資　料

ハイエンドコンピューティング技術に関する調査研究Ⅲ

平成 14 年 3 月

財団法人　日本情報処理開発協会
先 端 情 報 技 術 研 究 所

KEIRIN
この事業は、競輪の補助金を受けて実施したものです。

ハイエンドコンピューティング技術調査ワーキンググループ

主査　山口　喜教　　筑波大学
　　電子・情報工学系　教授
委員　天野　英晴　　慶應義塾大学
　　理工学部 情報工学科　教授
委員　笠原　博徳　　早稲田大学
　　理工学部 電気電子情報工学科　教授
委員　久門　耕一　　(株)富士通研究所
　　コンピュータシステム研究所　部長
委員　妹尾　義樹　　日本電気(株)
　　インターネットシステム研究所　研究マネージャー
委員　関口　智嗣　　産業技術総合研究所
　　情報処理研究部門　副研究部門長
委員　佐藤　裕幸　　三菱電機(株)
　　情報技術総合研究所　チームリーダー
委員　佐藤　真琴　　(株)日立製作所
　　システム開発研究所　主任研究員
委員　近山　隆　　東京大学
　　新領域創成科学研究科　教授
委員　中島　浩　　豊橋技術科学大学
　　情報工学系　教授
委員　福井　義成　　(株)東芝
　　IS センター　主査
委員　横川　三津夫　日本原子力研究所
　　地球シミュレータ開発特別チーム　副主任研究員

幹事　若杉　康仁　　(財)日本情報処理開発協会 先端情報技術研究所
　　技術調査部 主任研究員
幹事　宮田　哲治　　(財)日本情報処理開発協会 先端情報技術研究所
　　技術調査部 主任研究員

ハイエンドコンピューティング技術に関する調査研究Ⅲ

目次

第１章　まえがき

1.1　調査の活動方針

本報告書は、先端情報技術研究所(AITEC)内に設置された「ハイエンド・コンピューティング技術調査ワーキンググループ」、略して HECC（High End Computing and Communication）WG の議論に基づき、各委員の報告をまとめたものである。このワーキンググループは、先端的なコンピュータ技術の調査をより広範囲な視点で行うことを目的として設置されており、その委員は大学、メーカ等の若手研究者でアーキテクチャ、ソフトウェア、アプリケーションの各分野において実際に研究や開発に携わっている方々から構成されている。このワーキンググループでは、ハイエンドアーキテクチャ、ハイエンドハードウエアコンポーネンツ、アルゴリズム研究を含む基礎研究、ソフトウエアおよびハイエンドアプリケーションなどを対象として、調査や議論を行ってきた。

本年度は、HECC ワーキンググループとしては 3 年目の調査となる。一昨年度は、議論の手がかりとして、ハイエンド・コンピューティングの分野で先頭を走っている米国の研究開発計画を参考にして、議論をすすめた。そして、そのための具体的な材料として、米国連邦政府のコンピューティング・情報・通信委員会(CIC)がまとめた通称 Blue Book と呼ばれているドキュメントを参考資料とし、この中で指摘されている技術項目について、市場動向などを視野に入れて見たときに市場にインパクトを与えそうな分野やテーマについて、それらの技術的内容および成果の調査・評価と我が国の技術との格差などについて各委員を中心に議論し、各委員の HECC に関する見解をまとめた。昨年度の調査においては、HECC 領域の開発動向を重要な中心課題と認識しつつも、これのみにとらわれず対象を拡大し、コンテンツやユーザインタフェースの実現基盤としてのプラットフォーム技術全般を視野に入れて、調査・検討した。各分野の専門家の最新知識を集積して研究開発のリーディングエッジを浮かび上がらせるとともに、今後注力すべき技術分野の検討に必要な元データとしての利用を可能にするように努めた。

今年度の調査においては、昨年同様、各委員の専門およびその周辺分野において、今後、研究的に急速に発展したり、あるいは市場にインパクトを与えそうな分野やテーマの抽出と、それらの技術に関する、(米国をはじめとする)先端研究とわが国との格差の調査・評価を行うことを主眼にした。したがって、今後重要になると思われる領域にはどのようなものがあるかを抽出するという点に力点を置くことにした。たとえば、プラットフォーム技術研究の代表的なものは、計算性能の追求である。計算速度や記憶容量等の劇的な向上は、単に処理速度を速めるにとどまらず、コンテンツやユーザインタフェースに質的な変化をもたらし、情報技術の社会的・経済的な影響力が強い。しかし、それ以外にも新たなコンパイラ技術の展開や並列処理技術および Grid と呼ばれるグローバルコンピューティング技術など、近未来の基盤技術としてインパクトを与える可能性があるという意味で、

いずれも同様の重要性を備えており、次時代をなす情報技術の一次近似予測のためには、これらの動向を広く把握する必要がある。

今年度、ワーキンググループとしては5回の会合を持ったのみであるが、本ワーキンググループの各委員は、情報処理の分野において最先端の研究や開発に従事している方々であり、情報処理分野における重要と思われる分野について各委員の独自の判断で調査を行い、本報告書に記述をしていただいた。また専門的な分野のテーマによっては、WGの委員では把握しきれない部分があるため、これを外部の講師を招いてヒアリングを行うこととし、今年度は、4人の外部講師を招いた。理化学研究所、ゲノム科学総合研究センタの八尾徹氏には、「ゲノム・ポストゲノム時代におけるコンピュータ・情報技術へのニーズ」と題する講演を、またRedSwitch, Inc. のDr. Hungwen Li氏には「InfiniBand: Switch Fabric Solution for System Connectivity」という講演をお願いした。さらに、ネットワーク技術の飛躍的進歩に伴う新しい情報技術基盤として近年注目されているGrid技術に関して、産業技術総合研究所ハイエンド情報技術グループの田中良夫氏に「SC2001にみるGridの最新動向―Globus, Portal, SC Global」と題する講演をしていただいた。また、NTT未来ネット研究所の星合隆成氏には、ネットワークの大規模な普及に伴い、新しい考え方や技術として注目されているピア・ツー・ピア技術（Peer-to-Peer、略してP2P）に関して講演をお願いし、最新技術の紹介を含めてフリーなディスカッションを行うことができた。これらの講演を基に、田中良夫氏と星合隆成氏には講演の概要を、本報告書の中にまとめていただいた。ここで、あらためて感謝したい。

本報告書が、わが国のハイエンド・コンピューティングをはじめとする先端的な情報処理分野においてその技術開発や技術政策の一助になれば幸いである。

（山口　喜教　主査）

1.2　調査報告の概要

本節では、第 3 章に記載する各委員および外部講師による調査報告の概要を示す。

1.2.1　アーキテクチャ＆新計算モデル

(1) 半導体性能向上神話崩壊後のアーキテクチャ

天野　英晴　委員

CMOS LSI の動作速度はプロセスの進歩につれて向上を続け、新しいプロセスを利用することは、それだけで性能の向上につながった。これは、微細加工技術の発展と共に、電源電圧を落すことができれば、論理ゲートの遅延を減らすことができたことによる。ところが、2001 年、プロセスが 0.18um から 0.13um に進む際、電源電圧を低くすることがほぼ限界に達すると共に、LSI チップ内の遅延の支配的要因が、ゲート遅延から配線遅延に移行した。このため、銅配線などを用いない限り、新しいプロセスを単に利用するだけでは、性能は全く上がらなくなるに至った。半導体性能向上神話は崩壊し、今年から新しい時代に突入したのである。今後、CMOS LSI プロセス技術に画期的なブレークスルーがない限り、この傾向は続くと考えられる。

本レポートでは、半導体性能向上神話が崩壊した「新しい時代」を迎えて、アーキテクチャをいかに構成すべきかを提言する。まず、新しい時代に高い性能を実現するアーキテクチャについて述べ、ますます発展が予測される Reconfigurable Architecture についてまとめる。

(2) メガスケールコンピューティングの構想

中島　浩　委員

筆者らは科学技術振興事業団の戦略的基礎研究推進事業による研究プロジェクト「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」を実施している。このプロジェクトは百万プロセッサ級のメガスケールコンピューティングを真に実現するための基盤技術を追求するものであり、そのためにコモディティ技術、すなわち高性能計算のみをターゲットとしない一般性の高い技術をベースとした研究開発を行う。

プロジェクトでは、メガスケールコンピューティング実現の鍵は、(1) feasibility、(2) dependability、(3) programmability の３点にあるとし、それぞれ (1) ハード／ソフト協調による低電力高性能プロセッサ、(2) マルチポート・ネットワークとミドルウェアによる高信頼化、(3) グリッド／P2P 技術をベースとした高粒度大規模並列プログラミングの研究を行う。また、これらの技術を統合したプロトタイプとして、数千プロセッサ級の大規模低電力クラスタの構築も計画している。

(3) PC クラスタの現状と今後

久門　耕一　委員

現在 PC クラスタを取り巻く環境が激変しつつある。

１つは、PC クラスタを構成する要素部品である、CPU とネットワークの高性能化である。現在入手できる最高速の PC 向け CPU によれば、単一で 1GFlops を越える性能を得ることが出来る。更に、ネットワークの性能も CPU の外部バス性能と匹敵するスループットを達成している。つまり、ネットワークをこれ以上高速化しても、メモリのボトルネックによって律速されることになる。

一方、PC クラスタを利用する環境として従来は計算センタ内の１つの高性能システムとの位置づけであった。しかし、高速インターネットの発達により、地理的に分散している複数の計算センタを接続し全体として運用する、いわゆるグリッドコンピューティングを構成する要素システムの扱いを受けるようになってきた。

PC クラスタシステムを上の二つの観点から分析し、今後の動きを考える。

1.2 2 基本ソフトウェア＆ミドルウェア

(1) 手続き間解析の動向　～コンパイラとその周辺～

佐藤　真琴　委員

手続き間解析とは、手続き（関数、サブルーチン）の解析情報を、その手続きの呼び出し点で利用できるようにするコンパイル技術のことである。これによってプログラムの解析精度が向上し、プログラムを最適化する機会が増加することが期待できる。広い意味では、インライン展開も手続き間解析の一つと考えられるが、リカーシブな呼び出しに対応できないなど、適用範囲が狭い点が課題である。これに対して、狭義の手続き間解析はその一般性から、Fortran のみならず C++や Java に対する最適化コンパイラでも徐々に採用されつつあるなど、広がりをみせている。

本稿では、Fortran コンパイラにおける手続き間解析、及び、ツールにおける手続き間解析の利用に関する動向を、具体的な事例を通して紹介する。

(2) 自動並列化コンパイラによる SMP 上での粗粒度タスク並列処理

笠原　博徳　委員

ここでは、SMP 上における OSCAR マルチグレイン並列化コンパイラを用いた粗粒度タスク並列処理について述べる。現在、サーバーキテクチャの主流である SMP 上での自動並列化コンパイラを用いた並列処理では、ループレベル並列処理の性能が飽和状態に達しており、その限界を越えるため粗粒度タスク並列処理が注目されている。OSCAR FORTRAN コンパイラにおける粗粒度並列処理手法では、ソースプログラム中のサブルーチン・ループ・基本ブロック間の並列性を抽出し、各種 SMP 上で粗粒度タスク並列化

を実現するために、OpenMP を用いたワンタイムシングルレベルスレッド生成手法を用いている。さらに SMP で問題になる共有メモリアクセスオーバヘッドを軽減するため、複数タスク間での共有データの授受にキャッシュを最大限利用するデータローカライゼーション手法を併用することで、さらなる性能向上を図ることができる。ここでは、これらの技術を用いて、SMP サーバ IBM RS6000 SP 604e High Node、SMP ワークステーション SUN Ultra80 での粗粒度タスク並列処理を行った結果について述べる。

(3) P2P の理念および実現技術：SIONet の全貌

星合　隆成　講師

筆者は、1998 年にブローカレス型探索モデル（ブローカレスモデル）を提案して以来、その実現技術として「意味情報ネットワーク（SIONet Semantic Information-Oriented Network）」をこれまで提案してきた。そして、1998 年末に SIONet プロトタイプα版、1999 年末に SIONet プロトタイプβ版の試作を完了した。さらに、2000 年末に SIONet ver.1.0 の開発を完了した。

本稿では、ブローカレス型探索モデルを紹介することにより、「P2P の本質」について論ずる。さらに、ブローカレス型探索モデル（P2P モデル）の実現技術である SIONet について解説する。

1.2.3 応用システム＆応用分野

(1) SC2001 に見る Grid の最新動向

田中　良夫　講師

Grid とは、「高速ネットワークで接続された高性能計算機、大規模データベース、特殊な装置、人的資源などの様々な資源を柔軟に、容易に、安全に、統合的に、そして効果的に利用するためのネットワーク利用技術」である。本稿では、2001 年 11 月に米国コロラド州デンバーで開催された高性能計算および高性能ネットワークに関する国際会議である Supercomputing Conference(SC 2001)における講演、技術論文、企業展示および研究展示の内容をふまえ、Grid の現状および最新動向について報告する。本報告においては、(1)Grid における低レベルなソフトウェアの基盤として事実上の標準になっている Globus Toolkit、(2)ユーザに対して簡便なインタフェースを提供する Grid Portal システム、(3)高性能インターネット会議システムである Access Grid およびそれを利用して SC 2001 期間中に開催されたイベント SC Global、の 3 件に焦点をあてて Grid 技術および研究動向に関する報告を行う。今後各地で実際の Grid テストベッドの運営が進められると予想されるが、相互利用の可能性を高める意味でも Globus Toolkit がその基本ソフトウェアとして用いられ、ユーザに対してより上位なインタフェースを提供する高レベルミドルウェアや Web インタフェースを提供する Portal システムの研究開発が活発化すると予想され

る。また、Grid の本質である仮想的な組織(Virtual Organization)の運営においては実際に顔をあわせてのミーティングが重要になると思われ、Access Grid のようなシステムは有用であり、今後 Grid の技術を取り込んでより高機能、高性能なシステムの開発が望まれる。

(2) 地球シミュレータ開発を終えて

横川　三津夫　委員、妹尾　義樹　委員

平成 9 年度から開発を進めてきた地球シミュレータは、平成 14 年 2 月にピーク性能 40 テラフロップスの世界最高速を達成し完成した。そのハードウェアは、640 台の計算ノードをフルクロスバスイッチで結合したものであり、大規模な並列計算機システムを利用するための特徴あるプログラミングインターフェイスを具備している。

本稿では、地球シミュレータのハードウェア及びソフトウェアの概要、並列プログラミングインターフェイス、及びいくつかのシミュレーション結果について述べる。

(3) 高速ネットワーク環境下における高度医療アプリケーション

佐藤　裕幸　委員

がん治療法の一つである粒子線治療は、ヘリウム、炭素、ネオンなどの重粒子線を患部に集中して照射し、周辺の正常細胞への影響を最小限に抑え、患者の早期社会復帰を可能とする非常に有効ながん治療方法である。この粒子線を用いたがん照射治療を普及させるためには、遠隔地の複数の医療機関から高速なネットワークを介して、重粒子線がん照射施設の計算サーバにより、照射施設のノウハウを利用して患者毎に最適な照射方法を得る治療計画を立案できることが望ましい。このような高速なネットワークの利用法はまだ実現例に乏しく、基本的な性質の解明も進んでいない。インターネットの普及が進み、ネットワークを利用する研究者にも利便性を提供しているが、一方ではネットワークの輻輳が発生して、研究環境が悪化している例がある。またネットワークに不正に進入する事例が多数発生しており、医療関係の情報のようにプライバシーを守るべき対象を安全に保護する技術が必要不可欠となっている。このような状況から、インターネットにおける信頼性・高速性・利便性・安全性に関する種々の問題点を解決するために、これらについて極めて高度な要求を持つ遠隔地重粒子線がん照射影響シミュレータを業務として想定し、ギガビットレベルネットワークに向けた信頼性・高速性・利便性・安全性を確立することを目指して、高度医療ネットワークに関する研究が行われている。

1.3　その他の活動

本ワーキンググループでは、委員、外部講師による調査以外に、当研究所の若杉康仁主任研究員による米国のハイエンドコンピューティング研究開発に関する調査を行っている。主な参考資料としては、昨年公開された米国の Blue Book 2002 と、昨年 11 月米国コロラド州 Denver で開催された、ハイエンドコンピューティングに関する世界最大の国際会議である SC2001 に参加し入手した情報をベースとした。

調査結果は、第 2 章「米国のハイエンドコンピューティング研究開発動向」に記載している。

第２章　米国のハイエンドコンピューティング研究開発動向

2.1 概況

米国は2001年9月11日の同時多発テロを受けて一変した。この影響は米国のみに留まらず、世界中に大きな影響を与え現在に至っている。

続くアフガニスタンの空爆、アルカイダ政権打倒、暫定政府の成立と激動の世界史が動いている。この動きはハイエンドコンピューティングの研究開発のも影響を与えずにはおかないだろう。

2002 年 2 月ブッシュ大統領が FY2003 の予算教書を発表した。テロ対策のため、国防費が５年にわたって増加する見通しが発表されている。研究開発予算全体の傾向としてはバイオテロ対策もあり NIH がかなり突出して増加している。

大統領の技術的諮問機関である PITAC(President's Information Technology Advisory Committee)に関しては2003年の6月までの継続が決定している。今のところ共同議長およびメンバーの基本的な変更は無く、クリントン政権から NITRD 政策は継承されている。

経済面では、米国の景気はテロの影響を深刻に被っており、株価はダウ平均で 10,000 ドル近辺を、ナスダックは2,000 ドルを切り低迷している。

2001 年 11 月に米国デンバーで開催された第 14 回 SC2001 High Performance Networking and Computing コンファレンスでは、プラットフォーム分野はグローバルコンピューティングである Grid コンピューティングが、アプリケーション分野ではバイオインフォマティクス関連が主役にに躍り出ている。

ペタフロップスを追求していた、米国のハイエンドコンピューティング最近の動向はあくまで高性能を追求する動きから Grid に代表されるグローバルコンピューティングさらに民生用のハードウエアコンポーネントを利用するワークステーション／PC クラスタに向かっている。超並列、超分散がハイエンドコンピューティングの大きな流れとしてますます加速している。

片や日本でも、2001 年、2002 年は IT 業界の景気の不況は深刻であり、ハイエンドコンピュータを手がける大企業各社も経営難から一斉に早期退職制度を採用し、人員の整理、業種転換とう企業内構造改革に乗り出してた。これは従来日本では、大企業の研究機関を中心にハイエンドコンピュータの研究が行われてきたことを考えるとますます米国に遅れをとる懸念が強い。国研である産総研でも独立法人化が始まり中長期の研究開発が敬遠されるようになっているのも気がかりだ。

以下、本章ではハイエンドコンピューティングの新しい流れを中心に概観する。主な情報源としては2001年11月に開催された SC2001 コンファレンス、通称 Blue Book と

呼ばれている FY2002 大統領予算教書への科学技術に関する補足ドキュメント (FY2002 Blue Book)、及び Web で公開されている情報を元にしている。

FY2002 Blue Book に関しては、下記サイトに全文が載っている。（過去の分もある）

英語原本：http://www.itrd.gov/home.html

日本語訳：http://www.icot.or.jp/FTS/Ronbun/BlueBook2002-J.pdf

米国のハイエンドコンピューティング開発の背景、経緯は過去５期に渡る当研究所の報告書（ペタフロップスマシン技術に関する調査研究、同Ⅱ，同Ⅲおよびハイエンドコンピューティング技術に関する調査研究Ⅰ、同Ⅱ）にまとめられているのでこれを参照下さい。

報告書はいずれも AITEC のホームページ（ http://www.icot.or.jp/ ）から参照可能。

2.2 FY2002 Blue Book にみる政府支援の IT 研究開発

2.2.1 概要

ブッシュ政権初の FY2002 Blue Book は 2001 年 8 月に開示された。政権が民主党から共和党に交代しての最初であり内容に期待したが、ページ数も 50 ページと FY2001 の 1/4 の内容になって期待はずれ。 IT R&D 計画自体が民社党特に政敵であるゴア氏の功績が大きいので、あまり宣伝したくないというのも感じられた。基本的な枠組みは共和党になったからといって大きくは変わらず従来の枠組みを踏襲している。名前だけは先頭に Networking を付けて、NITRD (Networking and Information Technology and Development)に変更している。従来から 2001 年９月に下院を通過した NITRD 法案は、上院を通過していないので正式な法律にはなっていないが、Blue Book 内にも法案通過の記述があり政策はその流れを継承していると思われる。（長期に渡る IT 研究開発予算が認可されたのでは無いと言うことであろう。）

内容は具体的な研究内容の紹介が多かった FY2001 とは異なり、どのような目的でどのようなブレークスルーを目指すかといった一般国民受けのする概念の表現に変わっている。従来の省庁間をまたがる推進体制と６つの PCA(Program Component Area)自体は変化していない。

全体の推進体制を National Coordination Office(NCO)の発表プレゼンテーション資料“Federally-Funded Information Technology Research and Development”（ Jan 9 2001 発行）から見てみると図 2.1 の様になっている。

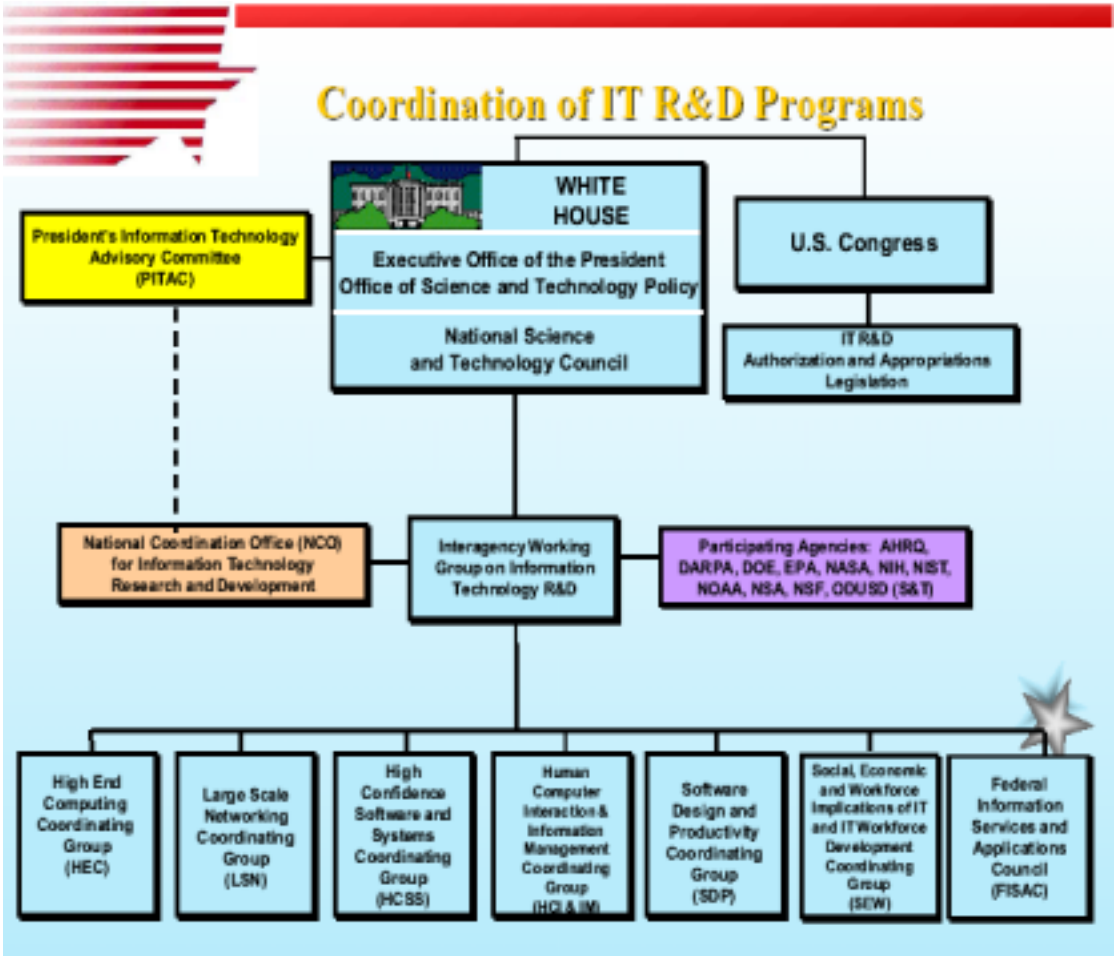


図 2.1　NITRD 推進体制

NITRD 計画の研究開発実体は、次の６つの PCA ので推進されており、以下のグループ（Coordination Group）がある。

1) **ハイエンドコンピューティング(HEC: High End Computing)**
   ここは更に２つに分かれている。（図 2.1 には記述はされていない）
   - **High End Computing Infrastructure and Application (HEC I&A)**
   - **High End Computing Research and Development (HEC R&D)**
2) 大規模ネットワーク(LSN: Large Scale Network)
3) 高信頼性ソフトウエアとシステム(HCSS: High Confidence Software and System)
4) ヒューマンコンピュータインターフェース及び情報管理（HCI&IM: Human Computer Interface and Information Management）
5) ソフトウエアの設計と生産性（SDP: Software Design and Productivity）
6) 社会、経済及び労働力の面から見た IT 労働力開発の意味(SEW: Social, Economic and Workforce Implication of IT and IT Workforce Development）

NITRD 計画への参加省庁は下記 12 省庁である。

1) National Science Foundation (NSF)
2) Defense Advanced Research Projects Agency (DARPA)
3) National Institutes of Health (NIH)
4) National Aeronautics and Space Administration (NASA)
5) Department of Energy Office of Science (DOE/OS)
6) National Security Agency (NSA)
7) National Institute of Standards and Technology (NIST)
8) National Oceanic and Atmospheric Administration (NOAA)
9) Agency for Health Research and Quality (AHRQ)
10) Office of the Deputy Under Secretary of Defense for Science and Technology(ODUSD (S&T))
11) Environmental Protection Agency (EPA)
12) Department of Energy National Nuclear Security Administration (DOE/NNSA)

基本的に昨年 FY2001 と同じであるが、ODUSD (S&T)は昨年は OSD/URI(Office of the Security of Defense's University Research Initiative)と記述されており、監督省庁名が変更された。

2.2.2 High End Computing 予算規模

既に NITRD FY2003 の予算要求は発表されているが、詳細の IT 研究開発予算の配分はこれからになる。よって FY2002 Blue Book で纏められている FY2002 の要求予算にて各省庁の予算規模を見てみる。表 2.1 に予算要求額を示す。

表 2.1　NITRD 計画　FY2002 年度　予算要求額

省庁	HECI&A	HECR&D	HCI&IM	LSN	SDP	HCSS	SEW	合計		比率
NSF	**249.7**	**65.1**	104.8	98	39.7	46.1	39.1	642.5		32.6%
DARPA	**55.5**	**42.7**	38.2	49.2	44.6	32.9		263.1		13.4%
NIH	**55.1**	**13.7**	74.6	81.1	6	10.1	11.4	252		12.8%
NASA	**36.1**	**26**	27.8	14.4	22.4	47.1	7	180.8		9.2%
DOE/OS	**98.3**	**31.5**	16.4	25.9			4	176.1		8.9%
NSA	**33.6**			1.9		46.6		82.1		4.2%
NIST	**3.5**	**6.2**	3.2	2	7.5			22.4		1.1%
NOAA	**13.3**	**1.8**	0.5	2.7	1.5			19.8		1.0%
AHRQ	**9.2**	**6.7**						15.9		
ODUSD(S&T)		**2**	2	4.2	1	1		10.2		0.5%
EPA	**1.8**	**1.8**								0.0%
小計	**513.3**	**216.4**	279.7	287.3	116	193	61.5	1,666.70		84.6%
DOE/NNSA	**133.8**	**37**		35.5	41.1		56.5	303.9		15.4%
合計	**647.1**	**253.4**	279.7	322.8	157	193	118	1,970.60		100.0%
比率	**32.8%**	**12.9%**	14.2%	16.4%	8.0%	9.8%	6.0%	100.0%		

NSF が最大の予算を使い、IT 関連の取りまとめ的な役割を果たしている。

FY2002 の HEC 関連である HEC I&及び HEC R&D の予算要求額は合計で 900.5M ドルであり、IT 関連予算の 50%弱を占める。米国がもっとも重視していることがうかがえる。

既に FY2003 の省庁毎の予算要求はでており、Office of Science and Technology Policy (OSTP)が発表する資料(http://www.ostp.gov/html/NetworkingI TR&D.pdf)によると NITRD 総額では FY2002 実績値の３%増加の$1,890M になっている。PCA 毎の分類はまだ纏まっていない。表 2.2 に予算概要を示す。

表 2.2　NITRD　FY2003 予算概要

	2002 ($M)	2003 ($M)	増加率
National Science Foundation	676	678	0%
Health and Human Services	310	336	8%
Energy	312	313	0%
Defense	320	306	-4%
NASA	181	213	18%
Commerce	43	42	-2%
Environmental Protection Agency	2	2	0%
TOTAL	**1,844**	**1,890**	**3%**

2.2.3 NITRD の研究テーマ

FY2002 Blue Book では中長期の研究課題を 10 個の研究のチャレンジ課題、および７つの国家的グランド・チャレンジ課題の２つのグループにカテゴリ分けして纏めている。

あまり専門技術的でない言葉で国民に語るように、書き方を変えている。

2.2.3.1 研究チャレンジ

中長期の研究課題が中心となっている。詳細項目は各 PCA で検討されているはずだがここでの表現は PCA の枠を消している。以下の 10 に分類されている。

（１）**次世代コンピューティングとデータ記憶技術**

HEC I&A が中心、ASCI を含む

（２）**シリコン CMOS における障壁の克服**

HEC R&D が中心。量子コンピュータ、DNA コンピュータ、光テクノロジ

（３）**２１世紀のための多目的、安全、拡張可能なネットワーク**

LSN 中心でスケーラブルが強調されている。SII(Scalable Information Infrastructure)

（４）**科学と工学における米国の強みを維持するための先進的 IT**

HECC,LSN 含む。モデリング、シミュレーション等ハイエンドアプリケーション

（５）**重要なシステムにおける信頼性と安全な運用の確保**

HCSS 中心。高信頼性とセキュリティ

（６）**現実世界のためのソフトウェア作成**

SDP 中心。高品質なソフトウエア設計と生産性

（７）人間の能力と万人の進歩に対する支援

HCI 中心。ユニバーサルアクセス、可視化、口語による対話等

（８）**知識世界の管理と実現**

IM 中心。ディジタルライブラリ、データマイニング、検索エージェント等

（９）世界レベルの IT 要員の教育と育成の支援

SEW 中心。IT リテラシー、指導者の訓練等

（10）利益を最大化するための IT の効果についての理解

SEW, FISAC(米連邦政府情報サービス・アプリケーション協議会)関連と思われる。

なお、わずかながら各チャレンジの省庁毎の代表的研究課題は紹介されている。これには必要以上に詳細研究内容を発表しないと言う国防上の気遣いも感じられる。

2.2.3.2 国家的グランド・チャレンジ・アプリケーション

国民に直接利益のある国家的グランドチャレンジとして次の７つをあげてる。

（１）次世代の国防と国家安全保障システム
（２）全国民のための改良された健康管理システム
（３）科学的に正確な人体の三次元機能モデルの制作
（４）大規模環境モデリングとモニタリングのための IT ツール
（５）生涯学習のための世界最良のインフラストラクチャの創出
（６）危機管理のための統合 IT システム
（７）先進的航空管理のための技術とシステム

これらは NITRD 計画が中心であるが、その他の研究開発の成果も含まれている。近未来の国民生活に影響の強い、分かり易い研究開発となっている。

2.2.3.3　ハイエンドコンピューティング（HEC）の目指すブレークスルー

HEC で特に関係の深い４つの研究チャレンジについて、各が追求するブレークスルーは次の様になっている。

（１）次世代コンピューティングとデータ記憶技術

・先進的なコンピューティング・コンセプト（**従来存在しなかったアーキテクチャ**、コンポーネント、アルゴリズムを含む）
・システムソフトウェア技術（オペレーティングシステム、プログラム言語、コンパイラ、記憶階層、インプット／アウトプット、パフォーマンス・ツールを含む）
・部品技術、システムソフトウェア、プログラミング環境を統合するシステムアーキテクチャ（部品技術、ノードの機能性、コンフィギュレーション、高度並列計算管理用ソフトウェア、階層的プログラミングを含む）、ネットワーク接続性
・ハイエンドコンピューティングのためのソフトウェア・コンポーネント技術

（２）シリコン CMOS における障壁の克服

・光バックボーンネットワーク上の超安全な通信
・未整理のデータベースの検索、あるいは大きな数字を因数分解するため等のアルゴリズムの速度の飛躍的増加
・分子的プロセスあるいは物理現象の詳細かつ忠実なシミュレーションを可能にする量子コンピュータ
・革新的なコンピュータ構造、3-D アーキテクチャ、ハイブリッド技術
・チップ上で再構成可能なシステム、適応性があり、かつ多様性があるコンピューティング
・プロセッサの性能に比例する記憶性能を提供するメモリ内プロセッサ（PIM）とその他の活動
・新しい計算基盤
　－量子コンピューティング
　－生物ベースコンピューティング

－「スマート・ファブリック」：織り合わせ形バッテリ、光ファイバ・ケーブル、金属コネクタ等の技術を利用して、科学者は、数十 Tera-ops クラスの処理（今日の大型スーパーコンピュータの規模）を行うのに充分なプロセッサを浮き彫り加工し、人が身につけられる織物を作ることが可能である。

－分子電子工学。軍事用次世代戦略コンピューティングのための極度に速く、高密度の処理能力を持った分子スケールの計算。

（3）21 世紀のための多目的、安全、拡張可能なネットワーク

・基礎的ネットワーク研究は下記を含む。

－光ネットワーキング（フロー、バーストとパケット交換。アクセス技術。毎秒ギガビットレベルのインタフェース。プロトコルの階層化。）

－ネットワークのダイナミクスとシミュレーション（自動管理、自動リソース回復、ネットワーク・モデリング）

－耐故障性と自律的管理

－リソース管理（発見と仲介、事前予約、共同スケジューリング、ポリシー駆動型の割当メカニズム）

－無線技術（発見、共存とコンフィギュレーションのための技術標準）

－帯域幅に対する要求を支援する能力の増強

－頑強性の改善、数十億のデバイスの一時的相互作用の取扱いとネットワークの遠端からのアクセスを最大化するためのネットワークの増強と拡張。これにはユビキタス・ブロードバンド・アクセス方式、無限ネットワーキング、セキュリティとプライバシ、ネットワーク管理等の研究がある。

－**地球規模の**ネットワークおよび情報インフラストラクチャを理解すること（エンドトゥエンド・パフォーマンス、バックボーン構造、アプリケーション）

・新しい分野の応用を可能にすること（**分散型データインテンシブ**・コンピューティング**共同**研究、**コンピュータ操縦**による科学的シミュレーション、**遠隔**ビジュアライゼーション、遠隔機器の操作、大規模**分散型**システム）

（4）科学と工学における米国の強みを維持するための先進的 IT

・科学者、技術者、社会が関心を持つ現象をより速く、より高解像度、より現実的な立体的シミュレーションで処理可能にするための、より高能力で相互運用性を持ったコンピューティング・システム、記憶システム、ネットワーク、ソフトウェア開発能力（言語とツールを含む）、人・コンピュータ間対話技術

・複雑で学際的なモデリング・システムを構築し、理解し、評価するためのアプローチ

・科学者や技術者が**分散されたハイエンドのコンピューティング・リソース**により良好にアクセスでき、利用することを可能にする言語とツール

・**分散型**の学際的チーム支援のための IT

2.2.3.4　FY2003 予算に見る研究開発目標の特徴

FY2002 Blue Book は同時多発テロ以前にまとめられたものであり、この対策は意識的には加味されていない。先に述べた OSTP の FY2003 予算の分析資料にはテロ以後の対策を含む NITRD の目指す重点目標が簡潔にまとめられている。詳細は別途 FY2003 Blue Book に纏められるはずであるが、それの予告が見られるのでここに紹介する。
次の様な技術の開発を達成することを重点課題に挙げている。

（1）エンドツーエンドの光ファイバーネットワーク
　これはネットワークバンド幅とネットワーク・セキュリティの大幅の向上をもたらす。
（2）クラスタおよび Grid コンピューティングを可能にする新しい技術
　これは科学研究ネットワークのための高性能計算機への初めてのアクセスをもたらす。
（3）ネットワーク･セキュリティ･プロテクション技術で次のようなものを含む。
　不正侵入検出、リスクおよび弱点解析、大規模情報レポジトリーの管理、運用

2.3　ハイエンドコンピューティング分野の新しい動向
―　実用化に向かう超並列、超分散コンピューティング　―

2.3.1　SC2001 に見る動向

ハイエンドコンピューティング関連の最大のコンファレンスは、毎年 11 月に開催される、正式名称を SCxxxx High Performance Networking and Computing というスーパーコンピューティング・ショーであり昨年の SC2001 は左記の通り開催された。（参照 http://www.sc2001.org/）テーマは“Beyond Boundaries”であった。

SC2001
Denver, CO
November 10-16, 2001

SC Global といって Access Grid のテクノロジを利用した遠隔会議による参加も初めて行われ、世界 40 カ所以上と接続された。同時に Grid2001 が特別ワークショップの形で丸 1 日開催された。展示会場でも SUN の“Sun Powers the Grid”なるキャッチフレーズ等 Grid にあやかった展示が多く見られ、SC2001 の主役は Grid の感が強かった。

基調講演はゲノム塩基配列の解析を国際チームに先立って 2001 年に終了したセレラ・ジェノミックス社の CEO クレッグ・ベンター博士であった。バイオインフォーマティクスは今や、ハイエンドコンピューティングのアプリケーション分野のトップに踊り出した

感がある。コンピュータパワーを駆使したショットガン方式が成功を収め、一説では、世界中のコンピュータパワーを集めてもまだ足りないとも言われている。

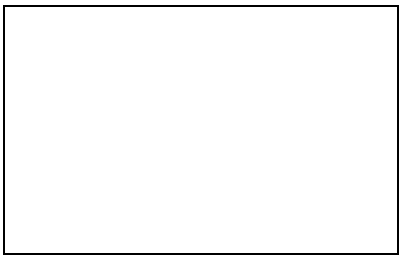
SC2001 の前日 11/9 に発表された Top500（18th 版）においても超並列マシン ASCI が上位を独占し、近年 PC クラスタもその順位をのばしておりハイエンドコンピューティングの流れはますます超並列、超分散化が加速している。今回、テロの影響もあり日本のベクトル型コンピュータは地球シミュレータ、富士通、日立が参加取り止めた為、更に印象が際だった気もする。付表１～３に世界の Top20、世界のクラスタ Top20、日本の Top20 を載せた。

なお、SC2001 の報告については恒例になった東大小柳教授のレポートが詳しいので御参照ください。

（http://olab.is.s.u-tokyo.ac.jp/~oyanagi/reports/SC2001）

次回 SC2002 は第 15 回目であり下記となる。

SC2002
Baltimore, MD
November
16-22, 2002



参照サイトは　http://www.sc-conference.org/SC2002/

テーマは“From Terabytes to Insights”となっており、大規模データの処理が重要テーマに上がっている。

2.3.2　超並列コンピューティング

（1）TOP500 にみる超並列マシン

SC2001 と同時に発表された Top500（参照サイト：http://clusters.top500.org/ ）によると超並列マシンが上位を占める、特に近年は ASCI マシンの独壇場であった。Top 1 は昨年と同様の ASCI White である。（実行性能は昨年の 4.9 Tflops から 7.2Tflops に向上しているが構成は同じ）表 2.3 に Top５を示す。第２位にクラスタシステムが入って来ているのが注目すべき流れ。

表 2.3　Top 5　世界のハイエンドコンピュータ

順位	メーカ	マシン	R_{max} (GFlops)	設置場所	CPU 数
1	IBM	ASCI White,SP Power3 375 MHz	7226	Lawrence Livermore National Laboratory	8192
2	Compaq	AlphaServer SC ES45/1 GHz	4059	Pittsburgh Supercomputing Center	3024
3	IBM	SP Power3 375 MHz 16 way	3052	NERSC/LBNL	3328
4	Intel	ASCI Red	2379	Sandia National Labs	9632
5	IBM	ASCI Blue-Pacific SST,IBM SP 604e	2144	Lawrence Livermore National Laboratory	5808

（2）Peta Flops を目指す Blue Gene プロジェクト

従来、IBM が独自に開発を進めていた Blue Gene プロジェクトであるが、2001 年 11 月に DOE/NNN(Lawrence Livermore National Laboratory が対応)と共同開発を進めることを発表した。2005 年に 200Tflops を目指す。

http://www.research.ibm.com/bluegene/BG_External_Presentation_January_2002.pdf に詳細計画が紹介されている。図 2.2 に構成概念図を示す。

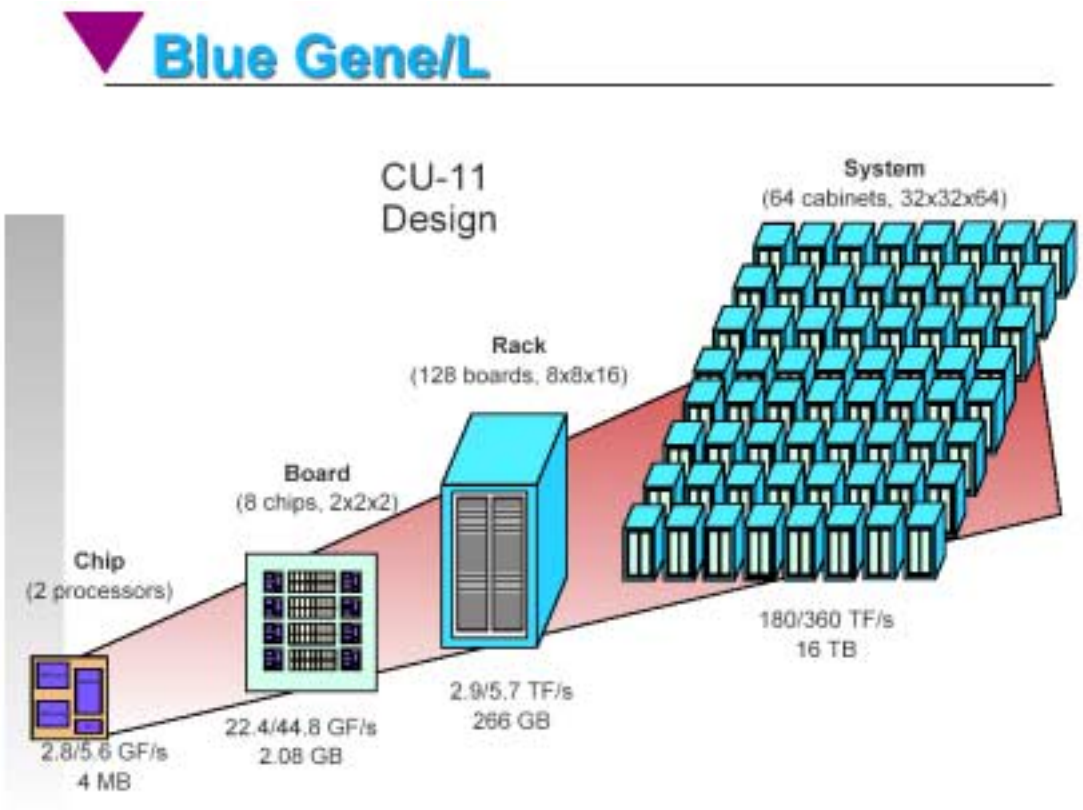


図 2.2　Blue Bene/L の構成概念図

現在最速のスーパーコンピュータ（Option White か）に比べて性能では 15 倍速く、電源容量は 1/15、大きさは 1/50 から 1/100 と豪語している。このあと更に Blue Gene 計画は 1Peta Flops を目指す。（規模縮小したとの声も有るが定かではない）

この Blue Gene 計画は ASCI 計画と合体したことになり、IBM は国家予算の後ろ盾を付けたことになる。日本も対抗が必要と思うが、地球シュミレータ計画の後のハイエンドコンピュータのビッグプロジェクトは無い。

（３）ベクター型超並列マシン“地球シミュレータ”

ベクター型プロッセサを超並列に利用し 40Tflops と汎用スーパコンピュータ世界一の性能を叩き出す地球シュミレータが横浜市の海洋科学技術センター内に 2002 年３月末に完成した。次回（2002 年６月）19th 版の Top500 で第１位が期待される。詳細はプロジェクトに当初より関わっている本WGの横川委員、妹尾委員の３章の報告書に詳しいので参照下さい。

今や、ベクター型は日本しか作っておらずトレンドから外れた感もあるが地球シミュレータの実績如何では、見直しの可能性もある。（がんばれ日本のスーパーコンピュータ！）

（４）専用スーパコンピュータ

汎用のスーパコンピュータのコスト、性能の壁を破るべく特定分野に特化した、専用スーパコンピュータの分野も性能向上がめざましい。SC2001 でも東大の重力多体シミュレーション専用の Grape6 は 11.55Tflops（ピーク性能 32Tflops）を達成し昨年に続きゴードンベル賞（Peak Performance 部門）を受賞した。更に 2002 年 3 月 5 日にはピーク性能 48Tflops を達成し世界最高速となったと発表した。なお専用マシンの低コストについては“GRAPE-6 の開発総予算はわずか 5 億円であり、400 億といわれる ASCI White の約 1/100 でしかない”と紹介されている。

参照サイト：

http://grape.astron.s.u-tokyo.ac.jp/pub/people/makino/press/2002-symposium.html

2.3.3 クラスタシステム動向

（１）Top500 に見るクラスタシステムの増加

今回は Top10 で見ても２台のクラスタシステムがランクインした。従来はミドルクラスを占めていたがハイエンドの世界でも民生の部品を応用したクラスタシステムの流れが押し寄せてきている。専用部品を使用したスーパーコンピュータに対するコストパフォーマンスの良さが一番の原因であろう。表 2.4 に世界の Top５を示す。チップの種類を見ると Compaq 社の Alpha チップがほぼ上位を独占している。

表 2.4　Top５　クラスタ型ハイエンドコンピュータ

順位	Top 500	メーカ	マシン	Rmax (GFlops)	設置場所	CPU 数
1	2	Compaq	AlphaServer SC ES45/1 GHz	4059	Pittsburgh Supercomputing Center	3024
2	6	Compaq	AlphaServer SC ES45/1 GHz	2096	Los Alamos National Laboratory	1536
3	30	Self-made	CPlant/Ross Cluster	706.7	Sandia National Laboratories	1369
4	31	Compaq	AlphaServer SC ES45/1 GHz	706	Australian Partnership for Advanced Computing (APAC)	480
5	34	IBM	Titan Cluster Itanium 800 MHz	677.9	NCSA	320

（注）本表は Top500 サイトで Computer クラスを Cluster 分類でソートしたもの。

（２）世界最速のクラスタマシン

PSC(Pittsburgh Super Computing Center に設置された Compaq 社の民生 Alpha チップを使用しており、概略仕様は下記。2001 年より稼働。　NSF の PACI(Partnership for Advanced Computational Infrastructure)を通して研究開発に共同利用される。

Alpha チップ：3024 個
ピーク性能：6.0 Tera flops
メモリ：3.0 Tera Bytes
ディスク：50 Tera Bytes

図 2.3　PSC に設置の世界最速クラスタ

参照サイト：http://www.psc.edu/publicinfo/news/2001/terascale-10-01-01.html

（３）日本のクラスタ動向

ところで日本のクラスタは今のところ世界の第６位であり、お台場の生命情報科学研究センタ（CBRC）が利用している NEC 製のクラスタ Magi Cluster（Pentium3,933Mh）である。（世界のクラスタ Top20 については付表 3 参照）

なお、日本でも PC クラスタコンソシアムが 2001 年 10 月発足しており、産官学共同の促進が始まった。新情報処理開発機構(RWCP)が開発した SCore クラスタシステムソフトウエアおよび Omni OpenMP コンパイラが中核となっている。

参照サイト：http://www.pccluster.org/

（4）InfiniBand

ベオウルフ型クラスタを構成する民生ネットワークとしては GigaBit Ether, Myrinet 等が使用されているが、最近 InfiniBand が注目を浴びている。ネットワーク性能としては単線の2.5Gbitを１本／４本／12本で各 2.5 G bits, 10 G bits, and 30 Gbits／秒の速度を提供する。Compaq, Dell, Hewlett-Packard, IBM, Intel, Microsoft and Sun Microsystems が主体になって 1999 年 10 月に InfiniBand Trade Association を設立し,2000 年 10 月に標準仕様を策定している。（これだけの利害関係のある、企業がうまく仕様を合意できるか心配であるが。）

参照サイト：http://www.infinibandta.org/ibta/

本ＷＧにおいても、米国で InfiniBand 関連ハードウエア主体のベンチャ企業 RedSwitch 社の CEO Dr.Wen Lee にご講演を頂いた。米国では InfiniBand 関連の企業化が始待っている。参照サイト：http://www.redswitch.com/

現在,InfiniBAnd 関連の製品は、量産化されておらずサンプル価格的なものが多いが、今後の市場拡大と価格の低下が期待される。

2.3.4 超分散コンピューティング

（1）Grid コンピューティング

SC2001 の展示会でも分かるように、Grid コンピューティングへの流れが加速している。

世界中のコンピュータを繋ぎコンピュータ資源を有効に活用しようと。ハイエンドコンピューティングの世界でも超並列を持ってしても単独のコンピュータセンタでは物理的、経済的にも制限がある。まずスーパコンピュータセンタを高速ネットワークで接続し仮想的研究所を構築する構想から始まっている。ASCI 等の超並列コンピュータ自体もその一つのノードになる。ASCI Grid Service という広報用 CD も SC2001 では用意されていた。ASCI の超高速コンピュータもポータルの一つになるのだ。

主に計算能力資源を活用する Computational Grid, ネットワークを介してマルチメディア情報をやりとりし遠隔会議を実現する Access Grid、大規模な高エネルギー物理のデータを扱う EU 連合の Data Grid 等多くのプロジェクトが世界的に行われている。

ただし Grid はまだ公式な標準が有るわけではなく、デファクトスタンダードの世界である。ソフトウエア・ツールキットとしてはGlobusがデファクトの位置を確立している。

今後この流れは、研究開発向けから一般ビジネスの世界に広まるのは必至であろう。

なお本WGの招待講演者出もある産総研の田中氏が展示者としてSC2001に参加し、３章に状況を詳細にレポートして頂いているので参照下さい。また昨年の当 WG 調査報告書に関口委員（産総研）の報告“Gridによる世界戦略とグローバルコンピューティング技術”で動向が分かり易く紹介されているので併せてご参考下さい。

（２）Tera Grid 計画

世界最大、最高速の分散ネットワーク環境を構築すべく、NSF から$53M 資金提供を受け、４つの研究所（NCSA, Argonne, Caltech, SDSC）を光ファイバ高速バックボーンネットワーク（40Gbits）で結ぶ。完成時には 13.6TeraFlops の Linux クラスタコンピュータパワーと 450TeraBytes データ容量と高精細の Access Grid 環境が提供される。

参照サイト：http://www.teragrid.org/index.html

図 2.4　４つの研究所を結ぶ Tera Grid バックボーンネットワーク

参照サイト：http://www.teragrid.org/img/teragrid.sm.jpg

IBM, Intel, Qwest 社が企業として参加しており、64Bit Itanium チップを 320 個用いた IBM 製の Linux クラスタマシンが 2001 年 10 月からテストを開始した。

日本では、Top100 スーパコンピュータのうち６台が集中する、世界でも有数なつくば地区に“つくば WAN”が完成し 2001 年３月 22 日に開通式を行った。ネットワークは 10Gbps の光ループが使われている（総容量は 570Gbps）。当初はつくば地区にとどまっているが、今後地球シミュレータはじめ全国の大学、研究所間の接続が検討されている。

（３）ベンチャ企業動向

Grid コンピューティングが研究開発分野で注目を浴びる中、Grid 関連で興味深い企業化が進行している。SC2001 等で目に付いた企業をあげる。

・AVAKI 社：http://www.avaki.com/

Virginia 大で Legion を開発した人（Dr. Andrew Grimshaw）がスピンアウトし Applied Meta Computing 社を設立（1998 年 8 月）、2001 年 6 月に AVAKI 社に。現在は Legion ベースにソフトウエアパッケージ AVAKI2.1 Software Grid を販売している。用途として“Grid-Based Collaboration in Life Sciences Research”なるホワイトペーパも発行されバイオ関連を有力なターゲットとしていることが伺われる。

・Platform Computing 社：http://www.platform.com/

創業は 1987 年で分散コンピューティングの LSF(Load Sharing Facility)を売り物にしているカナダの会社であるが、今回世界で最初に Globus Toolkit の商用版をリリースすると発表した。Grid 業界の Red Hut を目指している。

・Entropia 社：http://www.entropia.com/

分散コンピューティングを基盤モデルとして“どこからでもスーパーコンピュータのパワーがアクセス出来るように”とのコンセプトで 1997 年に創業された。Grid コンピューティングそのものがビジネスモデルになっている。利用例として２のN乗引く１が素数になる Mersenne 素数の発見では 17 万台以上の PC が接続された Entropia Grid を用いて 200１年 11 月には 400 万桁を越すの 39 番目の素数が発見された。Grid コンピューティングのパワーをうまく活用できた例といえる。

（参照：http://www.mersenne.org/prime.htm ）

2.3.5 量子コンピュータの動向

IBM 研究グループは 2000 年 8 月に５量子ドットの実験に成功した後、IBM Almaden 研究センターとスタンフォード大学は共同で 7 量子ドットの量子コンピュータにて 15=3x5 の因数分解に成功と 2001 年 12 月 20 の Nature 誌に発表された。

これは試験管に入れられた特別な分子を核磁気共鳴装置の核スピンの傾きを制御し計算を行うことで実現した。

これは、1994 年に当時 AT&T 社の Shor 氏が唱えた Shor のアルゴリズム（量子コンピュータは今日の汎用コンピュータに比べて圧倒的なスピードで因数分解を計算できる）の初めての実験による証明になった。

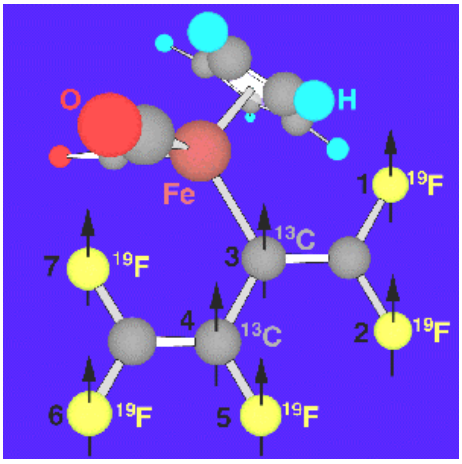


図 2.5　量子ビットを構成する核スピン

参照：http://www.research.ibm.com/resources/news/20011219_quantum.shtml

研究は一歩一歩に進んでいるが、SC2001 の展示でも、量子コンピュータ、分子コンピューティングはパネル展示の原理の解明段階であり目立ったアッピールはしていない。実用になる量子コンピュータの実現はまだ長期の研究開発が必要である。米国は長期研究テーマである新しい計算基盤に向けて研究開発投資を怠りなくやっている。

2.3.6　超並列、超分散を容易に開発できるソフトウエアの研究開発動向

FY2001 より PITAC のソフトウエアの生産性に対する憂慮からの勧告に従い、IT R&D 計画に SDP（Software Design and Productivity）プログラムコンポーネントエリアが新設された。FY2002 Blue Book でも研究チャレンジ “現実世界のためのソフトウェア作成”にて重要テーマにあげている。目標としては次がある。

（1）ソフトウェアとシステム設計の科学

－言語とコンパイラ、例えば、エンドユーザのためにソフトウェアの仕様と開発を容易にするドメイン専用言語、ならびに使用は容易だが誤りを生じにくい言語。

－ソフトウェアとシステムを構成する効果的な方法、すなわち複雑なシステムを構成し、分析し、検証し、広範囲にわたって分散された異種システム上で相互運用可能にするためのより優れた技術

－適応性があり、かつ反射性のコンポーネント、構成のフレームワークとミドルウェア、拡張可能分散型ソフトウェア・システム構築の理論的基礎

（2）工学的プロセスの自動化

－分散、自律、組込み型ソフトウェアの開発のための技術を含み、開発時間を短縮し、信頼性を向上させるソフトウェア・コンポーネントを組合せる方法。ソフトウェア開発の自動化

－ソフトウェアと物理システムを仕様化し、分析し、試験し、検証する、統合ソフトウェアとシステム開発プロセス

－ネットワーク・アプリケーションの相互運用性

－統合されたドメイン専用開発環境の迅速な構成とカスタマイゼーションを可能にする統合された構成可能ツール環境

（3）パイロット・アプリケーションと実験による評価

－組込み型ソフトウェア・アプリケーションとその他の複雑なアプリケーションのための技術

－ソフトウェアとシステム開発プロジェクトの実証的研究

研究課題の中には超並列、超分散を意識した研究開発内容が多々見られる。

単体によるハードウエア高速化の限界の打開が超並列、超分散に向かっているので、ハードウエアの困難さ以上にソフトウエアにその負担が大きくふりかかってきている。特定の分野（粗な結合で超並列、超分散でも性能が出やすい所）では実用になっているが、これを汎用コンピューティングの世界に向けて実用化するには更に長期の研究が必要である。

（若杉　康仁　幹事）

第３章　ハイエンドコンピューティング研究開発の動向

3.1　概要

本章では、各委員および外部講師による、ハイエンドコンピューティングに関する研究開発動向の調査報告をまとめている。

3.1.1　調査方針

各委員の専門およびその周辺分野において、今後、研究的に急速に発展したり、あるいは市場にインパクトを与えそうな分野やテーマの抽出と、それらの技術に関する、(米国をはじめとする)先端研究とわが国との格差の調査・評価を行うことを主眼にした。したがって、今後重要になると思われる領域にはどのようなものがあるかを抽出するという点に力点を置くことにした。また、調査範囲は多岐にわたっており、当ワーキンググループ委員の専門分野で網羅できない部分は外部講師に依頼した。

3.1.2　調査報告の分類

従来の半導体技術の延長による高速化は限界に達することが示され、これからの計算機の高性能化の方向として、様々なレベルの並列分散コンピューティングが研究の中心になると考えられている。本報告書の調査内容は、地球シミュレータのベクトル並列、Grid の広域分散、クラスタを Grid/P2P で統合するメガスケールの並列分散など多岐にわたり、各報告を従来の並列か分散かで分類することは適切でない。今年度は、平成 12 年度「わが国が行う情報技術研究開発のあり方に関する調査研究（その 4）[†]」を参考にし、「アーキテクチャ＆新計算モデル」、「基本ソフト＆ミドルウェア」、「応用システム＆応用分野」の 3 階層で分類を行った。

なお、今年度は次の 2 名の外部講師に調査報告をまとめていただいた。

(1) 産業技術総合研究所　ハイエンド情報技術グループ
　　田中　良夫氏
　　「SC2001 にみる Grid の最新動向」
(2) NTT　未来ねっと研究所
　　星合　隆成氏
　　「P2P の理念および実現技術：SIONet の全貌」

[†]http://www.icot.or.jp/FTS/REPORTS/H11-reports/H1203-AITEC-Report2/AITEC0003-R2-html/AITEC0003R2-ch2-1.htm#1.3

3.2 アーキテクチャ及び新計算モデル

3.2.1 半導体性能向上神話崩壊後のアーキテクチャ

天野　英晴　委員

1. 半導体のスケーリング則

CMOS LSI の動作速度はプロセスの進歩につれて向上を続け、新しいプロセスを利用することは、それだけで性能の向上につながった。これは、以下の原理に基づいていた。

- 微細加工技術が発展すると、ゲートのチャネル幅が狭くなり、キャリアの移動時間は減り、ゲートの外でも配線の長さが短くなり、これらは全て性能の向上に寄与する。
- 微細化に関する有名な法則が MOS FET のスケーリング則[1]である。微細加工技術の発展により、ゲート酸化膜厚を 1/K にすることができれば、チャネル幅、ゲート長など素子寸法を 3 次元的に 1/K にすることができる。これにより、集積度は K^2 となり、動作速度は配線長が短くなることにより 1/K となり、素子当りの消費電力は $1/K^2$ となる。

このスケーリング則は 0.25μm プロセスが用いられた 2000 年位までは、ほぼそのまま通用した。しかし、2001 年に状況は変化する。

2. 性能向上神話の崩壊

筆者は、1991 年以来、主として CMOS Embedded Array を用いて、スイッチ、ルータ、インタフェース用プロセッサなどの開発にたずさわって来た。開発した主なチップを表 1 にまとめる。

表 3.2.1-1　開発に携わったチップ

年代	プロセス	チップ名称	コア部動作周波数	機能
1991	1.0μ（川鉄）	TBSF チップ	50MHz	マルチプロセッサ用スイッチ
1993	0.5μ（NEL）	PBSF チップ	90MHz	3 次元スイッチ
1995	0.5μBiCMOS（日立）	RDT ルータ	60MHz	JUMP-1 用ルータ
1997	0.35μ（東芝）	MBP-light	50MHz	JUMP-1 用プロセッサ
1999	0.35μ（富士通）	RHiNET-1/SW	100MHz	RHiNET 用スイッチ
2000	0.18μ（日立）	RHiNET-2/SW	200MHz	RHiNET 用スイッチ
2001	0.14μ（日立）	Martini	66/133MHz	RHiNET 用インタフェース

ここで、1.0μm から 0.18μm まではスケーリング則そのままに、プロセスの発展につれて順調に性能を上げることができた。2000 年に RWCP、日立と共同で開発した RHiNET-2/SW は、8Gbps のリンクを 8 組交換することが可能であり当時の CMOS のスイッチとしては世界でも最高速レベルを達成することができた[2]。この時期までは、新しいプロセスをいち早く利用することこそ、性能向上への早道であった。

ところが 2001 年、プロセスが 0.18μm から 0.14μm に進む際、状況が一変した。今までは、新しいプロセスを用いる時には、ゲートの遅延時間が小さくなったことで、まず設計に余裕を持って当ることができた。また、0.5μm までは、仮負荷の STA(Static Timing Analysis:遅延解析)と実配線時の STA はほとんどぴったり一致した。0.35μm から 0.18μm では場合によってはやや相違を感じることがあったが、10%内外であった。

ところが、2001 年に日立の 0.14μm プロセスを用いて、クラスタコンピュータ用ネットワークインタフェースチップ Martini[3]の開発を行ったところ、まず、ゲート単体の性能が 0.18μm プロセスに比べ余り向上していないことに気づいた。次に、仮負荷時の STA において 8nsec 以内に抑えたクリティカルパスが、実配線後にモジュール間の配線が加わったところ、16nsec に増大するという事態が生じた。もちろん Martini で利用したプロセスおよびツールが、このチップ面積に対応する点で様々な問題を持っていたことが原因の一つではあるが、基本的には、この時点でスケーリング則が崩壊しつつあったことが大きい。

スケーリング則の崩壊は以下の原因による。

(1) 微細化技術が進むと、配線寸法が縮小されるため、配線抵抗が断面積の減少により大きくなるとともに配線間容量も配線間の距離の減少と共に大きくなる。このため、配線の信号遅延と相互干渉が大きくなる。
(2) スケーリング則により減少し続けるチャネル抵抗に代って、電極やソース、ドレイン部の寄生抵抗が支配的になり、素子の性能の向上が抑えられる。

このため、新しいプロセスを単に利用するだけでは、性能を上げるのは困難となった。今後、プロセス技術にブレークスルーがない限り、プロセスが進むにつれて益々この傾向は大きくなると考えられる。すなわち、半導体性能向上神話は崩壊し、今年から新しい時代に突入したのである。

このような節目の時期を迎えているのだから、例えば日経エレクトロニクスで特集が組まれ、学会では専門家が深刻な表情でパネルディスカッションをやり、など世の中がもっと騒いでも良いのではないかと思う。しかし、目下、業界の話題は C 言語ベースの高位レベル設計などであり、全く静穏である。これは以下の理由に基づくと考えられる。

(1) 深刻な事態に陥っているのは、最先端のプロセスを利用し、しかも巨大な面積を利用する一部の設計者に限られている。彼らは新しいプロセスの利用時のトラブルには慣れているし、予算額も大きいため、今のところ事態をなんとか回避する方法を見いだしている。

実際、0.18μm から 0.14μm/0.13μm への移行は、銅配線の利用、多層配線の活用など新しいプロセス技術の利用（コストはかかるが）、クロックの分離（Martini では 3 種類のクロックをモジュール毎に使い分けることで、性能低下をなるべく抑えて、配線遅延の問題に対処している）、フロアプランの改善などで、がんばれば、かなりなんとかなるレベルである。

(2) 性能向上に対する圧力が減っている。

今までは、新しいチップを新しいプロセスで作る以上、従来の同様の目的のチップと比較して、相当の性能向上が期待され、端的に言うと動作クロックが上がっていなければならなかった。しかし、最近はむしろ消費電力が少ない、実装が楽、コストが小さくて済む等のメリットがあれば動作クロックが高くならなくても良い（あるいは下手に周波数が高いと電力を食って大変）という考え方が広まった。このため、プロジェクトのボスからのこの面での圧力が減ってきた。

(3) 0.25μm 以下でスケーリング則が通用しなくなることは、以前から予測されていた。

したがって、プロセス屋さんは配線遅延や寄生抵抗を下げるための技術を着実に開発中であり、それでも今まで程には性能が上がらないのは、物理学の法則により当たり前であると考えているので、今更騒ぐ必要はなかった。

しかし、現役のアーキテクチャ屋は、筆者を含めて IC 世代であるため、「プロセスが新しくなっても、ぼーっとしていては性能が上がらない」というのは、生まれて始めての事態であり、従来の根本的な考え方が崩壊したわけであるから、充分騒ぐに足ると思う。また、今までアーキテクチャ屋は「限界が来ない」ことに慣らされてしまっていた。物心ついて以来、CPU の性能は上がり続け、DRAM の容量も増大を続け、プロセス技術も進み続けていた。スケーリング則だって色々言われていてもこのまま通用し続けるのではないか、と無意識のうちに考えていたところに、今度に限って予測が見事に的中し、「限界」にいきなり直面することになった。

ただし、ここで注意しなければならないことは、新しいプロセスの利用が、性能にまったく貢献しなくなったわけではない点である。まず、集積度に関しては依然としてスケーリング則は通用するし、以下の条件を満足すれば性能を向上させることができる。

(1) 同一面積に、同じような方法で実装した場合に、クロック周波数を上げることが難しいのであって、全く同一の回路ならば、面積は小さくなるし、周波数も若干上げることができる。
(2) レイアウトレベルで最適化すれば、クロック周波数を上げることが可能である。
(3) 集積度は依然として向上するので、今までチップ外に実装したモジュールをチップ内に持つことで、バンド幅の向上や入出力の遅延の減少により性能向上が見込まれる。

3. 絶滅するもの、発展するもの

3.1 さらば Embedded Array

さて、この新しい時代を迎えて絶滅するであろうと思われるのが、Embedded Array を含めたゲートアレイである。Embedded Array とは、メモリや CPU などのあらかじめ設計ができているモジュール(ハードコア IP)は、レイアウトの形で拡散層に実装し、その他の部分は、あらかじめ拡散層上に作られたトランジスタに対して配線のみを行ってランダムゲートを実現する方法であり、ASIC の主流として使われてきた。Embedded Array はセルベースド設計に比べて、レイアウトが自動化されており、設計期間が短いことから、新しいチップを短期間で設計する場合に向いており、表 1 に示すように、筆者らも一貫して新しいアーキテクチャの検証を行うのに用いてきた。部分的にレイアウトが行われている高速モジュールとランダムゲートの柔軟性を利用することにより、RHiNET-2/SW のように、相当複雑なルーティングを行うスイッチを、部分的には 800MHz という高速な周波数で動作させることが可能になる。

ところが、Embedded Array は、配線遅延が支配的になる新しい時代においては、致命的な弱点を抱えている。すなわち、今後、遅延を支配する要素である配線を、あらかじめ拡散層に配置されたハードコア IP やトランジスタを接続する形で、後から行なわなければならない点である。配線に自由度が大きいが故に、配線間の干渉についてはきちんとしたマージンを設けざるを得ない。配線をドライブする場合の遅延に対しては、最小と最大の遅延差を設計上大きめに設定せざるを得なくなる。このため、クロックのスキューを考えてホールドタイム合せを行うと、最小遅延に対処するために数多くのゲートを信号線上に配置せざるを得なくなり、これらは最大遅延を考えた場合には確実に性能の足を引っ張る要因になる。

また、配線遅延を克服するには、長い配線に対して適切なサイズのリピータを挿入する必要があるが、トランジスタの形状が決まってしまっている Embedded Array は基本的に通常のゲートをリピータとして用いざるを得なくなり、長い配線の随所にゲートが入ることになり、結局、遅延の短縮効果が大幅に制限される。

3.2　セルベースド設計と水平分業の発達

一方で、セルベースド設計は、チップやベンダに依存しないレイアウトツールの普及、Physical Compiler などのフロアプランと論理設計を結び付けるツールの発展、水平分業の発達、特に台湾のベンダを代表とする製造のみの半導体ベンダの台頭により、ASIC の主流になるであろう。すなわち、フロアプランや配置配線を半導体ベンダに頼って論理合成後の Netlist でデータインする設計法から、ベンダに依存しないレイアウトツールを用いてレイアウトレベルでデータインする設計法が主流になると考えられる。特に最近の台湾半導体ベンダは、同一ダイ上での複数チップの相乗り実装を押し進めている。この形式は MOSIS や VDEC などの研究レベルで用いられてきたもので、今後安価でチップを作成する方法として、中小企業や大学研究機関での利用が進むであろう。

この場合、配線遅延の克服が設計者の仕事となる代りに、リピータのサイズやゲートの種類、フロアプランを工夫することにより、性能を上げることができる。簡単に言って、これからのアーキテクチャ屋は自分でレイアウトまでできないと使い物にならないことになる。また、新しい時代に究極的に性能を追求するならば、レイアウトまで最適化せざるを得ないかもしれず、この場合、一つのチップの設計には何人もの技術者の協力による大掛かりなものとなる。

3.3　リコンフィギャブルデバイスの台頭

一方で、これとは逆の流れとして、FPGA などのリコンフィギャブルデバイスの発展がある。FPGA あるいは CPLD は様々な構成法があるが、基本的に以下の特色を持っている。

(a) Logic Block をモジュールとした規則的なアレイ構造を持つ。
(b) Logic Block 間は、隣接の配線および専用の長距離配線とスイッチによって結合される。
(c) Logic Block のテーブルやスイッチの接続はメモリ中のデータによって決まる。

従来 FPGA の動作速度は、Logic Block 間の配線によって遅くなるとされてきたが、厳密に言うとこれは誤りで、配線を変えるためのスイッチの遅延時間が大きかった。ところが、配線遅延が支配的になると、Embedded Array で単に線を引く場合も、長距離の配線については、一定の間隔でリピータの挿入を行わなければならない。一定の間隔で置かれたスイッチがリピータの代りをするとなると、長距離配線遅延に関しては Embedded Array と FPGA は変わりがないことになる。むしろ、最適化された抵抗の少ない専用配線領域と、きちんとしたドライバを持つ FPGA の方が有利になる可能性すら考えられる。

したがって、セルベースド設計を行う程の数量が見込まれず、性能も必要ない場合は、FPGA を用いるのが有利となる。このように新しい時代では、ある程度数量が見込まれる場合は、セルベースド設計をきっちりやって、レイアウトデータをデータインする。そうでなければ FPGA を用いる、という形で LSI 利用は二極化されることが予想される。

4. 新しい時代に適合したアーキテクチャ

さて、アーキテクチャ屋としては、新しい時代に適合したアーキテクチャは何かを考える必要がある。

4.1 VLSI アーキテクチャ

1980 年代に H.T.Kung のシストリックアレイを中心に、VLSI に適したアーキテクチャ(VLSI アーキテクチャ)の研究[4]が行われた。この研究成果は役立つだろうか？ この時代の VLSI アーキテクチャは以下の前提に基づいていた。

(1) 配線遅延が支配的になる。
(2) 配線領域が無視できない程大きくなる。
(3) I/O ピン数がボトルネックとなる。

この前提に基づき、配線は全て局所的、I/O をアレイの両端のみで行い、データをリズミカルに流すと共に計算を行うシストリックアーキテクチャが提案された。実際に配線遅延が支配的になった現在、この考え方は高い先見性を持っていたことになるが、それではシストリックアーキテクチャ自体が今後大幅に発展するか、というと必ずしもそうではないように思える。これは以下の理由に基づく。

(a) ３つの前提のうち、(1)は深刻だが、(2)(3)はさほどでもなかった。
　確かに配線領域は無視できないが、微細加工技術の発展に従って、ゲート同様縮小されるため、時代が進むにつれて問題が深刻化するに至ってはいない。また、確かに I/O ピン数のボトルネックは一貫して深刻な問題だが、パッケージ技術の発達によって相当緩和されているし、半導体面積の増大により、主要なパーツを内部に抱え込むことで対処することも可能になっている。したがって、チップ内のモジュール間の交信を無理に局所的に抑えることは効果があっても、配線面積や I/O ピンを抑制することはシストリックアーキテクチャ導入の要因にはならない。

(b) 80 年代の VLSI アーキテクチャは何ゆえかオンチップメモリの活用が考えに入っていない。
　膨大なオンチップメモリが混載可能になった現在、チップ内にデータを記憶しないで流してしまうシストリックアーキテクチャはむしろ不利であると言える。

当時の研究成果も踏まえ、今後、新しい VLSI アーキテクチャは、以下のようなガイドラインに従うことが考えられる。

① モジュール化された構成は高速化の可能性が高い。特にレイアウトレベルで最適化されたモジュール（CPU コア、メモリ等）を用いれば、効果はさらに大きい。
② 多数のモジュールの繰り返し構造は有効である。
③ 低レイテンシなデータ交換は局所的なものに限られる。大域的なデータ交換はレイテンシを前提に行う。
④ オンチップメモリを有効利用する。

残念なことに、上記のガイドラインに従うと単純な 2 次元のアレイ構造にスイッチ等の大域的通信手段を持つといった、ちょっと考えると「つまらない」構成以外なかなか思い浮かばない。しかし、応用分野を限ったオンチップ並列アーキテクチャは、今後の研究分野として大きいと思うし、一見つまらない構成以外思い浮かべない状況こそがアーキテクチャ屋の腕の見せどころと言えるかもしれない。

4.2 リコンフィギャブルアーキテクチャ

FPGA が新しい時代に大いに発展する素子である以上、この素子をベースとするリコンフィギャブルアーキテクチャは、大いに期待の持てる方式である。このアーキテクチャは、問題のアルゴリズムを直接ハードウェア化して、FPGA 等のリコンフィギャブルデバイス上で実行する方式である。

昨年のレポートで触れたので、ここでは省略するが、高位レベル設計技術の発達と共に、益々応用分野が拡大している。さらに、マルチコンテキスト FPGA や非同期 FPGA など、今後の発展が期待される新しいデバイスも登場しており、日本の企業、研究者の貢献も大きい分野である。

以上、まとめると、この稿の結論は以下の通りである。

(1) 半導体が新しい時代に突入したことを、はっきり認識すべきである。
(2) 配線遅延を克服するためのデバイス技術の開発と共に、この時代に向けたアーキテクチャ方式の確立も重要なテーマである。特に応用分野向けの並列アーキテクチャ、リコンフィギャブルアーキテクチャは有望であり、この分野の研究プロジェクトに研究資金を重点的に導入すべきである。

5. 計算機アーキテクチャ屋(Computer Architect)の将来

ぼーっとしていると性能が上がらない以上、今後性能を左右するのはアーキテクチャの良し悪しによるところが大きいわけで、アーキテクチャ屋は勇躍して、新しい時代に向けて研究を進めなければならない。

新しい時代の到来に歩調を合わせ、日本の計算機アーキテクチャ、ハードウェアの研究者も転換期を迎えている。

メインフレームの発展が限界に達すると感じられた時期から、日本のアーキテクチャ研究は主として並列アーキテクチャをベースに展開してきた。並列処理シンポジウム JSPP や並列分散処理に関するワークショップ SWoPP など、並列アーキテクチャ技術を中心に、ソフトウェアからチップ実装まで、広い範囲で研究交流が行われ、世界に誇ることのできるマシンの開発も行われた。

しかし、まず、コンピュータ自体が、コンピュータとして計算や事務に用いられるだけでなく、情報通信機器の様々な範囲に拡散して行った。本当にコンピュータとして用いるアーキテクチャは Intel の 86 系が支配し、この土俵では逆転することが不可能となった。したがって、自らの提案の現実化を目指すコンピュータアーキテクチャ研究者は、様々な応用分野に拡散せざるを得なくなった。並列分散処理技術も、チップ内からインターネットを利用した世界レベルの分散処理技術まで様々な範囲に拡散した。

結果として、並列アーキテクチャを中心に、様々な技術をまとめることは困難になりつつある。現在、アーキテクチャ、ハードウェアの研究者はその数を減らしつつ、一部は日本で開発することが可能なオリジナルアーキテクチャであるスーパコンピュータ方面に向い、残りはチップ設計実装方面に向っている。

筆者は、このような現状を残念とは思ってはおらず、むしろ歓迎し、加速したいと思っている。スーパコンピューティング方面に行かなかったコンピュータアーキテクトは、シリコンチップ上で、チップ設計技術を押えた上で、チップ屋さんと手を結んで、生きていくしかないと考える。今まで培ってきた並列処理技術は、新しい時代を迎えるに当って、非常に有効な技術基盤となるだろう。また、通信、画像信号処理、リアルタイム処理、バイオインフォマティクスなど様々な応用分野に出ていき、そこの研究者と手を携えて新しいアーキテクチャを探るべきであろう。

日本においてコンピュータアーキテクチャが一つの独立した研究領域を形成した時代は終わりつつある。2002 年 2 月の情報処理の特集「知られざる計算機」[5]は、日本のアーキテクチャ研究の終幕を飾る点で実にタイムリであり、また、象徴的である。

参考文献

[1] 日本電子機械工業会 編 “IC ガイドブック” 2000 年版

[2] S.Nishimura, et.al. “RHiNET-2/SW using parallel optical interconnection,”J. of lightwave technology, Vol.18, No.12, Dec. 2000.

[3] K.Watanabe, J.Yamanoto, J.Tsuchiya, N.Tanabe, H.Nishi, T.Kudoh, H.Amano, “Preliminary Evaluation of Martini: a Novel Interface Controller Chip for Cluster-based Parallel Processing,” Proc. of AI 2002.

[4] H.T.Kung, “Why Systolic Architecture? ” IEEE Computer, Vol.15, No.1, pp.37-46, 1982.

[5] 和田英一編 “知られざる計算機,” 情報処理 Vo.43, No.2 2002.

3.2.2　メガスケールコンピューティングの構想

中島　浩　委員

1.　はじめに

著者らは，科学技術振興事業団の戦略的基礎研究推進事業の研究領域「情報社会を支える新しい高性能情報処理技術」の研究プロジェクトとして、「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」を進めている。本稿では、この研究プロジェクトの構想を示す。

2.　概要

ゲノム情報／生命工学などのライフサイエンス、環境・気象シミュレーションなどの複雑な物理系が絡み合う系、大規模な都市における地震や山火事などの災害シミュレーション、などでは、1 ペタフロップス以上の計算能力が期待されている。その半面、従来のMPP／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、米国の ASCI プロジェクトによる MPP やわが国の地球シミュレータは 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、電力も数十メガワットを消費する。量子コンピュータやバイオコンピュータなどのブレークスルーも期待されるが、汎用性や実現性および信頼性の面で当面現実的とはいえない。

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る。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。本節の著者らの主張は、これらの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。これらの技術は以下に示す、ハードウェア／ソフトウェア協調による低電力化技術、サーバ計算機などのディペンダブル技術、インターネット上の数百万ノードの計算機を統合するグリッド／Peer-to-Peer(P2P)技術、などである。

2.1　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。すなわち高密度実装がもたらす発熱密度の上昇、高性能計算に不可欠なプロセッサ高速化による発熱量の上昇、およびメガスケールという規模の拡大の３つの相乗を打ち消し、現実的な冷却／給電を可能とする必要がある。

この問題を解決して、超低消費電力かつ超高性能で超高密度実装可能なコンピュータシステムを開発するためには、プロセッサ内部からオフチップメモリやネットワークに至るアーキテクチャの見直しが必要である。特に重要なポイントは、プロセッサ、メモリ、ネットワークの相互データ転送路が消費電力・性能・実装密度の全てを律していることであり、相互データ転送の量や頻度を削減することが問題解決の唯一の方策である。

そこで本プロジェクトでは、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目指した研究を行う。まずハードウェア側では、ソフトウェア制御可能なメモリ階層など、データ転送削減に有益でかつソフトウェアに可視のハードウェア機構について研究する（研究項目１）。一方ソフトウェア側では、消費電力を最適化の重点項目としたコンパイラについて、ハードウェアが提供する新たな機構の最適な活用を中心とした研究を行う（研究項目２）。

2.2　ディペンダビリティ技術

メガスケールのシステムは、インターネット並の故障率に対応しつつ、それを遥かに上回る信頼性を確保しなければならない。一方現状のクラスタ計算機では、特にネットワークの耐故障性の低さが信頼性の隘路となっている。また均質なノードを一括して組み上げるという構築手法や、それを前提としたソフトウェア基盤も、メガスケールシステムでの部分的故障やノード（群）の追加・更新に対応することができない。

そこでシステムの耐故障性を飛躍的に向上させるために、ネットワークについてはコモディティ技術であるマルチポート・ネットワークを基盤とする研究を行う（研究項目３）。すなわち各ノードが複数のポートを備えることで冗長性に基づく耐故障性を獲得し、同時に並列効果による高バンド幅も実現することができる。

またネットワークやノードの交換・追加・更新をシステム全体を停止せずに行う Plug-and-Play は、長期的なメガスケールシステムの運用に必須の機能である。この機能を checkpointing などの耐故障技術と組合わせたミドルウェアで実現することにより、OS に依存せず非均質な構成に対応可能なディペンダブルなソフトウェア基盤を構築する（研究項目４）。

2.3　グリッド／P2P 技術

メガスケールという規模は、前述の耐故障性以外にもインターネットやグリッドコンピューティングに相通じる性質を持っている。すなわちノードの非均質性、複数クラスタの統合、大きな通信遅延など、従来の MPP やクラスタとは全く異なる性質がある。したがって、メガスケールのソフトウェア基盤やプログラミングモデルの構築の出発点として、グリッド技術の枠組を利用することが妥当かつ有望である。

ソフトウェア基盤については、前述の耐故障性に加え、性能可搬性とクラスタ連携が重要な課題となる（研究項目４）。前者については、システムの漸進的構築や進化的改良によ

って生じるアーキテクチャや性能が非均質な状況下でも、一定の性能が保証されるように計算を分配する機構を研究する。後者は、複数の大規模クラスタをグリッド上でセキュアかつ高性能に連携させる技術であり、メガレベルのスケーラビリティを得るために必須の要件である。

プログラミングモデルについては、Peer-to-Peer(P2P)の枠組によるタスク並列を基盤として、汎用性の高い大規模計算モデルの構築を目指す（研究項目５）。すなわち、問題を独立性が高く粒度が大きな部分計算へ分割する宣言的な枠組と、部分計算の進行に大きなバラツキを許容する計算統括・制御機構の大規模分散化を中心に研究を進める。

このような新しいコモディティな計算基盤技術に根ざし、（単なる掛け声ではなく本気で）100 万プロセッサ級の汎用のディペンダブルなメガスケールコンピューティングを実現するために、上述の５つの研究項目を設定する。すなわち、コモディティメガスケールクラスタ構築のための基礎技術として；

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（プロセッサ技術）
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（コンパイラ技術）
3) 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（ネットワーク技術）
4) グリッド技術に基づくディペンダブルな大規模コモディティクラスタ構築技術（クラスタ構築技術）
5) メガスケールかつディペンダブルなプログラミングモデル（プログラミング技術）

の研究を行い、さらにこれらを大規模プロトタイプ上に統合的に実現する（図 1)。具体的には、従来の 10 倍以上の高密度実装で、1/10 以下の消費電力を達成する大規模高密度クラスタ計算機を実現する。成果は各研究項目の要素技術を中心に特許化するとともに、パブリックドメインあるいは IP として積極的に流布させ実際のメガスケール計算機の構築基盤技術とする。

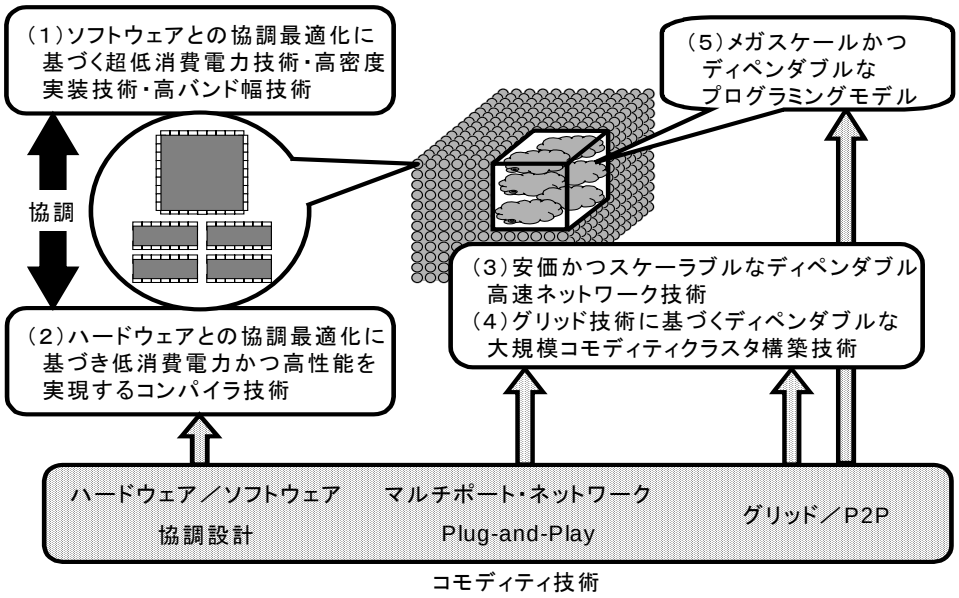


図１　プロジェクトの全体構成

3.　プロセッサ技術（図 2）

ソフトウェア制御可能なメモリ階層をハードウェア的に提供することを中心に、ハードウェアとソフトウェアの協調により、消費電力の多くを占める VLSI 間転送を減らし、高密度実装を妨げる VLSI 間配線を削減しても、豊富な VLSI 内バンド幅を有効に活用することで、システム全体としての高バンド幅を実現するアーキテクチャの実現を目指す研究を行う。

このサブテーマではまず、ソフトウェアから制御可能なメモリ階層を提供するハードウェアアーキテクチャを検討する。特徴は、従来のアーキテクチャとの互換性を維持する点である。本研究は、計算機・情報機器産業界全体へ、そして社会全体へ受け入れられるメガコンピューティングの実現を目指しており、そのため、提案するアーキテクチャは広い応用分野で有効でなくてはならない､と考えるからである｡具体的なアイディアとしては、従来のプロセッサではハードウェア制御下にある名前変換用レジスタ(renaming register)を、動的にソフトウェアによる制御も可能とし、さらに、ハードウェアだけが制御可能であった多階層のキャッシュメモリをアドレス指定可能なメモリ空間としても動的に解放することで、ソフトウェア可制御性を実現する。この動的な再構成自身もソフトウェアからの制御が可能とする。これにより、処理対象の特徴に応じてコンパイラが最適な構成で、名前変換用レジスタとアドレス指定可能なメモリを利用可能となると同時に、従来のアーキテクチャとの互換性も全く失われない。

次に、並行して行われている「コンパイラ技術」と緊密な研究体制をとり、そこで開発されるコンパイラと上記ハードウェアを協調させた場合の、単体のプロセッサの性能面と

消費電力の検討を行う。特に、多階層メモリをソフトウェアからも制御可能とするために増加するハードウェアが､性能と消費電力へ与えるオーバヘッドを定量的に評価しながら、必要となるメモリの階層性とハードウェア機構について検討を加える。

また､「ネットワーク技術」とも緊密な研究体制をとり、そこで開発される高速ネットワーク技術を実現する際の各ノードプロセッサの消費電力、実装密度、実効バンド幅の検討を行うことで、メガスケールコンピューティングを真に実現可能であることを示す。

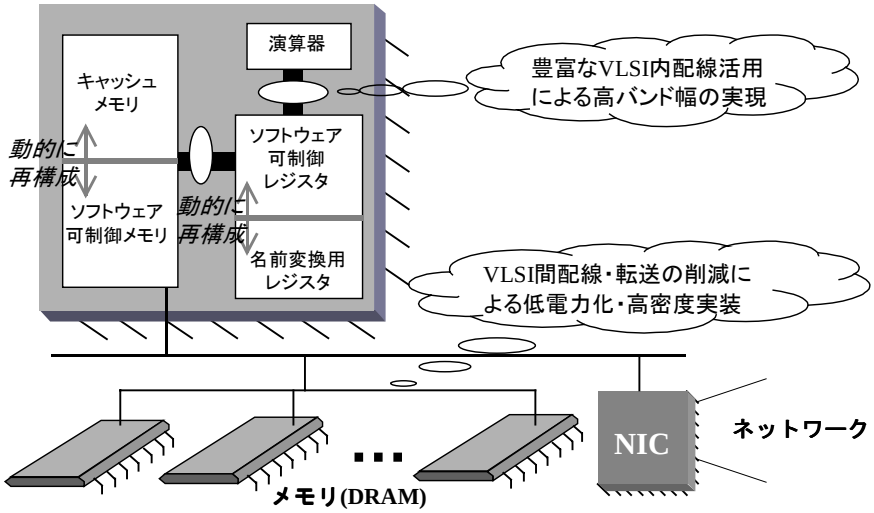


図２　プロセッサ技術

4.　コンパイラ技術（図３）

消費電力を最適化の重点項目とし、ソフトウェア制御可能なメモリ階層を有効活用するためのコンパイラについて研究する。

このサブテーマではまず、設計段階において,「プロセッサ技術」と協調し、最適なアーキテクチャを探るために、アーキテクチャの様々な要素に関する記述に対応する柔軟なコンパイラシステムを開発する。この記述にはレジスタ構成や命令セットの他に、電力消費などのコストもパラメータとして含め、ハードウェア設計の最適化に対応するほか、研究をするコンパイラ最適化アルゴリズムに反映させる。

プロセッサアーキテクチャ設計において、コンパイラとの協調に関し、低消費電力化、高性能化のポイントとして以下を検討する。

(1) プロセッサチップとメモリ間のデータ転送の最適化:消費電力の主要部を占めるのは、オフチップメモリアクセスであり、ソフトウェアから制御可能なメモリ階層を提供できるハードウェアのアーキテクチャを検討し、そのための最適化を行う。この最適化では、命令コードの再配置による命令キャッシュの最適化や、データのオンチップメ

モリ利用最適化を検討する。

(2) レジスタ構成、レジスタ数に関する最適化：レジスタは最も基本的なアーキテクチャの要素であり、与えられたレジスタ構成に関して、spill が少ない割り当てアルゴリズム、ソフトウェアパイプラインなどを検討するほか、最適なレジスタ構成についてもハードウェア設計と協調し、検討する。

(3) 命令コードの選択、スケジューリング：高性能かつ低消費電力となる命令の選択やスケジューリング技法について検討する。

以上の最適化に関しては、静的な解析と同時に実行のプロファイルを収集する機能を実装し、消費電力の最適化を含むプログラムの動的な挙動を反映したコード生成、最適化ができるようにする。また、アルゴリズムレベルにおいて、オンチップメモリを有効に利用するなどの低消費電力化のためのアルゴリズムについても検討を行う。

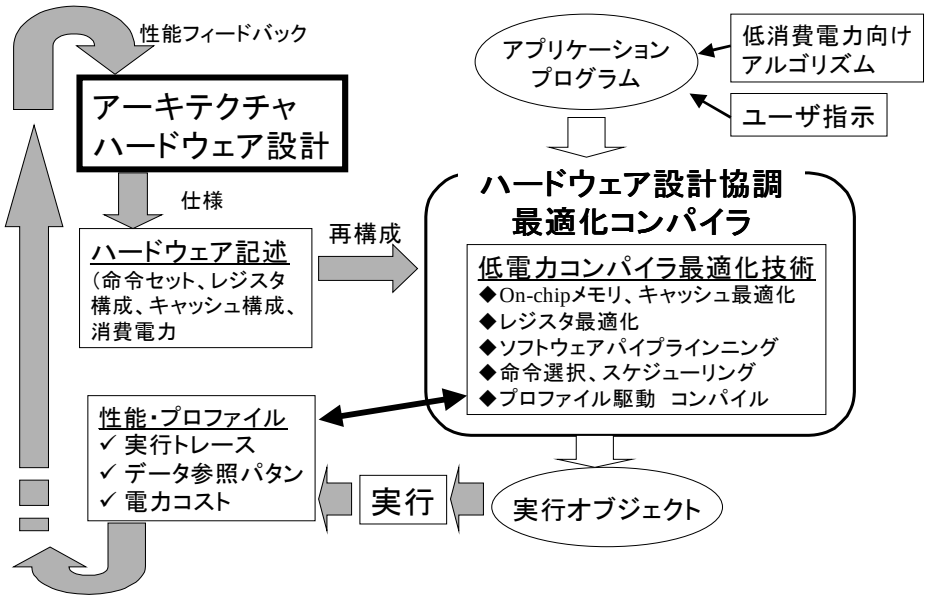


図３　コンパイラ技術

5. ネットワーク技術（図 4）

コモディティネットワークの高い価格性能比を利用したマルチポート・ネットワーク技術を基盤として、耐故障性、並列効果による高バンド幅、Plug-and-Play を実現する新たなメガスケールのネットワークソリューションとして RI2N(Redundant Interconnection with Inexpensive Network)を提案する。

RI2N のベースとして、高密度実装されたマルチポートネットワークインタフェースに対し、ドライバレベルでマルチパス制御・故障チェックを行うシステムを設計・開発する。故障チェックには定期的なテストパケットの交換を用い、一時的なエラーと恒久的なエラ

ーを区別する。通常はマルチパスはユーザから透過なバンド幅倍増モードで用いられるが、故障が検出された場合は残りチャネルを代替パスとして利用しつつ、上位レベルソフトウェアとの協調により故障検出・対応プロンプティングを行う。これらの基本システムを研究前期で設計・開発する。研究後期では、ネットワークインタフェース単位またはノード単位での故障部位の活線交換に関し,「クラスタ構築技術」との連携により一貫した Plug-and-Play システムを実現する。さらに、マルチポート NIC にどのようにトラフィックを割り振るかについて、メッセージパッシング、ソフトウェア DSM 等の用途に応じた最適化を可能とし、最終的にメガスケールシステムに対応したバンド幅とディペンダビリティを提供する。

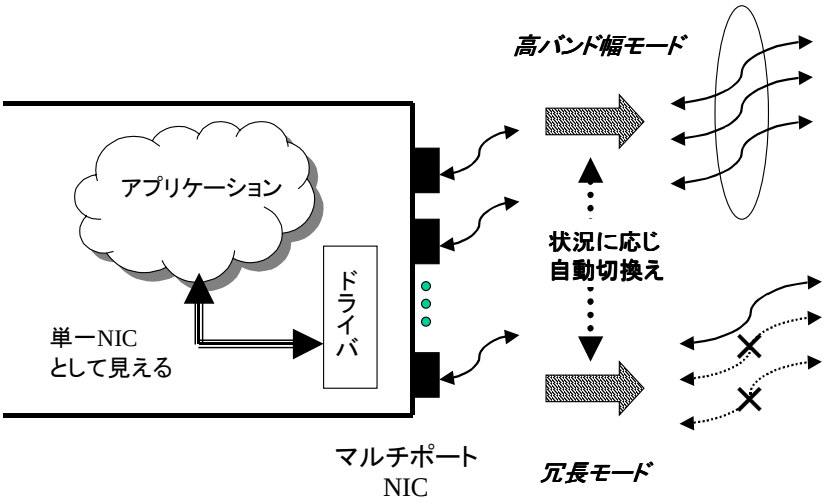


図４　ネットワーク技術

6. クラスタ構築技術（図５）

グリッド技術を基盤とするミドルウェアにより、耐故障性、Plug-and-Play、性能可搬性、クラスタ連携を実現するソフトウェア基盤技術について研究する。

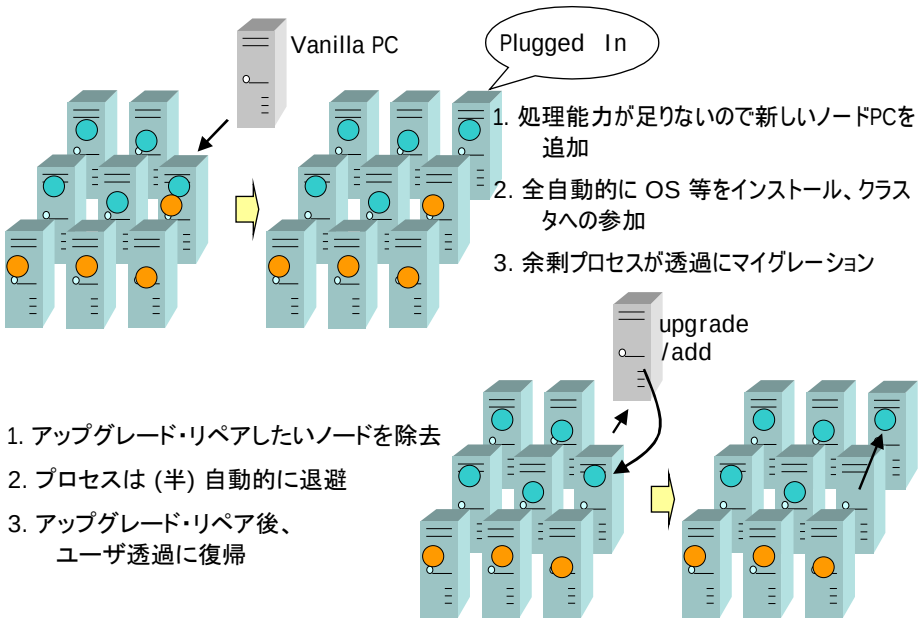
実行環境を限定してしまう特殊 OS に頼らず、グリッド技術によりミドルウェアレベルでディペンダブルなメガスケールクラスタを実現する技術研究を行う。開発されるミドルウェアは、(1) 柔軟でユーザ透過、かつ partial fault に対応する高性能な耐故障性、(2) クラスタの動的な更新を可能にする plug-and-play 性、(3) 異機種実行時の性能の可搬性、(4) グリッド上の複数の大規模クラスタのセキュアな連携(フェデレーション)、の４点を、タスクパラレル、データパラレル、およびそれらの混合的プログラミングのモデルでサポートすることを目指す。

耐故障性：単一ノード上の効率的なユーザレベル checkpointing, 種々の複数ノード checkpointing および logging&recovery アルゴリズム、ディペンダブルな通信ライブラリ、フォルトの検出器、さらに実際のメガスケールクラスタでの実行のシミュレータなどをツールキット上に構成し、様々なフォルトとリカバリモデルを柔軟に実現できるようにする。

特に、検出はモデルベースのシミュレーション実行と常時比較を行い、モデルと大幅にずれた場合は partial fault とする。

(1) Plug-and-Play：(1)の耐故障性を基礎とし、さらに(3)並びに【研究項目３、５】と協調して、松岡らのグリッドの研究成果を用いて動的なノードの追加削除に柔軟に対応する。
(2) 性能可搬性：データパラレル・MPI メッセージパッシングにおいては仮想プロセッサ技法を、データパラレル・マルチスレッド実行では佐藤らの OpenMP コンパイラを用い、プログラム解析並びにランタイムでヘテロ性に対処する。タスクパラレルでは松岡らが研究してきた Ninf 並びに【研究項目５】の実行モデルをスケーラブルに資源並びに問題の複雑さに応じてスケジューリングする。
(3) クラスタ連携：グリッド技術を用いクラスタ間の単一の認証ドメインを実現するとともに、 クラスタを接合してメガスケールまで発展させるためのスケジューリング、高速データ転送、ファイアウォールやプライベートアドレスなどの既存セキュリティインフラへの高効率な対応、などを研究する。

研究の基盤として、初期はすでに松岡研究室に存在する大規模グリッド・クラスタテストベッド(3 台の 200 プロセッサ級クラスタ)を増設してミドルウェア開発に用いるが、大規模高密度クラスタ計算機の開発にも関与し、後期には高密度クラスタプロトタイプ上に研究基盤を移す。



Plug-and-Play (動的なクラスタの再構成)

図５　クラスタ構築技術

7. プログラミング技術（図 6）

Peer-to-Peer(P2P)コンピューティングの枠組を基盤として、問題を独立性が高く粒度が大きな部分計算へ分割する宣言的な枠組と、部分計算の進行に大きなバラツキを許容する計算統括・制御機構の大規模分散化を中心に、メガスケールのプログラミングモデルを構築するための研究を行う。

7.1 問題分割・統合の宣言的記述

このサブテーマでは、スクラッチからの並列プログラミングではなく、parameter survey などのように既設計の逐次・並列プログラムを多数並列に実行するための枠組みを研究する。すなわち既設計の「オブジェクト・プログラム」を入力引数を変化させて繰り返すなどの「メタ・プログラム」を記述する枠組みの構築が目的である。

このメタ・プログラム言語は、オブジェクト・プログラム間の関係だけではなく、オブジェクト・プログラムの挙動を部分的に記述できるように設計する。たとえばオブジェクト・プログラムが以下の関数 foo(i,n)の形に表現できるとする。

```
foo(i,n) {
    Result_Type r = initial_value_of_reduction;
    for (;i<n;i++) {
        r = reduction_function(r, bar(i));
    }
    return(r);
}
```

このとき、メタ・プログラムの基本的な表現は；

```
r = initial_value_of_reduction;
for (i=0;i<N;) {
    n = next_step(i);
    r = reduction_function(r, foo(i,n));
    i = n;
}
```

のようになり、foo を単位とする処理であることが記述される。また *next_step* をシステムに委ねて、自動的な負荷調整を図ることもできる。さらにメタ・プログラムを；

```
r = initial_value_of_reduction;
for (i=0;i<N;i++) {
    r = reduction_function(r, bar(i));
}
```

のように記述し、この構造が foo を利用可能であることをシステムに自動認識させてメタ

レベルの記述性向上を図ることもできる。

なお上記の例では C 風の記述としたが、プログラミング容易性を高くするために Perl などをベースとしたスクリプト言語とし、オプショナルにコンパイルも可能とする。

7.2　予測・投機による計算負荷調整

上記のメタ・プログラムの並列実行は、コンパイル時あるいは実行時の解析に基づいて行われるが、オブジェクト・プログラムの挙動を完全に解析することはできない。そこで、以下の手法を用いて計算量・通信量の予測を行い、その結果に基づき投機的に負荷調整を行う。

- 過去の実行履歴によるプロファイル
- 小規模なテストラン（事前または並行実行）によるプロファイル
- 実行時のプロファイル

7.3　計算統括機構の大規模分散化

メガスケールコンピューティングでは、上記の負荷調整などの計算の統括を集中的に行うのは非現実的である。そこで計算統括機構自体も 10^3〜10^4 スケールに分散し、統括機構のボトルネック化を回避すると同時に負荷調整等の通信遅延耐性を高める。

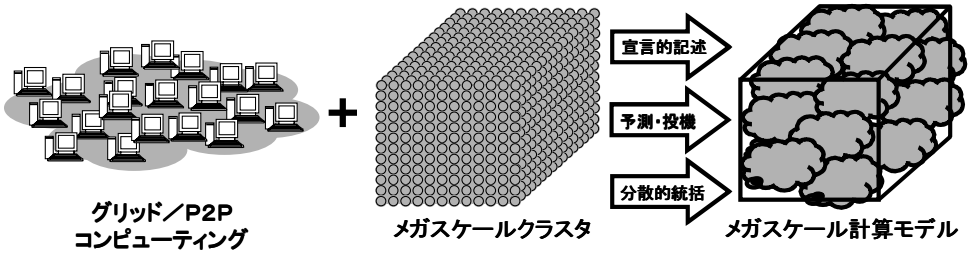


図６　プログラミング技術

3.2.3 PC クラスタの現状と今後

久門　耕一　委員

1. はじめに（PC クラスタと Commodity 部品）

HPC 向の計算機システムとして、PC クラスタが急速に普及し始めている。PC クラスタを構成する要素は、クラスタのノードを構成するための PC と、それらをつなぐためのインタコネクトネットワークである。ノードは一般的な PC であり、CPU、メモリ、ネットワークとの接続点である NIC (Network Interface Card)から構成されている。PC クラスタが急激に普及し始めたのは、PC に用いられている IA(Intel Architecture)プロセッサが、大量生産を武器に、量産効果による価格の低下、開発費の大量投入による高性能化の双方の優位性を兼ね備え、IA プロセッサが本来あまり性能面で重視してこなかった浮動小数演算を多用する HPC の分野で使用しても、満足出来る性能を得ることが出来るようになってきたためである。逆に、IA プロセッサはビジネス計算分野、つまり整数系演算では十分高速である。IA プロセッサの命令セットは RISC でなく CISC であるため、RISC がコンパイラとアーキテクチャの双方の最適化技術で高速化を達成したのに比べ、IA アーキテクチャではコンパイラの性能に大きくは依存していなかった。このことは、高性能化を目指すために IA プロセッサがその実装方式(マイクロアーキテクチャ)を変更してクロック周波数を向上させてきた場合でも、最適化コンパイルの手法に変更を必要とせず、結果的にほぼ予想された性能向上が得られる原因ともなってきた。

一方、PC クラスタ内を接続するインタコネクトネットワークに関しても、ネットワークの高速化技術により Commodity ネットワークはすでに Giga bit Ether(GbE) ネットが普及品価格となっており、数年後には 10 Giga bit Ether (10GbE) が簡単に入手できるだろうと推定される。

現在まで、このように汎用の Commodity を用いた PC クラスタが順調に性能を向上させてきた。しかし、最近の急速な CPU あるいはネットワークの性能向上を達成するために、Commodity が従来のように「汎用」ではなく、特別な目的のために専用化される傾向が現れてきた。以下に、CPU、ネットワークの順番で現在と今後の動きを評価する。

2. 汎用プロセッサとは言えなくなる IA プロセッサ

CPU による実行時間は、

プログラムの動的命令数 ＊ 1 命令実行の平均クロック数 ＊ クロック周期

で表され、後者の二つはそれぞれ CPI(Clocks per Instructions)、τ (=クロック周波数（f）の逆数)と表される[1]。

命令セットが固定されると、プログラムの動的実行命令数は変わることがなくなり、プログラムの実行時間は、CPI と τ(=1/f) だけにより決まる。プロセッサの性能を向上させるため、クロック周期を短くする典型的な手法は、

- プロセスの微細化によるトランジスタの高速化
- パイプラインピッチの縮小

である。

プロセスの微細化によるトランジスタの高速化は、プロセスの縮小と共にチップサイズを縮小しない場合には、プロセスの微細化にともなう配線抵抗の増大により高速化が得られないことが知られている。これが DSM(Deep Sub-Micron)問題として知られている。

一方の、パイプラインピッチの縮小状況を、Intel の Pentium から Pentium4 までの違いで見てみる。Intel の P6 アーキテクチャ(Pentium-PRO から Pentium-III まで）で用いられているパイプラインは、およそ 13 段で、P6 以前の Pentium での 5-6 段のパイプラインに比べほぼ倍に段数が多くなっている[2]。さらに、P6 の次の世代である Pentium4 アーキテクチャでは、パイプライン段数は 22 段といわれ、単純に考えると Pentium4 プロセッサは Pentium3 プロセッサのクロック周波数をほぼ 2 倍近くにできることがわかる。

クロック周波数の伸びの状況は、95 年に Pentium がほぼ 100MHz で動作し、2000 年に Pentium4 が 1GHz を越えたことを考えると 5 年間に 1 桁以上 CPU コア周波数が高速化してきたことになる。5 年で 10 倍(年率 1.58 倍)の周波数向上は、トランジスタ数の伸びを予測するムーアの法則の 3 年で 4 倍(年率 1.59 倍)と同じ伸び率となっている。

これに比べ、主記憶となる DRAM アクセス時間は、制御を行なうコントローラ込みで見たアクセスレイテンシは、95 年当時すでに 300ns 以下であったし、現在でも 100ns 以上かかることから、高々2-3 倍程度しか高速化されていない。一方、DRAM のデータスループットは、同期 DRAM によるパイプライン処理、DRAM 内のマルチバンク化、さらに DRAM アクセスインタフェースへの高速伝送路採用などで飛躍的に向上し、RAMBUS DRAM では DRAM チップあたりのバンド幅は 1GB を越えている。この結果 5 年間に 1 桁以上の DRAM スループット向上が得られている。

このように、CPU コア周波数の向上と、DRAM のスループット向上はほぼ歩調を合わせているが、DRAM のアクセスレイテンシはこれに追い付いていない。このため、システム全体の性能を向上させるには、DRAM のスループットの伸びを使いレイテンシを隠蔽する技術を採用せざるを得ない。Pentium4 では、ハードウェアが CPU のメモリアクセスパタンを監視してプリフェッチを自動で行なうハードウェアプリフェッチ機構が導入され、この監視は同時に 8 箇所のストリームへのアクセスをプリフェッチ対象とする。実測によると、1.7GHz の Pentium4 と RDRAM 制御を行なう i860 チップセットで作成されたシステムは、メモリの連続読み出しで 2GB/s 以上のメモリスループットが得られている。このように、製造技術の向上にともない、順調にシステム性能が向上してきているよ

うに見える。図 1，2 に SPEC CPU2000 をベンチマークとして整数/浮動小数性能を表した。

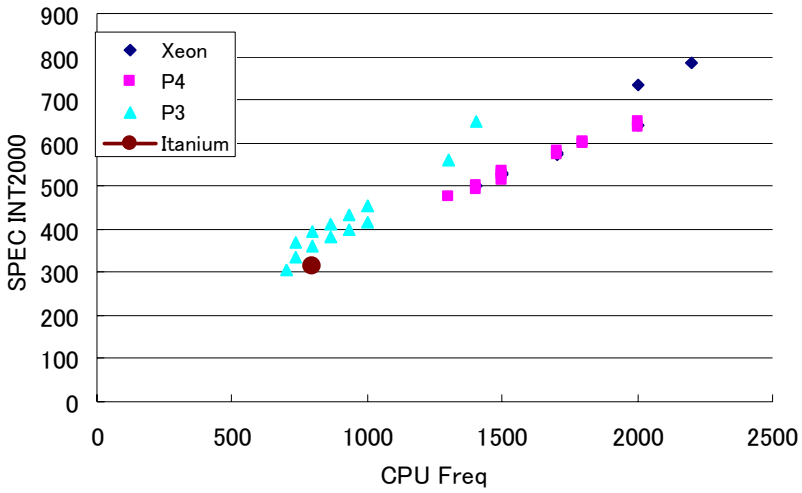


図 1　各種 IA プロセッサの SPEC CPU 2000 の整数演算性能

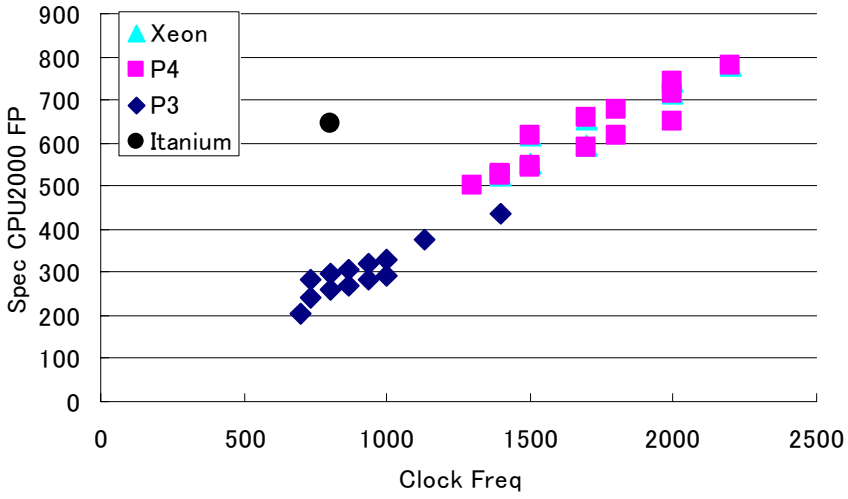


図 2　各種 IA プロセッサの SPEC CPU2000 の浮動小数演算性能

図 2 は IA32/IA64(Itanium)を用いたシステムの整数演算性能、図 3 は浮動小数演算性能を示す。双方のグラフとも、横軸は CPU のクロック周波数、縦軸は SPEC 値である。

この結果から、IA32 の CPU が Pentium3 系列から Pentium4 系列に移行した際の性能の伸びが素直でないことがわかる。一般に、同じアーキテクチャでクロック周波数だけを向上させる場合、性能は向上する。図 1 を見ると、SPEC の整数系ベンチマークでは

Pentium3 系列から Pentium4 系列に移行した際、同一のクロック周波数(例えば 1400MHz 近辺)で見比べると、Pentium4 の速度が 10-20%遅いことがわかる。つまり、アーキテクチャを変更したことにより遅くなっている。一方、図 2 を見ると SPEC の浮動小数系ベンチマークでは逆に Pentium4 の方が 10-20%高速になっていることがわかる。さらに、次世代の IA プロセッサの 64 ビットアーキテクチャ CPU である Itanium は、図 1 と図 2 で際だった違いを見せている。整数性能では同一クロックの IA32 に及ばない性能だが、逆に浮動小数性能では 2 倍以上高速である。図 3 のグラフでは Pentium3 と Pentium4 の双方の系列の性能が 2 段に分かれて見えるが、キャッシュサイズの違いでの性能差である。

SPEC のようにコンパイラや CPU 設計者が最適化対象の中心におくプログラムではなく、プログラムとしては浮動小数演算が主となる一般のエンジニアリング系のソフトウェアを実行した結果で評価したものが図 3 である。

このプログラムは、大きく分けて、入力データの前処理を行なう部分(pre)と、大規模な行列演算を行なう中間部分(MKL)、演算結果をユーザに表示するため、グラフィカル処理を行なうためのデータ作成処理などの処理(post)の 3 つに分かれている。このグラフから、Itanium では中間の処理（MKL）がとても速くなっているが、全体を通しての実行時間は、同一クロックの IA32 よりも実行時間が長い。実は、中間部の処理のほとんどは、インテル社が提供している行列演算ライブラリの実行時間であり、ユーザが書いたコードが実行されているわけではない。このような実行時間の分布は、ベクトル計算機でよく見られる。すなわち、専用に作られたライブラリや、特定の計算方法(規則正しい配列アクセスなど)の場合にとても高速に実行するが、そうでない場合には性能が出ていない。VLIW 構造をとる Itanium のコンパイラ依存の高速化が機能していないことを示している。

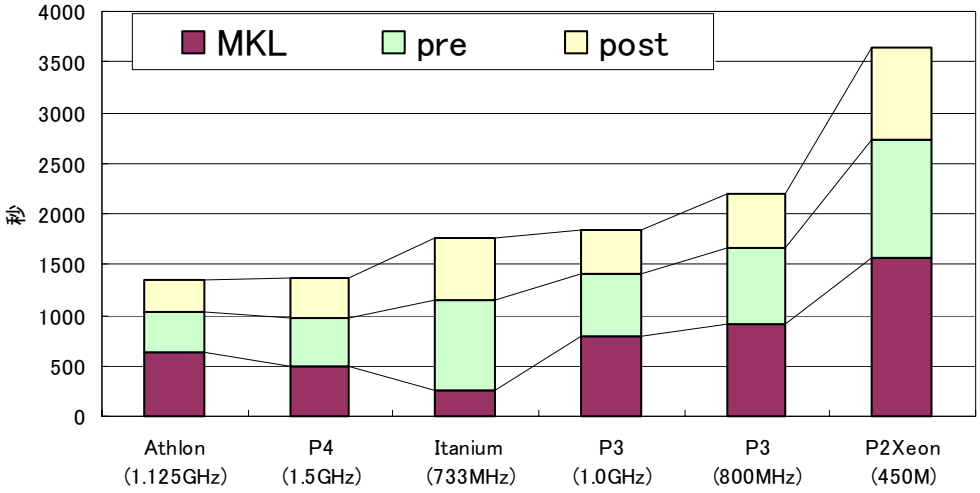


図３　エンジニアリング系アプリケーションでの実行時間

従来汎用プロセッサであった IA プロセッサが高速化を図る過程で特定の目的だけが高速化される結果になったことには、現在高速化を特に必要とするアプリケーションの主なものがマルチメディア処理、たとえばビデオストリーム処理、動画圧縮など HPC 的な動作を要求するものであることが理由として考えられる。このことは PC クラスタが HPC 向けに使用される場合には有効に働くと推定される。一方、SPEC INT のような挙動を示すプログラムでは、今後の性能の伸びは、クロック周波数の向上ほどではないことも同時にわかる。

3. インタコネクトとして使いにくい Commodity ネットワーク

汎用のネットワークとして Ethernet が広く使われており、クラスタ内部のインタコネクトとしても 100Mbps Ethernet から 1Gbps Ethernet が使われている。クラスタ内の接続用の高性能ネットワークには、当初から Myrinet に代表される proprietary なネットワークが用いられてきた。しかし Ethernet が 1Gbps の速度に達したため、Myrinet は高々 2 倍程度スループットが高いだけに過ぎず、ネットワークも Commodity 使用に移行する傾向がある。そこで、クラスタシステムとして SCore を用い MPI 上で Ping-Pong 性能を測定した結果を図 4 に示す。横軸は 1 回の Ping-Pong 転送のデータサイズ、縦軸はデータサイズを 1 往復する時間の半分で割ったもので、スループットである。Ethernet のネットワークインタフェースカード(NIC)としては、100Mbps 用に RTL8139 と言う廉価なカード、1Gbps 用には Intel 社の EEPRO1000 を用い、送受ノードの間はスイッチを経由せず直接ケーブル接続している。

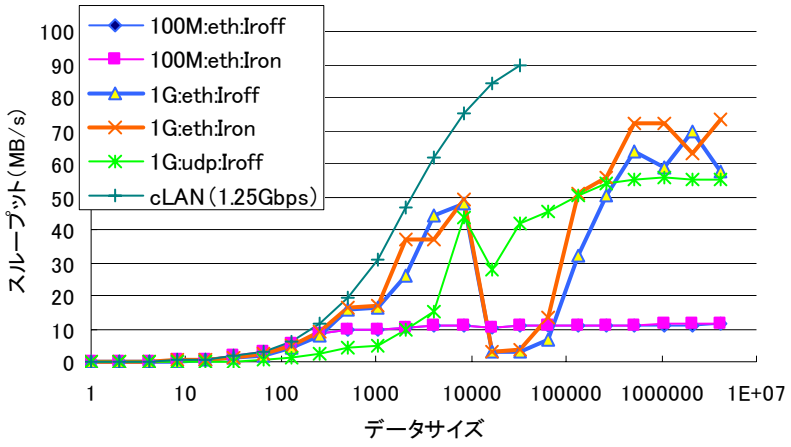


図４　SCore により計ったネットワークの Ping-Pong 性能

データサイズが小さいときは、転送を起動するためのオーバヘッドの比率が大きいためスループットが得られないが、データサイズが大きくなるにつれ、ネットワークのスループット性能による限界へと近づいてゆく。クラスタインタコネクトとして設計された cLan や、100Mbps の Ethernet は素直な性能であることがわかる。一方、1Gbps の Ethernet は、10Kbyte のデータサイズ近辺で急激に性能が劣化し、100Mbps での性能よりも劣る場合があることがわかる。これは、標準 Ethernet のパケットサイズは 1.5KB であるが、1Gbps のスループットを有効に活用するためより大きなパケットである 9KB のパケットに切り替わった時に、レイテンシが大幅に悪化したことが主因である。EEPRO10000 では、1Gbps のネットワーク処理を行なう時に NIC からの頻繁な割り込みによる CPU 負荷の増大を防ぐため、パケットがひとつ到着してもすぐには割り込みを出さず、複数のパケット処理を 1 回の割り込みで処理する Interrupt Coalescing と言う最適化を行っている。ストリームのように次々とデータが到着する場合には、この最適化は有効であるが、パケットの Ping-Pong のように 1 回データが来るとそのデータを処理しない限り後続のパケットが来ない場合には、パケット処理開始のレイテンシが大きくなってしまい、図のようにスループットの劣化を引き起こす。通常、TCP/IP の処理を行なう場合、100Mbps のイーサネットを処理するには、概算で 100MHz の CPU を 100%使うことが必要と言われている。筆者が実際に測定した結果でも、この値はほぼ裏付けられている。これに対する対策のひとつが、複数パケットの一括処理である。このように、クラスタシステムを構成するためには単にインタコネクトネットワークの伝送速度が高速になるのでは不十分で、送信元ノードのメモリ空間にあるデータを、受信先ノードのメモリ空間に高速に転送することが必要である。このためには、ホストメモリのデータをインタコネクトネットワークに高速に送付する処理、とインタコネクトネットワークから送られてくるデータを指定されたホストメモリに高速に格納する処理が必要で、従来の NIC では CPU に対する負荷を減らさずクラスタインタコネクトに必要な低レイテンシの達成はきわめて難しい。

CPU の性能向上がいわゆるムーアの法則に乗って 18 ヶ月で 2 倍の性能向上で推移してきたのに対し、ネットワークバンド幅の向上は、いわゆるギルダーの法則に従い、9 ヶ月～16 か月で 2 倍の向上を達成し、CPU の性能向上率を上回っている[3]。このため、今後ますますインタコネクトネットワークの有効利用が困難になって行くと予想される。一方、Myrinet や cLan のようにクラスタインタコネクトを目的として作成されたインタコネクトシステムはパケット処理を NIC 上のハードウェアにオフロードしているため、ホスト処理を重くせずレイテンシの短縮を図ることが可能である。しかし、Commodity ネットワークが採用する TCP/IP プロトコルの標準 API である socket インタフェースでは OS と NIC の役割を分けることが難しくハードウェアオフロードが困難である。socket インタフェースを用いない TCP/IP 実装として用途は異なるが iSCSI が実用化され始めており、クラスタインタコネクトにも socket インタフェースに依存しない新規の API、たとえば

RDMA インタフェースを用意し、NIC 上にプロトコル処理のオフロードを行わなければネットワーク性能を十分に使用することが出来ないと考えられる。

4. 今後の PC クラスタと技術開発について

以上説明してきたとおり、従来 PC クラスタは Commodity を組み合わせることにより低価格高性能を達成してきた。しかしムーアの法則やギルダーの法則に従い CPU やネットワークが高速化してゆくと、Commodity が現在のように「速い安い」を兼ね備え続けられる事は必ずしも自明ではない。しかし HPC 専用に高性能なシステムを開発することは、従来のハイパフォーマンスコンピューティング市場が衰退してきたことから私企業に求めることは無理がある。そういう意味では、地球シミュレータのような大規模な HPC システムを国家戦略として実現することに意味があるとも言える。しかし、ごく限られた研究者だけが高性能な HPC システムを使う状況では社会へのインパクトが限られる。現時点での選択肢は 2 つある。

一つは限定された場所にある高性能な HPC システムをどこからでも使用可能にする枠組みの作成、すなわち Grid Computing への道。もう一つは、高性能な HPC システムを大半を Commodity 部品を使用し、少数の Commodity 流用では賄えない部分を最小限開発する技術開発に投資することである。特にクラスタインタコネクトには、低レイテンシと低ホスト負荷が必要なため、ネットワークとホスト計算機との接続を担う NIC 部分は、技術開発が必要な第一候補であると考える。

参考文献

[1] “Computer Architecture: A Quantitative Approach” 2nd Edition, J.L.Hennessy, D.A. Patterson, Morgan Kaufman Publishers, Inc. 1996

[2] “Pentium Pro Family Developer’s Manual”, Intel Corporation, 1996

[3] “Unleashing the POWER of Networks” http://www.johnsoncontrols.com/Metasys/articles/article7.htm

3.3　基本ソフトウェア＆ミドルウェア

3.3.1 手続き間解析の動向　～コンパイラとその周辺～

佐藤　真琴　委員

1.　はじめに

手続き間解析とは、手続き（関数、サブルーチン）の解析情報を、その手続きの呼び出し点で利用できるようにするコンパイル技術のことである。これによってプログラムの解析精度が向上し、プログラムを最適化する機会が増加することが期待できる。広い意味では、インライン展開も手続き間解析の一つと考えられるが、リカーシブな呼び出しに対応できないなど、適用範囲が狭い点が課題である。これに対して、狭義の手続き間解析はその一般性から、Fortran のみならず C++や Java に対する最適化コンパイラでも徐々に採用されつつあるなど、広がりをみせている。

本稿では、Fortran コンパイラにおける手続き間解析、及び、ツールにおける手続き間解析の利用に関する動向を、具体的な事例を通して紹介する。

2.　手続き間自動並列化コンパイラ WPP

我々は、手続き間自動並列化コンパイラ WPP（Whole Program Parallelizer）[1][2][4]を開発している。これは、逐次 Fortran プログラムを入力し、手続き間解析・並列化およびループ変換を施し、その結果を OpenMP[18]プログラムとして生成する。本章では WPP の手続き間解析・並列化について述べる。

2.1　構成

WPP は以下の 4 フェーズから構成される。

(1)手続き間フレームワーク
(2)手続き間スカラ解析
(3)手続き間配列解析
(4)手続き間並列化

(1)は全体制御、コールグラフを含む階層的制御フローサマリグラフ（Control Flow Summary。以下、CFS グラフと呼ぶ。）やクローン手続きの作成と管理等を行なう。(2)はスカラ変数に対する手続き間データフロー解析、手続きクローニング、及び手続き間定数伝播を行なう。(3)は配列に対する手続き間データ依存解析を行なう。(4)は上記解析結果およびリダクションなどの独自解析の結果から、手続き呼び出しを含むループを並列化する。

(2)と(3)を別々に、しかも、(2)を先に行なうのは、手続き間定数伝播により配列添字やループの上下限値が定数化した場合、配列解析の精度向上が期待できるためである。

(2)から(4)までの 3 つのフェーズの各々は、さらに以下の 3 つの小フェーズに分かれる。これらの小フェーズ構成により手続き間解析が実現できる。

(A)手続き内前解析
(B)手続き間伝播
(C)手続き内後解析

(A)は、各手続きを解析し、必要なら解析結果を要約してテーブルに保持する。この時は、各手続きの中間語はメモリ上に存在し、それを用いて解析を行なう。この時点では手続き呼び出し先の解析情報がないので、手続き呼び出し点に対する解析結果は一般に、未定等の扱いとなる。但し、(A)をコールグラフを逆 DFN 順にたどって実施すれば、ボトムアップに得られる解析情報に関しては手続き呼び出し点でその解析結果が利用可能となる。

(B)は、コールグラフや CFS グラフをたどりながら、(A)で得られたテーブルのマージ等を行ないながら、各手続きとそこから呼び出される全ての手続き呼び出しの効果を総合した解析情報等を得る。この解析では、各手続きの中間語等は用いず、上記２つのグラフと(A)で得られたテーブルを使って処理を行なう。これにより、(A)と（(C)では従来のコンパイラにおける中間語を処理するためのインターフェースを変えずに、(B)ではメモリ消費量を少なくして手続き間解析が実現できる。

(C)は、(B)によって得られた各手続きの解析結果を手続き呼び出し点での解析結果に翻訳し、この翻訳結果と各手続きの中間語等を使って、呼び出し先手続きにおける効果を考慮した解析を行なう。

以上の 3 つの小フェーズでの解析方法は、解析内容によって多少の違いがある。注意すべきなのは、従来のコンパイラにおける解析方法は以上の 3 つに分割にする必要があるため、その解析アルゴリズムはかなり変更される、という点である。例えば、全ての手続きに対する中間語が同時に存在することを前提とした解析アルゴリズムは使えない。

2.2　手続き間フレームワーク

手続き間フレームワークは、手続き間解析全体の処理の制御、コールグラフを含む CFS グラフやクローン手続きの作成と管理等を行なう。

コールグラフとは、手続きをノード、呼び出す手続きと呼び出される手続きに対応するノード同士を有向エッジで接続した有向グラフである。我々の方法では、１つの手続きを表わすノードはコールグラフ上でただ１つである。リカーシブな呼び出しがある場合は、コールグラフ中にループができる。

CFS グラフとは、ループ、基本ブロック、手続き呼び出し点を１つのノードとする制御フローグラフでである。ループボディはそのループノードを親ノードとする下層のサブグラフとなる。したがって、ｎ重ループは(n+1)階層のグラフとなる。この CFS グラフをコールグラフと接続することによって、手続き全体のデータフロー解析が可能となる。
手続きクローニングでは、ある手続きの情報をコピーしてクローン手続きを作成する。その際、コールグラフにおいてクローン手続きに対応するノードの追加や、対応する CFS グラフの作成、及びコールグラフとの接続が必要となる。

2.3　手続き間スカラ変数解析

手続き間スカラ変数解析はスカラ変数に対する手続き間データフロー解析を行ない、各手続きに対して参照集合を決定する。各集合は、各手続きにおける変数の定義・使用情報を要約したものである（表 1)。これらの集合うち、最初の２つは制御フローに関係しない情報（flow-insensitive）であり、後の 3 つは制御フローに関係する（flow-sensitive）情報である。

手続き間スカラ変数解析はまた、手続きクローニングと手続き間定数伝播も行なう。１つの手続きに対する異なる呼び出し点で、同じ引数に異なる定数値が伝播された場合、このままでは呼び出し先手続きに各々の定数値を伝播できない。これを可能にするために、手続きクローニングを適用する。これによって、上記の場合でも呼び出し先手続きに定数伝播ができる。この結果、次に行なう手続き間配列解析の精度向上が見込める。図 1 はこの最適化の適用例である。手続き f に手続きクローニングが適用され、その手続きの引数 m に定数伝播が適用される。

表１：スカラ変数に対する参照集合

参照集合	意味
MOD	定義の可能性のある変数
USE	使用の可能性のある変数
KILL	確実に定義される変数
EUSE	定義前に使用の可能性の有る変数
LIVE	ループや手続きの最後でライブな変数

2.4　手続き間配列解析

手続き間配列解析は配列に対して参照リージョンと呼ばれる集合を解析して、手続き間データ依存解析を行なう。ここで、ある配列に対する参照リージョンとは、あるプログラム単位内でのその配列の参照情報を要約した配列添字の集合である（表 2）。

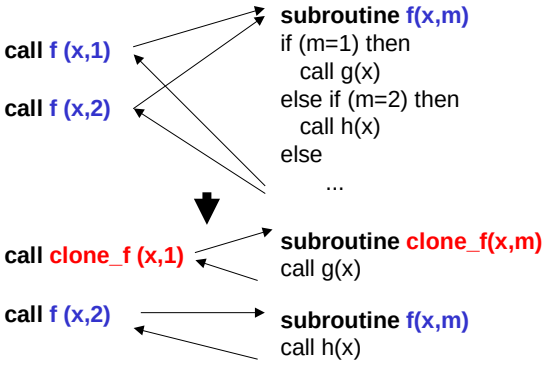


図 1：手続き間定数伝播の適用例

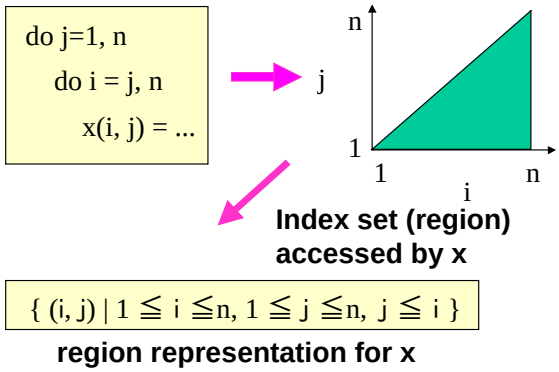


図 2：連立 1 次不等式表現の例

表 2：配列に対する参照リージョン

参照リージョン	意味
MOD_A	定義の可能性のある添字範囲
USE_A	使用の可能性のある添字範囲
$KILL_A$	確実に定義される添字範囲
$EUSE_A$	定義前に使用の可能性のある添字範囲
$LIVE_A$	ループや手続きの最後でライブな添字範囲

参照リージョンのコンパイラ内部表現には矩形表現と連立 1 次不等式表現を用いている。矩形表現は各次元毎の区間表現の直積である。連立 1 次不等式表現は各次元の添字の 1 次結合から成る連立不等式である。添字が単純な場合は矩形表現を用いて高速に処理を行ない、添字が複雑な場合は連立 1 次不等式表現を用い、その計算にはオメガテスト[19]と同様な手法を用いる。後者はストライドが 2 以上の場合や三角ループにおける多次元配列の参照領域を表現でき、また、実引数と仮引数の配列次元が異なる場合にも対応可能である。図 2 は三角ループの例である。プログラム中に異なる参照パターンが多く出現する場合、参照リージョンは各々のパターンの和集合として表現される。この数がある閾値を越えた場合は、和集合を近似する。近似された集合はより大きい 1 つの集合への近似（拡大近似）とより小さい 1 つの集合への近似（縮小近似。空集合の場合もある）のペアとして表現する。尚、参照リージョンの表現に現われる式は定数だけでなく、変数を含むことも許している。

2.5　手続き間並列化

手続き間並列化は上記解析結果およびリダクションなどの独自解析の結果から、手続き呼び出しを含むループを並列化する。以下では、DOALL 並列化のみ説明する。

特別な変換なしにループを並列化できる条件は、ループ運搬のフロー依存、逆依存、及び出力依存がないことで与えられる。スカラ変数ではこれらの条件を一つひとつチェックするのではなく、スカラ変数への定義があったらループ運搬依存があると判定している。即ち、

$$MOD \neq \phi \qquad (1)$$

なら、そのループは並列化不能である（インダクション変数は除く）。配列では、上記の各ループ運搬依存をチェックし、一つでも依存が存在すればループは並列化不能であると判定している。以下に参照リージョンを用いたループ運搬フロー依存のチェック方法を説明する。

まず、各参照リージョンに対して、解析対象ループに関する以下の 3 種類の参照リージョンを計算する。1 つ目はループ繰り返し i 回目。但し、i は一般の値を取る変数を表わす。２つ目はループ繰り返し 1 回目から(i-1)回目までの和集合。3 つ目は全ループ繰り返しに対する和集合である。

これらを用いると、ループ運搬フロー依存が存在することのチェック方法は以下となる。

$$MOD_A [1:i-1] \cap EUSE_A [i:i] \neq \phi \qquad (2)$$

式(2)がループ運搬フロー依存を示すのは以下のように説明できる。まず、次の式(3)は依存距離が 1 のループ運搬フロー依存の存在を示すことは定義から容易にわかる。

$$MOD_A [i-1:i-1] \cap EUSE_A [i:i] \neq \phi \qquad (3)$$

したがって、式(2)は依存距離が 1 から i -1 までの範囲のいずれかのループ運搬フロー依存があることを示す。i は一般の値を取る変数なので、式(2)は結局、いずれかの依存距離を持つループ運搬フロー依存の存在を示す。

次に、変換を伴う並列化方法について以下に概略を説明する。

(1) 手続き間変数プライベート化：ループの各繰返しで変数の使用の前に必ず定義がある時、この変数への参照を各スレッド固有の変数への参照に変更することで、ループ運搬依存をなくしてループが並列化できる。これが可能な条件はスカラ変数に対しては $EUSE = \phi$、配列に対しては $EUSE_A [i]= \phi$ である。
(2) 保証コード生成：プライベート化された変数・配列に対して、それらの初期値設定（初期値保証）、並列ループ終了時の変数値の設定（終値保証および条件付終値保証）

を行う。初期値保証は、例えば、$EUSE_A$ [i]={1}, MOD_A [i] ∩{1} = ϕ の時、ループ直前の A(1)の値を各スレッドのプライベート配列の A(1)に設定する。終値保証は、変数の定義が条件文下になければ、最終ループ繰返しにおけるプライベート変数の値をループ終了時の変数の値として設定する。条件付終値保証は、変数の定義が条件文下にある場合、定義が発生した最後のループ繰返しを記録しておき、その繰返しを実行したスレッドの値をループ終了時の変数の値として設定する。ループが手続き呼出しを含む場合は、ループから呼び出される全ての定義が条件文下にあるか否かをチェックし、そのような定義に対してはある大域変数に定義の発生を記録して、並列ループ終了直前に定義が発生した最後のループ繰返しを検出する。

(3) リダクション並列化 : SUM, PRODUCT, MAX/MIN, MAXLOC/MINLOC などのリダクションパターンを検出してリダクション固有処理に変換した後、ループを並列化する。リダクションパターンの検出は、ループから呼び出される各手続きで各変数毎に、リダクションパターンを持つか否か、リダクション以外のパターンを持つか否か、リダクション演算の種類を解析し、ある変数がループ中でリダクションパターンのみを持ち、リダクション演算の種類が唯一である場合にリダクション並列化を適用する。

2.6 性能評価

SPECfp95/turb3d に対して WPP が生成する OpenMP プログラムの性能評価を行なった。評価マシンは、SGI™ 社の Origin™ 2000 である。Origin 2000 は、2CPU から成る共有メモリプロセッサ（SMP）を 1 ノードとする分散共有メモリ型マシンである。CPU は MIPS R10000(195 [MHz])である。生成した OpenMP プログラムは MIPS Pro Fortran でコンパイルした。コンパイルオプションは以下である。

```
-mp -Ofast=ip27 -OPT:IEEE_arithmetic=3
```

2.6.1 並列実行性能の評価

Turb3d には、6 つの手続き呼出しのみを含む 4 つの主要ループがあり、実行時間のほとんどはこれらのループによって占められている。これらの各ループは、ループ毎に同じ手続きを呼び出しているが、各呼出し毎に実引数の定数値が異なる。WPP による手続き間並列化では、同じ手続きでも実引数値が異なるごとに別のクローン手続きを作成することによって、これら実引数値を呼出し先手続きに伝播させ、IF 文の削除、配列添字の単純化をコンパイル時に行なうことができた。更に、連立 1 次不等式を用いた手続き間配列解析により、ストライド参照の検出や異なる次元の引数配列に対応でき、4 つの主要ループ全てが並列化された。

図 3 は並列実行性能の評価結果である。ほぼリニアな台数効果を得たことがわかる。

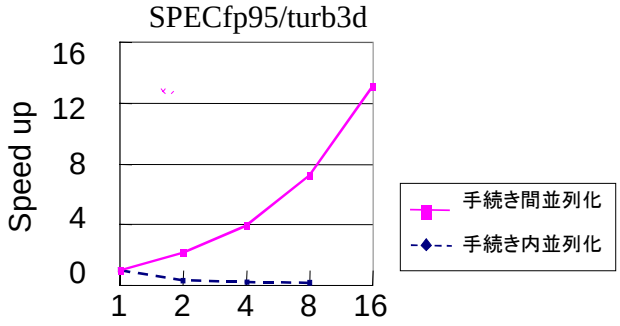


図３：Origin2000 上での並列実行性能

2.6.2　手続き間定数伝播の評価

表３は、turb3d から WPP が自動生成した OpenMP プログラム（IP）とその OpenMP プログラムにおける OpenMP 指示文を元の turb3d プログラムに挿入しただけのプログラム（manual）との Origin 2000 上での実行時間を比較した結果である。

IP の実行時間は manual に較べて、最低で 14.4%、最高では 33.5%高速である。この原因を確かめるために、Origin 2000 の 1 プロセッサでプロファイルを取った。その結果、IP と manual のキャッシュミス回数は 1%しか差がなかったが、実行命令数は IP の方が 31%だけ少なかった。よって、手続き間定数伝播により、IF 文の削減等による命令数の削減や変数の定数化による命令数の少ないコードの生成ができたと考えられる。

表３：manual 版と WPP 版との実行時間の比較[sec]

台数	1	2	4	8	16
manual	324.9	179.7	101.3	54.4	36.0
IP	274.6	148.4	79.8	46.6	23.9
$\frac{\text{manual - IP}}{\text{manual}}$	15.5%	17.4%	21.3%	14.4%	33.5%

3.　他コンパイラにおける手続き間解析

本章では、商用及び研究コンパイラにおける手続き間解析の動向を述べる。

IBM のXL Fortran v.7.1[5]では、以下のように積極的に手続き間解析を行なっている。しかし、手続き間自動並列化は未サポートと思われる。

- 手続きインライン展開
- 手続き間定数伝播
- 手続きクローニング

・手続き間Alias解析
・コードマッピング（caller-calleeの位置関係より）
・大域変数マッピング（参照解析結果より）

SGIのMIPSPro Fortran 90 [6]も以下のように積極的に手続き間解析を行なっている。このコンパイラでは自動並列化の時、インライン展開は必須とあるので、狭義の手続き間並列化は未サポートと思われる。

・手続きインライン展開
・手続き間定数伝播
・手続きクローニング
・手続き間Alias解析
・デッド関数・デッド変数・デッドCALL文の削除
・コモンブロック配列パディング（配列次元増加）

以上の 2 社は共に手続き間並列化をサポートしてない。このこととサポート項目より、手続き間解析の対象はスカラ変数のみであり、配列に対する参照リージョン解析は未サポートと予測できる。

以下の各社のコンパイラは手続きインライン展開のみサポートしている[7][8][9]。

- Forte Fortran/HPC (Sun Microsystems Inc.)
- Fortran Compiler for Linux (Intel)
- Visual Fortran (Compaq)
- Compaq Fortran for Tru64 UNIX/Linux Alpha/OpenVMS Alpha (Compaq)

KAP™ [10]も手続きインライン展開のみサポートしている。OpenMP プログラムを生成する時は、インライン展開した時の解析結果を利用するが、指示文はインライン展開しない元のプログラムに挿入する。

Stanford 大の SUIF コンパイ [11]は､狭義の手続き間解析による以下の解析を行なう。

・手続き間スカラ変数参照解析
・手続き間配列参照リージョン解析
・手続きクローニング
・手続き間定数伝播
・手続き間並列化（プライベート化、リダクション認識）

Illinois 大の Polaris コンパイラ及び Purdue 大の Polaris/OpenMP [12][13]はインライン展開を行なって手続き間解析を行なっている。Purdue 大の Eigenmann 教授によると最近、狭義の手続き間解析をやっているとのことだがその研究に関する発表はない。

・手続きインライン展開
・手続きクローニング

4. コンパイラ解析情報表示ツール Aivi

我々は、手続き間解析の結果を利用したコンパイラ解析情報表示ツールAivi（Analysis Information Visualizer）を開発している[3][4]。本章では、Aiviの機能について述べる。

4.1 並列化チューニング方法

我々は、手続き呼び出しを含むループに対する並列化チューニング方法は以下のように進むと考えている。

ステップ1：最も時間のかかるループの検出
ステップ2：そのループの外側ループの検出
ステップ3：外側ループ群中、最も効率的な並列ループの検出

ステップ1は既存のツールが利用できる。手続き呼び出しを含まないループに対しては、ここで検出されたループを最適化すれば良い。ところが、このループを含む手続きが複数の呼び出し点から呼ばれている場合、このループは何度も呼び出されるために実行時間が長くなり、重要なループとして検出されるが、実際はその呼び出し点を含むループを並列化する方が実行性能が良いことがあり得る。そこで、ステップ2が必要となる。次に、こうして得た外側ループに対して、ステップ3でデータ依存関係を調べ、なるべく外側で、効率良く並列化できるループをさがす。ループが自動並列化できない原因は、自動並列化コンパイラがデータ依存を精度良く解析できないことにある。ユーザはそれらの不明確なデータ依存をチェックし、ループ並列性を判定しなければならない。

以上を実現するため、Aiviは以下の２つの機能を持つ。

1) 与えられたループの外側ループを手続き境界を越えて検出すること
2) あるデータ依存の依存元と依存先を手続き境界を越えて検出すること

4.2　並列化構造表示・外側ループ検出機能

プログラム全体の並列化構造を視覚化し、外側ループを検出しやすくするためにAiviは以下の機能を持つ[3][4]。各機能のループ等は相互にリンクされている。

(1) *コールグラフ*　ループや手続き呼出しを、各々、インデント付きの四角形や三角形でコールグラフのノード上に実行順に表示する。並列化ループは色付けする。これらから手続き内の外側ループと呼び出し元手続きが分る。

(2) *プログラムフィルタリング*　プログラムから、手続きの開始・終了、ループ、手続き呼出し点等の、並列化に関連する文のみ抜き出して表示できる。これから手続き呼出し点やそれを含むループがすばやく検出できる。

(3) *インライン展開表示*　手続き呼出し点直後に呼び出し先プログラムを挿入して表示できる。これにより多重の呼び出しにまたがるループ間の包含関係がわかる。

4.3　手続き間データ依存位置検出機能

図4は手続き間データ依存位置検出機能を説明した図である。 (a)は従来のツールによるデータ依存位置検出機能が表示可能なデータ依存位置である[14][15][16]。解析対象ループLと同じ手続き（ベース手続きと呼ぶ）内にある、ループLに関して運搬依存のある文が表示される。 (b)は手続き間データ依存位置検出機能が表示可能なデータ依存位置である。ベース手続き内の文S1とベース手続き内から呼び出される手続き内にある、ループLに関して文S1とループ運搬依存のある文Qが表示される。本機能は以下の２つに分かれる。

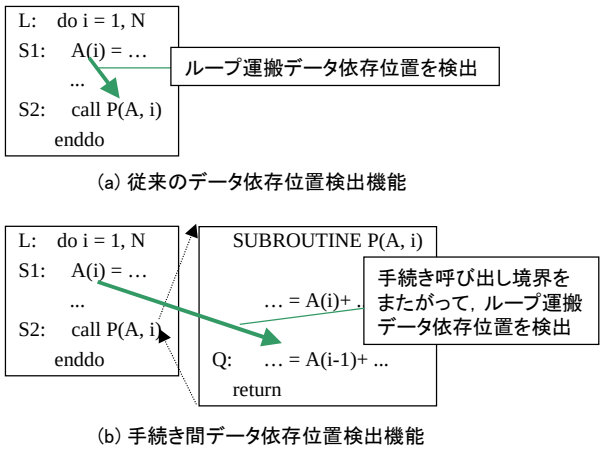


図４：手続き間データ依存位置検出機能

4.3.1　ループ内全データ依存位置検出機能

本機能は、ループに含まれる。ループと同じ手続き内にある文の対のうち、データ依存関係にあるものを全て検出する。この検出方法として、全ての文同士を比較するよりも比

較回数が少なくなることを期待して、本機能では階層的解析方式を採用している。これはループボディに対するCFSグラフを用い、データ依存のあるノードの対を、上位階層から下位階層に向かって順番に検出していく方式である。

4.3.2　手続き間段階的データ依存位置検出機能

本機能は、前項で検出された文の対のうち、一方が手続き呼び出しで、それをユーザが選択した場合、その呼び出し先手続き中にある、データ依存関係にある文を検出する。本機能では、与えられたループから呼び出される全ての手続きに対して一括して解析するよりも解析時間・解析データ量が少なくなることを期待して、段階的解析方式を採用している。これは解析対象ループ中の、ユーザが選択した手続きに対してのみデータ依存位置をその都度、解析し、その結果をユーザに表示する方式である。正確には、一括解析は、初回には解析に多くの時間がかかるが、ユーザ選択時にはユーザ指示に従って解析結果を表示するだけなのでレスポンスは良い。しかし、ユーザが選択してない手続きに対しても解析することになり、解析時間・データ量が無駄になるので、段階的解析方式を選択した。

図5はデータ依存位置リストの表示である。メインプログラムの8行目から呼び出されるDEFAと9行目から呼び出されるUSEA内のデータ依存（17、23行目）をユーザが表示したことを示し、現在、ユーザが選択しているのはDEFA内の17行目の文であることを強調表示している。

図５：データ依存位置リスト表示ウィンドウ

5.　他ツールにおける手続き間解析

本章では、ツールにおける手続き間解析の利用に関する動向を述べる。

Rice大のParaScope Editor[14]は対話的解析・プログラム変換ツールである。データ依存元と依存先をリスト表示し、ソースプログラム上でそのデータ依存を矢印で表示する。また、両方のデータ依存位置が離れている場合にはソースプログラムの一部を隠蔽して、それらを1画面に表示することが可能である。以下を行なう。

・手続き間配列参照解析（矩形表現）
・手続き間スカラ変数参照/シンボリック解析
・データ依存対のソースプログラム上表示

Stanford大のSUIF/Explorer[15]は対話的解析ツールである。ある文で参照する変数に対して、その変数値に影響を与える可能性のある文から成るプログラムスライス[20]を表示する。また、データ関係にある文をソースプログラム上で強調表示する。以下を行なう。

・手続き解析・並列化（SUIFコンパイラを利用）
・コールグラフ表示
・手続き間プログラムスライシング

Parallel Software Products Inc.のCAPTools [16] は、Greenwitch大、Nasa Ames等で開発されてきた。CAPToolsは対話的解析及びSPMDコード生成ツールである。データ依存元と依存先をノードとする依存グラフを表示する。以下を行なう。

・手続き間スカラ変数参照解析
・手続き間配列参照範囲解析　（Omega Testレベル）
・手続きクローニング
・コールグラフ表示
・データ依存グラフ表示（文と文）

GrammaTech Inc.のCodeSurfer[17]は、Sun, IBM, HP, SGI 各社のマシンにインストールされている。以下を行なう。

・手続き間Alias解析
・手続き間プログラムスライシング

6. おわりに

手続き間解析は商用の最適化コンパイラでも採用されつつある。その解析対象はまだスカラ変数程度であるが、自動並列化を目指した配列解析も徐々にサポートされると考えられる。一方、ツールに関しては、手続き間解析をサポートした商用ツールはまだ少ない。しかし、プログラムがモジュール化されてゆくにつれて、ツールにおいても手続き間解析のサポートが広がるものと考えられる。

参考文献

[1] 青木等：手続き間自動並列化コンパイラ WPP の試作 －実機性能評価－, 情処研究報告, 98-ARC-130, pp.43-48 (1998).

[2] 佐藤等：手続き間自動並列化コンパイラ WPP の評価, 第 130 回計算機アーキテクチャ研究会（SHINING 2001 (2001)).

[3] 佐藤等：コンパイラ解析情報ビジュアライザ Aivi, 情処第 62 回全国大会, 4R-05, Mar., (2001).

[4] M. Satoh et al. “Interprocedural Parallelizing Compiler WPP and Analysis Information Visualization tool Aivi”, Proceedings of The Second European Workshop on OpenMP (EWOMP 2000), pp. 38-47, 2000.

[5] Bob Blainey. “Performance Programming with IBM pSeries Compilers”, SCICOMP4 (IBM SP Scientific Computing User Group) Oct. 2001.

[6] SGI's online-manual (IRIX 6.5 Man Pages: IPA(5)), http://www.sgi.com.

[7] http://www.sun.com/forte/developer/documentation/mr/READMEs/c.html#upd2a

[8] http://developer.intel.com/software/products/compilers/f50/linux/finfo_ipo.htm

[9] http://www.compaq.com/fortran/docs/

[10] Kuck and Associates, Inc., http://www.kai.com/

[11] M. Hall et al. “Maximizing Multiprocessor Performance with the SUIF Compiler”, IEEE Computer, pp.84-89, Dec. 1996.

[12] Polaris/OpenMP, http://polaris.cs.uiuc.edu /

[13] W. Blume et al. “Parallel Programming with Polaris”, IEEE Computer, pp.78-81, Dec. 1996.

[14] M. Hall et al. “Experiences using the ParaScope Editor: an Interactive Parallel Programming Tool”, Proceedings of PPoPP '93, 1993.

[15] S. -W. Liao et al. “SUIF Explorer: An Interactive and Interprocedural Parallelizer”, In proceedings of the 7th ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming (PPOPP'99), pp. 37-48, Atlanta, Georgia, May 1999.

[16] S. P. Johnson et al. “Computer Aided Parallelization Tools (CAPTools) User Manual, CAPTools Version 2.0Beta”, Oct. 1998. http://captools.gre.ac.uk.

[17] http://www.grammatech.com/

[18] OpenMP Fortran Application Program Interface Ver 1.0, Oct. 1997. http://www.openmp.org.

[19] W. Pugh. A practical algorithm for exact array dependence analysis. Communications of the ACM, 35(8), pp. 102-114, 1992.

[20] M. Weiser, Program slicing, IEEE Transactions on Software Engineering, 10(4), pp. 352-357, 1984.

3.3.2 自動並列化コンパイラによる SPM 上での粗粒度タスク並列処理

笠原　博徳　委員

1. はじめに

現在、SMP アーキテクチャはシングルチップマルチプロセッサからワークステーション、各種サーバマシンまで多くのシステムで採用されている。このような SMP マシン上では従来よりループレベル並列処理技術が用いられ、様々なプログラムリストラクチャリング技術を用いたループ並列化コンパイラが開発されてきた。例えば、Polaris コンパイラ[1,2]は、サブルーチンのインライン展開、シンボリック伝搬、アレイプライベタイゼーション[3,1]、実行時データ依存解析[2]によってループ並列性を抽出する。また、SUIF コンパイラ[4,5,6]は、インタープロシージャ解析、ユニモジュラ変換、アフィンパーティショニングを用いたループ内とループ間のデータローカリティ[7,8,9]に関する最適化などを用いてループを並列処理する。

これらをはじめとする優れたループ並列化コンパイラにより、様々なプログラムの解析・評価が行われてきたが、ループレベル並列性抽出技術は既に成熟期にあり、今後の画期的な性能向上は見込めないと言われている。したがって、今後の並列処理の性能向上のためには、これまでの並列化コンパイラでは抽出できなかった粗粒度並列性の利用が重要となる。

粗粒度並列性を利用するコンパイラとしては、NANOS コンパイラ[10,11]と PROMIS コンパイラ[12,13]、そして OSCAR マルチグレイン並列化コンパイラが挙げられる。NANOS コンパイラでは、階層並列化実現のための拡張 OpenMP API[14,15,16]を用いて、粗粒度並列性を含むマルチレベル並列性を抽出しようとしている。また、PROMIS コンパイラは、HTG とシンボリック解析[17]を用いる Parafrase2 コンパイラ[18]をベースとして、実用レベルのコンパイラを開発する努力が行われている。

筆者らが開発中の OSCAR FORTRAN マルチグレイン並列化コンパイラ[19,20]は、ループ並列化に加え、ループ・サブルーチン・基本ブロック間の並列性を利用するマルチグレイン並列処理[21,19,20]を実現し、OpenMP を用いたワンタイムシングルレベルスレッド生成手法[22]によって、NANOS コンパイラのような言語拡張を行うこと無く、低オーバヘッドでの粗粒度並列処理を可能としている。さらに SMP マシンで問題となる大きな共有メモリアクセスオーバヘッドを軽減するため、粗粒度タスク間共有データをキャッシュメモリ上で授受できるようにするデータローカライゼーション手法[23]の開発も行っている。

本報告では、この OSCAR マルチグレイン並列化コンパイラを用いて、SPEC95FP ベンチマークの 101.tomcatv、102.swim、107.mgrid、103.su2cor、Perfect Club ベンチマークの ARC2D について、SMP サーバ IBM RS6000、SMP ワークステーション SUN

Ultra80 の 2 機種のマシン上で粗粒度タスク並列処理性能の評価を行った結果について述べる。

2. 粗粒度タスク並列処理

ここでは、逐次プログラムを粗粒度タスクに分割し粗粒度タスク間並列性解析を行う手法、及び並列性抽出後の OpenMP を用いた粗粒度並列化プログラム生成法について述べる。

粗粒度タスク並列処理とは、プログラムを基本ブロック、あるいはその融合ブロック（BPA)、繰返しブロック（RB)、サブルーチンブロック（SB）の 3 種類のマクロタスク（MT）に分割し、その MT をプロセッサエレメント（PE）や複数 PE をグループ化したプロセッサクラスタ（PC）に割り当てて実行することにより、MT 間の並列性を利用する方式である。

OSCAR マルチグレイン並列化コンパイラにおける粗粒度並列処理の手順は次のようになる。

1) 逐次プログラムの MT への分割
2) MT 間の制御フロー、データ依存解析によるマクロフローグラフ（MFG）の生成
3) 最早実行可能条件解析によるマクロタスクグラフ（MTG）の生成[24,21]
4) MTG がデータ依存エッジのみで構成される場合は、スタティックスケジューリングによる MT の PC または PE への割り当て。 MTG がデータ依存エッジと制御依存エッジを持つ場合は、コンパイラによるダイナミックスケジューリングルーチンの自動生成
5) OpenMP による粗粒度並列化プログラム生成

以下、各ステップの概略を示す。

2.1 粗粒度タスク生成

粗粒度タスク並列処理では、ソースプログラムは、基本ブロック、あるいはその融合ブロック（BPA)、繰返しブロック（RB)、サブルーチンブロック（SB）の 3 種類のマクロタスク（MT）に分割される。生成された RB が並列化可能ループ、すなわち Doall ループの場合は、 PC 数やキャッシュサイズを考慮した数の粗粒度タスクにループを分割し、それぞれ異なった粗粒度タスクとして定義する。また、ループ並列化不可能で実行時間の長い RB やインライン展開を効果的に適用できない SB に対しては、そのボディ部を階層的に粗粒度タスクに分割し並列処理を行う。

2.2 粗粒度並列性抽出

次に、各階層（ネストレベル）において、生成された MT 間のデータ依存と制御フローを解析する。解析結果は、図 1（a）に示すようなマクロフローグラフ（MFG）として表される。図中、ノードは MT を表し、実線エッジはデータ依存、点線エッジは制御フローを表す。また、ノード内の小円は条件分岐を表す。図中のエッジの矢印は省略されているが、エッジの向きは下向きを仮定している。

MFG 生成後、MT 間の並列性を抽出するために、データ依存と制御依存を考慮し、各 MT の最早実行可能条件を解析する。最早実行可能条件とは、各 MT が最も早い時点で実行可能になる条件を表し、例えば、図 1（a）における MT6 の最早実行可能条件は、MT3 が終了するか MT2 が MT4 に分岐、となる。各 MT の最早実行可能条件は、図 1（b）に示すようなマクロタスクグラフ（MTG）として表される。

MTG においても、ノードは MT を、ノード内の小円は条件分岐を表す。また、実線エッジはデータ依存を表し、点線エッジは拡張された制御依存を表す。ただし、拡張された制御依存とは、通常の制御依存だけでなく、MTi のデータ依存先行 MT が実行されない条件も含むものである。図 1（b）中のエッジを束ねる実線アークは、アークによって束ねられたエッジが AND 関係にあることを示し、点線アークは OR 関係にあることを表す。MTG においても、エッジの矢印は省略されているが、下向きを仮定している。また、矢印のあるエッジはオリジナルの制御フローを表している。

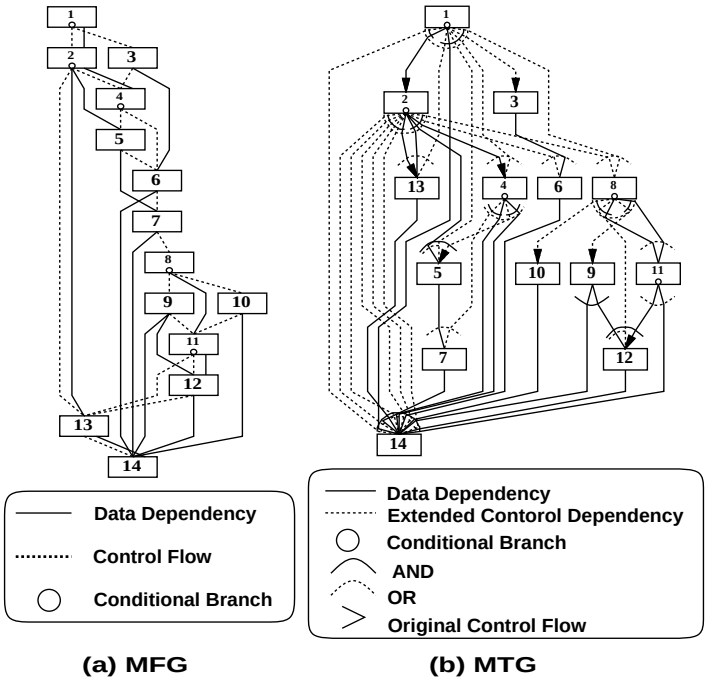


図 1: マクロフローグラフ（MFG）とマクロタスクグラフ（MTG）の例

2.3　OpenMP を用いた SMP 用粗粒度タスク並列化コード生成

“SECTIONS”によってプログラムの実行開始時に一度だけ並列スレッドを生成するワンタイムシングルレベルスレッド生成手法[22]を用いる。OSCAR コンパイラによって生成された粗粒度タスクは、ダイナミックスケジューリング、あるいはスタティックスケジューリングを用いて、PE に対応するスレッド、あるいは PC に対応するスレッドグループに割り当てられる。スレッドグループとは、MT の割り当て単位として、任意の数のスレッドをプログラム的にグループ化したものである。MTG 内に条件分岐のような実行時不確定性が存在する場合はダイナミックスケジューリングが適用され、MT は実行時にスレッドグループもしくはスレッドに割り当てられる。ダイナミックスケジューリングルーチンは、OS コールによるスレッドスケジューリングオーバヘッドを避けるために、コンパイラによって各プログラムに合わせて生成され、出力される並列化プログラム内に埋め込まれる。一方、MTG がデータ依存のみで構成される場合には、コンパイル時にスタティックスケジューリングを適用し、実行時スケジューリングオーバヘッドの最小化を図っている。

階層的な MTG が生成されている場合は階層的粗粒度並列処理を行うが、一般的に、階層的並列処理は、上位スレッドが子スレッドを生成することによって実現される。しかし、本研究で用いる OSCAR FORTRAN コンパイラでは、“PARALLEL SECTIONS”～“END PARALLEL SECTIONS”ディレクティブ間の“SECTION”ディレクティブ間に、生成される全 MTG 階層での処理やダイナミックスケジューリングを適用する全 MTG 階層のスケジューリングルーチンを記述することによって、シングルレベルスレッド生成のみで階層的並列処理を実現する。これにより、スレッドの fork/join オーバヘッドの最小化、及び既存の言語仕様のみで階層的粗粒度並列処理を実現することができる。

また、スタティックスケジューリング、ダイナミックスケジューリングにおいて、MT 間でキャッシュメモリを用いて共有データを受け渡すためのデータローカライゼーション手法[23]を適用することも可能である。データローカライゼーションを適用する際は、各 MT の使用データ量がキャッシュサイズ以内に収まるように粗粒度タスク分割し、同一データ領域を使用する MT ができるだけ同一プロセッサ上で連続実行されるようにスケジューリングを行う。

3.　SMP 上での性能評価

ここでは、OSCAR マルチグレイン並列化 Fortran コンパイラを用いて、SPEC95 ベンチマークの tomcatv、swim、mgrid、su2cor、Perfect Club ベンチマークの ARC2D プログラムを粗粒度タスク並列化し、IBM RS6000 SP 604e High Node、SUN Ultra80 上で性能評価を行った結果を述べる。

3.1 OSCAR FORTRAN Compiler

マルチグレイン並列化コンパイラ“OSCAR FORTRAN コンパイラ”は、フロントエンド・ミドルパス・バックエンドから構成される。本研究では、OpenMP ディレクティブを含む並列化 Fortran ソースコードを自動的に生成する OpenMP バックエンドを用いる。すなわち、OSCAR コンパイラは逐次 Fortran を OpenMP Fortran に変換するプリプロセッサとして利用される。

3.2 RS6000 上での評価

ここでは、RS6000 上での粗粒度並列処理結果について述べる。

評価に用いた RS6000 SP 604e High Node は、200MHz の PowerPC 604e を 8 プロセッサ搭載した SMP サーバである。1 プロセッサあたり、32KB ずつの L1 命令、データキャッシュと、1MB のユニファイド L2 キャッシュを持ち、共有主メモリは 1GB である。使用したコンパイラは XL Fortran Compiler Version 7 であり、OSCAR コンパイラの出力をコンパイルする際のオプションは、“-O3 -qsmp=noauto -qhot -qarch=ppc -qtune=auto -qcache=auto”である。XL Fortran Ver.7 単独での評価においては、逐次性能評価でのオプションとして“-O5 -qhot -qarch=ppc -qtune=auto -qcache=auto”を用い、自動並列化では“-O5 -qsmp=auto -qhot -qarch=ppc -qtune=auto -qcache=auto”を用いた。

図 2 に tomcatv、swim、mgrid、su2cor、arc2d の評価結果を示す。横軸はプログラム名を示し、プログラム名の下の数字はオリジナルプログラムを XL Fortran で逐次処理コンパイルした際の実行時間を表す。縦軸は、XL Fortran Ver.7 での逐次処理時間を 1.00 とした場合の速度向上率を示す。各プログラムについては、図中左からオリジナルソースプログラムを XL Fortran によって自動並列化した際の 8 スレッドでの性能、XL Fortran による自動並列化において 8 スレッドまで用いて並列化を行った場合の最高性能が得られたスレッド数とその性能、OSCAR コンパイラの OpenMP 出力を XL Fortran でコンパイルした際に最高性能を与えたスレッド数とその際の性能を表している。また、棒グラフ上部の数字はそれぞれの並列処理時間である。

図 2 中、tomcatv では、XL Fortran による自動並列化においては 3 スレッドで最高性能が得られ、1.7 倍の速度向上となったのに対して、OSCAR コンパイラは 8 スレッドで 5.7 倍の速度向上が得られた。 swim では、逐次処理に対する XL Fortran は 6 スレッドで 4.2 倍だが、OSCAR コンパイラの性能は 8 スレッドで 8.3 倍となった。また mgrid においては、XL Fortran では 4 スレッドで 3 倍の速度向上であったが、OSCAR コンパイラでは 8 スレッド時で 6.8 倍の最高性能が得られた。tomcatv、swim、mgrid では、スタティックスケジューリングを用いて粗粒度並列処理を行っている。これらのアプリケーションはループ並列性が大きいため、XL Fortran も OSCAR コンパイラもループ並列性を利用しているが、OSCAR コンパイラのスレッド数に対するリニアな性能向上に対して、XL Fortran ではスレッド数の増加に対してスケーラブルな性能向上が見られない。この差の

理由として、time コマンドによる計測結果における System time の差が使用スレッド数が増えるにつれて非常に大きくなることが挙げられる。例として、表 1 に、tomcatv と swim における各スレッド数ごとの System time を示す。表 1 の tomcatv では、XL Fortran の自動並列化で最高性能を示す 3 スレッドでの System time が 110 秒であるのに対して、OSCAR コンパイラでは最高性能が得られる 8 スレッドで 3.1 秒であり、差が非常に大きい。また、RS6000 上で各スレッドやプロセスの処理時間などを計測できる PPROF コマンドの出力結果より、XL Fortran による自動並列化の際の各スレッドでの処理時間は、マスタースレッドの処理時間とスレーブスレッドでの処理時間に非常に大きな差が出ることを確認しており、負荷の不均衡が生じている。

一方、図 2 における su2cor、arc2d は、分散ダイナミックスケジューリングによる粗粒度並列処理を行った結果である。ただし、OSCAR コンパイラによる評価では、su2cor ではインライン展開、配列リネーミング、arc2d ではインライン展開、ループアンローリングをオリジナルの逐次プログラムに対して適用したプログラムを逐次プログラムとして OSCAR コンパイラに入力した。これは、研究用コンパイラである OSCAR コンパイラは既存のコンパイラで行われていない機能の実現・実装に主眼を置いているため、多くの市販コンパイラで実現されているリストラクチャリング技術のいくつかは未実装、あるいはデバッグ作業を遅らせているため、今回の評価では手動でリストラクチャリングを行った後、OSCAR コンパイラを用いた粗粒度並列化を行った。

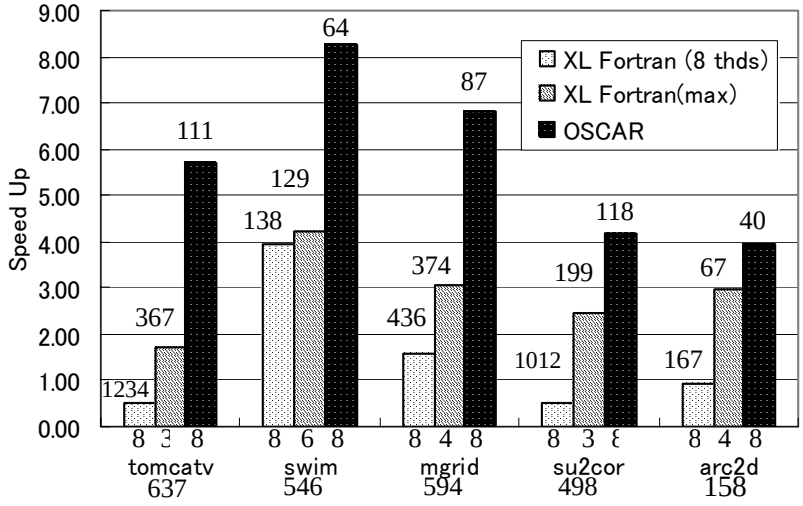


図 2: RS6000 上での評価

逐次処理に対する XL Fortran による自動並列化での最高性能は、su2cor では 2.5 倍、arc2d では 3.0 倍の速度向上であったが、OSCAR コンパイラにおける最高性能は、su2cor では 8 スレッドで 4.2 倍、arc2d では 8 スレッドで 3.9 倍となった。su2cor、arc2d においては、OSCAR コンパイラによる評価では、分散ダイナミックスケジューリングを適用することによって、図 2 に示すような性能が得られたのに対して、XL Fortran の性能は tomcatv、swim、mgrid と同様、OSCAR コンパイラよりも低い性能を示している。また、使用スレッド数と性能で比較した場合、OSCAR コンパイラでは全てにおいて 8 スレッドを用いた場合が最高性能を示すが、 XL Fortran での 8 スレッド使用時の性能は、swim と mgrid を除いて、逐次処理よりも性能が低下している。

したがって、XL Fortran では OS および OpenMP ランタイムルーチンによるスレッドスケジューリングを含めたスレッド管理のためのオーバヘッドが大きいため、性能が向上しにくいことがわかった。同時に、OSCAR コンパイラでのワンタイムシングルレベルスレッド生成手法を用いた粗粒度並列処理手法は、このスレッド管理オーバヘッドを低く抑えられるため、有効であることが確認された。

3.3 Ultra80 上での評価

次に、OpenMP を用いた粗粒度並列処理手法のポータビリティを示すため、tomcatv、swim の Ultra80 上での粗粒度並列処理結果について述べる。

本研究で用いた SUN Ultra80 は、450MHz の Ultra SPARCII を 4 つ搭載しており、 1 プロセッサあたり、16KB ずつの L1 命令、データキャッシュと 4MB のユニファイド L2 キャッシュを持ち、共有主メモリは 1GB である。使用したネイティブコンパイラは Forte Developer 6 Update 1 であり、OSCAR コンパイラの出力した粗粒度並列化 Fortran をコンパイルする際のオプションは“-fast -mp=openmp -explicitpar –stackvar”である。Forte 自体の性能評価においては、逐次処理では“-fast”、自動並列化では“-fast -parallel -reduction –stackvar”を用いた。Ultra80 は RS6000 と比較して共有メモリアクセスオーバヘッドが非常に大きいため、今回の tomcatv の評価においてはローカライゼーション手法を適用した。その際、前述のように OSCAR コンパイラの未実装部分、およびバグを避けるため、ループ終値変数の定数化、収束ループ内のループインターチェンジ、配列の次元入れ替えといったリストラクチャリングを手動で行った。今回は、Forte による逐次処理、自動並列化、OSCAR コンパイラによる評価全てに対してこれらの改変後のプログラムを用いた。また、swim についても、サブルーチンのインライン展開とクローニング、ループ終値変数の定数化を行い、tomcatv 同様の評価を行った。これらの変換についても、RS6000 での評価と同様に、既存のコンパイラによって実現されている変換技術であり、現在 OSCAR コンパイラへ組み込み中である。

図 3、4 に Ultar80 上での tomcatv、swim の評価結果をそれぞれ示す。グラフの横軸は使用スレッド数、縦軸は Forte による逐次処理時間を 1.00 とした速度向上率を表し、

“OSCAR(Localize) ”は OSCAR コンパイラによるローカライゼーション手法、“OSCAR(Loop) ”は OSCAR コンパイラでループ並列性のみの利用、そして“Forte”は Forte 6 による自動並列化の結果を表す。また、各プロット横の数字は実行時間（単位は秒）を表す。

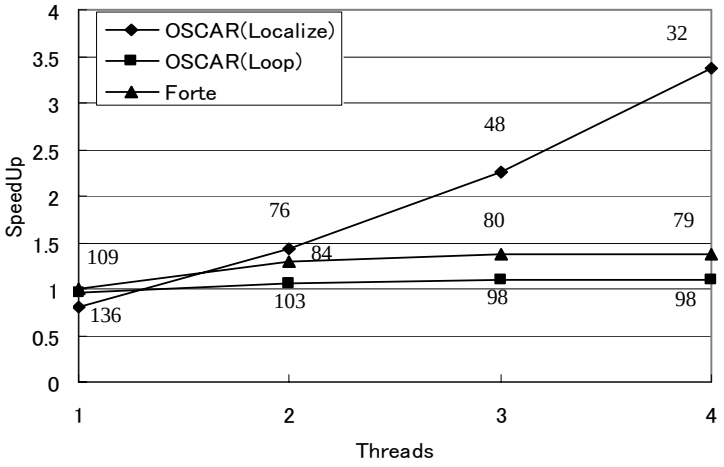


図 3: Ultra80 上での tomcatv の評価結果

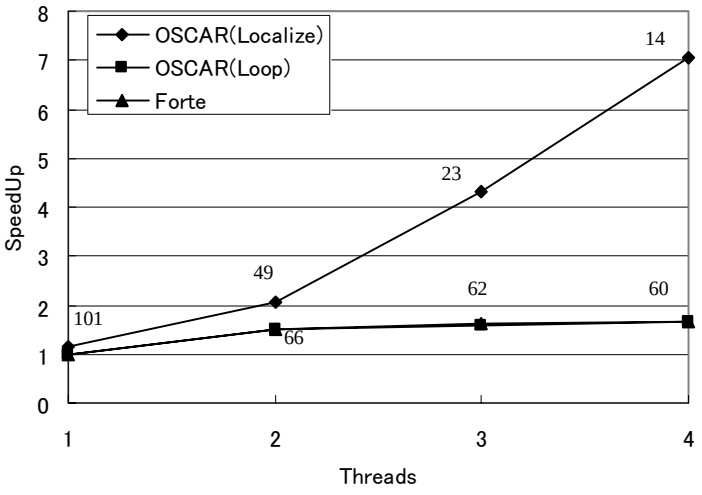


図 4: Ultra80 上での swim の評価結果

tomcatv、swim ともに、OSCAR コンパイラによるデータローカライゼーション手法を適用した結果は優れた性能を示している。Forte による自動並列化では性能はほとんど向上しないが、OSCAR コンパイラの評価においては、tomcatv では、ローカライゼーションの適用により 4 スレッドで 3.4 倍の速度向上が得られた。swim では、4 スレッドで 7.1 倍の速度向上を達成している。

図 3 の 1 スレッド部分を見ると、OSCAR コンパイラによってローカライゼーション手法を適用した結果は Forte による逐次処理の性能よりも低い性能を示しているが、スレッド数が増えるにつれて高効率を示す。この結果は、Forte ではオリジナルソースプログラムに対するキャッシュタイリング技術を中心とした逐次処理最適化は非常に強力だが、ローカライゼーション手法を適用した OSCAR コンパイラによる逐次処理用出力への最適化には向かないことを示している。また図 3、4 より、並列処理時における OSCAR コンパイラでのループ並列性利用結果は、Forte の性能とほぼ同等かそれよりも若干低下していることが分かる。この結果は、Ultra80 ではキャッシュミスヒットした際の主メモリへのアクセスが非常に高コストであるため、OSCAR コンパイラによるデータローカライゼーション手法が有効であることを示している。

4. まとめ

本報告では、SPEC95FP ベンチマーク、Perfect Club ベンチマークから、101.tomcatv、102.swim、103.su2cor、107.mgrid、ARC2D について、SMP サーバ IBM RS6000 SP 604e High Node と SMP ワークステーション SUN Ultra80 上での粗粒度タスク並列処理について述べた。これらのマシン上での性能評価の結果、RS6000、Ultra80 用のネイティブコンパイラ XL Fortran Version 7、 Forte Developer 6 Update 1 の自動ループ並列化性能を 60%から 430%上回る性能を得られることが確かめられた。この評価より、筆者等が開発した粗粒度タスク並列処理スレッド生成手法であるワンタイムシングルレベルスレッド生成手法、およびキャッシュ最適化のためのデータローカライゼーション手法が、スレッド管理オーバヘッド、共有メモリアクセスオーバヘッドを軽減し、ＳＭＰ上で効果的な並列処理を実現するために有効であることが確認された。

なお本研究の一部は、経済産業省／NEDO ミレニアムプロジェクト IT21 “アドバンスト並列化コンパイラ”により行われたものである。

参考文献

[1] Eigenmann, R., Hoeflinger, J. and Padua, D.: On the Automatic Parallelization of the Perfect Benchmarks, IEEE Trans. on parallel and distributed systems, Vol. 9, No. 1 (1998).

[2] Rauchwerger, L., Amato, N. M. and Padua, D. A.: Run-Time Methods for Parallelizing Partially Parallel Loops, Proceedings of the 9th ACM International

Conference on Supercomputing, Barcelona, Spain, pp. 137-146 (1995).

[3] Tu, P. and Padua, D.: Automatic Array Privatization, Proc. 6th Annual Workshop on Languages and Compilers for Parallel Computing (1993).

[4] Hall, M. W., Murphy, B. R., Amarasinghe, S. P., Liao, S., and Lam, M. S.: Interprocedural Parallelization Analysis: A Case Study, Proceedings of the 8th International Workshop on Languages and Compilers for Parallel Computing (LCPC95) (1995).

[5] Hall, M. W., Anderson, J. M., Amarasinghe, S. P., Murphy, B. R., Liao, S.-W., Bugnion, E. and Lam, M. S.: Maximizing Multiprocessor Performance with the SUIF Compiler, IEEE Computer (1996).

[6] Amarasinghe, S., Anderson, J., Lam, M. and Tseng, C.: The SUIF Compiler for Scalable Parallel Machines, Proc. of the 7th SIAM conference on parallel processing for scientific computing (1995).

[7] Lam, M. S.: Locallity Optimizations for Parallel Machines, Third Joint International Conference on Vector and Parallel Processing (1994).

[8] Anderson, J. M., Amarasinghe, S. P. and Lam, M. S.: Data and Computation Transformations for Multiprocessors, Proceedings of the Fifth ACM SIGPLAN Symposium on Principles and Practice of Parallel Processing (1995).

[9] Lim, A. W. and Lam., M. S.: Cache Optimizations With Affine Partitioning, Proceedings of the Tenth SIAM Conference on Parallel Processing for Scientific Computing (2001).

[10] Martorell, X., Ayguade, E., Navarro, N., Corbalan, J., Gozalez, M. and Labarta, J.: Thread Fork/Join Techniques for Multi-level Parllelism Exploitation in NUMA Multiprocessors, ICS'99 Rhodes Greece (1999).

[11] Ayguade, E., Martorell, X., Labarta, J., Gonzalez, M. and Navarro, N.: Exploiting Multiple Levels of Parallelism in OpenMP: A Case Study, ICPP'99 (1999).

[12] PROMIS: http://www.csrd.uiuc.edu/promis/.

[13] Brownhill, C. J., Nicolau, A., Novack, S. and Polychronopoulos, C. D.: Achieving Multi-level Parallelization, Proc. of ISHPC'97 (1997).

[14] : OpenMP: Simple, Portable, Scalable SMP Programming http://www.openmp.org/.

[15] Dagum, L. and Menon, R.: OpenMP: An Industry Standard API for Shared Memory Programming, IEEE Computational Science & Engineering (1998).

[16] Ayguade, E., Gonzalez, M., Labarta, J., Martorell, X., Navarro, N. and Oliver, J.: NanosCompiler: A Research Platform for OpenMP Extensions, Proc. of the 1st Europian Workshop on OpenMP (1999).

[17] Haghighat, M. R. and Polychronopoulos, C. D.: Symbolic Analysis for Parallelizing Compliers, Kluwer Academic Publishers (1995).
[18] Parafrase2: http: //www.csrd.uiuc.edu/parafrase2.
[19] 岡本雅巳, 合田憲人, 宮沢稔, 本多弘樹, 笠原博徳: OSCAR マルチグレインコンパイラにおける階層型マクロデータフロー処理手法, 情報処理学会論文誌, Vol. 35, No. 4, pp. 513-521 (1994).
[20] Kasahara, H., Okamoto, M., Yoshida, A., Ogata, W., Kimura, K., Matsui, G., Matsuzaki, H. and H.Honda: OSCAR Multi-grain Architecture and Its Evaluation, Proc. International Workshop on Innovative Architecture for Future Generation High-Performance Processors and Systems (1997).
[21] Kasahara, H. et al.: A Multi-grain Parallelizing Compilation Scheme on OSCAR, Proc. 4th Workshop on Languages and Compilers for Parallel Computing (1991).
[22] 笠原博徳, 小幡元樹, 石坂一久: 共有メモリマルチプロセッサシステム上での粗粒度タスク並列処理, 情報処理学会論文誌, Vol. 42, No. 4 (2001).
[23] 吉田明正, 越塚健一, 岡本雅巳, 笠原博徳: 階層型粗粒度並列処理における同一階層内ループ間データローカライゼーション手法, 情報処理学会論文誌, Vol. 40, No. 5 (1999).
[24] 本多弘樹, 岩田雅彦, 笠原博徳: Fortran プログラム粗粒度タスク間の並列性検出手法, 電子情報通信学会論文誌, Vol. J73-D-1, No. 12, pp. 951-960 (1990).

3.3.3　P2P の理念およびその実現技術：SIONet の全貌

星合　隆成　講師

1.　はじめに

筆者[1]-[53]は、1998 年にブローカレス型探索モデル（ブローカレスモデル）を提唱して以来、その実現技術として「**意味情報ネットワーク(SIONet: Semantic Information-Oriented Network)**」を考案し、これまで提案してきた。そして、1998 年末に SIONet プロトタイプα版、1999 年末に SIONet プロトタイプβ版の試作を完了した。さらに、2000 年末に SIONet ver.1.0 の開発を完了した。

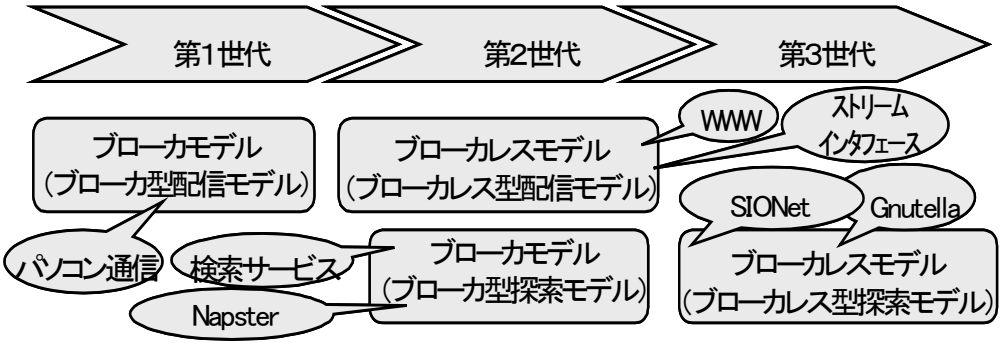
本稿では、ブローカレス型探索モデル（図 A，図 B）を紹介することにより、「**P2P の本質**」について論ずる。さらに、ブローカレス型探索モデル（P2P モデル）の実現技術である SIONet について解説する[1]-[31]、[34]-[44]。

2.　ブローカレス型探索モデル　― 情報のよりよい探索と配信をめざして ―

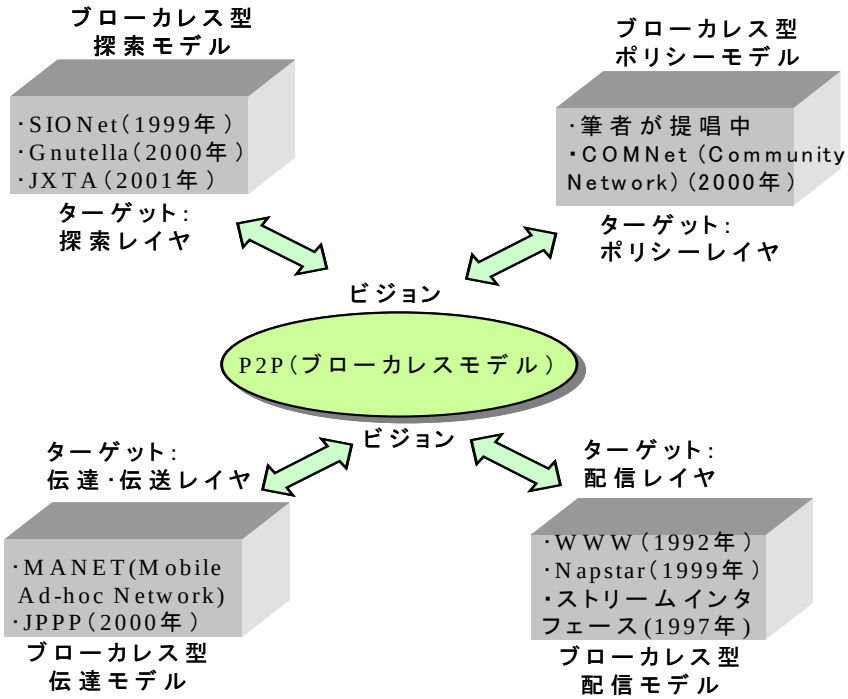
米国国防総省が 1969 年に開始した実験ネットワーク(ARPANET)を礎とするインターネットは、今では我々の生活に欠かせないものになっている。世界中のあらゆるコンピュータを相互接続したインターネットは、いくつかの技術革新を経て、誰もがいつでもどこでも、気軽にネットワーク上の情報にアクセスできるネットワーク環境を我々に提供するようになった。

第 1 世代におけるインターネットでは、「**ブローカ型配信モデル**」に基づいた情報配信が一般的であった。このビジネスモデルでは、ブローカとして位置付けられる情報配信者（例えば、パソコン通信会社）が、情報提供者の情報を集中管理し、必要に応じてユーザに対して情報の配信を行った。すなわち、ブローカがユーザと情報提供者を結びつける役割を果たしていた。そのため、ブローカは、膨大な情報を格納するための格納庫（ハードディスクなどのストレージ)、情報を処理するための高性能演算装置（高性能ワークステーションなど)、そしてユーザとの通信帯域の十分な確保など、膨大な設備投資が必要であった。

これに対し、インターネットの利便性を飛躍的に向上させ、利用者の裾野を拡大させた革新技術が、1992 年に発表された WWW(World Wide Web)である。これがインターネット第 2 世代への扉を開いた。ユーザはブラウザを介して、世界中のあらゆる情報（コンテンツ）を利用できる一方で、情報提供者はブローカ（情報配信者）を介することなく、自らが情報発信することが可能になった。



図A ビジネスモデルの変遷



図B P2Pモデルのディメンション

WWW では、個々の情報提供者が情報を超分散管理し、情報同士をハイパーリンクすることにより、情報提供者のみで情報の超分散データベースを構築することができる。これにより、情報配信者を必要としない「**情報配信ネットワークの構築**」が可能になった。これこそが、インターネットが目指した真の姿であり、WWW が第 2 世代の革新技術と言われる所以である。本稿では、WWW のような配信技術によって可能になった新たなビジネスモデルを、「**ブローカレス型配信モデル**」と呼ぶ。[1]

ブローカレス型配信モデルに基づく情報配信、すなわち、情報提供者主導の情報配信が可能になる一方で、世界中に氾濫する情報の中から、ユーザが望む情報を簡単に特定する

[1] End-to-End でストリームデータの配送を行うストリームインタフェース[32],[33]もブローカレス型配信モデルに位置付けられる。

ことや、情報提供者に取って情報を配信するに相応しいユーザを特定できることが重要な課題になった。残念ながら、WWW はこの課題に対するソリューションを与えてはくれない。そのため、ポータルサイト、検索サービス、リコメンデーションサービス、ディレクトリサービス、トレーダ、ルックアップサーバと呼ばれる新たなブローカが登場した。このビジネスモデルを「**ブローカ型探索モデル**」と呼ぶ。ブローカ型探索モデルでは、情報（源）の探索、もしくは、情報の配信先の探索まではブローカを介して行うが、発見された相手との情報配信は、end-to-end (peer-to-peer)で直接的に、すなわち、ブローカレス型配信モデルに基づいて行われる。従って、1999 年 1 月に発表され話題になった Napster も、ブローカ型探索モデルに位置付けることができる。[2]

しかしながら、ブローカ型探索モデルでは、ブローカが情報のメタデータを一括管理しているために、ユーザや情報提供者の多用なニーズにリアルタイムに応えることが困難であること、すべてをブローカの運営ポリシーに委ねなければならないこと、ブローカが運営しているシステム（サーバ）がダウンするとすべてのサービスが停止してしまうこと、ユーザや情報提供者のプライバシー保護が困難なことなどから、より自由で、特定のブローカに支配されない新たなビジネスモデルが求められるようになった。すなわち、これまでのビジネスモデルにおいて一般的に用いられてきたブローカ（集中管理者）を排除し、エンティティ間（例えば、ユーザと情報提供者との間）で、互いに相応しい相手を直接探索・発見することが可能なビジネスモデルが求められるようになった。

このような問題認識から、筆者が 1998 年に提唱した新たな概念が、「**ブローカレス型探索モデル**」である。これは、2000 年 3 月に発表された Gnutella を契機に、世界中で注目されるようになった P2P(Peer-to-Peer)と同じ世界を指向しているものである。また、2001 年 4 月に発表された JXTA も同様の世界を目指している。P2P は、WWW 以来の革新技術であり、第 3 世代と呼ばれている。なお、近年、伝送レイヤに P2P の概念を適用したブローカレス型伝送モデルや、ポリシーレイヤに P2P の考え方を適用したブローカレス型ポリシーモデル（COMNet [4], [5]）の研究開発が活発化していている。

ブローカレス型探索モデル（P2P モデル）は、ビジョン･概念･理念であり、これが目指す世界は非常に単純で明快である。具体的には、以下のような世界を構築することを目標としている。

---

[2] 余談ですが、Napster を P2P（ブローカレス型配信モデル）として位置付けることに若干の疑念があります。それは、WWW によって初めて提唱されたブローカレス型配信モデルは、ブローカの存在なしに、自らが情報の配信ネットワークを構築できる点にあります。しかしながら、Napster はネットワーク（面）ではなく、単なるポイント（点）に過ぎません。すなわち、Napster を P2P と定義した時点で、FTP や糸電話、ありとあらゆる通信形態が P2P として定義されることになります。これは、P2P（ブローカレス）の本質を全く反映しておりません。なぜなら、通信の起源は、peer と peer の通信からスタートしているからです。これは、P2P 通信方式（P2P インタラクション方式）と呼ばれる古典的な考え方であり、今、話題の P2P とは次元の違うものです。現状を省みると、このような誤解が多々見受けられます。これらについては文献[1], [2], [3], [13]を参照ください。

- 従来、ブローカ（仲介者、運営者、集中管理部など）が担っていた役割を、それぞれのエンティティがボランティアとして分担することにより、特定の運営者、管理者、集中管理部の存在を前提としなくても、様々なネットワークサービスを構築、運営できること
- ボランティアとして運営に参加している任意のエンティティが、障害や退去などの様々な理由により、運営から脱退しても、ネットワークサービス全体に影響を与えないように、残されたエンティティが自律的に自己組織化することによりサービスを継続できること[3]

SIONet は、互いに相応しい人が、自律的に、ダイナミックに、フレキシブルに、ゆるやかに、グループ（コミュニティ）を形成することを支援する次世代コミュニケーションツールであり、そのキーコンセプトは「緩やかさの追求」と「局所性の追求」にある。以下では、SIONet がどのような技術で、ブローカレス型探索モデルを実現しているかについて解説する。特に、以下の項目について述べる。

(1) 自己組織化機構
  - 自律性、増殖、縮退
  - ブローカレス（集中管理部が存在しない）
  - エンティティの寿命（ライフタイム）は短いことを前提
  - セキュアー、スケーラブル、耐障害性、低コスト（設備投資が少ない）

(2) アドバタイズメントとディスカバリ
  - ベースイベントプレースに参加しているエンティティによるエントランスの公開と探索（SIONet の入り口の探索）
  - エンティティプロパティの公開とディスカバリイベント
  - 最適なイベントプレースの芋づる式探索（緩やかな絞込み）

(3) 刺激と発火の連鎖反応
  - セッション、シェアードリンク
  - イベントパス（エンティティ間の相関関係）
  - 機能追加は、語彙概念の追加に相当（仕組みは連鎖反応のみ）

(4) プラグイン機構とアプリケーション共有（SE 共有）

(5) サーバ、ハイブリッド、ピュア P2P ネットワークの構築
  - エンティティの運営・配置問題に帰着
  - 共通の仕組み（連鎖反応のみ）で様々な P2P ネットワークを実現

[3] エンティティグループへの参加・退去は、各エンティティの自律性に委ねられる。さらに、膨大な数のエンティティの各々に、規則性がなく、行動予測が困難であるブローカレス型探索モデルにおいては、エンティティの寿命が短いことを想定する必要がある。すなわち、IP ルータなどは、無限の寿命を持つ（安定的に運営される）ものとして見なすことができる。そのため、P2P の世界ではこの点に十分配慮した方式設計が重要になる。

3.　SIONet 概要

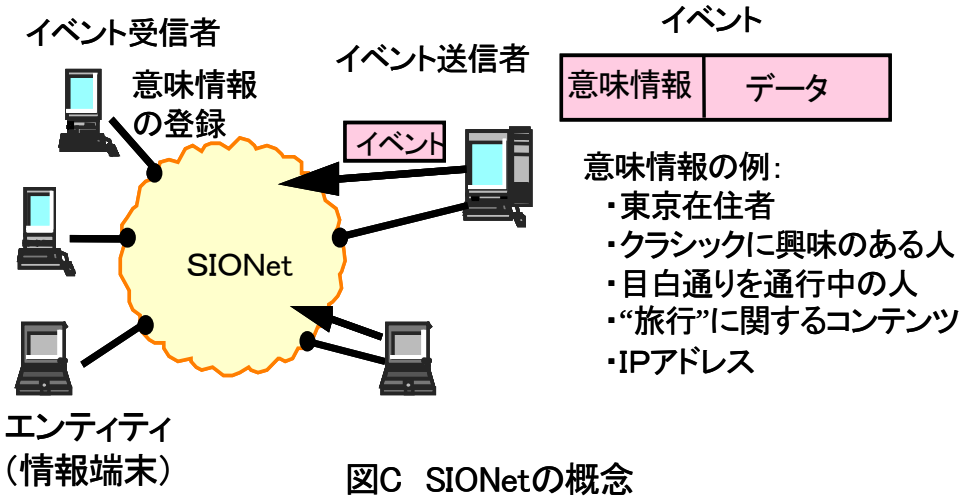
3.1　概念モデル

SIONet は、意味情報（メタデータ）に基づいてイベントを配送するメタネットワークであり、ネットワーク上に超分散する不特定多数のエンティティの中から、特定のエンティティを動的に探索・発見することができる。すなわち、SIONet は、従来のネットワークで用いていた宛先アドレス（誰に対して送信する）の代わりに、意味情報（どういう人に対して送信する）に基づいてイベントを配送するネットワークである（図 C)。

また、SIONet は、SIONet の構成要素を含めたすべてのエンティティが自律分散協調することにより、ネットワークが自己組織化される自律分散協調ネットワーク（自律分散コンピュータ）である。SIONet のネットワーク構成要素には、「意味情報スイッチ(SI-SW)」、「意味情報ルータ(SI-R)」、「意味情報ゲートウエイ(SI-GW)」、「イベントプレース」、「セッション」などがあり、これらが必要に応じて自己組織化することにより、セキュアーでスケーラブルな P2P ネットワークをボトムアップアプローチで構築することができる。

SIONet の基本概念・原理は単純で一元的である。“エンティティによるフィルタの登録とイベントの送出”という単純操作の繰り返し、すなわち、“エンティティの「刺激」と「発火」の「連鎖反応」”により、すべてのエンティティを自律分散協調させる点にある。この連鎖反応の振舞いを制御するものが「フィルタ」である。フィルタに登録するフィルタ値により、エンティティの連鎖反応の仕方を動的に制御することができる。SIONet におけるもう一つの基本概念がイベントプレースである。イベントプレースは、シェアードリンクにより、相互に接続されたエンティティグループであり、これにより連鎖反応の範囲を制限することができる。

具体的には、イベントルーティング、イベント転送モード（ユニキャスト、ブロードキャスト、マルチキャスト、属性付きマルチキャスト、リプライ通知)、ファイアウォール機構、エンティティのオンライン増減設機構、障害処理機構、統計情報収集機構などが、これらの考え方に基づいて設計･実装されている。



図C　SIONetの概念

3.2　SIONet エンティティの定義

パーソナルコンピュータ、ワークステーション、情報携帯端末、携帯電話、ウエアラブルコンピュータなどのあらゆるコンピュータを総称して、ホストと呼ぶ。さらに、SIONet ソフトウエアを実装したホストを、「SIONet エンティティ」もしくは単に「エンティティ」と呼ぶ。エンティティは、SIONet ソフトウエアを実装したホストを、SIONet における自律分散協調の単位として仮想化したものである（図 1）。なお、SIONet ソフトウエアは、個々のエンティティが自律分散協調を行うための仕組みを提供する。すなわち、各ホストが自律分散コンピュータとなる。SIONet では各エンティティが自律分散協調することによりエンティティグループ（イベントプレース）を形成する。

エンティティは、図 2 に示すように主に 3 つのタイプに分類される。

- サービスとして振る舞うエンティティ（サービスエンティティ:**SE**）
- ネットワーク構成要素として振る舞うエンティティ（ネットワークエンティティ:**NE**）
- サービス、および、ネットワーク構成要素の両者の観点から振る舞うエンティティ（サービス・ネットワークエンティティ:**SNE**）

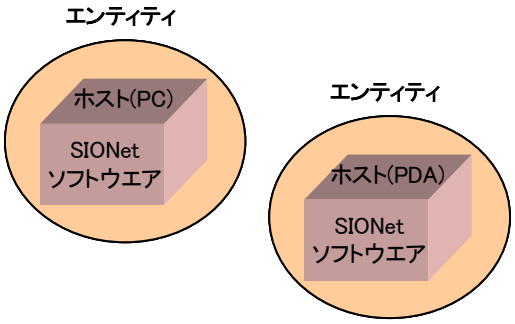


図1　SIONetエンティティ

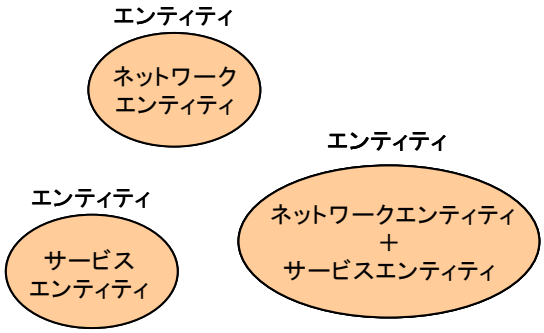


図2　エンティティの分類

3.3　エンティティ構造と状態遷移

図 3 にエンティティの状態遷移を示す。“Non-Existent”状態は、ホストに SIONet ソフトウエアをインストールした時点でのエンティティ状態を表している。

SIONet ソフトウエアを実行（ラウンチ）することにより、“Non-Existent”状態から“Suspend”状態に遷移する。このとき、エンティティ内に、エンティティ制御部、コントロールパネル、ネットワークエンティティファクトリ、ネットワークエンティティ、などの内部モジュールが生成される（図 4）。以下に、それぞれの役割を簡単に説明する。

① エンティティ制御部：エンティティの実体であり、他のエンティティやコントロールパネルからの様々な要求（後述するセッションの確立要求、シェアードリンクの確立

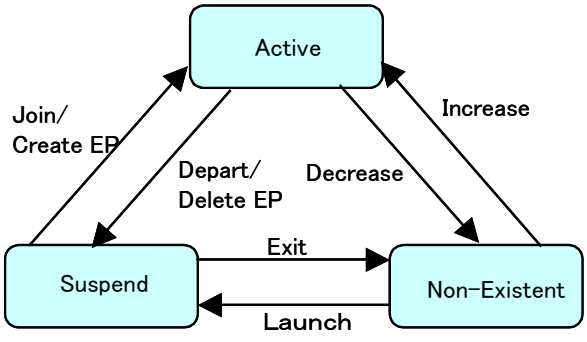


図3　エンティティの状態遷移

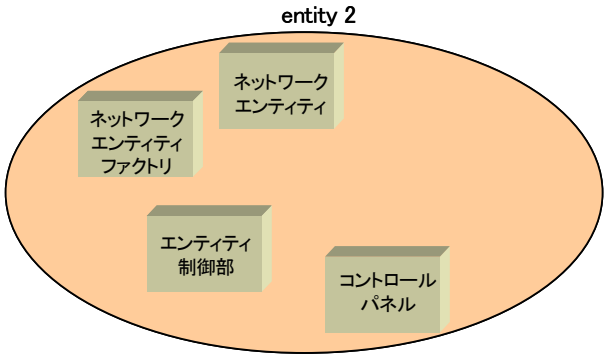


図4　エンティティの内部構造

要求など）を受け付ける。なお、“Suspend”状態のエンティティに対して、他のエンティティからセッションの確立要求、シェアードリンクの確立要求などを行うことはできない。

② コントロールパネル：人などのエンティティの所有者に対して、GUI を提供する。例えば、エンティティにエンティティ名を付与することができる。図 4 においては、“エンティティ 2”と命名している。このエンティティ名を、後述する「グローバルエンティティ名」と特に区別して「ローカルエンティティ名」と呼ぶことがある。

③ NE ファクトリ：NE を動的に生成するためのファクトリ

④ NE（ネットワークエンティティ）：SI-R、SI-GW、アライブエンティティ、障害処理エンティティ、統計情報収集エンティティなど、ネットワーク制御のために振る舞うエンティティ

⑤ SE（サービスエンティティ）：エンティティが提供するプラグイン機構を用いて、エンティティ内に SE として組み込まれたアプリケーション。SE は、SIONet 上で動作する P2P アプリケーションを仮想化したものである。エンティティが提供するプラグイン機構を用いて、アプリケーションを SE としてエンティティ内に組み込むことができる（図 11 参照）。また、SE は、SI-SW に対して、任意数のセッションを確立することができる。SE はセッションを介してのみ、イベントの送受信が可能である。SE は、SIONet のネットワーク構築･運営に関与しない。なお、SE をエンティティ内に組み込むプラグイン機構について後述する。

“Suspend”状態のエンティティ（エンティティ 2）が、自らイベントプレースを生成したとき（図 5）、既存のイベントプレースに Join（参加）したとき（図 6）、など任意のイベントプレースに属し、エンティティグループのメンバーになったときに、エンティティ（エンティティ 2）は“Suspend”状態から“Active”状態へと遷移する。“Active”状態のエンティティのみが自身の存在をアドバタイズし、他のエンティティからのセッションの確立要求やシェアードリンクの確立要求を受け付けることができる。

ここでは、図 5 に示すように、エンティティ 2 が“イベントプレース A”と命名したイ

ベントプレースを生成した場合を考える。このとき、エンティティ制御部は、NE ファクトリに SI-SW の生成を依頼し、生成された SI-SW と、NE などの内部モジュールとの間にセッションを確立する。

以下に SI-SW とセッションの役割について説明する。SI-SW（意味情報スイッチ）は、意味情報に基づいて、イベントをスイッチングするスイッチング機構を提供する。SI-SW はセッションを介して、NE や、プラグイン機構によりエンティティ内に組み込まれた SE をスター型で収容する。以下では、SE を例にとり説明するが、NE などの他の内部エンティティも同様である。

「セッション」とは、SI-SW と SE 間のコネクションであり、SE は、セッションを介してのみイベントの送受信を行うことができる（図 7）。セッションには、イベントの送信セッション、受信セッション、送受信セッションの 3 タイプがある。

図 7 に示すように、SE は、受信セッションを介して、イベントの取得条件を SI-SW に登録する。これを「フィルタ」と呼ぶ。フィルタには、取得したいイベントの語彙概念（イ

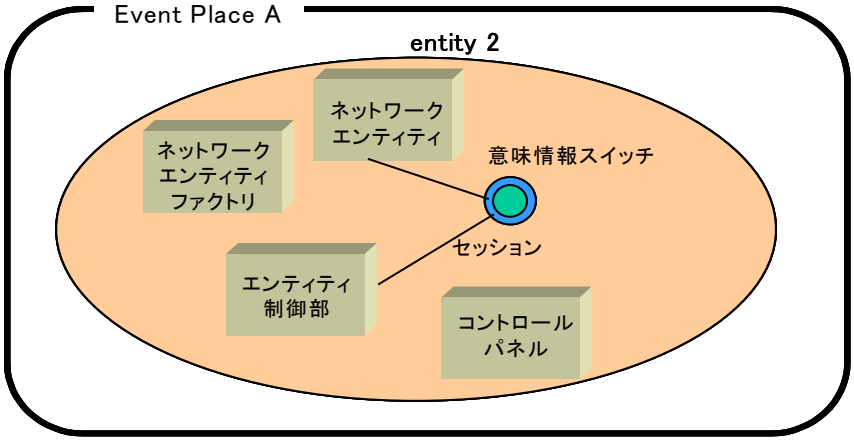


図5　イベントプレースの生成

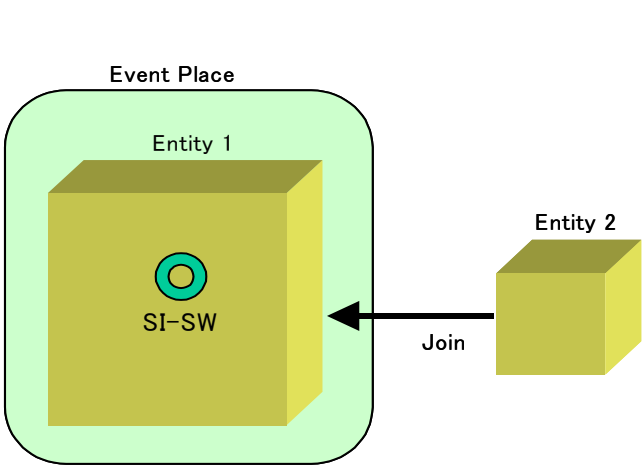


図6　エンティティのイベントプレースへの参加

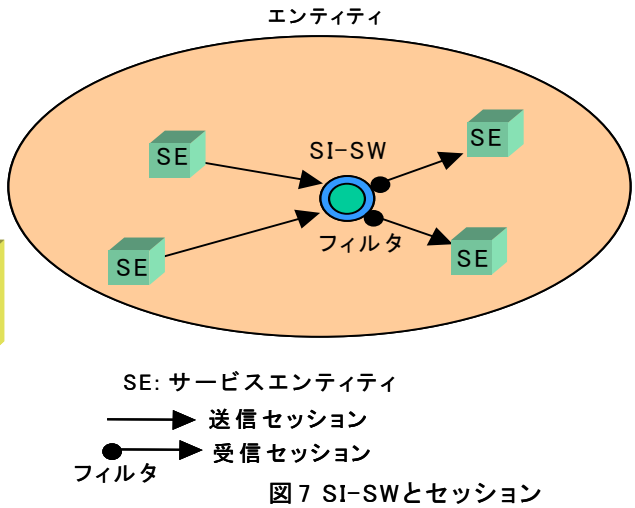


図 7 SI-SWとセッション

ベントタイプ)、および、意味情報との照合条件（例えば、語彙“Price”が“$20 から$40 の範囲”のものを取得対象とする）を設定する。なお、当該イベントタイプに定義されているすべての語彙との「完全一致」、一部の語彙との「部分一致」、「重み付け一致」など、照合条件をフィルタ単位で選択することができる。

一つの受信セッションから、複数のフィルタを登録可能であるが、同一セッションを介して登録されたフィルタ間は“or”関係を有する。すなわち、一つのイベントに対して、一つの受信セッションは高々1 回しか発火しない。取得したいイベントのイベントタイプにワイルドカードを指定した場合には、すべてのイベントタイプが取得の対象となる。また、意味情報との照合条件に 1（論理値が真）が設定された場合には、無条件に意味情報との照合条件が満足されたことを意味する。

一方、SE は、送信セッションを介して、SI-SW にイベントの送信を行う。このとき、SI-SW はイベントの意味情報部とフィルタとを照合する。具体的には、まず、受信したいイベントタイプであるかどうかをチェックし、これを満足した場合には、意味情報と照合条件をチェックする。照合の結果、条件を満足した場合には、合致したフィルタを登録したエンティティを起動するとととともに、当該イベントを通知する。

SIONet では、イベントの送信を「刺激」、フィルタがイベントに合致することを「反応」、合致したフィルタを登録したエンティティを起動しイベント通知することを「発火」と呼ぶ。

イベントは、制御情報部、意味情報部、データ部から構成される（図 8)。データ部には、送信データが格納される。送信データとして、テキストデータ、バイナリデータ、リファレンス、プロキシ、エージェントなど、様々なデータ、プログラムを格納することができる。

意味情報部には、送信データの意味情報（語彙とその値)、および、意味情報の語彙概念（イベントタイプ）が格納される。ここで、意味情報とは、送信データの特性を記述したメタデータであり、イベントタイプのインスタンスである。一方、イベントタイプは意味情報のテンプレートである。イベントタイプ間には継承関係を定義できる。

図 9 に、意味情報体系の一例を示す。意味情報体系の記述言語としては XML などがある。なお、ネットワークエンティティ用に一部のイベントタイプ名（SIONet から始まるすべての名称）が予約済みである。

制御情報部は、SIONet の実行制御のために用いられる制御フィールドであり、これのみが SIONet のユーザである SE に対して開放されていない。制御情報部には、合致したフィルタの識別子、合致したフィルタの照合得点、同期型統計情報収集フラグ、TTL 値（イベントプレース内、イベントプレース間)、ホップ数（イベントプレース内、イベントプレース間)、ホップ属性などの制御情報が設定される。

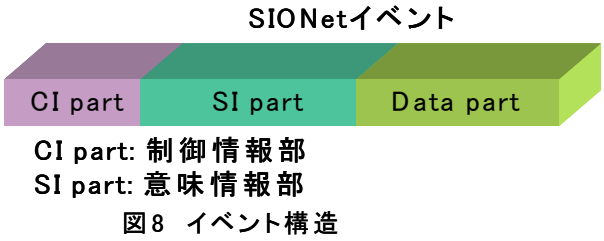


図8　イベント構造

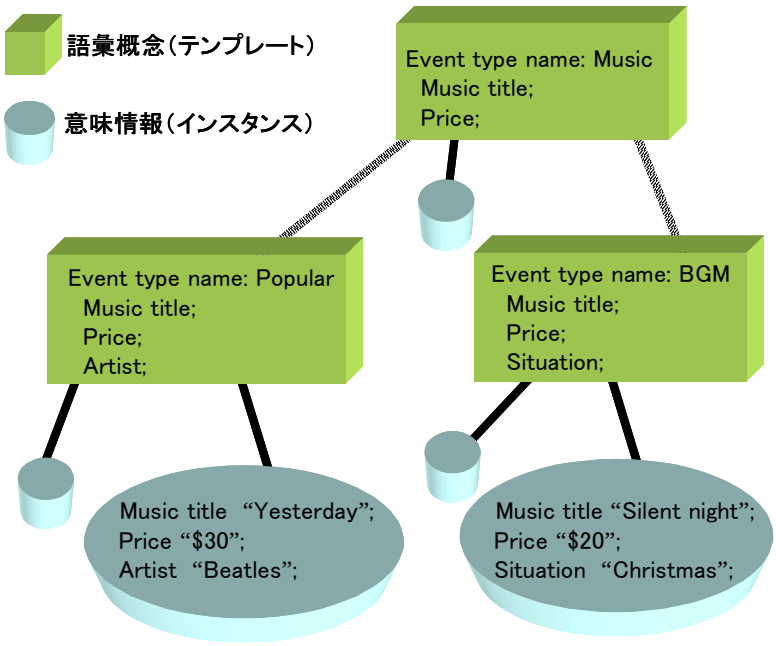


図9　意味情報体系の例

3.4　エンティティの名称付与方法

図 5 に示すように、エンティティ 2 が“イベントプレース A”と命名したイベントプレースを生成した場合を考える。このとき、前述したように、エンティティ制御部は、NE ファクトリに SI-SW の生成を依頼し、生成された SI-SW と、NE などの内部モジュールとの間にセッションを確立する。さらに、生成された SI-SW の名前を「イベントプレース名+ローカルエンティティ名」、すなわち「イベントプレース A+ エンティティ 2」として記憶する。この名前を「グローバルエンティティ名」と呼ぶ。このグローバルエンティティ名の作成がイベントプレースの生成に相当する。すなわち、SIONet では、イベントプレースの生成により、必ずしも何らかの管理実体が生成されるとは限らない。つまり、SIONet では、イベントプレースに属しているエンティティのメンバー管理を、後述するエンティティ間のシェアードリンクにより実現しているため、メンバー管理を行う集中管理部自体が存在しない。そのため、イベントプレースを生成したエンティティが当該イベントプレースから退去しても、残されたエンティティが自律的に自己組織化することにより、イベントプレースの運営が継続される。すなわち、イベントプレースから最後のエンティティが退去することが、イベントプレースの消滅に相当する。

グローバルエンティティ名の役割を以下に示す。図 5 に示したエンティティ 2 がさらにイベントプレース B を生成した場合を考える（図 10）。前述と同じ手順で、エンティティ 2 内部に新たな SI-SW が生成され、生成された新たな SI-SW にグローバルエンティティ名として「イベントプレース B +エンティティ 2」が付与される。すなわち、エンティティ 2 は二つのグローバルエンティティ名（SI-SW 名）を持つ。ここで、エンティティ 1 がイベントプレース A に Join する場合を考える。エンティティ 1 が、エンティティ 2 に対して Join 要求を行うとき、エンティティ 2 のグローバルエンティティ名が「イベントプレース A + エンティティ 2」である旨を通知する。これにより、シェアードリンクを確立する SI-SW を一意に特定することが可能になり、ひいては、イベントプレース A への参加が可能になる。このように、シェアードリンクやセッションの確立先である SI-SW はグローバルエンティティ名として仮想化されるため、SI-SW を直接意識することはない。なお、“Active”状態のエンティティは、自身の存在を公開することができるが、その際にはグローバルエンティティ名が公開されることになる。

3.5　SE のプラグイン方法

以下に、エンティティへの SE の組み込み方法を示す（図 11）。

(1) エンティティの所有者は、コントロールパネルに対して、アプリケーションのプラグインを指示する。このとき、プラグインするアプリケーションの実行ファイル名をパ

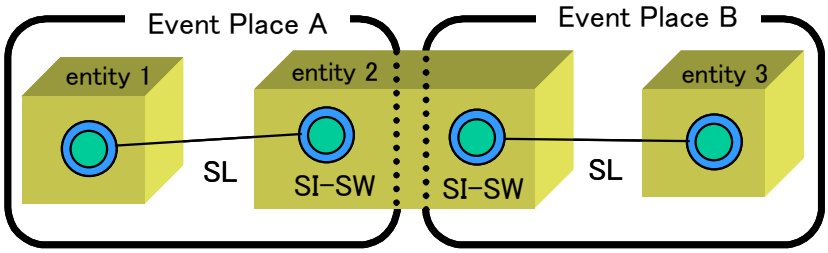


図10　グローバルエンティティ名

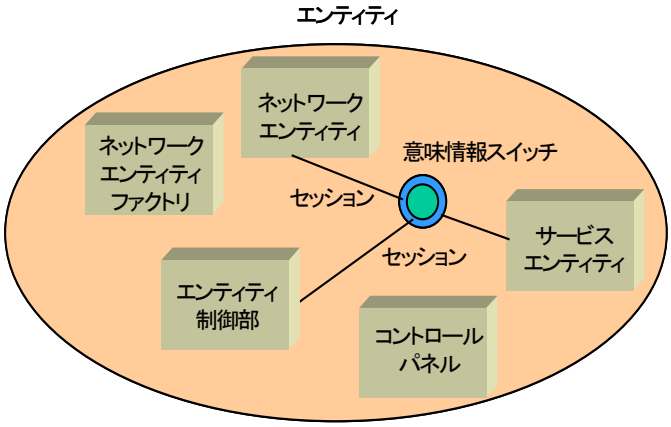


図11　サービスエンティティのプラグイン

ラメータとして与える。なお、アプリケーションプログラムの実体がファイルであるかどうかは実装依存である。

(2) コントロールパネルはこの旨をエンティティ制御部に通知する。

(3) エンティティ制御部は、プラグインする実行ファイル名を記憶するとともに、与えられた実行ファイル名を用いて、アプリケーションを起動し、SI-SW とアプリケーション間にセッションを確立する(アプリケーションにセッションを払い出す)。すなわち、SIONet におけるプラグインとは、アプリケーションと SI-SW との間にセッションを確立し、アプリケーションや COMNet などのミドルウエアを、エンティティ内の内部エンティティ(SE)として仮想化することに他ならない。すなわち、エンティティを汎用ゲーム機として、プラグインされた SE をゲームソフトとして見なすことができる。SIONet では、SE、NE などのすべての動作実体は、セッションを介して連携する。そのため、プラグイン/プラグアウト（着脱）を容易に実現できるとともに、プラグイン/プラグアウトが他の動作実体（内部エンティティ）に影響を与えない（超疎結合)。

(4) この考え方を拡張することにより、後述するようなエンティティ間での SE 共有が実現できる。

3.6 PREFERENCE アーキテクチャ

SIONet は図 12 に示すようにリファレンスモデルの考え方を採用している。[4] SIONet はイベントの伝達層であり、上位層に対してネットワークインタフェースを提供する。ミドルウエア層がコミュニティ（コミュニティネットワーク: COMNet）であり、SIONet のインテリジェンス層に相当する。このレイヤにおいて、情報の局所化、認証（信頼性、信用度、価値観など)、セキュリティ、インセンティブ（名誉、貢献度、寄付など)、情報の権利保証、ポリシー制御、自動化制御などが行われる。そして、最上位層がアプリケーション層（SE）となっている。SE としては、Personal TV Service（個人 TV 局)、Smart Messenger（メッセージ交換サービス）などがすでに実装されている。SIONet は、分散コンピューティング、情報交換、コラボレーション、メッセージ配信などのあらゆる P2P サービス（SE）に対して、共通の P2P ネットワーク基盤を提供するとともに、プラグイン機構や SE 共有機構の提供により、アプリケーション開発の効率化をサポートする。

---

[4] PREFERENCE アーキテクチャは、御用聞き社会（御用聞き型サイバーソサイアティ）構築に向けたアーキテクチャであり、ブローカレス型配信モデル、ブローカレス型探索モデル、ブローカレス型ポリシーモデルに対するトータルソリューションを提供する。具体的には、ストリームインタフェース、SIONet、COMNet などから構成される。これは、誰もが、自由に、相応しい人と出会うことが可能な仮想社会の構築を目指すものであり、SIONet はそのための次世代コミュニケーションツールとして位置付けられる。

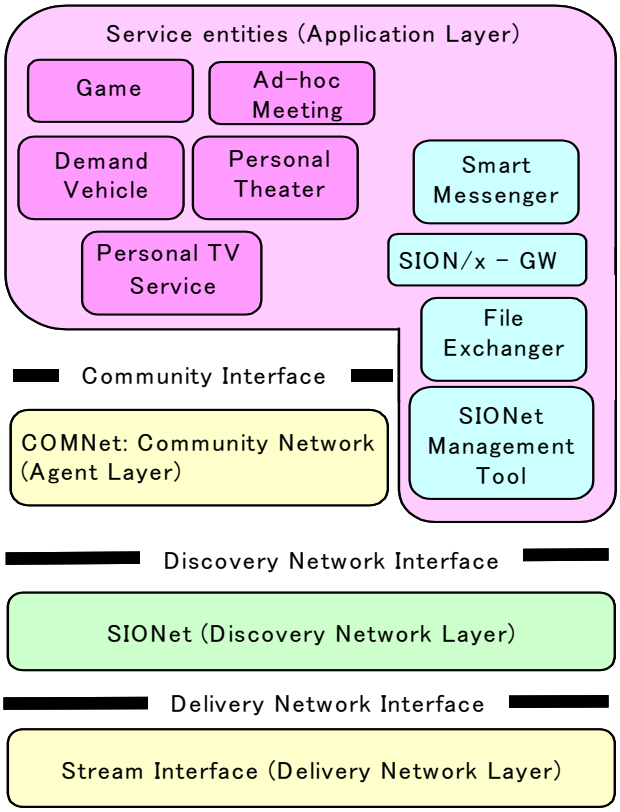


図12　PREFERENCEアーキテクチャ（リファレンスモデル）

3.7　意味情報ルータとシェアードリンク

SIONet は、個々のエンティティが互いに助け合うことにより、ボランティア型のネットワークを構築する形態であり、それぞれが自律分散協調することにより、ネットワークを自己組織化する。例えば、図 6 は、エンティティ 1 がイベントプレースを生成している状況を表している。このとき、エンティティ 2 がエンティティ 1 に対して、イベントプレースへの Join 要求を行うと、図 13 に示すように、エンティティ 2 に SI-SW が生成され、

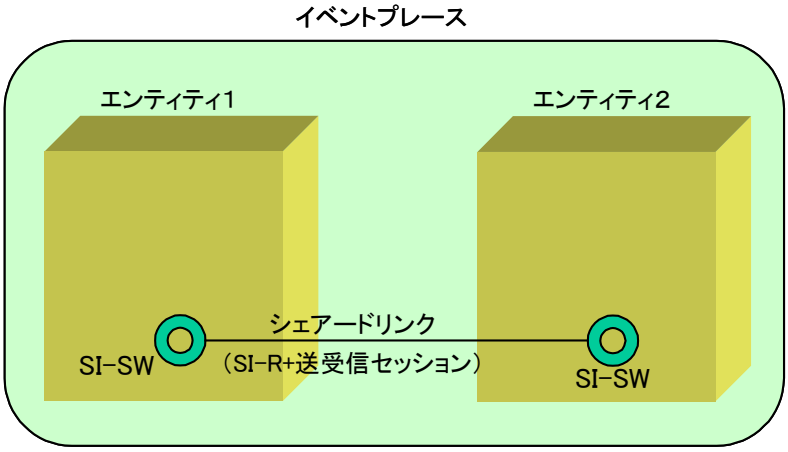


図13　エンティティのグループ化（シェアードリンク）

さらに、SI-SW 間にシェアードリンクが確立され、エンティティグループが形成される。このとき、イベントプレースの生成者であるエンティティ 1 がイベントプレースから退去すると、SI-SW 間のシェアードリンクが解除され、以降は、エンティティ 2 のみでイベントプレースの構築･運営が継続される。ここで、シェアードリンクは、エンティティグループを形成するための仕組みであり、マルチホップの経路でもある。

SIONet では、目的に応じてエンティティを配置･組み合わせることにより、様々な形態の P2P ネットワークを構築することができる。また、異なる形態の P2P ネットワークをシームレスに連携させることもできる。これは、

- すべてのエンティティを、“刺激と発火に基づく連鎖反応”という一元的かつ単純な仕組みで自律分散協調させること、
- P2P ネットワーク形態の差異をエンティティの配置・運営問題に帰着させること、

などにより達成される。このように、単一の仕組みで様々な形態の P2P ネットワークを構築できる点が SIONet の特徴の一つである。

「シェアードリンク（共有リンク）」は、複数の異なるエンティティ間において、双方向のイベント共有を行うための概念である。例えば、図 14 に示すように、エンティティ 2 がエンティティ 1 に対して、シェアードリンク(SL: Shared Link)の確立要求を行うことにより、図 15 に示すように、SI-SW2 と SI-SW1 との間にシェアードリンクが確立され、新たなエンティティグループが形成される。シェアードリンクの確立に成功したエンティティ 2 に対し、エンティティ 1 からイベントプレース名（グローバルエンティティ名）、当該イベントプレースにおけるイベント転送方式（ルーティング方式）、当該イベントプレースのディスクリプション（説明文）、プラグインされている SE 情報、リンクトポロジー等のさまざまな情報が返却される。なお、後述する SI-R によるフィルタ値の設定によって、SI-SW 間に様々なイベントルーティング方式を動的に設定できる。以降において、SI-R の役割、仕組みを詳述する。

3.7.1 シェアードリンクの確立

図 14 および図 15 を用いて、シェアードリンク確立までの仕組みを説明する。

① 図 14 において、エンティティ 2 がエンティティ 1 に対して、シェアードリンクの確立要求を発行した場合を考える。
② このとき、図 15 に示すように、エンティティ 1 は、$SL_{1,2}$ を確立するために、$SI\text{-}R_{1,2}$ を動的に生成するとともに、エンティティ 2 をシェアードリンクの確立要求元であることを記憶する。厳密には、エンティティ 2 のエントリポイントとグローバルエンティティ名(SI-SW2)を記憶する。ここで、SI-R は、意味情報に基づいて、SI-SW 間の

イベントルーティング（イベント転送）を行う「意味情報ルータ」である。

③ SI-$R_{1,2}$は、エンティティ 2 に対してイベント受信セッションの確立要求を行うことにより、SI-SW2 に対してイベント受信セッションを確立する。さらに、自身が属する SI-SW1 に対して、イベント送信のためのセッションを確立する。このように SI-R が確立する送受信セッションの組み合せをシェアードリンクと呼ぶ。

④ これにより、$SL_{1,2}$が確立され、例えば、SE4 が SI-SW2 に対して送出したイベントが、SI-$R_{1,2}$を介して、SI-SW1 へも送出される。

⑤ このとき、SI-$R_{1,2}$が SI-SW2 に対して登録するフィルタの設定値により、イベントの転送方式を動的に制御できる。例えば、

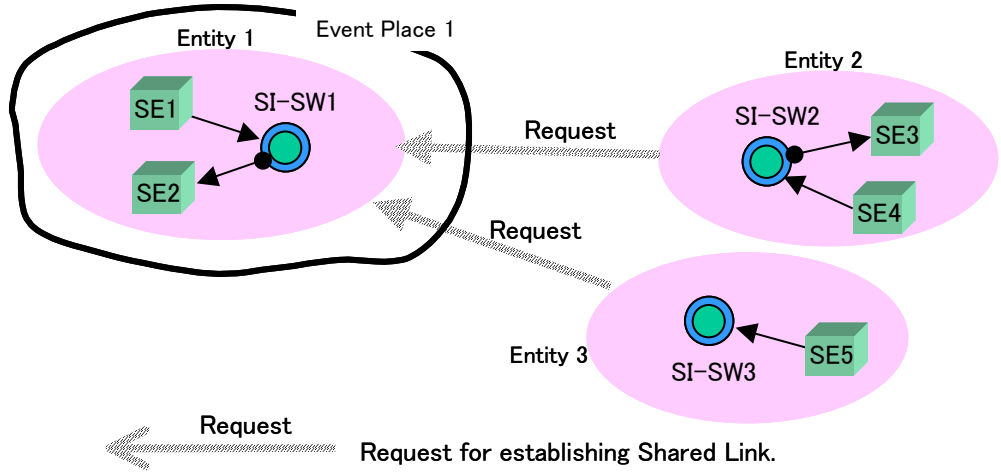


図14　SI-SW間のシェアードリンクの確立要求

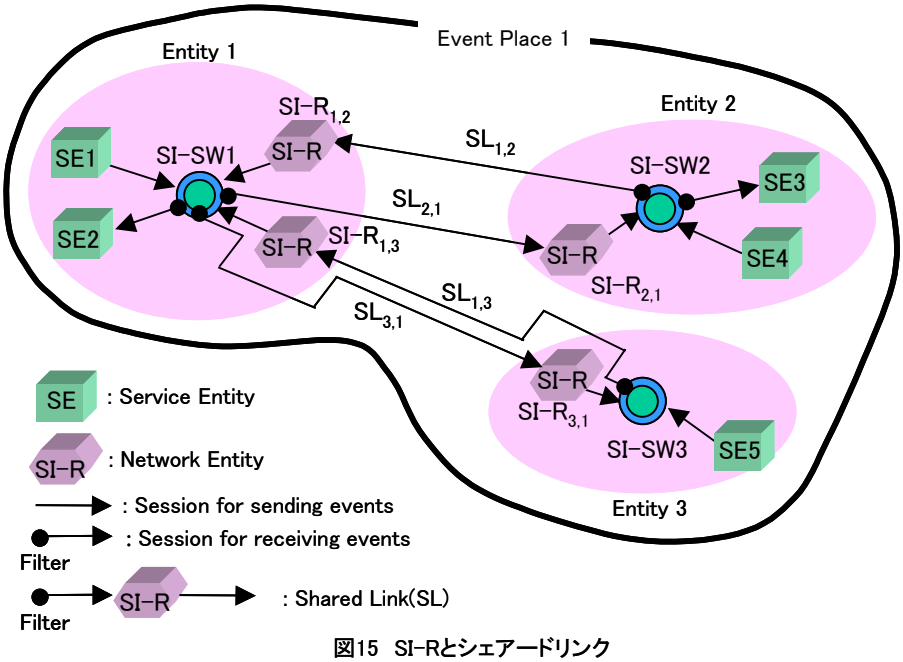


図15　SI-Rとシェアードリンク

(a) SI-SW2 に対して送出されたイベントを、無条件に SI-SW1 へと転送する。すなわち、すべてのイベントに対して、$SI\text{-}R_{1,2}$ が発火する。これは SI-R が SI-SW に登録するフィルタとして、「取得する語彙概念にワイルドカードを、および意味情報（語彙）との照合条件に真」を設定することにより可能になる。これを「無条件ルーティング」と呼ぶ。この設定は、シェアードリンク確立時に一度だけ行えばよい。この方式では、無駄なイベント転送と転送先でのイベント照合処理オーバヘッドが発生する可能性があるが、後述するイベントパス確立（ルーティング情報の設定）のためのオーバヘッドが発生しないため、フィルタ登録数がイベント送出数よりも十分大きい場合に有効な方式である。このルーティング方式は、主に、マルチホップ型ブロードキャスト通信に用いられる。

(b) エンティティ 1 に取って必要なイベントのみを SI-SW2 から SI-SW1 へ転送する。これにより、不要なイベント転送を行わない。すなわち、特定のイベントに対してのみ、$SI\text{-}R_{1,2}$ が発火する。これは、SI-R が語彙概念のみ（照合条件は常に真）をフィルタに設定する「語彙概念によるイベントルーティング」と、語彙概念と語彙との照合条件をフィルタに設定する「語彙によるイベントルーティング」に大別される。前者は主にマルチホップ型属性付きマルチキャスト通信に、後者は主にマルチホップ型ユニキャスト通信、マルチホップ型マルチキャスト通信に用いられる。これは、障害処理通知、統計情報通知、アライブ通知、リプライ通知、アドバタイズメント通知などで用いられている。これらについては以下にて詳述する。

⑥ ②から⑤の手順は、シェアードリンクの確立要求元であるエンティティ 2 においても同様に行われる。すなわち、エンティティ 2 は、エンティティ 1（SI-SW1）をシェアードリンクの確立要求先として記憶するとともに、$SL_{2,1}$ を確立するために、$SI\text{-}R_{2,1}$ を動的に生成し、以降の処理を同様に行う。これにより、SI-SW1 と SI-SW2 の間に、双方向のシェアードリンクが確立され、エンティティ間でイベントを互いに共有することが可能になる。なお、SI-R がフィルタの登録を行わないことにより、片方向のシェアードリンクを確立すること、すなわち、SI-R を発火させないことができる。

3.7.2 イベントパスの確立

イベント転送（イベントルーティング）のための経路選択情報（SI-R がイベント共有のために登録するフィルタの集合）を「イベントパス」と呼ぶ。[5] 例えば、図 15 において、

[5] シェアードリンクとイベントパスを用いたマルチホップ通信（ユニキャスト、ブロードキャスト、マルチキャスト、属性付きマルチキャスト、リプライ通知）のほかに、通信先エンティティのエントリポイントを用いた直接通信も可能である。SIONet では、SI-R が設定するフィルタ値の違いだけで、エンティティ間の連鎖反応の仕方を動的に制御することができる。すなわち、前述した様々な通信モードを実現することができる。このような単純で単一な仕組みも SIONet の大きな特徴の一つである。

SE3 が SI-SW2 に対してフィルタを登録した場合を考える。このとき、$SI\text{-}R_{2,1}$ は SE3 が登録したフィルタを、受信セッションを介して SI-SW1 に登録する。同様に、$SI\text{-}R_{1,3}$ は当該フィルタを、受信セッションを介して SI-SW3 に登録する。このように SE3 によるフィルタ登録をトリガに、SI-R が登録したフィルタがシェアードリンクに基づいて、伝言ゲームのように順次隣接する SI-R に波及することにより、イベントパスが確立される。これを「**イベントパスの設定、もしくは波及**」と呼ぶ。

「語彙概念によるイベントルーティング」のためのイベントパスを確立するためには、SE3 が SI-SW2 に対して登録したフィルタにおいて、$SI\text{-}R_{2,1}$ は、イベントタイプのみ（照合条件は常に真）を SI-SW1 へのフィルタとして設定する。すなわち、SE3 がフィルタ登録した照合条件を活用しない。つまり、イベントタイプのみを活用することにより、例えば、SE1 がイベント送信したとき、イベントタイプのみを SI-SW1 において照合し、その結果、$SI\text{-}R_{2,1}$ が発火すると、$SI\text{-}R_{2,1}$ は当該イベントを SI-SW2 へと送出し、意味情報との照合を SI-SW2 において行う。

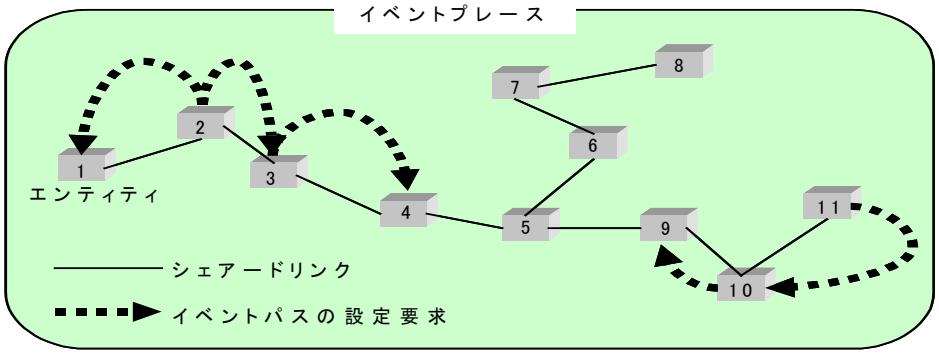
一方、「語彙によるイベントルーティング」のためのイベントパス確立に際しては、$SI\text{-}R_{2,1}$ は、SE3 が登録したフィルタ値を、そのまま SI-SW1 へのフィルタ値として設定する。すなわち、SE3 がフィルタ登録したイベントタイプと意味情報の両者を $SI\text{-}R_{2,1}$ が SI-SW1 へとフィルタ登録し、SI-SW1 において完全なフィルタリングを行う形態である。

そのため、語彙によるイベントルーティング方式では、無駄なイベント転送が全く発生しないが、ルーティングのためのイベントパス設定オーバヘッドが膨大になるため、フィルタ登録数に比べてイベント送出数が十分大きい場合に有効な方式である。一方、イベントタイプに基づくルーティング方式は、前述の 2 方式の折衷案的な位置付けにある。すなわち、フィルタ登録数とイベント送出数が同程度の場合、もしくは、フィルタ登録数とイベント送出数の比率を予測できないような利用形態において有効である。これらの仕組みにより、意味情報に基づく、イベントのマルチホップ通信が実現される。なお、イベントプレースの生成時、もしくは、SE からのフィルタ登録時に、イベントルーティング方式を指定できる。

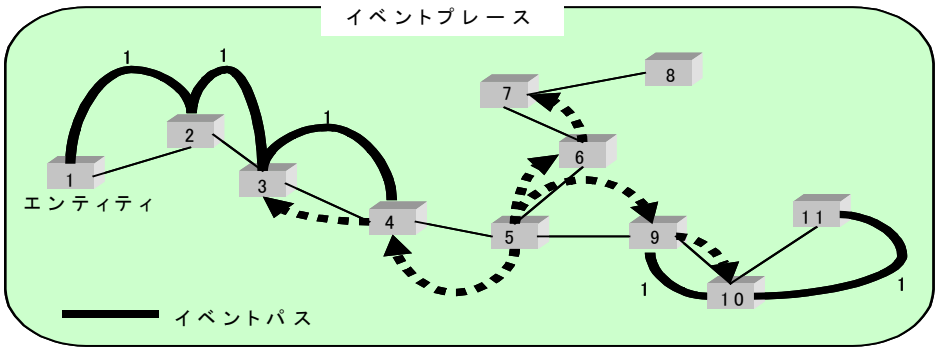
イベントパスの設定要求はイベントプレース内のすべての SI-R に対して波及するが、TTL 値により、その波及範囲を制限することができる。図 16 を用いて簡単に説明する。ここでは説明の便宜上、語彙概念ルーティングのためのイベントパス確立を前提とする。また、各エンティティは、同じ語彙概念をアドバタイズ（フィルタ登録）するものとする。図 16(a)において、エンティティ 2（厳密には、エンティティ 2 に属する SI-R）がイベントパスの設定要求を開始する。ここでは、TTL 値を 2 としたため、エンティティ 1、エンティティ 3、そしてエンティティ 4 に対してのみイベントパスの設定要求が波及し、その結果、図 16(b)に示すようにイベントパスが確立される。同様に、エンティティ 11 がイベントパスの設定要求を開始することにより、エンティティ 10、およびエンティティ 9 に対してイベントパスが確立される。そのため、エンティティ 2 とエンティティ 11 の間では、

イベントの共有が行われない。つまり、エンティティ 2 から生起したイベントでは、エンティティ 5～11 は連鎖反応しない。そのためエンティティ 5～11 へはイベントが転送されないことになる。

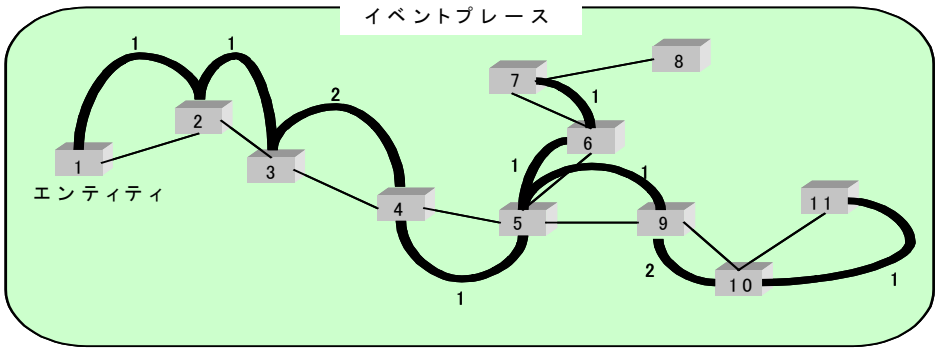
しかしながら、図 16(b)に示すように、エンティティ 5 がイベントパスの設定要求を開始することにより、エンティティ 4、エンティティ 5、エンティティ 6、エンティティ 7、そしてエンティティ 9 に対してイベントパスが確立される。結果的に、図 16(c)に示すように、エンティティ 8 を除く、イベントプレース内のすべてのエンティティに対してイベントパスが確立される。このように、当該語彙概念が当該イベントプレースにおいて、使用頻度の高いもの（評判が高い、流行しているもの）であるならば、最終的に、イベント



(a) イベントパスの設定要求



(b) イベントパスの確立



(c) イベントパスの融合

図 16　イベントパス

プレース内にイベントパスの確立が行き渡ることになる。その逆で、あまり流行していない語彙概念は、いずれ淘汰されることになり、これにより緩やかな連鎖反応を実現できる。なお、イベントパス上の数字は、その多重度を示している。[6]

分散オブジェクト技術の観点から、イベントパスを以下のように解釈することもできる。世界中に超分散しているエンティティは、エンティティ間でなんらかの「相関関係」を有する。相関関係を与えるものとしては、エンティティ名、グループ名、属性（位置、興味、評判、流行、サービス）など様々なプロパティがある。エンティティは、相関関係に基づいて、エンティティ間の結びつきを持ち分散協調する。SIONet では、この相関関係を語彙概念と語彙により表現し、シェアードリンクに基づいて、イベントパスとして設定する。また、相関関係の強弱が、語彙概念の汎化･特化、イベントパスの多重度などに相当する。すなわち、SIONet では、イベントパスにより、エンティティ間の相関関係を動的に制御･管理している。これこそが、フィルタによる連鎖反応の制御である。そのため、エンティティは固定的なエンティティ識別子を有していない。例えば、IP アドレスは、位置に基づいたエンティティの固定的な識別子であるが、SIONet では、これの代わりに、エンティティプロパティを語彙概念･語彙として記述したものをエンティティ識別子として用いる。エンティティは、これらをフィルタとして SI-SW に登録することにより、エンティティのプロパティ（エンティティ識別子）を宣言するとともに、自身のプロパティをシェアードリンクに基づいてアドバタイズする。これがイベントパスの波及に相当する。これにより、イベントパスが確立される。そのため、エンティティの匿名性、プライバシーが保証される。すなわち、誰がフィルタを登録したかを隠蔽するとともに、物理的な ID（例えば、IP アドレス）を漏洩しない。

これまでの説明から明らかなように、SI-R は、イベント送信とイベント受信の両者の側面を持つネットワークエンティティであり、一般のサービスエンティティと本質的な違いはない。SIONet では、SI-R のような SIONet の制御に用いられるエンティティを、サービスエンティティと特に区別して、ネットワークエンティティと呼ぶ。SIONet においては、サービスエンティティ、ネットワークエンティティのすべての内部エンティティを共通の内部エンティティとして扱い、さらに、イベント送出とイベント受信、すなわち刺激と発火の連鎖反応という単純かつ一貫性のある共通ロジックに従って自律動作させることにより、すべてのエンティティが自律分散協調可能な超分散・超疎結合アーキテクチャを提供する。

---

[6] エンティティは、イベントパスの多重度をエンティティプロパティの属性として公開することができる。一方、エンティティプロパティのディスカバリにより、イベントパスの多重度が高いエンティティを発見することができる。このとき、当該エンティティに対して、シェアードリンクの張り替えを行うことにより、同好の志（同好のエンティティ）が緩やかに近傍に集まることができる（同好エンティティの局所化）。

3.7.3　シェアードリンクの解除と再確立

イベントプレースに参加しているエンティティが、障害に陥ったり、イベントプレースから退去する場合など、様々な理由により当該エンティティがネットワークの運営に関わることができないケースにおいては、残されたエンティティが自己組織化することによりネットワークサービスを継続できることが必要になる。

SIONet では、このようなケースにおいては、SI-SW 間において確立されているシェアードリンクを解除し、さらにシェアードリンクの再確立を行う。以下に、その仕組みを紹介する。

図 17 において、シェアードリンク確立要求の先頭数字は、それらの要求順序を示している。つまり、エンティティ 2 からエンティティ 1 へシェアードリンクの確立要求を行う。次にエンティティ 3 からエンティティ 2 へシェアードリンクの確立要求を行う。最後に、エンティティ 4 からエンティティ 2 へシェアードリンクの確立要求を行う。なお、ここで、「各エンティティは、同一イベントプレース内においてシェアードリンクの確立要求を高々 1 回しかできないが、無制限に確立要求を受け付けることが可能なリンクトポロジーに基づいて、シェアードリンクの確立を行う」リンク確立手法を提案する。これにより、シェア

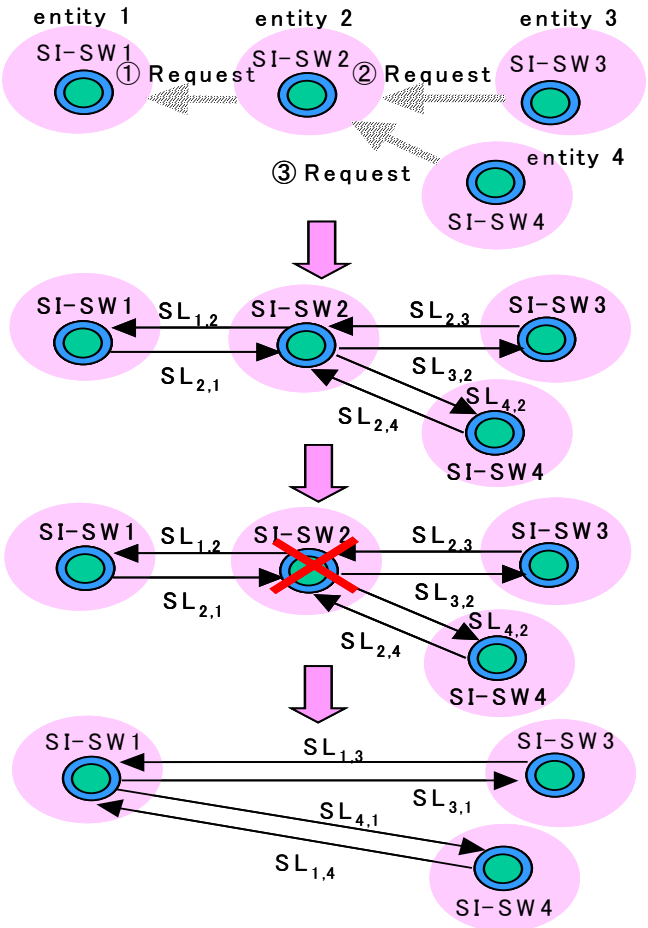


図 17　シェアードリンクの再確立

ードリンクで結合されたエンティティ間に、ループが発生しないことが保証される。すなわち、この手法を用いることにより、隣接エンティティ間のリンク再確立のみで開リンクトポロジーを簡単に実現できる。

この確立要求が順次成功したとき、前述した手順で各 SI-SW 間にシェアードリンクが確立される。このとき、例えば、エンティティ 2 は、自身が確立要求を行ったエンティティ 1 と、自身への確立を受け付けたエンティティ 3, エンティティ 4 のリストを保持している。

この状況において、例えば、エンティティ 2 の退去、減設などが行われる場合、エンティティ 2 は SI-SW2 に対して確立されているシェアードリンクの解除要求を発行し、シェアードリンクの解除を行う。その後、各エンティティは新たなシェアードリンクの確立を行う。以下に手順を示す。

(1) エンティティ 2 が退去する場合を考える。
(2) エンティティ 2 は、自身のシェアードリンクを解除する旨を、1 ホップのエンティティ（エンティティ 1, エンティティ 3, エンティティ 4)に対して通知する。これは、イベントの TTL 値を 1 にしてイベント送出を行うことにより可能になる。なお、このとき、自身へのシェアードリンク確立を受け付けたエンティティ(エンティティ 3, エンティティ 4)に対しては、自身に代わる新たなシェアードリンクの確立要求先として、自身が確立要求を行ったエンティティ 1 を教える。なお、自身が確立要求を行ったエンティティが存在しない場合には、自身への確立要求元エンティティ（1 ホップ先のエンティティ）の中から任意のエンティティを選択して、これを自身に代わるシェアードリンクの確立要求先とする。その後、エンティティ 2 は、自身が確立したシェアードリンク($SL_{2,1}$, $SL_{2,3}$, $SL_{2,4}$)を解除するために、その旨を $SI\text{-}R_{2,1}$, $SI\text{-}R_{2,3}$, $SI\text{-}R_{2,4}$ に通知する。これらの SI-R は、これを契機に SI-SW へのセッションを解除する。
(3) 一方、エンティティ 3、エンティティ 4 は、自身が SI-SW2 に対して確立していたシェアードリンク($SL_{3,2}$, $SL_{4,2}$)をすべて解除し、新たなエンティティ(エンティティ 1)に対してシェアードリンクの確立要求を行い、シェアードリンクを再確立する。なお、エンティティ 1 は、$SL_{1,2}$ を解除し、エンティティ 3、エンティティ 4 からの確立要求を待つ。上述したように、シェアードリンクの確立処理、再確立処理は、隣接するエンティティ間においてのみ行われ、他のエンティティには影響を与えない。すなわち、全てのエンティティに対するリンクの再構築を必要としない。
(4) なお、エンティティの退去や減設などの正当な手順を踏んだシェアードリンクの解除要求ばかりでなく、エンティティの障害、電源断、セッション（物理的な通信路）の障害などに起因して、シェアードリンクを再確立しなければならないケースがある。しかし、このようなケースにおいては、シェアードリンクの再確立時に必要となる代わりのエンティティを教授できない可能性がある。SIONet では、このような状況に対応するために、各エンティティは n ホップ（n は任意）のイベントを送出すること

により、代わりの確立要求先エンティティの把握を行っている。なお、SIONet では、TTL 値とホップ属性を用いることにより、さらにきめ細かなホップ制御が可能である。代表的なホップ属性には以下のものがある。

① シェアードリンクの確立要求先エンティティのみをホップ対象とする。
② シェアードリンクの確立要求元エンティティのみをホップ対象とする。
③ すべてのエンティティをホップ対象とする。

例えば、①のホップ属性を指定した場合の、イベントの流れを図 18 の破線で示している。図 18 において、(i /j)は各エンティティにおいて記憶している、シェアードリンクの確立情報である。 i はエンティティがシェアードリンクの確立要求を行った先のエンティティを示しており、一方、j はエンティティがシェアードリンクの確立要求を受け付けたエンティティを示している。例えば、エンティティ 3 の(2/5,6)は、エンティティ 3 がエンティティ 2 に対して確立要求を行い、エンティティ 5 およびエンティティ 6 から確立要求を受け付けていることを表している。各エンティティがシェアードリンクの確立情報を保持することにより、例えば、エンティティ 3 が他のエンティティに対して、更なるシェアードリンクの確立要求を行った場合、その要求をエラーとしてリジェクトすることができる。これにより、前述した開リンクトポロジーの一貫性を保証している。この状況において、エンティティ 6 が 3 ホップで①のホップ属性を持つイベントを送出することにより、エンティティ 1 とエンティティ 2 の存在を知ることが可能になり、これらがエンティティ 3 障害時の代替エンティティとなる。[7]

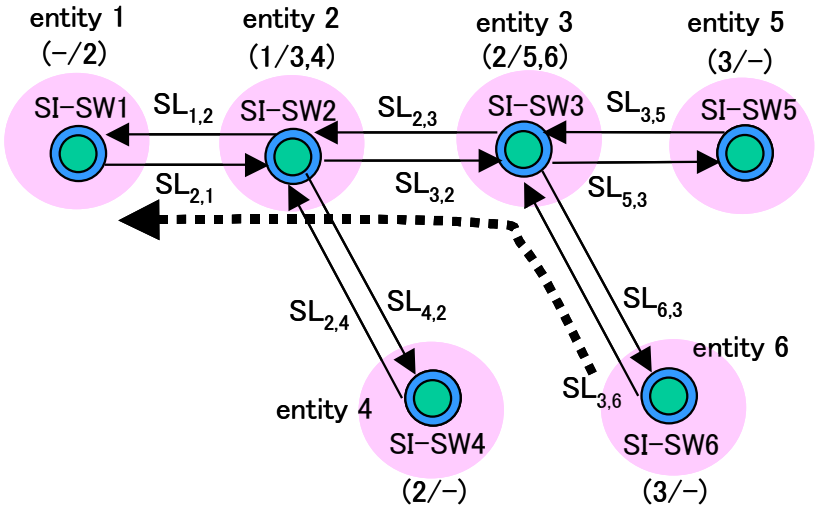


図18 ホップ属性

[7] 障害時の代替エンティティの探索ばかりでなく、シェアードリンクの確立を受け付け可能なエンティティの探索、トップ（誰にもシェアードリンクの確立を行っていない）エンティティの探索などに有効である。SIONet では、トップのエンティティのみが、他のイベントプレースに対して、フェデレーション（イベントプレース間のシェアードリンクに相当）を確立することができる。

3.7.4　イベントパスのキャッシング

表１は、図 22 においてサービスエンティティがフィルタを登録したときに、SI-R が設定するイベントパス（イベントルーティング情報）がどのように更新･管理されるかを示したものである。

表１において、SE $_i$+は、サービスエンティティ i が、フィルタ登録したことを意味している。一方、SE $_i$-は、サービスエンティティが当該フィルタを解除したことを意味している。また、表中の数字は、SI-R が保持しているイベントパス管理のためのカウンタ値であり、SI-R が SI-SW に対して、イベントパス設定のためのフィルタ登録を行っている回数を示している。このような管理は、イベントタイプ毎に行われる。但し、SI-R から SI-SW に対する実際のフィルタ登録は、カウンタ値が 0 から 1 へインクリメントされたときに行われるだけであり、たとえば、1 から 2 へ遷移する場合には、フィルタ登録は行わずに、

	Initial value	①SE3+	②SE5+	③SE2+	④SE3-	⑤SE5-	⑥SE2-
SI-$R_{5,3}$	0	0	1	1	1	0	0
SI-$R_{3,2}$	0	1	2	2	1	0	0
SI-$R_{2,4}$	0	1	1	2	2	1	0
SI-$R_{2,1}$	0	1	1	2	2	1	0

Table 1. Management of event path for each event type.

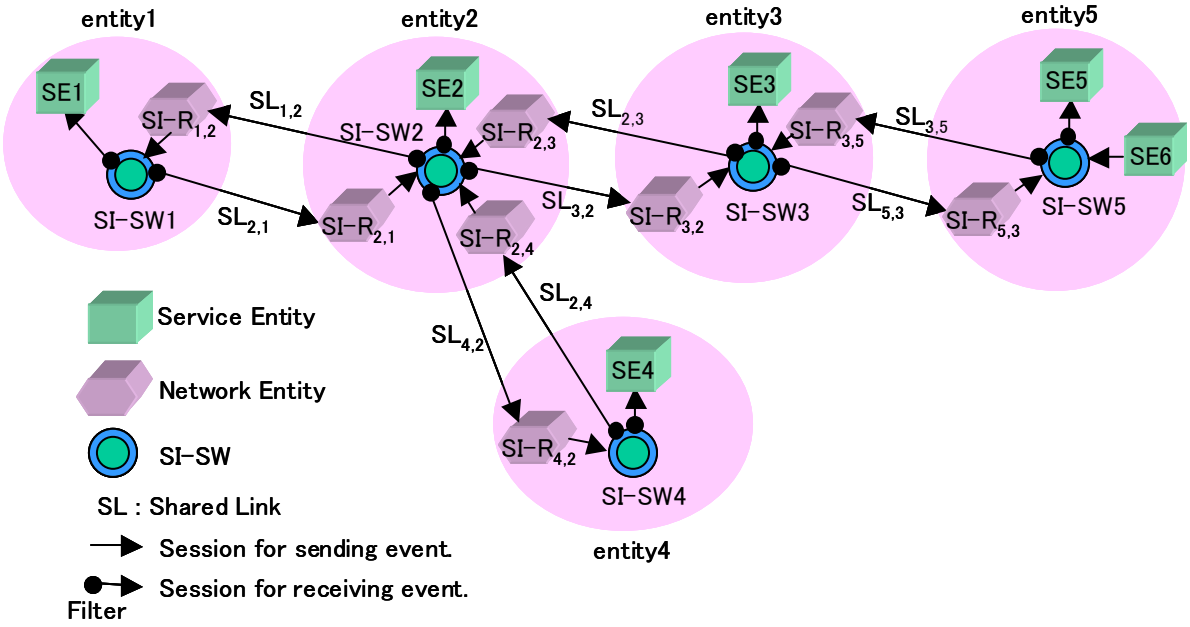


図22　Shared Link

フィルタ登録が要求された旨のみをカウンタで記憶している。同様に、サービスエンティティがフィルタ解除を行った場合には、イベントパスの更新のために、カウンタをデクリメントするが、SI-R から SI-SW に対する実際のフィルタ解除は、カウンタ値が 0 になったときに始めて行われる。これにより、イベントパスの設定要求や解除要求が、順次、自律的に波及（伝播）していく回数を必要最低限に抑止したキャッシングが可能になり、イベントパス制御のオーバヘッドを低減することができる。

なお、説明を簡略化するために、片方向のシェアードリンクのみを表中に示した、実際には、双方向のシェアードリンクにおいて同様のイベントパス管理が行われる。

以下に、簡単に説明する。

① 初期値において、当該イベントタイプに対するイベントパスは確立されていないことを示している。

② SE3 がフィルタを登録することにより、イベントパスの設定要求が順次、自律的に波及し、その結果、$SI\text{-}R_{3,2}$ が SI-SW2 にフィルタを登録し、同様に、$SI\text{-}R_{2,1}$ と $SI\text{-}R_{2,4}$ がそれぞれ SI-SW にフィルタ登録することによりイベントパスが設定される。このとき、該当の SI-R はそれぞれカウンタをインクリメントする。

③ さらに、SE5 がフィルタを登録すると、$SI\text{-}R_{5,3}$ を介して、イベントパスの設定要求が $SI\text{-}R_{3,2}$ に対して波及するが、$SI\text{-}R_{3,2}$ は、既に当該イベントタイプに関するイベントを転送するためのフィルタを登録済み（カウンタ値が１）であるため、カウンタを単に２にインクリメントするだけで、更なるフィルタ登録は行わない。そのため、$SI\text{-}R_{2,1}$ と $SI\text{-}R_{2,4}$ に対して、イベントパスの設定要求が波及しないため、両者のカウンタには変更が生じない。

④ 同様の考え方で、表１に示すように、表中のカウンタ値が更新される。

3.7.5 イベントプレース

イベントプレースは以下の目的から考案されたグルーピングの概念である。

①イベントの転送範囲：刺激と発火に基づく連鎖反応の波及範囲を制限する。

②イベントルーティングの対象範囲：シェアードリンクにより接続されているエンティティの集合である。イベントプレースごとに、リンクトポロジーやイベントルーティング方式を選択できる。

③意味情報体系の一意性が保証される空間：取り扱うイベントタイプの爆発的な増大を抑止することができる。

一般的に、サービスごとに必要とされる意味情報体系（語彙概念や語彙）が異なる。たとえば、証券サービス、医療サービスなどサービス種別ごとにイベントプレースを構築することができる。

3.8　オンライン増減設

3.8.1　オンライン増減設の目的

SIONet におけるエンティティ増減設の目的は、主に以下の二つに大別される。

- イベントプレースのトータル処理能力向上の観点から、イベントプレース内のエンティティを増設し、イベントのフィルタリング処理を負荷分散する。その逆の観点から、エンティティを減設する。これは、主に、ハイブリッド P2P、バックボーン P2P ネットワークの運用において用いられる。
- 動的に生成されたエンティティに対して、シェアードリンクを柔軟に確立することにより、フレキシブルでグローバルな P2P ネットワークをボトムアップ的に構築する。これは、主に、ピュア P2P ネットワークの運営において用いられる。

3.8.2　増減設の形態

SION では幾つかの増減設形態を提供しているが、ここでは、図 19 に示す代表的な形態について述べる。

(1) イベントプレースの合成と分離

図 19(a)に示すように、複数のイベントプレースを合成することができる。ここで合成とは、複数のイベントプレースにそれぞれ属するエンティティを、一つのイベントプレース内のメンバーとして集めることを言う。典型的な例として、異なるサービス運営者間の業務提携に基づく、サービス統合（情報共有）が考えられる。

これは、イベントプレースに属する任意のエンティティ、もしくはイベントプレースに対して合成要求を行うことにより、合成要求先イベントプレース内のエンティティに対してシェアードリンクの確立要求が発行される。その結果、SI-SW 間にシェアードリンクが確立され、両者の合成が実現される。なお、合成の要求元および要求先は、それぞれ、エンティティでもイベントプレースでもどちらであってもかまわない。一方、分割する場合には、確立されているシェアードリンクを解除し、それぞれのイベントプレースに分離する。

(2) イベントプレースへの参加と退去

図 19(b)に示すように、イベントプレース内のエンティティ、もしくは、イベントプレースに対して Join 要求を行うことにより、要求元エンティティに SI-SW が生成される。そして、要求先エンティティに対してシェアードリンクの確立要求が発行されることにより、SI-SW 間にシェアードリンクが確立され、当該イベントプレースに参加することができる。

エンティティがイベントプレースから退去する場合には、エンティティ間のシェアードリンクを解除し、さらに、シェアードリンクの再構築を行い、当該イベントプレースから退去する。このとき、退去したエンティティの状態は、サスペンド状態へと遷移する。

なお、イベントプレース側からエンティティに対してシェアードリンクの確立を要求す

ることにより、当該エンティティをイベントプレース内に取り込むことも可能である。これを吸収という。その逆を分裂という。吸収は、リコメンデーションサービスやプッシュ配信の実現に利用することができる。

(3) エンティティ（SI-SW）の増減設

図 19(c)に示すように、イベントプレース内のエンティティ、もしくは、イベントプレースに対して、エンティティの増設要求を行うと、指定されたエンティティに SI-SW が新たに生成され、既存の SI-SW との間にシェアードリンクが確立される。一方、イベントプレースにエンティティの減設要求を行うと、指定されたエンティティに対しシェアードリンクの解除要求が発行され、SI-SW 間のシェアードリンクが解除された後、シェアードリンクの再確立が行われ、当該エンティティが削除される。このとき、エンティティの状態は、Non-Existent に遷移する。これは、後述するサーバやハイブリッドモデルの実現に利用することができる。

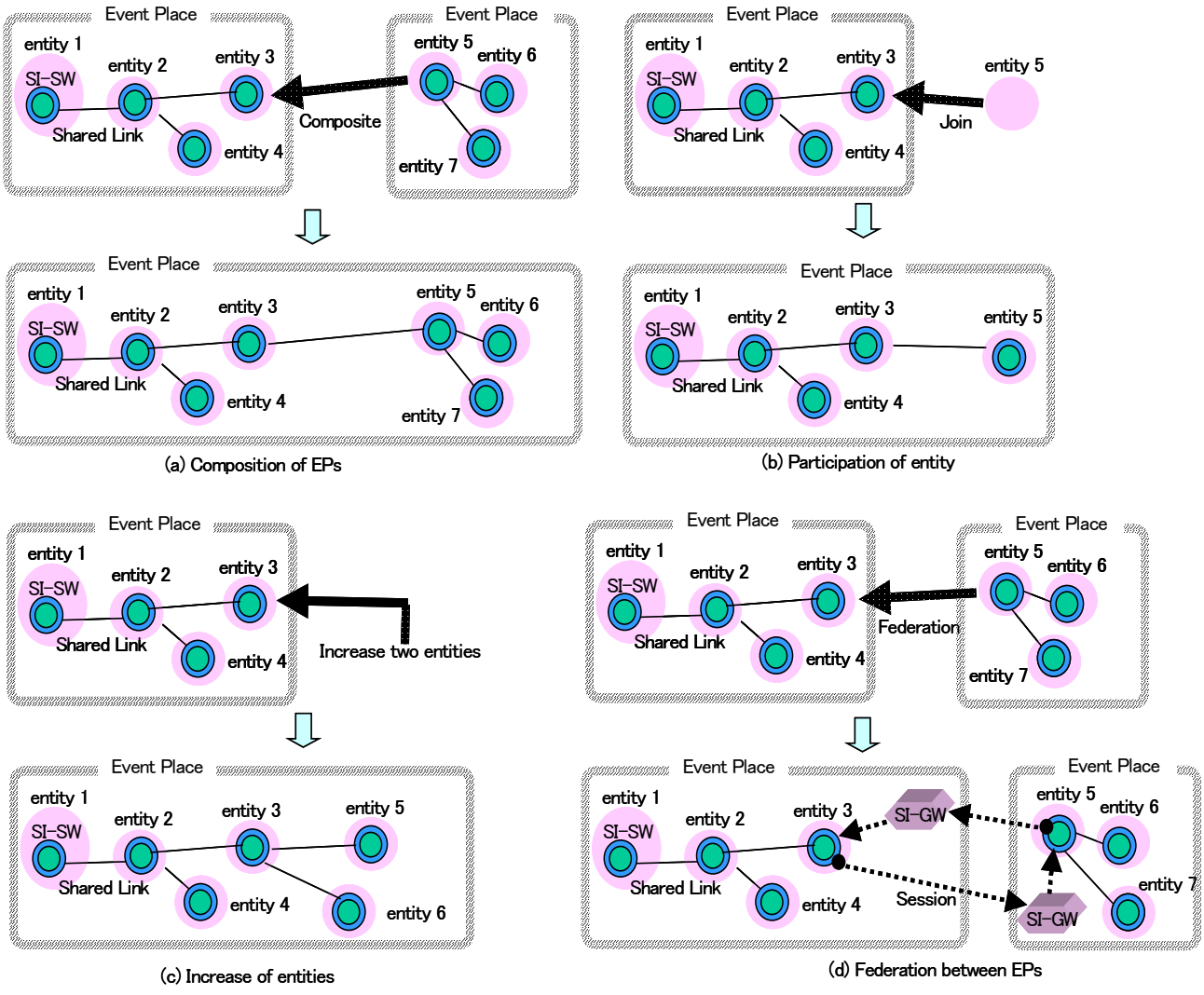


Fig.19 増減設の分類

(4) イベントプレース間のフェデレーション

図 19(d)に示すように、イベントプレース内のエンティティ、もしくは、イベントプレースに対して、フェデレーション（連携）要求を行うことにより、SI-GW が動的に生成され、セッションを介して両イベントプレースが連携する。なお、フェデレーションの要求元および要求先は、それぞれ、エンティティでもイベントプレースでもどちらであってもかまわない。

(5) コネクト

エンティティに対して、シェアードリンクではなく、セッションを確立する。これは、クライアントサーバモデルにおけるクライアントの実現に有効である。

3.9　エンティティのアドバタイズメント

図 20(a), (b)を用いて、エンティティのアドバタイズメント（公開）の目的と仕組みを説明する。SIONet におけるエンティティのアドバタイズメントには、以下の 2 つの観点がある。

- 観点 1：エントランスのアドバタイズメント

ベースイベントプレースは、エンティティに取って、最適なイベントプレースをいもづる式に探索（ディスカバリ）するためのベース（起点）となるイベントプレースである。つまり、ベースイベントプレースが SIONet へのエントランス（入り口）となる。そのため、ベースイベントプレースに参加しているエンティティは、SIONet のエントランスを公開することができる。ここで、エントランスの公開とは、シェアードリンクの確立要求先となるエンティティのエントリポイントとグローバルエンティティ名をアドバタイズすることを意味する。この公開情報は、後述するブロードキャストを用いた探索により発見される。

- 観点 2：エンティティプロパティのアドバタイズメント

シェアードリンクで接続されている任意のエンティティグループ（ベースイベントプレースを含むすべてのイベントプレース）において、各エンティティは自身のエンティティプロパティをアドバタイズすることができる。これが前述したイベントパスの波及に相当する。この公開情報は、後述するディスカバリイベントの送出により発見される。

以下に、エンティティの公開、探索から、エンティティグループ形成までの流れを説明する。

① 図 20(a)に示すように SIONet ソフトウエアをホストにインストールする。この時点でのエンティティの状態は、“Non-Existent”である。
② SIONet ソフトウエアを実行することにより、“Non-Existent”から“Suspend”状態

に遷移する。この状態のエンティティは、SIONet からその存在がまだ認知されていない。[8]

③ “Suspend”状態のエンティティは、SIONet の構成要素としてネットワークに参加するために、ベースイベントプレースに属しているエンティティ（エントリポイントとグローバルエンティティ名）を探索する。具体的には、ブロードキャストにより、ベースイベントプレースに参加しているエンティティの中から、近傍のエンティティを探索する。ブロードキャストの方法は、実装に依存する。例えば、無線ネットワークで実装されている場合には、無線の到達範囲すべてのエンティティが探索の対象と

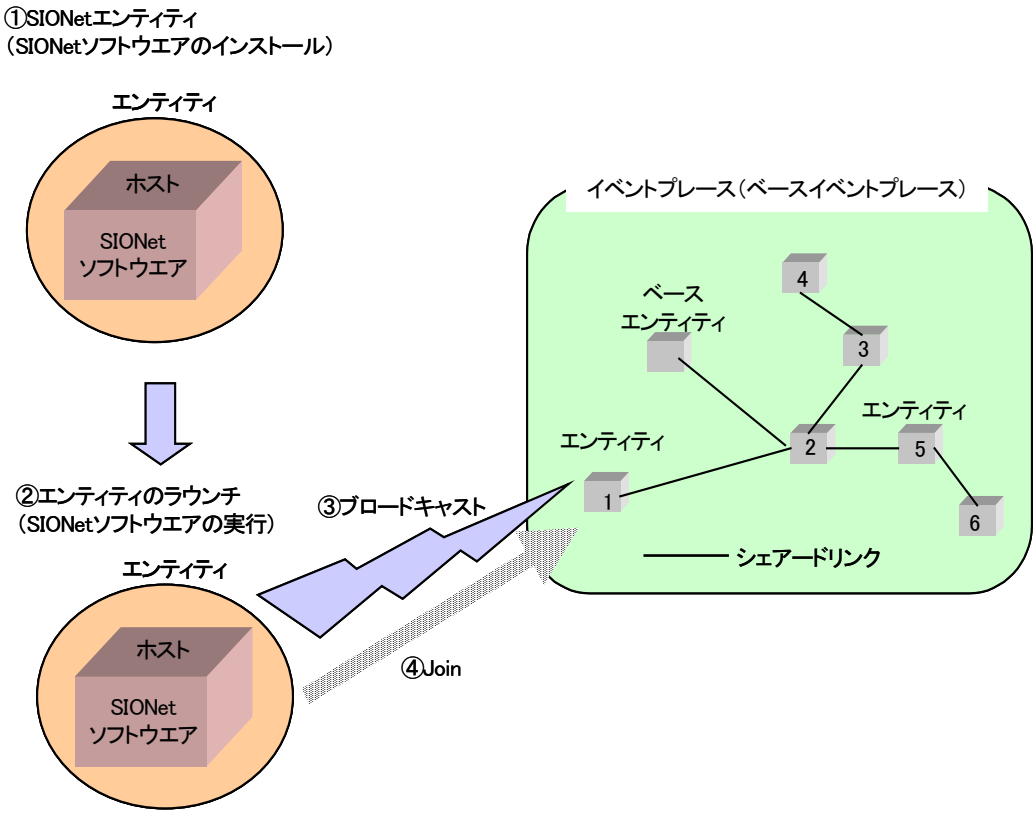


図20(a) アドバタイズメント方式

[8] “Suspend”状態のエンティティは、エントランスやエンティティプロパティを公開することができない。“Suspend”状態のエンティティは、任意のイベントプレースに Join することにより、“Active”状態に遷移する。“Active”状態のエンティティのみが、エントランスやエンティティプロパティを公開することが可能である。なお、ベースイベントプレースに Join しているエンティティのみがエントランスの公開が可能である。

「コントロールパネルにより公開が指示されている」かつ「当該エンティティに対するセッション（シェアードリンク）の確立数がデフォルト値を越えていない」を満足するとき、エンティティは、エントランスやエンティティプロパティを実際に公開する。例えば、エンティティの所有者がコントロールパネルを用いて公開を指示しても、デフォルト値を超えてシェアードリンクが確立されている場合には、自動的に非公開モードとなる。なお、デフォルト値は、コントロールパネルで設定可能であり、例えば、エンティティの処理能力に応じて決定することができる。

なる。一方、IP ネットワークで実装されている場合には、IP ブロードキャストもしくは IP マルチキャストを行うことになる。図 20(a)では、ブロードキャストによる探索により、エンティティ 1（エンティティ 1 のエントリポイントとグローバルエンティティ名）が発見されたことを表している。なお、ブロードキャストにより、近傍のエンティティを発見できない場合には、Well-Known のエンティティを利用することもできる。Well-Known のエンティティをベースエンティティと呼ぶ。なお、ベースエンティティは、ベースイベントプレースを含むすべてのイベントプレース内に存在することができる。

④ 発見されたエンティティ 1 に対して、ベースイベントプレースへの Join 要求を行う（厳密にはエンティティ 1 のエントリポイントに対して、グローバルエンティティ名をパラメータとした Join 要求を行う）ことにより、当該エンティティ（説明の便宜上、以降、エンティティ Y と呼ぶ）とエンティティ 1 との SI-SW 間にシェアードリンクが確立される。[9] これにより、エンティティ Y は、ベースイベントプレース内のネットワーク構成要素として自己組織化される（図 20(b)）。このとき、エンティティ Y は、“Suspend”状態から“Active”状態へと遷移する。

ベースイベントプレースに Join したエンティティ Y は、エントランスの公開が可能になる。また、エンティティプロパティをアドバタイズすることもできる。エンティティプロパティには、グローバルエンティティ名（およびエントリポイント）、ニックネーム、グループ名、アライブ（存在だけの表明であり、シェアードリンクの確立先情報となるエントリポイントとグローバルエンティティ名は公開しない、非公開モードでも公開の対象）、ディスクリプション（エンティティの説明文）、属性などがある。属性には、プラグインされている SE 情報、イベントパス多重度、登録済みのイベントタイプ情報、イベントプレース情報などがある。例えば、エンティティ 3 は、エンティティ 3 が同時に参加しているイベントプレース α をエンティティプロパティの属性として公開することができる。さらに、現在 Join しているイベントプレースばかりでなく、過去に Join したイベントプレース情報を公開することもできる。なお、エンティティプロパティの記述言語としては、XML などがある。

公開時には、これらのエンティティプロパティを、エンティティ内の意味情報スイッチにフィルタとして登録することにより、エンティティプロパティがシェアードリンクに基づいて他のエンティティに波及し、イベントパスが確立される。基本的には、イベントパスの確立要求がイベントプレース内のすべてのエンティティに対して波及するが、TTL 値で、イベントパス確立要求の波及範囲を制限することができる。なお、

[9] エンティティ Y からの Join 要求を受信したエンティティ 1 は、自身の状態遷移が“Active”であり、かつ、公開モードのときに、Join 要求（シェアードリンクの確立要求）を受け付ける。なお、エンティティ Y は、最大同時 Join 数を越えて、Join 要求を発行することはできない。最大同時 Join 数は、コントロールパネルで設定可能である。

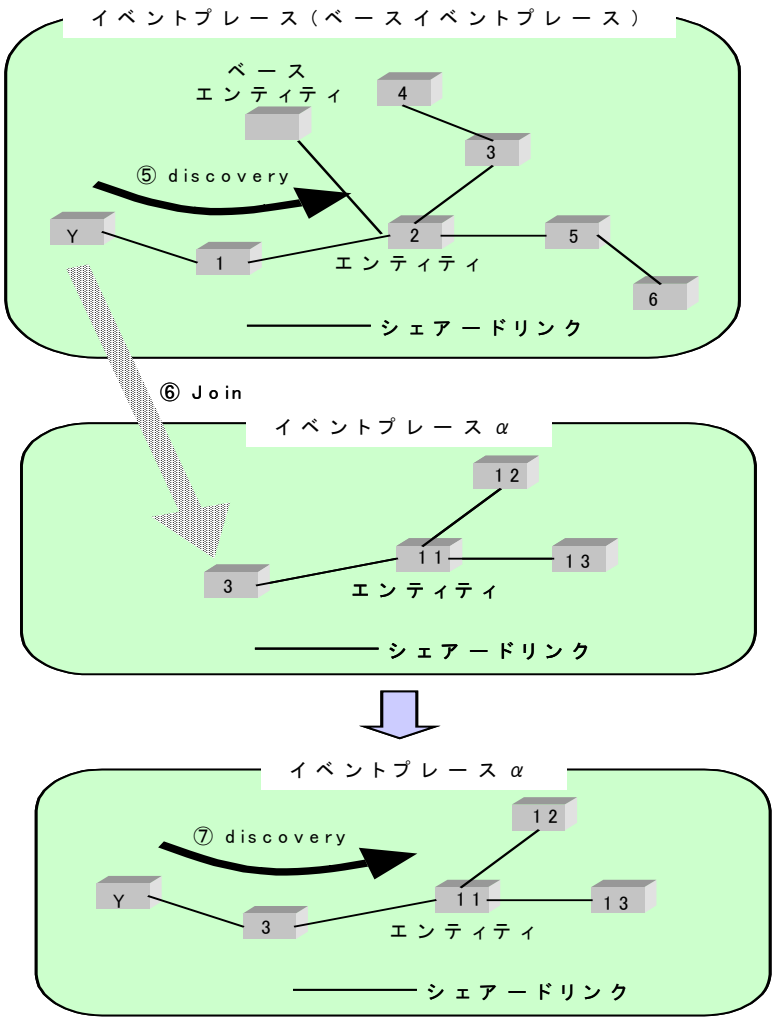


図 20(b) アドバタイズメント方式

グローバルエンティティ名はマルチホップ型ユニキャスト通信時の識別子として、グループ名はマルチホップ型マルチキャスト通信時の識別子として、属性はマルチホップ型属性付きマルチキャスト通信時の識別子として用いられる。例えば、マルチホップ型ユニキャスト通信の代表例として、イベント受信エンティティからイベント送信元エンティティへのリプライ通知がある。

⑤ ベースイベントプレースに参加したエンティティ Y は、「自身が参加したいイベントプレース」を探索するためのディスカバリイベント[10]を送出することにより、「自身のニーズに合致するエンティティプロパティ」を公開しているエンティティを発見する。

[10] ディスカバリイベントは、SIONet の制御のために用いられるイベントであり、SE が送出するイベントと基本的には何も違わない。すなわち、NE を発火させるためのイベントであり、イベントの意味情報部には、エンティティプロパティとの照合条件が設定される。すなわち、SIONet における新たな機能追加とは、新たな語彙概念と語彙を規定することを意味する（仕組みの追加ではなく、連鎖反応条件の追加）。これにより、単一な仕組み（連鎖反応）だけで、様々な機能追加を可能にしている。

ここでは、その結果、エンティティ 3 が発見されたものとする。なお、エンティティ 3 は、同時にイベントプレース α に参加しているものとする。これにより、イベントプレース α の存在を知ることができる。

⑥ エンティティ 3 に対してイベントプレース α への Join 要求を発行することにより、エンティティ Y はイベントプレース α に参加することができる。なお、エンティティ 3 は、前述したように、自身の状態遷移が“Active”であり、かつ、公開モードのときに、Join 要求（シェアードリンクの確立要求）を受け付ける。

⑦ さらに、エンティティ Y が、イベントプレース α において、同様の探索のためのイベントを送出することにより、自身のニーズに合致する新たなエンティティを発見することができる。例えば、ここでは、エンティティ 12 が発見されたものとする。これは、ベースイベントプレースにおいて発見することができなかったエンティティ 12 を、エンティティ 3 を介して発見できるということを意味している。このような操作を繰り返すことにより、自身のニーズに合致したイベントプレースに、次第にたどり着くことが可能になる。

ここで示したアドバタイズメント方式は以下の効果を与える。

(1) エンティティの公開情報を管理するブローカ（管理部）が存在しないため、耐障害性に強い自己組織化ネットワークを低コストで構築することができる。また、膨大な数の公開情報（エントリポイントやエンティティプロパティ）をブローカで管理することは現実的でない。

(2) エンティティに取って相応しいイベントプレースを、芋づる式に絞り込むことができるので、効率的に所望のイベントプレースを探索することができる。例えば、連鎖反応の波及範囲（ディスカバリイベントのホップ数、および、エンティティプロパティのためのイベントパス設定要求の波及範囲）が TTL 値で制限されたとしても、エンティティ 3 経由でエンティティ 12 を発見し、その結果、エンティティ 12 が同時に属している別のイベントプレース（例えば、イベントプレース β）を発見することができる。なお、連鎖反応の波及範囲を制限することにより、ネットワークトラヒック（イベントの転送回数、イベントパス設定要求の転送回数）を軽減することができる。

なお、自身は情報を提供することなく（例えば、エンティティプロパティを公開することなく）、他のエンティティからの情報提供を受けるだけのエンティティは、ペナルティーとして（貢献度に応じて）、強制的にシェアードリンクが解除され、一定条件を満足しないと再度、イベントプレースに参加できないことがある。この考え方は、様々なインセンティブ問題に応用することができる。

3.10　SE の共有方法（自動配信）

以下に、ゲームイベントプレースにおける SE（ゲームアプリケーションプログラム）の共有を例に、SE の共有方法の仕組みを示す（図 21）。

(1) ゲームイベントプレースの運営者であるエンティティ 2（厳密には、エンティティ 2 の所有者）は、図 5 において説明した手順で、イベントプレースを生成する。なお、イベントプレース生成時に当該イベントプレース名として「ゲーム」と付与したものとする。

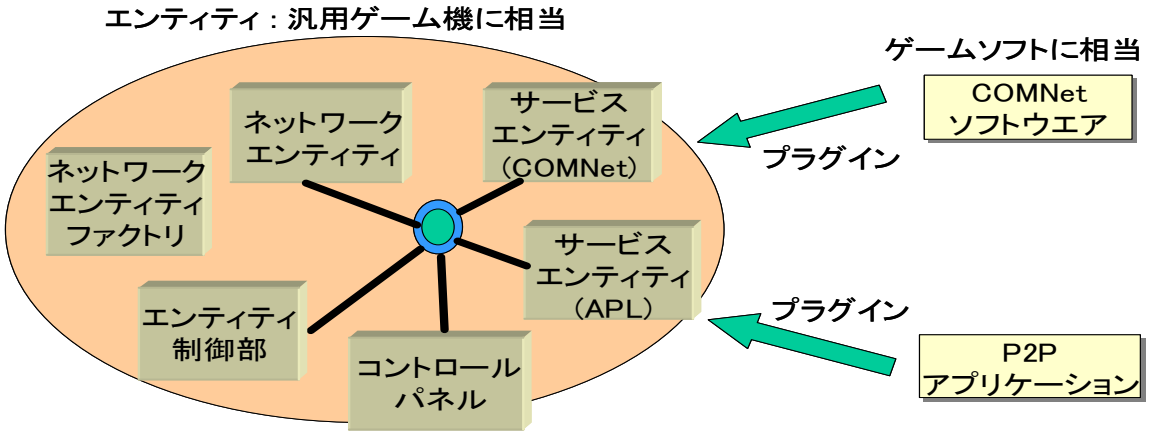
(2) ゲームイベントプレースの生成者であるエンティティ 2 は、生成時に当該イベントプレースに自動的に参加する。このとき、SI-SW が生成され、図 10 に示すように、当該 SI-SW に対してグローバルエンティティ名「ゲーム＋エンティティ 2」が付与される。

(3) エンティティ 2 は、当該エンティティのコントロールパネルに対して、ゲームアプリケーションプログラム（SE）のプラグインを指示する。このとき、プラグインするアプリケーションの実行ファイル名、および、SE 共有の有無をパラメータとして与える。

(4) コントロールパネルはこの旨をエンティティ制御部に通知する。

(5) エンティティ制御部は、プラグインする実行ファイル名を記憶するとともに、与えられた実行ファイル名を用いて、アプリケーションを起動し、図 11 に示すように、SI-SW とアプリケーション間にセッションを確立する。すなわち、SIONet におけるプラグインとは、アプリケーションを SE として仮想化し、SE と SI-SW との間にセッションを確立することを意味する。

(6) エンティティ 2 は、エンティティ 2 のエンティティプロパティをアドバタイズする。エンティティ 2 のエンティティプロパティの公開形態としては、以下のものが考えられる（エンティティのアドバタイズメントの章を参照）。



COMNet: SIONetのミドルウエア(UA, CA, SA, COMAなどのエージェント)

図21　プラグイン&SE共有

・エンティティ 2 がベースイベントプレースに参加し、ベースイベントプレース内で、エンティティ 2 のエンティティプロパティを公開する。
・エンティティ 2 が、芋づる式探索を行うことにより、ゲームイベントプレースと関係の深いイベントプレースを発見、参加し、当該イベントプレース内で、エンティティ 2 のエンティティプロパティを公開する。
・エンティティ 2 をベースエンティティとする。すなわち、エンティティ 2 のエンティティプロパティ（エントリポイントとグローバルエンティティ名）を何らかの方法で周知する。

(7) エンティティ 2 の存在を知ったエンティティ 1 が、エンティティ 2 に対して、ゲームイベントプレースへの参加を要求する。

(8) この要求が承認されたとき、エンティティ 1 は、エンティティ 2 に対して、シェアードリンクの確立を要求する。そして、エンティティ 1 とエンティティ 2 の間にシェアードリンクが確立される（図 14、図 15 参照）。

(9) シェアードリンクが確立されたとき、シェアードリンクの確立要求先（エンティティ 2）は確立要求元（エンティティ 1）に対して、様々な情報を返却する。具体的には、イベントプレース名（グローバルエンティティ名）、参加したイベントプレースがベースイベントプレースであるか否か、当該イベントプレースにおけるイベント転送方式（ルーティング方式）、イベントプレースのディスクリプション、プラグインされている SE 情報（SE 共有が有りの場合）、等の情報が返却される。

(10) SE 共有においては、これらの返却される情報の内、SE 情報を利用する。具体的には、(3)において、プラグインするアプリケーションの実行ファイル名をパラメータとして与えられたが、シェアードリンクの確立成功時に、当該ファイルをエンティティ 1 に対して配送する。なお、SE の実体がファイルであるか、SE 共有時にファイルを転送するかは実装依存であり、SE はこれらの実装を仮想化するための抽象的な概念であることに注意を要する。

(11) 当該ファイルを受信したエンティティ 1（エンティティ 1 のエンティティ制御部）は、(5)の処理を行うことにより、エンティティ 1 内部に、ゲームアプリケーションプログラムを SE としてプラグインする。

(12) すなわち、イベントプレースへの参加とは、シェアードリンクにより結合されたエンティティグループを形成し、プラグイン済みのアプリケーションプログラムを参加者が共有することを意味する。これにより、イベントプレースに参加するだけで、当該イベントプレース内のメンバーであるエンティティが、プラグインされているアプリケーションを、自動的に利用することが可能になる。なお、エンティティ 1 がゲームイベントプレースから退去するとき、プラグインされたゲームアプリケーションプログラムは、プラグアウトされる。なおその際、エンティティ 1 は、受信した当該ファイルを保有しつづけることもできるし、プラグアウト時に、当該ファイルが削除され

ることもある。

上述した仕組みは、ピュア型の P2P ネットワーク環境でのアプリケーション共有である。そのため、アプリケーションプログラムの配信は、シェアードリンク確立先エンティティと確立元エンティティの 2 者間でのみ行われるため、シェアードリンクにより既に結合されている他のエンティティに対して影響を与えない。すなわち、ファイル転送のため処理は 2 者間に局所化されるため、スケーラブルな SE 共有を達成することができる。

4. ビジネスモデル

(1) クライアントサーバモデル

予め決められたエンティティのみでイベントプレースを運営する形態である。すなわち、予め決められたエンティティのみが当該イベントプレースに参加(Join)することが可能であり、参加を許されたエンティティは、エンティティ間のシェアードリンクを確立することにより、エンティティグループを構成する。一方、それ以外のエンティティは当該エンティティグループに対してセッションを確立し、当該セッションを介したイベントの送受信を行う。すなわち、イベントプレース（エンティティグループ）がサーバであり、その以外のエンティティがクライアントに相当する。このビジネスモデルでは、エンティティグループのメンバーである各エンティティは、自律分散コンピュータとして振舞い、任意のエンティティが障害に陥った際には、残されたエンティティが自己組織化しイベントプレース（サーバ）の運営を継続する。また、サーバの処理能力向上を図りたい場合には、新たなエンティティを当該イベントプレースに参加させれば良い。そのため、従来のサーバ実現方式と比べて、耐故障性に強く、安価で、スケーラブルなサーバを実現することができる。

(2) ハイブリッドモデル

例えば、ゲームイベントプレースの運営者となるエンティティは、ゲームイベントプレースを生成し、当該イベントプレース内でのベースエンティティとなる。一方、ゲームイベントプレースのユーザとして位置付けられるエンティティは、当該イベントプレースに参加することにより、当該イベントプレースでのサービスを享受する。従来のクライアントサーバモデルでは、ユーザ数の増加に比例して新たな設備投資（CPU パワー、ストレージ、通信帯域など）が必要とされたが、本ビジネスモデルは、ユーザがイベントプレースに参加することにより､ユーザ自身のエンティティをサービス運営のために提供するため、イベントプレース運営者は新たな設備投資を必要としない。そのため、安価で、スケーラブルな P2P サ･ビス環境を構築できる。なお、ベースエンティティにおいて、当該イベントプレースに参加するユーザ（エンティティ）の認証、課金が可能である。また、ベースエンティティは、ディスカバリイベントを送出することにより、エンティティグループの中から、シェアードリンクの確立が可能なエンティティ（代替エンティティ）を発見し、

発見されたエンティティを参加希望のエンティティに対し通知し、両者の間でシェアードリンクを確立させることも可能である。そのため、営利目的のビジネスに直結しやすいが、特定の役割を担ったベースエンティティは、当該イベントプレースから退去することができない。

(3) ピュアモデル

すべてのエンティティが自由にイベントプレースに対して参加（退去）することが可能なビジネスモデルである。地域コミュニティなどのボランティアネットワークとの親和性が高い。このモデルでのエンティティのライフタイムは比較的短いことを前提とする必要がある。

上述したように、SIONet では、運営・配置問題に帰着することにより、様々なビジネスモデルを共通の仕組みで実現できる。そのため、開発工数、ソフトウエア規模、デバッグ効率、維持管理の容易性等に大きな効果を与える。

参考文献／SIONet の文献（1997-2002）

[1] 星合隆成、久保田稔: “インターネットの新潮流－非ブローカ型探索モデルと自律分散技術”, 電学誌, 121 巻, 3 号, pp.178-184(2000.12 受付、2001.3 掲載).

[2] 星合隆成: “招待講演: インターネットの新潮流「非ブローカモデル」とその実現技術「SION: 意味情報ネットワーク」”, SSE2000-235&IN2000-191, pp.65-72(2000.12 受付, 2001.3 掲載).

[3] 星合隆成, 柴田弘: “御用聞き型情報提案のための自律分散照合環境アーキテクチャとその性能評価”, 電子情報通信学会論文誌(D-I), Vol.J83-D-I, No.9, pp.1001-1012 (1999.10 受付、2000-09 掲載).

[4] 星合隆成、小柳恵一、ビルゲー・スクバタール、久保田稔、柴田弘、酒井隆道: “意味情報ネットワークアーキテクチャ”, 電子情報通信学会論文誌 B, Vol.J84-B, No.3, pp.411-424 (2000.7 受付, 2001-3 掲載).

[5] 星合隆成、柴田弘、酒井隆道、小柳恵一: “意味情報ネットワークアーキテクチャ: SION アーキテクチャ”, NTT R&D, Vol.50, No.3, pp.157-164(2000.12 受付、2001.3 掲載).

[6] 星合隆成、柴田弘:“御用聞き社会構築に向けてのコンテンツ情報流通網と自律分散照合環境アーキテクチャ”, 信学技報 SSE99-189, pp.55-62(1999.12 受付、2000-3 掲載).

[7] 星合隆成、山本和史、柴田弘: “PREFERENCE を用いたマルチメディアサービスの構築”, 信学技報 SSE98-113, pp.31-40(1998-10 掲載).

[8] 星合隆成、柴田弘:“最適コンテンツ抽出のためのエージェント技術に関する一考察”, 信学秋季全大, B-7-7 (1999-9 掲載).

[9] 星合隆成、山本和史、柴田弘: “パーソナライズ情報提案型コンテンツ配送コンポーネント: PREFERENCE”,電子情報通信学会論文誌(D-I), Vol.J83-D-I , No.2, pp.306-312 (1999.4 受付、2000-2 掲載).

[10] 未来ねっと研究所ホームページ
http://www.onlab.ntt.co.jp
http://www.onlab.ntt.co.jp/jp/ni/preference/index.html
[11] NTT ニュースレター　http://www.ntt.co.jp
[12] PREFERENCE ホームページ
http://ns0.t.onlab.ntt.co.jp/an/preference
[13] 星合隆成, “Jnutella ワークショップ招待講演”, 2001.5.22
http://www.jnutella.org/jnudev/jws-052201_2.shtml
[14] 星合隆成 : “意味情報ネットワーク SIONet の試み”, Software Design 誌, 2001 年 8 月号, pp.144-pp.146, 技術評論社.
[15] 日経インターネットテクノロジ,“NTT が分散ディレクトリを製品化”, 2001.6 月号.
[16] 日経コミュニケーションズ,“NTT 研究所の新 P2P 技術”, 2001.5.21.
[17] 日経エレクトロニクス, “P2P 接続環境で NTT が開発”, 2001.5.21.
[18] 日本経済新聞,“ネット端末間で直接交換、NTT が技術開発”, 2001 年 4 月 28 日発刊.
[19] 日刊工業新聞,“NTT 開発の SIONet 次世代ネット向けに脚光”, 2001 年 5 月 10 日発刊.
[20] 日経産業新聞,“サーバを介さず情報交換、NTT が新技術”, 2001 年 5 月 1 日発刊.
[21] 日本工業新聞,“NTT がピアツーピアで新技術”, 2001 年 5 月 1 日発刊.
[22] 科学新聞,“P2P の新技術 SIONet”, 2001 年 5 月 31 日発刊.
[23] 朝日新聞夕刊, 2001 年 7 月 27 日発刊.
[24] 星合隆成,:“意味情報ネットワーク:SIONet の紹介”, 電子情報通信学会 CQ 研究会, CQ2001-52, pp.45-pp.55(2001-09 掲載)
[25] 日経システムプロバイダ 2001.6.8
[26] 日経 Web カンパニー2001.7
[27] 日経バイト 2001.8
[28] 日経エレクトロニクス 2001.7-10
[29] 星合隆成:“意味情報ネットワーク SIONet におけるエンティティのオンライン増減設機構”, 信学論 B, Vol. J85-B, No.2, pp.180-199(2001.5.2 受付、2002.2 掲載).
[30] 星合隆成:“意味情報ネットワーク:SIONet の全貌”, 「Peer-to-Peer の動向と提案」講演会テキスト, 電子情報通信学会東京支部(2001-10 掲載).
[31] Takashige Hoshiai:“Semantic Information-Oriented Network”, IEICE（掲載予定）
[32] 星合隆成, パトリック ツィナニー, 佐藤規男:“要求応答型ストリームインタフェースの提案とその実装方式”,電子情報通信学会論文誌(D-I), Vol. J82-D-I, No.8, pp.1080-1092, (1999-8).
[33] Takashige Hoshiai, Patric Tsinany, Norio Sato:“Proposal of Stream Interface Based on Request-Response Model and Its Implementation”, Electronics and Communications in Japan, Vol.31, No.8, pp.10-21, (2000-07).

[34] 星合隆成“意味情報ネットワーク SIONet の全貌”,「P2P 技術の事例と今後の展開」情報処理学会関西支部主催講演会テキスト, 情報処理学会関西支部主催講演会（2001.10.17 掲載）.
[35] 星合隆成:“P2P の理念およびその実現技術 : SIONet の全貌”, IECP 研究会講演テキスト, 国際大学 IECP 研究会主催講演会 2001.11.29.
http://www.glocom.ac.jp/top/topic/
[36] P2P ビジネスモデル戦略,バガボンド, 2001-12
[37] 日経ネットナビ, No. 68, pp.22, 2002-1.
[38] 星合隆成:“P2P ネットワーキング技術とブロードバンド”, NTT-AT 技術セミナーテキスト, 2001.12.14
[39] 「NTT R&D フォーラム 2001 in 厚木」 レポート, IMPRESS, 2001-10.
http://www.watch.impress.co.jp/
http://www.watch.impress.co.jp/internet/www/article/2001/1015/nttrd.htm
[40] 星合隆成:“P2P ネットワーキング技術とその応用および今後の動向”, 日本テクノセンターセミナーテキスト, 2002-01.
[41] 星合隆成:“P2P の理念とその動向”, 情報処理開発協会ハイエンドコンピューティング技術調査ワーキンググループ主催セミナーテキスト(HECC), 2002-01.
[42] 星合隆成“特別講演:ブローカレス型探索モデル(P2P モデル)と意味情報ネットワーク SIONet”, 信学技報, CQ2001-91, MVE2001-124, pp.55-pp.62, 2002.2.7
[43] 小柳恵一、星合隆成、梅田英和:“招待論文 : P2P ネットワーキング技術の提案と紹介”, 信学論 B, pp.319-pp.332(2001.12 受付、2002-03 掲載)
[44] 星合隆成:“P2P の理念､SIONet の全貌およびデモンストレーション”, 第２回 Jnutella Workshop 講演テキスト(2002.4.11).
[45] 第 1 回 Jnutella Developers Workshop 招待講演, 2001.5
[46] 電子情報通信学会 SSE&IN 研究会招待講演, 2001.3
[47] 電子情報通信学会東京支部主催セミナー講演, 2001.10
[48] 情報処理学会関西支部セミナー講演, 2001.10
[49] 電子情報通信学会北海道支部主催セミナー講演
[50] 日本テクノセンター主催セミナー講演, 2002.1
[51] NTT-AT 社主宰技術セミナー講演, 2001.12
[52] 電子情報通信学会 CQ 研究会特別講演, 2002.2
[53] GLOCOM-IECP 講演会招待講演, 2001.11
[54] P2P 研究会招待講演, 2001.5

3.4　応用システム&応用分野

3.4.1　SC2001 に見る Grid の最新動向

田中　良夫　講師

1. はじめに

Grid とは、「高速ネットワークで接続された高性能計算機、大規模データベース、特殊な装置、人的資源などの様々な資源を柔軟に、容易に、安全に、統合的に、そして効果的に利用するためのネットワーク利用技術」である。Grid により、単体のスーパーコンピュータやデータ記憶装置の能力を超える大規模計算や大規模データ処理および高圧電子顕微鏡のような特殊な装置の遠隔利用といった様々な応用技術が開発され、大規模科学技術計算などのグランドチャレンジからビジネス応用まで多様な応用が可能である。Grid の技術的な本質は Virtual Organization（仮想的な組織による運営）にあり、複数の異なる組織にまたがる様々な資源の利用技術に関する研究が主たる課題となる。Grid への研究開発投資は米国で年間 6 億 3 千万ドル、欧州で 2 億ユーロを越え、国際的に研究開発が推進されている。日本においても総合科学技術会議において重要課題として Grid がとりあげられており、ネットワーク技術の飛躍的進歩に伴う新しい情報技術基盤として非常に重要な技術であると世界的に認知されている。

本稿では、2001 年 11 月に米国コロラド州デンバーで開催された高性能計算および高性能ネットワークに関する国際会議である Supercomputing Conference(SC 2001)における講演、技術論文、企業展示および研究展示の内容をふまえ、Grid の現状および最新動向について報告する。本報告においては、(1)Grid におけるソフトウェアの基盤として事実上の標準になっている Globus Toolkit、(2)ユーザに対して簡便なインタフェースを提供するポータルシステム、(3)高性能インターネット会議システムである Access Grid およびそれを利用して SC 2001 期間中に開催されたイベント SC Global、の 3 件に焦点をあてて Grid 技術および研究動向に関する報告を行う。以下、2 節では SC2001 における Grid 関連の講演および研究発表等、Grid 関連の活動を総括する。3 節では Globus Toolkit の概要および今後の動向を、4 節では Grid Portal システムについて、5 節では Access Grid および SC Global イベントに関して説明し、最後にまとめを述べる。

2. SC2001 における Grid 関連の発表

SC2001 は米国コロラド州デンバーにあるコロラドコンベンションセンターで 2001 年 11 月 12 日から 16 日まで開催された。SC2001 では基調講演、招待講演、原著講演、チュートリアル、Birds of Feather(BOF)、企業展示、研究展示などの様々な発表が行われるが、その中でもキーワードは Grid とバイオであった。以下に、Grid に関する発表についてま

とめる。

- 招待講演

  4 件の招待講演のうち、以下の 2 件が Grid に関係する講演であった。

  - ✓ The World Wide Telescope: Mining the Sky

    Jim Gray, Microsoft Research

    天文におけるデジタル化された観測データは膨大な量となり、公開されてはいるが生のデータを ftp などで取り出すような形態であり、利用者にとって使いやすいものとはいえない。そこで、天文の学会における Data Grid（大規模データを Grid 上で扱うインフラ）を構築し、World Wide Telescope という仮想的な天文台を作ろうとしているという内容の講演であった。高エネルギー物理学における Data Grid と異なるのは、そこにデータマイニングの技術が大きく関与することである。

  - ✓ Grid Computing in the Terascale Age

    Fran Berman, SDSC and NPACI

    San Diego Supercomputer Center の所長である Dr. Fran Berman が米国、特に NPACI における Grid 研究へのアプローチについて、ちょうどプロジェクトが開始したところである Tera Grid およびこれから始まる PRAGMA に関する講演を行った。Grid の定義および今までの歴史を振り返り、Grid の要素技術は成熟してきていること、そして、今後は PRAGMA をはじめとする高レベルミドルウェアやアプリケーションに主題を置いた研究開発が活発になることを述べた。PRAGMA のコンテクストでいえば、ApGrid のようなアジア地域の Grid テストベッドとの交流が重要であることに言及していた。Tera Grid は NSF が 53M$の予算をつけた大規模プロジェクトであり、SDSC, NCSA, CalTech, ANL の 4 つの研究所・大学を 40Gbps の高速ネットワークで接続し、理論ピーク性能 13.6TFlops、主記憶合計 6.8TB、内部ディスク 79TB、ネットワークディスク 576TB の Grid インフラである。主に脳の研究や実時間視覚化などの応用分野における画期的な研究成果創出をはかっている。

- 原著論文

  原著論文は 240 件の投稿があり、60 件が採録された。投稿論文および採録論文のいずれも、約 20%は Grid 関連の論文であった。Grid 関連のセッションとしては、以下の 4 つのセッションが開かれた。

  - ✓ Computational Grid Portals and Networks
  - ✓ Computational Grid I/O
  - ✓ Computational Grid Applications

- ✓ Computational Grid Environments and Security

- チュートリアル

  ここ数年 Supercomputing Conference では Globus Toolkit のチュートリアルが行われていたが、今年は Globus Toolkit のチュートリアルはなく、Grid の概論的なチュートリアル(The Emerging Grid: Introduction, Tools, Applications)と Data Grid に関するチュートリアル(Data Grids: Drivers, Technologies, Opportunities)の 2 件のみであった。

- BOF

  Grid に関する BOF は以下の 4 件であった。
  - ✓ Global Grid Forum Advanced Programming Research Group BOF
  - ✓ Terascale Computing Infrastructure and You
  - ✓ Distributed Information Systems Lab (DISL) and ASCI
  - ✓ Grid Web Service

- 企業展示および研究展示

  SC2001 の企業展示においては、Grid あるいは Globus のキーワードが非常によく目についた。IBM は CoG（Globus Toolkit の Java API を提供するパッケージ）を用いた Globus Toolkit の GUI デモを行っていた。デモそのものは Globus Toolkit の Java API を提供する CoG を利用しているだけで技術的な新規性はないが、Globus Toolkit の各機能に対して非常によくできた GUI を構築していた。Sun Microsystems 社は、Sun がフリーソフトとして配布している Sun Grid Engine(SGE)と Grid を絡めた展示を行っていた。具体的には、Globus Toolkit と SGE を組み合わせた（ジョブの送信には Globus Toolkit を利用し、ローカルなジョブマネージャとして SGE を使う）デモと、AVAKI と SGE の相互利用に関するデモを行っていた。Globus Toolkit と SGE のデモは本稿の報告者である産総研の田中が、デンバーの会場から Globus Toolkit を使って産総研のマシンにジョブを飛ばし、産総研のマシン側では SGE を使ってジョブのスケジューリングを行うというデモを行った。図 1 は Sun Microsystems 社の展示ブースの様子である。

  Compaq Computer はオーストラリアの Grid 基盤に Compaq のクラスタが利用されていることを、オーストラリアとネットワーク会議を行いながら説明していた。Platform Computing は Globus Toolkit をオフィシャルサポートすると発表し、展示においても Globus Toolkit に関する展示を行っていた。

  また、Grid における企業の取り組みとして目を引いたのは、企業連合を組むところが多かったことにある。たとえば、Platform Computing、AVAKI、Compaq の三社が

図１　Sun Microsystems 社の展示ブース

組んで、それぞれ Grid ミドルウェアのサポート(Platform Computing)、Grid におけるファイルシステムの提供(AVAKI)、ハードウェアサポート(Compaq Computing)といったように役割分担をして Grid 市場に出ようとしているほか、Sun Microsystems 社は AVAKI と共同で展示を行っていたし、Sun Microsystems 社と IBM との研究者が今後の Grid ミドルウェアに関する標準化について話をしようという会話をしていた。

研究展示に関しては、ここ数年どおり Grid に関して非常に数多く活発に展示が行われていた。目を引いたのは、米国で 53M$の予算がついた TeraGrid の展示（NPACI/SDSC）とアジア太平洋地域における Grid プロジェクトである Asia Pacific Grid Partnership for Grid Computing(ApGrid)の展示である。ApGrid のブースでは、日本の産総研、東工大、京大からの展示に加え、オーストラリアの Queensland University of Technology と Monash University、タイの Kasetsart University、および韓国の Korea Institute of Science and Technology Information のデモ展示および、ポスター展示も行われていた。図 2 に ApGrid の展示ブースの様子である。

- SC Global

SC Global は米国の Argonne National Laboratory で開発された Access Grid と呼ばれる技術（一種のネットワーク会議システム）を使って世界中の各国、各都市とデンバーの会場とを接続し、デンバーでの会議の様子を世界中に放送したり、世界各地

図 2　ApGrid の展示ブース

の講演の様子をデンバーの会場で放送したり、あるいは世界各地とデンバーとをつないでパネル討論を行うなど、さまざまなイベントが会議期間中に開催された。SC Global に関しては、後ほど詳しく述べる。

全般的な印象としては、Globus あるいは Globus Toolkit が絡む発表、展示が非常に多かったということがある。詳細は後ほど述べるが、Globus Toolkit は Grid におけるソフトウェアを構築する際の基盤ツールとして事実上の標準になっており、今後各地で実際の Grid テストベッドの運営が進められると予想されるが、相互利用の可能性を高める意味でも Globus Toolkit が共通のソフトウェア基盤として利用されることはほぼ間違いない。そのような状況において、ユーザに対してより上位なインタフェースを提供する高レベルミドルウェアや Web インタフェースを提供する Portal システムの研究開発が活発化すると予想される。また、Grid の本質である仮想的な組織(Virtual Organization)の運営においては実際に顔をあわせてのミーティングが重要になると思われ、Access Grid のようなシステムは有用であり、今後 Grid の技術を取り込んでより高機能、高性能なシステムの開発が望まれる。そこで、本稿では引き続き Globus Toolkit の紹介、Web インタフェースを持つ Grid ポータルシステムおよび Accss Grid と SC Global について詳しく述べる。

3. Globus Toolkit

3.1 概要

Globusは1995年に開始された米国の大規模なGridプロジェクトであり、Argonne National LaboratoryとUniversity of South California/Information Science Instituteが中心となって活動している。Globus ToolkitはGlobusプロジェクトによって作られたツールキットであり、ユーザ認証、通信、遠隔資源管理・監視機構などの、Gridにおいて必要となるさまざまな要素技術をライブラリおよびAPIという形で提供する巨大なパッケージである。Globus Toolkitを用いてアプリケーションやミドルウェアを構築することができ、I-WAYやGUSTOなどのテストベッドによる実験や、Globus Toolkitを使ってGrid上で動作可能なMPICH-G/G2(Grid enabled MPI)の実装などが行われている。Globus Toolkitは1998年10月にバージョン1.0がリリースされ、2001年11月の時点での最新バージョンは1.1.4である。バージョン2.0のベータ版も2001年11月12日にリリースされ、近いうちにバージョン2.0の正式版がリリースされる予定である。Globus Toolkitは開発されて間もないソフトウェアであるが、またたく間に世界中に広まり、今やGridシステムのソフトウェアインフラストラクチャを構成する要素の事実上の標準になっている。

3.2 Globus Toolkit のサービス

Globus Toolkitが提供するサービスを以下に示す。

- セキュリティ（Grid Security Infrastructure, GSI）
  ユーザ認証などのセキュリティサービス
- 情報サービス（Grid Information Service, GIS）
  Metacomputing Directory Service (MDS)/Grid Resource Information Service (GRIS)/Grid Index Information Service (GIIS)などの情報サービス
- 資源管理（Resource Management）
  Globus Resource Allocation Manager(GRAM)/Resource Specification Language(RSL)/Dynamically-Updated Request Online Co-allocator(DUROC) などの資源管理および資源要求等のサービス
- データ管理（Data Management）
  Global Access to Storage Systems(GASS)/GSIFTPなどのリモートデータへのアクセスサービス
- 通信（Communication）
  NexusやGlobus IOなどの通信サービス
- 故障検知（Fault Detection）
  Heat Beat Monitorなどのシステムの状態および障害検知サービス

- 移植性（Portability）
  libc，pthread libraryなど高い移植性を提供

Globus Toolkitが提供するこれらのサービスは必要に応じて個別に利用することができるようになっており、既存アプリケーションへのインクリメンタルな導入が可能である。Globus Toolkitは階層的な構造を持ち、高レベルのGlobusサービスはローカルサービスの上に構築され、ローカルサービスの種類に依存しない均一なインタフェースを提供している。ここでは，特に重要である3つのサービス(セキュリティ、資源管理、情報サービス)およびGlobus Toolkitにおける遠隔ジョブ実行の仕組みについて詳しく説明する。

- セキュリティ

  ユーザ認証や安全な(secureな)通信を実現するため、Globus ToolkitはGrid Security Infrastructure (GSI)を提供している。GSIは公開鍵による暗号化、X.509証明書およびSecure Socket Layer (SSL)通信プロトコルに基づいて実装され、これにsingle sign-on(パスワードを1度だけ入力すれば良い)および委任(delegation)の拡張がなされている。ユーザは自分の証明書を認証局に発行してもらう必要がある。サーバ上にはユーザのGlobus ID(証明書に記載されている)とローカルユーザID(Unixのアカウントなど)との対応表があり、この表に従ったローカルユーザの権限でジョブが実行される。

- 資源管理

  Globus Resouce Allocation Manager (GRAM)は計算資源(プロセッサ)管理のためのサービスを提供する。GRAMはLSF、NQS、Condorなど様々な(サイト固有の)資源管理ツールに対する共通のインタフェースを提供する。GRAMのアーキテクチャにおいてはgatekeeperとjobmanagerが主要なコンポーネントである。gatekeeperはクライアントからのジョブ要求を待ち受けるサーバであり、jobmanagerはgatekeeperによって生成され、実際の資源管理方法に従ってジョブを生成するプロセスである。jobmanagerはローカルな資源管理ツールに応じた型を持ち、例えばLSF型のjobmanagerの場合はLSFの*bsub*コマンドを使ってジョブをキューに投入し、LSFのスケジューリングに応じてホスト上でジョブが実行される。また、これらの資源要求に関する情報はRSL (Resource Specification Language)と呼ばれる言語を用いて記述され、各コンポーネント間で交換される。

- 情報サービス

  Globus Toolkitは Metacomputing Directory Service (MDS)と呼ばれるLDAP(Light-weight Directory Access Protocol)を用いた情報サービスシステムを提

供している．Globus ToolkitのMDSは非集中管理方式(pullモデル)であり（バージョン1.1.2までは集中管理方式、pushモデルであった）、情報の問い合わせや情報を格納する名前空間の構築のためにLDAPを用いている。MDSの主要なコンポーネントはGrid Resource Information Service (GRIS)とGrid Index Information Service (GIIS)である。GRISは計算サーバ上で動作するLDAPのサーバであり、GRISに問い合わせることによってその計算資源に関する情報を獲得することができる。GRISが提供する情報はデフォルトではGlobus Toolkitのバージョン、計算機のハードウェア/ソフトウェア情報およびgatekeeper/jobmanagerに関する情報であるが、必要に応じてGRISが提供する情報を追加/削除する事ができる。GRISが単体の計算サーバの情報を扱うのに対し、GIISは複数のGRISの情報をまとめて扱う仕組みを提供し、それによって「複数のサイトにまたがった仮想的な組織」の情報を提供するMDSサーバとして機能する。

3.3　Globus Toolkit におけるジョブ実行の仕組み

Globus Toolkitを用いて遠隔資源上でジョブを実行する場合、以下のような流れになる。

① クライアントからユーザコマンドあるいはAPIを利用してgatekeeperにジョブ実行の要求を出す。実行すべきプログラムはRSLに従った記法でクライアントが指定する。
② gatekeeperがクライアントからの要求を受け取ると、クライアントとgatekeeperとの間で相互認証が行なわれる。
③ gatekeeperは適切なjobmanagerを生成する。
④ 生成されたjobmanagerはクライアントによって指定されたプログラムを実行するジョブプロセスを生成する。
⑤ ジョブ実行の成功/失敗/終了/結果の通知やジョブの取り消し要求などのやりとりは、クライアントとjobmanagerの間で行なわれる。ジョブプロセスの出力等はGASSを介してクライアント側の出力に送られる。

このように、Globus Toolkitはクライアント/サーバモデルに基づく単純な実行モデルを提供すると同時に、ジョブプロセスの生成や状態確認、実行の取り消しといった資源管理システム固有の機能をjobmanagerに任せることにより、各サイトのスケジューリングポリシーに依存しない柔軟な資源管理機構を実現している。

3.4　Globus Toolkit を使った例

ここでは実際にGlobus Toolkitを使ってGridソフトウェアを構築する例を示し、Globus Toolkitの使い方および有効性を示す。Gridソフトウェアの1つに、遠隔手続き呼び出しに

よるプログラミングを行うためのミドルウェアであるNinfシステムがある。Ninfシステムはクライアント・サーバモデルに基づき設計、実装されているシステムであり、最初のバージョン(Ninf V.1)は1996年にリリースされている。Ninf V.1においては、リモートライブラリの引数や返り値の情報（スタブ情報）やリモートライブラリの実際の所在等はクライアントとサーバとの間でNinfプロトコルを用いてやりとりされていた。また、サーバ側はクライアント側のIPアドレスをチェックするなどの簡単なセキュリティ機能はあったが、個別のユーザごとのセキュリティなどはない。Globus Toolkitを用いてNinfシステムを再実装することにより、GSIを利用したユーザ認証機構をNinfシステムに組み込むことが可能となる。実装に際しては、NinfサーバをGlobusのgatekeeperに置き換え、スタブ情報とリモートライブラリの所在はGRISに問い合わせるという方式を採用した。図3と4は、

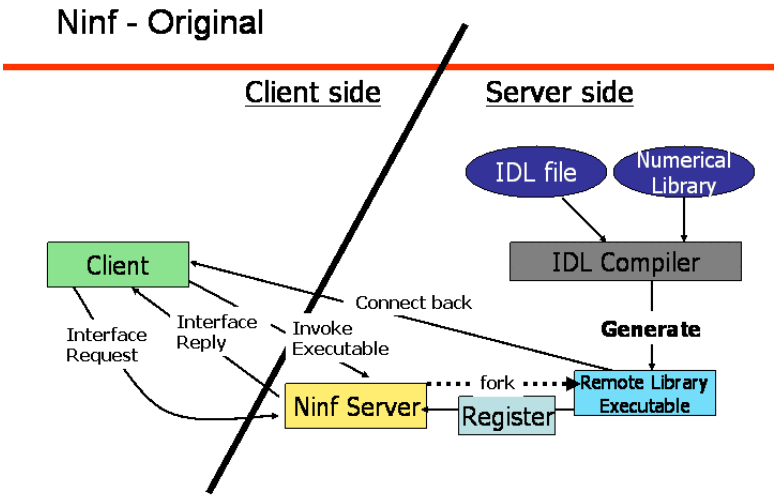


図 3　Ninf V.1 プロトコル

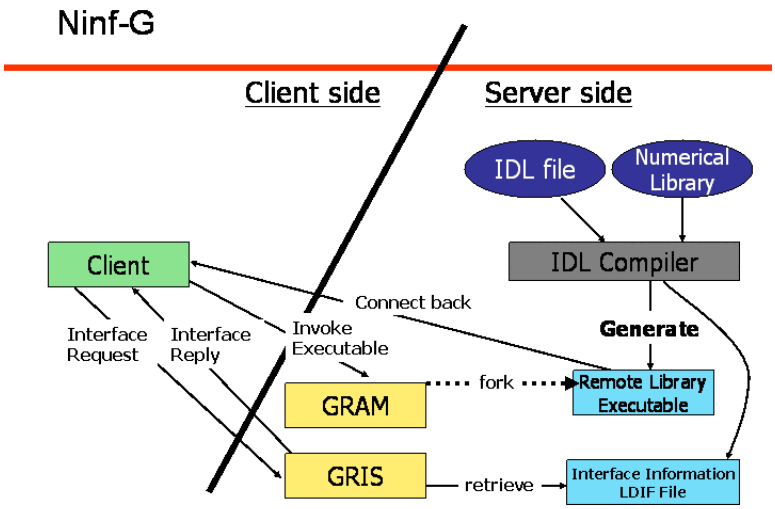


図 4　Globus 版 Ninf プロトコル

それぞれNinf V.1およびGlobus版Ninfにおけるクライアントとサーバ間のプロトコルを表したものである。

このようにGlobus Toolkitを用いて実装することにより、比較的容易にGSIに基づくセキュリティ機構が組み込め、他の多くのGridソフトウェアと共通したセキュリティ機構を提供でき、相互利用の可能性が高まる。また、GRISにライブラリの情報を登録することにより、他の情報サービスシステムを通してリモートライブラリの情報を提供することができ、利用性が高まる。

3.5　Globus Toolkit の最新の動向

3.5 1　Globus バージョン 2.0

Globus Toolkitの正式版はバージョンが1.1.4であるが、ベータ版としてバージョン2.0が配布されている。バージョン2.0は以下のような特徴を持つ。

- パッケージ化
  利用目的に応じて簡単にインストールできるように、いくつかのコンポーネントに分割パッケージ化されている。また、バイナリ版も配布されている。
- MDS 2.1
  セキュリティの機能を強化した情報の検索が可能となっている。
- GRAM 1.5
  （特に）耐故障性機能が強化されている。
- Data Grid向け機能の強化
  GSIftpやReplica Catalogue/Replica Managementなど、Data Grid向けの機能が強化されている。

3.5 2 企業とのからみ

SC2001開催中の11月12日に12の企業がGlobus ToolkitをGridにおける標準的なプラットフォームとして採用するという記者発表があった。Compaq、Cray、SGI、Sun Microsystems、Veridian、Fujitsu、Hitachi、NECの各社はGlobus Toolkitをそれぞれのプラットフォーム上へ移植および性能改善の作業を行っている。Entropia、IBM、Microsoftは以前よりのGlobus Toolkitに対する認知を深め、Platform ComputingはGlobus Toolkitの商用サポートを決定した。Globus Toolkit自体はフリーソフトウェアであるが、各企業はGlobus Toolkitを用いたビジネスの展開を狙っていると思われる。

3.6 まとめ

Globus ToolkitはGlobus Projectが提供しているツール群であり、いまや事実上の標準技術となっている。企業もGlobusを用いたビジネスの展開を模索しており、自社のプラットフォームへの移植や商用サポートなどを進めている。しかし、各企業の展示ブースで質問をした限りでは、まだ具体的なビジネスモデルは見えていないとのこと。しかし、今後の展開を予測してGlobusを中心としたGridの世界に参入しているとのことである。Globus Toolkitは非常によくできたソフトウェアであるが、Globus Toolkitが提供するのは低レベルなライブラリやAPIであり、エンドユーザが利用するには少々しきいが高い。そこで、今後はGlobus Toolkitの上に構築され、ユーザに対してより使いやすいインタフェースを提供するミドルウェアやポータルの分野の研究開発が加速すると予想される。

4. ポータル

ポータルとは「入り口」を意味する言葉であり、大きく分けて Web Portal と Computing Portal に分類される。

- Web Portal
  多様な Web コンテンツに対してディレクトリサービスや検索機能を提供し、統一性・透過性・可搬性のあるインタフェースを提供する。たとえば google や yahoo など。
- Computing Portal
  Grid 上に散在する資源およびサービスに対して、一元的、透過的かつ可搬性のあるインタフェースを提供する。

図 5 に米国 NPACI の HotPage という Computing Portal ページのイメージを示す。

HotPage は仮想的なスーパーコンピュータセンターをユーザに対して提供し、ユーザは HotPage の WEB ページからログインすることにより、SDSC、TACC、Caltech などのコンピュータセンターにあるスーパーコンピュータへのジョブを投入、ジョブの状態チェックや各コンピュータの負荷情報の取得などを行うことができる。また、ファイルシステムの操作も行うことができる。地理的にも離れていて運営する組織も別々のコンピュータを１つの仮想的な組織としてまとめあげ、ユーザに対してはポータルという形で簡単なインタフェースを提供し、ユーザが簡単に複数のスーパーコンピュータを利用することができるようになっている。今後 Grid の普及に向けては、エンドユーザに対してこのような簡単なインタフェースを持つポータルの提供は必須であると考えられる。

ポータルの現状および今後の動向を以下にまとめる。

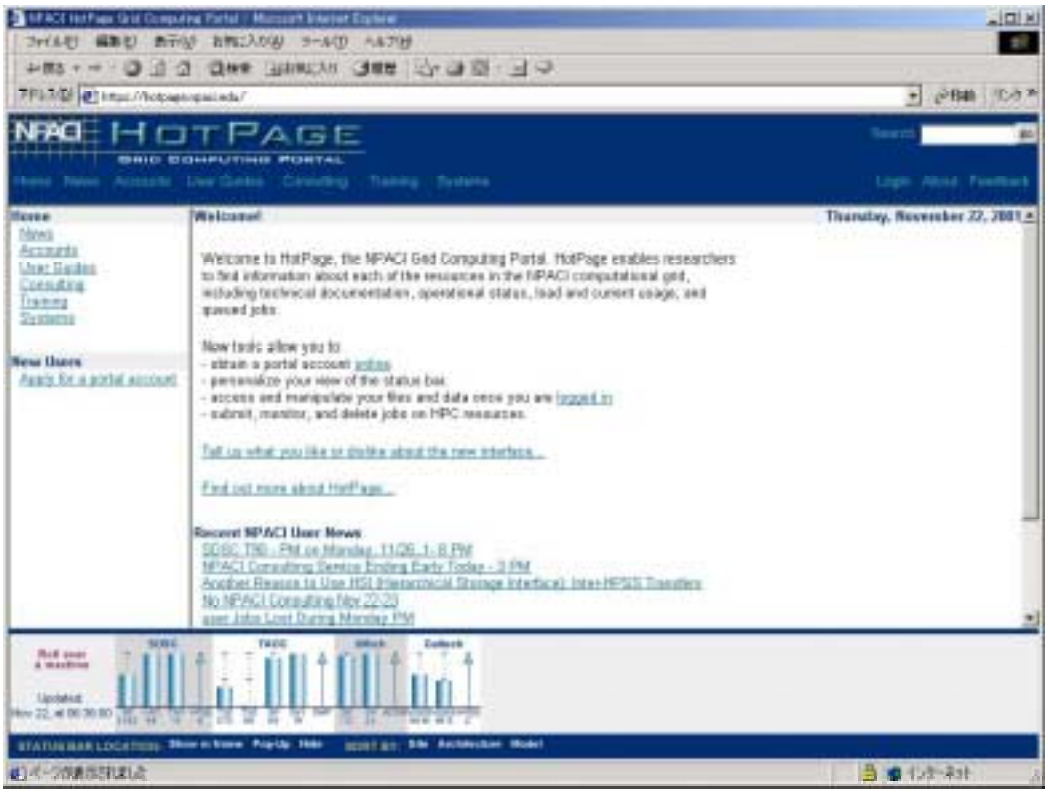


図５　NPACI HotPage

- Gaussian やAmber などのアプリケーションポータルはすでに実用可能なレベルにある。
- ScaLAPACK などのライブラリレベルのものを利用するポータルの場合はクライアントとのインタフェース等を考慮する必要があり、簡単ではない。
- ポータルを構築するためのポータルツールキットが開発されれば、既存のソフトウェアを容易にポータルで利用できるようになる。
- Grid におけるビジネスモデルとして Application Service Provider(ASP)は非常に重要であると考えられており、今後もポータルの研究開発は重点的に進められると考えられる。

5.　Access Grid と SC Global

5.1　Access Grid

Access Gridは人という情報資源へのアクセスを支援するプロジェクトおよびソフトウェアである。その基本部分は大規模ビデオ会議システムであり、IPマルチキャストで通信するために拠点数のスケーラビリティが高いことや、コモディティPC上のソフトウェアで

構成されるので構成の自由度が高く、手を入れやすいことなどが特徴である。プロジェクトはArgonne National Laboratoryが始めたもので、メーリングリストには現在300名を超えるメンバが登録している。知られている範囲で北米、ヨーロッパ、南米、アジア太平洋地域では日本、オーストラリア、中国に約80のサイトがあり、それ以上の数のノードが運用されている。ネットワークを用いたComputer Supported Collaborative Workというアイディアは特に新しいものではない。しかし、Access Gridのように世界規模で高品質なインタラクションが可能になったのは近年著しいネットワークの普遍化と広帯域化の恩恵であり、最近ようやく現実のものとなって実証の環境が整ったと言える。

日本では産総研、東工大，九州大学にてAccess Gridノードを構築すると同時に、Access Gridでの利用に耐える数Mbpsから100 Mbps以上のIPマルチキャスト接続を設けて運用してきた。図6に産総研の設備を示す。

現在、ほとんどのビデオ会議システムはTCP/IP上ではH.323というITU-T勧告に準拠している。これは呼制御や端末間のネゴシエーションなどを定めたプロトコル群で、Microsoft社のNetMeeting、Polycom社のViewStationやViaVideoなど多くのシステムが準拠しており、準拠システム間では相互運用が可能である。ビデオ会議の手段としてだけ考えた場合、これらH.323端末に対するAccess Gridの利点は拠点数が増えても映像、音声といったメディアの質が損なわれないというスケーラビリティである。これは、IPマルチキャストを利用するか否かという違いから生じている。H.323は基本的に一対一のビデオ会議のための規格であり、端末間に呼び出し側と応答側という関係を仮定しているため、

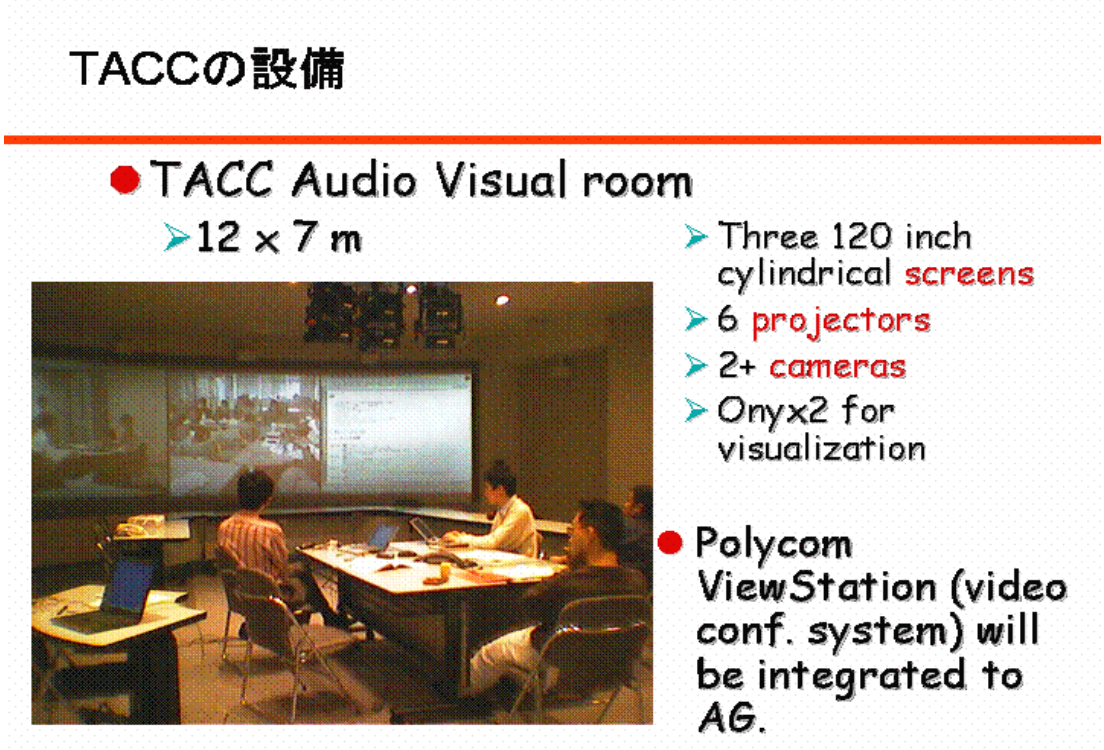


図 6　産総研の設備

複数の拠点で会議を行うためにはMCU（多地点接続装置）が必要となる。多拠点会議では、全H.323端末がMCUを相手にユニキャストで通信し、MCUは全端末からの映像と音声をミックスしてその結果を各端末に送る。この際に各端末がMCUから受け取る映像はあくまで一画面分に限られ、全拠点分をQCIF（Quarter-Common Interface Format，176x144 pixel）やFCIF（Full-CIF，352x288 pixel）の限られた一画面に収めたものとなる。MCUは、一般的には画面を分割したり各拠点からの映像を適当なタイミングで切り替えることで一画面に収める。

一方、Access Gridは当初より多拠点を指向しており、IPマルチキャストの利用を前提としている。どの拠点も対等であり、どの拠点も全拠点からの完全な映像と音声を受け取る。各拠点からの映像はPC（Windows，Linux）のウィンドウとして表示されるので、どういった大きさ、配置で表示するかは各拠点の自由である。MCUが構成したお仕着せの一画面を全員で見るといった不自由はない。また、Access Gridノードのハードウェアは PCであるため、もし縦横比3:4の一画面では手狭であれば数画面に分けて複数のモニタやビデオプロジェクタに出力することも容易である。Access GridはPC上のソフトウェアとして実現されており、実体は既存のソフトウェアといくつかの独自ソフトウェアの集まりである。そのため、ソフトウェアを差し替えたり改良、拡張することが可能である。また、ハードウェア構成の自由度が非常に高い。現に、映像の送受信に用いられるソフトウェアVICはAccess Grid向けに改造、機能追加がなされている。また、遠隔地の多数の聴衆に対するプレゼンテーションに用いられるDPPT（distributed PowerPoint）もAccess Gridのために開発されたものであり、Access Gridの拡張可能性を示している。ハードウェア構成の自由度が高いため、PCにビデオカードを追加したり複数ディスプレイへの出力が可能なビデオカードを用いることで容易に表示画面数を増やせる。これもPCをプラットフォームとしていることの利点である。

Access Gridの構築に際しては、機材の中心はコモディティPCであり、1台から4台用いる。表示用にWindowsを1台、映像キャプチャ用と音声キャプチャ用にそれぞれLinuxを1台ずつ用意することが推奨されている。エコーキャンセラやビデオカメラの制御用という名目でもう1台用意することが推奨されているが、実際はほとんど不要である。図7に産総研が構築したコンパクトなAccess Grid装置およびデバイスを示す。

PC以外に音声および映像用のデバイスが必要であるが、カメラ、マイク、プロジェクタ以上に重要であるのがエコーキャンセラである。エコーキャンセラを用意できない場合スピーカではなくヘッドホンで音声を聞くことになり、ヘッドホンの数が参加可能人数を制限してしまう。もしエコーキャンセラを持たないノードでスピーカを使うとスピーカからの音声を再びマイクが拾って送出してしまい、他のノードからの参加者はエコーを聞くことになる。2拠点間のビデオ会議ならともかく拠点数が多ければそれだけエコーの発生源も増えるため、Access Gridではこの問題は非常に大きい。

IPマルチキャストを前提としていることがAccess Gridの特徴のひとつである。これによ

図７　産総研の Access Grid ノードおよび機材

って、すべての拠点において全拠点からの映像と音声を完全な形で手に入れることが可能となっている。しかし、実際にIPマルチキャストを利用できる環境の構築は敷居が高い。Access Gridで「ロビー」と呼ばれる仮想会議室には常時30から40のノードが常駐していて、トラフィックは20 Mbpsに達する。それを受信し切れない場合、映像と音声の質が低下するだけで参加そのものができないわけではないが、同じ仮想会議室に入る拠点数が多ければ多いほど必要な帯域は増えるため許容帯域幅が広いに越したことはない。これまでのIPマルチキャストのアプリケーションであった一方向の放送や小規模（数拠点）ビデオ会議では必要な帯域幅はせいぜい数十Kbpsから数百Kbpsであった。広帯域IPマルチキャスト網の構築, 応用技術という観点でAccess Gridは興味深いアプリケーションだと言える。

5.2　SC Global

SC2001と同時に、世界各地で様々なイベントを共有しようというGrid上の国際会議SC Globalが開催された。これは、Access Gridを用いて世界中からSC2001に参加できるようにすると同時に、各サイトでもイベントを企画、開催し、それを世界各地で共有しようという試みであった。SC2001の展示会場にShowcaseノードが設けられた他、テクニカルプログラムのために3部屋（収容人数450、353、162人）にAccess Gridノードが設置された。遠隔からの参加サイト数は、9月時点のリストに載っているもので北米33、ヨーロッパ4、アジア太平洋地域3、ブラジル1、南極1であった。アジア太平洋地域の3サイトは、産総研、

シドニー大学、北京航空航天大学（BUAA）である。しかし、実際には東工大、九州大学、中国の清華大学などからの参加もあった。図8にSC2001展示会場のShowcaseノードの様子を示す。　プログラムは、Showcaseプログラムと3並列のテクニカルプログラムから構成されていた。テクニカルプログラムにはSC2001の中継としてKeynote、Gordon Bell Finalist Showcase、そして4つのテクニカルセッションが取り上げられた他、29の提案されたイベントが行われた。Showcaseプログラムの中には南極（Center for Astrophysical Research）からの中継もあり、参加者の興味をひいていた。この中継には17ノードが参加し、映像音声合わせて81のストリームが流れたとのことである。

我々もパネルディスカッションを提案し、運営にあたった。アジア太平洋地域のGrid研究協力体制ApGridの立ち上げ時期であったので、「Can the Asia Pacific Grid Contribute to the Science and Technology in the Asia Pacific Region?」と題し、各国参加組織の代表による発表と議論を企画した。このパネルでは、デンバーの会場、オーストラリアのシドニー大学、および産総研の三箇所をAccess Gridで接続し、日本、韓国、台湾、タイ、オーストラリア、米国の6カ国からパネリストが参加して行われた。司会をデンバーにて東工大の松岡先生が務め、日本からはパネラとして産総研大蒔情報処理研究部門長が参加した。図9にApGridパネルセッションの様子を示す。

図 7　SC Global Showcase

図 9　ApGrid パネルセッション

また、Data Gridに関するパネルディスカッションには日本から高エネ研渡瀬先生が参加した。その他、産総研の関口、田中も別のイベントのパネラを務めるなど多方面から参加、貢献した。

近年、家庭にまで数Mbpsのネットワークが普及しつつあり基幹ネットワークの広帯域化も進んでいる。ネットワーク越しのビデオ会議、Computer Supported Collaborative Workという考えは特に新しいものではないものの、広帯域ネットワークの普及にしたがってその現実性は急速に増しつつある。Gridの本質であるVirtual Organizationの運営にはこのようなネットワーク越しのビデオ会議は非常に重要であり、今後セキュリティや情報サービスなどのGrid技術を取り入れた充実した機能の提供が望まれる。

6. まとめ

本稿では、SC2001における講演や研究展示をふまえ、Gridの現状および動向について報告した。Gridは基盤技術が成熟してきており、今は実際のアプリケーションを動かす段階にきている。Globus Toolkitは洗練された基盤ソフトウェアであり事実上の標準となっているが、アプリケーションユーザが直接利用するのは難しい。計算基盤、情報基盤としてGridを普及し、Gridのユーザを広げるためにはより上位に位置するミドルウェアやポータルの研究開発が必要であり、実際に世界中で研究開発が加速されている。また、Gridの本質である仮想的な組織による運営(Virtual Organization)に際しては、電子メールだけでの交流ではなく、実際に顔を顔をあわせての運用が重要である。Access Gridは多地点接続

を考慮し、IPマルチキャストを用いた質の高いネットワーク会議システムである。現時点ではIPマルチキャストがどこでも使える技術ではないことや、それなりの帯域を必要とする点など、特にネットワークインフラの整備が遅れているアジア地域における利用には若干の問題は残るが、今後セキュリティや情報サービスなどのGrid技術を取り込むことにより、より豊富な機能を持ち、より性能の高いシステムとなり、今後のGridテストベッドの運営に有用な手段となることは間違いない。

3.4.2　地球シミュレータ開発を終えて

横川　三津夫　委員、　妹尾　義樹　委員

1.　はじめに

平成 9 年度から開始した地球シミュレータ開発は、平成 14 年 2 月末にピーク性能 40 テラフロップスの世界最高速を達成し、成功裡にその開発が終了した。Linpack ベンチマーク性能では、638 台の計算ノード、5104 プロセッサにて 35.61 テラフロップス（ピーク性能比 87.2%）という驚異的な実効性能を達成した。これは、米国 ASCI White システムが持っていた世界一の記録（7.22 テラフロップス）の約 5 倍にあたり、ピーク性能、実効性能ともに世界一と認められたわけである。

実際の応用プログラムにおいても、大気大循環モデルの T1279L96（格子点数 3840×1920×96）において、計算ノード 320 台を用いて 14.5 テラフロップス（ピーク性能比 70.8%）の高い実効性能を達成しており、地球シミュレータを利用した今後の数値シミュレーションの成果に大きな期待がかかっている。(末尾に付録として大気大循環と海洋大循環の計算結果例を添付しておく。海洋の方では、黒潮が見えていることに注目。)

当初から開発に携わってきた我々としては、たった 5 年間の開発期間ではあったが、さまざま出来事が思い浮かび感無量の気持ちで一杯である。本稿では、開発の経緯を振り返るとともに、完成した地球シミュレータのハードウェア及びソフトウェアの概要、並列プログラミングインタフェース、及び応用プログラムの性能などについて述べる。

2.　地球シミュレータ開発の始まり

地球シミュレータ開発が決定された平成 8 年頃は、地球温暖化やエルニーニョ現象等の地球規模の現象が注目されていた。温室効果ガス等の排出は、地球温暖化に甚大な影響を及ぼしていると言われており、排出削減目標が設定されるなど我々の日常生活にも大きな影響を与えると考えられた。また、エルニーニョ現象は局所的な集中豪雨や干ばつなどの被害をもたらすものとの推測がなされていた。

旧科学技術庁航空・電子等技術審議会地球科学技術部会は、平成 8 年 7 月に取りまとめた報告書「地球変動予測の実現に向けて」において、地球変動予測研究の推進にあたっては、地球変動プロセス研究、地球観測及び数値シミュレーションを三位一体として、総合的かつ計画的に研究開発を推進すべきであるとの提言を行った。これを受けて、旧科学技術庁計算科学技術推進会議地球シミュレータ部会は、平成 9 年 7 月に報告書「『地球シミュレータ』計画の推進について」をまとめ、数値シミュレーション分野の推進について、地球規模の複雑な諸現象をシミュレートできる超高速並列計算機システム「地球シミュレータ」の開発と、地球シミュレータ上で動作する高度な応用ソフトウェアの開発を目標とする計画を策定した。この報告書では、近年の科学技術分野に特化したスーパーコンピュ

ータのめざましい発展とそれに基づく計算科学の進展について述べられ、観測困難な現象や実験不可能な現象を解明するための強力なツールとして、数値シミュレーションによる方法が非常に重要であるとされている。これに基づき、気象・気候分野の代表的な数値シミュレーションにおいて、約 5 テラフロップス（TFLOPS：1 秒あたり 1 兆回の浮動小数演算を実行できる計算処理速度を表わす単位）の実効処理性能を有する大規模並列計算機システム「地球シミュレータ」の開発が開始された訳である。

平成 9 年度の開発開始時点では、宇宙開発事業団及び旧動力炉・核燃料開発事業団の共同プロジェクトとしてスタートしたが、平成 10 年度からは宇宙開発事業団、日本原子力研究所及び海洋科学技術センターの 3 者により開発が実施されることとなり、共同プロジェクトチーム「地球シミュレータ研究開発センター」を組織し、完成までの研究開発を行った。平成 14 年 2 月の地球シミュレータ完成を以って、このチームは解散した。

3.　地球シミュレータ施設

海洋科学技術センター横浜研究所（横浜市金沢区）に建設された地球シミュレータ施設は、図 1 に示すように、シミュレータ研究棟、シミュレータ棟、冷却施設棟から構成されている。

地球シミュレータが設置されるシミュレータ棟（図 2）は、幅 50m、奥行き 65m、高さ 17m の鉄骨 2 階建て、免震構造をしている。3 層構造をしており、第 1 層は計算機システムを冷却するための空調機器などが設置される。高さ約 170cm の第 2 層のフリーアクセ

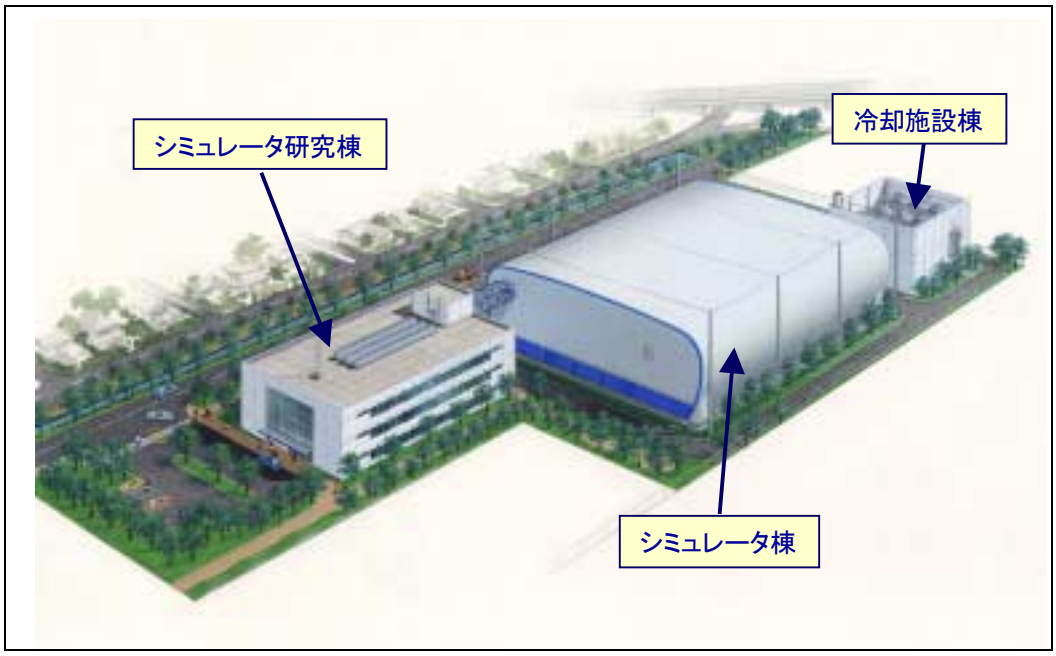


図１　地球シミュレータ施設

図２　シミュレータ棟の構造

スフロアは、計算ノード（PN）と結合ネットワーク（IN）を繋ぐ 83200 本の電気ケーブル（全長約 2800km）を収納する他、冷気を計算機システム全体に分配する空調ダクトスペースとして利用している。第 3 層は、地球シミュレータ本体及び周辺機器が設置されるスペースである。

また、この施設の特徴として、地球シミュレータの誤動作の原因となる外部からの電磁波を完全にシールドしており、また設置スペースではライトガイド式照明システムを採用し、蛍光灯による照明システムで発生する電磁ノイズを除去した。ライトガイド式照明システムとは、計算機室外部に設置された光源部（マルチハロゲン灯）からの光だけを特殊フィルムに反射させて照明するものである。これにより、電磁ノイズを軽減するばかりか、ランプ交換等の保守性向上、天井の空気排出面積の増大による冷却効果の増加などの効果がもたらされた。

4.　地球シミュレータのハードウェア

地球シミュレータのハードウェアが達成すべき計算性能は気象・気候分野の数値シミュレーションで 5 テラフロップス以上の実効処理速度である。ハードウェアの設計指針については既にいろいろなところで述べているのでそちらを参照されたい［1, 2］。

地球シミュレータは、図 3 に示すように、640 台の計算ノードをクロスバネットワークで結合させた分散メモリ型並列計算機である。各計算ノード（PN: Processor Node）は、ピーク性能 8Gflops のベクトル型計算プロセッサ（AP: Arithmetic Processor）8 台が主記

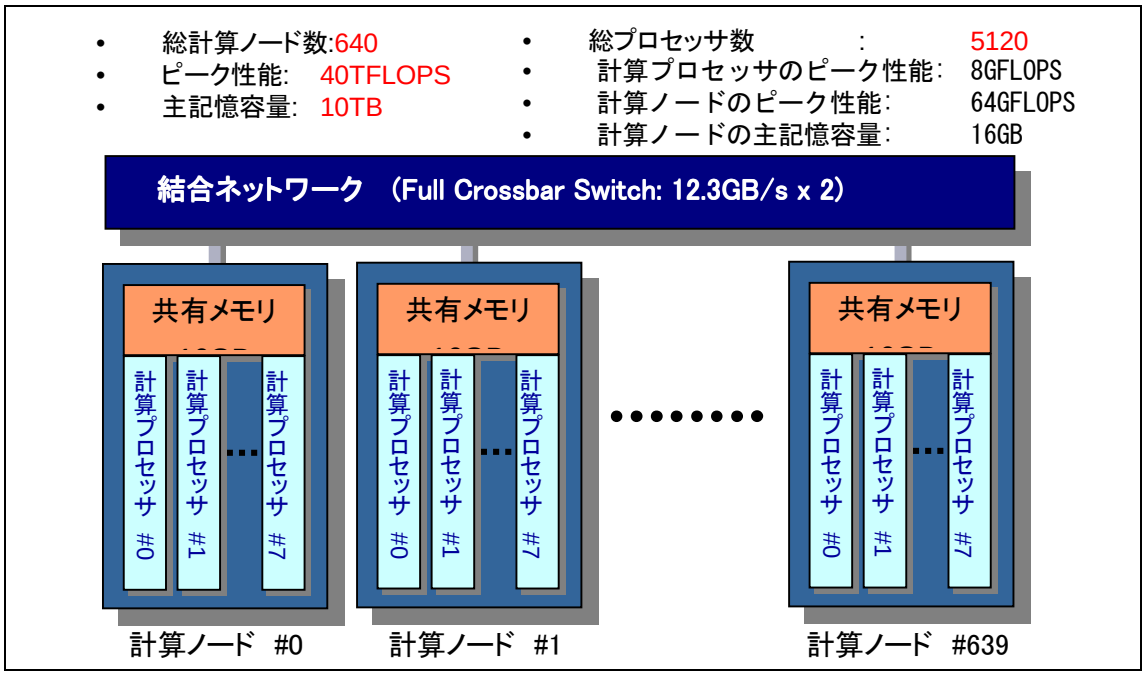


図３　地球シミュレータのハードウェア構成

憶装置 16GB を共有する共有メモリ型並列計算機となっている。したがって、全体では AP が 5120 台、ピーク性能は 40Tflops、主記憶容量 10TB となる。各 PN は、8 台の AP、32 台の主記憶ユニット（MMU: Main Memory Unit）、リモートアクセス制御装置（RCU: Remote Access Control Unit）及び入出力プロセッサ（IOP:I/O Processor）から構成されている［3, 4］。 外部磁気ディスク装置は全部で 600TB、カートリッジライブラリシステムは 1.5PB の容量を持っている。

AP は、ベクトル処理部（VU: Vector Unit）、スカラ処理部(SU: Scalar Unit)、プロセッサネットワークユニット（PNU: Processor Network Unit）及びアドレス制御部（ACU: Address Control Unit）から構成され、1 つの LSI 上に実装される。AP 用 LSI のクロック周波数は 500MHz（一部 1GHz）である。SU は、4 ウェイのスーパースカラであり、128 個の汎用レジスタ、2 ウェイセットアソシアティブ方式の命令キャッシュとデータキャッシュをそれぞれ 64KB づつ実装している。また分岐予測や投機実行の機能を持つ。VU は、6 種類（加算、乗算、除算、論理、ビット列論理、ロード/ストア）のベクトル演算器と 72 個のベクトルレジスタからなるベクトル演算器セット 8 個で構成され、最大 8Gflops の性能を有している。32 台の MMU には、主記憶素子として DRAM ベースの 128Mbit の高速 RAM を採用し、2048 バンク構成とした。各々の AP は、主記憶システムとの間に 32GB/s のバンド幅を持っており、1PN で 256GB/s を確保している。

RCU は、クロスバネットワークと直接接続され、クロスバを介した送信、受信を AP と独立に動作させることができる。クロスバネットワーク（IN: Interconnection Network）

は、128 台のクロスバスイッチを実装したデータパス部（XSW: Internodes Crossbar Switch）と 2 台の制御部（XCT: Internodes Crossbar Control Unit）から構成されている。XCT は、PN 間の転送経路の予約、競合調停等を行い、128 台の XSW を介して実際のデータが転送される。IN 及び RCU は、PN 間データ転送機能、PN 間同期等の機能を持つ。データ転送機能では、同期型転送と非同期型転送がサポートされており、主記憶上の連続した領域のデータを転送するブロック転送の他に、非同期型転送としてストライド付きベクトル転送、リストベクトル転送が可能である。

地球シミュレータでは、省スペース及び省電力を図るために、半導体技術、パッケージング技術、実装技術、冷却及び電源技術等に 21 世紀初頭の技術を大胆に取り入れた。LSI 加工技術では、0.15μm の CMOS テクノロジーと銅配線技術を用いた。これによって、CMOS によるベクトルプロセッサで 500MHz（一部 1GHz）の高速動作を可能にした。また、1 チップ・ベクトルプロセッサを達成するために LSI の大型化を追求し、約 20mm×20mm のダイに約 6000 万トランジスタ、約 5000 ピンを実装した。パッケージング技術では、微細配線加工技術を駆使した大型ビルドアップ基板上に、LSI をベア実装する技術を用いた。さらに、冷却技術ではヒートパイプによる冷却方法を採用し、約 140W にも達する AP 用 LSI の発熱に対処した。図 4 に計算プロセッサ（AP）のパッケージ図（左）と写真（右）を示す。

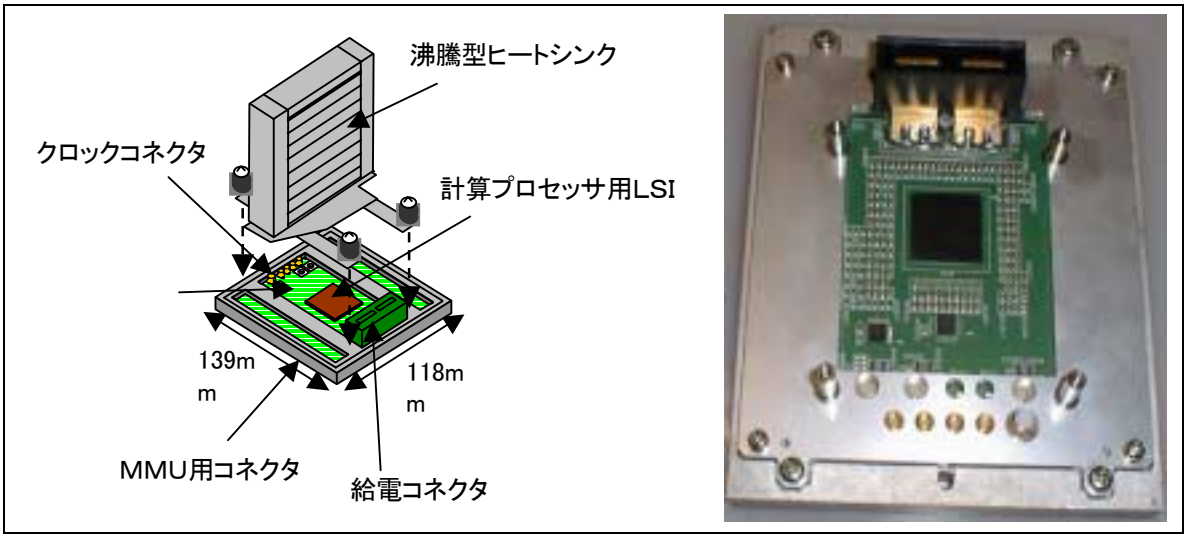


図４　計算プロセッサ（ＡＰ）パッケージ

5.　地球シミュレータのシステムソフトウェア

本節では、地球シミュレータのシステムソフトウェアについて、特に高並列処理とこれを支えるプログラミングインタフェースを中心に解説する。

5.1　地球シミュレータシステムソフトウェアの概要

地球シミュレータのシステムソフトウェアは、図 5 に示すとおりである。地球シミュレータは、640 ノード 5120 プロセッサという世界最大の計算リソースが結合されたシステムである。スケーラビリティー拡大は、この大規模な計算リソースが効率よく動作させるための機能拡張である。地球シミュレータ特有機能には、専用ジョブスケジューラ、並列ファイルシステム PFS、バッチデバッグ機能、地球シミュレータ専用ハードウェア制御機能、HPF 地球専用機能などが含まれる。ジョブスケジューラは、5120 のプロセッサ、メモリ、I/O などの巨大リソースをスケジュールするためのソフトウェアで、ジョブの効率よいスケジューリングやステージング・デステージングによる I/O 処理と計算処理のオーバラップなどにより、システム全体の稼働率を高め、しかも投入されたジョブのターンアラウンド時間を最小化するように設計されている。

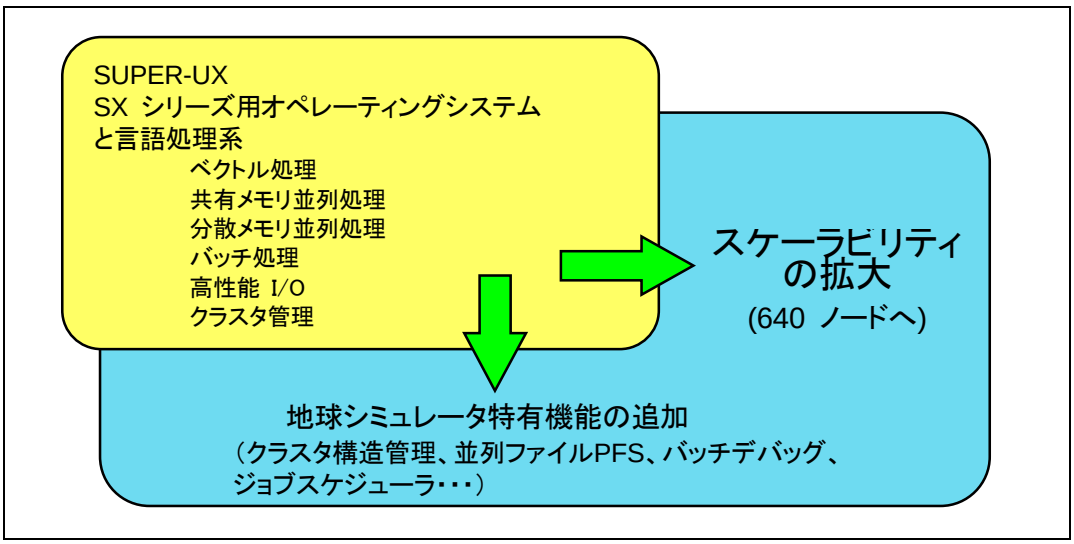


図５　地球シミュレータのソフトウェア構成

図 6 に、地球シミュレータ上で巨大ジョブが実行される様子とこれを支えるシステムソフトウェアの特徴を示す。地球シミュレータの複数ノードを用いる計算は分散メモリ上のプログラミングが必要になるため、MPI（Message Passing Interface）または HPF(High Performance Fortran)のいずれかでプログラムが作成される。コンパイル、リンク処理を経て生成された実行形式プログラムは、基本的にバッチ処理としてジョブスケジューラに投入される。プログラムは、ハードウェアと MPI によって実現された高性能のノード間通信を用いて実行され、また、複数ジョブは効率よくジョブスケジューラにより計算リソ

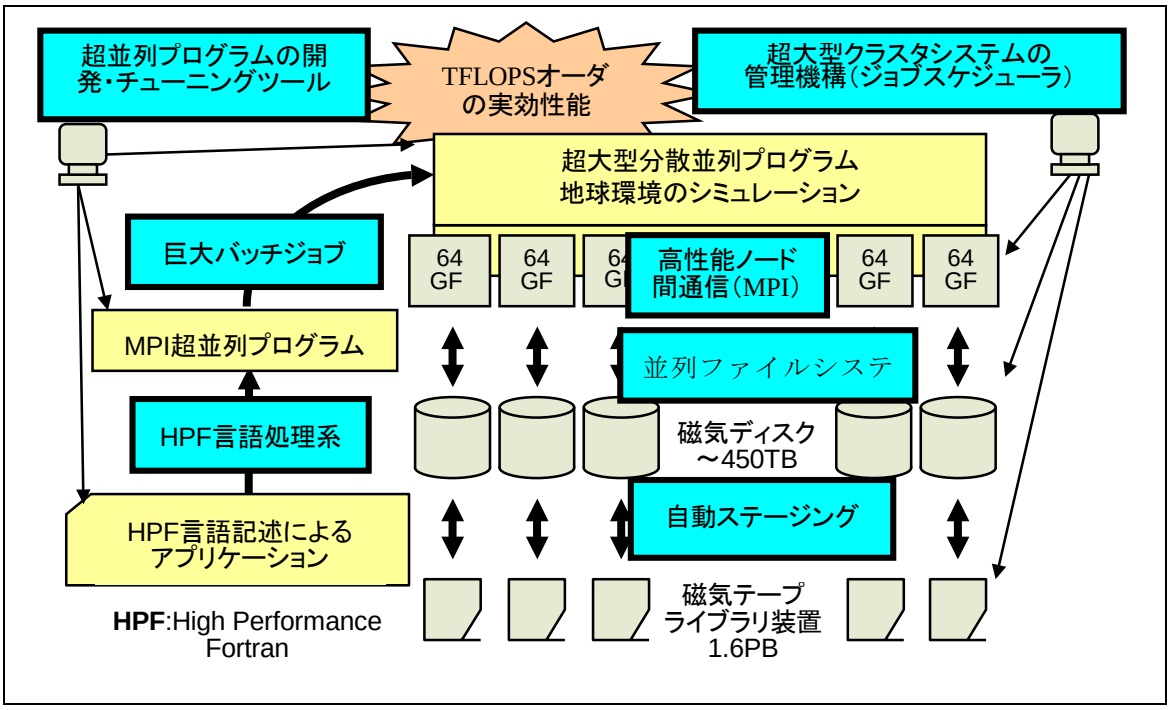


図 6　地球シミュレータソフトウェアの特徴

ースが割り当てられる。ファイルシステムは、ノードごとにバラバラに扱うこともできるが、並列 I/O が必要なプログラムは、並列ファイルシステム PFS を Fortran や C でサポートされる通常のファイルアクセスで利用可能である。また MPI-IO を用いて PFS を用いることも可能である。大量のデータは最終的には磁気テープライブラリに格納されることになるが、このライブラリと磁気ディスクとの間のステージング・デステージングは自動化が可能である。また、高並列マシンで重要となる、プログラムのデバッグ、チューニング作業をサポートする種々のツール群も用意されている。

5.2　地球シミュレータのオペレーティングシステム

地球シミュレータの OS は、640 ノードという多数の計算リソースを管理するために、クラスタ構造を導入して制御している。クラスタとは、16 ノードをまとめたものであり、OS によるシステム管理の単位である。また、大規模分散並列処理に対応するために、高速ファイルアクセスが可能なファイルシステムを導入するとともに、並列ファイル入出力機能を提供している。運用におけるジョブ配置では、1 つのクラスタ（S クラスタ）を会話型処理、小規模バッチ処理用とし、その他の 39 個のクラスタ（L クラスタ）を大規模バッチ処理用に割り当てることを想定した機能を充実させた。

地球シミュレータ研究開発センターが独自に開発した運用・管理ソフトウェアは、ジョブスケジューラ、大容量テープライブラリシステム上のファイル管理システムなどから構成されている。L クラスタ群では、単一ジョブ環境を提供しているが、そのためのジョブ

配置は使用するファイル等の配置を含めて非常に複雑である。ジョブスケジューラは、ジョブが実行されるべき計算ノード予約やそのノードにローカル接続されている磁気ディスクへのファイル配置、ジョブのアロケーション等複雑な制御を行い、地球シミュレータ利用効率の向上を目指している。

地球シミュレータの OS 開発でジョブスケジューラの開発とともに、最も苦労した点は、5120 という膨大な数のプロセッサが結合された巨大システム上でスケーラビリティーを確保することである。スケーラビリティーについては、地球シミュレータの開発スタートから、OS のすべての部分について詳細な検討が行われた。解決したスケーラビリティーの問題は大きく分けて、次のものがある。

(1) 性能的問題

逐次的にすべてのノードに渡って処理を行うことを避ける必要がある。たとえば、マルチノードジョブの開始にあたり、マスターノードがすべてのノードに順次必要なリソースをコピーするような、ノード数ｎとしてｎに比例する時間を要する処理は徹底的に排除する必要がある。バイナリーツリーなどの方式で、すべて log n の処理に置き換えられている。集中制御を分散制御に置き換えて、1 個所へのアクセス集中を避けることも重要である。

また、マルチノード間のリソースの排他制御などのためのノード間ロック制御の高速化も重要である。単に、ロック取得開放処理の高速化だけではなく、ロック取得要求がスピンウェイトを構成するなどして、ノード間の通信機構に負荷をかけないようにすることが肝要である。数百台のノードが取得失敗を繰り返すとシステム全体に非常な負荷をかけることになる。

(2) 容量的問題

リソースの管理などで、オーダｎの記憶容量を要する管理を徹底的に排除する。通信バッファとして用いる領域も、通信相手ごとに確保するような方式が採用できない。また、リソースにより、トータルの容量諸元が膨大となるため、これら諸元を管理するビット幅にも注意を払う必要がある。

(3) タイムアウト関連

ノードの故障検出など種々の異常の判断にタイムアウト（一定間隔の無応答）が用いられるが、これらの値の最大システムでの最悪値を厳密に推定し、設定することが重要である。OS 処理で高速性が必要でノード間高速通信装置（IN）を用いるものがあるが、MPI が大容量のブロック通信を処理していると、これらが待たされるケースがある。MPI の高速通信を妨げないようにしつつ、OS 処理でタイムアウトを発生させないために種々の工夫が必要であった。

(4) 故障対策

640 ノード 5120 プロセッサがあると、いくらそれぞれの構成要素の信頼性が高くと

も、一定以上の故障率を覚悟する必要がある。このためには、1 構成要素のエラーでシステム全体がストップすることのないシステム管理方法が重要となる。また、システムを落とさずに構成要素の交換や復旧処理が行えることも重要である。

5.3　並列ファイルシステム

640 ノードをすべて使うような巨大なジョブが 1 ノードに接続されたディスクを共有して大量の I/O を行うと、大きくスケーラビリティーを損なう。また、それぞれのノードがローカル I/O を明示的に記述することで高性能 I/O を実現できるが、プログラミングが大変であり、また生成された数百個のファイルの管理も大変となる。これらの問題を解決するために、ファイルシステムとしてはユーザからはグローバルに唯一のものとして扱え、しかも各ノードからの並列 I/O 処理を実現したのが、並列ファイルシステム PFS である。PFS の構成を図 7 に示す。

PFS 上のファイルは、ストライピングされ、指定されたブロック単位でサイクリックにそれぞれのノードに接続されたファイルシステムに格納される。このファイルにプログラムからアクセス要求が出ると、FAL と呼ばれるライブラリがアクセス対象となるデータの位置を同定し、IN ネットワークを用いて対応するノードの File Server に要求を出す仕組みになっている。各ノードから特定のノードに接続された物理ディスクにアクセスが集中すると性能は出ないが、そうでなければ、並列 I/O による性能向上が得られる。

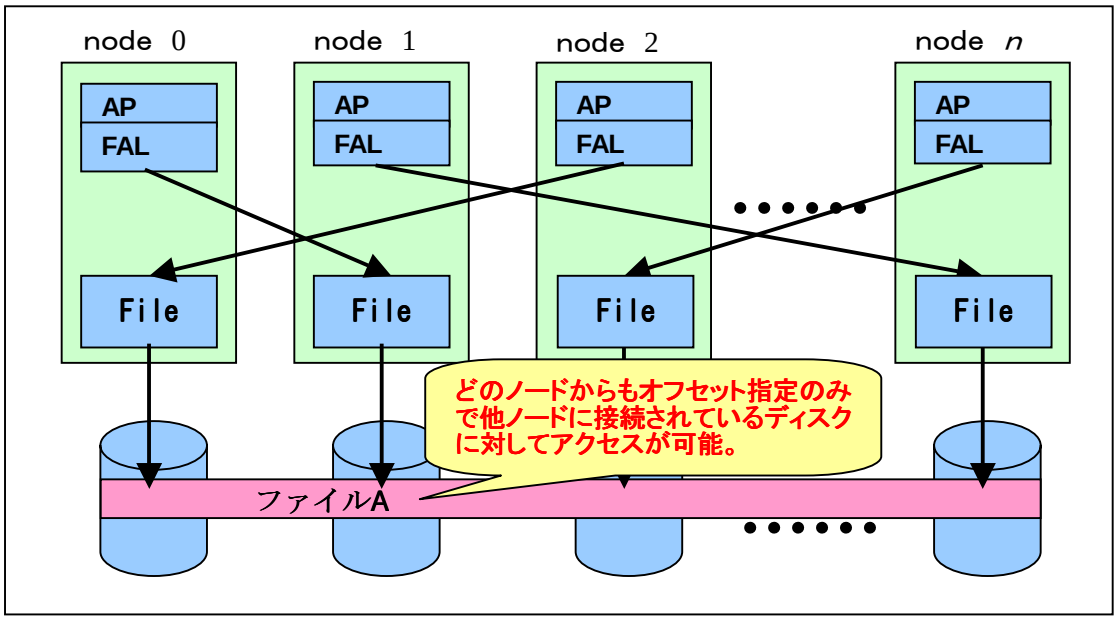


図 7　並列ファイルシステム PFS

5.4　地球シミュレータのプログラミングインタフェース

地球シミュレータで満足できる性能を得るには、1 プロセッサ内のベクトル処理、1 ノード内 8 プロセッサの共有メモリ並列処理、ノード間の分散メモリ並列処理のすべてを効率よく行う必要がある。図 8 に、これらそれぞれで利用できるプログラミングインタフェースを示す。これらは、基本的には SX シリーズで提供されるものと同じであるが、地球シミュレータではノード数が多いため、ノード間のスケーラビリティーが得られるように強化がなされている。

ここで、ノード内の共有メモリ並列化においても MPI と HPF がサポートされていることに留意する必要がある。たとえば、5120 プロセッサの並列処理を行う場合には、ノード間を 640 並列の並列処理として MPI または HPF で記述し、ノード内を自動並列化または OpenMP を用いるハイブリッドの並列処理か、5120 プロセッサを対象として MPI または HPF を用いる 1 階層の並列処理のいずれかを選択できる。これらの正確な得失については、今後の種々のアプリケーション並列化と評価結果をまたねばならないが、一般にいって、ノード内での並列処理においてデータアクセスローカリティーが得られる場合には、ハイブリッドの方が有利である。たとえば、すべてのプロセッサが参照する read only のデータがある場合、ハイブリッドでは、これらのデータのコピーはプロセッサ数ではなくノードの数だけ用意すればよい。また、Linpack の場合には、係数行列をブロック分割して、ブロック単位に消去操作を進めるが、分散メモリシステムの場合、枢軸ブロック内の消去操作が逐次処理となり、並列化率を低下させる要因になる（これを避けるため、高並列システムでは、係数行列を碁盤の目状に分割して、枢軸ブロック内消去も並列処理ができるようにするのが一般的である。ただ、この場合には partial pivoting において、列方向の通信が発生する）。ハイブリッドの並列化では、共有メモリ内で、枢軸ブロック内消去が、

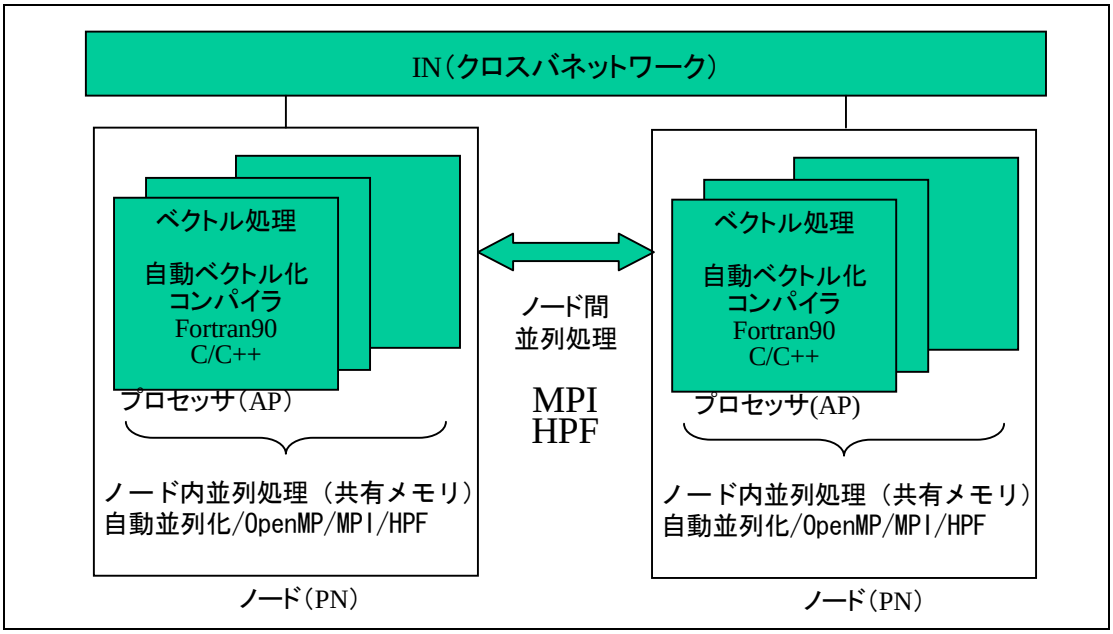


図 8　地球シミュレータ上のプログラミングインタフェース

これ以外の処理と合わせて負荷分散できるため、係数行列の行方向の分割のみでも、高い並列化効率が得られる。

一方で、1 階層の並列処理は、1 種の並列化インタフェースだけで並列処理が記述できるため、プログラミングはハイブリッドより容易である。また、同期、通信操作も、データアクセスローカリティーにおいて効率に差がなければ、ハイブリッドよりも、1 種の並列化インタフェースを用いる方が、わずかだが効率が良い（たとえば、5120 プロセッサのバリア同期を行う場合、自動並列化で 8 プロセッサ内の同期を行い、さらにノード単位で 640 ノードの待ち合わせを MPI で実行するより、5120 プロセッサの同期を MPI だけで行う方が効率が良い。ノード内の MPI の同期処理は共有メモリ機構を用いて効率よく実装されていることにも留意する必要がある）。

5.5 地球シミュレータの MPI（Message Passing Interface）

地球シミュレータのような大規模分散メモリ型計算機上でプログラムを効率的に実行するには、並列プログラムの高速化、特に通信処理の最適化が大切であり、そのためにユーザプログラムから利用できるライブラリのデータ転送性能値を評価することは重要である。ここでは、地球シミュレータの計算ノード内および計算ノード間での MPI の通信性能に関する測定結果を示す。地球シミュレータの MPI ライブラリでは、MPI-1 及び MPI-2 をサポートしており、MPI-1 の関数、MPI-2 の主要関数の性能評価を行った。MPI の性能測定には、新しく開発された MPI-1/MPI-2 関数の性能を詳細かつ多角的に測定するためのツール MPI Benchmark program library for MPI-1/MPI-2 (MBL1/MBL2)を用いることにした[5]。

地球シミュレータ上での仮想メモリ空間は、ローカルメモリとグローバルメモリの 2 つが存在している。グローバルメモリは RCU が直接アクセス可能な領域であり、グローバルメモリ間での RMA（Remote Memory Access）転送が可能である。グローバルメモリは全グローバルメモリを一元的に管理する仮想グローバルメモリ空間にマッピングされている。このグローバルメモリ領域中に、Fortran90 の動的割り付け（Allocate 文）によってユーザアプリケーション中の配列を割り付けることができる。

これらのメモリ空間を利用した MPI ライブラリの実装では、グローバルメモリを MPI ライブラリの通信バッファ（以下、MPI 通信バッファ）に用いている。したがって、MPI 関数の引数として指定するユーザが確保したデータ領域（送受信バッファ）をローカルメモリまたはグローバルメモリに置くか、さらにノード内通信及びノード間通信のどちらを行うかによって、MPI の内部処理は以下の 4 つのケースに分類される。

(1)送受信バッファがローカルメモリにあり、かつノード内通信が行われる場合
(2)送受信バッファがローカルメモリにあり、かつノード間通信が行われる場合
(3)送受信バッファがグローバルメモリにあり、かつノード内通信が行われる場合
(4)送受信バッファがグローバルメモリにあり、かつノード間通信が行われる場合

送受信バッファがローカルメモリにある場合は、MPI 通信バッファがとられるグローバルメモリ領域との間でメモリコピーが必要であるが、送受信バッファをグローバルメモリ上に確保した場合にはメモリコピーは必要ない。ノード間通信が行われる場合には、送信元ノードでの MPI 通信バッファであるグローバルメモリ領域から、送信先ノードでの MPI 通信バッファであるグローバルメモリ領域へクロスバネットワークを介したデータ転送が行われる。

この結果、内部処理におけるメモリコピーやデータ転送等の処理数（メモリコピー回数とノード間データ転送回数の和）は異なる。内部処理で用いられるメモリコピーはベクトル化されており、特にノード内のグローバルメモリを用いた通信は、グローバルメモリ間でのメモリコピー1 回でデータ転送が実現されているため高速処理が期待される。また、ベクトル化されたメモリコピーがパイプライニング処理されるために、データ量が大きい場合には 2 回のメモリコピー処理が 1 回とみなせるようになる。これによってノード内通信では、転送量が多くなるにつれて、送受信バッファをローカルに配置した場合の処理性能はグローバルメモリに配置した場合の処理性能に近づくと考えられる。

(1)MPI-1 関数の性能評価

MPI は、MPI/SX をベースに開発されており、MPI-2、MPI-IO がサポートされている。また、ノード間並列化におけるスケーラビリティーの改善はもとより、地球シミュレータの専用ネットワーク機能の利用インタフェースのサポートや、MPI プログラムのチェックポイントリスタート機能など種々の強化が行われている。

地球シミュレータ上で MPI-1 の関数について性能測定を行った。MPI-1 の同期通信関数 MPI_Send と非同期通信関数 MPI_Isend について、MPI ランク 0 とランク 1 のプロセス間で行われる pingpong 通信パターンでの性能を測定した。また送受信バッファをグローバルメモリに割り付けた場合と、ローカルに割り付けた場合の両方の性能を測定した。

まず、計算ノード内での MPI_Send と MPI_Isend による pingpong 通信パターンでのスループットを図 9 に示す。この計測結果から、MPI_Send、MPI_Isend 共に最大スループットは 64MB 転送時でほぼ飽和し、その値は 14.8GB/s であった。MPI_Send と MPI_Isend 間での最大スループットにおいて大きな差は見られない。なおグローバルメモリを使用した方が、256KB 転送時で最大約 50.6%、4MB 転送時で最大約 27%処理性能値が向上している事も確認できた。また、pingoing パターンでの最小レイテンシは、MPI_Send の 0B 転送について計測し、送受信バッファのローカル配置時で 4.86 マイクロ秒, グローバル配置時で 4.56 マイクロ秒であった。MPI_Isend についても同様に計測し、送受信バッファのローカル配置時で 5.12 マイクロ秒, グローバル配置時で 4.58 マイクロ秒であった。

また、計算ノード間での MPI_Send と MPI_Isend による pingpong 通信パターンでのスループットを図 10 に示す。グローバルメモリ使用時の MPI_Send、MPI_Isend 共に最

大スループットは64MB転送時で11.8GB/sが得られたが、ローカルメモリ使用時は64MB転送時で 8.7GB/s に留まっている。ノード間クロスバネットワークの理論最大性能は12.3GB/s であることから、11.8GB/s の実効性能は約 96%の効率を実現しており、十分高い実装であることが示された。この場合の最小レイテンシは、送受信バッファのグローバル配置時で 8.56 マイクロ秒であった。

図 11 に、1 対 1 通信をシステム全体で行った場合の転送性能合計値を示す。これは、全プロセッサが、同時に隣のプロセッサにデータを送信した際の転送性能の合計値である。512 ノードの際のシステムスループットは、5.8Tbyte/秒であり、12Gbyte/秒を 512 倍した値とほとんど遜色ない値が得られている。ほとんど同様の傾向なので、グラフは掲載しないが、MPI_PUT と MPI_GET の場合には、512 ノード時のシステムスループットは、6.2Tbyte/秒であった。

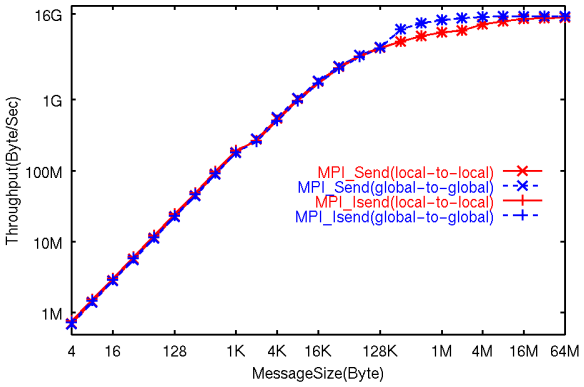


図 9　Pingpong 通信性能（ノード内）

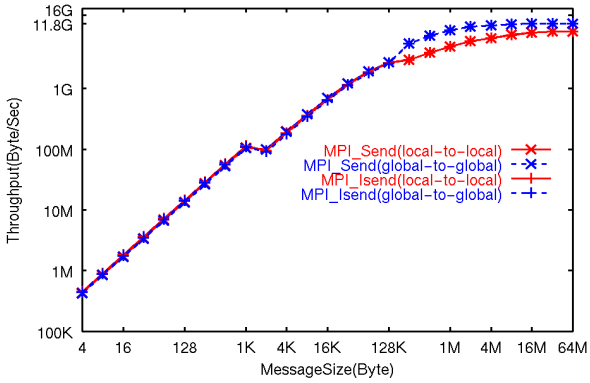


図 10　Pingpong 通信性能（ノード間）

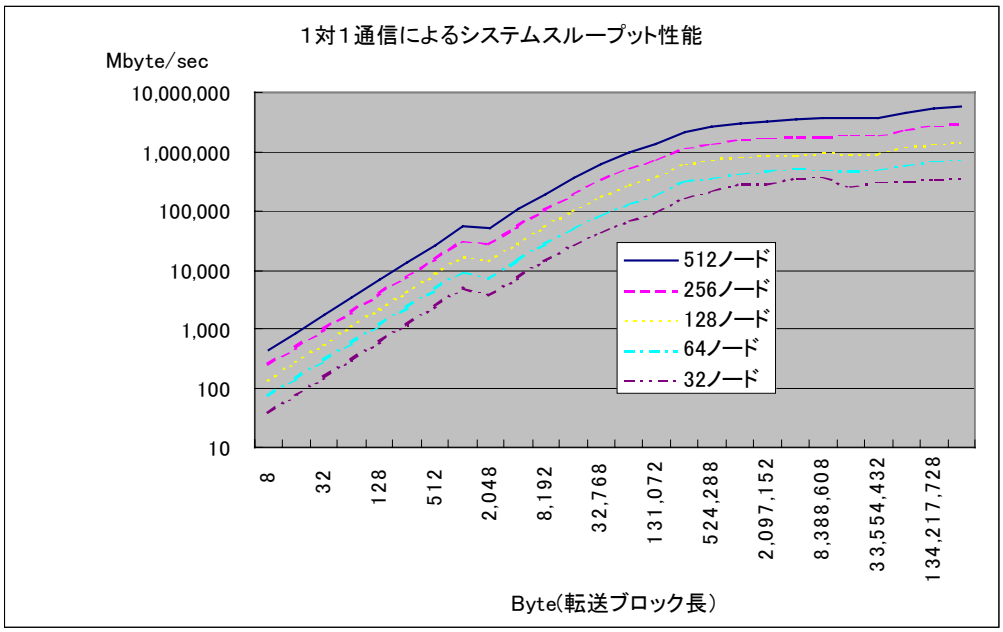


図 11　1 対 1 通信のシステムスループット

最後に MPI_Barrier の時間を図 12 に示す。ハードウェアの専用機能を用いているため、ノード数によらず約 3.3μ秒前後という高速バリアが実現されている。

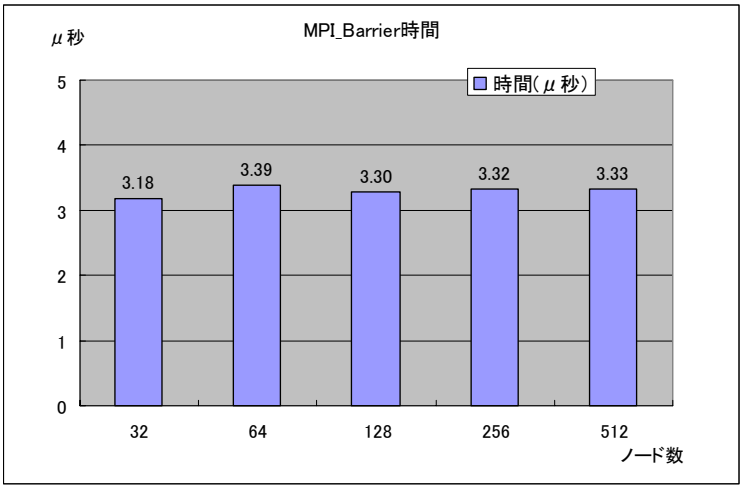


図 12　ノード間バリア性能

(2)MPI-2 関数に関する計測結果

地球シミュレータ上で MPI-2 の RMA 関数の性能測定を行った。MPI プロセス数を 2 とした時に、ランク 0 からランク 1 への MPI_Get, MPI_Put, MPI_Accumulate（reduction 演算は総和演算）による RMA データ転送について、RMA 関数起動から fence 処理終了までの処理時間を計測した。なお送信バッファや RMA ウィンドウ用の配列はグローバルメモリ上に割り付けた。

この性能測定結果として、MPI_Get と MPI_Put、および MPI_Accumulate 関数のスループットを図 13 に示す。MPI_Get および MPI_Put の最大スループットは 64MB 転送時に約 11.63GB/s で、MPI_Accumulate は 3.16GB/s であった。レイテンシは、4B 転送時においてそれぞれ 7.02 マイクロ秒、6.63 マイクロ秒、7.06 マイクロ秒であった。

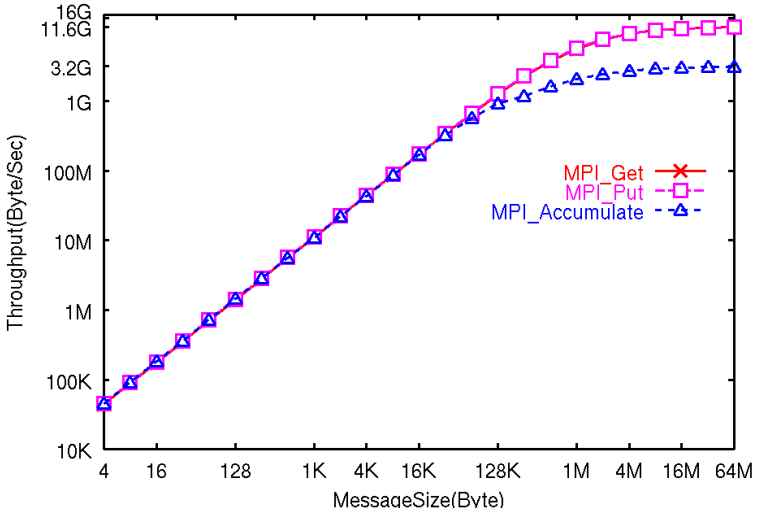


図 13　ノード間の RMA 関数の処理性能

5.6　地球シミュレータの HPF(High Performance Fortran)

地球シミュレータで実装されている HPF の機能を図 14 に示す。

機能としては、HPF/JA1.0 拡張仕様[8]をベースとして、これに独自機能として、HALO/REDUCE HALO の非定型通信の高速処理機能、ベクトル化指示認識機能、自動並列化機能（Independent 指示自動挿入機能）、並列 I/O 機能が付加されている。

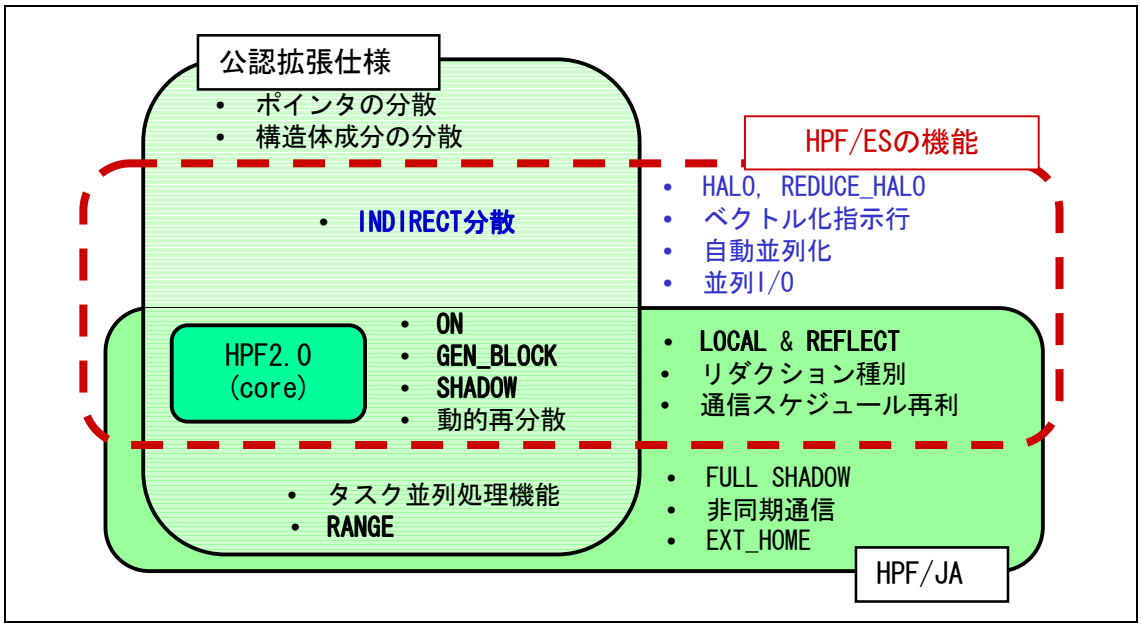


図 14　地球シミュレータ用 HPF の機能

HALO は、有限要素法や非定型の計算グリッドを用いたシミュレーションなどで現れる、連続ではない近傍参照エリアのことで、特定プロセッサから特定プロセッサへのデータ転送領域が非連続のため、高速通信には、データのパッキング処理が必要となる。HALO は、これらプロセッサ間通信が必要になるデータ参照エリアを明示的に指定する機能を言語として用意することにより、MPI で明示的に記述する並列処理と遜色ない性能を HPF で実現することができる[9]。

HPF 並列 I/O は、ノードごとに独立して行うローカルディスクアクセスを用いて、HPF 処理系が仮想的な並列ファイルを提供する仕組みである。図 15 に HPF 並列 I/O の概念図を示す。配列 A が 2 次元目で分割され、4 ノードに分散配置されているとする。このとき、HPF 並列 I/O を用いて配列 A を書き出すと、それぞれのノードが保持している領域をそれぞれが独立（並列）に 4 つの別々のファイルに書き出す。ファイルには、配列 A のマッピング情報（どのように分散配置されているか）に続いて、データが書き出される。
性能的には、マッピング情報の管理とマッピング情報の出力のオーバヘッドが余分にかかるが、I/O のサイズが MB 単位以上であれば、これらオーバヘッドは無視できる。

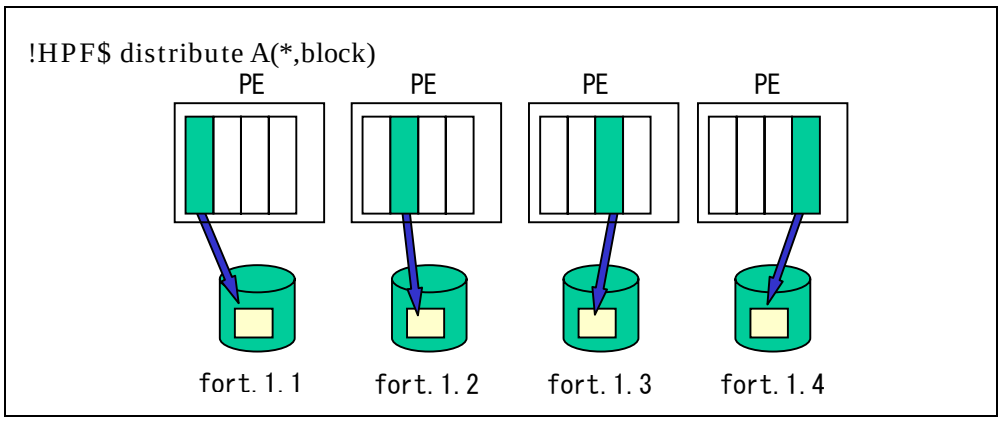


図 15　HPF 並列 I/O の動作例

ただし、ここで生成されたファイルは HPF からしかアクセスできない。また、ファイル生成時のプロセッサ（ノード）構成と読み込み時の構成が異なると、読み出しができないという問題がある。このため、分割数変更ツールが用意されている。本ツールを用いることにより、たとえば 4 ノード実行で生成された HPF 並列ファイルを 8 ノード実行用に変換し、8 ノードで実行中の HPF 並列プログラムから読み出すことが可能になる。また、逐次フォーマット（通常の Unix ファイル）に変換することも可能である。

HPF を用いた並列化評価は、まだ実施途中であるが、APR ベンチマーク、有限要素法プログラムの評価結果を図 16 に示す。左は APR ベンチマークのうち、shallow、scalgam、grid の 3 本について、1 ノード 2 プロセッサ構成で 2 ノードから 512 ノード（1024 プロセッサ）までの加速率を示している（横軸がプロセッサ数で縦軸が加速率）。問題規模が不十分のため、1024PE で性能劣化が見られるが、概ね良好な結果と言える。

図 16 右のグラフは、東大奥田研で開発された有限要素法カーネル[7]の評価結果である。系列 1 は HALO 指定あり、系列 2 は HALO 指定なしの場合である。1 ノード 1 プロセッサを用い、1 ノードから 512 ノードまで測定した。これも問題サイズが不十分で、128 ノ

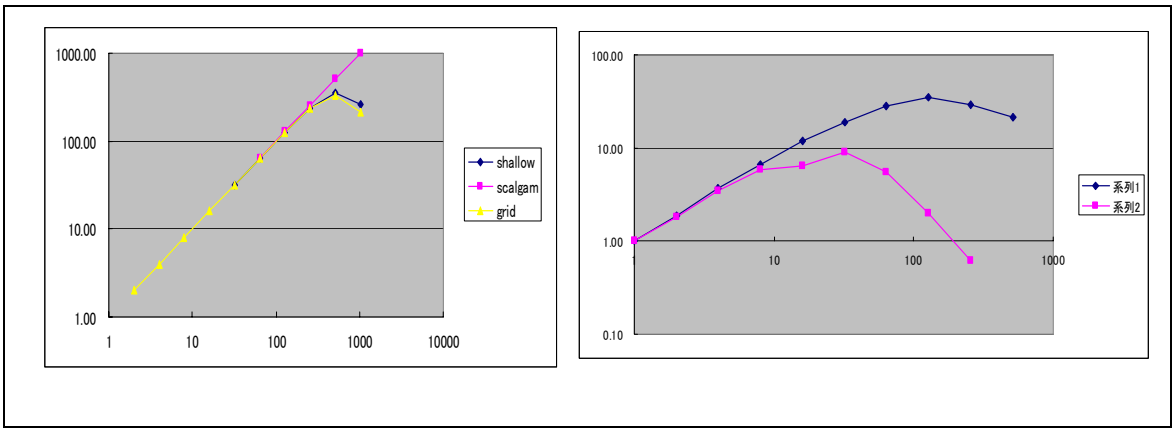


図 16　HPF による並列化評価結果

ードで 38 倍の性能向上に留まっているが、HALO 機能の効果を確認することができた。

この他に、まだ実験途中であるが、姫路工業大学の坂上さんの開発した、impact-3D[10]という TVD 差分コードの大規模構成での評価を行っている。これまでの結果では、160 ノード、1280 プロセッサを用いて 4Tflops の実効性能を達成している。ピーク性能が 10Tflops であるので、40%の効率で HPF プログラムの走行が実現された。

5.7 バッチデバッグツール

地球シミュレータは、巨大な計算パワーを可能な限り効率よく用いるために、バッチ処理が運用の基本となっている。仮に、インタラクティブ処理が可能であったとしても、数千プロセッサを用いる処理を、実行中に止めてデバッグすることは計算パワーの浪費という点で現実的ではない。これら大規模バッチジョブのデバッグのために開発されたツールがバッチデバッグツールである。

バッチデバッグの基本的な考え方は、実行時にログ情報を各ノードで別々に採取し、実行後にこれらログ情報をもとにデバッグを行うというものである。本ツールは、ログ情報の採取を簡便に、しかも効率良いものにするためのログ採取支援機能と、数百以上のファイルとして生成されたログ情報から有用な情報を得ることを支援するログファイルブラウザから構成される。

ログ採取支援機能は、Fortran90 および C/C++をベースに MPI で並列処理を記述したプログラムあるいは HPF から利用でき、ログ採取の指定は、指示行(pragma)形式で記述できるため、コンパイルオプションでログ採取の on/off を切り替えられる。また、トレースバック情報や、手続き遷移の情報、配列データの中身などを簡易な指示文で手軽に出力できる機能、これら出力の実行時の on/off 制御機能（これは膨大になりがちなログ出力の制御に必須）などがサポートされている。

ログファイルブラウザは、less ライクなインタフェースで、プロセッサ（ノード）ごとに生成された（巨大かもしれない）ログを横断的に高速に検索したり、特定の条件でフィルタリングしたりする機能を有している。

6. 一様等方性乱流シミュレーション

地球シミュレータを利用して、ナビエ・ストークス方程式の大規模な直接数値シミュレーション（DNS）を計画している。乱流は、大小さまざまな渦が不規則に入り混じった極めて複雑な流体の運動であり、その解明にはスーパーコンピュータを用いなければ詳細な解析は不可能である。非圧縮性流体の DNS を現在のスーパーコンピュータで行おうとすると格子点数 1024^3 が限界であるが、地球シミュレータを用いればさら大きい格子点数の DNS が可能と考えられる。

我々は、地球シミュレータ開発と同時に、地球シミュレータで実行できる DNS のプログラム Trans7 を開発してきた。ここでは、地球シミュレータ上で Trans7 の実効性能の予

備評価を行った結果を示す[6]。

6.1　並列化コードのテスト結果

Trans7 の並列化では、スペクトル空間のフーリエ係数は k_z 方向、物理空間の変数は y 方向に分割する領域分割法を用いた。この際必要なデータ転置は、3 次元実 FFT において z 方向に 1 次元 FFT を適用する部分と y 方向に 1 次元 FFT を適用する部分の間で行うこととし、MPI ライブラリを用いてそれを実装した。

Trans7 による計算結果の妥当性を検証するために、Trans7 を用いて N=512(格子点数 512^3)の DNS を行った。速度場の初期条件は、エネルギースペクトル $E(k)=a\,k^4\exp\{-b\,k^2\}$ に従う一様等方的なランダムな速度場で、周期境界条件を満足するように与えた。$\Delta t = 0.001$、動粘度 $\nu = 0.000763$ とし、十分乱流場が発達しエンストロフィー(渦度の 2 乗平均の 1/2)が安定する t=10 まで時間発展させた。乱流の統計的準定常状態を得るための外力 f として、低波数領域($|k| < 2.5$)に負の粘性を仮定し、その値は全エネルギーが一定に保たれるように各時間ステップで調整した。t=10 でのテーラー長 λ に基づくレイノルズ数は、$R_\lambda \fallingdotseq 164$ であった。図 17 に t=10 でのエネルギースペクトルを示す。この図から、波数領域 $4 < k < 16$ 近傍において慣性小領域におけるコルモゴロフのエネルギースペクトルが確認された。

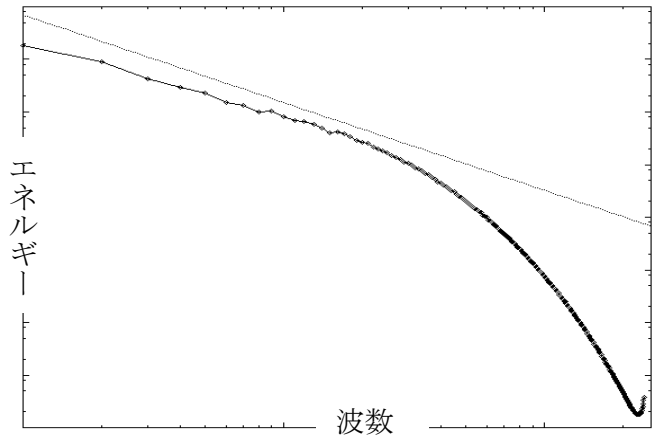


図 17　エネルギースペクトル(格子点数 512^3)

6.2　地球シミュレータでの実効性能

Trans7 における時間発展ループの 1 ステップに対して、1 台の PN 上での実行時間を計測した。逐次版 Trans7 において、格子点数 512^3、8AP の場合の計算時間は約 25 秒であり、ハードウェアモニタによる実効性能で約 27.8Gflop/s が得られた。これはピーク性能 64Gflop/s のほぼ 43%である。

また、MPI により並列化した Trans7 に対し、格子点数(問題サイズ N の 3 乗) を 32^3、

64^3、128^3 および 256^3、PN 内のプロセッサ数を 1、2、4、8 と変化させた時の実行時間を計測した。格子点数 256^3、1AP の時の実効性能は約 3.72Gflop/s であり、ピーク性能の約 47% が得られた。地球シミュレータのベクトルプロセッサのベクトルレジスタ長は固定の 256 要素(8 バイト長)、倍精度演算性能とデータ供給能力の比は 2:1 となっており、2 基底 FFT をベースとする Trans7 では、FFT のカーネルループの演算数とロード/ストア回数はほぼ 1 対 1 であることを考慮すれば、ピーク性能の約半分の性能が得られており、十分な性能である考えられる。8AP では約 25Gflop/s を達成しているが、逐次プログラムの性能と比較すると若干低下している。これは並列化に伴うオーバーヘッドと考えている。

図 18 に、各問題サイズに対し、逐次実行時間に対する速度向上率を示す。N=32 では、ベクトル処理の平均ループ長が 32 と短いため速度向上率が十分ではないが、それ以外は 8AP で 7 倍近い速度向上が得られ、ノード内並列処理においてほぼスケーラビリティを達成していると言える。

また、格子点数 128^3、256^3、および 512^3 の DNS に対して、地球シミュレータの PN を 1 台、2 台、4 台、および 8 台を用い、各 PN 内のプロセッサを一つとした時の時間発展ループ 1 ステップの実行時間を計測した。この並列実行では、PN 内はマイクロタスクによる並列実行を用いた。

ノード間並列実行の速度向上率を図 19 に示す。速度向上率の計算では、単体 PN での逐次版の実行時間を基準にした。PN 内並列処理の速度向上率より若干低い値が得られたが、8 ノードで 7 倍近い値が得られた。PN 間データ転送については、グローバルメモリを利用した実装ではなく、さらチューニングが必要と考えられる。

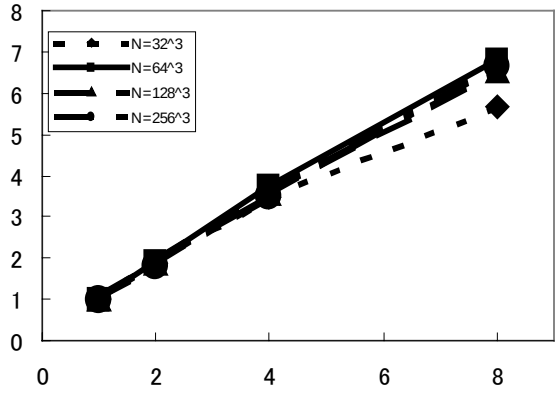


図 18　ノード内の速度向上率

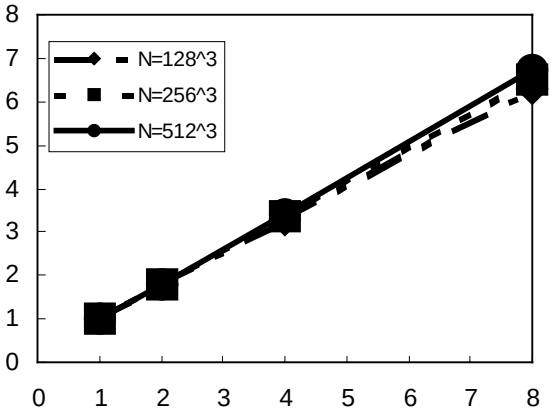


図 19　ノード間の速度向上率

7. おわりに

地球シミュレータ計画は、地球変動研究を飛躍的に進展させるために、所謂ニーズ先行で、計算機ハードウェアの開発が開始された。5 年間の開発により地球シミュレータは目標以上の実効性能を確認することが出来たが、まだ稼動して間もないシステムであり、本格的な評価や､実運用によりプロジェクトの真価が問われるのはまさしくこれからである。地球シミュレータはという 5120 プロセッサのとてつもない巨大システムは、フルに使いこなすのも大変だが、いともたやすくテラフロップスオーダの性能が出るというモンスターマシンである。大規模な数値シミュレーションによりどのような結果が得られるか楽しみである。また、日本はソフトウェア開発能力が低いと云われる中で、世界一のハードウェアを持ったことがソフトウェア開発に対しどのように影響するかについても興味深い。

最後になってしまったが、完成を間近にひかえた平成 13 年 11 月 17 日、地球シミュレータの構想から開発までを先導してきた三好甫氏が亡くなられた。地球シミュレータ完成を見ずに他界されたことはさぞご無念であったことと思われる。我々は氏の遺志を引き継ぎ、世界に冠たる日本の HPC 技術を受け継いでいくことが重要であると思う。今後、「地球シミュレーション」により地球科学の発展に寄与することを願ってやまない。

参考文献

[1] 谷啓二，地球シミュレータ計画のバックグランドとその位置付け，RIST News (2000).

[2] 横川三津夫，谷啓二，地球シミュレータ計画，情報処理，Vol.41，No.4，pp.369-374 (2000).

[3] M. Yokokawa, et al., Performance Estimation of the Earth Simulator, Proceedings of the ECMWF Workshop, November (1998).

[4] M. Yokokawa, et al., Basic Design of the Earth Simulator, High Performance Computing, LNCS 1615, Springer, pp.269-280 (1999).

[5] 上原, 田村, 横川, 地球シミュレータの計算ノード上での MPI 性能評価, HPCS2002, 73-80 (2002).

[6] 横川，斉藤，石原，金田，地球シミュレータ上の一様等方性乱流シミュレーション，HPCS2002，125-131 (2002).

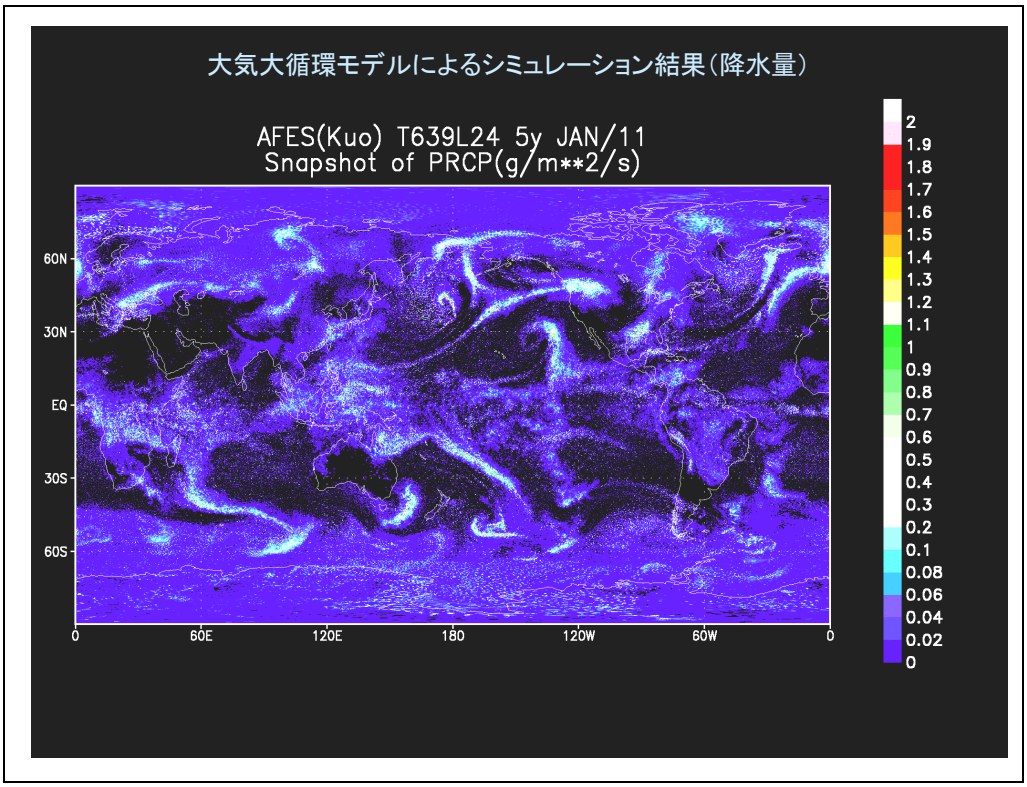
[7] Hiroshi Okuda and Norihisa Anan, Optimization of Element-by-Element FEM in HPF 1.1, Concurrency and Computation: Practice and Experience, Wiley & Sons Ltd., (to appear in Spring 2002)

[8] Japan Association of High Performance Fortran, 1999, HPF/JA Language Specification Version 1.0, RIST (English version is available at http://www.tokyo.rist.or.jp/jahpf/index-e.html).

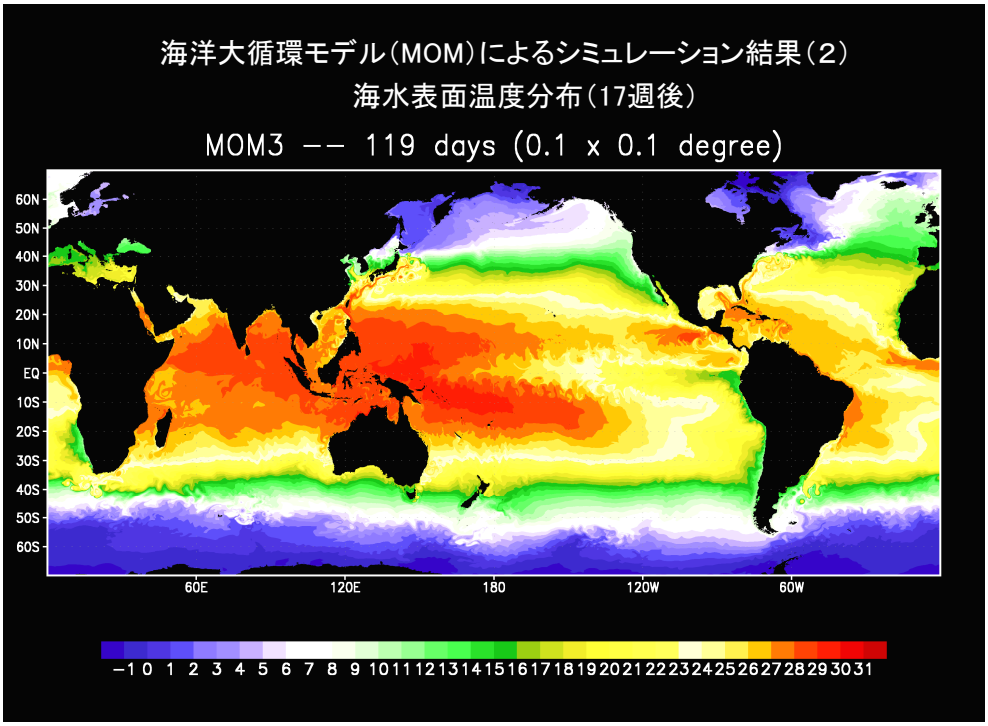
[9] Siegfried Benkner, Erwin Laure, and Hans Zima. 1999. HPF+: An Extension of HPF for Advanced Industrial Applications. Technical Report TR 99-1, Institute for Software Technology and Parallel Systems, University of Vienna.

[10] H. Sakagami and T. Mizuno, Compatibility Comparison and Performance Evaluation for Japanese HPF Compilers using Scientific Applications, Concurrency and Computation: Practice and Experience, Wiley & Sons Ltd., (to appear in Spring 2002).

<付録 1>　実機を用いた気象シミュレーション例（大気大循環モデル）



<付録 2>　海洋大循環モデルの計算例



3.4.3 高速ネットワーク環境下における高度医療アプリケーション

佐藤　裕幸　委員

1. はじめに

がん治療法の一つである粒子線治療は、ヘリウム、炭素、ネオンなどの重粒子線を患部に集中して照射し、周辺の正常細胞への影響を最小限に抑え、患者の早期社会復帰を可能とする非常に有効ながん治療方法である[1]。この粒子線を用いたがん照射治療を普及させるためには、遠隔地の複数の医療機関から高速なネットワークを介して、重粒子線がん照射施設の計算サーバにより、照射施設のノウハウを利用して患者毎に最適な照射方法を得る治療計画を立案できることが望ましい。

このような高速なネットワークの利用法はまだ実現例に乏しく、基本的な性質の解明も進んでいない。インターネットの普及が進み、ネットワークを利用する研究者にも利便性を提供しているが、一方ではネットワークの輻輳が発生して、研究環境が悪化している例がある。またネットワークに不正に進入する事例が多数発生しており、医療関係の情報のようにプライバシーを守るべき対象を安全に保護する技術が必要不可欠となっている。

このような状況から、インターネットにおける信頼性・高速性・利便性・安全性に関する種々の問題点を解決するために、これらについて極めて高度な要求を持つ遠隔地重粒子線がん照射影響シミュレータを業務として想定し、ギガビットレベルネットワークに向けた信頼性・高速性・利便性・安全性を確立することを目指し、高度医療ネットワークに関する研究が行われている。

2. 粒子線がん治療

2.1 粒子線とは

粒子線は、放射線の一種である。放射線は、従来からよく利用されている X 線やガンマ線等の光子線と粒子線に分けることができ、更に粒子線は、陽子線とヘリウム、炭素、ネオン等の重粒子線に分けることができる。これらの各放射線を患者体内へ照射した場合の線量の変化を図 1 に示す。図から分かるように、ガンマ線より陽子線、陽子線より炭素の方が患者のからだの表面からの深さに従って線量が鋭く変化している。すなわち、炭素のような重粒子の方がより正常細胞を傷つけずに照射することができ、これにより患者の早期社会復帰を可能にしている。

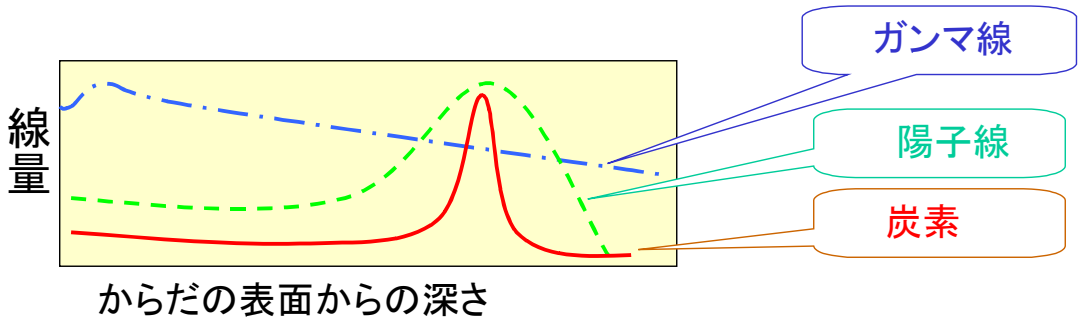


図１　各放射線の線量

2.2 粒子線がん治療装置

このような重粒子線を用いたがん治療装置の１つを図２に示す。図２は、千葉市稲毛区にある放射線医学総合研究所の HIMIC（Heavy Ion Medical Accelerator in Chiba）であり、125m×65m の巨大な装置（施設）である。本装置は、ビームを発生させる照射線源、ビームを加速する加速器及びビームを照射する３つの治療室から構成される。治療室が３つあるのは、ある治療室で照射中に別の治療室で照射位置合わせ等の準備が平行して行えるようにするためである。

このように重粒子線がん治療装置は巨大であるため、各病院に設置するということは不可能である。現在、国内外の治療施設には、以下のものがある。

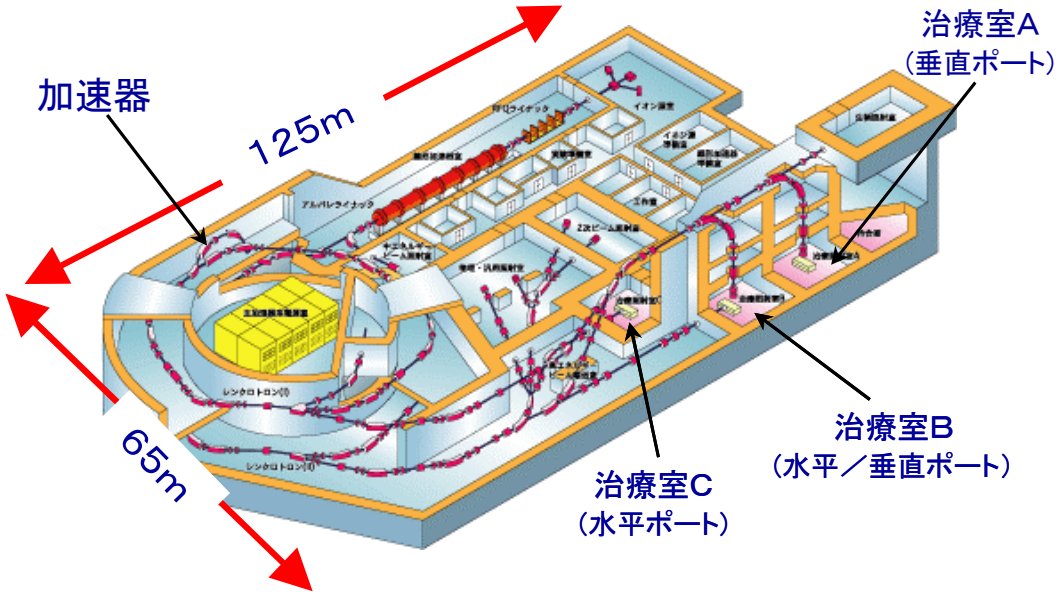


図２　重粒子線がん治療装置（HIMAC）

- 国内
  - 筑波大学（陽子線）
  - 国立がんセンター東病院（陽子線）
  - 兵庫県立粒子線医療センター（陽子線、炭素）
  - 静岡県がんセンター（陽子線）
  - 今後、各県に１施設ずつ設置予定
- 海外（稼動中のもの）
  - 米国：４、カナダ：１、ロシア：３、スウェーデン：１、スイス：１、ベルギー：１、フランス：２、英国：１、南アフリカ：１

3. 高速ネットワーク環境下における高度医療アプリケーションの研究開発

以上のように、各病院に粒子線がん治療装置を設置するのは不可能であり、また、放射線治療を行うには、前もってシミュレーションを行って最適な治療の計画を立てる必要がある。そこで、がん患者が入院している病院とがん治療施設との間を高速ネットワークで接続し、病院から CT 画像や患者情報を治療施設へ転送し、治療施設でその情報を基に照射シミュレーションを実施して、病院へその結果を送り返す、といった治療の遠隔シミュレーションが行えると良い。このような高速ネットワークを用いた遠隔シミュレーションがスムーズに行えることを目的に、平成 10 年度科学技術振興調整費により、研究が行われている。最初の３年間が第 I 期であり、あとの２年間が第 II 期である。

本研究は、以下に示す３つの研究項目に分かれており、それぞれ図３に示すような関係となっている。

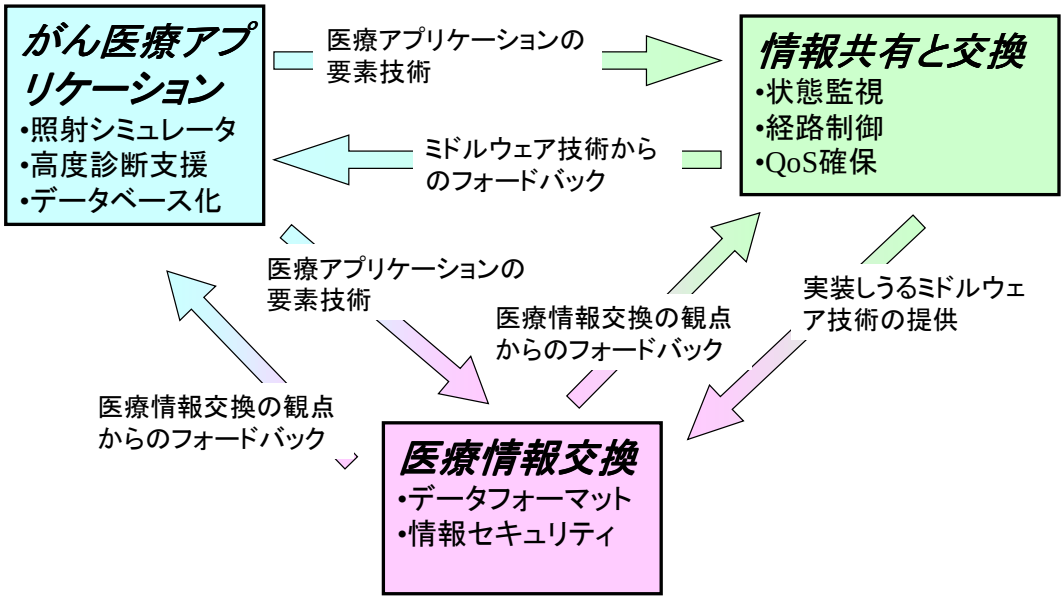


図３　研究項目

3.1 がん医療アプリケーションに関する研究

がん医療における画像情報を主体とした大量の情報の信頼性、安全性を保ちながらネットワーク上で高速かつ効率よく伝達し応用することを目的として、遠隔地がん照射シミュレーション、がん画像伝送、三次元画像構築及びコンピュータ支援がん画像自動診断装置に関する開発、研究を行ってきている。

(1) 遠隔地がん照射影響シミュレータの研究

重粒子線がん治療を普及させるためには、がん患者が入院している多くの医療機関において、治療計画を立案できるようにすることが必要である。そのため、放射線医学総合研究所に設置されている重粒子治療計画装置を高速ネットワークを介して利用することを試みる。第 I 期においては、本装置をネットワークにより利用できるように改造し、インターネットを通じて国立がんセンター東病院から操作する予備実験を行った。第 II 期においては､国立がんセンター東病院と放射線医学総合研究所を高速ネットワークにより接続し、本装置の本格運用を試みている。

(2) がんの高度診断支援技術に関する研究

がんの広がりを正確に診断するため、CT、MRI 等の形態画像と Functional MRI、SPECT などの機能画像との合成画像を構築する技術の開発とデジタル画像情報を用いて肺がん、乳がん、脳腫瘍に対する三次元画像表示とコンピュータ支援自動診断装置の開発を行っている。また、これらに加えて CT、MRI 画像と PET 画像の合成を行うとともに三次元画像表示及びコンピュータ支援自動診断装置の精度向上を計り、実際の医療現場においてネットワークを用いた実用を行っている。

(3) 画像を含む医療データのデータベース化に関する研究

遠隔地がん照射治療を実施するためには、担当医師が画像を含む様々な種類の医療データを分散した医療拠点から収集する必要がある。そのため、インターネット上に分散して存在する画像データとファクトデータを効率よく利用可能とすることを目的として、インターネット上に分散した画像を含むデータを高速に収集しディレクトリを構築する技術を研究し、さらには特定の臓器のがんの画像を収集するために領域分割やテクスチャに注目した画像の特徴抽出と患者名や医師の所見等のテキスト情報を組み合せた画像検索技術を研究するとともに、多様なフォーマットから成るデータから数値を中心としたファクトデータを抽出することにより、時系列等の要素でバーチャルに統合化されたデータベースを構築する技術を研究している。

3.2 情報共有と交換に関する研究

インターネット上での情報の共有と交換に関する研究を行っている。

(1) ギガビットレベルネットワークに向けた QoS 確保技術の研究

ギガビットレベルネットワークにおいては、利用者の要求する通信品質を確保し、必要な帯域を提供する技術が必要となる。そのため、通信品質を保証するためのトラフィック

モデルを構築すると共に、既存のリソース予約プロトコルに比して低負荷かつ高速な計量 Qos 確保プロトコルを開発し、これらを基に制御管理サーバの開発及び経路制御技術と密接な連係を前提とした QoS 確保技術を研究開発している。さらには、端末側における QoS 要求自動調整技術の研究も行っている。さらには、ネットワークとアプリケーションのより効果的な連携を実現するため、ネットワークの提供する通信品質に基づいた、あるいはアプリケーションレベルの QoS 要求を考慮した QoS 確保技術の研究も行っている。

(2) ギガビットレベルネットワークに向けた経路制御技術の研究

ギガビットレベルネットワークで信頼性のある経路制御を実現するためには、動的な経路制御が性能に大きく影響する点、そして、分散された経路情報の矛盾が大きな無駄なトラフィックを生む点が従来の研究の結果として確認されている。そこで、本研究では、分散されている経路制御情報を少ない矛盾で共有するための経路制御アーキテクチャの開発と実証実験を行っている。

(3) ギガビットレベルネットワークに向けた状態監視技術の研究

ギガビットレベルネットワークにおいては、QoS 確保技術や経路制御技術が円滑に運用されるように、ネットワークの状態を短期的および長期的にかつ正確に把握することが必要である。そのため、多地点における監視手法と伝送路に依存しない監視手法の確立を目的として、実地のネットワークにおける検証を行っている。本研究においては、医療アプリケーションにおける情報の共有と交換が順調に行われることに重点を置く。更に、なるべく少量の測定データを基に、できるだけ短時間に分析ができるように種々の技法を検討し、適用している。

3.3 医療情報交換に関する研究

超高速ネットワークを用いて実現される医療モデルを作成し、その中での最適化されたデータ交換形式、交換用アプリケーションとデータベースの設計に関する研究を行い、合せて医療モデルとネットワーク環境の最適化法に関する研究（Network Oriented Healthcare Modeling）を行っている。また、情報セキュリティに必要な要件情報の利用制御やフロー制御を円滑に行うことを目的として、ポリシーベース情報管理システム、公開鍵暗号方式または鍵事前配布方式による照明・認証方式の確立及び耐タンパーソフトウェア構成技術に関する研究を行っている。

(1) 医療情報・データのフォーマット等に関する調査研究

放射線治療上考えられる医療の過程として放射線治療の適応に関する検討、放射線治療の臨床試験に関する適応の検討、放射線治療の方法の選択、放射線治療計画の作成、インフォームドコンセント、放射線治療期間中の診療、放射線治療急性障害の有無の確認、放射線照射終了記録、放射線治療後の治療評価、放射線晩期障害の有無、放射線治療終了後のフォーローアップ診療が考えられる。この各診療の過程に必要な診療の内容を整理し、診療に必要な情報項目とそのデータ形式、データ量、必要な表示方法について調査し、各

診療における人（医師、患者、看護婦、技師）の権限関係や情報の交換についての医療モデルを作成している。高度医療ネットワークにおいては、利用可能なネットワーク能力、診察にかかるコストなどを勘案して医療モデルとネットワーク環境の最適化を図る必要がある。そこで、医療モデルとネットワーク環境を最適化するための評価尺度および分析方法に関する調査研究を実施している。

(2) 医療情報交換に係わる情報セキュリティに関する調査研究

遠隔地がん照射治療の普及のためには、患者の症例情報がインターネット上を流通することが必要であり、こうした特定の用途や目的のみに利用する情報の安全性を確立することが急務となっている。そのため、情報の利用制御やフロー制御を円滑に行うことを目的として、柔軟で強力な利用条件記述方式の研究、情報本体に付加された利用条件から提示されるアクセス権限を合致する CBR（Capability-Based Resource Management）と呼ばれる方式により安全で簡易な利用資格確認方式の研究を行うとともに、公開鍵暗号方式または鍵事前配布方式をベースとした軟らかい証明・認証方式の確立、情報利用方法や利用履歴を不当な改竄や解析から防止する耐タンパーソフトウェア構成技術を研究している。

4. 重粒子がん照射影響シミュレータ

治療計画において重粒子がん照射の影響をシミュレートするためには、患者体内の線量分布を正確に計算しなければならない。しかし、従来の高精度な三次元の線量分布計算は処理時間がかかるため、治療計画の効率が悪いという問題点があった。そのため、治療計画の高速化の研究が幾つか行われてきた[2][3][4]。これらの研究では、トランスピュータや CRY-YMP のような従来型の並列計算機や PentiumPro による特殊な自製の並列計算システムを用いており、コストパーフォーマンスや拡張性の面で問題がある。

近年、パーソナルコンピュータ（PC）の低価格化とギガビット・イーサネットを代表とする汎用ネットワークの高速化により、PC やワークステーション（WS）を構成要素とする並列処理クラスタが、安価で拡張性の高い計算サーバとして注目されている[5]。そこで、計算サーバの処理能力及び信頼性などを向上させるために、特に浮動小数点演算性能に優れた Alpha WS を高速な汎用ネットワークで接続したクラスタを用いた並列処理により、治療計画処理の高速化を行っている。

この並列化においては、CT 画像などの大量データを扱うので，分散メモリ環境でデータ通信時間をいかにして抑えるかが大きな課題となる。また、領域分割により並列化を行うが、各領域の計算負荷の予測が困難なため、プロセッサ間の負荷の均等化も課題の一つとなる。我々は、コストパフォーマンスと拡張性に優れた WS クラスタをプラットフォームとし、対象問題の性質及び高速化を望んでいる部分や計算精度等のユーザの要求を考慮して、並列処理方式を検討している。

4.1 線量分布計算

線量分布は、患部周辺の平行横断面である CT 画像を複数枚（～100 枚）取得することで得られる三次元情報（図４）を基に計算する。1 枚の CT 画像は 512×512 ピクセルであり、1 ピクセル当たり 4 バイトで表現しているので、その三次元データ量は 0.5K ピクセル×0.5K ピクセル×100 枚×4 バイト=100M バイトと大量となる。線量分布計算では、この三次元化された CT 画像データや各種照射に関するデータを基にして各点の吸収線量を計算する。そして、計算結果は CT 画像と重ね合わせて表示するため、各線量を CT 画像データと同じ形式（座標系）の三次元データとして格納する。したがって、計算結果も 100M バイトのデータ量となる。このように、線量分布計算では、大量データを扱うので、並列化の際にデータの通信時間をいかにして抑えるかが課題となる。

線量分布の計算は、以下のような手順で行う。

- 照射される面を格子状に分割し、ビームを各格子への線（Ray）に分割する。図５では、3×3 の 9 つの Ray に分割している。ここで、コリメータ（Collimator）とは、ビームの横方向の広がりを調整するためのカメラの「絞り」のようなものである。
- 各 Ray 毎に Ray 上に沿って一定間隔に計算点を決める（図５の星印）。
- 各計算点毎に，数式(1)～(3)により線量を算出する。計算結果を CT 画像と同じ座標系の形式に格納する際に、計算点座標と CT 画像座標が正確に一致しないため、その距離に応じて計算結果を線形補間して格納する。

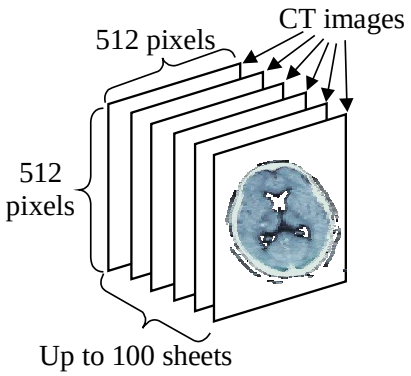


図４　三次元 CT 画像

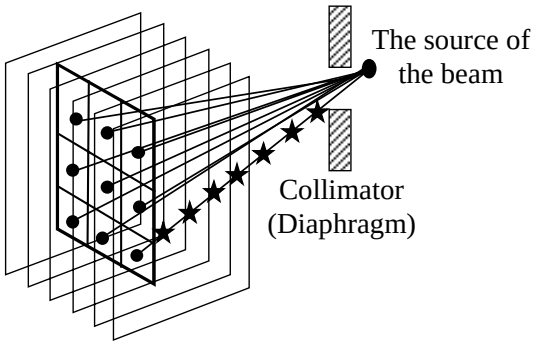


図５　線量分布計算

$$D_p = DD_p \times OAR_p \qquad (1)$$

計算点 p での線量値 D_p は、ビームの中心軸上の線量 DD_p と中心軸外の線量比率である OAR_p から算出される。

$$DD_p = BST(WEL_p) \times \left(\frac{SSD + WEL_p}{L_p} \right)^2 \qquad (2)$$

別途水中で実際に計測した線量値がテーブル BST に格納されており、計算点 p のビームの中心軸上の線量 DD_p は、線源と計算点 p との距離を水等価厚 WEL_p に変換して、水等価厚でテーブル BST を引くことで得ることができる。なお、SSD は線源から計測開始点までの距離で、L_p は線源から計算点 p までの距離であり、式(2)の第 2 項はビームが逆二乗則で広がることによる粒子数の変化を補正するためのものである。

$$WEL_p = \sum_{i=1}^{p} (ED_i \times Step) \qquad (3)$$

計算点 p までの水等価厚 WEL_p は、各計算点 i での CT 値から算出される電子密度 ED_i と計算ステップ幅 $Step$ の積分により算出される。計算点 p での中心軸外の線量比率 OAR_p は、線源のビームサイズと計算点 p でのビームの進行方向に対して横方向の座標を基に算出されるが、その詳細は文献[6]に記載されている。なお放射線としては、陽子及び炭素を対象としている。この方式による線量分布計算は、通常の WS を用いて数十秒程度かかり、更にペンシルビーム法[6]と呼ばれるより正確な結果が得られる方式では更に時間がかかる。

計算点毎の処理は、式(3)により水等価厚の算出で積分しているので、同一 Ray 上の前計算点の（中間）結果が利用可能であり、逐次的になっている。一方、 Ray 毎の計算は独立しているので Ray 束の単位で並列に処理できる。つまり、計算領域（Ray）を複数の部分領域に分割し、それぞれの分割された領域を各プロセッサで独立に計算し、計算結果を最後にマージするという並列化方法である。各部分領域（Ray）は、その位置により CT 画像を通過する距離が異なり、それにより必要な計算時間も異なるので、部分領域によって負荷が異なる。したがって、並列化においては、各プロセッサ間で負荷をいかにして均等化するかということも課題となる。

また、治療計画においては、照射線源から照射体を見て撮影したような画像を CT 画像を基に再構成する DRR（Digitally Reconstructed Radiograph：ディジタル再構成 X 線撮影）画像の生成も処理時間がかかるので、線量分布計算と同様の方法で並列化した。

4.2 並列処理システム

治療計画の計算処理システムは、10 台の WS を汎用のネットワークで接続した、いわゆる WS クラスタである（図６）。各 WS の CPU は Alpha21164A 600MHz であり、10 台の内 1 台がサーバとして使用される。サーバには 1GB のメモリが搭載され、それ以外

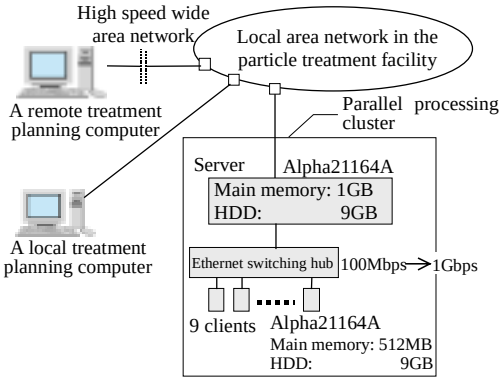


図６　システム構成

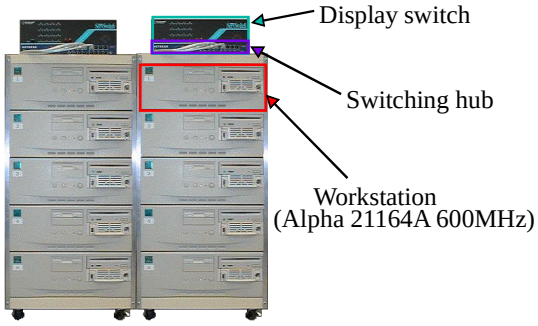


図７　並列処理クラスタ

のマシンには 512MB 搭載されている。各マシンは、100Mbps のイーサネット・スイッチで接続されており、まもなく 1 Gbps のイーサネットに置き換える予定である。このクラスタが遠隔地の医療機関からギガビットレベルの高速なネットワークを介して利用されることを想定している。

図 7 に並列処理クラスタの実物写真を示す。このクラスタは、２セットのデスクトップ型の WS を 5 台積み重ねたものから成り、1 台数十万円の WS とネットワーク機器などで、総計数百万円という安価かつ高性能で拡張性のある並列処理環境となっている。また、オペレーティングシステムは Linux を用い、プロセッサ（WS）間の通信ライブラリは、多くの種類のマシン上で動作し広く利用されている PVM (Parallel Virtual Machine)[7]を用いている。

並列処理を支援するツール（ミドルウェア）として，我々の開発した分散型並列処理支援ツール ParaJET[8]を使用している。ParaJET は、指定されたジョブ群を各プロセッサの負荷状況に応じて分配する負荷分散ツールである。特に、並列動作するジョブ間で全く通信がなく独立して実行できる場合は、従来の逐次プログラムを変更することなく並列処理できるという特長を持っている。また、各プロセッサの負荷状況と実行状態（実行中及び実行を完了したジョブ群）を表示するモニター機能を持っており、粒度調整等の並列処理のチューニングが容易に行えるようになっている。

ParaJET の負荷分散方式は、いわゆる要求駆動型の動的負荷分散である。Server マシンのジョブプールにジョブ群を貯めておき、実際にジョブを実行する各 Client マシンは実行するジョブがなくなったら（又はなくなりそうになったら）、Server マシンにジョブの要求を行い、Server マシンはその要求によりジョブプールに貯めているジョブを 1 つ要求元の Client マシンに与える。なお、Server マシンが Client マシンを兼ね、Server マシンは負荷分散処理を行っていない間は自らもジョブの実行を行っても良い。

4.3 並列処理方式

線量分布計算は、以下のように処理する（図 8）。

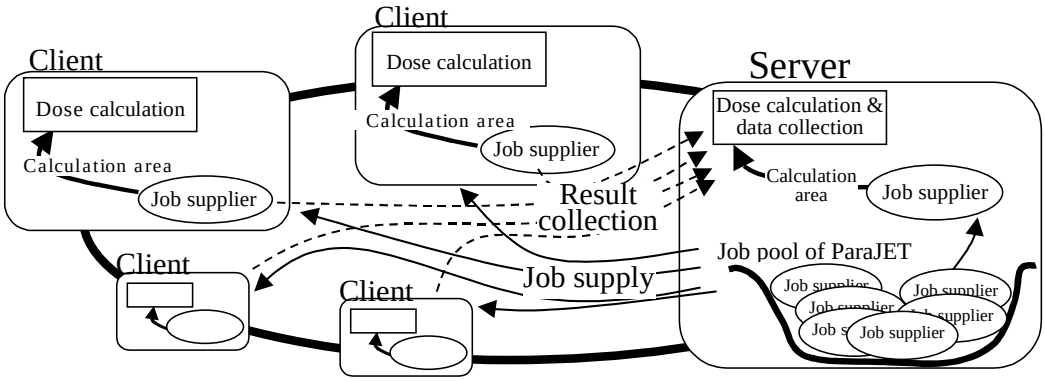


図８　並列処理方式

- 予め ParaJET 及び PVM デーモンを起動しておく。
- 本システムの起動プログラムが、サーバ及びクライアント上に線量分布計算デーモン（Dose calculation）を起動し、ParaJET のジョブプール（Job pool of ParaJET）に、計算領域の分割数分の Job Supplier を格納する。なお、計算領域の分割数はプロセッサ数の定数倍とする。
- 線量分布計算デーモンは、計算領域（計算開始の Ray 番号と担当する Ray の本数）を PVM 通信により渡されると、その領域の線量分布計算を行うプログラムである。ただし、サーバでは、このプログラムはデータ収集プログラム（data collection）を兼ねている。つまり、線量分布計算の合間に定期的にクライアントから結果データが送られてきたかをポーリングして、データ収集を行う。
- 各 Job Supplier は、担当する計算領域（Ray の開始番号と担当する Ray の本数）をそれぞれ保持しており、それらを自プロセッサ上の線量分布計算デーモンに渡すプログラムである。このプログラムは、線量分布計算デーモンに担当する計算領域を通信ライブラリ PVM により通信して渡し、線量分布計算が終了したら PVM 通信によりその通知を受け終了する。したがって、線量分布計算を行っている間（実際には処理を全く行わないが）生きているプログラムである。このプログラムを ParaJET のジョブプールに格納することで， ParaJET によってアイドル状態の（仕事のない又は終了した）プロセッサ上に割り付けられる（Job supply）。そして、計算領域（Calculation area）がそのプロセッサ上の線量分布計算デーモンに渡される。
- 各プロセッサ上の線量分布計算デーモンは、計算領域を受け取るとその領域の線量分布計算を行い、計算結果をサーバへ送信する（Result collection）。その際、計算した領域のみデータ転送し、線量値も 4 バイトから 2 バイトに圧縮して転送する。
- サーバ上のデータ収集プログラムは、分割数分の計算結果を受け取ると、全クライアントの線量分布計算デーモンに終了を通知すると共に、結果をディスクに書き込んで終了する。

4.4 評価実験

4.4.1 計算モデル

この実験で用いた線量分布計算の計算モデルは、256mm×256mm×200mm の水で満たされた直方体に直径 40mm の球（球の媒質も水）があり、線源側に直径 40mm のコリメータと球の表面に均一線量を照射するために半径 20mm の半球状にくり貫かれた補償体（放射線吸収体）がある（図９）。計算の基となる CT データ及び線量分布結果は、

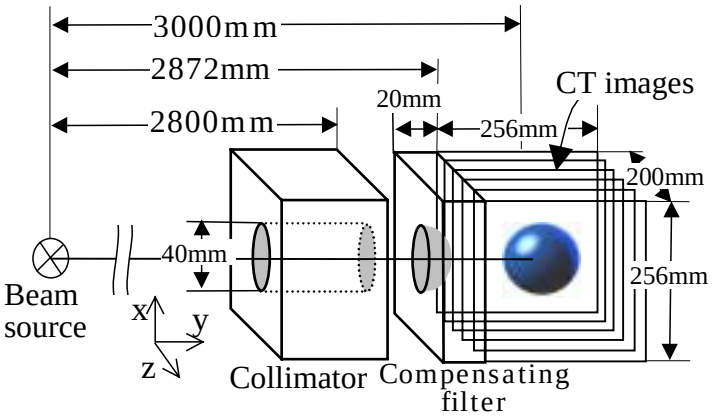


図９　計算モデル

512×512×100 ピクセルであり、1 ピクセル 4 バイトで表現しているので、そのデータ・サイズは 100M バイトとなる。このモデルでの線量分布の計算結果の二次元表現は、図 10 のようになる(実際にはカラー表示)。治療計画においては、このような等線量分布表示は、患部の CT 画像と重ね合せて表示される。

DRR 計算の計算モデルは、線量分布計算の計算モデルとほとんど同じであるが、コリメータや補償体は DRR 計算にとって不要なので削除してあり、また、照射方向により計算結果が変わるようにターゲットを半円柱（かまぼこ状）にしている。このモデルでの

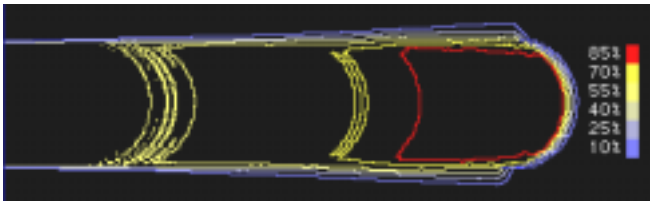


図 10　線量分布計算の計算結果

図 11　DRR 計算の計算結果

DRR の計算結果は図 11 のようになり、上部が薄く下部が厚いかまぼこ状になっているのが分かる。この DRR 計算の結果は 512×512 の二次元データなので、そのサイズは 1M バイトと線量分布計算に比べて小さい。

4.4.2 実行時間と速度向上率

逐次実行と 10 台で並列実行したそれぞれの実行時間とその内訳を表 1 に示す。なお、これらの実行における計算領域の分割数は、線量計算が 20 で、DRR 計算が 40 である。ここで、「入力」とは CT 画像をファイルから読込む時間であり、 「出力」とは計算結果をファイルに書出す時間である。また、「初期化」とは照射領域のサイズを算出する初期化処理であり、「通信」は通信時間, 、「計算」は計算時間である。なお、この通信時間は、計算結果をクライアントからサーバへ送信する時間だけであり、その他の並列制御のための通信時間は個々に計測できないため、「計算」の方に含まれている。これらの時間の中で、「入力」と「初期化」は全プロセッサで実行され、「出力」はサーバでのみ実行され、「通信」と「計算」のみが並列に実行される。そのため、「通信」と「計算」以外は、逐次実行と並列実行で同じ時間になっている。

DRR 計算の実行時間が線量分布計算よりかなり長くなっているのは、DRR 計算では CT 画像の全領域を計算対象としているのに対して、線量分布計算では照射領域のみを計算対象としているからである。

線量分布計算では、逐次で 23.76 秒かかっていた処理が 10 台並列実行で 7.11 秒になっているが、これは全実行時間であり、「計算」だけを見ると 2 秒である。したがって、通信時間を除いた速度向上率（並列実行による効果）は 9.63 倍になり、これは使用プロセッサ台数にほぼ等しいと言ってよい。DRR 計算では、逐次で 52.40 秒かかっていた処理が 6.85 秒になっており、速度向上率は 9.42 倍である。これもほぼ使用台数に等しくなっている。したがって、動的負荷分散によるオーバヘッドはほとんどないと言ってよい。また、サーバが各クライアントに最初に計算領域を割り付ける部分が逐次化されているが、計測

表１　実行時間（秒）とその内訳

		線量計算			DRR生成		
		逐次	10並列	並列効果	逐次	10並列	並列効果
(1)	入力	1.20	1.20		1.20	1.20	
(2)	出力	1.30	1.30		0.04	0.04	
(3)	初期化	1.90	1.90		0.00	0.00	
(4)	通信	0.00	0.70		0.00	0.18	
(5)	計算	19.36	2.01	9.63	51.16	5.43	9.42
(6)	合計	23.76	7.11	3.34	52.40	6.85	7.65
(7)	(4)+(5)	19.36	2.71	7.14	51.16	5.61	9.12

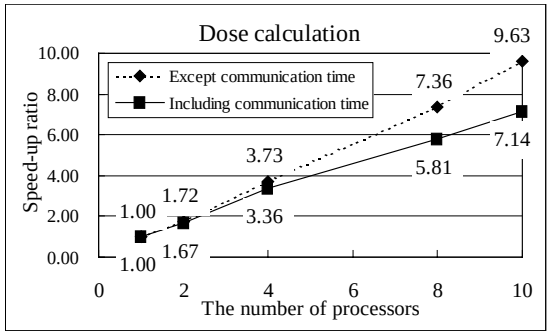


図 12　線量分布計算の速度向上率

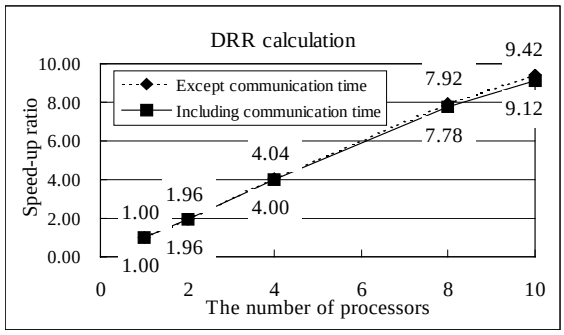


図 13　DRR 計算の速度向上率

結果からは少なくとも 10 台程度までであれば、そのオーバヘッドはほとんどないと言える。

実際の医療現場では，線源や照射機器情報を調整しながら何度か計算を繰り返す利用形態が多い。つまり、初期化以降（「計算」及び「通信」）が繰り返されると考えてよく、この部分の高速化が重要となる。図 12 及び図 13 にこの部分の使用プロセッサ台数毎の並列実行効果（逐次実行を基準にした速度向上率）を示す。両者とも、台数が増えるにしたがって線形に高速化されており、スケーラブルであると言える。すなわち、プロセッサ台数を増やせば更に高速化できる。DRR 計算では、通信時間を含めた 10 台実行での速度向上率は 9.12 倍でありかなり良く、線量分布計算では 7.14 倍で DRR 計算ほど良くない。これは、DRR 計算の通信時間が計算時間の 3%であるのに対し、線量分布計算では 35%もあり、線量分布計算の結果のデータ量が DRR 計算に比べてかなり多いため、通信時間が大きくなっているのが原因であると思われる。

いずれにせよ、実際の医療現場で何度も繰り返し行われている処理が、これまで線量分布計算で約 20 秒、DRR 計算で約 50 秒かかっていたものが、3 秒～6 秒と端末の前で待っていられる時間まで短縮されたことは、非常に有意義であると言える。

5. 遠隔操作実験

およそ 30km 離れた国立がんセンター東病院（がんセンター：千葉県柏市）と放射線医学研究所（放医研：千葉市稲毛区）の間で放射線治療計画を模擬した通信試験を行った。接続形態は、放医研に線量分布計算を行う並列クラスタを置き、約 30Km 離れたがんセンターに治療計画装置を置き、この間を商用のインターネット（通信仕様は 1.5Mbps）で接続する。治療計画装置はファイヤウォールの内側であるがんセンター内の LAN に設置し、並列クラスタは放医研のファイヤウォールの外側に設置する（図 14）。治療計画装置は、外部へはアクセスできるが、外部からのアクセスはできない。一方、並列クラスタはファイヤウォールの外側に設置されているので、外部からのアクセスも外部へのアクセスも可能である。なお、がんセンター側のファイヤウォールの関係で、通信に用いることのできるプロトコルは、FTP のみである。

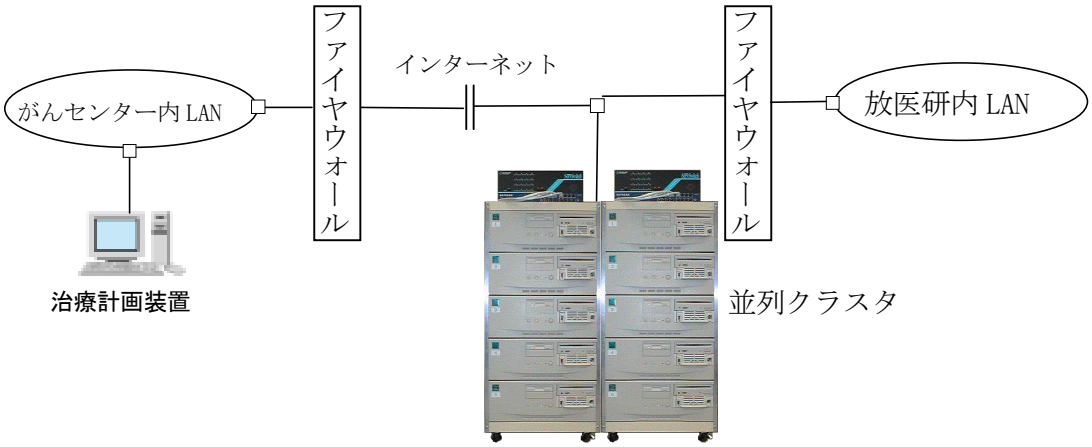


図 14　放医研とがんセンターの接続形態

放射線治療計画を模擬した試験として、がんセンターの治療計画装置で計算条件ファイルを作成し、CT 画像と共に放医研へデータを送って、線量分布を放医研にある並列クラスタで計算し結果をがんセンターへ返す。そして、計算結果をがんセンターの治療計画計算機上で CT 画像と重ね合わせて表示する。

試験は、以下のような手順で行った。

(a) CT 画像を治療計画装置から並列クラスタに転送し、電子密度分布データに変換する。

(b) 治療計画装置で、オペレータが照射数、照射方向、ターゲット直径を設定する。計算結果は、全方向で重ね合わせたものになるので、1 方向の結果だけを見たい場合は、照射数を１と指定する。この設定はファイル編集で行えるようパラメータファイはテキストファイルとする。

(c) オペレータが治療計画装置から並列クラスタに線量計算の開始を指示する。

(d) 全向の計算が終了したら、治療計画装置画面上にその旨（計算結果ではなく、実行時間等）を表示する。

(e) 計算結果は全方向で重ね合わせたものがあるので、オペレータは適当に結果ファイルを選択して、表示装置の画面に表示させる。

(f) 適当に(b)からやり直す。

表２に実行時間の計測結果を示す。なおこの表で、“－”は状態ファイルの更新や読み込みの時間であり、その性質上、試験では経過時間の計測ができなかった。この計測結果から、以下のことが言える。

- 今回の試験では、インターネットを使用し、FTP によるファイル転送を用いたため、通信性能がかなり悪くなっている。実効性能として、1Mbps 程度しか出ていない。並列クラスタでの処理は、半分以上がファイル I/O の時間となっている。
- 個々のフェーズの実行時間を総計した時間と、開始から終了まで実測した時間の差（(2)－(1)）が約 14 秒となっている。これは、恐らく、状態ファイルの更新・取得の

時間と考えられる。状態ファイルのサイズは非常に小さいので、状態ファイルの更新・取得の 1 回当たりの時間は、1 秒以下であることが別計測で分かっている。しかし、状態ファイルの更新・取得を 2 回連続して実行した時の時間は、5 秒以上と増大していた。恐らくこれは、FTP の終了（陰での後始末）に時間を要していると考えられ、2 回目の FTP の起動がその前の FTP の終了処理が終わるまで開始できないためであると思われる。従って、FTP を使って定期的な監視を行うことは、非常に効率が悪いと言える。

以上のことから、仮により高速なネットワークを使用し、ファイル転送ではなくメモリ間転送を行えば、かなり高速化できると言える。例えば、試験に使用したインターネットより 100 倍高速なネットワーク（実効性能が 100Mbps）を使用し、PVM などを用いてファイルを介さずメモリ間転送で通信するとすると、全体の実行時間は表 2 の右側のようになる。なお、第 II 期の最後には、専用の高速ネットワーク（通信仕様は 622Mbps）を用いた実験を予定している。

- CT データの転送は、ネットワークが高速化されるので圧縮せずに送る。CT データは圧縮することによりデータサイズが半分になっているので、圧縮しない場合のデータ転送時間は 1/50（2*1/100）となる。また、データ圧縮、伸長の時間は 0 となる。
- パラメータファイルの転送時間は、1/100 となる。
- メモリ間転送を行うので、CT 画像読み込み及び線量分布ファイルの出力時間は 0 となる。
- 線量分布計算結果は、圧縮することによりデータサイズが 1/20 になっているので、圧縮しない場合のデータ転送時間は 1/5（20*1/100）となる。同様に、データ圧縮、伸長の時間は 0 となる。

これにより、全実行時間が 316 秒から 95 秒に短縮される。この時間の内、治療計画装置上の処理時間は 76 秒であり、約 8 割が表示に係わる処理となる。表示に関しては、今後高速化できる余地がかなり残されていると思われる。

実際の治療計画においては、CT データを一旦転送しておき、照射方向やターゲットサイズを何度も変更しながら線量計算が繰り返される。その繰り返される CT データ転送後の時間を見てみると、治療計画装置上の処理時間は 60 秒であり、線量計算の時間が 9 秒強であることから、ネットワークが高速化されることにより、通信時間はほとんど無視できる程度の時間となっている。

表２　計測時間結果

装置名	フェーズ	計測時間（秒）		高速ﾈｯﾄﾜｰｸ化（秒）
治療計画装置	CTデータ作成		16.86	16.86
	CTデータ圧縮		47.29	
	CTデータ転送		134.87	2.70
並列クラスタ	状態ファイルの取得1		－	
	CTデータ伸張		5.00	
	電子密度データ変換		5.00	5.00
	状態ファイルの更新		－	
治療計画装置	状態ファイルの取得2		－	
	パラメータファイル作成		0.01	0.01
	パラメータファイル転送		5.25	0.05
並列クラスタ	状態ファイルの取得3		－	
	CT画像の読み込み	11.86		
	線量計算 方向１	3.10		3.10
	線量計算 方向２	3.14		3.14
	線量計算 方向３	3.19		3.19
	線量分布ファイルの出力	2.01	23.31	
	線量分布ファイル圧縮		3.00	
治療計画装置	状態ファイルの取得4		－	
	線量分布ファイル転送		11.07	2.21
	線量分布ファイル伸張		0.84	
	線量分布読み込み		4.72	
	等線量分布計算		58.82	58.82
	等線量線表示		0.19	0.19
	(1) 合計時間		316.23	95.28
	CTデータ転送後の合計時間		107.21	70.72
	(2) 表示計測時間		330.54	
	差((2)-(1))		14.31	

6. 米国における研究状況

米国においても、線量分布計算の高速化や高速ネットワークを用いた遠隔治療計画の研究開発が行われている。

6.1 ワシントン大学

1993 年ころからワシントン大学で、トランスピュータを用いた線量分布計算の並列処理研究が行われている[2]。[2]では、Ray を分割して並列化するのではなく、三次元 CT 画像の方を分割して並列化している。その理由は、Ray を分割する方式では、CT 画像を全てのプロセッサで保持する必要があるからと述べている。しかし、パラメータを変えながら何度も繰り返される処理部分を高速化すれば良いので、一旦 CT 画像を各プロセッサに転送してしまえば、何度も繰り返される処理においては，CT 画像の転送は不要である。

また、[2]では、プロセッサの境界で Ray 上の計算点の計算結果を隣接プロセッサに送

信する必要があり、従来の逐次方式からの変更も大きくなる。一方、先に述べた方式では、1 本の Ray 上の計算は従来の逐次方式（プログラム）をそのまま利用できるという利点がある。また、計算結果は最後に 1 度だけ圧縮して送信すれば、それほど大きなオーバヘッドになっていないことが、計測結果から確認できた。

6.2 VISTANET

こちらも 1993 年頃から、ノースカロライナ大学で CRY-YMP を用いて線量分布計算の並列処理研究が行われている[3]。本研究においても、ギガビットレベルの高速ネットワークを用いた遠隔実験を行っている。

6.3 PEREGRINE

Lawrence Livermore 国立研究所により、商用の x86 プロセッサボードを複数毎 small private network で接続して、線量分布計算を高速化している（図 15）。

7. まとめ

以上、高速ネットワーク環境下における高度医療アプリケーションについて、報告した。今後、ネットワークはますます高速化され、医療アプリケーションも高度化されていくので、この種の研究の重要性はますます高くなると思われる。

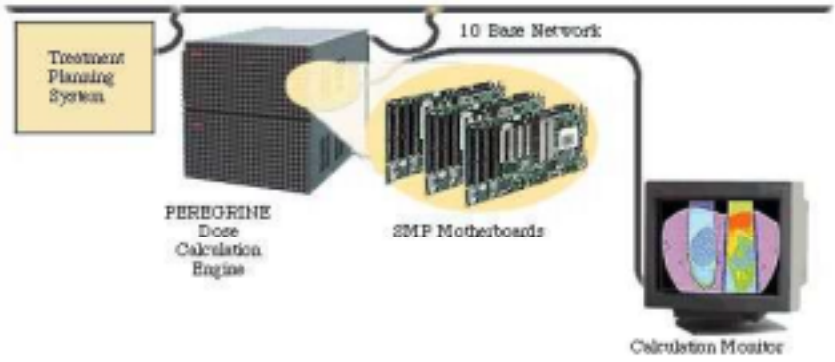


図 15　PEREGRINE

参考文献

[1] 辻井博彦, “重粒子放射線によるがん治療の現状,” バイオメカニズム学会誌, vol.21, no.3, pp.108-112, Mar. 1997.

[2] F. U. Rosenberger, J. W. Matthews, G. C. Johns, R. E. Drzymala, and J. A. Purdy, “Use of transputers for real time dose calculation and presentation for three-dimensional radiation treatment planning,” Int. J. Radiat. Oncol. Bool. Phys.,

vol.25, no.4, pp.709-719, Apr. 1993.

[3] J. G. Rosenman, E. L. Chaney, T. J. Cullip, J. R. Symon, V. L. Chi, H. Fuchs, and D. S. Stevenson, “VISTANET: Interactive real-time calculation and display of 3-dimensinal radiation dose: an application of gigabit networking,” 'Int. J. Radiat. Oncol. Bool. Phys., vol.25, no.1, pp.123-129, Jan. 1993.

[4] Lawrence Livermore National Lab., “PEREGRINE - Advancing the field of radiation treatment for cancer,” http://www-phys.llnl.gov/peregrine/index.html, Oct. 2000.

[5] 緑川博子編, “特集：計算機クラスタ,” 情報処理, vol.39, no.11, pp.1072-1100, Nov. 1998.

[6] L. Hong, M. Goitein, M. Bucciolini, R. Comiskey, B. Gottschalk, S. Rosenthal, C. Serago, and M. Urie, “A pencil beam algorithm for proton dose calculations,” Phys. Med. Biol. no.41, pp.1305-1330, Apr. 1996.

[7] Computer Science & Mathematics Division Oak Ridge National Lab., “PVM: Parallel virtual machine,” http://www.csm.ornl.gov/pvm/, Dec. 2000.

[8] 白石將,佐藤裕幸,中島克人, “分散型並列処理支援ツール ParaJET,” 信学技報, CPSY96-60, pp.23-30, Aug. 1996.

第４章　あとがき

本年度における HECC ワーキンググループの調査は、過去２年間の「HECC ワーキンググループ」における調査活動の結果や報告書を踏まえ、高性能コンピューティングに関連する情報処理技術のあるべき姿を探るとともに、今後注力すべき技術分野の検討に参考となる調査報告や意見などを集約したものである。第２章では、昨年と同様に米国のハイエンドコンピューティング研究開発動向について、主として米国における政府支援の研究開発に関して、Blue Book を参考にして、まとめてある。また、３章では、ハイエンドコンピューティング研究開発の動向と題して、本ワーキンググループの各委員および外部講師による、ハイエンドコンピューティングに関する研究開発に対する見解がまとめられている。ここでは、関連する項目を、アーキテクチャ＆新計算モデル、基本ソフトウェア＆ミドルウェア、応用システム＆応用分野の３つに大別して述べてあるが、各項目における技術に関する見解はその項目だけにとどまらずに、より多岐にわっていることに注目して欲しい。情報処理における基本的な技術の方向性や将来性あるいはその有効性は、それ単独で検討されるべきものではなく、他の技術との関連のなかでいかにとらえるべきであるかということの方がますます重要になってきていると考えられるからである。

本報告書では、アーキテクチャ＆新計算モデルの項目において、情報処理を支えるハードウェアの基である半導体の将来についていくつかの見解が述べられている。その１つは、半導体の性能向上の神話について、従来とは異なる新たな時代に入ったという指摘である。その１つは、配線遅延がゲート遅延に比べてより支配的になるであろうという点と、電源電圧の低下に歯止めがかかるのではないかという点である。もう一つの見解は、プロセッサの高性能化が進むに従って、その消費電力の増大が問題になってきている点である。これらの問題を解決するために、リコンフィギュラブル・アーキテクチャやハード／ソフト協調による低電力高性能プロセッサの開発などがあげられている。これらを概観すると、これからの半導体デバイスやハードウェアアーキテクチャの研究開発指針としては、従来型の微細加工技術やデバイス技術だけに頼るのではなく、アーキテクチャ技術やコンパイラ技術などを包含した技術開発が必要になることが指摘できる。折しも、2001 年の 12 月に東芝の汎用 DRAM メモリからの撤退というニュースが駆けめぐった。東芝だけではなく、メモリ製造業は近年最悪の不況に苦しんでいた。過剰な製造キャパシティと PC 需要の停滞による深刻な価格の落ち込みにより、多数のメーカーがコスト割れの価格で製品を販売せざるを得なくなってきており、このような事態は早晩予想されるものではあったが、これが具体的な形で出てきたわけである。東芝のメモリ事業自体は、今後フラッシュメモリを中核に、アプリケーションを強く意識した、高付加価値のメモリ製品に特化するとのことであるが、その一方で最先端の DRAM 混載システム LSI やシステム・オン・チップの研究開発に力を入れることが予想される。

このような一連の流れの中には、例えば、新エネルギー・産業技術総合開発機構(NEDO)が半導体理工学研究センター（STARC）に委託して行っている、システムオンチップ先端設計技術の研究開発というプロジェクトがある。これは、平成 12 年度から平成 16 年度のまでの５年間で、Ｖコアと呼ばれる再利用可能な機能ブロックをシステムレベル、アーキテクチャレベルなどの上位の抽象度レベルで表現したものを開発するというものである。このように、我が国の半導体産業においては、いわゆるシステム LSI を最重点強化として取り上げている。しかしながら、日本の大手の半導体部門の戦略として、すべてが横並びになっているのは問題があるのではないだろうか。米国のマイクロンのように、「DRAM で世界ナンバーワンの地位を奪回」という方針を打ち出す企業が現れてもよいような気もする。システム LSI の強化についても、そのビジネスの本質である「少量多品種」に対しての対策が行われているかどうかは疑問である。たとえば、0.15 ミクロンのような微細化プロセスと 300mm ウェハーでの半導体生産で 100 万ゲートの規模のロジック LSI を生産すると仮定してみるとする。歩留まりが良いと仮定すれば、１枚のウェハーで約 4000 個～8000 個のチップが、1 回のロット（ここでは 24 枚としてみる）で約 10 万個以上のチップ（1000 万ゲートとしても約 1 万個以上）が生産されてしまうことになる。システム LSI として、成功を収めたプレイ・ステーション 2 用の LSI のような特殊な例を除けば、このようなアプリケーションはそれほど多くは見つけることができないかも知れない。従って、少量生産でも効率の高い生産ラインの構築を図るとともに、デバイスに FPGA などのリコンフィギュラブルな領域を組み込むなどによって多品種に対応するなどの方式も必要になってくると考えられる。世界の半導体メーカの競争においては、インテルの一人勝ちの状況が続いており、たとえば米国の半導体メーカである TI やモトローラも各メーカの得意分野に特化するという企業戦略のものに、たとえば TI は DRAM から完全撤退して DSP や信号処理などの分野にフォーカスしており、モトローラ社はディスクリートや汎用ロジック製品を担当していた部門を売却するという思い切った決断を下している。我が国の半導体各社も、横並びではなく、各社の特性を生かした戦略が必要になっているのではないだろうか。

一方、我が国のハイエンドのコンピュータに目を向けると、2002 年の 3 月に「地球シミュレータ」が稼働を始めたことは特筆すべきことである。地球シミュレータの中身については、本報告の中で詳しく述べられているので、ここでは特に言及はしないが、米国のハイエンドコンピュータあるいは、日本の富士通や日立のスーパーコンピュータが全ていわゆる超並列スカラー計算機に向かっているのに対して、ベクトル並列という方式で押し進めたという点が特徴的であり、１つのユニークな戦略性として高く評価できる。また、地球シミュレータという名前の通り、地球規模の問題である地球温暖化やエルニーニョ現象などの環境変動を解明および予測することに貢献すれば、米国の ASCI 計画とは違った意味において、大きなインパクトを与えることができよう。

平成 10 年度の報告書（その時は「ペタフロップスマシン技術調査ワーキンググループ」）

のあとがきの中では、我が国における情報処理産業において脱メモリのあるべき姿について議論を行ったが、これからは「脱 PC」という観点から情報処理産業の姿を考えてみたらいかがであろうか。PC は、1990 年代の情報処理分野における牽引車であったが、21 世紀に入って、その牽引力も少し小さくなってきているように感じる。これからの情報処理機器は PC ばかりではなく、携帯電話や PDA それに情報家電やスマートカードあるいはユビキタスコンピュータと言われるように、いろいろなところに入り込んでいくことになろう。そのような状況においては、PC 時代を WinTel という 2 大企業が制覇したような状況にはなっていかないと思われる。これは、我が国にとっては挽回するチャンスであり、一見すると我が国にとって有利な状況であるように見えるが、しかしながら 1980 年代や 90 年代などとは異なる状況が生まれている点も注意が必要である。それは、アジアの情報処理産業における韓国や台湾それに中国などの台頭であり、またインドなどにおけるソフトウェア産業の著しい成長である。我が国としては、これらの状況を踏まえ、各企業の得意分野を定めて、戦略を持って対峙することがますます必要になってくると考えられ、個々の企業や研究者の主体性がさらに要求されるようになってくるのではないだろうか。

（山口　喜教　主査）

付表1　ハイパフォーマンス・コンピュータ世界のTop 20

順位	メーカ	マシン	R_{max}(GFlops)	設置場所	国	設置年	使用分野	CPU数
1	IBM	ASCI White,SP Power3 375 MHz	7226	Lawrence Livermore National Laboratory	USA	2000	Research	8192
2	Compaq	AlphaServer SC ES45/1 GHz	4059	Pittsburgh Supercomputing Center	USA	2001	Academic	3024
3	IBM	SP Power3 375 MHz 16 way	3052	NERSC/LBNL	USA	2001	Research	3328
4	Intel	ASCI Red	2379	Sandia National Labs	USA	1999	Research	9632
5	IBM	ASCI Blue-Pacific SST,IBM SP 604e	2144	Lawrence Livermore National Laboratory	USA	1999	Research	5808
6	Compaq	AlphaServer SC ES45/1 GHz	2096	Los Alamos National Laboratory	USA	2001	Research	1536
7	Hitachi	SR8000/MPP	1709.1	University of Tokyo	Japan	2001	Academic	1152
8	SGI	ASCI Blue Mountain	1608	Los Alamos National Laboratory	USA	1998	Research	6144
9	IBM	SP Power3 375 MHz	1417	Naval Oceanographic Office (NAVOCEANO)	USA	2000	Research	1336
10	IBM	SP Power3 375 MHz 16 way	1293	Deutscher Wetterdienst	Germany	2001	Research	1280
11	IBM	SP Power3 375 MHz 16 way	1272	NCAR (National Center for Atmospheric Research)	USA	2001	Research	1260
12	NEC	SX-5/128M8 3.2ns	1192	Osaka University	Japan	2001	Academic	128
13	IBM	SP Power3 375 MHz	1179	National Centers for Environmental Prediction	USA	2000	Research	1104
14	IBM	SP Power3 375 MHz	1179	National Centers for Environmental Prediction	USA	2001	Research	1104
15	Cray Inc.	T3E1200	1127	Government	USA	2001	Classified	1900
16	IBM	SP Power3 375 MHz 16 way	1100	Lawrence Livermore National Laboratory	USA	2001	Research	1088
17	Hitachi	SR8000-F1/112	1035	Leibniz Rechenzentrum	Germany	2000	Academic	112
18	IBM	SP Power3 375 MHz 8 way	929	UCSD/San Diego Supercomputer Center	USA	2000	Academic	1152
19	Hitachi	SR8000-F1/100	917	High Energy Accelerator Research Organization /KEK	Japan	2000	Research	100
20	Cray Inc.	T3E1200	892	US Army HPC Research Center at NCS	USA	2000	Research	1084

出典：http://www.top500.org/　における2001年11月版
全てのカテゴリーの総合順位でRmax順にソート

付表2　クラスタコンピュータ世界のTop20

順位	Top500	メーカ	Computer	Rmax(GFlops	設置場所	国	設置年	分野	CPU数
1	2	Compaq	AlphaServer SC ES45/1 GHz	4059	Pittsburgh Supercomputing Center	USA	2001	Academic	3024
2	6	Compaq	AlphaServer SC ES45/1 GHz	2096	Los Alamos National Laboratory	USA	2001	Research	1536
3	30	Self-made	CPlant/Ross Cluster	706.7	Sandia National Laboratories	USA	2001	Research	1369
4	31	Compaq	AlphaServer SC ES45/1 GHz	706	Australian Partnership for Advanced Computing (APAC)	Australia	2001	Academic	480
5	34	IBM	MHz	677.9	NCSA	USA	2001	Academic	320
6	39	NEC	Magi Cluster PIII 933 MHz	654	CBRC - Tsukuba Advanced Computing Center - TACC/AIST	Japan	2001	Research	1040
7	40	Self-made	SCore IIIe/PIII 933 MHz	618.3	Real World Computing (RWCP)/Tsukuba Research Center	Japan	2001	Research	1024
8	41	IBM	Netfinity Cluster PIII 1 GHz	594	NCSA	USA	2001	Academic	1024
9	54	Compaq	AlphaServer SC ES40/EV67	507.6	Compaq Computer Corporation	USA	2000	Vendor	512
10	55	Compaq	AlphaServer SC ES40/EV67	507.6	Lawrence Livermore National Laboratory	USA	2000	Research	512
11	68	HPTi	ACL-580	442.7	Forecast Systems Laboratory/NOAA	USA	2001	Research	580
12	70	Sun	HPC 4500 400 MHz Cluster	420.44	Defense	Sweden	1999	Classified	896
13	71	Sun	HPC 4500 400 MHz Cluster	420.44	Service Provider	USA	2000	Industry	896
14	72	Sun	HPC 4500 400 MHz Cluster	420.44	Service Provider	USA	2000	Industry	896
15	73	Sun	HPC 4500 400 MHz Cluster	420.44	Sun	USA	2000	Vendor	896
16	81	Compaq	MHz	344.1	ERDC MSRC	USA	2001	Research	256
17	82	Compaq	MHz	344.1	Japan Marine Science and Technology	Japan	2001	Research	256
18	83	Compaq	MHz	344.1	Los Alamos National Laboratory	USA	2001	Research	256
19	86	Self-made	Presto III Athlon 1.2 GHz	331.7	Technology	Japan	2001	Academic	256
20	103	Dell	Precision 530 Cluster XEON 1.7 GHz	288.9	Sandia National Laboratories	USA	2001	Research	256

出典：http://www.top500.org/　における2001年11月版
ComputerクラスをClusterにしてソートしたもの

付表3　日本製ハイパフォーマンスコンピュータ　Top20

順位	TOP500順	メーカ	マシン	R_{max}(GFlops)	設置場所	国	設置年	使用分野	CPU数
1	7	Hitachi	SR8000/MPP	1709.1	University of Tokyo	Japan	2001	Academic	1152
2	12	NEC	SX-5/128M8 3.2ns	1192	Osaka University	Japan	2001	Academic	128
3	17	Hitachi	SR8000-F1/112	1035	Leibniz Rechenzentrum	Germany	2000	Academic	112
4	19	Hitachi	SR8000-F1/100	917	High Energy Accelerator Research Organization /KEK	Japan	2000	Research	100
5	21	Fujitsu	VPP5000/100	886	ECMWF	UK	2000	Research	100
6	22	Hitachi	SR8000/128	873	University of Tokyo	Japan	1999	Academic	128
7	26	Hitachi	SR8000-G1/64	790.7	Institute for Materials Research/Tohoku University	Japan	2001	Academic	64
8	28	Fujitsu	VPP5000/80	730	University of Tsukuba	Japan	2001	Research	80
9	32	Hitachi	SR8000-E1/80	691.3	Japan Meteorological Agency	Japan	2000	Research	80
10	39	NEC	Magi Cluster PIII 933 MHz	654	CBRC - Tsukuba Advanced Computing Center - TACC/AIST	Japan	2001	Research	1040
11	42	Hitachi	SR8000-F1/60	577	University of Tokyo/Institute for Solid State Physics	Japan	2000	Academic	60
12	43	Fujitsu	VPP5000/64	563	Kyushu University	Japan	2000	Academic	64
13	57	Fujitsu	VPP5000/56	492	Nagoya University	Japan	1999	Academic	56
14	58	Fujitsu	VPP800/63	482	Kyoto University	Japan	1999	Academic	63
15	61	Hitachi	SR8000/64	449	Tsukuba Advanced Computing Center - TACC/AIST	Japan	1999	Research	64
16	89	Fujitsu	VPP700/160E	319	Institute of Physical and Chemical Res. (RIKEN)	Japan	1999	Research	160
17	98	NEC	SX-5/40M3	303	CNRS/IDRIS	France	2000	Academic	40
18	101	Fujitsu	VPP5000/32	296.1	Central Research Institute of Electric Power Industry/CRIEPI	Japan	2000	Research	32
19	105	Fujitsu	VPP5000/31	286	Meteo-France	France	1999	Research	31
20	109	Fujitsu	VPP5000/30	277	National Inst. for Molecular Science	Japan	2000	Research	30

出典：http://www.top500.org/　における2001年11月版
CountryをJapanとしてソートしたもの

平成１３年度ワーキンググループ活動記録

開催場所：先端情報技術研究所（AITEC）内会議室
開催時間：18:00　～　　（第 5 回を除く）

開催日	討議内容
第 1 回 平成 13 年 10 月 15 日	・山口主査　挨拶 ・内田所長　挨拶 ・新任委員の紹介（佐藤裕幸委員、佐藤真琴委員） ・WG 幹事「平成 13 年度活動方針案」 ・招待講演 理化学研究所　ゲノム科学総合研究センタ　八尾徹氏 「ゲノム･ポストゲノム時代におけるコンピュータ･情報技術へのニーズ」
第 2 回 平成 13 年 11 月 26 日	・招待講演 産業技術総合研究所　ハイエンド情報技術グループ　田中良夫氏 「SC2001 にみる Grid の最新動向―Globus, Portal, SC Global」 ・委員講演　佐藤裕幸委員 「高速ネットワーク環境下における高度医療アプリケーション」
第 3 回 平成 13 年 12 月 17 日	・招待講演 RedSwitch, Inc.　　Dr. Hungwen Li 「InfiniBand: Switch Fabric Solution for System Connectivity」 ・委員講演　佐藤真琴委員 「手続き間解析の動向　～コンパイラとその周辺」
第 4 回 平成 14 年 1 月 28 日	・招待講演 NTT　未来ねっと研究所　　星合隆成氏 「Ｐ２Ｐの理念とその動向」 ・委員講演　中島浩委員 「超低電力技術に基づくディペンダブルメガスケールコンピューティング」
第 5 回 平成 14 年 2 月 25 日 PM4:00	・地球シミュレータ見学 （場所：海洋科学技術センター　横浜研究所） 概要説明　横川三津夫委員

ハイエンドコンピューティング技術に関する調査研究Ⅲ

© 平成 14 年 3 月発行

発行所　財団法人　日本情報処理開発協会
先端情報技術研究所
東京都港区芝 2 丁目 3 番 3 号
芝東京海上ビルディング 4 階
TEL(03)3456-2511