encadrer des quantités globales telles que l'image d'un ensemble par une fonction. Disposer d'informations globales permet également de résoudre le problème de l'optimisation globale d'une fonction continue. Cependant il n'y a pas encore de consensus sur les algorithmes d'optimisation globale basés sur l'arithmétique par intervalles. Essayer et comparer les algorithmes existants et éventuellement proposer de nouveaux algorithmes tirant parti de l'arithmétique par intervalles multi-précision constitue l'autre volet de nos travaux sur l'arithmétique d'intervalles.

3.3.3. Algorithmes de calcul pour l'algèbre linéaire formelle

Les méthodes exactes en algèbre linéaire sont en nette évolution depuis quelques années. Cela s'illustre par la complexité des problèmes de base qui évolue encore : celle du calcul du polynôme caractéristique sur un corps abstrait K ou sur un corps fini pour des matrices creuses ; celle de la résolution de systèmes ou du calcul du déterminant sur $\mathbb{Q}$; celle des formes normales d'Hermite ou de Smith sur $\mathbb{Z}$ ou $\mathsf{K}[x]$. Nous travaillons sur ces questions.

Au-delà des avancées théoriques, un objectif est d'arriver aussi à de meilleurs résultats en pratique. Ces études s'appuient notamment sur des algorithmes récursifs souvent probabilistes et par blocs, sur la technique des **baby steps / giant steps** ainsi que sur la méthode de Lanczos–Wiedemann et ses variantes.

3.3.4. Arithmétiques accessibles depuis Linbox

L'action internationale LINBOX[5] (voir §6.10 et §8.2) a été démarrée par E. Kaltofen (*North Carolina State University*), B.D. Saunders (Université du Delaware) et G. Villard. Elle est le pendant logiciel des études du paragraphe précédent. L'objectif est l'écriture d'une bibliothèque générique, basée sur le « branchement » à la demande (**plug-and-play**) de composants externes (**plug-in**).

Cette bibliothèque (cf. §5.6) est spécialisée pour l'algèbre linéaire exacte creuse et structurée. Le langage C++ est utilisé pour le noyau de la bibliothèque. Les arguments ayant conduit à ce choix sont les avantages du langage (héritage, généricité, diffusion, efficacité) ainsi que les expériences respectives des équipes participantes avec leurs précédents prototypes de bibliothèques (GIVARO[6] pour ce qui nous concerne).

Les performances de la bibliothèque sont particulièrement sensibles aux arithmétiques sous-jacentes. Pour les corps finis et leurs extensions algébriques ainsi que pour les entiers et les polynômes, nous expérimentons plusieurs bibliothèques (y compris certaines qui nous sont propres) et poursuivons le développement.

5. Logiciels

5.1. Opérateurs arithmétiques pour Fpga

Participants : Florent de Dinechin, Jérémie Detrey.

Mots clés : *coeurs arithmétiques*, *opérateurs arithmétiques*, *FPGA*, *JBits*, *Virtex*.

F. de Dinechin et J. Detrey ont écrit en JBits les composants arithmétiques suivants pour FPGA Xilinx Virtex : multiplication par une constante, évaluation de fonctions arbitraires en précision de 8 à 24 bits. Ils sont distribués à l'adresse

http://www.ens-lyon.fr/ fdedinec/recherche/jbits/.

5.2. Boîte à outils pour le calcul en précision multiple avec l'arithmétique flottante

Participants : Sylvie Boldo, Marc Daumas, Claire Moreau-Finot.

[5] http://www.linalg.org.
[6] http://www-apache.imag.fr/software/givaro.

Mots clés : *virgule flottante, précision multiple, expansion, adaptatif.*

Le calcul en précision multiple réalise des opérations sur des nombres dont la précision est plus grande que celle implantée en machine. Dans la plupart des applications, on stocke un nombre en précision multiple grâce à un tableau de nombres entiers d_i, que l'on pourra appeler chiffres, et que l'on interprète dans une grande base β, par exemple $\beta = 2^{32}$. Le tableau a alors pour valeur

$$\sum d_i \beta^i \qquad \text{avec} \qquad d_i \in \{0, \cdots \beta - 1\}.$$

Par contre, dans notre travail, on manipule un nombre stocké comme un tableau de nombres à virgule flottante c_i. On interprète le tableau par

$$\sum c_i \qquad \text{avec} \qquad c_i = n_i \times 2^{e_i} \qquad \text{où} \qquad (n_i, e_i) \in \mathbb{Z}^2.$$

Les opérateurs ne manipulent plus des nombres entiers, mais des nombres à virgule flottante. Nous avons continué à étudier les opérateurs de calcul dans ce contexte, pour les rendre plus efficaces et permettre une approche adaptative. Cela signifie que l'opérateur ne calcule pas directement le résultat à partir de toutes les composantes des entrées, mais calcule petit à petit les composantes du résultat en fonction des besoins et des composantes disponibles. Nous avons finalisé la bibliothèque de calcul sur les expansions.

Grâce au travail présenté au point 6.5, notre code en C s'accompagne maintenant de preuves validées en Coq qui permettent de garantir le bon fonctionnement des algorithmes implantés.

5.3. Calcul des fonctions élémentaires en précision multiple

Participants : Marc Daumas, Claire Finot-Moreau.

Mots clés : *virgule flottante, erreurs d'arrondi, fonctions élémentaires.*

Nous avons rendu disponible la bibliothèque de calcul des fonctions élémentaires développée l'année dernière en remplacement des fonctions de la bibliothèque de D. Bailey. Elle se trouve à l'adresse

http://www.ens-lyon.fr/LIP/Arenaire/News/Double-Double/.

L'utilisateur peut télécharger le programme Maple utilisé pour vérifier les étapes de conception ou inclure de nouvelles fonctions.

5.4. Interface C++ pour la bibliothèque MPFR à précision multiple

Participant : Nathalie Revol.

Mots clés : *bibliothèque C++, interface, MPFR.*

MPFR++ est une interface C++ pour la bibliothèque MPFR développée par le projet SPACES de l'INRIA Lorraine. Elle est disponible à l'adresse

http://www.ens-lyon.fr/ nrevol/nr_software.html.

F. Backeljauw, membre du projet Arithmos à l'Université d'Anvers en Belgique, utilise MPFR++.

5.5. Bibliothèque d'arithmétique par intervalles à précision multiple

Participant : Nathalie Revol.

Mots clés : *bibliothèque C et C++, arithmétique par intervalles, MPFI.*

MPFI (*Multiple Precision Floating-point Interval arithmetic library*) est une bibliothèque d'arithmétique par intervalles multi-précision basée sur MPFR, en C, co-développée avec F. Rouillier. Une interface C++, très semblable à MPFR++, sera prochainement construite. Ce développement de logiciels s'effectue en étroite collaboration avec les membres de SPACES. On le trouve également à l'adresse

http://www.ens-lyon.fr/ nrevol/nr_software.html.

F. Rouillier utilise MPFI pour l'isolation des racines réelles de polynômes de grands degrés. Enfin, M. Martel et S. Putot, du LIST/DTSI/SLA/LSL au CEA à Saclay, sont chargés d'une étude de sécurité de logiciels utilisés par Aérospatiale. E. Goubault est le chef de projet et ils utilisent une arithmétique par intervalles multi-précision.

5.6. Algèbre linéaire formelle et arithmétiques adaptées dans la bibliothèque Linbox

Participant : Gilles Villard.

Mots clés : *arithmétique exacte, corps finis, algorithmes probabilistes, méthode de Lanczos–Wiedemann, bibliothèque C++.*

Un prototype de la bibliothèque Linbox (voir la section 3.3) est disponible en interne. Les codes sont en accès partagé (CVS) pour les différents sites impliqués (voir §8.2). Nous avons participé à la conception de l'interface utilisateur. La bibliothèque se veut générique relativement aux matrices utilisées et aux domaines de leurs coefficients. Elle repose sur la notion d'**archétype** (classes virtuelles C++ correspondant au concept d'interface en Java proposées par Kaltofen et Turner) pour représenter l'interface standard et sur la notion d'**adaptateur** — **wrapper** — pour que chaque composant externe souscrive à cette interface standard. Ces aspects ont été stabilisés en 2001, ce qui permet des tests avec des bibliothèques externes telles que ALP[7] ou NTL[8]. Dans le cadre de la thèse de J.G. Dumas (Laboratoire ID – Projet APACHE INRIA) nous avons participé aux expérimentations de pré-conditionnements ainsi qu'au développement de logiciel pour les méthodes de Lanczos et de Wiedemann, scalaires ou par blocs. Ceci a abouti aux résultats expérimentaux cités au §6.10 pour des calculs d'homologie.

6. Résultats nouveaux

6.1. Méthodes à base de tables

Participants : Florent de Dinechin, Arnaud Tisserand.

Mots clés : *évaluation de fonctions, tables.*

Nous avons généralisé et amélioré les méthodes à bases de tables et d'additions pour l'évaluation de fonctions à un argument en précision faible à moyenne (jusqu'à 24 bits). Notre méthode multipartite donne des architectures plus petites et plus rapides qu'avec les méthodes précédemment connues. F. de Dinechin et A. Tisserand ont amélioré les résultats de deux papiers récents[SS99b][Mul99a]. Chacun de ces papiers présentait une méthode d'évaluation des fonctions élémentaires à base de tables et d'additions uniquement. L'unification de ces deux méthodes fait apparaître un espace de recherche plus vaste que ceux étudiés par les auteurs précédents, dans lequel on peut trouver une solution meilleure. Cette méthode unifiée donne ainsi des opérateurs jusqu'à deux fois plus petits que ceux précédemment publiés. Ces résultats ont été présentés à la conférence ARITH 15 [dDT00][dDT01].

Le remplissage de chacune des tables obtenues implique un arrondi et la méthode garantit que la somme de ces erreurs d'arrondi reste inférieure à un bit du résultat final. Cependant, elle utilise les pires cas d'arrondis pour déterminer ces erreurs. Les tables étant petites, il arrive souvent que l'on soit loin du pire cas. F. de Dinechin a encadré un projet étudiant dont l'objet était une interface graphique permettant de rechercher plus finement la solution optimale. Cette interface permet encore une réduction de la taille des tables jusqu'à 10%.

6.2. Opérateurs arithmétiques pour Fpga

Participants : Florent de Dinechin, Arnaud Tisserand, Guillaume Melquiond, Jérémie Detrey.

[7] http://www-sop.inria.fr/galaad/logiciels/ALP.
[8] http://www.shoup.net/ntl.

Mots clés : *FPGA, circuits reconfigurables, opérateurs à virgule flottante IEEE.*

Nous avons implanté et analysé la méthode multipartite sur les FPGA Xilinx Virtex en utilisant le système de développement JBits. Nous avons également mis en oeuvre une bibliothèque VHDL d'opérateurs flottants conformes à la norme IEEE-754. F. de Dinechin a encadré le stage de première année de magistère de J. Detrey visant à implanter les méthodes à base de tables et d'additions sur FPGA (voir section 5.1). Sa spécificité était d'utiliser le système JBits développé par Xilinx. Le composant FPGA fonctionne et J. Detrey a étudié plusieurs méthodes de compression des tables pour l'optimiser. Le résultat de ces travaux est un composant offrant, par exemple, des sinus rapides jusqu'à 20 bits de précision, alors que les bibliothèques constructeur ne vont que jusqu'à 10 bits. Un article est soumis à la conférence RAW 2002.

G. Melquiond, encadré par A. Tisserand, a réalisé dans le cadre de son stage de six semaines en première année du magistère d'informatique, une bibliothèque VHDL synthétisable d'opérateurs en virgule flottante. Un additionneur et un multiplieur ont été testés sur un circuit Virtex. Ces opérateurs sont conformes à la norme IEEE-754 en simple précision sauf pour la gestion des nombres dénormalisés. C'est à notre connaissance la première bibliothèque synthétisable sur FPGA qui gère totalement les problèmes d'arrondi et des valeurs exceptionnelles (infinis, **Not a Number**). La surface obtenue pour les opérateurs de cette bibliothèque est légèrement meilleure que celle des autres travaux du domaine. Pour ce qui est de la vitesse, elle est 30% moins bonne mais en intégrant l'arrondi et la gestion des valeurs exceptionnelles. Il nous reste encore à optimiser la vitesse et nous pensons soumettre ce travail à une conférence dès le début 2002.

6.3. Opérateurs arithmétiques asynchrones

Participants : Nicolas Boullis, Arnaud Tisserand.

Mots clés : *circuits asynchrones, multiplieurs/additionneurs rapides, algorithmes de division.*

Les circuits asynchrones permettent d'obtenir des solutions radicalement différentes de celles utilisées habituellement en arithmétique des ordinateurs matérielle. Dans ce cadre, nous avons travaillé sur la réalisation d'opérateurs arithmétiques pour la multiplication/addition fusionnée et la division. A. Tisserand a étudié la réalisation d'un opérateur asynchrone pour la multiplication et addition fusionnée (capable de faire l'opération $p = a \times b + c$). Ce travail s'est intégré dans une collaboration avec la division **Advanced Designs and Tools** de la société ST Microelectronics à Crolles. Dans ce cadre, un multiplieur/additionneur a été complètement réalisé pour la version asynchrone du processeur ST 20. Les premiers résultats de simulation électrique font état d'une vitesse supérieure à 700 MHz sans aucun étage de pipeline. Les travaux sur le sujet continuent et feront l'objet d'une soumission à la conférence ASYNC 2002.

Lors de son stage de DEA, N. Boullis a étudié l'implantation d'algorithmes de division sur circuits asynchrones. Il a réalisé un simulateur spécifique permettant d'utiliser des descriptions fines au niveau des portes logiques des opérateurs. Il a complètement décrit et comparé toute une gamme d'opérateurs de division basés sur des algorithmes à récurrence sur les chiffres du quotient. Les résultats de cette étude montrent que l'on peut réaliser des diviseurs asynchrones tout aussi rapides qu'en synchrone mais avec un matériel bien moindre. En particulier, le codage redondant des restes partiels n'est pas utile pour obtenir les meilleures performances. Ce travail a été présenté à la conférence SPIE 2001 dans un article écrit en collaboration avec A. Tisserand [BT01a][BT01b].

6.4. Propriétés de l'arithmétique à virgule flottante

Participants : Marc Daumas, Philippe Langlois.

Mots clés : *virgule flottante, correction, erreur d'arrondi.*

Nous présentons de nouvelles propriétés sur l'addition de deux nombres à virgule flottante. Ces propriétés entrent en oeuvre dans la qualité de méthodes correctives telles que CENA. Nous continuons une activité d'exploration des propriétés de l'arithmétique à virgule flottante. Le travail de l'étude présenté en conférence et dans la revue TCS [DL01a][DL01b][DL02] permet de mieux comprendre les limites des méthodes correctives

comme CENA. Il s'agit d'une méthode dont nous reparlons au point 6.7 où le résultat est évalué puis corrigé avec un terme calculé automatiquement. Il apparaît très vite que 0 joue un rôle à part.

Si le résultat exact à trouver est 0, il suffit que la correction $\tilde{x}$ soit opposée au premier résultat approché $\hat{x}$ pour que le résultat final $\circ(\hat{x} + \tilde{x})$ soit juste. Par contre, si le résultat exact est très petit devant le premier résultat approché, la meilleure correction possible est encore l'opposé du résultat approché, et l'on trouve 0 et pas le résultat exact.

On isole trois régions : si le résultat approché et le résultat exact sont très proches, nous sommes dans le cas que l'on pourrait appeler **normal** où la correction compense de petites erreurs de calculs ; si le résultat approché est beaucoup plus petit que le résultat exact, la correction ne pourra être efficace que si le terme correctif est très proche du résultat final, cela signifie que le calcul est en fait réalisé dans la phase de correction ; enfin, si le résultat approché est beaucoup plus grand que le résultat exact, le processus ne peut renvoyer au mieux que 0.

6.5. Formalisation des spécifications et preuves en arithmétique à virgule flottante

Participants : Sylvie Boldo, Marc Daumas.

Mots clés : *virgule flottante, preuve d'algorithmes, erreurs d'arrondi, arithmétique exacte, précision multiple, expansion, division, géométrie algorithmique.*

Nous travaillons à la preuve formelle de propriétés et d'algorithmes en arithmétique à virgule flottante. Ceci est rendu possible grâce aux spécifications fournies par la norme IEEE-754. Grâce à notre collaboration dans le cadre de l'action de recherche coopérative INRIA AOC, et en particulier au travail de L. Théry (INRIA Sophia), nous avons maintenant un certain nombre de définitions et théorèmes sur l'arithmétique à virgule flottante accessibles sous l'outil COQ.

Les résultat sont présentés sous la forme la plus générale possible, par exemple indépendamment de la base ou du mode d'arrondi. La spécification suit un chemin de preuve passant par des résultats intermédiaires connus et documentés dans la littérature. Cette approche nous garantit qu'un lecteur extérieur à notre développement pourra en extraire du savoir sans devoir d'abord se familiariser avec tout notre formalisme.

Utilisateurs des bibliothèques existantes, comme par exemple celle sur les nombres réels, nous avons aussi rendu disponibles certains théorèmes manquants. L'état actuel de nos preuves peut être consulté à l'adresse

`http://www-sop.inria.fr/lemme/AOC/coq/`.

Ces travaux ont été utilisés dans la validation de nos outils pour le calcul en précision multiple avec l'arithmétique flottante (voir section 5.2). Les travaux ont été présentés dans deux conférences dont TPHOL et sont envoyés pour publication dans une revue [DMFT01][DRT01][BD01a].

6.6. Calcul des fonctions élémentaires

Participants : David Defour, Jean-Michel Muller.

Mots clés : *virgule flottante, erreurs d'arrondi, dilemme du fabricant de tables, fonctions élémentaires.*

Nos algorithmes et programmes permettant de construire les pires cas pour le dilemme du fabricant de tables nous ont permis de continuer à obtenir ces pires cas pour la « double précision », pour des fonctions non étudiées l'année précédente. Ces pires cas nous permettent de construire une bibliothèque calculant les fonctions élémentaires avec « arrondi correct ». V. Lefèvre, qui a soutenu sa thèse au sein de notre projet en 2000 et qui est maintenant Chargé de Recherche à l'INRIA Lorraine, a mis au point un algorithme et une « batterie » de programmes permettant de construire les pires cas, pour le format virgule flottante « double précision » du dilemme du fabricant de tables (voir rapport d'activité 2000). Nous avons publié les résultats obtenus à l'occasion de la conférence ARITH 15 [LM01c].

Les pires cas nous permettent de mettre au point des algorithmes rapides de calcul des fonctions élémentaires avec arrondi correct. Un premier algorithme, dédié à la fonction exponentielle, a été proposé par D. Defour et J-M. Muller [DM01a][DM01b].

Lorsque l'on évalue des fonctions trigonométriques, la réduction d'argument est la phase initiale qui permet de se ramener à l'intervalle de convergence de l'algorithme utilisé. V. Lefèvre et J-M. Muller ont proposé un algorithme de réduction d'argument « à la volée », destiné à une implantation matérielle rapide [LM01b], tandis qu'en collaboration avec P. Kornerup (Université d'Odense au Danemark), D. Defour, J-M. Muller et N. Revol travaillent sur des algorithmes de réduction d'argument très précis, destinés à une implantation logicielle garantissant un arrondi correct [DKMR01].

En collaboration avec J.A. Pineiro et J.D. Bruguera, de l'université de Santiago de Compostella (à l'occasion d'une visite au LIP de 3 mois de J.A. Pineiro), J-M. Muller a proposé un petit opérateur matériel dédié à l'évaluation de fonctions [PBM01a][PBM01b].

T. Lang de l'Université de Californie à Irvine et J-M. Muller ont également obtenu des résultats sur le dilemme du fabricant de tables lorsque la fonction calculée est algébrique [LM01a].

6.7. La méthode Cena

Participants : Philippe Langlois, Nathalie Revol.

Mots clés : *Correction linéaire des erreurs d'arrondi, virgule flottante.*

La méthode CENA automatise la correction linéaire des erreurs d'arrondi pour améliorer la précision du résultat et la stabilité de certains algorithmes numériques. Les résultats obtenus cette année permettent une interprétation approfondie de la validité théorique de la méthode, de son efficacité et des limites de son applicabilité. Les résultats obtenus l'an passé concernaient la validation théorique, l'interfaçage avec une autre méthode automatique et l'utilisation de la différentiation automatique pour la méthode CENA. Cette année 2001 a d'abord permis d'approfondir la validité théorique du principe de correction linéaire en précision fixée. Nous prouvons que la correction linéaire en précision fixée u permet de calculer un résultat de précision optimale (u) tant que le conditionnement de la solution n'excède pas $1/\mathrm{u}$. Au-delà, la correction linéaire en précision fixée u se comporte comme un calcul en précision double u^2 [Lan01a]. Ce résultat illustre la différence entre correction à précision fixée et doublement de précision. Nous vérifions ce comportement général sur une famille générique de matrices triangulaires arbitrairement mal conditionnées définie à cet effet dans [Lan01a].

En 2000, nous avons montré l'intérêt complémentaire de la méthode stochastique CESTAC sur laquelle travaillent J-M. Chesneaux et J. Vignes du LIP6 et de la méthode CENA. Nous localisons dynamiquement les parties de code responsables de l'instabilité numérique que nous corrigeons ensuite automatiquement en appliquant la méthode CENA. L'efficacité de cette approche était illustrée sur un calcul de racine polynomiale mal conditionnée. Dans [LR01], nous étendons cette étude à la validation d'un tel résultat en intégrant en particulier l'analyse du calcul par arithmétique d'intervalles en précision fixe et en précision variable grâce à la librairie MPFI (voir point 6.9).

Les résultats de ces trois dernières années nous permettent de présenter la synthèse [Lan01c] des principales propriétés de la méthode CENA dans le cadre plus général de l'augmentation automatique de la précision des résultats.

L'automatisation de la correction linéaire s'appuie sur la différentiation automatique (des parties) du code à corriger. En collaboration avec Th. Braconnier (Université de l'Iowa), nous avons proposé une stratégie de parallélisation adaptée à la correction automatique qui a été implantée cette année en Fortran 90 et MPI. Les premières expérimentations fournissent des **speed up** prometteurs et sont actuellement poursuivies.

6.8. Algèbre polynomiale en précision finie

Participants : Stef Graillat, Philippe Langlois, Gilles Villard.

Mots clés : *ϵ-pseudozéros, ϵ-PGCD, polynômes à coefficients ou racines approchés.*

Nous considérons quelques problèmes d'algèbre polynomiale en précision finie : polynôme le plus proche d'un polynôme donné, primalité et PGCD de polynômes à coefficients ou racines entachés d'une incertitude. Nous étudions comment la notion d' ϵ-pseudozéros peut permettre de répondre à ces problèmes. Les ϵ-pseudozéros

définissent l'ensemble des zéros des polynômes proches (à la distance ϵ) d'un polynôme donné. Peu de travaux ont utilisé cette notion définie par R. Mosier[Mos86] pourtant similaire à celle de pseudospectre largement appliquée en algèbre linéaire numérique. On pourra voir par exemple L. Trefethen[Tre99] pour des références.

S. Graillat, dans son stage de DEA encadré par Ph. Langlois en collaboration avec G. Villard, a étudié l'intérêt des ϵ-pseudozéros pour résoudre quelques problèmes d'algèbre polynomiale en précision finie. Il a généralisé un algorithme de calcul des pseudozéros en norme presque quelconque, ce qui fournit ainsi la forme explicite et simplifiée du polynôme proche cherché par [HKL99]. Cet algorithme permet aussi de déterminer la primalité de deux polynômes à coefficients approchés de façon géométrique et simple. Une note aux CRAS en fin de rédaction présente ces différents résultats.

Ces premiers développements très encourageants vont être poursuivis en prenant en compte l'effet de la précision finie dans les algorithmes de résolution et non plus simplement dans la définition du problème.

6.9. Arithmétique d'intervalles en précision multiple

Participant : Nathalie Revol.

Mots clés : *arithmétique par intervalles en précision multiple, algorithme de Newton par intervalles, adaptation automatique de la précision.*

La bibliothèque MPFI (voir §5.5) d'arithmétique par intervalles en précision multiple permet de revisiter certains algorithmes numériques, en particulier l'algorithme de Newton pour la recherche des racines d'une fonction à une seule variable [Rev01b]. La précision est adaptée automatiquement de façon à calculer sans que les erreurs d'arrondi ne soient prépondérantes et à atteindre la précision requise pour les résultats. Le critère d'arrêt peut être celui que l'on utiliserait avec une arithmétique idéale, puisque l'on peut imposer une précision absolue arbitraire simultanément sur tous les résultats. Enfin les résultats usuels sur la vitesse de convergence s'appliquent : convergence quadratique dans le cas d'une racine simple (si l'évaluation de la dérivée n'est pas nulle) et plus lente sinon [LR01].

Une introduction à l'arithmétique par intervalles et à quelques algorithmes numériques développés pour cette approche a été publiée dans [Rev01a].

6.10. Algèbre linéaire formelle

Participant : Gilles Villard.

Mots clés : *arithmétique exacte, corps finis, algorithmes probabilistes, méthode de Lanczos–Wiedemann.*

Nous avons amélioré la complexité du **calcul du déterminant** d'une matrice de $\mathbb{Z}^{n\times n}$. Pour le coût binaire, la seule borne connue était $O(n^{\omega+1}\log||A||)$ ($\tilde{O}(n^{e_1})$ signifie $O(n^{e_1}\log^{e_2} n)$ pour un certain e_2), cette borne pouvant être vue comme le produit d'un coût arithmétique n^ω (ω étant l'exposant du produit de matrices) par une taille d'entiers $\tilde{O}(n\log||A||)$. Nous avons prouvé que ce produit n'est pas représentatif de la complexité du problème. Après une réduction d'un facteur $\sqrt{n}$ en collaboration avec W. Eberly et M. Giesbrecht [EGV00], nous proposons avec E. Kaltofen un algorithme qui réduit le coût au pire d'un facteur $n^{2/3}$ [KV01]. Autrement dit, cette méthode probabiliste de type Las Vegas a un coût de $\tilde{O}(n^{2.698}\log||A||)$ pour $\omega < 2.376$. Dans le rapport de recherche [SSV01], avec comme co-auteurs B.D. Saunders et A. Storjohann, nous proposons un certificat pour le calcul du rang d'une matrice. Sur un corps réel, ce certificat permet de « transformer » certains algorithmes probabilistes de type Monte Carlo en algorithmes de type Las Vegas. Ceci améliore en particulier — en faisant appel aux résultats précédents sur le déterminant — la complexité du calcul du rang d'une matrice entière.

Sur le thème de l'algorithmique des matrices structurées, avec J.G. Dumas et B.D. Saunders nous avons produit un nouvel algorithme pour le calcul de la **forme normale de Smith** de matrices entières. J.G. Dumas a mené des expérimentations qui en démontrent l'intérêt pratique pour des calculs d'homologie[BBL+99] impliquant des matrices creuses de dimensions jusqu'à plusieurs centaines de milliers [DSV01].

Quant à l'algorithmique des matrices sur des corps finis nous proposons dans un article à six auteurs [CEK+] un tour d'horizon et une classification des méthodes courantes de pré-conditionnement algébrique (modification de propriétés algébriques et non numériques). Nous y avons également poursuivi l'étude et l'amélioration de pré-conditionneurs structurés (Toeplitz, creux...).

Les résultats nouveaux sur le plan logiciel ont été implantés dans la bibliothèque Linbox (voir §5.6).

7. Contrats industriels

7.1. St Microelectronics

Participants : Jean-Michel Muller, Arnaud Tisserand.

Mots clés : *algorithmes de division.*

Suite au contrat avec la division compilation de ST Microelectronics commencé en 2000, notre travail sur les algorithmes de division pour les processeurs DSP a été intégré dans la version de juin 2001 du compilateur vendu par ST Microelectronics pour le DSP ST 120. Nous travaillons toujours sur ce projet. Un nouveau contrat est en préparation. Un article commun sur ce sujet sera prochainement soumis pour publication.

7.2. Posic sa

Participant : Arnaud Tisserand.

Dans le cadre d'un contrat entre l'INRIA et la société POSIC SA (Neuchâtel, Suisse), nous avons étudié et implanté sur circuits programmables FPGA un algorithme d'interpolation pour un capteur de position. Cet algorithme est basé sur des primitives arithmétiques spécialement conçues et optimisées pour ce problème. Le travail réalisé dans ce contrat sera utilisé par la société POSIC SA pour une pré-série de ces produits.

7.3. Xilinx

Participants : Florent de Dinechin, Arnaud Tisserand.

Lors d'une nouvelle campagne de donation, la société Xilinx a donné à l'ENS Lyon un nouveau circuit FPGA (valeur d'environ 15kF) pour les travaux de recherche du projet Arénaire. La contrepartie est la publication sur le web des réalisations effectuées sur ce circuit.

7.4. Itanium (Hp-Intel) conjointe avec le projet Spaces, Inria-Lorraine

Participants : David Defour, Jean-Michel Muller, Nathalie Revol.

Une réponse commune à l'équipe ARÉNAIRE et au projet SPACES de l'INRIA Lorraine a été élaborée pour l'appel lancé par HP et Intel, pour mettre à disposition des projets retenus des ordinateurs équipés de processeurs Itanium IA 64. Elle porte sur l'évaluation de la qualité et des performances des calculs flottants et entiers (via les bibliothèques MPFR et MPFI) et sur une proposition de bibliothèque, dédiée à ce processeur, de calcul avec arrondi correct des fonctions élémentaires. Notre proposition a été acceptée en septembre 2001, et nous avons obtenu deux machines (une pour ARÉNAIRE et une pour SPACES) à base d'Itanium.

7.5. Itanium (Hp-Intel) conjointe avec l'Université de Perpignan

Participant : Philippe Langlois.

Mots clés : *IA-64, haute précision, correction automatique.*

Un second projet a été proposé à l'appel d'offre Itanium (HP-INTEL) dans le cadre d'une collaboration scientifique entre l'équipe Arénaire et l'Université de Perpignan que Ph. Langlois rejoint à partir du 01/10/01. Il s'agit d'étudier l'impact de certaines nouvelles facilités de cette famille de processeurs (instruction FMA et support 64 bits) sur le calcul des erreurs d'arrondis élémentaires et son application à la méthode CENA. Cette

proposition a été acceptée en septembre 2001, et nous avons obtenu une machine à base d'Itanium qui sera utilisée à Perpignan.

8. Actions régionales, nationales et internationales

8.1. Aci Jeunes Chercheurs

Participants : Florent de Dinechin, Arnaud Tisserand.

Grâce au financement accepté en 2000 sur une « ACI jeunes chercheurs » du MENRT, nous avons acheté une machine rapide et des outils de CAO pour nos recherches sur le thème : arithmétique sur FPGA.

8.2. Action Linbox

Participants : Gilles Villard, Pascal Giorgi.

Mots clés : *algèbre linéaire, corps finis, bibliothèque C++.*

En collaboration internationale dans le groupe LINBOX (http://www.linalg.org) de onze chercheurs sur cinq sites (Calgary, Lyon–Grenoble, Newark, Raleigh, Waterloo–London), nous menons nos recherches sur le développement d'algorithmes efficaces en algèbre linéaire, sur leur traduction en une bibliothèque de programmes et sur l'interfaçage de cette bibliothèque avec les logiciels de calcul scientifique les plus couramment utilisés. Le projet a démarré grâce à une action incitative CNRS / NSF (1998-2000). Cette première phase a conduit à des échanges de documents et de codes (CVS, WEB internes), au développement d'un prototype de bibliothèque interne, en l'organisation de rencontres de travail régulières (11 rencontres en France ou sur un des sites américains ou canadiens) et à plusieurs publications communes de nouveaux algorithmes. La deuxième phase du projet a débuté courant 2001. Elle est soutenue comme action incitative JEMSTIC du CNRS ainsi que par le BQR ENS de Lyon et par la NSF côté nord-américain. Cette deuxième phase a pour objectif de rendre publics les codes LINBOX (voir §5.6).

8.3. Séjour Orcca

G. Villard a été invité à séjourner de février à avril 2001 comme **Orcca Research Chair in Computer Algebra** à l'Université de London, Ontario (http://www.orcca.on.ca/files/research.html). Ce séjour a renforcé les collaborations avec M. Giesbrecht, A. Storjohann et G. Labahn (ORCCA).

8.4. Contrat européen INCO-DAPPI

Participant : Nathalie Revol.

Mots clés : *parallélisme, parallélisation d'applications irrégulières.*

Le contrat européen INCO-DAPPI (**Développement d'Algorithmes Parallèles pour des Problèmes Irréguliers**, du 01-11-1998 au 31-10-2001) a été conclu entre les universités de Mons (Belgique, coordinateur du projet : P. Manneback), Lille (France, co-responsable : N. Revol) et Oujda (Maroc, co-responsable : M. Daoudi). Il a pour but de favoriser la dissémination des connaissances et techniques nouvelles en parallélisme auprès du partenaire méditerranéen. Cet échange consiste d'une part en le co-encadrement en thèse de deux étudiants marocains (soutenances 28/09/01) et d'autre part en l'organisation de deux manifestations au Maroc : une école d'automne (octobre 1999) et une conférence (mai 2001).

N. Revol a continué à assurer le suivi de ce projet lors de sa délégation INRIA à Lyon et cela a donné lieu à un rapport de recherche [MEHR01] sur la parallélisation d'une application de reconnaissance automatique de la parole, avec Y. O. Mohamed El Hadj, le doctorant marocain qu'elle co-encadre, ainsi qu'à une présentation [RDMP01] à la conférence PPAM 2001 du travail réalisé avec B. Planquelle, doctorant et participant lillois à ce contrat, sur la parallélisation de l'algorithme de Hansen pour l'optimisation globale par intervalles.

8.5. Action franco-marocaine

J. Della Dora, F. Jung (Laboratoire LMC-IMAG de Grenoble) et G. Villard collaborent avec l'Université Mohamed V (S. El Hajji) et l'Institut National de Postes et Télécommunications de Rabat (A. Hilali) dans le cadre d'une action intégrée MENRT. Soutenu par un accord Tempra Sud-Méditerranée (région Rhône-Alpes) G. Villard co-encadre le travail de A. El Ghazi, doctorant à Rabat, qui a effectué plusieurs séjours au LIP sur le thème de la similitude simultanée de matrices.

9. Diffusion des résultats

9.1. Organisation de conférences, édition de numéros spéciaux de journaux

Participants : Marc Daumas, Jean-Michel Muller, Nathalie Revol, Gilles Villard.

Mots clés : *organisation de conférences, édition de revues, numéros spéciaux.*

- M. Daumas a été membre du comité d'organisation et J-M. Muller a été membre du comité de programme de la conférence SympA'7 (**Symposium en Architectures nouvelles de machines**), Paris, avril 2001 [CD01]. Il sont reconduits dans leur comités respectifs pour SympA'8.
- M. Daumas, F. de Dinechin et A. Tisserand ont été éditeurs invités de la revue **Réseaux et Systèmes Répartis, Calculateurs Parallèles** pour le numéro spécial sur l'arithmétique des ordinateurs, vol 13(4-5).
- J-M. Muller a été membre du comité de programme du *15-th* IEEE *Symposium on Computer Arithmetic* (ARITH-15, Vail, Colorado, Juin 2001). Il est depuis juin 2001 président du **Steering Committee** des IEEE Symposia on Computer Arithmetic.
- un numéro spécial **Real Numbers and Computers**, de **Theoretical Computer Science**, dont les éditeurs invités sont J-C. Bajard (LIRMM, Montpellier), C. Frougny (LIAFA, Paris), P. Kornerup (Univ. Odense, Danemark) et J-M. Muller est en préparation.
- C. Frougny (du LIAFA) et J-M. Muller ont organisé à Prapoutel (Isère), du 26 au 30 mars 2001, une école de Printemps **Arithmétique des Ordinateurs**. 37 personnes ont participé à cette école.
- N. Revol et G. Villard ont été membres des comités d'organisation et de programme de la conférence ALA'2001 (**Algèbre Linéaire et Arithmétique**), Rabat, Maroc, mai 2001 [EHRV01].
- N. Revol est membre du comité d'organisation de la conférence *Numerical Algorithms*, Marrakech, Maroc, octobre 2001 à l'occasion des 60 ans de C. Brezinski (directeur de laboratoire dans l'université de N. Revol).
- G. Villard a été responsable du comité de programme de ISSAC'2001 (**International Symposium on Symbolic and Algebraic Computation**), London, Ontario Canada, juillet 2001. Il est membre du comité de programme de ISSAC'2002. Il a également été membre du comité de programme de CASC'2001 (**Computer Algebra in Scientific Computing**), Konstanz, Allemagne, septembre 2001. Il est éditeur invité au **Journal of Symbolic Computation** pour un numéro spécial (`http://www.ens-lyon.fr/ gvillard/jsc2002.html`).

9.2. Enseignement de 3ème cycle

Depuis 1999-2000, M. Daumas donne un cours sur l'**arithmétique des ordinateurs** et les **opérateurs de l'arithmétique** au DEA d'Informatique Fondamentale de Lyon. En 2000-2001, P. Langlois a donné un cours intitulé **algorithmique numérique et précision finie** commun aux DEA d'Informatique Fondamentale et d'Analyse Numérique à l'ENS de Lyon. N. Revol et G. Villard ont proposé dans ce même cadre un cours intitulé **algorithmique et arithmétiques** et F. de Dinechin et A. Tisserand ont proposé un cours intitulé **conception d'architectures matérielles** pour l'année 2001-2002.

9.3. Autres enseignements et responsabilités

F. de Dinechin assure son service de Maître de Conférences au sein du Magistère d'Informatique et de Modélisation de l'ENS de Lyon. Il est responsable du programme « Europe » du MIM. P. Langlois assure

son service de Maître de Conférence à l'IUT de Perpignan à compter de la fin de sa délégation au 01/10/2001. C. Finot a assuré son service de moniteur à l'IUT Lumière (Lyon 2). D. Defour a effectué des vacations à l'ENS de Lyon et à l'IUT de Bourg en Bresse. M. Daumas, F. de Dinechin et G. Villard ont participé aux concours d'entrée aux Écoles Normales Supérieures.

S. Boldo et N. Boullis ont effectué leur stage de DEA au sein du projet sous la direction respective de M. Daumas et A. Tisserand. S. Graillat a effectué son stage de troisième année de l' ENSIMAG validé comme stage de DEA sous la direction de P. Langlois. J. Detrey et G. Melquiond ont effectué leur stage de première année de magistère sous la direction respective de F. de Dinechin et A. Tisserand. M. Perache et B. Masson ont réalisé un projet de première année de magistère sous la direction de F. de Dinechin.

9.4. Animation de la communauté

J-M. Muller est directeur du LIP (UMR CNRS–ENS-Lyon–INRIA). Le LIP, situé dans les locaux de l'École Normale Supérieure de Lyon, est un laboratoire composé au 7/9/01 de 61 personnes (7 chercheurs INRIA permanents, 2 personnes en délégation et une en détachement à l'INRIA, 6 chercheurs CNRS, 13 enseignants-chercheurs dont 11 ENS Lyon, 20 doctorants, 1 post-doc, 2 ingénieurs CNRS, 1 ingénieur MENRT, 2 ingénieurs experts INRIA, 3 assistantes (2 MENRT et 1 INRIA)).
Il est aussi responsable du thème « Arithmétique » du PRC-GDR ARP.
J-M. Muller est membre élu de la Commission d'Évaluation de l' INRIA. Il est également membre des Commissions de Spécialistes de 27ème section de l'ENS de Lyon et de l'Université de Provence.

M. Daumas a participé à l'organisation de journées de travail (2 jours chaque fois) à Montpellier en janvier et à Paris en décembre, dans le cadre du thème « Arithmétique » du PRC-GDR ARP.
Il est membre assesseur de la Commission de Spécialistes de 27ème section de l'ENS de Lyon.

N. Revol est membre des Commissions de Spécialistes de 26ème section de l'Université Joseph Fourier de Grenoble et de l'Université des Sciences et Technologies de Lille (suppléante).

A. Tisserand est responsable des moyens informatiques du LIP.

G. Villard est membre suppléant de la Commission de Spécialistes de 26ème section de l'Université des Sciences et Technologies de Lille.

9.5. Participations à des jurys

J-M. Muller a participé aux jurys de thèse d'A. Birebent à l'Université Grenoble 1 en mai, d'A. Surarerks à (rapporteur) l'Université Paris 6 en juin, de C. Brunie à l'Université de Limoges en juillet, de C. Finot [MF01] (directeur de thèse habilité) à l'ENS de Lyon en juillet et de F. Rico à l'Université de Montpellier en juillet. Il a aussi participé aux jurys d'habilitation à diriger des recherches de M. Daumas [Dau01] et de P. Langlois [Lan01b] à Lyon en juillet.
En tant que membre de la commission d'évaluation de l' INRIA, J-M. Muller a participé aux jurys d'admissibilité des concours CR2 et DR2 de l' INRIA. Il a également fait partie du jury d'admission CR2 et DR2.

M. Daumas a participé aux jurys de thèse de C. Finot (directeur de thèse non habilité) [MF01] à l'ENS de Lyon en juillet et de Y. Dumonteix (rapporteur) à l'Université Paris 6 en octobre.

A. Tisserand a participé au jury de thèse de J-L. Beuchat (rapporteur) à Lausanne en juin.

N. Revol a participé au jury de thèse de Y. O. Mohamed El Hadj (co-directrice de thèse) à Oujda, Maroc, en septembre.

G. Villard a participé au jury de thèse de A. Storjohann (rapporteur) à l'ETH de Zurich en novembre 2000 et à ceux de C-P. Jeannerod (directeur de thèse) et J-G. Dumas (co-encadrement) à Grenoble en décembre 2000.

9.6. Participation à des colloques, séminaires, invitations

- J-M. Muller, N. Revol et G. Villard ont été conférenciers invités à la conférence ALA'2001 déjà évoquée.
- M. Daumas, J-M. Muller, N. Revol et A. Tisserand ont présenté les cours respectifs sur sur l'**arithmétique à virgule flottante**, l'**évaluation de fonctions élémentaires** et l'**arithmétique en ligne**, l'**arithmétique par intervalles** et enfin l'**arithmétique asynchrone** à la 29ème école de printemps d'informatique théorique sur l'arithmétique des ordinateurs à Prapoutel en mars.
- N. Revol a présenté l'**arithmétique par intervalles** et F. de Dinechin les **méthodes par table** au séminaire des élèves du Magistère d'Informatique et de Modélisation de l'ENS de Lyon.
- A. Tisserand a présenté un séminaire au laboratoire sur la **basse consommation d'énergie et opérateurs arithmétiques matériels** en mai.
- S. Boldo, N. Boullis, M. Daumas, F. de Dinechin, N. Revol et A. Tisserand sont intervenus pour la Semaine de la Science en 2000. S. Boldo, N. Boullis, M. Daumas, C. Moreau-Finot, N. Revol interviennent en 2001.

10. Bibliographie

Bibliographie de référence

[DM97] éditeurs M. DAUMAS, J.-M. MULLER., *Qualité des calculs sur ordinateur : vers des arithmétiques plus fiables.* Masson, 1997, http://www.ens-lyon.fr/~jmmuller/livre_masson.html

[Hig96] N. J. HIGHAM. *Accuracy and stability of numerical algorithms.* SIAM, 1996, http://www.ma.man.ac.uk/~higham/asna.html

[Mar00] P. MARKSTEIN. *IA-64 and elementary functions: speed and precision.* Prentice Hall, 2000, http://www.markstein.org/

[Mul97] J.-M. MULLER. *Elementary functions, algorithms and implementation.* Birkhauser, 1997, http://www.ens-lyon.fr/~jmmuller/book_functions.html

[vzG99] J. VON ZURGATHEN. *Modern Computer Algebra.* Cambridge University Press, 1999, http://www-math.uni-paderborn.de/mca/

Livres et monographies

[CD01] éditeurs F. CAPPELLO, M. DAUMAS., *7ème Symposium sur les Architectures Nouvelles de Machines.* address Paris, France, éditeurs F. CAPPELLO, M. DAUMAS., 2001, http://www.ens-lyon.fr/LIP/Sympa/Sympa7/

[EHRV01] éditeurs S. EL HAJJI, N. REVOL, G. VILLARD., *Algèbre Linéaire et Arithmétique : Calcul Numérique, Symbolique et Parallèle.* address Rabat, Morocco, éditeurs S. EL HAJJI, N. REVOL, G. VILLARD., 2001.

Thèses et habilitations à diriger des recherche

[Dau01] M. DAUMAS. *Écrire les nombres autrement pour calculer plus vite.* Habilitation à diriger des recherches, Université Claude Bernard, address Lyon, France, 2001.

[Lan01b] P. LANGLOIS. *Précision finie et méthode automatiques.* Habilitation à diriger des recherches, Université Claude Bernard, address Lyon, France, 2001, ftp://ftp.ens-lyon.fr/pub/LIP/Rapports/HDR/HDR2001/HDR2001-01.ps.gz

[MF01] C. MOREAU-FINOT. *Preuves et algorithmes utilisant l'arithmétique normalisée IEEE.* thèse de doctorat, École Normale Supèrieure de Lyon, address Lyon, France, 2001, ftp://ftp.ens-lyon.fr/pub/LIP/Rapports/PhD/PhD2001/PhD2001-03.ps.Z

Articles et chapitres de livre

[BBC+02] A. BARRAUD, C. BREZINSKI, J.-M. CHESNEAUX, P. LANGLOIS, S. LESECQ, G. PLATEAU, A. RENAUD, J. VISCONTI. éditeurs A. BARRAUD., *Outils d'analyse numérique pour l'automatique.* série Traité IC2, Hermès, 2002, chapitre 1, pages 9-43.

[CEK+] L. CHEN, W. EBERLY, E. KALTOFEN, B. SAUNDERS, W. TURNER, G. VILLARD. *Efficient matrix preconditioners for black box linear algebra.* in « Linear Algebra and its Applications, **special issue on** Infinite Systems of Linear Equations Finitely Specified ».

[DdDMre] M. DAUMAS, F. DEDINECHIN, J.-M. MULLER. *Arithmétique et calcul sur machine.* Hermès, À paraître, chapitre Arithmétique flottante.

[DL02] M. DAUMAS, P. LANGLOIS. *Additive symmetric: the non-negative case.* in « Theoretical Computer Science », 2002.

[DM01b] D. DEFOUR, J.-M. MULLER. *Évaluation des fonctions élémentaires.* in « Calculateurs Parallèles », 2001.

[DSV01] J. DUMAS, B. SAUNDERS, G. VILLARD. *On efficient sparse integer matrix Smith normal form computations.* in « Journal of Symbolic Computation, **special issue on** Computer Algebra and Mechanized Reasoning », numéro 1-2, volume 32, 2001, pages 71-99.

[GBVB01] L. GERBAUD, M. BARKATOU, G. VILLARD, A. BOLOPION. *Résolution symbolique d'équations différentielles ordinaires en vue de l'étude de circuits d'électronique de puissance.* in « Revue Internationale de Génie Electrique », numéro 1-2, volume 4, 2001.

[Lan01a] P. LANGLOIS. *Automatic linear correction of rounding errors.* in « BIT Numerical Mathematics », numéro 3, volume 41, 2001, pages 515-539.

[Lan01c] P. LANGLOIS. *Une méthode de correction automatique.* in « Calculateurs Parallèles », 2001.

[LLMS01] D. LAVENIER, D. LITAIZE, J. M. MULLER, P. SAINRAT. *Et demain, quel PC ?.* in « Technique et Science Informatiques », numéro 1, volume 20, 2001.

[LM01b] V. LEFÈVRE, J. M. MULLER. *On-the-fly range reduction.* in « Journal of VLSI Signal Processing », 2001, http://www.wkap.nl/jrnltoc.htm/0922-5773

[Rev01a] N. REVOL. *Arithmétique par intervalles.* in « Calculateurs Parallèles », 2001.

Communications à des congrès, colloques, etc.

[BD01a] S. BOLDO, M. DAUMAS. *A mechanically validated technique for extending the available precision.* in « 35th Asilomar Conference on Signals, Systems, and Computers », address Pacific Grove, California, 2001.

[BD01b] S. BOLDO, M. DAUMAS. *Performances d'implantations de l'addition en précision quad-double sur différentes machines.* in « 7ème Symposium sur les Architectures Nouvelles de Machines », pages 105-112, address Paris, France, 2001, http://www.ens-lyon.fr/LIP/Sympa/Sympa7/15.pdf

[BL01] T. BRACONNIER, P. LANGLOIS. *From rounding error estimation to automatic correction with automatic differentiation.* éditeurs G. CORLISS, C. FAURE, A. GRIEWANK, L. HASCOET, U. NAUMANN., in « Automatic Differentiation: from Simulation to Optimization », pages 351-359, 2001.

[BT01b] N. BOULLIS, A. TISSERAND. *On digit-recurrence division algorithms for self-timed circuits.* éditeurs F. LUK., in « Advanced Signal Processing Algorithms, Architecture and Implementations XI », address San Diego, California, 2001.

[dDT01] F. DEDINECHIN, A. TISSERAND. *Some improvements on multipartite table methods.* éditeurs N. BURGESS, L. CIMINIERA., in « Proceedings of the 15th Symposium on Computer Arithmetic », pages 128-135, address Vail, Colorado, 2001.

[DKMR01] D. DEFOUR, P. KORNERUP, J.-M. MULLER, N. REVOL. *A new range reduction algorithm.* in « 35th Asilomar Conference on Signals, Systems, and Computers », address Pacific Grove, California, 2001.

[DL01b] M. DAUMAS, P. LANGLOIS. *From x to z with finite precision addition.* in « Algèbre Linéaire et Arithmétique : Calcul Numérique, Symbolique et Parallèle », pages 189-194, address Rabat, Morocco, 2001.

[DM01a] D. DEFOUR, J.-M. MULLER. *Correctly rounded exponential function in double precision arithmetic.* éditeurs F. LUK., in « Advanced Signal Processing Algorithms, Architecture and Implementations XI », address San Diego, California, 2001.

[DRT01] M. DAUMAS, L. RIDEAU, L. THÉRY. *A generic library of floating-point numbers and its application to exact computing.* in « 14th International Conference on Theorem Proving in Higher Order Logics », address Edinburgh, Scotland, 2001.

[EGV00] W. EBERLY, M. GIESBRECHT, G. VILLARD. *Computing the determinant and Smith form of an integer matrix.* in « 41st Annual IEEE Symposium on Foundations of Computer Science », pages 675-679, address Redondo Beach, California, 2000.

[KV01] E. KALTOFEN, G. VILLARD. *On the complexity of computing determinants.* in « Fifth Asian Symposium on Computer Mathematics », World Scientific Publishing Company, address Singapore, Mailaisia, 2001.

[LM01a] T. LANG, J.-M. MULLER. *Bound on run of zeros and ones for algebraic functions.* éditeurs N. BURGESS, L. CIMINIERA., in « Proceedings of the 15th Symposium on Computer Arithmetic », pages 13-20, address Vail, Colorado, 2001.

[LM01c] V. LEFÈVRE, J.-M. MULLER. *Worst cases for correct rounding of the elementary functions in double precision.* éditeurs N. BURGESS, L. CIMINIERA., in « Proceedings of the 15th Symposium on Computer Arithmetic », pages 111-118, address Vail, Colorado, 2001.

[PBM01a] J. A. PIÑEIRO, J. D. BRUGUERA, J.-M. MULLER. *Faithful powering computation using table look-up and a fused accumulation tree.* éditeurs N. BURGESS, L. CIMINIERA., in « Proceedings of the 15th Symposium on Computer Arithmetic », pages 40-47, address Vail, Colorado, 2001.

[PBM01b] J. A. PIÑEIRO, J. D. BRUGUERA, J.-M. MULLER. *FPGA implementation of a faithful polynomial approximation for powering function computation.* in « Proceedings of EUROMICRO Symposium on Digital System Design (DSD'2001) », address Warszawa, Poland, 2001.

[RDMP01] N. REVOL, Y. DENNEULIN, J.-F. MÉHAUT, B. PLANQUELLE. *Parallelization of continuous global optimization.* in « Parallel Processing and Applied Mathematics », address Naleczów, Poland, 2001.

[Rev01b] N. REVOL. *Newton iteration using multiple precision interval arithmetic: the univariate case.* in « Algèbre Linéaire et Arithmétique : Calcul Numérique, Symbolique et Parallèle », pages 121-129, address Rabat, Morocco, 2001.

[Vil01] G. VILLARD. *Evaluation du signe et calcul du déterminant d'une matrice à coefficients entiers.* in « Algèbre Linéaire et Arithmétique : Calcul Numérique, Symbolique et Parallèle », pages 147-153, address Rabat, Morocco, 2001, http://www.ens-lyon.fr/~gvillard/BIBLIOGRAPHIE/POSTSCRIPT/ala.ps

Rapports de recherche et publications internes

[BT01a] N. BOULLIS, A. TISSERAND. *On digit-recurrence division algorithms for self-timed circuits.* Research report, numéro INRIA 4221 / LIP 27, 2001, http://www.inria.fr/rrrt/rr-4221.html

[CEK+01] L. CHEN, W. EBERLY, E. KALTOFEN, B. SAUNDERS, W. TURNER, G. VILLARD. *Efficient matrix preconditioners for black box linear algebra.* rapport de recherche, numéro LIP 05, 2001, ftp://ftp.ens-lyon.fr/pub/LIP/Rapports/RR/RR2001/RR2001-05.ps.Z

[dDT00] F. DEDINECHIN, A. TISSERAND. *Some improvements on multipartite table methods.* Research report, numéro INRIA 4059 / LIP 38, 2000, http://www.inria.fr/rrrt/rr-4059.html

[DL01a] M. DAUMAS, P. LANGLOIS. *Additive symmetric: the non-negative case.* Research report, numéro INRIA 4115 / LIP 06, 2001, http://www.inria.fr/rrrt/rr-4115.html

[DMFT01] M. DAUMAS, C. MOREAU-FINOT, L. THERY. *Computer validated proofs of a toolset for adaptable arithmetic.* Research report, numéro INRIA 4095 / LIP 01, 2001, http://www.inria.fr/rrrt/rr-4095.html

[KPV01] P. KOIRAN, N. PORTIER, G. VILLARD. *A rank theorem for Vandermonde matrices.* rapport de recherche, numéro LIP 34, 2001, ftp://ftp.ens-lyon.fr/pub/LIP/Rapports/RR/RR2001/RR2001-34.ps.Z

[LR01] P. LANGLOIS, N. REVOL. *Validating polynomial numerical computations with complementary automatic methods.* Research report, numéro INRIA 4205 / LIP 18, 2001, http://www.inria.fr/rrrt/rr-4205.html

[MEHR01] Y. O. MOHAMED EL HADJ, N. REVOL. *Parallelization of automatic speech recognition.* Research report, numéro INRIA 4110 / LIP 02, 2001, http://www.inria.fr/rrrt/rr-4110.html

[SSV01] B. SAUNDERS, A. STORJOHANN, G. VILLARD. *Matrix rank certification.* rapport de recherche, numéro LIP 30, 2001, ftp://ftp.ens-lyon.fr/pub/LIP/Rapports/RR/RR2001/RR2001-30.ps.Z

Bibliographie générale

[BBL+99] E. BABSON, A. BJÖERNER, S. LINUSSON, J. SHARESHIAN, V. WELKER. *Complexes of not i-connected graphs.* in « Topology », numéro 2, volume 38, 1999, pages 271-299.

[BL98] B. BECKERMANN, G. LABAHN. *A fast and numerically stable Euclidean-like algorithm for detecting relatively prime numerical polynomials.* in « Journal of Symbolic Computation », numéro 6, volume 26, 1998, pages 691-714.

[CHGM99] M. A. CORNEA-HASEGAN, R. A. GOLLIVER, P. MARKSTEIN. *Correctness proofs outline for Newton-Raphson based floating point divide and square root algorithms.* éditeurs I. KOREN, P. KORNERUP., in « Proceedings of the 14th Symposium on Computer Arithmetic », pages 96-105, address Adelaide, Australia, 1999, http://computer.org/proceedings/arith/0116/0116toc.htm

[Coo78] J. T. COONEN. *Specification for a proposed standard for floating point arithmetic.* Memorandum, numéro ERL M78/72, institution University of California, Berkeley, 1978.

[DSM95] D. DAS SARMA, D. W. MATULA. *Faithful bipartite ROM reciprocal tables.* éditeurs S. KNOWLES, W. H. MCALLISTER., in « Proceedings of the 12th Symposium on Computer Arithmetic », pages 17-28, address Bath, England, 1995, http://computer.org/proceedings/arith/7089/7089toc.htm

[Fau99] J.-C. FAUGÈRE. *A new efficient algorithm for computing Gröbner bases F_4.* in « Journal of Pure and Applied Algebra », numéro 1-3, volume 139, 1999, pages 61-88.

[HKL99] M. A. HITZ, E. KALTOFEN, Y. N. LAKSHMAN. *Efficient algorithms for computing the nearest polynomial with a real root and related problems.* in « Proceedings of the 1999 International Symposium on Symbolic and Algebraic Computation », pages 205-212, address Vancouver, British Columbia, 1999, http://www.acm.org/pubs/articles/proceedings/issac/309831/p205-hitz/p205-hitz.pdf

[HKPM97] G. HUET, G. KAHN, C. PAULIN-MOHRING. *The Coq Proof Assistant: A Tutorial: Version 6.1.* Technical Report, numéro 204, institution Institut National de Recherche en Informatique et en Automatique, address Le Chesnay, France, 1997, http://www.inria.fr/rrrt/rt-0204.html

[Kah96] W. KAHAN. *Lecture notes on the status of the IEEE standard 754 for binary floating point arithmetic.* 1996, http://HTTP.CS.Berkeley.EDU/~wkahan/ieee754status/ieee754.ps note : http://HTTP.CS.Berkeley.EDU/ wkahan/ieee754status/ieee754.ps.

[KL98] N. K. KARMARKAR, Y. N. LAKSHMAN. *On approximate GCDs of univariate polynomials.* in « Journal of Symbolic Computation », numéro 6, volume 26, 1998, pages 653-666, http://www.idealibrary.com/links/doi/10.1006/jsco.1998.0232/pdf note : Symbolic numeric algebra for polynomials.

[Mos86] R. G. MOSIER. *Root neighborhoods of a polynomial.* in « Mathematics of Computation », numéro 175, volume 47, 1986, pages 265-273.

[Mou98] B. MOURRAIN. *Computing the isolated roots by matrix methods.* in « Journal of Symbolic Computation », numéro 6, volume 26, 1998, pages 715-738, http://www.idealibrary.com/links/doi/10.1006/jsco.1998.0236/pdf

[Mul99a] J.-M. MULLER. *A few results on table based methods.* in « Reliable Computing », numéro 3, volume 5, 1999.

[Mül00b] N. T. MÜLLER. *The iRRAM: Exact Arithmetic in C++.* éditeurs J. BLANCK, V. BRATTKA, P. HERTLING., in « 4th International Workshop on Constructivity and Complexity in Analysis », pages 222-252, address Swansea, United Kingdom, 2000, http://link.springer.de/link/service/series/0558/papers/2064/20640222.pdf

[Odl00] A. M. ODLYZKO. *Discrete logarithms: the past and the future.* in « Designs, Codes, and Cryptography », numéro 2/3, volume 19, 2000, pages 129-145, http://www.wkap.nl/article.pdf?253937

[SL97] P. SODERQUIST, M. LEESER. *Division and square root: choosing the right implementation.* in « IEEE Micro », numéro 4, volume 17, 1997, pages 56-66.

[SS99b] M. J. SCHULTE, J. E. STINE. *Approximating elementary functions with symmetric bipartite tables.* in « IEEE Transactions on Computers », numéro 8, volume 48, 1999, pages 842-847.

[Tre99] L. N. TREFETHEN. *Computation of pseudospectra.* in « Acta Numerica », numéro 1, volume 8, 1999, pages 247-295.

[WG95] W. F. WONG, E. GOTO. *Fast evaluation of the elementary functions in single precision.* in « IEEE Transactions on Computers », numéro 3, volume 44, 1995, pages 453-457.