Rapport INRIA 1994 — Programme 1

Compilation, Architectures Parallèles et Système

Projet CAPS

3 mai 1995

Projet CAPS

Compilation, Architectures Parallèles et Système

Localisation : *Rennes*

Mots-clés : antémémoire (1, 4), CAPS (1), évaluation de performance (1, 6), Fortran-S (1, 6), hiérarchie mémoire (1, 4–6, 9), KOAN (1, 6), localité (1, 4, 6, 9), mémoire virtuelle partagée (1, 6), microprocesseur (1, 3), migration de tâches (1, 9), multiprocesseur (1, 9), optimisation de code (1, 5), pipeline (1, 5), RISC (1, 3).

Caps est un projet commun Inria/CNRS (URA 227).

1 Composition de l'équipe

Responsable scientifique

André Seznec, DR Inria

Secrétaire

Evelyne Livache, SAR Inria

Personnel Inria

François Bodin, détaché de l'université de Rennes 1 à partir du 1er octobre 1994
Yvon Jégou, CR
Thierry Priol, CR

Personnel Ura 227

François Bodin, maître de conférences à l'université de Rennes 1 (jusqu'au 30 septembre 1994)
Jacques Lenfant, professeur, université de Rennes 1

Ingénieur expert Inria

Mike O'Boyle, du 1er mars au 31 août 1994

Chercheurs doctorants

Gilbert Cabillic, bourse Intel (co-encadrement avec le projet Solidor)
Stéphane Chauveau, bourse MESR (co-encadrement avec le projet Aladin)
Nathalie Drach, bourse MESR, jusqu'au 31 août 1994
Mounir Hahad, boursier BGF (co-encadrement avec le projet Aladin)
Sébastien Hily, bourse Inria-région
Lionel Kervella, bourse MESR
Thierry Montaut, allocataire moniteur normalien, jusqu'au 31 août 1994
Nam Dam Truong, bourse MESR, à partir du 1er octobre 1994
Pierre Rabain, bourse école des Mines de Nantes, (co-encadrement avec le projet Aladin)
Thierry Vauléon, bourse Cifre, Stern Computing Systems (à partir du 1er octobre 1994)

Collaborateurs extérieurs

William Jalby, professeur, université de Versailles
Yves Robin, Boursier Thomson

Autres personnels

Yann Mevel, scientifique du contingent, à partir du 15 septembre 1994
Thierry Vauléon, scientifique du contingent, jusqu'au 31 août 1994

2 Présentation générale et objectifs

Les performances de crête des calculateurs hautes performances croissent régulièrement ; cependant ces niveaux de performances ne sont

aujourd'hui accessibles, sur des applications particulières, qu'à des spécialistes du calcul parallèle. Le projet a pour objectif de rapprocher le niveau effectif des performances de ces calculateurs de leurs limites théoriques, ceci pour le plus grand nombre d'utilisateurs et pour le plus large domaine d'applications.

À cette fin, les recherches menées au sein du projet visent à rendre la hiérarchie mémoire des calculateurs hautes performances le plus transparente possible à l'utilisateur. Pour ce faire, de nouvelles structures matérielles d'antémémoires sont étudiées afin de réduire les pénalités engendrées par les accès à la mémoire principale. D'autre part, la mémoire des calculateurs hautes performances est de plus en plus souvent physiquement distribuée, afin que l'utilisateur n'ait pas à être conscient de cette complexité, des travaux au niveau système visent à lui offrir la vision d'une mémoire globale à travers un mécanisme de mémoire virtuelle partagée. Cependant sans l'aide d'un compilateur exploitant et optimisant la localité des références à la mémoire, les performances des antémémoires comme de la mémoire virtuelle partagée seront toujours décevantes, c'est pourquoi des techniques logicielles sont étudiées pour optimiser la localité dans les applications.

3 Actions de recherche

3.1 Architectures de processeurs

Participants : Nathalie Drach, François Bodin, Sébastien Hily, Yvon Jégou, Yann Mevel, André Seznec, Thierry Vauléon

3.1.1 Veille technologique

Les progrès en architecture hautes performances ont été longtemps tirés par le calcul scientifique ; ceci est aujourd'hui moins net, en particulier depuis l'irruption des microprocesseurs comme briques de base dans les machines parallèles. Aujourd'hui, les microprocesseurs sont utilisés dans un grand nombre de systèmes matériels : stations de travail, multiprocesseurs, systèmes temps réels... L'évolution est extrêmement rapide dans ce domaine au niveau de l'intégration sur le circuit intégré (9.3 millions de transistors sur le DEC 21164), au niveau de la fréquence (300 MHz pour le DEC 21164), au niveau architecture (émergence des Riscs, puis des processeurs superscalaires ou des superpipelines) et au

niveau du logiciel (compilateurs optimiseurs, exploitation de la localité des données).

Depuis début 1991, une activité de veille technologique et de diffusion d'information sur les microprocesseurs a été lancée. Un rapport détaillé comparant les architectures Intel Pentium et PowerPC601 a été diffusé en 1994. Une étude similaire sur les microprocesseurs MIPS R8000, DEC 21164 et IBM Power2 est en cours et sera diffusée courant 1995. La Dret soutient cette activité (convention + mise à disposition d'un scientifique du contingent).

Cette activité de veille technologique permet d'orienter les recherches du projet en architecture de processeurs.

3.1.2 Les antémémoires

Le taux de succès lors des accès à une antémémoire dépend de nombreux facteurs liés à son organisation matérielle et à l'application. Aujourd'hui, la technologie permet de mettre sur un même composant l'unité de calcul du microprocesseur et des antémémoires de petite taille. La place disponible sur le composant limite cette taille et il convient de l'utiliser au mieux.

Antémémoires associatives brouillées L'antémémoire associative brouillée ou *skewed-associative cache* est une nouvelle organisation d'antémémoire partiellement associative. Une antémémoire partiellement associative par ensemble à X voies est physiquement organisée en X bancs de mémoire. Dans une antémémoire associative par ensemble, un mot d'adresse A peut occuper n'importe laquelle des X lignes d'adressse $f(A)$ dans ces X bancs. Une seule fonction f détermine un ensemble de cases où peut être rangée une donnée : si sur X+1 adresses (A_j), la fonction f est constante, alors les X+1 données correspondantes ne peuvent pas toutes cohabiter au même instant dans l'antémémoire. Dans une antémémoire associative brouillée à X voies, un mot d'adresse A peut occuper dans chacun des bancs une et une seule place : la case d'adresse $f_i(A)$ dans le banc i de l'antémémoire. Les fonctions f_i utilisées pour indexer les différents bancs peuvent être différentes. Des simulations ont montré qu'une antémémoire associative brouillée à 2 voies exhibe le même taux de succès qu'une antémémoire associative par ensemble à 4 voies, mais a la complexité matérielle d'une antémémoire associative par ensemble à 2 voies.

Une étude a montré que, contrairement aux antémémoires associatives par ensembles, le comportement des antémémoires associatives brouillées est très peu sensible aux placements relatifs des données en mémoire ; la performance d'une application avec un placement déterminé des données en mémoire est donc représentative de la performance que l'on obtiendra quel que soit le placement relatif des données en mémoire, par exemple lors d'un changement de taille du problème.

Antémémoires à secteurs découplés Les antémémoires contiennent deux types d'information : les données et les étiquettes.

Depuis plus de 20 ans, les antémémoires à secteurs sont utilisées pour réduire le volume des étiquettes. Un secteur comprend plusieurs lignes, mais une seule étiquette est associée à tout un secteur. Ce type d'antémémoire offre un compromis de conception entre le volume réduit d'étiquettes permis par l'utilisation de lignes longues et le trafic mémoire réduit permis par l'utilisation de lignes courtes au prix d'une dégradation du comportement global.

L'antémémoire à secteurs découplés est une nouvelle solution pour réduire le volume des étiquettes. Ce type d'antémémoire réconcilie volume réduit d'étiquettes et faible taux d'échec.

Antémémoires à lignes virtuelles Les gains apportés par l'utilisation des antémémoires sont liés à deux types de localité :

- la localité spatiale : des éléments proches dans l'espace d'adressage sont accédés dans un délai relativement court.
- la localité temporelle : le même élément est accédé plusieurs fois dans un délai relativement court.

Les lignes d'antémémoire de grande taille favorisent la localité spatiale par un effet de préchargement alors que les lignes de petite taille favorisent la localité temporelle en limitant les interférences.

Le système d'antémémoire à lignes virtuelles permet l'exploitation simultanée des deux types de localité. Il est basé sur l'utilisation d'une antémémoire primaire à lignes longues et d'une petite antémémoire secondaire fortement associative à lignes plus courtes par laquelle transitent les données rejetées de l'antémémoire primaire.

3.1.3 Gestion-optimisation du pipeline

Les performances des processeurs dépendent à la fois du temps de cycle du processeur, de l'organisation interne des unités fonctionnelles et des performances de la hiérarchie mémoire. Ces performances dépendent en outre de la prise en compte au niveau de la génération de code des caractéristiques fines de la machine.

Les travaux sur l'outil de transformation de code assembleur (OCO) se poursuivent cette année. Ces travaux sont effectués en collaboration avec le projet Api. Cet outil permet, à partir d'une description du format de l'assembleur et de l'architecture cible, de mettre en œuvre des techniques d'ordonnancement de code tel le pipeline logiciel. OCO est actuellement en cours d'extension pour permettre une description simple des architectures superscalaires.

Cet outil associé à un simulateur d'architectures superscalaires a permis d'entreprendre des études sur des choix d'architectures :

- utilisation d'antémémoires pipelinées associatives ou à correspondance directe ;
- position relative des étages d'exécution des unités entières, flottantes et d'accès à la mémoire.

3.1.4 Parallélisme sur le composant

D'ici quelques années, plusieurs processeurs pourront être réunis sur un même composant. De plus, le fossé relatif entre horloge interne et horloge système ne cessera de s'accroître, rendant de plus en plus coûteux les accès externes. Parmi les solutions pour exploiter ces nouvelles données technologiques, le *multiflot* semble l'une des méthodes les plus efficaces. Le *multiflot* est basé sur l'exécution de plusieurs flots d'instructions indépendants. Lorsqu'un flot est bloqué, par exemple en attente d'un accès externe long, le processeur commute sur un autre flot. De nombreux paramètres entrent en jeux dans la conception d'une architecture *multiflot* : nombre de supports physiques de flots, type et partage des ressources, hiérarchie mémoire, etc .

Un simulateur va être développé afin d'explorer les choix possibles pour les architectures multiflots et de comparer ce modèle aux architectures multiprocesseurs classiques.

3.2 L'environnement de programmation KOAN/Fortran-S

Participants : François Bodin, Mounir Hahad, Lionel Kervella, Thierry Montaut, Mike O'Boyle, Gilbert Cabillic, Thierry Priol

Cette activité vise à concevoir un environnement de programmation pour les architectures massivement parallèles munies d'un dispositif d'adressage global. Ce dispositif peut apparaître sous différentes formes :

- mémoire virtuelle partagée (MVP) comme la KSR-1, l'iPSC/2 avec la MVP Koan ou plus récemment la Paragon XP/S avec la MVP Myoan.
- mémoire à accès non-uniforme comme le Cray T3D ou la Meiko CS-2.

Nos travaux se sont concentrés autour de quatre axes: la conception d'une MVP, la génération et la parallélisation de code, la réalisation d'un outil d'analyse de performance et enfin des expérimentations avec des applications en collaboration avec d'autres centres de recherche en France et en Europe.

Conception de MVP En collaboration avec le projet Solidor, le portage de la MVP Koan a été réalisé sur la Paragon XP/S. Ce travail a été effectué dans le cadre d'un contrat de recherche et développement (ERDP) avec la société Intel SSD. Cette MVP reprend les fonctionnalités de Koan et a été implémentée comme une unité de pagination externe du système d'exploitation OSF/1 exploitant le micro-noyau MACH. Les premières expériences sont en cours afin de tester la stabilité et les performances de cette MVP.

Le générateur de code Fortran-S Fortran-S est un générateur de code conçu à l'Irisa pour des architectures parallèles disposant d'un mécanisme d'adressage global. Il constitue une plate-forme de recherche nous permettant de valider schémas de compilation et optimisations. Fortran-S offre à l'utilisateur un ensemble de directives lui permettant de spécifier le parallélisme dans un programme séquentiel Fortran. L'utilisateur indique essentiellement, à l'aide de directives, les variables partagées et les boucles parallèles. Le programme, généré à partir d'un source Fortran-S, s'exécute selon le modèle SPMD (*Single Program Mul-*

tiple Data streams). Fortran-S est disponible sur plusieurs architectures comme l'iPSC/2, la Paragon XP/S et la KSR-1.

Les optimisations qui ont été particulièrement étudiées concernent le traitement du faux-partage, la réduction du nombre de synchronisations ainsi que la prise en compte des algorithmes à accès irréguliers.

Le faux-partage est un facteur important de dégradation de performance d'une MVP telle que Koan. Des techniques de transformation de programme ont été mises au point pour réduire et éliminer les effets du faux partage. La première technique consiste en une vectorisation des écritures dans les pages de la MVP, afin de supprimer les effets de ping-pong . La deuxième technique permet d'aligner l'utilisation des pages par rapport aux processeurs, supprimant ainsi le faux partage.

Un des obstacles à la génération de code SPMD efficace est la réduction du nombre de synchronisations. En collaboration avec Mike O'Boyle, chercheur de l'université de Manchester qui a passé six mois au sein du projet Caps, dans le cadre du projet Esprit BRA Apparc, les études qui ont été réalisées ont permis la comparaison du modèle SPMD avec celui du *fork & join* sur la machine KSR-1. Les résultats ont montré que les codes SPMD obtenus par nos techniques sont meilleurs que ceux offerts par le restructureur KAP, grâce notamment à un meilleur placement des synchronisations.

Le faux partage de données et les synchronisations engendrés par des accès irréguliers aux données diminuent notablement les performances des architectures parallèles offrant une MVP. L'étude de quelques algorithmes irréguliers (assemblage et factorisation de matrices creuses) a permis l'élaboration d'un schéma de compilation spécifique. Ce schéma, appelé *boucles à itérations conditionnelles*, a été intégré à Fortan-S et testé avec succès. Des travaux en cours visent à optimiser les performances de ce schéma par une technique dite d'apprentissage.

Expérimentation d'applications Il est particulièrement intéressant de tester la plate-forme de recherche que nous avons développée avec des applications parallèles. En collaboration avec l'Onera, et dans le cadre du projet Esprit Apparc, nous avons étudié le portage d'une application pour la résolution de l'équation de Boltzmann. La parallélisation de ce code sur une architecture distribuée est rendue complexe par la gestion des données distibuées. Nous avons utilisé la MVP Koan ainsi que le générateur de code Fortran-S afin d'effectuer la parallélisation. Les

résultats obtenus ont montré un bon comportement de l'algorithme parallèle lorsque l'on augmente à la fois la taille du problème et le nombre de processeurs pour le résoudre.

Outils d'analyse de performance Les expériences que nous avons réalisées avec la MVP Koan ont montré qu'il est très difficile d'analyser le comportement des codes parallèles s'exécutant avec une MVP. Dans le cadre du projet Esprit BRA Apparc, nous avons conçu l'analyseur de performance SVMview. Cet outil, fondé sur une analyse post-mortem, permet de mettre en relation les événements générés par la MVP (défauts de page, invalidations, migrations de pages, ...) avec ceux générés par Fortran-S (variables partagées, sections de code séquentielles, boucles parallèles, ...).

3.3 Compilation pour architectures parallèles

Participants : François Bodin, Stéphane Chauveau, William Jalby, Yvon Jégou, Thierry Montaut, Pierre Rabain, Yves Robin

3.3.1 Optimisation de la localité des programmes

L'analyse de la localité des programmes est essentielle pour obtenir de bonnes performances sur les machines dotées d'une hiérarchie mémoire. Il s'agit de détecter les réutilisations potentielles de données et de caractériser les parties de structures de données sujettes à des réutilisations. Ces informations sont ensuite utilisées pour transformer les programmes afin d'améliorer l'utilisation de la hiérarchie mémoire. Ce type de transformations est essentiel pour une utilisation efficace des processeurs actuels dont le goulet d'étranglement principal est constitué par les accès à la mémoire. Des méthodes d'évaluation quantitative des propriétés de localité d'un code ont été mises au point. Elles sont fondées sur la notion de *fenêtres de références*. Ces méthodes s'appliquent tout aussi bien à l'optimisation des accès aux antémémoires et mémoires locales qu'à l'allocation de registres. Nos efforts portent maintenant sur la généralisation de ces résultats ainsi que sur leur validation dans le cas d'applications numériques industrielles. Ces travaux constituent les fondements de nos activités de recherche en compilation et sont la base de nombreuses optimisations de programmes, que ce soit dans le cadre

de Fortran-S ou encore pour la hiérarchie mémoire des processeurs Risc. Ces travaux font partie du projet Esprit BRA Apparc.

3.3.2 Environnement pour le prototypage de transformation de programmes

La recherche en compilation et outils de parallélisation doit s'appuyer sur des validations grandeur nature : mise en œuvre d'une optimisation au sein d'un compilateur pour un monoprocesseur, développement d'un compilateur prototype, parallélisation complète d'une application sur la MVP.

Permettre ce type d'expérimentation à moindre coût en temps de développement est particulièrement important. En collaboration avec l'équipe du Pr. Gannon de l'université d'Indiana, F. Bodin participe à la définition et la mise en œuvre du système Sage. Il s'agit d'un environnement dédié à la mise en œuvre de transformations source à source de programmes Fortran.

3.3.3 Outils pour la parallélisation

Une interface pour l'écriture de transformations de boucles est actuellement en cours de développement en collaboration avec Mike O'Boyle de l'université de Manchester. Cette interface propose, entre autre, une représentation sous forme d'espace d'iteration des boucles et des accès aux tableaux sous forme affine. De nombreuses transformations de boucles peuvent ainsi s'écrire de manière simple. Cette interface est construite sur le système Sage++.

A l'aide de cet outil, une nouvelle version du paralléliseur expérimental Parka est en cours de conception. Cette version permet entre autres d'exploiter à la fois un parallélisme par distribution des données et un parallélisme de contrôle.

3.3.4 Abstraction des algorithmes numériques pour la génération de codes parallèles

En collaboration avec l'équipe Aladin, une thèse a commencé sur le thème des approches de très haut niveau pour l'expression des algorithmes numériques ainsi que sur les méthodes de génération de code parallèle associées. On prendra comme point de départ des programmes

de type MatLab pour l'expression des algorithmes. Une partie importante de ce sujet sera consacrée l'étude des bibliothèques de calcul numérique creux qui constitueront la cible du générateur de code.

3.3.5 Exécution de codes irréguliers sur des architectures à mémoires distribuées par migration de tâches

La puissance potentielle des architectures massivement parallèles est difficile à exploiter par les codes irréguliers (calculs sur des maillages, matrices creuses, ...). Cette difficulté n'est pas due au manque de parallélisme de ces codes, qui est en général massif. Elle est surtout liée à l'absence de localité dans les accès aux structures de données. Cette irrégularité des accès se traduit par un volume important d'échanges entre les processeurs au cours de l'exécution. Lorsque la granularité des échanges est faible, ce qui est fréquent, il est primordial de chercher à recouvrir les phases de calcul et d'échange.

Dans la technique d'exécution par migration de tâches, une tâche est associée à chaque itération d'une boucle parallèle. Lorsque l'exécution d'une tâche entraîne un accès à une donnée non locale au processeur, l'exécution de la tâche est suspendue et la tâche *migre* sur le processeur possédant la donnée. Après migration, son exécution est reprise sur le processeur destinataire. Ce système est totalement asynchrone ; cet asynchronisme laisse espérer un recouvrement entre les communications et le calcul.

Cette technique a été expérimentée sur la Paragon XP/S de l'Irisa sur un code d'assemblage. Les résultats montrent que cette technique accélère l'exécution des codes irréguliers sur des architectures à mémoires distribuées dès lors qu'ils contiennent un parallélisme potentiel suffisant, même lorsqu'ils présentent très peu de localité dans les accès aux données. Une technique de transformation automatique de programmes spécifique à ce schéma d'exécution a été développée. Son introduction dans un compilateur Fortran expérimental est à l'étude.

4 Actions industrielles

T. Priol est responsable d'un contrat de recherche d'une durée de trois ans (juin 1993–juin 1996) avec la société Intel impliquant plusieurs projets de l'Irisa : Aladin, Caps, Solidor et Pampa (contrat no

193C21431318012). Dans le cadre de ce contrat, les membres de ces projets effectueront des recherches sur la Paragon XP/S.

Les études de veille technologique sur les microprocesseurs sont soutenues par la Dret (convention no 93-082 et mise à disposition d'un scientifique du contingent).

F. Bodin a encadré la thèse de Y. Robin au laboratoire Thomson LER (soutenue en novembre 1994) . Cette thèse a pour objet les environnements de programmation pour le traitement d'images sur machines parallèles SIMD à mémoire distribuée.

Une convention de partenariat de recherche entre le projet Caps et l'équipe Engineering ACRI (Stern Computing Systems) a été signée en novembre 1994. Ce partenariat porte d'abord sur l'évaluation de l'architecture et des outils de compilation disponibles sur la machine ACRI-1. Les travaux porteront ensuite sur la génération et l'optimisation de code pour les architectures découplées, l'évaluation des techniques d'optimisation de localité dans le contexte d'une architecture découplée, ainsi que sur des propositions pour l'évolution de l'architecture (en particulier, hiérarchie mémoire et mécanismes d'accès).

5 Actions nationales et internationales

5.1 Actions nationales

L'activité architecture du projet est soutenue par le PRC-GDR Architecture de Machines Nouvelles. Le projet participe à un projet commun inter-PRC (appelé Iliad) avec l'Irit, l'Imag, l'IEF et le Masi.

F. Bodin participe au groupe de travail Paradigme du GDR PRS.

5.2 Actions internationales

Dans le cadre du projet Esprit BRA Apparc *Performance-critical applications of parallel architectures*, le projet entretient des relations suivies avec les universités de Leiden (Pr. Wishoff), de Manchester (Pr. Gurd), le KFA (Pr. Gerndt) et l'institut Polytechnique de Barcelone (Pr. Valero).

F. Bodin a été invité à l'université d'Indiana du 25 Juin au 15 Juillet. Les travaux ont essentiellement porté sur le système Sage++.

La collaboration avec l'université de Rice, en particulier avec E. Granston (visite d'une semaine en janvier 1994), continue et a donné lieu à une présentation des travaux communs au *Seventh Annual Languages and Compilers for Parallel Computing Workshop*.

Dans le cadre du projet Apparc, F. Bodin et L. Kervella ont passé une semaine à l'université de Manchester. Ceci fait suite au séjour de 6 mois de Mike O'Boyle dans l'équipe Caps. La collaboration se poursuit sur les thèmes de la parallélisation et de l'optimisation de la localité des programmes.

T. Priol préside le groupe européen des utilisateurs de machines parallèles Intel.

T. Priol participe à la collaboration France-Espagne entre l'Irisa et l'université de Barcelone et de Saragosse et a été invité pour un séjour de 2 semaines au Centre Européen du Parallélisme de Barcelonne (Cepba).

A. Seznec a participé au workshop Ercim *Parallel Processing Network*.

6 Diffusion des résultats

6.1 Comités de programme et de lecture de revues

T. Priol est membre du comité de lecture de la *Lettre du Transputer et des Calculateurs Parallèles*.

T. Priol a été membre du comité de programme pour la conférence ICS en 1994.

F. Bodin a été membre du comité de programme de la conférence AGI' 94.

6.2 Animation scientifique

T. Priol anime le cmpi (club des machines parallèles de l'Irisa). Des ingénieurs de l'Atelier ainsi que des chercheurs ou enseignant-chercheurs de diverses équipes de recherche de l'Irisa participent également à l'animation du cmpi. Ce club a pour objectif de faciliter l'accès des établissements publics régionaux (epst, universités, écoles d'ingénieurs) aux machines parallèles de l'Irisa. Le cmpi a également un rôle de conseil auprès des projets de l'Irisa désirant utiliser une machine parallèle. Le

CMPI organise régulièrement des cours, séminaires ou conférences sur le thème architectures parallèles .

Dans le cadre de l'activité veille technologique sur les microprocesseurs , Y. Jégou et A. Seznec organisent une journée microprocesseurs rapides , le 10 janvier 1995.

6.3 Actions d'enseignement

F. Bodin et A. Seznec interviennent dans les cours d'architectures et compilation du DEA informatique et du DIIC.

A. Seznec intervient dans un cours sur les architectures Risc à l'ENST de Bretagne.

T. Priol donne un cours sur les architectures parallèles dans le cadre du DEA de l'université de Nantes.

T. Priol a donné un cours sur les langages Fortran pour les architectures parallèles au Laboratoire d'Analyse Numérique de l'université Lyon-1, en juin 1994.

6.4 Séminaires, communications

Outre les conférences et workshops donnant lieu à publication des actes listés dans la bibliographie, les membres du projet Caps ont présenté leurs travaux dans de nombreux séminaires ou workshops :

F. Bodin a été conférencier invité lors d'un minisymposium a' International Conference on Supercomputing 94 à Manchester.

F. Bodin a présenté l'environnement Koan-Fortran-S à un workshop à l'institut franco-russe Liapunov à Moscou en septembre 1994.

Y. Jégou a présenté un séminaire sur les techniques de compilation de programmes parallèles à une journée AFUU en novembre 1993.

Y. Jégou a présenté un séminaire intitulé Migrating tasks au séminaire NEC-Inria en septembre 1994 ainsi qu'à un workshop à l'institut franco-russe Liapunov à Moscou en septembre 1994.

Y. Jégou a présenté un séminaire intitulé tâches migrantes aux journées du CMPI en octobre 1994.

L. Kervella a présenté les travaux sur Fortran-S au 4th International Workshop on Compilers for Parallel Computers à Leiden, Pays-Bas en décembre 1993.

T. Priol a été invité à faire une présentation du concept de mémoire virtuelle partagée lors du workshop Europort, GMD St Augustin, février 1994.

T. Priol a présenté l'expérience de l'Irisa dans l'utilisation de la Paragon XP/S lors du séminaire du SEH à l'ETCA, Arcueil, juin 1994.

T. Priol a présenté les travaux sur l'environnement de programmation Koan Fortran-S dans le cadre de l'école d'été du CNRS à l'ENS de Lyon, juillet 1994.

T. Priol a présenté un exposé sur les architectures massivement parallèles lors de l'université d'été d'IBM, Cergy-Pontoise, septembre 1994.

T. Priol a présenté un exposé sur l'environnement Koan/Fortran-S à la Compagnie Générale de Géophysique, Massy, septembre 1994.

T. Priol a présenté les travaux sur la MVP Myoan et le compilateur Fortran-S au workshop PEP'94 organisé par l'université de Bergen (Para//ab) et Intel SSD, Bergen, Norvège, octobre 1994.

T. Priol a présenté ses travaux sur la programmation des architectures massivement parallèles à l'université Polytechnique de Barcelone dans le cadre d'un séjour de deux semaines au Centre Européen de Parallélisme de Barcelone, octobre 1994.

A. Seznec a présenté un séminaire sur les antémémoires à secteurs découplés au LIP à Lyon en avril 1994.

6.5 Prix

T. Priol a reçu le prix Jeune Chercheur 1994 de la Direction Générale de l'Armement pour ses travaux dans le domaine de la mémoire virtuelle partagée.

7 Bibliographie du projet

Thèses

[1] N. DRACH, *Optimisation du pipeline sur les processeurs monocomposants*, thèse de doctorat, université de Rennes 1, septembre 1994.

[2] Z. Lahjomri, *Conception et évaluation d'un mécanisme de mémoire virtuelle partagée sur une machine multiprocesseur à mémoire distribuée*, thèse de doctorat, université de Rennes 1, janvier 1994.

[3] Y. Robin, *Programmation et compilation sur architecture SIMD à mémoire distribuée dans un contexte de traitement d'image temps-réel vidéo*, thèse de doctorat, université de Rennes 1, novembre 1994.

Articles et chapitres de livre

[4] D. Badouel, K. Bouatouch, T. Priol, «Strategies for Distributing Data and Control for Ray-Tracing on Distributed Memory Parallel Computers», *IEEE Computer Graphics and Application*, juillet 1994.

[5] K. Bouatouch, T. Priol, «A Data Management Scheme for Parallel Radiosity», *A paraître dans "Computer Aided Design"*, 1994.

[6] C. Eisenbeis, W. Jalby, D. Windheiser, F. Bodin, «A strategy for array management in local memory», *Mathematical Programming*, 1994.

[7] A. Seznec, J. Lenfant, «Interleaved Parallel Schemes», *IEEE Transactions on Parallel and Distributed Systems*, décembre 1994.

[8] A. Seznec, J. Lenfant, «Odd memory systems: A new approach», *A paraître dans "the Journal of Parallel and Distributed Computing"*, 1994.

Communications à des congrès, colloques, etc.

[9] D. Bernard, F. Bodin, A. Goasguen, C. Fechant, «Implementing a two dimensionnal pore-scale flow model on different parallel machines», *in : Proceedings of X international Conference on Computational Methods in Water Resources*, Heildelberg (Allemagne), juin 1994.

[10] R. Berrendorf, M. Gerndt, Z. Lahjomri, T. Priol, «A Comparison of Shared Virtual Memory and Message Passing Programming Techniques Based on a Finite Element Application», *in : CONPAR*, Linz (Autriche), septembre 1994.

[11] F. Bodin, P. Beckman, D. Gannon, J. G. S. Srinivas, «Sage++: A Class Library for Building Fortran and C++ Restructuring Tools», *in : Second Object-Oriented Numerics Conference*, Oregon (usa), avril 1994.

[12] F. Bodin, E. Granston, T. Montaut, «Compile-time Techniques for Attacking False Sharing», *in : Poster à Supercomputing*, Washington (usa), novembre 1994.

[13] F. Bodin, E. Granston, T. Montaut, «Evaluating Two Loop Transformations for Reducing Multiple-Writer False Sharing», *in : Springer-Verlag in the Lecture Notes in Computer Science*, août 1994.

[14] N. Drach, D. Windheiser, «Comparaison des caches à correspondance directe et associatifs par ensemble», *in : 6ème rencontres du parallèlisme*, Lyon (France), juin 1994.

[15] M. Hahad, J. Erhel, T. Priol, «Scheduling Strategies for Sparse Cholesky factorization on a Shared Virtual Memory Parallel Computer», *in : International Conference on Parallel Processing*, St-Charles (Illinois, USA), août 1994.

[16] F. Hamelin, J. Jezequel, T. Priol, «A Multi-paradigm Object Oriented Parallel Environment», *in : International Parallel Processing Symposium*, Cancun (Mexique), avril 1994.

[17] A. Malony, B. Mohr, D. Gannon, F. Bodin, P. Beckman, S. Yang, «Performance Analysis of pC++: A Portable Data-Parallel Programming System for Scalable Parallel Computers», *in : International Parallel Processing Symposium*, Cancun (Mexique), avril 1994.

[18] A. Seznec, «Decoupled sectored caches: reconciliating low tag volume and low miss ratio», *in : Proceedings of the 21th International Symposium on Computer Architecture(IEEE-ACM)*, Chicago (usa), avril 1994.

[19] A. Seznec, «DASC cache», *in : Proceedings of the First High Performance Computer Architecture(IEEE-ACM)*, Raleigh (usa), janvier 1995.

[20] O. Temam, Y. Jégou, «Using virtual lines to enhance locality exploitation», *in : Proceedings of the 1994 International Conference on Supercomputing*, Manchester (GB), juillet 1994.

[21] S. X. Yang, D. Gannon, S. Srinivas, F. Bodin, «An HPF Fortran Interface for Parallel C++ (pC++)», *in : SIAM Scalable, High Performance Computing Conference*, Knoxville (Tenessee, USA), 1994.

Rapports de recherche et publications internes

[22] R. Berrendorf, M. Gerndt, T. Priol, «Performance Analysis Tools For SVM Architectures», *Ops4b deliverable*, projet Esprit Apparc, 1994.

[23] F. Bodin, W. Jalby, A. Seznec, M. O'Boyle, G. Hedayat, «Optimizing locality of programs», *Cod3 deliverable*, projet Esprit Apparc, 1994.

[24] F. Bodin, L. Kervella, M. O'Boyle, «Synchronization Minimization using SPMD execution model», *publication interne n° 863*, Irisa, 1994.

[25] F. Bodin, A. Seznec, «Cache organization influence on loop blocking», *publication interne n° 803*, Irisa, 1994.

[26] G. Cabillic, T. Priol, I. Puaut, «MYOAN: an Implementation of the KOAN Shared Virtual Memory on the Intel Paragon», *publication interne n°812*, Irisa, avril 1994.

[27] P. d'Anfray, F. Rogier, T. Priol, «Finite difference method for solving the Boltzmann equations with virtual shared memory», *Pca3a deliverable*, projet Esprit Apparc, 1994.

[28] N. Drach, A. Seznec, D. Windheiser, «Direct-mapped versus set-associative caches», *publication interne n°804*, Irisa, mars 1994.

[29] M. Hahad, T. Priol, J. Erhel, «Irregular Loop Patterns Compilation on Distributed Shared Memory Multiprocessors», *publication interne n°862*, Irisa, septembre 1994.

[30] S. Hily, «Evaluation de cache», *rapport de convention n°93-082*, contrat Dret-Inria, 1994.

[31] Y. Jégou, «Exécution de code irrégulier sur architecture à mémoire distribuée par migration de tâches», *publication interne n°867*, Irisa, 1994.

[32] Y. Robin, B. Guérin, F. Bodin, «IPF : un langage pour le traitement d'image sur architecture SIMD. Definition et performances.», *rapport de recherche n°2159*, Inria, janvier 1994.

[33] A. Seznec, T. Vauléon, «Etude comparative des architectures des microprocesseurs Intel Pentium et PowerPC601», *publication interne n°835*, Irisa, juin 1994.

[34] A. Seznec, T. Vauleon, «Etude comparative des architectures des microprocesseurs Intel Pentium et PowerPC601», *rapport de convention n°93-082*, contrat Dret-Inria, 1994.

8 Abstract

8.1 General goals

The Caps project focuses on the design and utilization of high performance computers, with the overall goal of making the memory hierarchy in high performance computers more transparent to users. This involves both hardware and software solutions. New hardware cache memory structures are being studied to reduce penalties due to main memory access. To reduce user awareness of the complexity of memory hierarchies in a parallel computer, a shared virtual memory is being studied at system level. Compilation techniques using and optimizing locality

of memory references have been developed to maximize performance at the hardware and system level.

8.2 Research actions

8.2.1 Processor architecture

Since 1991, the Caps project is involved in a technologic watch on state-of-the-art microprocessors. A comparison report on Intel Pentium and IBM PowerPC601 has been circulated.

Research on new cache organizations has continued focussing on skewed-associative caches and decoupled sectored caches. We have shown that when using skewed-associative caches, the performance of applications is quite insensitive to the relative placement of arrays in memory. Decoupled sectored cache is a new cache organization particularly adapted for second level caches where the tag volume is a major issue.

Work on OCO (Object Code Optimizer) has been continued this year ; OCO and a simulator of a superscalar processor has been used to measure the impact of some architectural choices on processor performance (pipelined caches, relative positions of execution and memory stages in the pipeline, ...).

8.2.2 Migrating tasks

In the migrating task model, each task is in charge of a chunk of the computation. When the execution of a task needs to access to a non local variable, the computation of the whole task moves on the processor owning the variable: the task migrates. This model has been shown to be efficient for irregular code computations on distributed memory parallel architectures.

8.2.3 The Koan/Fortran-S programming environment

The Koan/Fortran-S environment aims at providing a parallel programming environment for global address space multiprocessor architectures. To achieve this goal, we are interested in the design of shared virtual memory (SVM), parallelization, SPMD code generation, performance analysis tools and numerical application implementation.

In collaboration with the Solidor project, the Koan SVM has been implemented on the Paragon XP/S system on top of the Mach micro-kernel. Much work is devoted to optimizing the generated code when dealing with regular data access patterns as well as irregular ones. Part of this work is supported by the Esprit BRA Apparc projects.

8.3 Industrial actions and international actions

The Caps project is involved in the External Research and Development Program with Intel SSD.

The technological watch on microprocessors is supported by the DRET (french ministry of defense).

A research partnership with the Stern Computing Systems society has been signed in october 1994.

The Caps project is involved in the BRA Esprit project Apparc *Performance-critical applications of parallel architectures*.

Table des matières