

OSCAR FORTRAN Compiler を用いたマルチグレイン並列性の評価

小幡元樹[†]、松井巖徹^{††}、松崎秀則^{†††}、木村啓二[†]、稲石大祐[†]、
宇治川泰史[†]、山本晃正[†]、岡本雅巳^{††}、笠原博徳[†]

早稲田大学理工学部電気電子情報工学科 [†]、松下電器産業(株) ^{††}、(株) 東芝 ^{†††}

現在スーパーコンピュータは、数 TFLOPS のピーク性能を持ち、今後も伸び続けると考えられるが、価格性能比、使い難さの問題から市場を拡大できないという問題を持っている。また、マイクロプロセッサにおいては、スーパースカラ、VLIW 等で利用されている命令レベル並列性の限界が顕在化しており、次世代のプロセッサとして、シングルチップマルチプロセッサ(SCM)が注目されつつある。著者らは、マルチプロセッサ方式のマイクロプロセッサ(SCM)、サーバマシン、スーパーコンピュータの実効性能、すなわちコストパフォーマンス、使い易さを高めることを可能とするために、マルチグレイン自動並列化コンパイル手法を提案している。マルチグレイン並列処理とは、命令あるいはステートメントレベルの細粒度並列性、ループイタレーションレベルの中粒度並列性、サブルーチン・ループ・基本ブロックレベルの粗粒度並列性という、プログラムに内在する並列性を最大限に引き出す方式である。

本論文では、Perfect Club Benchmark の 2 次元流体解析プログラム ARC2D を例に、OSCAR マルチグレイン FORTRAN 並列化 Compiler を用いたマルチグレイン並列性利用の有効性を示す。

Evaluation of Multigrain Parallelism using OSCAR FORTRAN Compiler

Motoki Obata[†], Gantetsu Matsui^{††}, Hidenori Matsuzaki^{†††}, Keiji Kimura[†], Daisuke Inaishi[†],
Yasushi Ujigawa[†], Terumasa Yamamoto[†], Masami Okamoto^{††}, Hironori Kasahara[†]

Department of Electrical Electronics and Computer Engineering, Waseda University [†]
Toshiba Corporation^{††}
Matsushita Electric Industrial Co.,LTD. ^{†††}

Currently, the supercomputer have peak efficiency of TFLOPS order and it is higher and higer. But difficulty of enlargement of the world caused by unusefulness and cost-performance, which does not seem excellent for effective performance. Also on the microprocessor, limitations of extraction of instruction level parallelism being used by super scalar and VLIW architecture are getting clear, so single chip multiprocessor is received much attention as next generation processor. Therefore multigrain automatic compile method is proposed to be useful supercomputer and improve real efficiency, namely cost-performance, of single chip multiprocessor, server machine, and supercomputer. Multigrain parallel processing is a method which extract all parallelism from a program, where parallelism is including coarse, medium, and near fine grain.

This paper evaluates fluid flow problem solver ARC2D (Perfect Club Benchmark) on multiprocessor system OSCAR and show availability of multigrain parallel processing using OSCAR multigrain FORTRAN parallelization compiler.

1. はじめに

現在、スーパーコンピュータのピーク性能は数 TFLOPS に達し、2001 年には数十 TFLOPS に達すると考えられる。このようにピーク性能値は今後も向上し続けると考えられるが、プログラムを実行した際の実効性能のスケラブルな向上は難しく、

価格性能比に問題が生じている。マルチプロセッサシステムにおける実効性能低下の原因として、通信コストの増大や並列性の抽出が不十分であることが挙げられる。またスーパーコンピュータの使い勝手に関しても、ユーザーがプログラム内の並列性を抽出し、HPF, MPI, PVM といった拡張言語やライブラリを用いてプログラミングを行うのは非常に困難

である。この価格性能比、使い易さの問題は、HPC市場の拡大の阻害要因とも考えられ、産業界にとっても大きな問題となっている。また、マイクロプロセッサにおいても、現在主流のスーパースカラ、VLIW のように命令レベル並列処理[1,2,3]を行うプロセッサでは、命令レベル並列性の限界が顕在化しつつあり、今後のスケラブルな実効性能向上が難しいと考えられている。

今後、シングルチップマルチプロセッサ[2,4]からマルチプロセッサスーパーコンピュータまでのマルチプロセッサシステムの価格性能比を改善し、かつユーザーフレンドリーなプログラム開発環境を提供するためには、ユーザーが使い慣れた FORTRAN, C などの逐次処理言語で記述されたプログラムを自動的に並列化する自動並列化コンパイラがより重要になると考えられる。従来より、企業ベースでは SUN、IBM、PGI の C, FORTRAN, HPF コンパイラ、大学ベースではイリノイ大学の Polaris[5, 6, 7]、Paraphrase2[8]、スタンフォード大学の SUIF[9,10,11]などで自動並列化コンパイラの研究が行われている。Polaris では、サブルーチンのインライン展開、変数のシンボリックな伝播、スカラおよび配列のプライベート化を組み合わせた大規模ループの並列処理を行っている。また、SUIF は強力なサブルーチン間依存解析を用い、大規模ループの並列処理を可能としている。また、このコンパイラの研究推進のために行われている National Compiler Infrastructure Project(NCI)は、本年よりプログラムを一部変更し、スタンフォード大学、バージニア大学、PGI が共同で NCI の一部を担当することとなっている[12]。これらのコンパイラの多くは、ループ並列処理[13] (中粒度並列処理) に主眼を置いているが、ループ並列化技術は既に成熟期に入っており、今後の大幅な性能向上は困難であると考えられている。

そこで著者らは、ループ並列性に加えて、サブルーチンやループ間の並列性を利用する粗粒度並列性[14,15,16]や、基本ブロック内のステートメントレベルでの並列性を利用する近細粒度並列性[17,18]を階層的に利用するマルチグレイン並列処理[14,15,19]を提案し、OSCAR FORTRAN Compiler によって、世界で初めてマルチグレイン並列処理を実現している。これにより、ループ並列処理が不可能で、シーケンシャル処理しかできなかったプログラムに置いても、粗粒度及び近細粒度並列処理を用いることで、プロセッサを有効利用することが可能となる。

本論文では、Perfect Club Benchmark 内の ARC2D をサンプルとして、OSCAR FORTRAN Compiler を用いたマルチグレイン並列処理の有効性を示す。

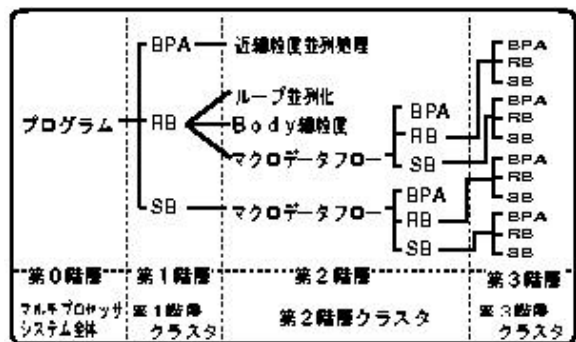


図 1. 階層的なマクロタスクの定義

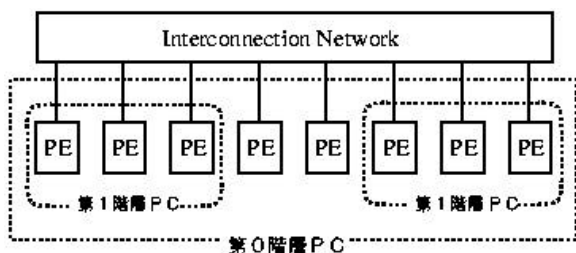


図 2. 階層的なプロセッサクラスタの定義

2. マルチグレイン並列性の抽出

本章では、粗粒度並列処理を実現するためのマクロデータフロー処理[14,15,16]及びループ並列処理[13]、ステートメントレベル近細粒度並列処理を階層的に用いたマルチグレイン並列処理手法[14,15,19]について述べる。

従来のループ自動並列化コンパイラでは、ループ並列性を抽出するためのデータ依存解析[20]やリストラクチャリング技術[21]が成熟してきており、今後の大幅な性能向上は望めない。そこで著者らは、中粒度並列性に加えて、プログラムに包含されるサブルーチン、ループ、基本ブロック間の粗粒度並列性や基本ブロック内のステートメント間の近細粒度並列性を用いるマルチグレイン並列処理を提案している。本手法では、まず粗粒度並列性を抽出するために、プログラム全体を以下に示す BPA, RB, SB の 3 種類のマクロタスク(Macro Task、以下 MT)に分割する。

- **BPA(Block of Pseudo Assignments)**
基本ブロック及び複数の基本ブロックを融合したマクロタスク。または、1つの基本ブロック中に独立したデータ依存グラフが存在する場合には、各グラフ毎に分割したブロックをマクロタスクとして扱う。
- **RB(Repetition Block)**

DO ループや IF 文による後方分岐などによって構成され、最外側ナチュラルループとして定義されるマクロタスクである。

- **SB(Subroutine Block)**

コード量などによりインライン展開が有効に適用できないと判断されたサブルーチンをマクロタスクとして定義したものである。

マクロデータフロー処理で生成されたマクロタスク(以下 MT)は、プロセッサエレメント(Processor Element、以下 PE)のグループであるプロセッサクラスタ(Processor Cluster、以下 PC)に割り当てられ、それぞれの MT 間の並列性を利用する粗粒度並列処理が実現される。そして、プロセッサクラスタ内のプロセッサエレメントを使用して粒度にあわせて、各 MT をループ並列処理や、基本ブロック内のステートメント間の並列性を利用する近細粒度並列処理により階層的に並列処理する。

例えば、BPA が割り当てられた PC 内では、各ステートメントを一つのタスクとした近細粒度並列処理が行われる。また、RB が割り当てられた PC では、ループが Doall ループや Reduction ループならばループ並列処理が適用される。ループ並列処理が適用できない場合は、ループボディの近細粒度並列処理、あるいは階層的なマクロデータフロー処理の有効な方を適用する。そして、SB が割り当てられた PC では、階層的にマクロデータフロー処理を適用する。

階層型マクロデータフロー処理では図 1 に示すように MT の階層的な定義を行う。各階層の MT は、図 2 に示すように階層的に定義される PC で並列処理が行われる。実際には、プログラム全体(第 0 階層マクロタスク)を分割して生成した MT を第 1 階層 PC に割り当て、その MT 内部をさらに MT に分割し、それらの MT を第 1 階層 PC を分割した第 2 階層 PC に割り当ててマクロデータフロー処理を行う。

階層的に MT を生成した後、各階層で MT 間のコントロールフロー解析、データフロー解析を行う。解析したコントロールフローとデータフローは各階層毎に生成するマクロフローグラフで表現される。次に各階層のサブマクロフローグラフに対して最早実行可能条件解析^[14,15]を行い、MT 間の並列性を抽出する。この結果得られた各階層における MT の最早実行可能条件をグラフで表現したマクロタスクグラフを各階層ごとに生成する。このようにして得られた各階層の MT は、マクロタスクグラフに条件分

岐などの実行時不確定性が存在する場合にはダイナミックスケジューリング、それ以外の場合にはスタティックスケジューリングで実行される。

3. OSCAR FORTRAN Compiler

本章では、マルチグレイン並列処理を行うために開発してきた OSCAR FORTRAN Compiler について述べる。OSCAR FORTRAN Compiler は、字句解析を行う Front End、並列性の抽出を行う Middle Path、各種アーキテクチャ用の並列化コードを出力する Back End の 3 ステージからなる。全ての並列性の抽出は Middle Path にて行われ、まずマクロデータフロー処理により、プログラム全体をサブルーチン、ループ、基本ブロックに分割して、制御フローをグラフ化したマクロフローグラフ^[14,21]を生成し、それを利用して最早実行可能条件を求めてマクロタスクグラフ^[14,16]を生成する。その後、データ依存解析、リストラクチャリング、各種最適化、タスクスケジューリングなどを行い、マルチグレイン並列処理のための中間言語を出力する。

4. ベンチマークプログラムの並列性及び性能評価

本章では、コンパイラによりマルチグレイン並列性を評価するために用いた Perfect Benchmark ARC2D と対象アーキテクチャ OSCAR について説明し、OSCAR 上での実行結果とそれに対する考察を述べる。

4.1. ARC2D(Perfect Club Benchmark)

ARC2D は 2 次元流体解析プログラムであり、解析手法としてオイラーの方程式を使用している。コード量は約 4500 行で、40 のサブルーチンが含まれており、並列化可能なループが 30 ある。

メインルーチンから呼び出されるサブルーチンの中で、全体の実行時間のほぼすべてを占めるのがサブルーチン INTEGR であり、その中には 20 のサブルーチンがあるが、オリジナルのサブルーチン INTEGR にはデータ依存により粗粒度並列性はほとんどない。しかし、サブルーチンのインライン展開、Loop Distribution、Loop Unrolling、Scalar & Array Privatization などの最適化をコンパイラが行うことにより、粗粒度並列性を抽出できる。図 3,4 はサブルーチン INTEGR 内から呼ばれるサブルーチン FILERX のマクロタスクグラフ、図 5,6 は STEPFX のマクロタスクグラフのオリジナルのものと、最適化適用後のものであり、図中の番号は

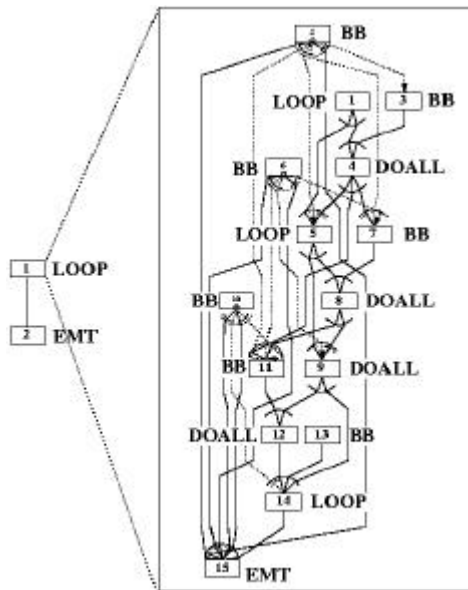


図 3. FILERX の MTG (Original)

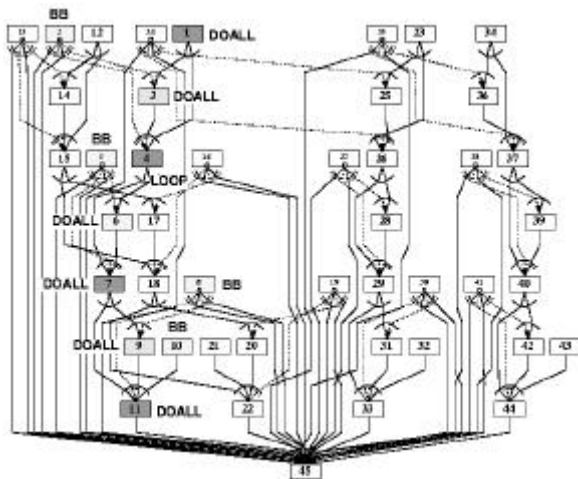


図 4. 粗粒度並列性抽出後の FILERX の MTG

MT 番号、BB は基本ブロック、LOOP はシーケンシャルループ、DOALL は DOALL ループ、SB はサブルーチンブロック、EMT は最終 MT であり、MT 間の点線エッジは制御フロー、実線エッジはデータ依存をそれぞれ示している。以下に、これらのサブルーチンの最適化について述べる。

まず、サブルーチン FILERX は全体の実行時間の約 15%を占める Doall ループであり、オリジナルのマクロタスクグラフは図 3 のようになる。しかし、最外側ループが 4 回の固定回転であるため、通常の Doall 処理を行うと PE 数が 4 より多い時には、その分だけアイドルとなるプロセッサが増えて効率が低下する。そこで、最外側ループをアンローリングし、テンポラリ配列に対する Array

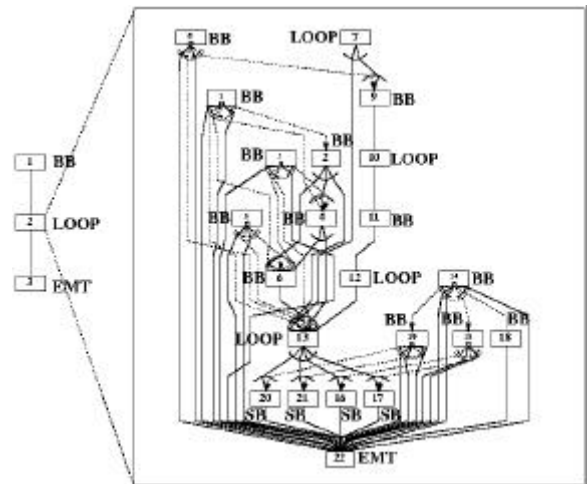


図 5. STEPFX の MTG (Original)

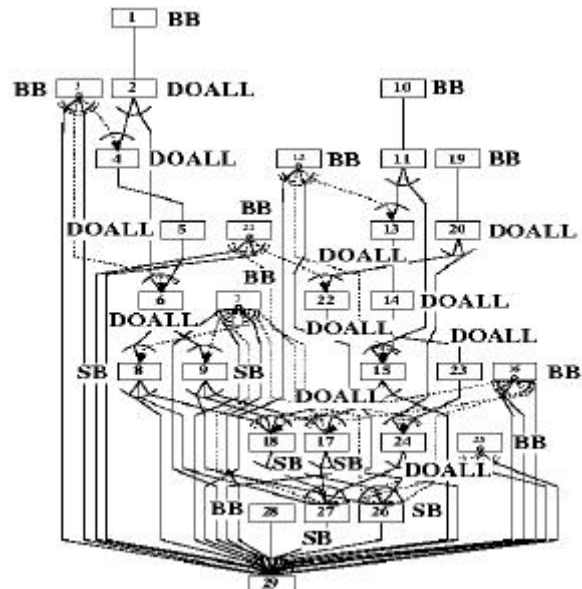


図 6. 粗粒度並列性抽出後の STEPFX の MTG

Privatization を適用することにより、図 4 に示すマクロタスクグラフを生成でき、粗粒度並列性を抽出することができる。

次にサブルーチン STEPFX の粗粒度並列化について述べる。STEPFX は実行時間の 30%を占めるシーケンシャルループであり、図 5 に示すマクロタスクグラフとなる。このループに対しては、アンローリングや Privatization の他に図 5 中のサブルーチンブロック 16,17,20,21 に対するインタープロシージャ解析により、図 6 に示す最適化適用後マクロタスクグラフ中のサブルーチンブロック 8 と 9、17 と 18、26 と 27 間に依存がないことが分かり、粗粒度並列性が抽出できる。この他にも、サブルーチン FILERX、STEPFX、などの実行時間の大きい

表 1. ARC2Dの実行時間

ALL PEs	PC	PE	Grain	Time(s)	Ratio
1	1	1		14802	1.00
4	1	4	LOOP	3971	3.73
	2	2	Multigrain	3824	3.87
15	1	15	LOOP	1983	7.47
	3	5	Multigrain	1494	9.91

サブルーチンが存在するが、同様の並列性抽出手法が適用される。

4.2. マルチプロセッサシステム OSCAR のアーキテクチャ

本節では、マルチグレイン並列性の評価対象としたマルチプロセッサシステム OSCAR のアーキテクチャ[22]について述べる。

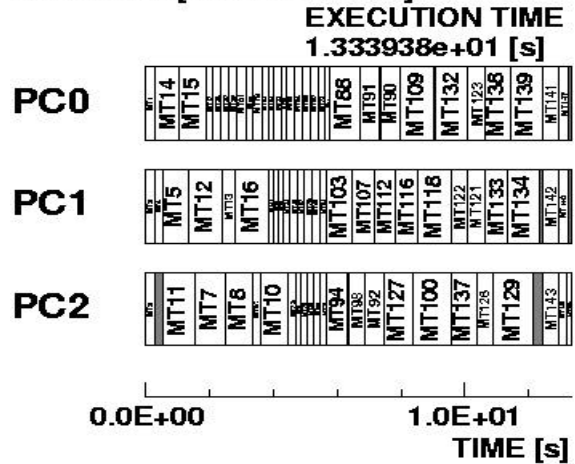
OSCAR は、コンパイル時のスタティックスケジューリングと実行時のダイナミックスケジューリングの長所を組み合わせることにより、マルチグレイン並列処理を効率的に実現するために設計されたマルチプロセッサシステムである。メモリ構成として、集中共有メモリ、分散共有メモリ、ローカルメモリの階層的なアーキテクチャを採用し、プロセッサエレメントにはカスタムメイド 32bit RISC プロセッサを用いている。集中共有メモリはダイナミックスケジューリング時の MT 間共通データの格納用、分散共有メモリは近細粒度並列処理の際の PE 間データ転送用、そしてローカルメモリはデータの局所性を利用するデータローカライゼーション手法[23]適用時のデータ格納にそれぞれ用いられる。マルチグレイン並列処理を行う上で重要なプロセッサクラスタの形成では、PE を結合する 3 本のバスに高速バリア同期機構を持たせ、複数のプロセッサクラスタを持つシステムとしても効率よく実行できるように設計されている。

4.3. 実行時間と考察

表 1 に 2 次元流体解析プログラム ARC2D の実行時間とシーケンシャル実行に対する速度向上率を示す。従来研究との比較のため、マルチグレイン並列処理を行った場合と、ループ並列処理のみの場合の実行時間を示している。

表 1 より、合計プロセッサ数が 4 の場合は、マルチグレイン並列処理では 3824 秒、Polaris による解析結果を用いたループ並列処理では 3971 秒となりほとんど差がでないが、プロセッサ数が 15 と増加した場合は、ループ並列処理では 1983 秒かかるのに対してマルチグレイン並列処理では 1494 秒となり、25%速度が向上している。マルチグレイン

Chart:0002 [Chart0001-MT6]



度並列処理) に対して、プログラムをループ、サブルーチン、基本ブロックに分割するマクロデータフロー処理により粗粒度並列性を抽出し、さらに各ブロック内に階層的にマクロデータフロー処理を適用して中粒度、近細粒度並列性も利用するマルチグレイン並列処理手法について述べた。

有効性の評価のために、Perfect Club Benchmark 中の ARC2D を用い、マルチプロセッサシステム OSCAR 上において 15 プロセッサを用いた場合、ARC2D では、現在最速のループ並列化コンパイラの 1 つである Polaris の解析結果を用いて 7.47 倍の速度向上が得られるのに対して、9.91 倍の速度向上を得ることができ、ループ並列処理よりも 25% 処理時間を短縮することができた。従来のループ並列化のみの並列処理に対して、プログラム全体における全ての並列性を抽出するマルチグレイン並列処理ではプロセッサが有効に利用できており、今後の並列処理手法として有用であることが確認された。

今後は、さらに多くのアプリケーションプログラムを用いた評価を行い、マルチグレイン並列性の抽出手法の改善を行っていく予定である。

また、マルチグレイン並列処理を実現するためには、OSCAR のような速いネットワークが必要であるが、今後実用化されるとされるシングルチップマルチプロセッサに対して、細粒度並列処理が有効に機能するものと思われ、今後のマルチプロセッサシステムとマルチグレイン並列処理の結合が期待されている。

本研究の一部は、通産省次世代情報処理基盤技術開発事業並列分散分野マルチプロセッサコンピューティング領域研究の一環として行われた。

参考文献

- 1 D.Burger, J.R.Goodman: Billion Transistor Architectures, IEEE Computer, Vol.30, No.9, pp.47-49, Sep.1997.
- 2 L.Hammond, et.al: A Single-Chip Multiprocessor, IEEE Computer, Vol.30, No.9, pp79-85, Sep.1997
- 3 M.H.Lipasti, J.P.Shen: Superspeculative Microarchitecture for Beyond Ad 2000, IEEE Computer, Vol.30, No.9, pp59-66, Sep.1997
- 4 木村, 笠原: マルチグレイン並列処理用シングルチップマルチプロセッサアーキテクチャ、情報処理学会第 56 回全国大会、1N-03, Mar.1998
- 5 W.Blume, R.Doallo, R.Eigenmann, J.Grout, J.Hoeflinger, J.Lee, D.Padua: Advanced Program Restructuring for High-Performance Computers with Polaris, Technical Report 1473, CSRD, University of Illinois, Urbana-Champaign, 1996

- 6 W.Blume, R.Doallo, R.Eigenmann, J.Grout, J.Hoeflinger, T.Lawrence, J.Lee, D.Padua, Y.Paek, B.Pottenger, L.Rauchwerger, P.Tu: Parallel Programming with Polaris, IEEE Computer, Vol.29, No.12, pp.78-82, 1996
- 7 R.Eigenmann, J.Hoeflinger, D.Padua: On the Automatic Parallelization of the Perfect Benchmarks, IEEE Trans. on PARALLEL AND DISTRIBUTED SYSTEMS, Vol.9, No.1, Jan.1998
- 8 Constantine Polychronopoulos, M.B.Girtar, M.R.Haghighat, C.L.Lee, B.P.Leung, D.A.Schouten: Paraphrase-2: An Environment for Parallelizing, Partitioning, Synchronizing, and Scheduling Programs on Multiprocessors. Proc. of the International Conference on Parallel Processing, Aug.1989.
- 9 S.Amarasinghe, J. Anderson, M.Lam, C.Tseng: The SUIF Compiler for Scalable Parallel Machines, Proc. of the 7th SIAM conference on parallel processing for scientific computing, SIAM, 1995
- 10 M.Hall, J.Anderson, S.Amarasinghe, B.Murphy, S.Liao, E.Bugnion, M.Lam: Maximizing Multiprocessor Performance with the SUIF Compiler, IEEE Computer, Vol.29, No.12, pp.84-89, 1996
- 11 M.Hall, S.Amarasinghe, B.Murphy, S.Liao, M.Lam: Detecting Coarse-Grain Parallelism Using an Interprocedural Parallelizing Compiler, Proceedings of Supercomputing '95, 1995
- 12 David L. Moore: The National Compiler Infrastructure Project, CPC'98 Seventh International Workshop on Compilers for Parallel Computers, Jun.1998
- 13 D.Padua, M.Wolfe: Advanced Compiler Optimizations for Super Computers, C.ACM, 29,12,pp.1184-1201, 1986
- 14 笠原博徳: 並列処理技術, コロナ社, 1991
- 15 H.Kasahara, H.Honda, S.Narita: A Multi-Grain Compilation Scheme for OSCAR, Proc. 4th Workshop on Languages and Compilers for Parallel Computing, 1991
- 16 笠原, 合田, 吉田, 岡本, 本多: FORTRAN マクロデータフロー処理のマクロタスク生成法, 信学論(D-I), J75-D-I, pp.511-525, 1992
- 17 笠原博徳: マルチプロセッサシステム上での近細粒度並列処理, 情報処理, Vol.37, No.7, Jul.1996
- 18 H.Kasahara, S.Narita: Practical Multiprocessor Scheduling Algorithms for Efficient Parallel Processing, IEEE Trans. on Computers, Vol.C-33, No.11, Nov.1984
- 19 H.Kasahara, et.al: OSCAR Multigrain Architecture and its performance, Proc. International Workshop on Innovative Architecture for Future Generation High Performance Processors and Systems, IEEE Press, Oct.1997
- 20 U.Banerjee: Loop Parallelization, Kluwer Academic Pub, 1994
- 21 M.Wolfe: High Performance Compilers for Parallel Computing, Addison Wesley, 1996
- 22 笠原, 成田, 橋本: OSCAR のアーキテクチャ, 信学論(D), J71-D, 1993
- 23 H.Kasahara, A.Yoshida: A Data-Localization Compilation Scheme Using Partial Static Task Assignment for Fortran Coarse Grain Parallel Processing, Journal of Parallel Computing, Special Issue on Languages and Compilers for parallel Computers, May. 1998