戦略的創造研究推進事業 CREST

研究領域「情報社会を支える新しい高性能情報処理技術」

**研究課題「超低電力化技術によるディペンダブル メガスケールコンピューティング」**

# 研究終了報告書

研究期間　平成 13 年 12 月～平成 18 年 11 月

研究代表者：中　島　　浩

（京都大学学術情報メディアセンター 教授）

## 1. 研究実施の概要

PetaFlops 級の計算能力を得るためには極めて大規模な並列システムの構築が不可欠であるが，従来の MPP やクラスタ計算機技術の延長でのプロセッサ数増加は，設置面積，消費電力，メンテナンス，ソフトウェア開発の面で限界に来ている．例えば，本プロジェクト開始時に世界最大級のシステムであった ASCI プロジェクトの MPP や地球シミュレータは，数千～1 万プロセッサで既に小スタジアムほどの大きさを占め，電力も 10 メガワット以上を消費するものであった．

一方，より一般的な計算機技術分野において，メガスケールコンピューティングを実現する技術の別のコンテクストでの研究開発が進みつつ，または注目されつつあった．これらは従来のようにハイエンドではなく，むしろ汎用的なコモディティ技術の基盤となるもの，あるいはそれをベースとするものである．我々の主張は，このような技術をベースとするアプローチ，すなわち単純に高性能や高機能を目指した従来型の高性能システムの研究開発とは根本的に異なったアプローチで，はじめてメガスケールの高性能計算を達成できるというものである．

本研究の目的は

(1) ハードウェア／ソフトウェア協調による低電力化技術

(2) 大規模並列タスクの実行モデル構築・利用技術

を柱として，種々のコモディティ技術を活用したメガスケールコンピューティングの基盤技術を確立することにある．すなわち，この 2 つの技術を中核としてプロセッサ，コンパイラ，ネットワーク，クラスタ管理，およびプログラミングに関する研究を行い，それらにより 100 万プロセッサ級の汎用メガスケールコンピューティングが実現できることを示すことと，そのプロトタイプとして

(3) 低電力・高密度大規模クラスタ MegaProto

を構築して技術の有効性を実証することが，本研究の目的である．

(1) ハードウェア／ソフトウェア協調による低電力化技術

現実的な設置規模でメガスケールのシステムを構築するためには高密度実装が不可欠であるが，そのためにはまずプロセッサの消費電力を極力削減する必要がある．そこで我々は，ハードウェアとソフトウェアの協調によりデータ転送を中心とする最適化を行い，低消費電力と高性能の両立を

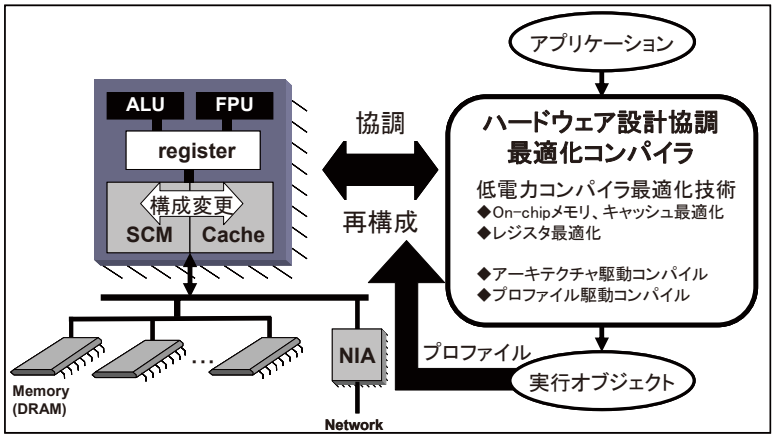



図 1：ハードウェア／ソフトウェア協調低電力化

目指した研究を行った．

この研究の鍵となるハードウェア技術は，SCIMA (Software Controlled Integrated Memory Architecture)と呼ぶ，ソフトウェアから可視かつ構成の変更が可能な高速メモリ階層アーキテクチャである（図 1）．SCIMA は，通常のキャッシュとの境界が可変である高速メモリ SCM を中心に構成され，配列などのデータは再利用性，アクセスの規則性，容量に応じて SCM あるいは通常のキャッシュ可能な空間に割付けられる．この割付けをコンパイラが最適化することにより，プロセッサチップと主記憶の間のデータ転送の回数や量を大幅に削減することができ，さらにオンチップメモリのアクセスによる消費電力も削減できる．この結果，実行時間と消費エネルギーの両面で，大きな削減効果が達成された．

一方，従来型のプロセッサについても，ハードウェア機構，特に電源電圧や周波数を動的に変更する DVS 機構を，最適化コンパイル技術により活用して，消費電力を削減することができる．本研究ではこの DVS とコンパイラの協調技術を多角的に追求し，プログラムフェーズごとの電力プロファイルに基づく最適化，負荷不均衡プロセス群の電圧・周波数最適設定，実効消費電力の動的制御による性能最適化，性能カウンタの統計情報に基づく電力最適化といった技術を提案し，いずれも優れた効果を得ている．

(2) 大規模並列タスクの実行モデル構築・利用技術

メガスケールのシステムは膨大な計算資源を持つため，ある意味で超大規模の広域分散計算環境に相通じる性格を持っている．すなわち，大きな粒度の並列タスクを単位としたプログラミングと，その実行と環境の管理の大規模な分散化は必然である．しかしその一方で，現実的な設置規模に収められた単一あるいは少数の計算環境の集合体であることを生かし，システム全体を統一的に管理・運用する機構を持つことが求められる．我々は，この分散と統一という背反する課題を解決する鍵が，並列タスクの実行挙動をあらかじめ把握することにあるとの考察に基づき，タスク実行モデルの構築とモデルを利用した実行および環境の管理技術を研究した（図 2）．

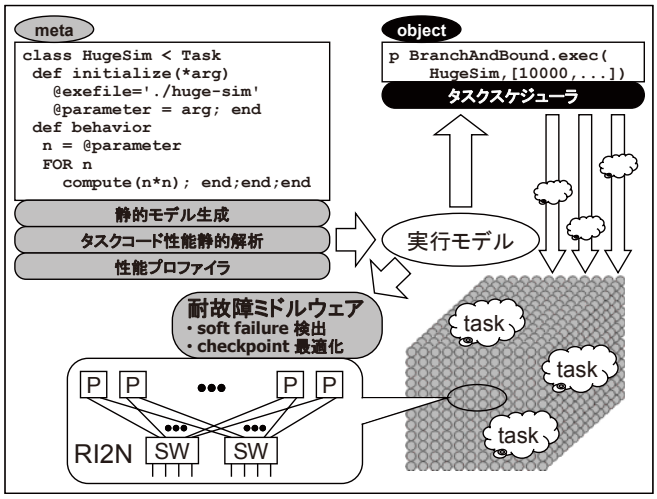



**図 2：実行モデルによる実行・環境管理**

実行モデルの構築のために，我々は並列タスクの挙動情報を記述可能なタスク並列スクリプト言語 MegaScript を設計した．この言語ではコンパイラの解析情報からは決定困難なタスク挙動に関する量的情報を与えることができ，静的あるいは動的な実行モデルを高精度に構築することができる．生成されたモデルは，MegaScript で記述された並列タスク実行のスケジューリングのために用いられ，タスク粒度の調整や最適な配置が行われる．

また，実行モデルの別の重要な応用として，システムの信頼性向上に関する研究も行っている．この研究では，大規模システムの各ノードで実行される，挙動が類似したプロセスの関数呼出／復帰のトレースにより挙動モデルを構築し，モデルから大きく外れたプロセス挙動を異常なものとして統計的に分別する．またこの手法を 129 ノードの大規模クラスタに適用して，長期間の運用中に生じたシステムダウンの原因を解析した結果，システムのミドルウェア中に間歇的に生じる異常の原因となる２つのバグを発見することができた．

また大規模システムにおける脆弱性の主要因であるネットワークの耐故障性のために，我々はノード間リンクやスイッチ間のパスをを多重化して高バンド幅と高信頼性を同時に実現する，RI2N (Redundant Interconnection with Inexpensive Network)と，VFREC-Net (VLAN-based Flexible, Redundant and Expandable Commodity Network)を提案し，Mega- Proto 含む種々のクラスタに実装して性能・信頼性の両面での有効性を実証した．さらにハードウェアレベルの耐故障機能や，チェックポイントの生成・回復機能を統合した，耐故障 MPI フレームワークである Cuckoo を開発した．

(3) 低電力・高密度大規模クラスタ MegaProto

前述のように我々は，コモディティ技術をベースとして低電力プロセッサを高密度に実装するアプローチこそが，メガスケール計算を実現する唯一の方法であると主張している．この主張を裏付けるひとつの方法は，現時点で利用可能な技術を用いて高密度・低消費電力・高性能のシステムを構築し，その延長線上に我々が目指すメガスケール計算システムが存在することを実証することである．

そこで我々は，多数の低電力プロセッサを高密度に実装し，それらを高信頼・高バンド幅のネットワークで結合したプロトタイプシステム MegaProto を開発した．また MegaProto は，プロジェクトで研究・開発中の様々な技術の実証プラットフォームとしても利用されている．

MegaProto の仕様設計は５名のグループリーダを中心とした設計チームを組織して行い，TM5800 (Crusoe)を用いたテスト機（MegaProto/C）と，TM8800 (Efficeon)を用いた実用機(MegaProto/E)の詳細な仕様を定めた．この仕様に基づき実装設計および製造を外注し，MegaProto/C を２台(32PE)，また MegaProto/E は２０台(320PE)をそれぞれ製造した．これらの MegaProto システムはいずれも電力性能比を達成し，特に MegaProto/E の 100MFLOPS/W はコモディティ技術を用いた並列システムとしては世界最高の値を達成した．

## 2. 研究構想及び実施体制

(1) 研究構想

本研究は前述のように

① ハードウェア／ソフトウェア協調による低電力化技術

② 大規模並列タスクの実行モデル構築・利用技術

③ 低電力・高密度大規模クラスタ MegaProto

の３項目を柱として実施した．当初における研究全体の構成とスケジュールは，それぞれ図 3 と図 4 に示すとおりであった．以下，各々について当初の構想と，その後の展開について述べる．

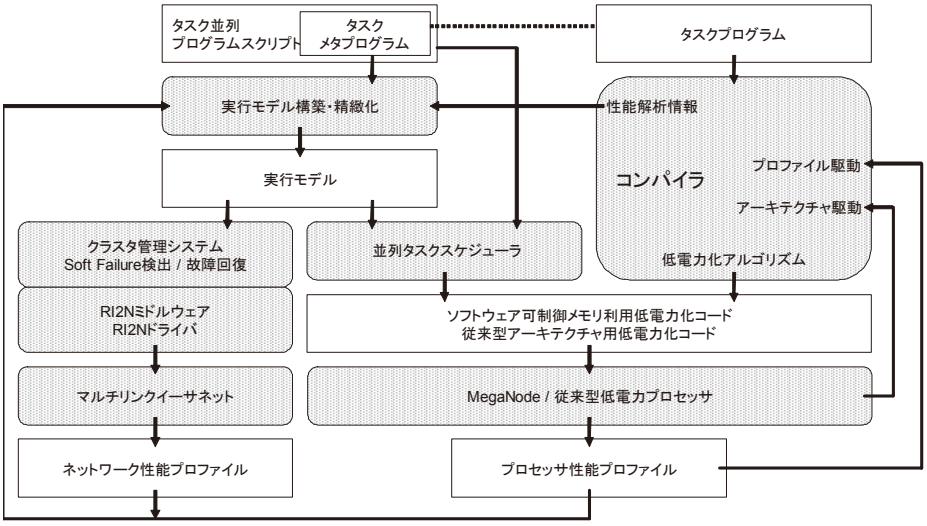



図 3：全体構成

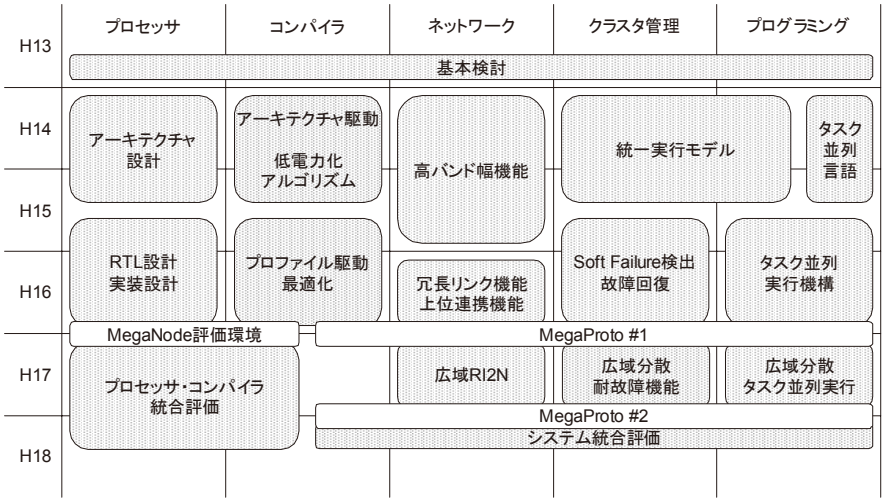



図 4：当初の研究スケジュール

① ハードウェア／ソフトウェア協調による低電力化技術

当初の構想では，「高性能プロセッサの消費電力の重要な支配項であるチップ内メモリ，プロセッサチップ入出力，およびチップ外メモリの消費電力削減を，レジスタ⇔チップ内メモリ⇔チップ外メモリ⇔ネットワークのデータ転送レートの削減により達成する」ために，ハードウェア（プロセッサアーキテクチャなど）とソフトウェア（コンパイラなど）の協調技術の研究を，以下の２グループにより実施することとした．

【研究項目１】 ソフトウェアとの協調最適化に基づく超低消費電力技術・高密度実装技術・高バンド幅技術：プロセッサグループ（リーダ：東大・中村）

【研究項目２】 ハードウェアとの協調最適化に基づき低消費電力かつ高性能を実現するコンパイラ技術：コンパイラグループ（リーダ：筑波大・佐藤）

また具体的な研究成果として，以下の目標を掲げた．

(a) ソフトウェアから可視のオンチップメモリと renaming register を持つ新たなプロセッサアーキテクチャの設計

(b) オンチップメモリ／renaming register を活用した電力最適化アルゴリズムと最適化コンパイラの設計

(c) プロセッサアーキテクチャパラメータと実行プロファイルに基づく電力最適化コンパイラの設計と既存プロセッサへの適用

一方，研究の初期段階でオンチップメモリを活用した低電力化はデータ転送レート削減だけではなく，メモリバンクの選択的活性化によっても大きな効果が得られること，また将来的にはスタティック消費電力の削減が重要な課題となることが明らかになった．そこで(a)と(b)については，アーキテクチャ・コンパイラの双方ともオンチップメモリ活用最適化に注力することとし，可視 renaming register に関する研究は断念した．この結果，コンパイラ設計に若干の遅れが生じたものの研究は概ね順調に推移し，ほぼ当初予定通り平成１６年度までに(a)と(b)の研究は完了した．また当初の「性能電力比を２倍とする」目標は，プロセッサ電力の約 1/2 とされているチップ内メモリに関して，ダイナミック消費エネルギーを 30%～60%に，またスタティック消費エネルギーを 25%とすることで概ね達成され，また他の低電力技術よりも優れていることが実証された．

研究期間の後半では２グループが協調しつつ(c)のテーマに注力し，DVFS (Dynamic Voltage and Frequency Scaling)を活用した電力最適化技術に様々な角度で取り組んだ．その結果，当初構想していた実行プロファイル（電力プロファイル）に基づく最適化だけではなく，実行中の電力挙動や性能カウンタ情報に基づく最適化という新たな方向にも研究が展開し，提案・実装した各々の技術でそれぞれ 10%～20%の消費エネルギー削減を，性能にほとんど悪影響を与えることなく達成した．また研究の進展に伴い，本研究のもう一つの柱である「実行モデル」と様々な観点で強く関係することが明らかになり，両者をより密接に結びつけた研究が今後の展開として期待される．

② 大規模並列タスクの実行モデル構築・利用技術

当初の構想では,「並列タスクの実行挙動をあらかじめ把握するための，タスク実行モデルの構築とモデルを利用した実行および環境の管理技術」を，以下の３グループにより研究することとした.

【研究項目３】 安価かつスケーラブルな高速・高信頼ネットワーク技術：ネットワークグループ（リーダ：筑波大・朴）

【研究項目４】 実行モデル構築とモデルを利用した大規模コモディティクラスタ管理技術：クラスタ管理技術グループ（リーダ：東工大・松岡）

【研究項目５】 メガスケールのタスク並列プログラミングとその実行技術：プログラミング技術グループ（リーダ：豊橋技科大／京大・中島）

また具体的な研究成果として，以下の目標を掲げた.

(a) トランキングと冗長リンクの統合による低コスト・高性能・高信頼ネットワーク技術 RI2N の開発

(b) 実行モデルに基づく Soft Failure 検出などの耐故障技術の開発

(c) タスク並列処理とモデル記述のための並列スクリプト言語 MegaScript 設計と実行モデルに基づくスケジューリング

まず(a)については，トランキングによる高バンド幅機能はほぼ順調に開発できたが，リンクの冗長性を利用した耐故障機能との関係で大幅な設計変更を余儀なくされ，当初構想した広域 RI2N は断念せざるを得なかった．しかし最終的に RI2N は，性能・信頼性の両面でほぼ完璧な特性を持つライブラリとして完成し，MegaProto への実装も実現した．また新たな展開として, スイッチ間の複数リンクを VLAN 技術を用いて活用する VFREC-Net の研究を開始し，従来のコモディティネットワークの性能限界を打破する極めて有効な技術を生み出すことができた.

次に(b)については，Soft Failure が生じた場合の挙動モデルの構築が困難を極め，当初予定のスケジュールからはかなりの遅れを生じたが，最終的には実行プロファイルを利用したモデルベースの故障検出・同定に成功し，東工大キャンパスグリッドで運用中の大規模クラスタ上のシステムソフトウェアの不具合を発見するという特筆すべき成果を得た．また大規模システムの耐故障基盤技術の研究開発も精力的に行い，耐故障 MPI フレームワーク Cuckoo をはじめ, チェックポインティングやマイグレーションに関する新たな技術を生み出した.

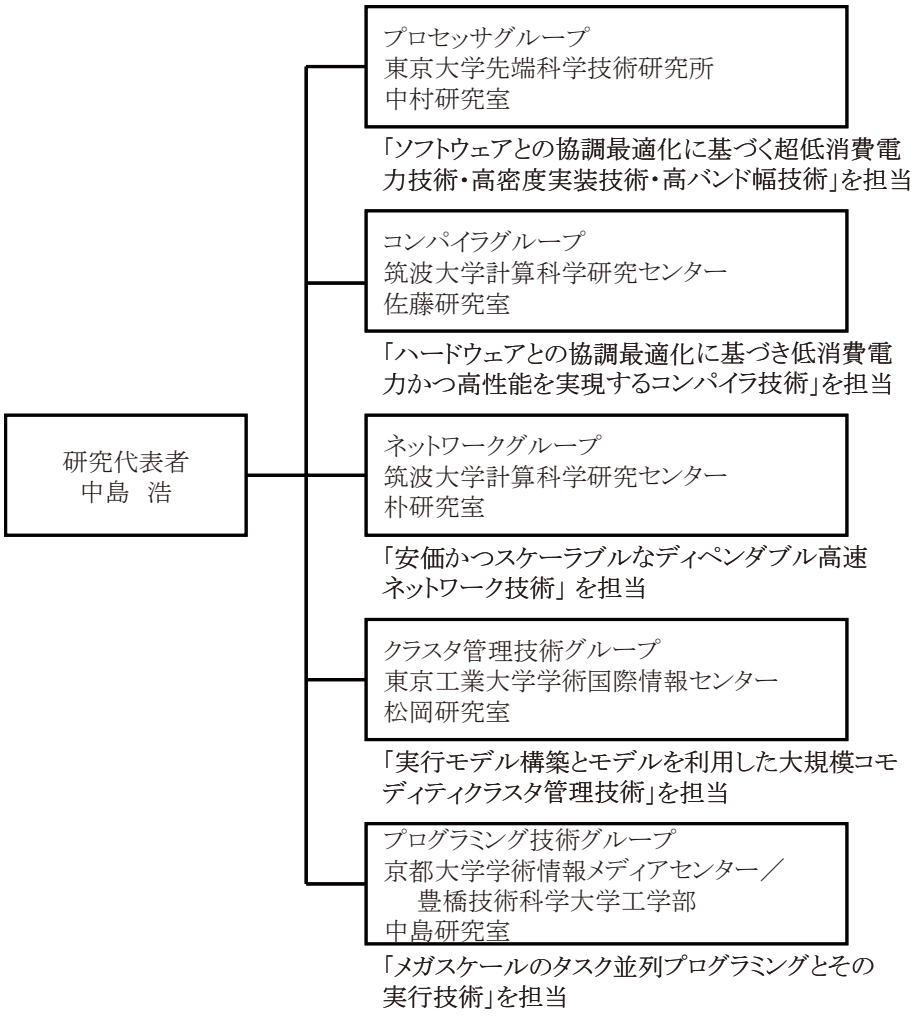
(c)については，言語設計，基本的モデル構築，モデル利用タスクスケジューリング（広域環境を含む）については，ほぼ順調な進展が得られた．しかし MegaScript の記述に基づく静的な初期モデルでは，並列タスクの実行モデルに大きな誤差が生じることが明らかになった．そこでコンパイラグループの協力を得て，プロファイルによるモデル補間・精緻化の研究を新たに実施した結果，高精度のモデル構築が実現できた．さらに実行環境に応じたモデル解釈技術にも取り組み，プロファイルによらない高精度モデルも実現できた.

③　低電力・高密度大規模クラスタ MegaProto

当初の構想では,「研究成果ソフトウェアによって実現される技術を統合したプロトタイプシステムとして，低消費電力かつ高密度実装の $10^3$ スケール大規模クラスタ MegaProto を構築する」ことを目標に，平成１６年度に 1/2 規模のシステムを，また平成１７年度にその改良版として 1/2 規模のシステムを構築することとした．また MegaProto の設計・開発は，各研究グループのリーダを中心としたプロジェクトチームを構成して実施することとした．

しかし研究の進展に伴い，早期にプロトタイプシステムが必要となることが明らかになり，開発時期を１年早め平成１５年度に小規模のプロトタイプ MegaProto/C（32 プロセッサ）を開発することとした．また技術市場の進歩に追随するため，最終システムではより高速のプロセッサを使用することとし，その開発時期も平成１６年度とした．この結果，平成１６年度中には第２バージョンである MegaProto/E の設計作業が完了し，平成１７年度初頭には 320 プロセッサのシステムが完成した．なおシステム規模については，設計外注費が予想を大幅に上回ったため当初構想よりも縮小せざるを得なかったが，開発スケジュールを早めた効果によりプロファイル駆動コンパイラ，RI2N, VFREC-Net, MegaScript などの主要な成果ソフトウェアの実装・評価を行うことができた．

(2) 実施体制


研究代表者
中島 浩
プロセッサグループ
東京大学先端科学技術研究所
中村研究室
「ソフトウェアとの協調最適化に基づく超低消費電力技術・高密度実装技術・高バンド幅技術」を担当
コンパイラグループ
筑波大学計算科学研究センター
佐藤研究室
「ハードウェアとの協調最適化に基づき低消費電力かつ高性能を実現するコンパイラ技術」を担当
ネットワークグループ
筑波大学計算科学研究センター
朴研究室
「安価かつスケーラブルなディペンダブル高速ネットワーク技術」を担当
クラスタ管理技術グループ
東京工業大学学術国際情報センター
松岡研究室
「実行モデル構築とモデルを利用した大規模コモディティクラスタ管理技術」を担当
プログラミング技術グループ
京都大学学術情報メディアセンター／
豊橋技術科学大学工学部
中島研究室
「メガスケールのタスク並列プログラミングとその実行技術」を担当

## 3. 研究実施内容及び成果

### 3.1. ソフトウェアとの協調最適化に基づく超低消費電力技術・高密度実装技術・高バンド幅技術（東京大学　プロセッサグループ）

(1) 研究実施内容及び成果

本プロジェクトの目的は，Peta-Flops クラスの計算能力を有する百万プロセッサ級のメガスケール計算システム構築のための基盤技術の開発であり，そのためには現実的な設置面積・容積と消費電力の制約下で，いかに多数の計算資源を実装して高い性能を得るかという点が重要である．このためには，システム全体の性能を決めるプロセッサが高性能であること，および，システム全体の消費電力の大部分を占めるプロセッサの消費電力が小さいことが必須である．

本グループでは，そのようなプロセッサの実現を目的とし，ソフトウェアとの協調最適化により高性能低消費電力を実現する新しいプロセッサアーキテクチャ SCIMA (Software Controlled Integrated Memory Architecture)を提案・設計・評価した．また従来型プロセッサの省電力ハードウェア機構である DVS を活用して，ハードウェア／ソフトウェア協調による電力最適化を行う技術として，「実効電力制御による高性能・低電力クラスタ」と「統計情報を用いた電力最適化」の２つを提案・実装・評価した．

① ソフトウェア可制御メモリアーキテクチャ SCIMA

図１に示したように，SCIMA に基づくプロセッサには従来のキャッシュに加えて，ソフトウェアから直接参照可能なオンチップメモリである SCM が搭載される．SCM には通常の load/store 命令によりアクセスできる他，page-load/page-store と呼ぶ拡張命令を用いた主記憶との間の一括データ転送が可能である．

また物理的には SCM とキャッシュは一つのメモリ構造を分割する形で実装され，その境界を変更することによってアプリケーションあるいはその実行フェーズに応じた構成とすることができる．具体的には $N$-way セット連想キャッシュの $m$-way を SCM に，残りの($N-m$)-way をキャッシュに割り当てることができる．

② SCIMA での性能最適化

SCIMA での性能最適化は，再利用性を持つデータと，再利用性を持たないデータの双方をターゲットとして行われる．再利用性を持つデータの場合，まず page-load によって主記憶から SCM へデータを転送し，SCM 上で所与の操作を行った後，(必要ならば) page-store により SCM から主記憶へデータを書き戻す．この一連の流れはキャッシュによる高速化と同様であるが，page-load/page-store はキャッシュの block load や write-back よりも効率的であること（特にストライド転送の場合），また SCM 上に領域を確保するため他のデータとの競合が完全に排除できることで，キャッシュよりも SCIMA が優位となる．

一方再利用性がないデータの場合，SCM の一定領域をバッファとして用い，page- load により主記憶からバッファへデータの断片を転送し，SCM 上で所与の操作を行った後，(必要ならば) page-store により主記憶へデータを書き戻す．これに相当する処理をキャッシ

ュで行う場合，一般にデータサイズはキャッシュ容量を大きく上回るため，キャッシュが再利用性のないデータで占有されるという大きな悪影響が生じる．また再利用性がある場合と同様に SCIMA の page-load/page-store が有効に働くが，その効果は再利用性がある場合に比べてはるかに大きい．

```
!$scm opt_notreusable(x,y)
    do i=1,N
        ip = ip + x[i] * y[i]
    enddo
!$scm opt_end(x,y)
```

図 5：ヒント情報による最適化指示

上記の２つの戦略に基づく最適化の手段として，データ転送のタイミングやバッファサイズをプログラマが陽に指示する「ディレクティブ方式」と，データの再利用性に関する情報だけを与える「ヒント情報方式」（図 5）とを提案した．プログラマにとってより使いやすい後者の方式では，コンパイラはまず再利用性のないデータについての最適化，すなわちキャッシュ利用に比べて効果が大きな最適化を，適切なバッファサイズの算出やループ細分化を行いつつ実施する．またその結果 SCM 容量に余裕があれば，再利用性のあるデータについての最適化を，ブロッキングサイズの決定やデータ転送タイミングの調整を行いつつ実施する．

③　SCIMA での電力最適化

SCIMA では，SCM を活用することによってダイナミック電力とスタティック電力の双方を削減することができる．

まず SCM はキャッシュのウェイに相当する複数のバンクから成るが，それらの中から実際にアクセスするバンクだけを活性化することは容易である．これに対しキャッシュでは，一般に全てのウェイを活性化してアクセスし，それらの中から目的のデータのみを抽出するため，SCM に比べてウェイ数倍のダイナミック電力を消費する．また前節で述べた性能最適化によって，オフチップのメモリトラフィックが大幅に削減されるが，これに伴い負荷容量が大きなメモリバスやプロセッサチップの I/O 素子のダイナミック消費電力が削減されることも重要な効果である．

一方デバイスの微細化の進行に伴い，トランジスタのリーク電流が相対的に増大し，これに起因するスタティック消費電力の増加が大きな問題となっている．このリーク電流の削減には，Gated-VDD などによって素子への電源供給を絶つことが最も効果的であるが，SCIMA では SCM の使用領域やその確保・解放のタイミングを事前に知ることができるため，性能に悪影響を及ぼすことなく電源供給を制御することができる．また SCM の使用領域サイズの増減は，実行時間とスタティック消費電力に対して逆の効果をもたらすが，このトレードオフ関係をコンパイラにより解析し，エネルギー遅延積などの指標に基づく最適なサイズ決定ができるのも，SCIMA の重要な特質である．

④　SCIMA の評価

本節では SCIMA での性能最適化と消費電力削減の効果を，キャッシュのみを持つ従来型プロセッサと比較して示す．なお SCIMA では SCM 容量を 48KB，キャッシュ容量を 16KB とし，従来型ではその合計に相当する 64KB のキャッシュを持つものとした．またライン

サイズはいずれにおいても 32B とし，性能指標は従来型のものを 1 として正規化した．

まず NAS Parallel Benchmark の CG と FT，および量子色力学計算を行う実用アプリケーションである QCD を対象に，SCIMA の実行時間とメモリ系の消費エネルギーをシミュレーションにより求めた結果を図 6 に示す．図から明らかなように，従来型プロセッサの２倍以上の速度性能を達成しつつ，メモリ系の消費エネルギーは 30~60%に削減されている．この結果，性能と電力の複合指標の１つであるエネルギー遅延積（ＥＤ積）は，従来型の 1/4～1/9 という極めて大きな改善を達成している．

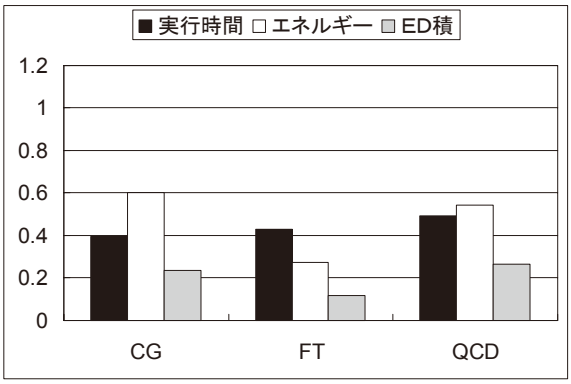



**図 6：SCIMA の性能とダイナミック消費電力**

次に，SPEC CPU95 ベンチマークの 171.swim を対象として，実行時間とスタティック消費エネルギーを求めた結果を図 7 に示す．なお図の横軸はデータ配列ごとに確保する SCM 領域の大きさ（バイト）であり，これを調整することによって実行時間やスタティック消費エネルギーが変化する．図に示すように，実行時間は SCM 領域の増大に伴って単調減少し，4KB では従来型の 1.5 倍程度の性能が達成されている．一方消費エネルギーは，SCM 領域が 256B の時に最小となり，従来型の約 1/4 にまで削減されている．この結果，ED 積も 256B で最小値となり従来型の 1/4 以下の値となっている．このように消費エネルギーや ED 積が SCM 領域サイズに対して単調に変化せず極小点を持つのは，前述のように実行時間と消費電力が領域サイズ増加に対して相反する特性（前者は減少，後者は増加）を持つことによる．すなわち消費エネルギーは両者の積，またＥＤ積はさらに実行時間を乗じたものであるため，この例のように極小点を生じることがしばしばあり，このような最適化が可能なことが SCIMA の重要な特質である．

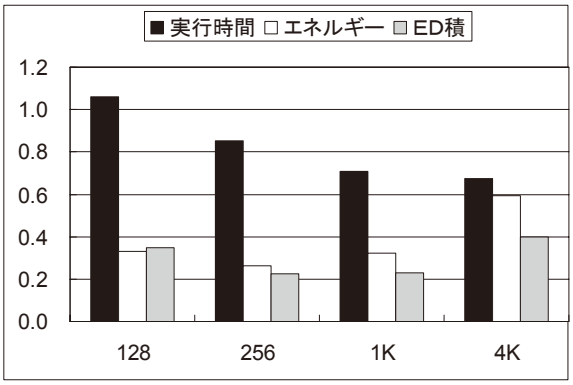



**図 7：SCIMA の性能とスタティック消費電力**

⑤　実効電力制御による高性能・低電力クラスタ

通常のシステムでは，冷却能力などが最大発熱量を定め，それが最大消費電力と最高周波数を定める．したがって省電力化は，性能低下がある許容値以内となるように電圧・周波数を低下させるというアプローチになるが，前述のように実行中のプログラムに対する電圧・周波数低下の影響を見積もるのは容易ではない．

そこで本研究では発想を逆転し，システムの最大消費電力・最高周波数が，最大発熱量で定まる最大許容値を上回るように設定し，実効消費電力を最大許容値にできるだけ近づ

けるように制御することで，高性能と低電力を両立するシステム構成法を考案した．この方法では図 8 に示すように，たとえば 300W の許容最大電力を超えるようなクラスタを構築し，その消費電力を一定時間間隔でリアルタイム計測する．計測した値が許容最大電力を超えている場合には直ちに電圧・周波数を 1 段階低下させ，かつ一定の待機時間の間は上昇させない．一方，待機時間を経過した後は電圧・周波数を 1 段階上昇させ，これを許容最大電力を上回るまで繰り返す．この結果，許容最大電力を上回る逸脱時間の比率は，[計測間隔]÷[待機時間]で抑えることができ，かつ許容最大電力に近い高性能実行を実現することができる．

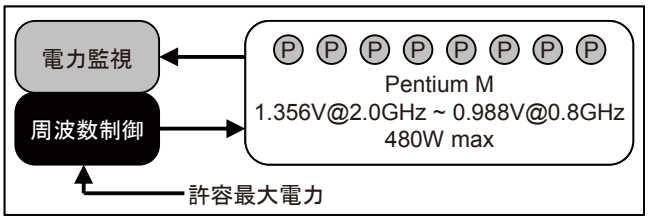



**図 8：実効電力制御による高性能・低電力クラスタ**

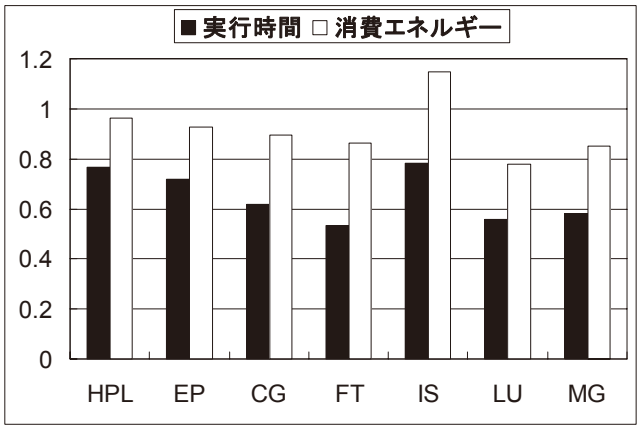



**図 9：実効電力制御の効果**

図 8 に示すシステムを構築し，許容最大電力を 300W，計測間隔を 20msec，待機時間を 2 秒として，HPL (High-Performance Linpack)と NPB (NAS Parallel Bench- marks)の実行時間と消費エネルギーを測定し，図 9 の結果を得た．なお同図は，最大消費電力が 300W となるように周波数を固定した通常の構成方式での実行時間と消費エネルギーを 1 として正規化している．図から明らかなように，実行時間は通常方式に比べ 22～45%削減され，換言すれば 1.3～1.9 倍の性能向上が達成された．また消費エネルギーも IS を除き 4～22%削減され，性能・電力の両面で大きな効果が得られることが明らかになった．

⑥　統計情報を用いた電力最適化

後述する電力プロファイルを用いた最適化は，消費エネルギーなどの正確な見積が可能であるため良好な結果が得られるが，電力プロファイルを得ることは必ずしも容易ではなく，より簡便に得られる情報を用いた最適化も検討する必要がある．本節で述べる方法は，比較的容易に得られるプロファイル情報である性能カウンタを用い，所与の性能低下許容値の範囲内で，できるだけ低い周波数・電圧で実行するように制御するものである．

この方法では図 10 に示すように，周波数を変更したときに実際の性能がどのように変化するかを予測するために，種々の性能カウンタ値の特定の周波数における性能への寄与度を，重回帰分析によって学習する．すなわち，まず種々の学習用ベンチマークプログラムをそれぞれいくつかのフェーズに分割し，個々のフェーズの実行時間と全ての性能カウンタ値を，設定可能な $n$ 通りの周波数ごとに取得する．続いてプログラム実行中に同時に観

測可能な性能カウンタの数を $p$，また装備されている性能カウンタの総数を $q$ としたとき，各フェーズについてある周波数 $v$ での実行で得られた $p$ 個のカウンタ値の線形式で，別の周波数 $u$ での性能が説明できるという仮定に基づき，各変数の重みと定数項の重回帰分析を，${}_qC_p \times n \times (n-1) = N$ 通りの組み合わせ全てについて行う．この結果得られる $N$ 個の線形回帰式の中から，各カウンタ値の寄与度に基づいて最適なカウンタの組み合わせを選択して回帰式 $f(v,u)$ を全ての $v, u$ について定め，学習を終了する．

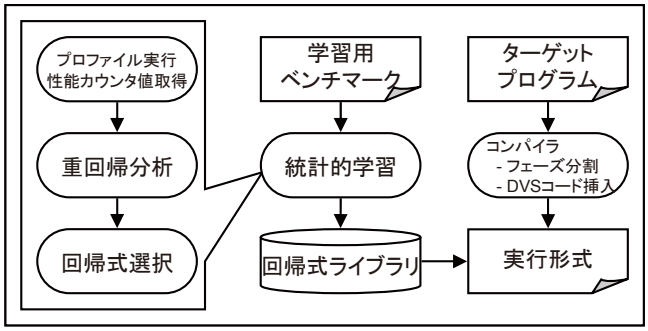



図 10：統計情報を用いた電力最適化

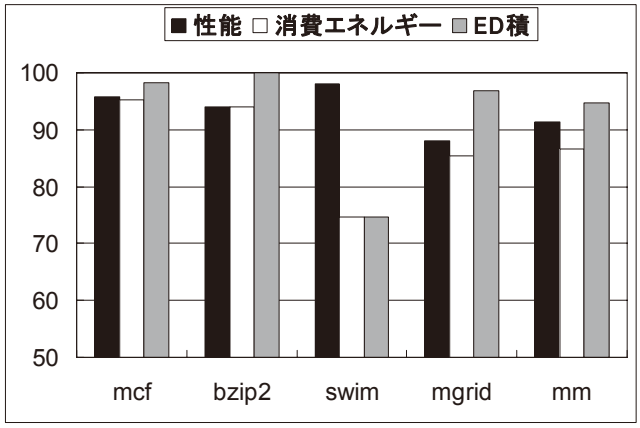



図 11：統計情報を用いた電力最適化の効果

次にコンパイラは，まずターゲットプログラムを解析して関数やループを単位とするフェーズに分割する．続いて各フェーズの末尾に，そのフェーズでの性能カウンタの値を取得し，それに基づき最適回帰式 $f(v,u)$ を現在周波数 $v$ と全ての $u$ について評価し，その結果得られる性能予測値が最高周波数のものに対する所与の性能低下許容範囲（たとえば 10%低下以内）内で最も低い周波数を選択するコードを挿入する．この選択された周波数は，フェーズの先頭に挿入された DVS 制御コードにより設定されるので，次に当該フェーズが実行されるときには可能な限り低い周波数で実行されることとなる．

上記の方式に基づくシステムを実装し，0.8GHz～1.8GHz の範囲で 9 段階の周波数制御を行える Pentium M を対象に実験を行った．Pentium M には 33 個のカウンタが備えられているが，同時に取得可能なものは 2 個に限られるので，線形回帰式は 2 変数となる．学習用ベンチマークとして SPEC CPU 2000 中の 16 種と行列乗算プログラムを用いて統計的学習を行った結果，2 次キャッシュのストアミス回数と，2 次キャッシュの全ミス回数の組み合わせが，最も寄与率が高く，その値は 86.8%であるという結果が得られた．これは 2 次キャッシュミス率が高いプログラムは，メモリアクセス遅延の影響を強く受けるため，周波数低下が性能に及ぼす影響が小さく，特に読み出しミス率（すなわち全体からストアミスを除いたもの）が高いとこの傾向が顕著に現れることを示唆している．

また性能低下許容値を 10%として，SPEC CPU2000 の 4 種と行列乗算(mm)を対象に，フェーズごとの周波数制御を行った結果を，最高周波数での性能指標を 1 として正規化したものを図 11 に示す．図から明らかなように，mgrid 以外では性能低下許容値以内に収ま

っており，mgrid での逸脱も比較的軽微（13.8%低下）である．一方，周波数制御の効果である消費エネルギーや ED 積の削減は swim で顕著に現れており，約 25%の削減が達成されている．他のプログラムについても消費エネルギーは 5〜13%削減できており，手法の有効性が確かめられた．

⑦　成果の位置付け・類似研究との比較

メモリ階層が性能の上でも消費電力の上でも問題であるのは衆目の一致するところで，特にキャッシュメモリの低消費電力化に関しては，多くの研究がなされている．しかし本質的にキャッシュはハードウェア制御であるため，必要なデータと不必要なデータが格納されている箇所を把握することは不可能であり，その結果キャッシュメモリの低電力化技術では何らかの「予測」に基づいて不要な動作を判断し，その部分の電力を削減する手法となる．それに対し，SCM のようなソフトウェアから制御可能なメモリをプロセッサ上に実装する手法は，消費電力上での優位性はあるものの，そのコンパイルの難しさから，実行するプログラムが決まっている組み込みプロセッサのみで採用されている．本研究課題は，実行するプログラムがあらかじめ決まっていない汎用の高性能マイクロプロセッサにおいて，コンパイラとの協調により，低消費電力を達成することに，その意義と独創性がある．

2004 年 11 月以降の TOP500 リストで世界最高速を維持している BlueGene/L は，その高性能・低消費電力・高密度実装において，本プロジェクトの目指す方向を共有しているが，そのプロセッサが搭載するレベル 3 キャッシュは，SCIMA の SCM と同じように，アドレス空間上主記憶の一部として利用できるハードウェア機構を持っている．その意味で着目点は同じであるが，コンパイラによる最適化はサポートされていない．また最近は Cell や ClearSpeed などの専用プロセッサチップやアクセラレータが注目されており，SCM に類似したオンチップメモリを搭載しているが，やはりコンパイラでの最適化は実現できていない．これらに対し，コンパイラとの協調で高性能・低消費電力を達成する本研究は，独創的であり優位性を確保している．

DVS を利用して消費エネルギーや ED 積を最適なものとする研究は，我々のコンパイラグループによるものも含めて数多くあり，また消費エネルギー等の電力に関する指標を制約条件として性能を最大化する研究もいくつか報告されている．しかし⑤で述べた実効電力制御手法は，通常はハードウェア設計段階で超過しないように定められる最大消費電力を，ソフトウェアによる最適化の制約条件として利用して性能を最大化するという発想が，これらの類似研究とは本質的に異なり，かつ優位性をもたらす重要な特徴となっている．

また⑥の統計情報を用いた最適化は，DVS の動的制御により性能劣化を最小限に留めて電力を削減するという点では，多くの電力最適化の研究と共通性を有している．しかし性能カウンタを用いて電圧・周波数と性能の関係を予測し，かつ予測モデルを統計処理によって構築することが，他の研究にはないユニークな特徴となっている．

(2) 研究成果の今後期待される効果

前述のように，SCM に類似したメモリを搭載した高性能プロセッサやアクセラレータが具現化しており，本研究の着眼点が正しいものであったことが証明されるとともに，研究成果である最適化コンパイル技法への期待が高まっている．また従来からオンチップメモリが一般的である組み込み系のプロセッサにおいても，ソフトウェア生産性向上の観点から，やはりコンパイラによる自動的な割付等の必要性が急速に高まっている．このように SCIMA に関する本研究の成果が直接的・間接的に生かされ，高性能・低電力システムのキー技術として貢献する可能性はきわめて高い．

本研究の後半で実施した２つの電力最適化の研究成果は，実用に向けての課題はいくつかあるものの，それぞれユニークな技術として注目を集めており，今後の発展が大いに期待される．また，実効電力制御の手法では動的なプロセス数の増減，統計情報による最適化では DVS 以外の可制御機構への適用など，新たな展開も期待できる．

## 3.2. ハードウェアとの協調最適化に基づき低消費電力かつ高性能を実現するコンパイラ技術（筑波大学　コンパイラグループ）

(1) 研究実施内容及び成果

本グループでは，メガスケールコンピューティングの実現に向けて，低消費電力化・性能モデリングを支援するコンパイラ技術と実行時支援ソフトウェアの研究開発を行った．研究期間の前半では，主としてプロセッサグループ，クラスタ管理技術グループ，プログラミング技術グループとの連携し，SCIMA コンパイラをはじめ各グループで必要なコンパイル技術やツール開発の基盤提供と支援を行った．

また後半では，研究開始当初からの重要な課題であったプロファイル駆動最適化に精力的に取り組み，消費電力のプロファイルと DVS を利用して消費エネルギーや ED 積を最適化する技法を提案・実装・評価した．また研究途上での新たな展開として，負荷が不均衡な粗粒度タスク並列処理を対象に，余裕時間を持つタスクの電力を抑制して性能を維持したまま消費エネルギーを低減する手法を提案・実装・評価した．

① プロファイルを用いた電力最適化

DVS は，プロセッサの電源電圧と動作周波数を段階的に変更するための機構であり，低電圧・低周波数では低電力だが低性能，高電圧・高周波数では高性能だが電力消費も大きい，という相反関係がある．したがってプロセッサに高い演算負荷がかかっている状況では高電圧・高周波数に，またキャッシュミスや I/O が頻繁に生じる状況では低電圧・低周波数とし，状況に応じた制御を行えば消費エネルギーやＥＤ積を削減することができる．しかし LongRun や SpeedStep などの DVS 制御機構は，過去の演算負荷に基づいて電圧・周波数を変更するため，負荷変動が激しいプログラムには追従できず，また電圧・周波数の増減がどのような性能をもたらすかの認識にも欠けている．

そこで本研究では，コンパイラがプログラムの各部分に対する最適な電圧・周波数を定

める方式の一つとして，各部分の電力および性能プロファイルをあらかじめ収集し，それに基づき消費エネルギーあるいは ED 積を最適化する方法を考案・実装した．この方法は図 12 に示すように，まずプログラム中のフェーズ（ここでは計算／通信関数）ごとの電力および性能プロファイルを，電圧・周波数を変更しながら収集する．次に，電圧・周波数の変更に伴うオーバヘッドを勘案しつつ，取得したプロファイルに基づき所与の指標，たとえば消費エネルギーを最小化する電圧・周波数を，プログラムの各フェーズごとに求める．最後に，求まった最適な電圧・周波数に設定するためのコードをプログラムの各フェーズに埋め込んだ実行形式を生成する．

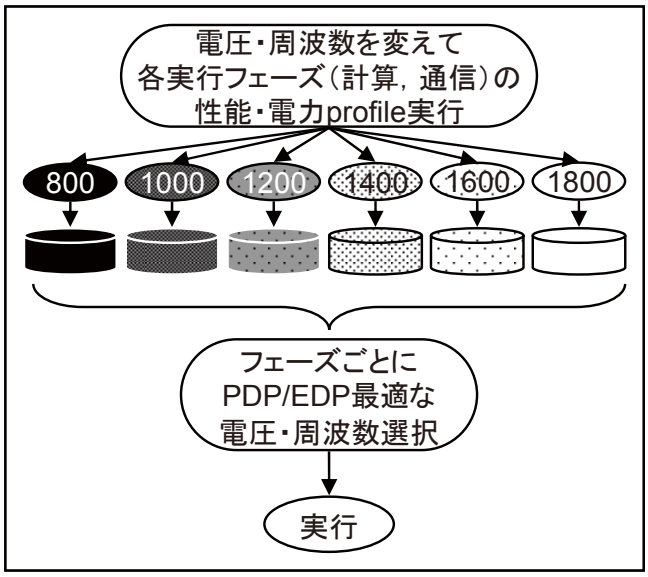



図 12：プロファイルを用いた電力最適化

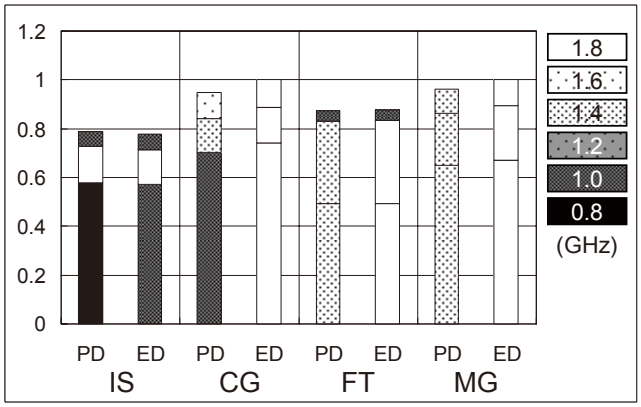



図 13：プロファイル駆動電力最適化の効果（Turion クラスタ）

図 13 と図 14 に，8 ノードの Turion クラスタ（最高周波数 1.8GHz）と，4 ノードの Crusoe クラスタ（最高周波数 933MHz）を用いて行った実験の結果を示す．これらの図で PD および ED は，消費エネルギー（PD 積）と ED 積を最適化指標とした場合の，それぞれの指標値を最高周波数に固定したときの値を 1 として正規化したものである．また棒グラフの濃淡は，対応する周波数で実行されたフェーズにおける，各指標の全体への寄与度を表している．さらに Crusoe クラスタでは，自動 DVS 制御機構である LongRun を用いたときの値も示している．

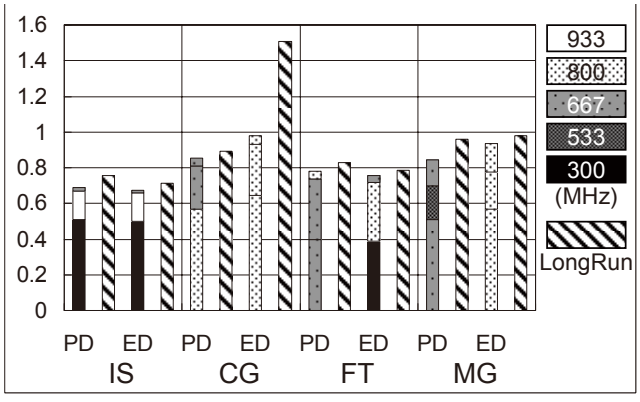



図 14：プロファイル駆動電力最適化の効果（Turion クラスタ）

図から明らかなように，Turion の CG と FT で ED 積を最適化指標とした場合を除き（これらは常に最高周波数での実行が最適），本手法を適用することにより消費エネルギーや

ED 積が改善されている．すなわち Turion では消費エネルギーが最大 21%，ED 積が最大 22% 改善され，Crusoe では消費エネルギーが最大 31%，ED 積が最大 32% 改善されている．また Crusoe の LongRun と比較しても常に本手法の値が優れており，特に CG の ED 積は LongRun の 2/5 以下の値となっている．これは CG で頻繁に実行される小メッセージの送受信関数実行に LongRun が追随できず，過度に低い周波数を選択したままとしてしまうことが要因であり，本手法のように正確に消費エネルギーや ED 積を見積もって制御することの重要性を示している．

② 負荷不均衡プログラムの電力最適化

①で述べた最適化手法はプログラムの実行挙動の観測結果を利用するものであったが，本節で示す手法はプログラムの意味的な挙動特性を認識することで，より効果的で確実な最適化を目指すものである．具体的には，計算ノードの負荷が均衡しない並列プログラムを対象に，負荷が軽いノードの周波数・電圧を下げてその実行時間を負荷が重いノードに合わせることで，全体の性能を損なうことなく電力を削減する手法である．

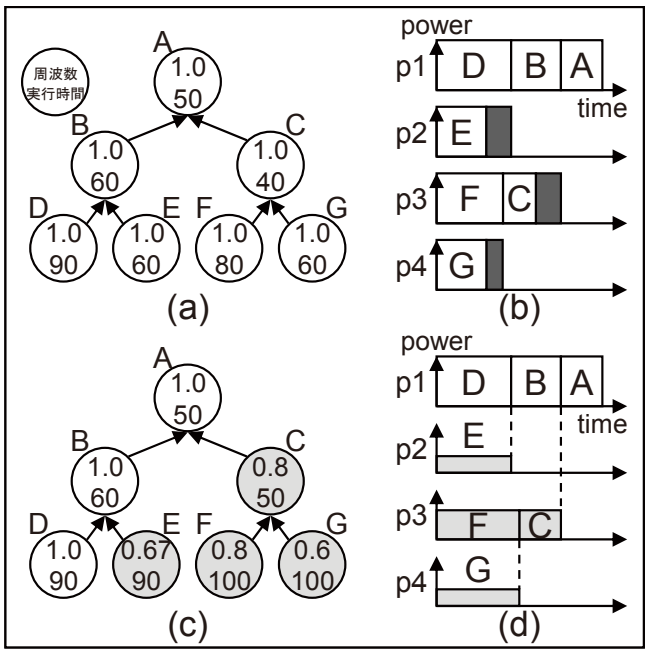



図 15：負荷不均衡プログラムの電力最適化

この手法では，たとえば図 15(a)に示す負荷が均衡しないタスクグラフに対し，それを最高周波数で実行したときには同図(b)に示すような余裕時間が，タスク C・F の組，タスク E，タスク G に生じることを利用する．すなわち同図(c)に示すように，クリティカルパスを規定するタスク B, D, F の終了時刻を超えない範囲で，余裕時間を持つタスクの周波数・電圧を下げて意図的に実行時間を延ばす．その結果，同図(d)に示すように余裕時間を持っていたタスクの消費エネルギーが削減される．

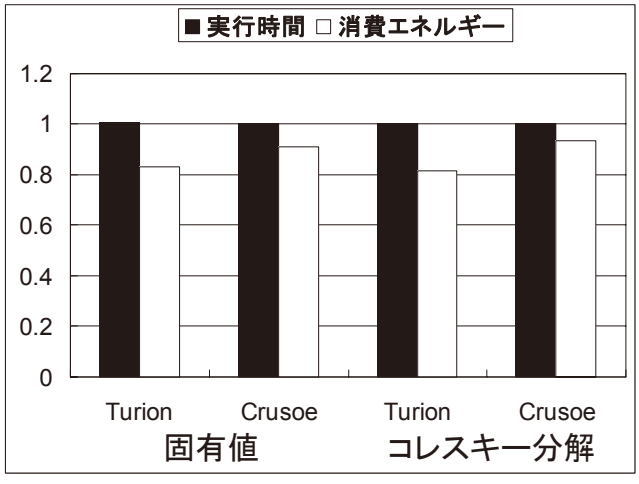



図 16：負荷不均衡プログラムの電力最適化の効果

この手法を，8 ノードの Turion クラスタと 4 ノードの Crusoe クラスタを対象に，マスターワーカ型の固有値計算問題と，コレスキー分解に適用した結果を図 16 に示す．同図の

性能値は，最高周波数・電圧で実行した場合の値を 1 として正規化したものである．図から明らかなように，余裕時間を持つタスク実行時の周波数・電圧を低下させても，全体の実行時間はほとんど増加しない（最大でも 0.81%の増加）．一方，消費電力については，特に Turion クラスタで 17〜18%の顕著な削減効果が見られ，Crusoe クラスタでも 7〜9%の削減が達成されたことから，本手法の有効性が確認された．

③　成果の位置付け・類似研究との比較

消費電力をソフトウェア技術で削減する取り組みは，DVS が登場して以来精力的に行われている．特に高性能計算の分野では電力削減の絶対量が大きいため，我々の研究着手と時を同じくして世界的に活発な研究が始められた．前述のように，DVS を利用して消費エネルギーや ED 積を最適なものとする研究は数多く行われているが，電力プロファイルという確実な挙動情報を用いているものは①で述べた我々の研究のみであり，これは我々が開発した精密な電力測定技術に負うものである．また電圧・周波数の切替に要するオーバヘッドを無視することはできないが，これを勘案して最適な値を定めることも本研究の重要かつユニークな特色である．また粗粒度タスクの余裕時間に着目した低電力化手法は，②で述べた我々の研究以外にもいくつか提案されているが，コンパイル時の静的な電圧・周波数割付や一般のタスクグラフへの適用性など，他の研究にはないユニークな特徴を有している．

(2)　研究成果の今後期待される効果

前述のように，高性能計算分野では電力削減の効果が高く，したがってソフトウェアによる電力最適化の必要性や期待も大きい．そのような状況下において，国内でこの分野の研究を実施しているのは実質的に我々のグループのみであり，今後も海外の研究と比肩する研究を継続することが我が国の高性能計算の発展に不可欠と言っても過言ではない．

また本グループでの研究の進展に伴い，プログラムの挙動把握や予測の重要性が改めて認識され，計算・通信状況の適切なモデリングとそれに基づく電力最適化が今後期待される．本研究での静的なモデル構築技術は，現時点では残念ながら電力に関する情報を対象とできてはいないが，たとえば通信量や同期タイミングといった情報は本グループの成果技術と密接に関係しており，これらの技術を融合したより効果的な最適化技法に展開することが期待できる．

### 3.3.　安価かつスケーラブルな高速・高信頼ネットワーク技術（筑波大学　ネットワークグループ）

(1)　研究実施内容及び成果

本グループの研究は，百万プロセッサ級のメガスケール計算システム構築のための基盤技術の一つとして，大規模コモディティクラスタにおける相互結合網における耐故障性に注目したものである．コモディティクラスタにおいては，ノード間通信，クラスタ管理システム，ユーザアプリケーションの各レイヤーにおける耐故障性が考えられる．ユーザ透

過な耐故障性をシステム性能を損なわずに提供するには，ノード間通信部で最低限の耐故障性を保障し，クラスタ管理システムでチェックポインティング等による耐故障性を持たせることにより，チェックポインティングのコスト（負荷）を低減させ，システムの信頼性を上げることが有効な戦略であると考えられる．コモディティクラスタでは安価で高性能なネットワークリンクを複数用いることが可能であり，これをもって冗長性のあるネットワークが構築可能である．また，非故障発生時にはこれらのリンクを全て有効利用し，バンド幅の向上を図ることができる．

以上のようなコンセプトの下，マルチリンクによる高バンド幅・耐故障性を持つコモディティネットワークとして，計算ノードが複数のネットワークリンクを備えた RI2N (Redundant Interconnection with Inexpensive Network) を提案し，その実装と MegaProto および他のシステムにおける有効性を示すことを本テーマの目標とした．また研究の進展に伴い，計算ノードだけではなく，ネットワークシステムを構成するスイッチ間にも複数のリンクを装備し，それらを VLAN 技術によって活用してさらなる高バンド幅を実現する VFREC-Net (VLAN-based Flexible, Redundant and Expandable Commodity Network) を想起するに至り，RI2N と併せて実装・評価を行った．

① 多重リンクによる高バンド幅・高信頼ネットワーク RI2N

RI2N は，複数のネットワークリンクを束ねて論理的に高バンド幅のリンクとするトランキング技術と，リンクやスイッチの故障が生じたときにネットワークを縮退して通信機能を確保する耐故障技術とを，初めて統合して実現した新たなコモディティネットワーク技術である．図 17 に示すように，RI2N は各リンクに対応する UDP ソケットを経由した通信を行うユーザレベルのスレッドとして実現されている．

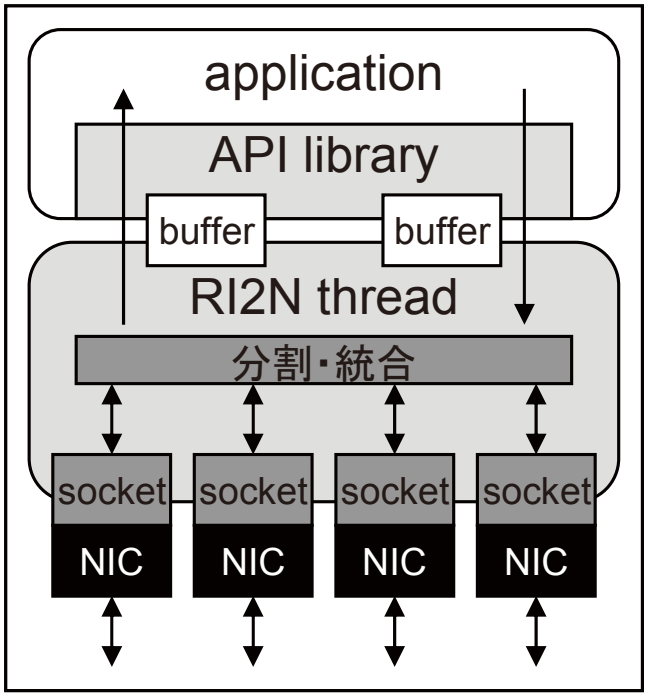



図 17：RI2N のアーキテクチャ

RI2N による通信は以下の手順で行われる．まずアプリケーションが送信するデータは，chunk と呼ぶ適当な大きさの断片に分割され，各断片は round-robin 選択される各ソケットから順次ネットワークへ送信される．この結果，データが各リンクにほぼ均等に分配されて送信されるため，データサイズが十分に大きければ，リンク数に比例したバンド幅の向上が期待できる．一方受信側では逆に各ソケットから順次データを受け取るが，送信順に受信するとは限らないため，パケットに付されたシーケ

ンス番号に従って送信順に並び替えて送信データの再構築を行う．また UDP には TCP のようなフロー制御機能がないため，RI2N スレッドがウィンドウを用いた制御を行う．

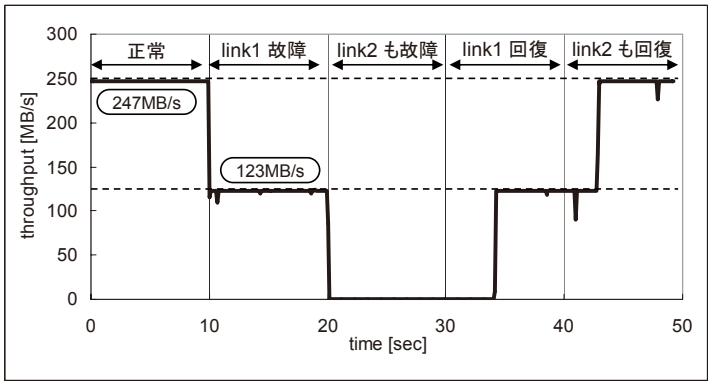



**図 18：RI2N の通信性能**

RI2N の耐故障性は，複数のメカニズムによって実現される．まず受信側ではシーケンス番号により未着パケットを認識し，最古の未着パケットと最新の既着パケットとの番号差が閾値以上になると，未着パケットが失われたものとみなし，全てのリンクを介して送信側に再送を要求する．この機能により，ネットワークの一時的輻輳やフロー超過による間歇的なパケット消失に対応できるほか，リンク切断のような恒久的な故障にもデータの保障という観点からは対応できる．

しかし再送機能のみでは，故障リンクに対してもパケット送出を繰り返すため，故障リンクを放置しておくと性能的に大きなダメージが生じる．そこで受信側は各リンクのパケット到着数をカウントし，特定のリンクの到着数が著しく低い場合には，これを故障リンクとして送信側に通知する．この故障通知を受けると，送信側は問題のリンクへの送出を停止するので，故障のないリンク群のみを使って性能的にも最善の縮退通信へ迅速に移行できる．またリンク接続の不良など修復可能な故障については，全リンクに定期的に送信するハートビートパケットの到着によって修復完了を認識し，それ以後は故障していたリンクによる送信も再開して最高性能を回復する．

上記の実装に基づく RI2N の性能と耐故障機能を，Xeon 2 台を 2 本のギガビットイーサネットリンクで結合したものを用いて評価した．この評価実験では，まず 2 つのリンクが正常動作している状況で継続的に通信を行ってスループットを計測し，10 秒後に 1 本のリンクを切断，さらに 10 秒後にもう 1 本のリンクも切断して縮退時のスループットを計測した．また 2 本目のリンクを切断して 10 秒後には最初に切断したリンクを再接続し，さらに 10 秒後には 2 本目のリンクも再接続して，故障修復時の回復機能も評価した．

図 18 に示した測定結果から明らかなように，2 本のリンクが正常であるときのスループットは 247MB/sec であり，理論最大性能である 250MB/sec の 98.8%という，極めて高い性能が確認された．また一方のリンクが切断されているときの性能も 123 MB/sec であり理論最大性能の 98.4%が達成されている．さらに，リンク切断時には速やかに縮退通信へ移行することと，故障修復後は比較的短時間に性能が回復することも確認できた．

② VLAN を用いた多重パスネットワーク VFREC-Net

RI2N を用いることで計算ノードあたりの通信性能はリンク数にほぼ比例して向上する

が，多数のノードからなるシステム全体の通信性能をコモディティ技術と複数リンクの組み合わせで向上させるためには異なる技術が必要となる．すなわち大規模ネットワークの構築には，図 19(a)のような全ノードを直接結合するような多ポートスイッチの利用は非現実的であり，一般には安価な少ポートの L2 イーサネットスイッチを用いて同図(b)のような多進木構造を用いる．しかしこの構成では，明らかに上位のスイッチの性能がボトルネックとなってしまう．一方，高スループットの上位層接続を実現する方法として，同図(c)や(d)に示す fat-tree などの多段接続が考えられるが，L2 スイッチの基本機能ではノード間のパスが一意に定まる必要があるため，結果的に上位層に存在する豊富なパスを活用することができない．

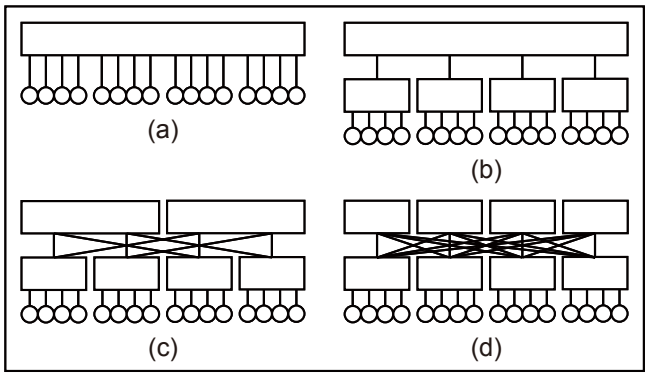



図 19：16 ノードネットワークの構成例

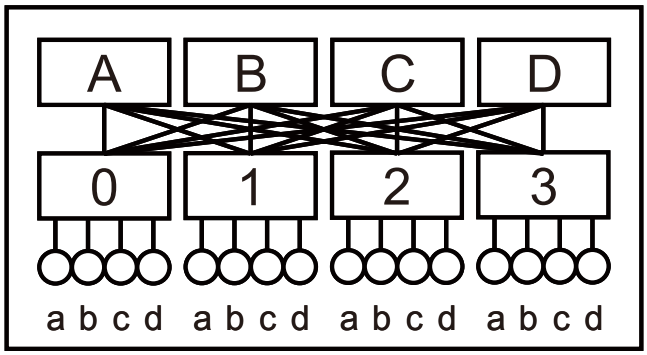



図 20：VFREC-Net の構成例

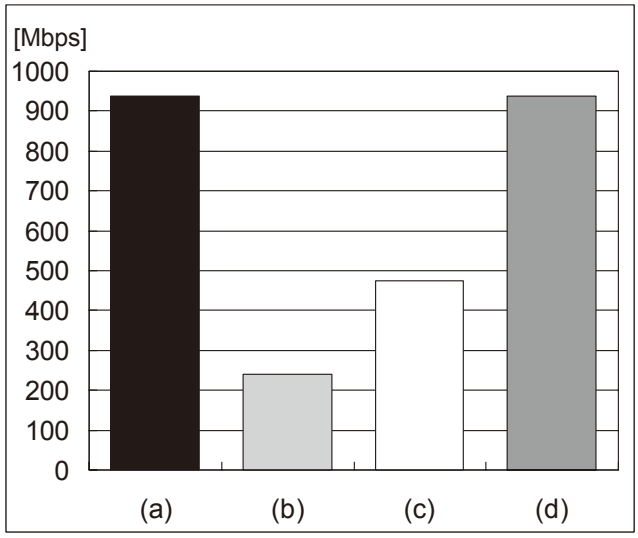



図 21：VFREC-Net のバンド幅

そこで本研究では，L2 スイッチにも備えられている tagged-VLAN の機能を用いて，図 19 (c)や(d)のような非多進木構造を可能とし，かつその上位層のスイッチ間多重パスを活用する方式である，VFREC-Net を提案した．図 19 (d)の fat-tree を構成する場合，VFREC-Net では図 20 に示すように４つの上位スイッチ A～D をルートとする４つの４進木を考え，各４進木に VLAN 識別子 A～D を与える．また下位層スイッチ $i$ ($0 \le i \le 3$) に接続される計算ノードを $x_i$ ($x \in$ {a, b, c, d}) としたとき，ノード $x_i$ と $y_j$ ($i < j$) 間の通信には VLAN: $X$ ($X \in$ {A, B, C, D}) を用いるように制御する．この結果，システム内の通信は４つの VLAN に平均的に分散され，下位層と上位層の間のリンクも平均的に使用されることとなり，リンク数に比例した高いバンド幅を得ることができる．

VFREC-Net の性能を評価するために，3.0GHz の Xeon PC をノードとする 16 ノードク

ラスタを，図 19 に示した４種類の構成からなるギガビットイーサネットのネットワークを用いて，２種類の実験を行った．まず図 21 は，システムの左半のノードと右半のノードとのペアを作り，合計８ペアで一対一の片方向通信を行ったときの，ペアあたりの転送スループットを示したものである．図 21 から明らかなように，(b)の４進木構成では上位層スイッチがボトルネックとなって，理論最高性能 1Gbps の約 1/4 の性能となっている．一方 VFREC-Net では上位層と下位層の間のリンク数を(c)２倍あるいは(d)４倍にした fat-tree では，リンク数に比例した性能向上が得られ，(d)では 16 ポートスイッチを用いた１階層構成(a)とほとんど同じ性能が達成されている．

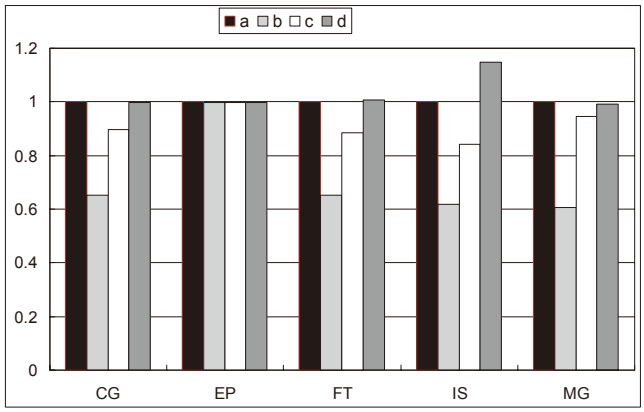



**図 22：VFREC-Net の NPB 性能**

次に NPB の５つのプログラムを用いた評価結果を，(a)の１階層構成の性能を１に正規化して図 22 に示す．通信がほとんど行われない EP を除き，(b)の４進木構成では(a)の構成の約 60%程度の性能しか得られない．これに対し(d)の fat-tree では，(a)の構成に匹敵する，あるいはむしろ凌駕する性能が得られている．なお構成(d)と構成(a)は図 20 に示したように，二分バンド幅はほぼ等しいため，構成(d)の性能が高い理由は，all-to-all 型の集合通信でのパケットロスの頻度に関係しているものと推測される．

③　<u>成果の位置付け・類似研究との比較</u>

マルチリンクネットワークの利用による高バンド幅化・耐故障機能実装については，いくつかの研究が進められている．高バンド幅化に限って言えば，SCore/PM デバイスによる GbE トランク利用技術が著名であり，最大で４リンクまでの GbE トランクによるスケーラビリティが確認されている．マルチリンク利用の規格化としては，IEEE802.3ad link aggregation がある．これは，１つの Ethernet スイッチに接続された複数の平行リンク間で高バンド幅化と故障時の代替リンク利用を可能とするものである．また VLAN をベースとしたマルチリンクの活用法としては，工藤らによる研究が知られている．

これらの類似研究に対し，まず RI2N はハードウェア規格にとらわれない完全なソフトウェア実装であることと，高バンド幅化と耐故障機能をシームレスに実現していることが特徴である．すなわち SCore/PM で耐故障機能はなく，IEEE802.3ad link aggregation では単一スイッチへの集約が条件のためスイッチそのものの故障に対して無力である．また工藤らによるマルチパスリンクでは経路ごとに異なる IP アドレス空間を必要とするので，大規模システムでの実装は不可能に近いほど煩雑であるが，VFREC-Net は単一空間で実現されるため実装は極めて容易である．

(2) 研究成果の今後期待される効果

本研究が進行している過程において，高性能システムのネットワーク技術は急速にコモディティ化しており，数年後の世界最高速システムは InfiniBand など汎用性が高いネットワーク技術で実現される可能性が高い．この点からも，我々の着想が妥当であったことが実証されている．一方 InfiniBand や Myrinet などは Ethernet に比べると普及度に大差があり，NIC やスイッチなどのコスト性能比は依然として Ethernet の優位が続くものと考えられる．

本グループの成果である RI2N と VFREC-Net は，真のコモディティネットワークである Ethernet に適用でき，その技術進歩，たとえば 10Gbps やそれ以上のバンド幅向上に対しても，常に追随可能な汎用的技術である．また OS やミドルウェアに関する制約やハードウェア機器の必要性もなく，一般的なクラスタに直ちにかつ容易に搭載可能である．したがって実用化の可能性が極めて高く，直接的に貢献可能な高性能計算技術として強く期待できる．

本グループでは，研究成果の今後期待される効果そのようなプロセッサの実現を目的とし，ソフトウェアとの協調最適化により高性能低消費電力を実現する新しいプロセッサアーキテクチャ SCIMA (Software Controlled Integrated Memory Architecture)を提案・設計・評価した．また従来型プロセッサの省電力ハードウェア機構である DVS を活用して，ハードウェア／ソフトウェア協調による電力最適化を行う技術として，「実効電力制御による高性能・低電力クラスタ」と「統計情報を用いた電力最適化」の２つを提案・実装・評価した．

### 3.4. 実行モデル構築とモデルを利用した大規模コモディティクラスタ管理技術（東京工業大学 クラスタ管理技術グループ）

(1) 研究実施内容及び成果

メガスケール規模のクラスタ環境での効率的なアプリケーション実行には，耐故障性の実現が必須である．我々は耐故障性の実現の鍵が，アプリケーションの挙動や故障によって生じるその変化をモデル化して予測する技術にあるという認識に基づき，soft failure と呼ばれる間歇的で検出困難な故障の検出手法を研究した．またチェックポインティングやリカバリに関する新たな技術を考案し，さらにそれらを統合する耐故障 MPI フレームワークである Cuckoo の提案・実装も行った．

① 耐故障 MPI フレームワーク Cuckoo

大規模な並列システムにおける耐故障戦略には多種多様なものがあり，特定の戦略だけでは多様な計算環境，故障原因，アプリケーションに適切に対応することはできない．たとえば一般的な耐故障機能であるチェックポイント・リカバリを例に取ると，チェックポイントを保存する方法も，また故障が検出された際に保存したチェックポイントを用いて実行を継続する方法も，それぞれ様々なものが提案されているが，これらは計算環境（た

とえば冗長計算ノードの有無や数量)，故障の原因 (たとえば計算ノードなのかネットワークなのか)，アプリケーションの性質 (通信頻度が高いか低いか) などによって，適合性が大きく異なる．

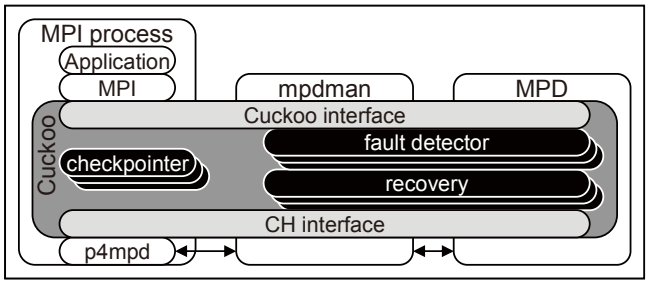



**図 23：Cuckoo のアーキテクチャ**

そこで本研究では，多様な耐故障機能を部品として取扱い，それらの適合性を左右する種々の外的要因に応じて適切なものを選択可能とするフレームワーク Cuckoo の提案・実装を行った．Cuckoo は図 23 に示すように，MPI 実装の一つである MPICHI-p4mpd を拡張する形で実現されている．すなわち，各種のチェックポイント機構(checkpointer)，故障検出機構(fault detector)，故障回復機構(recovery)は Cuckoo に内包され，アプリケーション，計算環境，故障原因によって適切なものが選択できる構成となっている．

たとえばチェックポイント機構には，通常の coordinated checkpointer の他に，log-base の uncoordinated checkpointer や，本研究の成果の一つである投機的チェックポインタなどを，アプリケーションの特性に応じて選択できる．また故障検出機構にも，各ハードウェアやプロセスのモニタ機構のほか，2.5.3 節で述べるモデルベース検出器などが利用可能である．さらに故障原因により，チェックポイントからの再実行を同一ノードで行うか，他のノードに移送して行うかなどの戦略を定め，それに基づく回復操作を実施することができる．このように Cuckoo は，本研究による様々な耐故障技術を組み込むプラットフォームとして機能するだけではなく，高度な知識・経験を必要とする耐故障戦略の選択を容易に実現する枠組みとして極めて有用なミドルウェアである．

② VM を用いたワークロード移送技術

ある計算ノードで実行中のプロセスを他のノードに移して実行を継続するプロセス移送は，前述の故障回復の有力な手段であるだけでなく，計算環境の保守や動的な負荷調整にも有用である．しかしプロセスを移送するためには，プロセスと OS との相互作用に関する状態も移送する必要があり，これは必ずしも容易ではない．また MPI プロセスのような並列プロセスでは，他のプロセスとの接続関係を変更する必要があり，これにも煩雑な手間を要する．

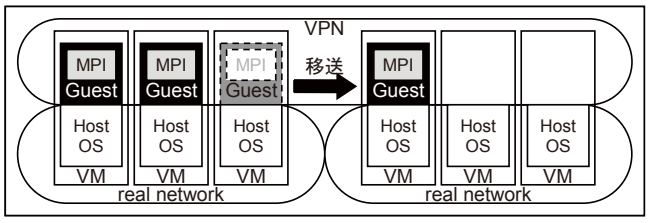



**図 24：VM を用いたワークロード移送**

そこで本研究では，プロセスではなくそれを管理している OS ごと他の計算ノードに移送することで，上記の問題を解決する低コストの移送技術を提案・実装した．この方法は図 24 に示すように，まず計算ノード上で，複数の OS が稼動可能な仮想計算機(VM)を動作させ，

その管理下にベースとなるOS（ホストOS）と移送対象とするOS（ゲストOS）の二つを稼動させる．MPIアプリケーションプロセスは，ゲストOSの管理下で実行し，このプロセスを移送する際にはゲストOSも含めることで，プロセスとOSの相互関係もそのまま移送できる．またネットワークにVPN (Virtual Private Network)を用いることで，MPIプロセス間の通信は実ネットワークと切り離して管理することが可能となり，移送の際にも接続を維持したままとすることができる．

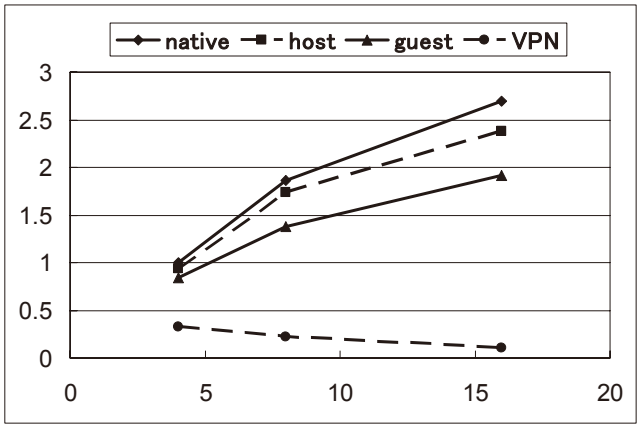



図25：VM上のCGの性能

この方式で性能上の問題となりうるのは，実計算環境の資源，たとえばネットワークへのアクセスのために通過すべきソフトウェアスタックの階層が増加することによるオーバヘッドである．この点に関する評価を，通信性能が全体性能に与える影響が大きいプログラムであるNPBのCGを対象とし，Opeteron242 PCをギガビットイーサネットで接続したクラスタにVMとしてXenを搭載したものを用いて評価した．図25は，4ノードでVMを使用しない場合(native)の性能を1として，VMのホストOSの支配下で実行した場合(host)，ゲストOSの支配下で実行した場合(guest)，およびゲストOSの支配下でVPNを用いて実行した場合(VPN)について，それぞれの4, 8, 16ノードの性能を示したものである．

図から明らかなように，OSスタックが深くなるに連れて性能は劣化しているが，CGのように通信頻度が高いプログラムに対してguestの性能はnativeの70%以上となっており，通信頻度が低いEPなどでは95%以上の性能を得られることを勘案すると，満足できるレベルであると考えられる．一方VPNを用いた場合の性能は著しく劣化しており，この実験で用いたOpen-VPNの性能上の問題が明らかになった．またOpen-VPNは広域環境での性能に問題があることも判明しており，これらの改善課題となった．

③　モデルベースの故障検出技術

大規模なシステムの内部で生じる故障は，たとえばシステムや構成要素の停止といった明確な「症状」を呈するとは限らない．たとえば一過性の例外に対応するための再試行はシステムの様々な部分で行われるが，この機能が真の故障，すなわち固定的ではないが頻繁に生じる例外を隠蔽してしまうことがある．また例外を検出して対応するメカニズムが全て完璧であることは期待できず，例外の組み合わせや発生タイミングによって誤った対応をして，結果的に故障を隠蔽してしまうこともある．さらに，故障が何らかの形で顕在化しても，多数の例外対応機構を伝播している間に，真の故障原因が不明瞭となることもしばしばある．

本研究では，このような潜在的でかつ間歇的にしか生じない故障を検出し，さらにその

原因を追究するために，並列動作するプログラムの実行過程を関数実行時間によってモデル化し，異常な振る舞いをする要素を発見するモデルベースの検出技術を設計・実装した．また，東京工業大学のキャンパスグリッドの主要な構成要素として運用稼動している 129 ノードのクラスタを対象に，その重要なミドルウェアである SCore の例外的挙動を例に評価した．

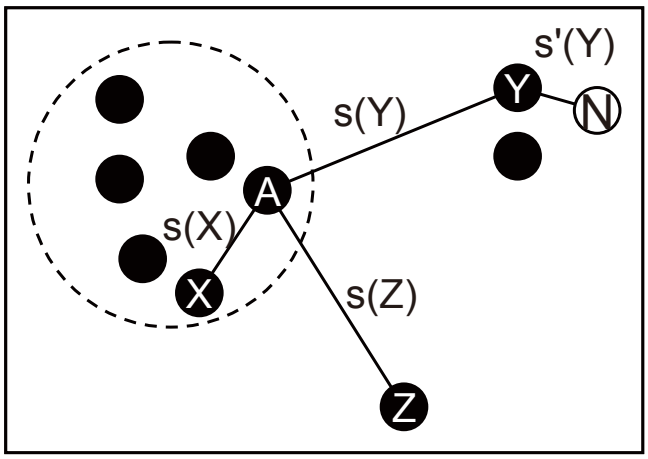



**図 26：容疑度による故障検出**

この方法ではまず，プログラム中の全関数呼出点にインストルメントを施して，呼出と復帰の時刻をメモリ上にトレースする．次に何らかの異常，たとえばプログラムの停止や著しい性能劣化が観測された時点で，上記のトレースを解析して故障の有無や原因の同定を行う．

故障がプログラムの停止といった形で明確に顕在化する場合，各ノードのトレースの末尾の時刻を比較することにより，他のノードと著しく異なる時刻を示すノードを故障原因として特定できる場合が多い．実際，２ヶ月間の実験期間中に数回だけ生じ，SCore によって自身の再起動という形で処理されたネットワークの例外事象を，この手法により特定することができた．

一方，故障が明確には顕在化しない，あるいは最終的に顕在化しても発生から顕在化までのプロセスが極めて長い場合には，以下の手法により故障の検出や同定を行う．まず関数の実行時間を呼出パスごとに集計し，解析対象のトレース全体の時間に対する比率を要素とするベクトルを生成し，これをノードごとの実行挙動モデルとする．次に各ベクトルについて他の全てのベクトルとの距離を算出し，その中で $k$ 番目に近いものの値を，そのノードの「容疑度」とする．たとえば図 26 の例で各々の黒丸が書くノードの挙動ベクトルとし，$k = 2$ とすると，ノード $X$, $Y$, $Z$ の容疑度（$s(X)$など）は全てノード $A$ との距離となる．ここで破線円の領域内の挙動は正常，外部の挙動は異常とすると，容疑度が他と比較して大きいノード $Y$ と $Z$ を不審なノードと判定できる．また正常な実行でのトレースが利用可能である場合，それから生成されるベクトルの中で最も近いもの，すなわち図では $Y$ に対する $N$ の距離が小さければ，容疑度をその値に置換する．このことによって，正常な実行でも他のノードと違う挙動を示すノード（たとえばマスターワーカ型のマスター）を誤って異常と判定することが避けられる．

この手法の有効性は，２ヶ月の実験中に一度だけ生じた，SCore のブロードキャスト関数に関係する異常の原因追及で確認された．すなわちこの異常では，前述のトレース末尾の時刻では特に不審なノードは発見できなかったが，挙動モデルによってある特定ノードの容疑度が他のノードの２０倍以上あったことで，当該ノードで異常が生じていることを

発見できた．また挙動ベクトルの各要素は関数とその呼出パスに対応しているので，大きな容疑度の要因となっている関数とそこへのパスが特定できた．さらにそれを起点に調査した結果，極めて例外的な事象が生じた場合の SCore の対応に問題があることが明らかになり，バグフィックスに極めて有用な情報を提供することができた．

④　成果の位置付け・類似研究との比較

耐故障機能のベースである Cuckoo に類似する実装としては，FT-MPI, LAM/MPI, Open MPI などがある．FT-MPI は Cuckoo と同様に環境に応じた復旧プロトコルの定義など，豊富な耐故障機能を実現するための機能を有しているが，機能のコンポーネント化が成されておらずユーザレベルでの実装が非常に困難である．一方 LAM/MPI や Open MPI はコンポーネント化の面では優れているが, Coordinated Checkpointing のみのサポートにとどまるなど，耐故障機能の実装は貧弱である．Cuckoo は柔軟かつ豊富な耐故障機能をコンポーネント化して実現する枠組みであり，これらの類似研究の長所を併せ持ったシステムである．またプロセス移送もいくつか実装されているが，いずれも特定のミドルウェアを前提としたものであり，②で述べた手法のようにプラットフォーム独立な実装はほとんど例がない．

モデルベースの故障検出については，大規模クラスタを対象とした前例が皆無に近く，関連技術は単一システム，ビジネスサーバ，一般的 client-server 型のシステムなどを対象としている．また，トレースの対象やその解析方法も③で述べた方法とは異なり，特に不具合の原因追及能力に劣るものがほとんどである．

(2)　研究成果の今後期待される効果

システムの大規模化や複雑化に伴い，その内部で故障や不具合が生じる確率は確実に増加している．このような状況において，システムの信頼性確保の大部分をハードウェアに委ねることは，いたずらにコスト増を招くのみであり得策とは言えない．したがって本研究の成果のようにソフトウェアによってハードウェア故障を補う耐故障技術は，今後更に必要性を増すことは疑いない．

一方, ソフトウェアによる耐故障性の確保は万能ではなく, (1) ③の例でも述べたように，故障をハンドリングするソフトウェア機能は，そのテストが困難であることやバグ発現が稀少であることなどから，品質確保が極めて難しくフィールドでのバグ特定や対処も難渋するのが通例である．本研究で考案したモデルベースの故障検出・特定技術は，このようなハードウェア・ソフトウェアの複合的トラブルの解析に非常に適しており，実際に適用する例が増える見通しも得られている．また現状ではオフラインの解析であるが，リアルタイムでの解析も原理的には可能であり，システム停止などの致命的状況に至る以前に故障や不具合を回避する応用も期待できる．

### 3.5. メガスケールのタスク並列プログラミングとその実行技術（豊橋技術科学大学／京都大学 プログラミング技術グループ）

(1) 研究実施内容及び成果

メガスケールの並列性を持つプログラムを一から記述するのは不可能に近く，一定の階層化が必須である．そこで我々は，SPMD などの既存のパラダイムに基づいて設計された $10^3$ スケール程度までの並列プログラムを１タスクとし，それをさらに $10^3$ 以上のスケールでタスク並列実行する，２階層の多重並列実行のパラダイムが有効であると考えた．またこのパラダイムでは，上位階層のプログラミング，すなわちタスク並列プログラムを以下に簡便に記述するかと，大規模・粗粒度の並列タスクの実行管理をいかに効率的に行うかが重要である．そこでこの両者を同時に解決するためのプログラミング言語として，タスク並列スクリプト言語 MegaScript を設計し，言語処理系をはじめとする種々のソフトウェア技術について研究することとした．

MegaScript の特徴は，多数の並列タスクからなる上位階層並列プログラムを簡便に記述できる「オブジェクトレベル」の記述能力と，効率的な実行管理のための情報提供手段として個々のタスクの挙動を抽象的かつプログラムの形式で与える「メタレベル」の記述能力の双方を，一つの言語により統一的に提供することにある．この特徴を最大限に生かし，大規模な並列プログラムを容易に記述し，かつそれを効率的に実行するメカニズムを提供することが，本研究の目的であった．

MegaScript では，タスクの特性を予測するための重要な手段として，スクリプトプログラム中に記述されたタスク情報であるメタプログラムを利用する．メタプログラムは，タスクを抽象化したプログラムで表現したものである．スケジューラは，この情報をもとにスケジューリング戦略を定め，さらにタスクの実行時間やシステムの状態など動的な情報をもとに実行中にもスケジュールの調整を行う．MegaScript は各タスク間の関係のみを記述するため，コードの処理コストは小さい．このためプログラムの記述性を優先しスクリプト言語を採用した．MegaScript はベース言語に Ruby を用い，Ruby を拡張する形で設計・実装している．

① 性能モデル記述とモデル精度

MegaScript では図 27 に示すような形式で，外部プログラムである逐次あるいは並列のタスクの基本的情報と，その性能モデルを記述する．このモデル記述は「メタプログラム」と呼ばれ，その名の通り具体的あるいは仮想的な実行が可能なスクリプトプログラムである．このようにプログラムの形でモデルを記述する提案はほと

```
class Pi < Task
  def initialize(arg)
    @exefile = './pi'
    @args = arg
  end
  def behavior
    n = @args[0]
    FOR n
      compute(2)
    end
  output()
  end
end
```

図 27：MegaScript によるタスク記述

んど例を見ないが，モデルの用途やプログラマの知識に応じて概略的にも詳細にも記述できるという優れた特徴を持つ．

メタプログラムの記述は，実行時に定まる引数などにより instantiate した性能モデルに変換されるが，その際にハードウェア構成や通信ライブラリなどタスクの実行環境に応じた調整が行われる．この結果，図 29 に示すように 3 種類の異なる計算環境のいずれについても高精度のモデリングが実現できている．特に，相対的に通信性能が低い Xeon クラスタでの 2 ノード性能の劣化や，階層ネットワークであるため上位層の通信がボトルネックとなる Pentium-III クラスタの 16 ノード性能の飽和傾向など，環境に特有の性能特性が忠実に再現できている．

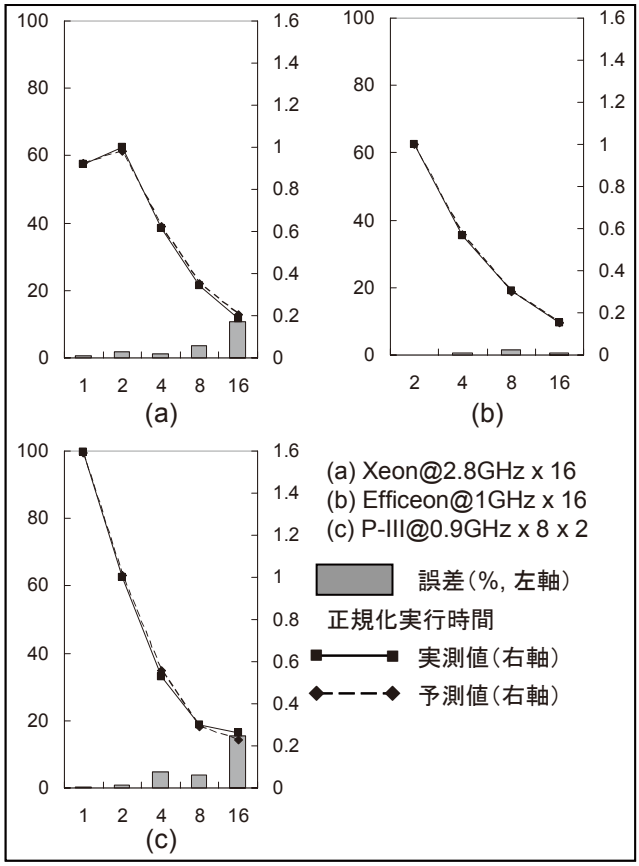



図 29：FT のモデル精度

② 処理系のユーザレベル拡張

MegaScript の処理系は，外部プログラムであるタスクをオブジェクトとして管理し，タスク間通信を含む実行制御や並列計算環境へのタスク割当を行う．これらの基本機能は C あるいは Ruby で実装されているが，それをユーザレベルで拡張する「アダプタ」と呼ぶ機構も実装されている．

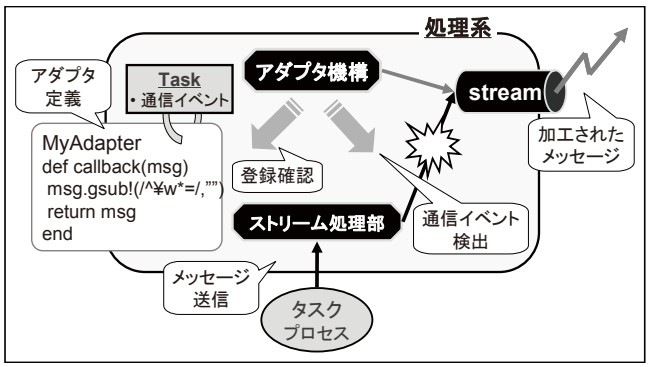



図 28：通信フィルタ用アダプタ

アダプタは図 28 に示すような Ruby のユーザクラスとして定義され，タスクやストリーム通信路を表現するオブジェクトに動的な登録，すなわち plug-in を行うことができる．この plug-in は処理系の内部で生じる様々なイベントに応じて行われ，たとえば図ではタスク間のストリーム通信イベントにアダプタが plug-in されている．処理系中のアダプタ機構は，各イベントの発生時に plug-in の有無をチェックし，plug-in されていればイベントに応じて定まる引数を設定してアダプタのコールバック関数を起動し，その結果を処理系内のイベント処理機構が受け取るべきデータとして渡す．

このようにアダプタは非常に柔軟な拡張機構であるが，そのオーバヘッドは極めて微小である．たとえば図の例のようなタスク間通信のフィルタリング処理を，遺伝子の相同性解析を行う１～３２ノードの並列タスクに対して行った場合，フィルタをタスク内部で行う場合に対して 1%以下のオーバヘッドで実現することができている．

③　成果の位置付け・類似研究との比較

スクリプト言語によって大規模な並列処理を記述する試みには Condor などがあるが，MegaScript のようにタスクの挙動を記述する枠組みを併せ持った言語はなく，その点で極めてユニークな存在である．またアプリケーションの挙動記述については，多くの研究が計算・通信コストをパラメータ化された数式で表現する方法を取っており，MegaScript のアプローチであるプログラム形式に比べて柔軟性・表現力の面で大きく劣っている．この点に関する例外的な試みとして ParaPhrase があるが，これは主としてソースプログラムからの性能情報抽出を目的としたものであり，ユーザによるモデル記述は可能であるものの，煩雑でありかつ可読性も乏しい．

また MegaScript の処理系自体が，ブラックボックスである並列タスクを効率的に扱う枠組みとしてユニークであるが，そこにユーザレベルで機能拡張できるアダプタを導入したことで，さらに特色ある処理系となっている．なお，一定の類似性を持つ並列タスクの制御フレームワークとして，UNICORE などワークフローに基づくものがあるが，これらにもアダプタのような機能を導入すれば，制御性が格段に向上するものと思われる．

(2)　研究成果の今後期待される効果

本研究プロジェクト全体を通じた重要な柱として，ワークロードの挙動・性能のモデリングを取り上げて研究したが，当初の想定応用であった耐故障性やタスクスケジューリングだけではなく，低電力化技術やネットワーク技術などでもモデリングの必要性・重要性が明らかになった．現状の MegaScript は，制御対象である並列タスクの挙動モデルを記述する枠組みとしているが，モデル記述だけを切り離して利用することも容易であり，スクリプト言語プログラム形式による簡便かつ強力な記述方式は，今後さまざまな応用を見出すことが期待される．

また並列システムの大規模化やマルチコア技術による階層深化に伴い，並列アプリケーションを from scratch で構築することが著しく困難になりつつある．実際多くの応用分野では，いわゆるアンサンブル実行のように一種の embarrassment parallelism に基づく多階層並列処理が日常的に行われるようになってきており，これらを簡便に記述し，柔軟かつ効率的に制御する枠組みとしても，MegaScript が果たす役割はさらに広がるものと期待される．

### 3.6. 高性能・低電力クラスタ MegaProto

(1) 研究実施内容及び成果

前述のように我々は，コモディティ技術をベースとして低電力プロセッサを高密度に実装するアプローチこそが，メガスケール計算を実現する唯一の方法であると主張している．この主張を裏付けるひとつの方法は，現時点で利用可能な技術を用いて高密度・低消費電力・高性能のシステムを構築し，その延長線上に我々が目指すメガスケール計算システムが存在することを実証することである．

そこで我々は，多数の低電力プロセッサを高密度に実装し，それらを高信頼・高バンド幅のネットワークで結合したプロトタイプシステム MegaProto を開発した．また MegaProto は，プロジェクトで研究・開発中の様々な技術の実証プラットフォームとしても利用された．

① システム構成

MegaProto は, 図 30 に示すように, 低電力プロセッサを中核とするプロセッサカード, それを 16 個搭載し 3.3 節で述べた RI2N の技術による 2 系統のギガビットイーサネットで結合した 1U サイズのクラスタユニット, それらを VFREC-Net の技術で結合したキャビネットシステム, さらにそれらを多数結合したマルチキャビネットシステムという, 階層的な構成となっている．

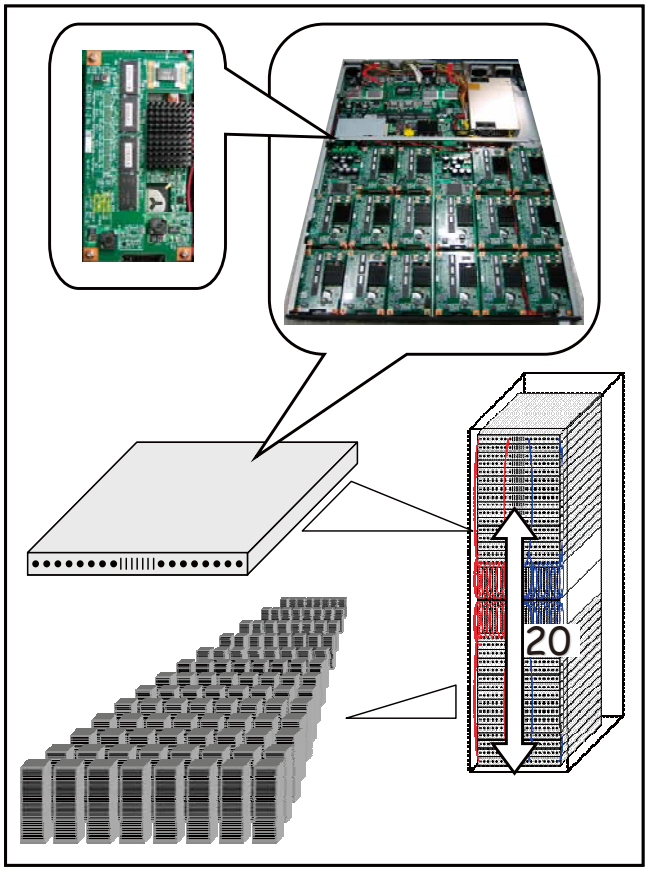



図 30：MegaProto のシステム構成

MegaProto には 2 つのバージョンがあり，2004 年に完成した MegaProto/C のプロセッサカードは, 933 MFLOPS の Crusoe プロセッサと，256MB のメモリ(SDR-133)および PCI バス(32bit, 33MHz)などから構成されている．一方 2005 年に完成した Mega- Proto/E は, 2GFLOPS の Efficeon プロセッサを採用して演算性能を向上させただけでなく，512MB の DDR-266 メモリの採用による容量・性能の増強，および PCI-X バス(64bit, 66MHz)の採用によるネットワーク I/O 性能の増強を行い，演算・メモリ・通信の 3 要素が性能的によくバランスした構成となっている．またいずれのバージョンでもプロセッサカードの消費電力は約 10W と極めて小さい．

一方，プロセッサカードを搭載するクラスタユニットのマザーボードは２つのバージョンに共通であり，図 31 に示す構成となっている．すなわち 16 個のプロセッサカードはそれぞれ２本のギガビットイーサネットリンクを持ち，２つの 24 ポートスイッチによって２系統の RI2N が構成されている．また，系統あたり８本のアップリンクによって，他のクラスタユニットとは RI2N /VFREC- Net の複合技術で接続される．クラスタユニットの消費電力は MegaProto/C が 300W, MegaProto/E が 320W であり，後者では 100MFLOPS/W という高い性能電力費を達成している．

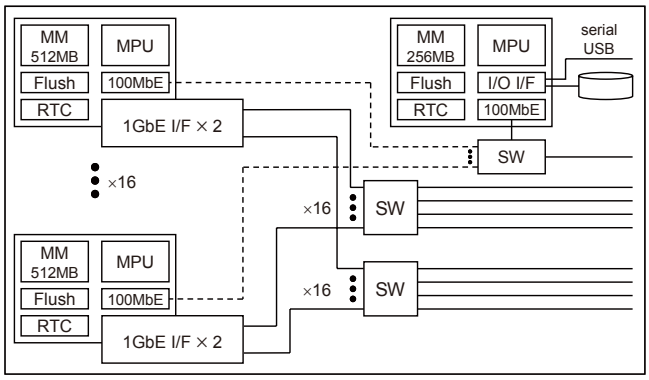



図 31：MegaProto のクラスタユニット

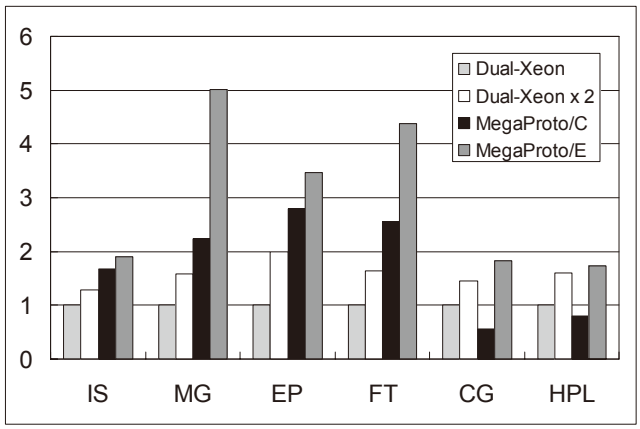



図 32：MegaProto クラスタユニットの性能

MegaProto/C は２ユニット製作され，基本的な性能評価，特に消費電力の動的な変化に関する詳細な評価に主として利用し，DVS による低電力化の効果と限界，ネットワーク素子の電力消費の問題など，低電力・高性能計算に関する有益な知見を多数もたらした．一方，本格的なシステムである MegaProto/E は 20 ユニット製作され，各サイトでの研究に活用されている．また電力最適化コンパイラ，RI2N, VFREC-Net, MegaScript 処理系など，主要な成果ソフトウェアも実装されている．

② 性能評価

図 32 は，MegaProto/C および Mega- Proto/E のクラスタユニット（16 プロセッサ）の性能を，同じ 1U サイズで消費電力が約 400W の Dual-Xeon サーバと，それを２台結合したミニクラスタの性能と対比して示したものである．まず MegaProto/C は，IS, MG, EP, FT の４プログラムについて，サイズと消費電力が同等である Dual-Xeon サーバを大きく凌駕する 1.7～2.8 倍の性能を達成しただけでなく，サイズ・消費電力が２倍の Dual-Xeon ミニクラスタすら凌駕するという，優れた性能を示している．しかし，通信が頻繁に生じる CG ではネットワークを駆動する I/O バスの性能が，また HPL ではメモリ容量とバンド幅が，それぞれボトルネックとなり，Dual-Xeon サーバよりも若干劣る性能となった．

一方 MegaProto/E は MegaProto/C の性能をさらに大きく上回り，MG での対 Dual- Xeon サーバ比は５倍を超えている．また前述のようにメモリ容量・性能および I/O バスの性能を増強した結果，MegaProto/C では比較的低かった CG と HPL の性能が２～３倍向上し，

Dual-Xeon サーバだけでなく，Dual-Xeon ミニクラスタをも上回る性能となった．これらの評価結果より，少数の高性能・高電力プロセッサによる構成よりも，多数の低電力プロセッサを用いて多ノードのクラスタを構成することが，高性能計算に適した方式であるという，我々の主張が完全に裏付けられた．

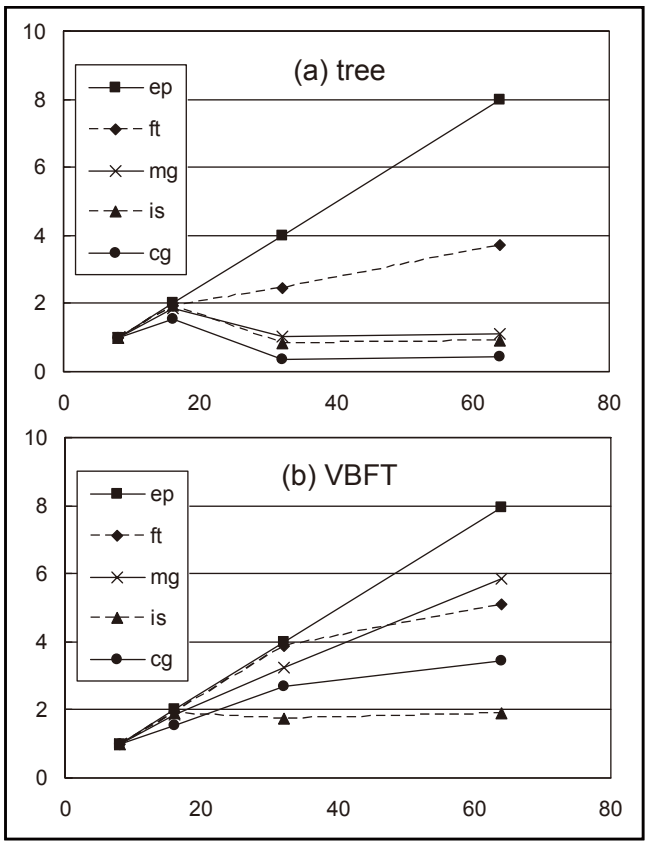


**図 33：MegaProto マルチクラスタユニットの性能**

次に，図 33 にクラスタユニットを４台まで結合したときの性能を，８プロセッサ（1/2 クラスタユニット）の性能を１として正規化して示す．図の(a)は，通常の多分木結合の構成のものであり，通信がほとんどない EP 以外では，複数クラスタユニット構成（32 プロセッサ以上）の性能が伸びず，MG, IS, CG では劣化さえしている．一方，図の(b)は VFREC-Net の技術を用いた fat-tree (VLAN-Based Fat-Tree: VBFT)構成の性能であり，系統あたり８本のアップリンクが活用された結果，性能が大幅に改善している．特に通信性能が問題となる CG では，64 プロセッサ性能が Tree 構成の８倍以上となり，MegaProto のような多ノードシステムでのネットワーク性能の重要性と，それに十分配慮して多数のアップリンクを持たせた設計の妥当性が実証された．

③　成果の位置付け・類似研究との比較

高性能計算を目的として設計・構築されたクラスタのほとんどは，ハイエンドのマイクロプロセッサを用いたものであり，消費電力や実装密度の観点で MegaProto とは全く異なった方向の設計方針に基づいている．またビジネス応用をターゲットするクラスタの中には，モバイルプロセッサを利用して低消費電力を謳ったものがいくつかあるが，プロセッサあたりの消費電力は 20W 以上であり，MegaProto が 10W 以下であるのに比べて大きな差がある．

例外的に低電力・高性能のシステム構築を目指した研究としては，MegaProto と同様に Crusoe を用いた LANL の Green Destiny があるが，実装密度や性能の面で MegaProto とは大きな差がある．また IBM の BlueGene/L は極めて優れた性能電力比を持ち，絶対性能の面でも現時点で世界最高速の記録を有している．しかし，BlueGene/L はそのベースとなるプロセッサやネットワークに専用のチップを用いており，コモディティ技術をベースと

する MegaProto に比べて開発・製造コストの面で大きく劣っている．

(2) 研究成果の今後期待される効果

コモディティ技術のみを用いて，高性能かつ低電力の大規模クラスタを構築できたことは，高性能計算システムのアーキテクチャに対して大きなインパクトを与えた．また最近の傾向として，個々のプロセッサの動作周波数と消費電力を抑えたコモディティマルチコアプロセッサが高性能計算システムで多用されているが，MegaProto はこの一歩二歩先を進んだシステムとして Supercomputing などでも注目を集めた．

今後の展開として MegaProto の成果が，より低電力のプロセッサ，たとえば組み込みシステムで用いられているプロセッサをコアとする低電力マルチコアを building block とした，1GFlops/W 級の高性能・低電力システムに繋がることが期待される．またメモリやネットワークの低電力化が急務であることも MegaProto の開発・評価を通じて明らかになっており，この方向での技術進歩も強く期待される．

## 4. 研究参加者

① プロセッサグループ（ソフトウェアとの協調最適化に基づく超低消費電力技術・高密度実装技術・高バンド幅技術）

| 氏名 | 所属 | 役職 | 研究項目 | 参加時期 |
|---|---|---|---|---|
| 中村　宏 | 東京大学 | 助教授 | 【研究項目１】 | H13.12～H18.11 |
| 今井　雅 | 東京大学 | 助手 | 【研究項目１】 | H13.12～H18.11 |
| 齋藤　寛 | 東京大学 | 特任助手 | 【研究項目１】 | H15.4～H16.3 |
| 近藤　正章 | 東京大学 | 特任助手 | 【研究項目１】 | H14.4～H18.11 |
| 藤田　元信 | 東京大学 | 博士課程 | 【研究項目１】 | H15.4～H18.3 |
| 倉田　憲一 | 東京大学 | 博士課程 | 【研究項目１】 | H16.4～H17.9 |
| 佐々木　広 | 東京大学 | 博士課程２年 | 【研究項目１】 | H17.4～H18.11 |
| 渡辺　亮 | 東京大学 | 博士課程１年 | 【研究項目１】 | H18.4～H18.11 |

② コンパイラグループ（ハードウェアとの協調最適化に基づき低消費電力かつ高性能を実現するコンパイラ技術）

| 氏名 | 所属 | 役職 | 研究項目 | 参加時期 |
|---|---|---|---|---|
| 佐藤　三久 | 筑波大学 | 教授 | 【研究項目２】 | H13.12～H18.11 |
| 前田　敦司 | 筑波大学 | 助教授 | 【研究項目２】 | H14.4～H18.11 |
| 高橋　睦史 | 筑波大学 | 研究員 | 【研究項目２】 | H16.4～H18.11 |
| 堀田　義彦 | 筑波大学 | 博士課程２年 | 【研究項目２】 | H17.4～H18.11 |
| 中島　佳宏 | 筑波大学 | 博士課程２年 | 【研究項目２】 | H17.4～H18.11 |

③ ネットワークグループ（安価かつスケーラブルなディペンダブル高速ネットワーク技術）

| 氏名 | 所属 | 役職 | 研究項目 | 参加時期 |
|---|---|---|---|---|
| 朴　泰祐 | 筑波大学 | 教授 | 【研究項目３】 | H13.12～H18.11 |
| 高橋　大介 | 筑波大学 | 助教授 | 【研究項目３】 | H13.12～H18.11 |
| 三浦　信一 | 筑波大学 | 博士課程３年 | 【研究項目３】 | H16.4～H16.8<br>H18.2～H18.11 |

④ クラスタ管理技術グループ（実行モデル構築とモデルを利用した大規模コモディティクラスタ管理技術）

| 氏名 | 所属 | 役職 | 研究項目 | 参加時期 |
|---|---|---|---|---|
| 松岡　聡 | 東京工業大学 | 教授 | 【研究項目４】 | H13.12～H18.11 |
| 中田　秀基 | 東京工業大学 | 助教授 | 【研究項目４】 | H13.12～H18.11 |
| 小川　宏高 | 産業技術総合研究所 | 研究員 | 【研究項目４】 | H13.12～H16.3 |
| 田中　康司 | 東京工業大学 | CREST 研究員 | 【研究項目４】 | H14.4～H15.3 |
| 栄　純明 | 東京工業大学 | 研究員 | 【研究項目４】 | H14.4～H18.11 |
| 丸山　直也 | 東京工業大学 | 博士課程３年 | 【研究項目４】 | H15.4～H18.11 |
| 吉村　優子 | 東京工業大学 | 研究補助員 | 【研究項目４】 | H16.4～H16.9 |
| 池田　理恵 | 東京工業大学 | 研究補助員 | 【研究項目４】 | H16.12～H17.1 |
| 實本　英之 | 東京工業大学 | 博士課程２年 | 【研究項目４】 | H17.4～H18.11 |

⑤ プログラミング技術グループ（メガスケールのタスク並列プログラミングとその実行技術）

| 氏名 | 所属 | 役職 | 研究項目 | 参加時期 |
|---|---|---|---|---|
| 中島　浩 | 京都大学 | 教授 | 【研究項目５】 | H13.12～H18.11 |
| 大野　和彦 | 三重大学 | 講師 | 【研究項目５】 | H13.12～H18.11 |
| 丸山真佐夫 | 木更津工業高専 | 助教授 | 【研究項目５】 | H14.4～H18.11 |
| 津邑　公暁 | 名古屋工業大学 | 助教授 | 【研究項目５】 | H16.4～H18.11 |
| 米倉ひとみ | 豊橋技術科学大学 | 研究補助員 | | H14.4～H18.11 |

## 5. 招聘した研究者等

（該当なし）

## 6. 成果発表等

(1) 原著論文発表　（国内誌15件, 国際誌1件）

[1] 近藤正章，中村宏，朴泰祐．SCIMA における性能最適化手法の検討．情処論 HPS, Vol. 42, No. SIG12 (HPS4), Nov. 2001.

[2] Masaaki Kondo and Hiroshi Nakamura. Reducing Memory System Energy by Software-Controlled On-Chip Memory. *IEICE Trans*, Vol. E86-C, No. 4, pp. 550-588, Apr. 2003.

[3] 高橋睦史，近藤正章，朴泰祐，高橋大介，中村宏，佐藤三久．　HPC 向けオンチッ

プメモリプロセッサアーキテクチャ SCIMA の SMP 化の検討と性能評価．情処論 ACS, Vol. 44, No. SIG6 (ACS1), pp. 76-86, May 2003.

[4] 藤田元信，近藤正章，中村宏．ソフトウェア制御オンチップメモリ向け自動最適化コンパイラの提案．情処論 ACS, Vol. 44, No. SIG1(ACS4), pp. 77-87, Jan. 2004.

[5] 丸山真佐夫，山本繁弘，大野和彦，中島浩．巻き戻し実行をサポートする並列プログラムデバッガ．情処論 ACS, Vol. 45, No. SIG3 (ACS5), pp. 109-121, Mar. 2004.

[6] 堀田義彦，佐藤三久，朴泰祐，高橋大介，中島佳宏，高橋睦史，中村宏．プロセッサの消費電力測定と低消費電力プロセッサによるクラスタの検討. 情処論 ACS, Vol. 45, No. SIG11 (ACS7), pp. 207-218, Oct. 2004.

[7] 藤田元信，田中慎一，近藤正章，中村宏．ソフトウェア制御オンチップメモリにおけるスタティック消費電力削減手法．情処論 ACS, Vol. 45, No. SIG11 (ACS7), pp. 219-228, Oct. 2004.

[8] 中島浩，中村宏，佐藤三久，朴泰祐，松岡聡，高橋大介，堀田義彦．高性能計算のための低電力・高密度クラスタ MegaProto．情処論 ACS, Vol. 46, No. SIG12 (ACS11), pp. 46-61, Aug. 2005.

[9] 湯山紘史，津邑公暁，中島浩．タスク並列言語 MegaScript 向け高精度実行モデルの構築．情処論 ACS, Vol. 46, No. SIG12 (ACS11), pp. 181-193, Aug. 2005.

[10] 丸山真佐夫, 津邑公暁, 中島浩. データ再演法による並列プログラムデバッギング. 情処論 ACS, Vol. 46, No. SIG12 (ACS11), pp. 214-224, Aug. 2005.

[11] 三浦信一，岡本高幸，朴泰祐，佐藤三久，高橋大介．VFREC-Net: ドライバ制御による tagged-VLAN を用いた PC クラスタ向けマルチパスネットワーク．情処論 ACS, Vol. 47, No. SIG12 (ACS15), pp. 35-45, Sep. 2006.

[12] 池田佳路，近藤正章，中村宏．実効電力制御による高性能計算機クラスタ構成手法の提案．情処論 ACS, Vol. 47, No. SIG12 (ACS15), pp. 262-271, Sep. 2006.

[13] 堀田義彦，佐藤三久，木村英明，松岡聡，朴泰祐，高橋大介．PC クラスタにおける電力実行プロファイル情報を用いた DVS 制御による電力性能の最適化．情処論 ACS, Vol. 47, No. SIG12 (ACS15), pp. 272-284, Sep. 2006.

[14] 木村英明，佐藤三久，堀田義彦，朴泰祐，高橋大介．DVS 制御による負荷不均衡のある並列プログラムの電力量削減手法. 情処論 ACS, Vol. 47, No. SIG12 (ACS15), pp. 285-295, Sep. 2006.

[15] 阪口裕輔，大野和彦，佐々木敬泰，近藤利夫，中島 浩．タスク並列スクリプト言語処理系におけるユーザレベル機能拡張機構．情処論 ACS, Vol. 47, No. SIG12 (ACS15), pp. 296-307, Sep. 2006.

[16] 佐々木広，浅井雅司，池田佳路，近藤正章，中村宏．統計情報に基づくコンパイラ協調型実行時最適化手法．情処論 ACS, Vol. 47, No. ACS16.（採録決定）．

(2) その他の著作物(総説, 書籍などを記載してください. )

(該当なし)

(3) 学会発表(国際学会発表及び主要な国内学会発表)

① 招待講演　　(国内会議1件, 国際会議1件)

[1] 中島浩. メガスケール計算の構想と低電力化技術. 情処学会関西支部大会, Nov. 2002.

[2] Satoshi Matsuoka. Saving energy in high-performance computing -- means to an end or the driving force? In *HP-PAC 2006 (incl. IPDPS 2006)*, Apr. 2006.

② 口頭発表　　(国内会議17件, 国際会議21件)

[1] Hiroshi Nakamura, Masaaki Kondo, Taku Ohneda, Motonobu Fujita, Shigeru Chiba, Mitsuhisa Sato, and Taisuke Boku. Architecture and Compiler Co-Optimization for High Performance Computing. In *IWIA 2002*. IEEE Computer Society, 2002.

[2] Masaaki Kondo, Motonobu Fujita, and Hiroshi Nakamura. Software-Controlled On-Chip Memory for High-Performance and Low-Power Computing. In *HPCA-8*, Feb. 2002.

[3] Masaaki Kondo, Mitsugu Iwamoto, and Hiroshi Nakamura. Cache Line Impact on 3D PDE Solvers. In *ISHPC 2002*, pp. 301-309. LNCS 2327, May 2002.

[4] 近藤正章, 大根田拓, 田中慎一, 中村宏. ソフトウェア可制御オンチップメモリを用いた低消費電力化の検討. JSPP 2002, pp. 285-288, May 2002.

[5] 高宮安仁, 松岡聡. ユーザー透過な耐故障製を実現する MPI へ向けて. JSPP 2002, pp. 217-224, May 2002.

[6] 外崎由里子, 大野和彦, 中島浩. 並列スクリプト言語(Perl)+の実装と設計. JSPP 2002, pp. 241-244, May 2002.

[7] Masaaki Kondo, Shinichi Tanaka, Motonobu Fujita, and Hiroshi Nakamura. Reducing Memory System Energy in Data Intensive Computations by Software-Controlled On-Chip Memory. In *COLP 2002*, Sep. 2002.

[8] Yoshiaki Sakae, Satoshi Matsuoka, Mitsuhisa Sato, and Hiroshi Harada. Towards Dynamic Load Balancing Using Page Migration and Loop Re-partitioning on Omni/SCASH. In *EWOMP 2002*, Sep. 2002.

[9] Taku Ohneda, Masaaki Kondo, Masashi Imai, and Hiroshi Nakamura. Design and Evaluation of High Performance Microprocessor with Reconfigurable On-Chip Memory. In *APCCAS 2002*, pp. 211-216, Dec. 2002.

[10] 高橋睦史，近藤正章，朴泰祐，高橋大介，中村宏，佐藤三久．HPC 向けオンチップメモリプロセッサアーキテクチャ SCIMA の SMP 化の検討と性能評価．HPCS 2003, pp. 47-54, Jan. 2003.

[11] Yoshihiko Hotta, Mitsuhisa Sato, Taisuke Boku, Daisuke Takahashi, and Chikafumi Takahashi. Measurement and Characterization of Power Consumption of Microprocessors for Power-Aware Computing. In *COOL Chips VI*, Apr. 2003.

[12] Yoshiaki Sakae, Satoshi Matsuoka, Mitsuhisa Sato, and Hiroshi Harada. Preliminary Evaluation of Dynamic Load Balancing Using Loop Re- partitioning on Omni/SCASH. In *DSM 2003 (incl. CCGrid 2003)*, pp. 463-470, May 2003.

[13] 丸山真佐夫，山本繁弘，大野和彦，中島浩．巻き戻し実行をサポートする並列プログラムデバッガ． SACSIS 2003, pp. 65-72, May 2003.

[14] 大塚保紀，深野佑公，西里一史，大野和彦，中島浩．タスク並列スクリプト言語 MegaScript の構想． SACSIS 2003, pp. 73-76, May 2003.

[15] 栄純明，松岡聡，佐藤三久，原田浩．Omni/SCASH のループ再分割を用いた動的負荷分散拡張の実装と評価．SACSIS 2003, pp. 307-314, May 2003.

[16] Shinichi Miura, Taisuke Boku, Mitsuhisa Sato, and Daisuke Takahashi. RI2N--Interconnection Network System for Clusters with Wide-Bandwidth and Fault-Tolerancy Based on Multiple Links. In *ISHPC 2004*, pp. 342-351, Oct. 2003.

[17] Motonobu Fujita, Masaaki Kondo, and Hiroshi Nakamura. Data Movement Optimization for Software-Controlled On-Chip Memory. In *INTERACT-8*, Feb. 2004.

[18] Hideyuki Jitsumoto, Satoshi Matsuoka, Towards a Portable Fault Tolerant Component Framework for MPI, In *WS FT-MPI Impl.(at SIAM-PP04)*, Mar. 2004.

[19] Yoshihiko Hotta, et al. MegaProto: A Prototype of the Ultra Low-Power Mega-Scale System. In *COOL Chips VII*, Apr. 2004.

[20] 藤田元信，田中慎一，近藤正章，中村宏．ソフトウェア制御オンチップメモリにおけるスタティック消費電力削減手法．SACSIS 2004, pp. 3-10, May 2004.

[21] 堀田義彦，佐藤三久，朴泰祐，高橋大介，中島佳宏，高橋睦史，中村宏．プロセッサの消費電力測定と低消費電力プロセッサによるクラスタの検討．SACSIS 2004, pp. 3-10, May 2004.

[22] Kenichi Kurata, Vincent Breton, and Hiroshi Nakamura, Secret Sequence Comparison in Distributed Computing Environments by Interval Sampling, In *CIBCB 2004*, pp.198-205, Oct., 2004.

[23] Hiroshi Nakashima, Hiroshi Nakamura, Mitsuhisa Sato, Taisuke Boku, Satoshi

Matsuoka, Daisuke Takahashi, and Yoshihiko Hotta. Mega-Proto: A Low-Power and Compact Cluster for High-Performance Computing. In *HP-PAC 2006 (incl. IPDPS 2006),* Apr. 2005.

[24] 湯山紘史，津邑公暁，中島浩．タスク並列言語 MegaScript 向け高精度実行モデルの構築．先端的計算基盤システムシンポジウム SACSIS 2005, pp. 43-52, May 2005.

[25] 丸山真佐夫, 津邑公暁, 中島浩. データ再演法による並列プログラムデバッギング. 先端的計算基盤システムシンポジウム SACSIS 2005, pp. 61-70, May 2005.

[26] Hiroshi Nakashima, Hiroshi Nakamura, Mitsuhisa Sato, Taisuke Boku, Satoshi Matsuoka, Daisuke Takahashi, and Yoshihiko Hotta. MegaProto: 1 TFlops/10 kW Rack Is Feasible Even with Only Commodity Technology. In *SC|05*, Nov. 2005.

[27] Masao Maruyama, Tomoaki Tsumura, and Hiroshi Nakashima. Parallel Program Debugging Based on Data-replay. In *PDCS 2005*, pp. 151-156, Nov. 2005.

[28] Shinichi Miura, Takayuki Okamoto, Taisuke Boku, Mitsuhisa Sato, Daisuke Takahashi. Low-cost High-bandwidth Tree Network for PC Clusters based on Tagged-VLAN Technology. In *ISPAN 2005*, pp.84-91, Dec. 2006.

[29] Motonobu Fujita, Masaaki Kondo, and Hiroshi Nakamura. A Temperature Aware Compilation for Software-Controlled On-chip Memory Architecture. In *ODES-4*. Mar. 2006.

[30] Yoshihiko Hotta, Mitsuhisa Sato. Hideaki Kimura, Satoshi Matsuoka, Taisuke Boku, Daisuke Takahashi. Profile-based Optimization of Power-Performance by using Dynamic Voltage Scaling on a PC cluster. In *HP-PAC (incl. IPDPS 2006),* Apr. 2006.

[31] Taisuke Boku, Mitsuhisa Sato, Daisuke Takahashi, Hiroshi Nakashima, Hiroshi Nakamura, Satoshi Matsuoka, Yoshihiko Hotta. MegaProto/E: Power-Aware High-Performance Cluster with Commodity Technology. In *HP-PAC (incl. IPDPS 2006),* Apr. 2006.

[32] 佐々木広，浅井雅司，池田佳路，近藤正章，中村宏．統計情報に基づくコンパイラ協調型実行時最適化手法．SACSIS 2006, pp. 67-74, May 2006.

[33] 三浦信一，岡本高幸，朴泰祐，佐藤三久，高橋大介．VFREC-Net: ドライバ制御による tagged-VLAN を用いた PC クラスタ向けマルチパスネットワーク. SACSIS 2006, pp. 117-126, May 2006.

[34] 阪口裕輔，大野和彦，佐々木敬泰，近藤利夫，中島 浩．タスク並列スクリプト言語処理系におけるユーザレベル機能拡張機構．SACSIS 2006, pp. 447-456, May 2006.

[35] 木村英明，佐藤三久，堀田義彦，朴泰祐，高橋大介．DVS 制御による負荷不均衡のある並列プログラムの電力量削減手法．SACSIS 2006, pp. 477-486, May 2006.

[36] 池田佳路，近藤正章，中村宏．実効電力制御による高性能計算機クラスタ構成手法の提案．SACSIS 2006, pp. 487-494, May 2006.

[37] 立薗真樹，中田秀基，松岡聡．仮想計算機を用いたグリッド上での MPI 実行環境．SACSIS 2006, pp. 525-532, May 2006.

[38] Alexander V. Mirgorodskiy, Naoya Maruyama, Barton P. Miller. Problem Diagnosis in Large-Scale Computing Environments. In *SC|06*, Nov. 2006.

③ ポスター発表 （国内会議12件, 国際会議0件）

[1] 柴田俊介，鈴木雄大，大野和彦，中島浩．メガスケールシミュレータ Anastasia の設計と実装. SACSIS 2003, pp. 153-154, May 2003.

[2] 藤田元信，近藤正章，中村宏．ソフトウェア制御オンチップメモリ向け自動最適化コンパイラの検討. SACSIS 2003, pp. 173-174, May 2003.

[3] 實本英之，高宮安仁，松岡聡．自律的な通信回復を行う Fault Tolerant MPI の実装と評価. SACSIS 2003, pp. 199-200, May 2003.

[4] 堀田義彦，佐藤三久，朴泰祐，中島浩，中村宏，松岡聡，高橋大介，中島佳宏， 高橋睦史，近藤正章，藤田元信．超低消費電力メガスケールコンピューティングシステムのプロトタイプ：MegaProto. HPCS 2004, January 2004.

[5] 湯山紘史，大塚保紀，西里一史，大野和彦，中島浩．タスク並列スクリプト言語 MegaScript によるタスクモデルの記述法. SACSIS 2004, pp. 135-136, May 2004.

[6] 鈴木雄大，柴田俊介，大野和彦，中島浩．ガスケールシミュレータ Anastasia におけるイベントスケジューラの設計と実装. SACSIS 2004, pp. 137-138, May 2004.

[7] 山形育平，中田秀基，松岡聡．Speculative チェックポイントの設計と実装．SACSIS 2004, pp. 155-156, May 2004.

[8] 實本英之，松岡聡．ポータブルな耐故障性コンポーネントフレームワークを持つ MPI 実装に向けて. SACSIS 2005, pp. 228-229, May 2005.

[9] 山形育平，松岡聡，實本英之，中田秀基．チェックポイント時の diskI/O 負荷を軽減する投機チェックポイント. SACSIS 2005, pp. 233-234, May 2005.

[10] 立薗真樹，中田秀基，松岡聡．仮想計算機を用いたマイグレーション可能な MPI 実行環境. SACSIS 2005, pp. 235-236, May 2005.

[11] Naoya Maruyama, Alexander V. Mirgorodskiy, Barton P. Miller, Satoshi Matsuoka. Root Cause Analysis of Failures in Large-Scale Computing Environment. SACSIS 2006, pp. 262-263, May 2006.

[12] 岡本高幸，三浦信一，朴泰祐，佐藤三久，高橋大介．Ethernet マルチリンクによる PC クラスタ向け耐故障ネットワーク RI2N/UDP. SACSIS 2006, pp. 271-272,

May 2006.

(4) 特許出願

① 国内出願 (0 件)

② 海外出願 (0 件)

(5) 受賞等

① 受賞

[1] 中村宏．坂井記念特別賞．情報処理学会，2002.5.20.

[2] 高橋睦史，近藤正章，朴泰祐，高橋大介，中村宏，佐藤三久．HPCS2003 最優秀論文賞．情報処理学会．2003.1.21.

[3] 松岡聡．日本学術振興会賞．日本学術振興会，2006.3.9.

② 新聞報道

（該当なし）

③ その他（研究展示）

[1] MegaScale Computing Based on Low Power Technology and Workload Modeling. *HPC ASIA 2004*, July 2004.

[2] MegaScale Computing Based on Low Power Technology and Workload Modeling. *SC 2004*, Nov. 2004.

[3] MegaScale Computing Based on Low Power Technology and Workload Modeling. *SC|05*, Nov. 2005.

[4] MegaScale Computing Based on Low Power Technology and Workload Modeling. *SC|06*, Nov. 2006.

(6) その他特記事項

（該当なし）

## 7. 研究期間中の主な活動

(1) ワークショップ・シンポジウム等

| 年月日 | 名称 | 場所 | 参加人数 | 概要 |
|---|---|---|---|---|
| 15.1.15-16 | 研究チーム内WS | 豊橋技術科学大学 | 20 | 研究状況の報告と討議 |
| 15.8.7 | 研究チーム内WS | 松江テルサ | 30 | 研究状況の報告と討議 |
| 16.1.29-20 | 研究チーム内WS | 豊橋技術科学大学 | 30 | 研究状況の報告と討議 |
| 16.8.2-3 | 研究チーム内WS | 東京大学 | 35 | 研究状況の報告と討議 |
| 17.1.6-7 | 研究チーム内WS | 筑波大学 | 35 | 研究状況の報告と討議 |
| 17.8.1-2 | 研究チーム内WS | アーバンオフィス | 35 | 研究状況の報告と討議 |
| 18.1.16-17 | 研究チーム内WS | 東京工業大学 | 35 | 研究状況の報告と討議 |
| 18.6.1-2 | 研究チーム内WS | 東京工業大学 | 35 | 研究状況の報告と討議 |

## 8. 結び

本報告書で述べたように，5 つのサブテーマおよび MegaProto の開発の成果は，概ね当初計画で目標としたものが得られている．また研究途上で得られた知見に基づき，いくつもの新たな展開が生まれたことにも深く満足している．さらに，5 年前に設定した本研究のメインテーマである低電力，耐故障性，モデリング，コモディティ技術などが，現在の高性能計算におけるキー技術として精力的に研究開発されていることで，我々の着目店の正しさが立証されたものと考えている．

中でも特筆すべき成果は MegaProto 開発の成功であり，研究代表者として高く自己評価している．研究開始当時はプロトタイプ開発の必要性を疑問視する声もあったが，目に見える形の成果物は研究全体を広くアピールする格好のシンボルであり，研究チームの求心力を高める効果も大きいなど，適切な提案であったと自負し

SC2004 展示

SC|05 展示

ている．また高性能計算分野で代表的かつ最大の国際会議である Supercomputing (SC) に，2004 年から３年連続して研究展示を行い，本研究の成果を国際的にアピールできた原動力も，MegaProto の研究開発であった．特に展示会場で得られた国際的に著名な研究者からの賛辞は，研究メンバーを強く勇気づけるとともに，研究方針の妥当性や成果の優秀性を裏付けるものであった．

このように数多くの重要な成果が得られた最大の理由は，各グループリーダをはじめとする研究メンバーの貢献であり，研究代表者として満足しかつ感謝している．本プロジェクトの発足当時，４拠点大学にまたがる５つの研究グループが総合に連携して１つのプロジェクトとして機能するかどうかについて危惧する声もあったが，５年間の研究遂行の円滑さと連携・協力の緊密さは，研究代表者の予想を超えるレベルであった．実際，研究代表者自身も含め全てのグループリーダが過去に多拠点型の研究プロジェクトに参画した経験を持つが，本プロジェクトの研究遂行状況は過去に類を見ないレベルであるとの認識で一致している．また MegaProto の設計は，その構想から詳細仕様の策定に至るまで，全グループリーダを中心とした共同作業で実施したものであり，この点でも研究チームの緊密な協力体制が実証されている．また研究予算の約半分を MegaProto 開発に投入した結果，各グループで使用可能な予算は金額と自由度両面で強く制約されたにも関わらず，研究チーム全体が開発に貢献したことについて感謝している．

このように円滑かつ順調に多拠点型研究が遂行できたことは，第一に各グループリーダや研究メンバーの参加意識と責任感の強さに負うものであるが，年２回のワークショップを含む頻繁なミーティングや国内外での学会等の機会を利用して，議論の機会を数多く設けたことも寄与したのではないかと自負している．また，年２回のヒアリングをはじめとする頻繁なフォローアップはグループリーダには負担であったようだが，代表者として各グループの進行状況を把握する好機であり，かつメンバーの意識を常に高く保つ効果もあって，適切な研究管理であると認識し感謝している．研究費の執行などの研究支援については概ね満足しており，特に MegaProto の開発スケジュール前倒しに伴う開発予算の年度移行，MegaProto/E の製作トラブルの措置，Supercomputing の展示経費の執行等に関して，非常に柔軟かつ機動的な対応がなされたことに深く感謝する．ただしその一方で，研究期間の終了間際になって事務処理の硬直性が見られたことは、まことに残念でならない。本研究プロジェクトは，上述のように満足すべき成果を収めつつ終了するが，本研究の柱である低電力・高性能計算，耐故障技術，モデリング技術などの研究テーマは，当然今後とも引き続き追求すべきものである．実際，各サブグループリーダは；

- 革新的電源制御による次世代超低電力高性能システム LSI の研究（中村：JST/CREST）
- 省電力高信頼組込み並列プラットフォーム（佐藤，朴：JST/CREST）
- 情報爆発時代に向けた新しい IT 基盤技術の研究（松岡，中島：科研費特定領域研究）

といった，本研究プロジェクトの将来に繋がる重要性の証左ともいうべき大規模研究プロ

ジェクトについて，平成１８年度より主宰者あるいは主要メンバーとして参画し，本研究の成果を発射台としてさらなる飛躍をそれぞれ開始している．本研究プロジェクトを終了するにあたり，各メンバーがこれらを始めとする新プロジェクトにおいて，本プロジェクトを凌駕する優れた成果を生み出すことを願いつつ，かつそれを確信して結びとする．