超並列システムの将来像と課題

井口 寧

2001年12月13日
IS-RR-2001-0028

北陸先端科学技術大学院大学
情報科学センター
〒923-1292 石川県能美郡辰口町旭台1-1
inoguchi@jaist.ac.jp

ISSN 0918-7553

要旨

大学における高度な教育研究活動を支援する情報環境を構築することは，情報科学センターにおいて非常に重要な役割の一つである．実用システムとして構築する場合は，既存の計算機システムを組み合わせて，必要とされる情報環境を提供する．一方で，将来にわたる大学の教育研究活動を支援するために，将来望まれる情報環境整備に必要な技術要素の研究開発もまた情報科学センターにとって重要な課題である．

本論文では，北陸先端科学技術大学院大学 情報科学センターの超並列計算サーバを通じて，学術目的の情報科学センターが将来提供すべき超高速計算サーバに必要とされる技術とその解決方法について検討する．最初に同センターにこれまで導入された超並列システムについて検討し，これらのシステムの特徴，利点，課題となっている点について議論する．同センターの超並列システムを用いて，相互結合網の通信性能がシステム全体の性能に与える影響について調べたところ，PE 間通信のバンド幅が重要であり，特に通信遅延がシステム全体に大きく影響することが分かった．

相互結合網のトポロジについて検討したところ，同センターが導入しているような二次元や三次元のトーラス結合網が科学技術計算に適していることが示された．三次元トーラス網のシステムは同センターで既に稼働しており，高い計算性能を有していることが示されているが，将来超高速計算サーバとなり得る数百万のプロセッサを有するシステムへの拡張のためには，再帰的な結合網が有効であることが分かった．再帰的なトーラス網として「再帰シフトトーラス相互結合網」について詳しく議論した結果，本相互結合網は従来の相互結合網と遜色ない通信性能を有しながら，非常に高い拡張性があることが分かった．

また，実装手法について議論し，システム全体の高速化のためには，システムを高度に集積することが可能な三次元実装が重要であることを示した．三次元実装では放熱が重要な課題であるが，放熱を考慮して予備 PE を選択する「WSI スタックのヒューリスティック配置方式」を用いることにより，ウェーハスタックのスタック内最高温度を大幅に冷却できることが分かった．

これらの知見を踏まえて，情報科学センターで提供すべき，将来の超高速計算サーバを実現するための課題の解決への見通しを述べた．

1 はじめに

教育研究活動を支援する情報環境を構築することは，大学の情報科学センターにおいて非常に重要な役割の一つである．実用システムとして構築する場合は，既存の計算機システムを組み合わせて，必要とされる情報環境を提供する．一方で，将来にわたる大学の教育研究活動を支援するために，将来望まれる情報環境整備に必要な技術要素の研究開発もまた情報科学センターにとって重要な課題である．本論文では，北陸先端科学技術大学院大学 (以下 本学) 情報科学センターの超並列計算機群を導入・管理・運営を通じて得られた知見をもとに，将来の情報科学関連のセンターに必要な超並列システムのアーキテクチャを明らかにし，その実現手法について展望する．

自然科学におけるシミュレーションやVLSI設計など，先端科学技術分野における大規模科学技術計算の需要は増大しており，多数のマイクロプロセッサ (MPU) を用いた超並列計算機による高速処理が求められている．このような学内の先進的な研究を支援するため，本学 情報科学センターは，同センターの機能の一つであるコンピュテーションセンターと情報科学研究科の教育研究設備との両面における基盤情報環境として，様々なアーキテクチャを有する超並列計算機群を導入してきた．本センターは最新鋭の超並列計算機システムやソフトウェアシステムの導入，維持管理，及びユーザー教育を行ない，研究基盤の整備を行なってきた．このような研究基盤を活用し，情報科学研究科 [1, 2, 3]，材料科学研究科，および知識科学研究科は，超並列計算機群を活用して優れた研究を行なってきている．

しかしながら，本学発足当初は,「超並列計算機」というだけで非常に先進的なシステムであったが，現在においては，超並列システムは既に実用システムとして用いられている．情報科学センターが本学の先進的な研究活動を支えるためには，新しいアーキテクチャや技術を開拓する必要が強く望まれている．

そこで，本論文では，これまでに本学 情報科学センターにおいて導入運営されてきた超並列システムについて分析し，次世代の研究を支える超高速計算機システムに必要な技術を展望し，現在研究がなされている課題について議論する．最初にこれまで情報科学センターが導入したシステムの特徴について述べる．第3章で現有システムを用いた性能評価を行ない，既存のシステムの問題点を探る．第4章で，相互結合網について議論し，数百万のプロセッサを用いた真の超並列計算機システムを実現するための結合方式について議論する．第5章では，システムを現実のものとするための実装技術について展望し，高度な集積を可能とするための三次元実装技術の放熱手法について研究を行なう．第6章はまとめである．

2 これまでの超並列計算機群

2.1 情報科学センターの超並列システムの遷移

本学には開学以来，多数の超並列計算機が設置され，教育および研究に盛んに活用されてきた．表1に本学に導入された超並列計算機の一覧を示す．超並列処理研究用システムとして CM5 と T3E，高度データベース処理研究用システムとして nCUBE2，nCUBE3，並列ソフトウェア研究用システムとして Parsytec GC/MPC-128，さらに情報環境の一部として，Ultra Enterprise 10000 を導入している．このうち，CM5, CRAY-T3E, RS/6000-SP, CRAY-T3E-1200E は，数値演算に適したアーキテクチャとなっており，高い演算性能を有している．

2.2 CM-5

本学で最初に導入された超並列計算機 Thinking Machines 社の Connection Machine CM5 は，64 プロセッサで構成され，最大で毎秒 80 億回の浮動小数点演算が可能である．使いこなすのがかなり困難な一面もあるが，適当な問題においてプログラマが注意深く使用すれば非常に高い性能が得られるシステムとなっている．主な特徴は次の通りである．

1. 最大で 8192CPU までの拡張性
2. 分散記憶，メッセージパッシングアーキテクチャ
3. 単位プロセッサにベクトル演算機構を付加したことによる高速性
4. 合計 2G バイトの記憶域，23G バイトのディスクアレイ (SDA)
5. 多様なソフトウェア環境

2.2.1 ハードウェア構成

CM5 の相互結合網は，図1に示すような不完全 FAT Tree である．

表2に，全体の構成および各プロセッシングエレメント (PE) の構成を示す．並列化されたプロセスがそれぞれのノードのローカルメモリにロードされ，実行される分散メモリ型のアーキテクチャである．

インターフェースユニットは 20MB/s のパケット発信能力を持っているが，ノードプロセッサ自体が低速であるため，実測値としてのノード対ノードの通信能力は最良でも 10MByte/sec 程度に制限される．

表 1: 本学における超並列計算機

メーカー・機種	CPU 数	設置期間
nCUBE 社・nCUBE2	256	1993-1996
Thinking Machines 社・CM5	64	1993-1996
	128	1996-1997
Parsytec 社・GC/MPC-128	128	1994-1997
nCUBE 社・nCUBE3	320	1996-1999
SiliconGraphics 社・CRAY T3E	136	1997-2000
SunMicrosystems 社・Ultra Enterprise 10000	32	1998-2001
IBM 社・RS/6000 SP	288	1999-
Cray Inc. 社・CRAY T3E-1200E	136	2001-

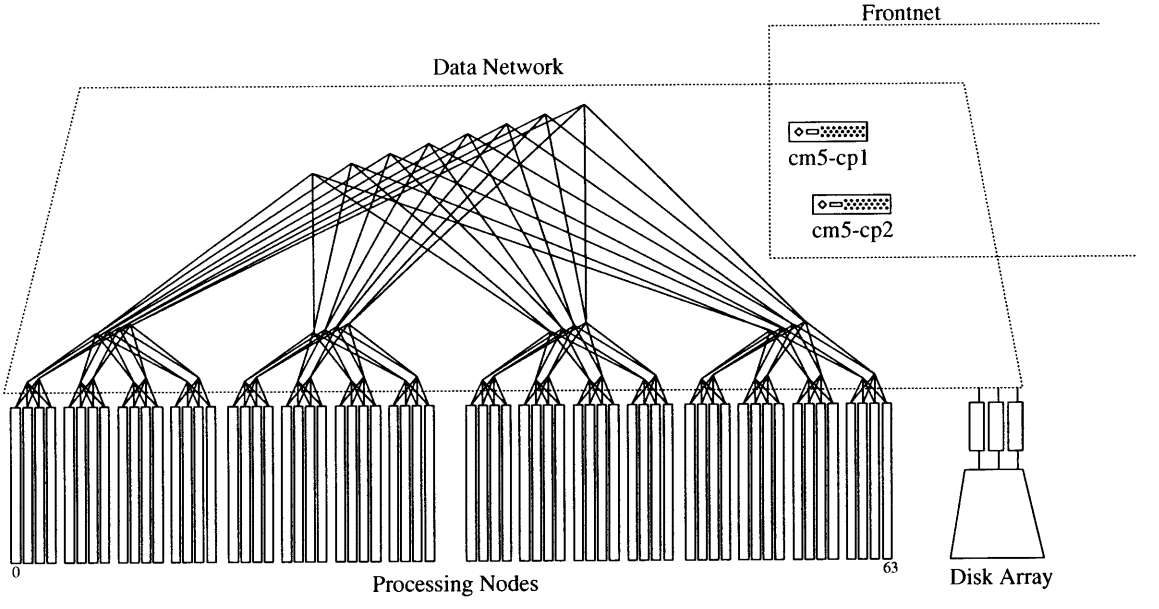


図 1: 64 プロセッサの CM5 の結合ネットワーク

2.2.2 ソフトウェア

CM5 の特徴の一つは，並列計算機としては極めて豊富な言語やソフトウェアツールを有していたことである．表 3 に利用可能なソフトウェアを示す．

特に CM-Fortran はその当時で Fortran90 の自動並列化記述を先取りする形で実現され，CM5 上で開発された超並列ソフトウェアは，その後の超並列システムに

表 2: CM5 の構成

相互結合網	FAT Tree
ノード数	64
ノードプロセッサ	Sparc IU, FPU (33MHz,22Mips,4.2MFlops), Vector Processing Unit (16MHz, 32Mflops) × 4
ローカルメモリ	32MB/node
PE 間通信速度	20MB/s
ノード演算能力/PE 間通信速度	6.4MFlops/(MB/s)

表 3: CM5 の構成

名称	概要
CM-Fortran	限定的な自動並列化コンパイラ言語，拡張 Fortran90
C*	限定的な自動並列化ができる拡張 C 言語
STARLISP	Commonlisp の並列拡張
Prism	並列グラフィカルデバッガ，兼簡易ビジュアライゼーション
CMMD	メッセージパッシングを提供するライブラリ
CMSSL	並列数値演算ライブラリ
CMX11	低レベルグラフィック出力ライブラリ
pndbx	並列用デバッガ
CMView	ハイパーテキスト風オンラインマニュアル
CMFS	SDA 用並列入出力ライブラリ
CMNA	低レベル通信ライブラリ
CDPEAC	VectorUnit 用アセンブラマクロ

おいても，スムースに利用できたことは，ソフトウェア上非常に有意義である．

2.3 Cray T3E

Cray T3E は，CM5 の機種更新によって導入された，本学の数値計算系の超並列システムの 2 世代目の機種である．表 4 に T3E の構成を示す．主に科学技術計算の超高速演算を目的とする超並列計算機で，CPU，ローカルメモリ及び通信プロセッ

表 4: T3E の構成

相互結合網	3D Torus
ノード数	128
ノードプロセッサ	Alpha 21164a (300MHz,600MFlops)
ローカルメモリ	64MB/node
PE 間通信速度	480MB/s
ノード演算能力/PE 間通信速度	1.25MFlops/(MB/s)

サから成る PE ノードを，3 次元トーラス相互結合網で 128 台結合した分散メモリ型の並列計算機である．相互結合網のネットワークインターフェースは，CPU のバスに直接接続され，480MB/s という CPU の能力に対して非常に高速な PE 間通信速度を有している．プロセッサ間通信は通信リンクごとに並行に行われるため，1PE ノードは合計 2.88GB/s の実効転送能力がある．

各 PE ノードは，Compaq 社の高速 RISC プロセッサ Alpha 21164 を核としており，C-chip(メモリ/データ制御)，R-chip(ネットワーク・ルータ)，及び 64MB のローカルメモリから構成されている．導入当時，Alpha 21164a プロセッサは，非常に高いクロック周波数で作動する，世界最速のマイクロプロセッサであり，ノード当り 300MHz の Alpha チップを用いて 600MFLOPS，1200MIPS の演算性能を実現している．

本計算機の特徴の 1 つとして，CPU とメモリ間のデータ転送能力が非常に高いことが挙げられる．ストリームバッファや E レジスタという高速なデータ供給を行う機構によって，1.2GB/s という高いデータ転送能力を達成し，実際の利用においても高い演算性能を有している．

ソフトウェアは，強力な資源割当て機能を持つ OS UNICOS/mk，データパラレルモデルに基づく Fortran90 や分散処理が可能な C++/C が利用可能である．CM5 の時には，PE 間通信ライブラリは標準化されたものがなく，Thinking Machines 社独自のものであったが，T3E では，標準化された並列通信ライブラリとして PVM や MPI が利用可能となっている．また，数値演算ライブラリとして，IMSL が用意されている．

表 5: T3E-1200E の構成

相互結合網	3D Torus
ノード数	128
ノードプロセッサ	Alpha 21164a (600MHz,1.2GFlops)
ローカルメモリ	512MB/node
PE 間通信速度	650MB/s
ノード演算能力/PE 間通信速度	1.85MFlops/(MB/s)

2.4 Cray-T3E-1200E

Cray T3E-1200E は，T3E の機種更新によって導入された，本学の数値計算系の超並列システムの 3 世代目の機種である．表 5 に T3E-1200E の構成を示す．2 世代目の T3E のグレードアップ版であり，ハードウェアの殆んどの機能は T3E と全く同一である．しかしながら，CPU のクロック速度が二倍となっており，それに合せてメモリ容量や PE 間通信速度も高速化されている．

プロセッサは，600MHz の Alpha 21164a チップであり，ノード当り 1200MFLOPS, 2400MIPS の演算性能を実現している．また，PE 間通信が高速化されており，通信リンク当り T3E の 480MB/s に対して 650MB/s の通信速度を有している．

ソフトウェアは，若干の改訂はあるものの，T3E のソフトウェアと同一のものとなっている．

2.5 IBM RS/6000 SP

本システムは，データベース処理の研究を目的とした超並列システムであり，nCUBE2, nCUBE3 に続く 3 世代目のデータベース処理研究用超並列システムである．この他に，暗号系の研究を目的とした大規模問題解決教育システムと統合されており，これらは相互に乗り入れることが可能となっている．表 6 に，システムの構成を示す．本システムの特徴は，データベースや暗号処理に多用される整数演算性能に優れる CPU を用いていることと，並列データベース処理を可能とするため，ノードごとにローカルディスクを有していることである．これらのノードを高速スイッチによる相互結合網で結合した分散メモリ型の超並列計算機となっている．システムのノードは 2 つに大別され，データベース処理や暗号の解読などを得意とする PPC604e ノード，および数値計算や高速な I/O 処理を得意とする Power3 ノー

表 6: RS/6000 SP の構成

相互結合網	3D Torus
ノード数	64+4
ノードプロセッサ	PPC 604e (332MHz,664MFlops) ×4 (64 node) Power3 (222MHz,888MFlops) ×8 (4 node)
ローカルメモリ	512MB/node
PE 間通信速度	150MB/s
ノード演算能力/PE 間通信速度	4.42MFlops/(MB/s)

ドから構成されている.

PPC604e ノード部分は，64 ノードが高速スイッチにより結合され，各ノードは，332MHz の PPC604e 4CPU，512M 共有メモリ，ローカルディスクから構成されている．Power3 ノード部分は，PPC604e ノード部分は，4 ノードから成り，各ノードは，222MHz の Power3 8CPU，8GB 共有メモリ，並列ローカルディスクから構成されている.

プロセッサ間通信は，シングルステージ SP スイッチにより行なわれる．相互結合網への接続は，ノード上の PCI バスのインターフェースカードを介して SP スイッチに結合される．PCI バスを介しているため，多様なシステムに対応でき，本システムのように異なる種類のノードの混載システムや，相互結合網部分だけのアップグレードなどが可能となっている反面，通信速度は PCI の速度に制限され，特に通信のレイテンシが大きいという欠点がある.

3 相互結合網の通信速度とシステム性能

3.1 はじめに

本節では，超並列システムの相互結合網の通信速度がシステム全体の性能に与える影響について議論する．PE 間通信速度の影響を調べるためには，CPU やメモリ等の性能が同一で，PE 間通信速度を変化させることが可能な実験システムを用いるのが最も理想的である．しかしながら，実システムを構築するのは困難であり，シミュレーションによる評価では，非常に単純化したベンチマークとなるため，現実のシステム性能を評価することは難しい．情報科学センターでは，幸い T3E および T3E-1200E という，CPU のクロック速度と PE 間通信速度のみが異なる超並列システムを相次いで導入したので，このシステムによって標準的なベンチマークによる評価を行ない，相互結合網の通信速度がシステム全体に及ぼす影響について考察する．

3.2 測定環境

3.2.1 CRAY-T3E システム

評価に用いた CRAY-T3E と CRAY-T3E/1200E について概略を述べる．CRAY-T3E は，CRAY Inc.(旧 CRAY Research) が開発した超並列システムである．これに対し，CRAY-T3E/1200E は，CRAY-T3E の CPU のクロック速度と PE 間通信速度を向上させており，システムの特徴は同一である．CRAY-T3E と CRAY-T3E/1200E に共通する基本的なシステムの特徴は，以下の通りである．各 PE は，1 つの CPU，ローカルメモリ，およびネットワークインターフェースから構成される．CPU は，Alpha 21164(EV5) を用いており，これを CRAY-T3E は 300MHz，CRAY-T3E/1200E は 600MHz で駆動している．一次および二次キャッシュを CPU 内に有しているが，CPU 外には三次キャッシュは持たない．代わりにベクトルパイプランの概念を取り込んだ Stream Buffer と呼ばれる定ストライド (4 段) のパイプライン型 (6 本) のバッファを持ち，二次キャッシュのキャッシュミス時に先読みによるデータ転送を行なう．グローバル・アドレス空間へのデータアクセスのために，E-Register と呼ばれるバッファを有しており，これを用いて全てのデータ (ローカル/リモート) にアクセスできる．例えば，Gather や Scatter 等の操作では実際には通信を行なわず，直接リモートメモリに対して操作が行なわれる．PE 間の相互結合網は，三次元のトーラス構造のネットワークトポロジである．

表 7 に，CRAY-T3E および CRAY-T3E/1200E のシステムの性能を示す．表中，

表 7: CRAY-T3E と CRAY-T3E/1200E の性能

	CRAY-T3E	CRAY-T3E/1200E
CPU (Clock)	EV5 (**300 MHz**)	EV5 (**600 MHz**)
Primary cache size (Bandwidth)	4KB (**4.8 GB/s**)	4KB (**9.6 GB/s**)
Secondary cache size (Bandwidth)	96KB (**4.8 GB/s**)	96KB (**9.6 GB/s**)
Stream Buffer Bandwidth	600MB/s	600MB/s
Main Memory	64 MB	512 MB
Interconnect Bandwidth	**480MB/s**	**650MB/s**

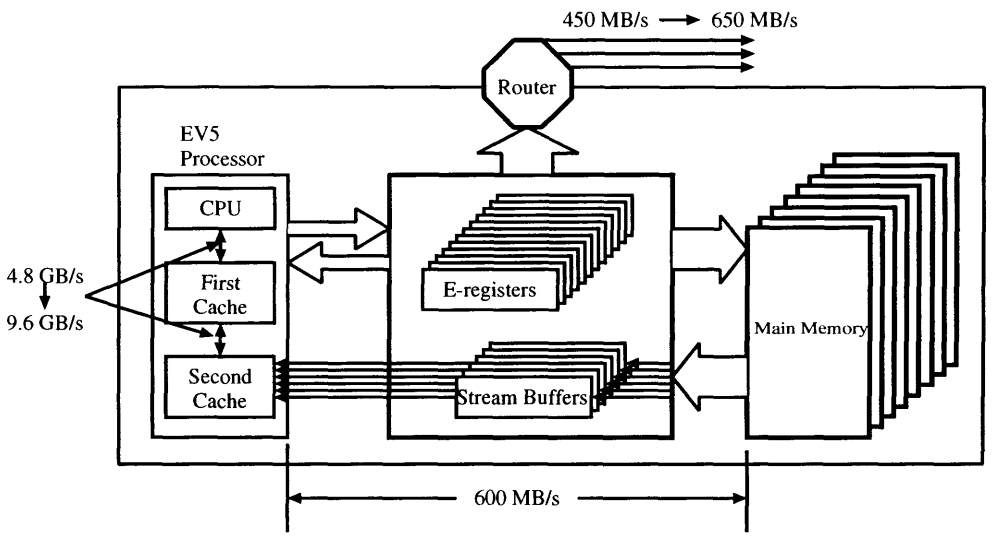


図 2: PE 内の概要および性能

CRAY-T3E と CRAY-T3E/1200E で性能が異なる部分を太字で示してある．また各 PE 内部の概念図を図 2 に示した．CPU のクロック速度向上によって演算性能および CPU の内部キャッシュ速度は 100%向上している．しかし，メインメモリへの帯域等，CPU のチップ外の性能は同一となっている．ネットワーク性能は，通信帯域の増加によって約 35 % 向上した．また，CRAY-T3E システムの通信機構は，PE(CPU) の動作と密接に関連したシステムであるため，通常の PC クラスタ等よりも通信遅延が非常に少ないシステムとなっている．

ソフトウェア環境としては，どちらも Programing Environment Version 3.4 を用いた．これに MPI，Fortran90 および C コンパイラが含まれている．コンパイル・オプションは，全ての場合について "-O3 -dp"，つまり最適化レベルを最高，デフォルトの精度を倍精度の指定とした．

3.2.2 NAS Parallel Benchmark

つぎに，NAS Parallel Benchmark (NPB) について概略を述べる．NPB は，NASA Ames Reseearch Center の NAS(Numerical Aerodynamics Simulation) Lab. で開発された熱流体関連の科学技術計算のベンチマークである．

NPB は，5つの主要アルゴリズムのカーネルと3つの数値流体計算コード(アプリケーションコード：圧縮性流体を擬似的に計算) からなる．それぞれ，並列計算コードと逐次計算コードからなり，並列計算コードは，Fortran 77 と MPI，逐次計算コードは，Fortran 77 で記述されている．また，問題サイズとして4種類用意されており，ベンチマークコードは，問題サイズ(メモリサイズ) の大きさによって CLASS 分けがなされている．評価で用いた CLASS は，問題サイズが小さい順に CLASS W，A，B，C である．

以下に各ベンチマークコードの概要を示す．

カーネル ベンチマーク

CG 正値対称大規模疎行列の最小固有値を共役勾配法により求めるプログラムである．非構造格子を用いた流体アプリケーションで良く用いられる．計算は，多くのメモリ帯域を必要とする．同一長のベクトルデータの通信と，内積を得るための1データの通信が行なわれる．

EP 乗算合同法によって一様な正規乱数を生成するプログラムである．モンテカルロ法でよく用いられ，並列処理では通信がほとんど発生しない．そのため，浮動小数点演算性能のみを示す．

FT FFT を用いて三次元偏微分方程式を解くプログラムである．FFT を各次元毎に解いていくため配列を持ち替える．特に FT は複素数型の配列を MPI_Alltoall により通信するため，通信負荷が非常に大きい．

IS 大規模な整数値ソートを行うプログラムである．粒子法等でよく用いられる．粒子を再び適切なセルに割り当てるためのソートを行なう．MPI_Alltoallv の負荷が通信の多くを占める．このベンチマークのみ C 言語で記述されている．

MG 三次元 Poisson 方程式を Multigrid 法によって求めるプログラムである．非圧縮流体計算中に現れる Poisson 方程式を解くために用いられる．メッセージ長が非一様な通信を行なうが，計算負荷は高くない．

アプリケーション ベンチマーク

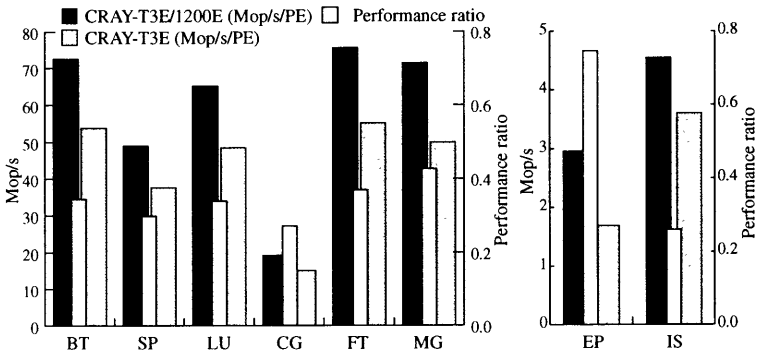


図 3: 1PE における性能 (CLASS W)

BT ブロック 3 重対角方程式を ADI 法を用いて解くプログラムである.

SP 5 重対角方程式をスカラー ADI 法を用いて解くプログラムである.

BT, SP ともに物理量等の格子面データの送受信を行ない, 一部のデータ通信に関して通信隠蔽が行なわれている. 演算量は, CLASS W を除いて, BT が多い.

LU 上下三角行列を対称 SOR 法を用いて解くプログラムである. 並列化にはパイプライン処理が用いられ, 通信処理の隠蔽が行なわれる. CLASS W では最も演算量が多い.

計測結果として, Mop/s(Mega OPeration per Second) 値が得られる. EP と IS 以外の Mop/s 値は, ほぼ MFlop/s と同等の値である. EP の Operation は, 乱数生成数であり, IS の Operation は, 整数演算の数であるため, プログラム中で実行される浮動小数点演算は極めて少ない. このことから, EP と IS に関しての Mop/s 値は, MFlop/s と異なる処理速度の指標として考える必要がある.

3.3 結果と考察

3.3.1 CPU クロック周波数の向上

CPU のクロック速度向上によるシステム性能への寄与について検討する. CPU のクロック速度の差による性能向上を示すために, 1 PE のみを用いて (つまり PE 間通信を行なわないで) CRAY-T3E および CRAY-T3E/1200E 上における並列計算コードの実行速度を評価した. 評価の結果を図 3 に示す. 横軸はベンチマークプログラムの種類, 左縦軸は Mop/s, 右縦軸は次式で定義される性能比 (Performance

ratio) を示した.

$$\text{Performance ratio} = \frac{\text{Mop/s/PE(CRAY-T3E/1200E)}}{\text{Mop/s/PE(CRAY-T3E)}} - 1.0 \qquad (1)$$

CRAY-T3E/1200E は CRAY-T3E の 2 倍のクロック周波数なので，理想状態では，2 倍の性能比 (Performance ratio = 1.0) となるが，メモリなどの動作速度は向上していないため，これよりは小さい性能比となる．CPU における演算量が大きい場合や，キャッシュメモリが有効に動作する場合においては大きな性能比が得られ，メモリへのアクセスが多い場合は，性能比は小さくなることが予想される．

図 3 に示すように，EP は最も大きな性能比が得られた．EP は，メモリアクセスに対して極めて演算量が大きく，CPU のクロック速度向上の効果が現れやすいベンチマークであると言える．このため，PE 単体の性能向上が高い結果となっている．一方，CG と IS は，他のベンチマークと較べて，性能向上が 30 % 以下となり，性能向上が低い．両ベンチマークとも，メインメモリへの広範なアクセスが行なわれる．CRAY-T3E/1200E のメモリ帯域の性能が上がっていないため，CPU のクロック速度向上による寄与が少ないと考えられる．これ以外の場合，FT と MG が 40 % 程度，BT，SP，LU が 35 % 程度の速度向上が得られた．EP を除いた平均的な PE の性能向上は，およそ 35 % である．

3.3.2 PE 間通信性能

本節では，PE 間通信の性能向上を検討する．各ベンチマークの結果を並べるだけでは，システムの性能向上は CPU の性能向上と通信性能が両方含まれた状態としてあらわされる．CPU の性能向上は 3.3.1 節で明らかになっている．その結果を元に，システム全体の性能比 (P_{tmp}) から既知の CPU 性能向上の割合 (P_{CPU}) を相殺することにより，通信性能比 (P_{comm}) の向上を類推することができる．つまり，次式を用いて間接的に推測する．

$$P_{comm} = P_{tmp} - P_{CPU} \qquad (2)$$

P_{CPU} として，各ベンチマークを N 個の PE を用いて並列化すると，PE 当りの演算量が基本的には $1/N$ となる．比較のため，演算量が 3.3.1 節の CLASS W と同程度になる問題サイズと PE 数を用いる．例えば，BT の CLASS W，1 PE で実行する際の演算量は，BT を CLASS A，16 PE で実行する場合の演算量とほぼ同一である．この場合，図 3 に示した CLASS W，1 PE の CPU の性能向上比 P_{CPU} が分かっているので，CLASS A，16 PE による CRAY-T3E と CRAY-T3E/1200E の

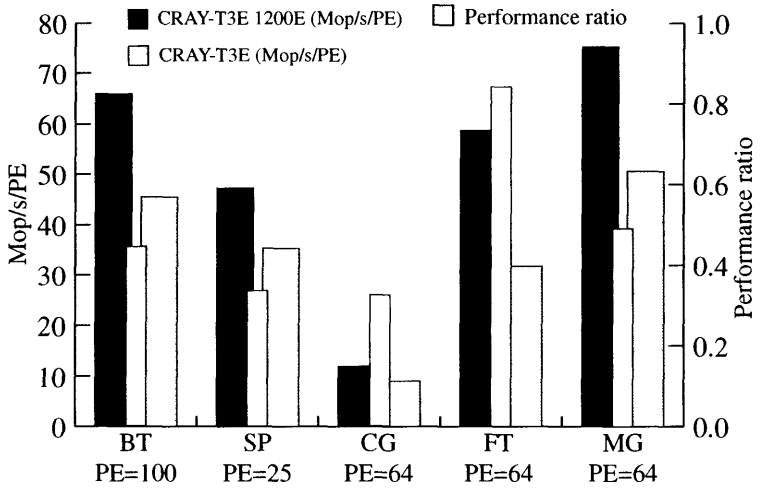


図 4: システム全体の性能比 (CLASS B)

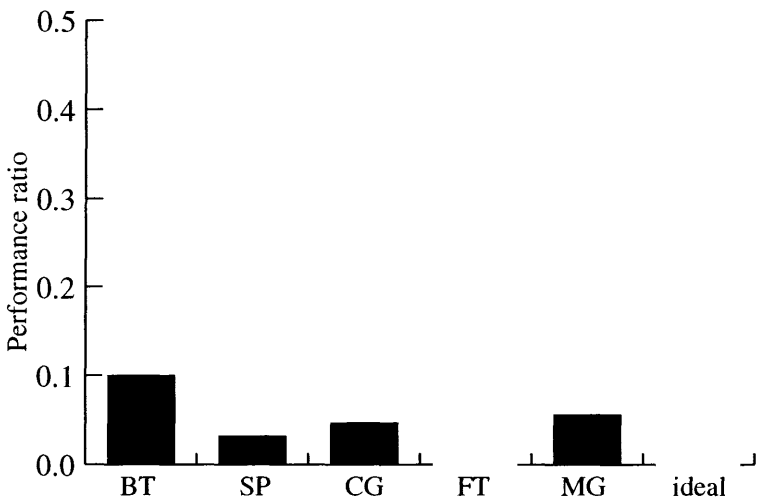


図 5: PE 間通信性能による性能向上率 (CLASS B)

性能比 P_{tmp} を求め，ここから $P_{tmp} - P_{CPU}$ を計算し，間接的に通信性能比 P_{comm} を評価できる．

PE 間通信性能を評価するため，NPB のうちで通信処理の振る舞いが明確なベンチマークプログラムを用いる．LU は，通信隠蔽処理を行なっているため，理論的には通信時間が陽にあらわれない．また，IS も MPI_Alltoallv の挙動が明確でない．また，EP は，並列実行時にも通信が発生しない．よって，LU，IS，EP 以外のベンチマークプログラムを用いる．用いた問題サイズは CLASS B とし，PE 数は CLASS W の 1 PE のメモリワークサイズと同程度の量となるように，ベンチマークごとに調整する．用いた PE 数は，BT で 100 PE，SP で 25 PE，CG で 64 PE，FT で 64 PE，MG で 64 PE である．図 4 に，ベンチマークプログラム

の各 PE 数での性能を示す．横軸に各ベンチマークプログラムと PE 数，左縦軸に Mop/s/PE，右縦軸に Performance ratio を示す．これに式 (2) を適用し，図 4 から図 3 の “1PE での性能比” を差し引いて推測された通信性能比 P_{comm} を図 5 に示す．

図 5 に示されるように，FT の通信性能向上率が最も高い．FT は複素型の大きな配列データを多く通信するため，通信帯域の向上が最も有効に働いたと考えられる．通信性能の向上率は，CRAY-T3E と CRAY-T3E/1200E の通信帯域の性能差に近い値となっている．その他の場合では，およそ 3 ～ 10 % 程度の通信性能向上が見られた．FT 以外では，通信回数が多いため，通信性能の多くは通信帯域ではなくネットワークのレイテンシーに依存するものと考えられる．SP, CG, および MG の結果より，平均的なアプリケーションプログラムを実行した場合の通信性能の向上は，およそ 5 % 前後と推測される．

3.4 まとめ

本節では，本学 情報科学センターの T3E および T3E-1200E を用いて，超並列システムの相互結合網の通信速度がシステム全体の性能に与える影響について議論した．実用的なアプリケーションによる通信性能の影響を調べることは難しいため，T3E および T3E-1200E において標準的な並列ベンチマークプログラムである NASA Parallel Benchmark を用いて測定を行ない，システム全体の性能向上から CPU のクロック速度向上による処理速度の改善を差し引くことにより，通信速度がシステムの性能に与える影響を推測した．

評価実験の結果，大きな配列データを相互に交換する FFT などのアプリケーションでは，CPU 性能よりも PE 間通信性能が律速であり，両システムの PE 間通信速度の差にかなり近い性能の差が現われた．平均的なアプリケーションでは，通信のバンド幅よりも，むしろ通信のレイテンシーに大きく依存することが推測された．

以上のことから，次世代の超高速超並列システムでは，非常に低レイテンシーの高速 PE 間通信を行なう必要があることが分かった．

4 相互結合網のトポロジ

4.1 はじめに

本章では，科学技術計算に適する相互結合網のトポロジについて議論する．

科学技術計算の多くは2次元または3次元構造のデータを対象とするため，格子型の結合網と適合しやすい．しかし，格子結合網は，システムの規模が大きくなると通信性能が急速に低下するという問題がある．そこで，ノード数の増加に応じて1ノード当りのリンク数(次数)を増加させるハイパーキューブ網が提案され，nCUBEなどの中規模並列処理システムに実装されてきた．ハイパーキューブ網は，平均距離などの通信性能に優れ，格子結合網をはじめ様々な他の結合網を内包できる利点を持つ．しかし，大規模なシステムではノードの次数が非常に大きくなり，実装が困難になるという問題点がある．

結合網を実装性の観点から考えると，配線密度によるチップ面積や配線の容易さから格子結合網は重要性を増している．また，格子結合網は故障回避に関する多くの研究がなされ，実装性に関して優れた性能を有している [4]．そこで，近年では，超並列計算機用の相互結合網として，格子結合網やトーラス網を基本として，遠距離ノード間の結合を付加した多くの相互結合網が提案されている [5, 6, 7]．

RDT網 [5] は，基本トーラス網に対して，グリッド間隔を $\sqrt{2}n$(n は整数) 倍し 45^o 傾むけて再帰的に上位のトーラスを構成する相互結合網である．RDTは，階層的な構造を持ち，直径も比較的小さく，木構造を内包し同報通信などが容易に行える利点を持つ．しかしながらRDT網は，斜め方向の配線を含むため，故障回避が難しくなるなどの問題がある．

2次元のPacked Expornential Connections (PEC) 網 [6] は，基本格子網に対して，グリッド間隔を 2^n(n は整数) 倍した格子を，互いに重ならないように再帰的に重ね合せて構成する相互結合網である．2次元格子網上で，グリッド間隔が大きな格子網を用いることにより木状網及びハイパーキューブ網のエミュレーションが可能である．しかしながら，2次元PEC網は表を用いて定義を行なっており，直径やルーティングアルゴリズムも明らかではない．2次元PEC網は，1次元PEC網 [7] を組み合わせて得られる．1次元PEC網については，1次元PEC網の直径と，特定のノードからのルーティングアルゴリズム，及びルーティングのステップ数の上界が示されているが，任意のノード間の効率的なルーティングについては未解決である．

本章では，トーラス結合網を基に，ノード間距離の異なるバイパスリンクを再帰的に付加したShifted Recursive Torus (SRT) 網 [8, 9, 10, 11, 12] について，ネット

ワーク性能について検討する. SRT は 1 次元から定義でき, 2 次元への拡張が容易である. ここでは, 1 次元 SRT の構成法を示し, ネットワーク直径を求める. 1 次元の SRT と PEC 網は類似した結合網である. しかしながら, 1 次元 PEC 網のルーティングでは, 経由リンクを動的計画法によって求めているのに対し, 1 次元 SRT ではノード間距離によって一義的に決定する手法を示す. 第 3 章で 1 次元 SRT を 2 次元に拡張し, 表を用いない式による定義を示す. また, 直径やルーティングアルゴリズムを示す. 2 次元 SRT は階層的なトーラスだけで構成でき, 斜め方向のリンクなどを持たない簡単な構造を持ちながら, 従来の結合網と比較して遜色ないネットワーク性能を持つことを明かにする.

4.2 1 次元 SRT

4.2.1 構成原理

1 次元 SRT は環状網を基に構成される. 通信性能向上のため, 遠隔ノードと直接接続されるリンクを, 各ノードの次数が一定となるように, 環状網に重ね合わせる. ノード数 $N(N = 2^n)$ から成る環状網で, あるノードを番号 0 とし, ノードを昇順に番号付け, 次の基本トーラスを定義する.

定義 1 (基本トーラス) N ノードから成る基本トーラスのノード $x(0 \leq x < N)$ は, 隣接する左右のノード $(x \pm 1)\ mod\ N$ と結合される. □

トーラスの両端は互いに結合されるので, ノード 0 の隣接ノードは 1 と $N-1$ である. 基本トーラスを構成する結合リンクをレベル 0 のリンクと呼ぶ.

トーラス結合網の直径および平均距離を短縮するために, 基本トーラスにバイパスリンクを付加し, 上位トーラスを構成する.

1D-SRT の考え方は次の通りである. 基本トーラスから, $x_1\ mod\ 2 = 1$ を満すノード群 x_1 を取り出す. このノード群をレベル 1 のノードと呼ぶ. レベル 1 のノードをバイパスリンクによって環状に結合する. 同様に基本トーラスから, $x_2\ mod\ 4 = 2$ を満すレベル 2 のノード群 x_2 を取り出す. レベル 2 のノードをバイパスリンクによって環状に結合する.

レベル 3 以上のリンクも, 同様の手続きを再帰的に行なうことによって, 定義 2 が定まる.

定義 2 (1D-SRT) $N = 2^n$ ノードから成る基本トーラスから, ノード番号が

$$x_l\ mod\ 2^l \quad = \quad 2^{l-1} \tag{3}$$

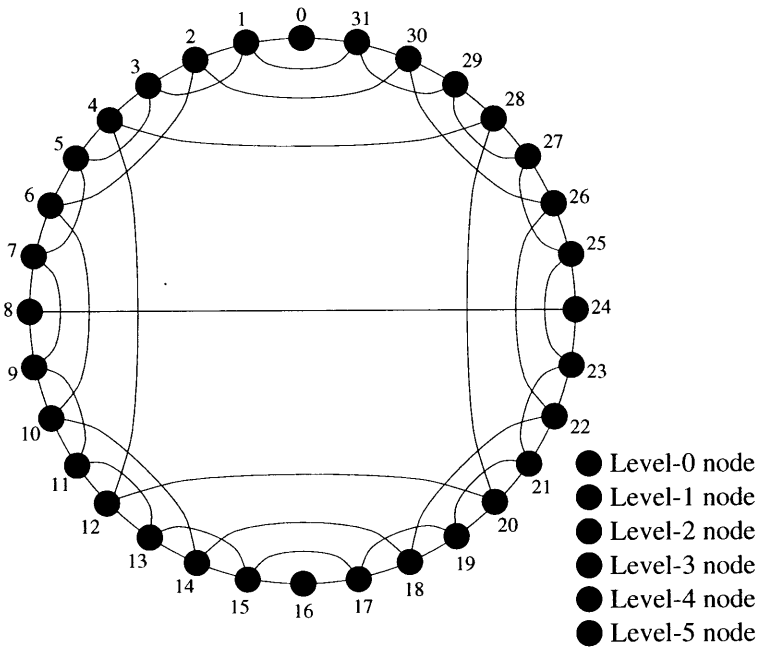


図 6: 32 ノードから成る基本型 1D-SRT.

を満すノード群 x_l を取り出し, 環状に接続する. この手続きをレベル l が 1 から $l_{max}-1$ まで繰り返す. l_{max} は, 基本型 *1D-SRT* の最大のレベルであり, $l_{max} = \log_2 N = n$ である. □

式 (3) を満すノード群をレベル l のノードと呼び, V_l とする. $l \geq 1$ のレベルを上位レベルと呼ぶ. ノード 0 は, どのような $l \geq 1$ でも式 (3) を満さないので, 上位レベルを持たない. このため, ノード 0 のレベルは 0 とする. V_l を環状に接続するリンクを上位リンクと呼び, E_l とする. x_l はレベル 0 のリンクとは別に, レベル l のリンクによって 2^l 離れた同じレベル数のノード $((x_l \pm 2^l)\ mod\ N)$ に接続される. 従って x_l は,

$$(x_l \pm 1)\ mod\ N, \quad (x_l \pm 2^l)\ mod\ N$$

の 4 ノードに接続される.

このように決定された 32 ノードから成る 1D-SRT の結合の様子を図 6 に示す. ノードはレベル毎に 2 本のリンクを持つため, 1D-SRT のどのノードも, 基本トーラス (レベル 0) を構成する 2 本と, 上位リンクを構成する 2 本の合計 4 本のリンクを持つことが分る. 但しノード $0, N/2$ は上位レベルが割り当てられないため, 上位リンクが無く, レベルは 0 となる. またノード $\frac{1}{4}N, \frac{3}{4}N$ に接続される上位リンクは 1 つであり, レベルは l_{max} である.

4.2.2 1D-SRT の直径

本節では, 最初に, 上位レベルの階層が下位の階層を包含することを示す. 直径は最上位の階層の, 最も離れたノード間の距離と考えれば良い. 包含関係を示すために, 上位ノードに対する下位ノードの位置を求め, これが上位ノードのレベルに依存しないことを示す. ここで位置とは, レベル 0 トーラスでのノード番号の差とする. これを用いて, レベルが異なる最近隣ノード間の距離 (レベル間距離) が, 下位ノードのレベルによって決定されることを示す. このことと, 上位の階層が下位の階層を包含することから, 1D-SRT の直径をレベル間距離によって表す. 更にレベル間距離を求め, 1D-SRT の直径を求める.

上位レベルのノードに対する下位ノードの位置が, 上位ノードの位置に依存しないことを示す. つまり, 図 6 において, レベル 1 のノードは, レベル 3 のノード 4 から $2^l \cdot i + 2^{l-1}$ 離れた位置にあるが, 同じレベルのノード 12 からも, 同様に $2^l \cdot j + 2^{l-1}$ 離れた位置に存在する.

補題 1 (上位レベルに対する下位レベルの位置) V_k から任意のノード $x_k(i)$ を取り出す. $x_k(i)$ に対する下位レベルのノード $x_l \in V_l$ の位置は, $k > l$ である限り, $x_k(i)$ の位置とレベルに依存しない.

証明 $x_k(i)$ に対するノード $x_l(j)$ の位置を, ノード番号の差とする. 定義 2 より, レベル k, l のノード番号は,

$$x_k(i) = 2^k \cdot i + 2^{k-1} \tag{4}$$

$$x_l(j) = 2^l \cdot j + 2^{l-1} \tag{5}$$

と書ける. ここで i, j はそれぞれレベル k, l のノードのインデックスである. $x_k(i)$ から見た $x_l(j)$ の相対的なノード番号は,

$$x_l(j) - x_k(i) = \left(2^l \cdot j + 2^{l-1}\right) - \left(2^k \cdot i + 2^{k-1}\right)$$

と書ける. 上式を定義式 (3) に当てはめると,

$$(x_l(j) - x_k(i)) \ mod\ 2^l = 2^{l-1} \qquad (k > l) \tag{6}$$

が成立する. 式 (6) は, $k > l$ であればどのレベル k でも成立するので, ノード $x_k(i)$ に対する $x_l(j)$ の位置は, $x_k(i)$ のレベル及びノード番号に依存しない. □

このことから, 次の系が言える.

系 1 任意のノード $x_k(i) \in V_k$ から, $x_k(i)$ に最も近いノード $x_l(j) \in V_l$ までの, ノードレベルの並びを $s(k,l)$ とする *(*$k > l$*)*. すると

$$s(k,l) = s(k+m,l) \qquad (0 < m \leq l_{max} - k)$$

となる. □

つまり,

$$\begin{aligned} s(2,1) &= s(3,1) = \ldots = s(l_{max},1) = s(0,1) \\ s(3,2) &= \ldots = s(l_{max},2) = s(0,2) \\ &\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ & s(l_{max}, l_{max}-1) = s(0, l_{max}-1) \end{aligned}$$

のように書くことができる. 従って次の系が言える.

系 2 任意のノード $x_k(i) \in V_k$ から, $x_k(i)$ に最も近いノード $x_l(j) \in V_l$ までの距離を $d(k,l)$ とする *(*$k > l$*)*. すると

$$d(k,l) = d(k+m,l) \qquad (0 < m \leq l_{max} - k)$$

となる. 距離とは, レベル *0* リンクや上位リンクを用いた $x_k(i)$ から $x_l(j)$ までの経路のホップ数である. □

$d(k,l)$ は, ルーティング時にノードのレベルを乗り換える (レベル k からレベル l のリンクへ乗り換える) 時の所要ホップ数に等しい. 系 2 から, 全ての k,l $(k > l)$ について, $d(k,l)$ は $d(0,l)$ が求まれば十分である. そこで, $d(k,l)$ を $d_0(l)$ と表記し, これをレベル間の距離と定義する.

補題 1 により, 下位レベルのノード配置は上位レベルのノード配置に包含されるので, 1D-SRT の直径は, 上位のノードを持たないノード 0 とこれから最も離れたノード $N/2$ までの距離となる. ノード 0 と $N/2$ までの距離は $d(l_{max}+1, 0)$ に等しい.

定理 1 ノード数 N から成る *1D-SRT* の直径 $D(N)$ は,

$$D(N) \equiv d(l_{max}, 0) \equiv d_0(\log_2 N). \tag{7}$$

次に, レベル間の距離 $d_0(l)$ を具体的に計算する. レベル $l-c$ のリンクを通過するという条件のもとでの, ノード 0 から (基本トーラス上で) 最近隣のレベル l ノードまでの距離は,

$$\begin{aligned} d_0^c(l) &= \left[2d_0(l-c) + (2^{c-1}-1)\right] \\ & (1 \leq c \leq l) \end{aligned} \tag{8}$$

と表わすことができる. ルートは, ホップ数 $d_0(l-c)$ でノード0から, ノード0に最近隣のレベル $l-c$ ノードに到達し, そこからレベル $l-c$ のリンクを $2^{c-1}-1$ ホップ伝わり, 最後に $d_0(l-c)$ のホップ数でノード l ノードに到達する. ノード間の距離 $d_0(l)$ は, c を変化させた時の $d_0^c(l)$ の最小値である. c は $2 \leq c \leq l-1$ の整数であるが, この中で, $d_0^c(l)$ を最小とする c は,

$$c \sim \frac{1}{2}\left\{\sqrt{8l+1}-1\right\} \tag{9}$$

である. c は整数なので, 次の補題の様にレベル間の距離を示す.

補題 2 (レベル間の距離) レベル間の距離 $d_0(l)$ は,

$$d_0(l) = 2^{c_f-1}(c_f-1+s_f)+1 \tag{10}$$

ここで

$$c_f = \left\lfloor \frac{1}{2}\left\{\sqrt{8l+1}-1\right\} \right\rfloor \tag{11}$$

$$s_f = l - \frac{1}{2}c_f(c_f+1) \quad (0 \leq s_f \leq c_f) \tag{12}$$

この時, 経路はレベル $l-c$ のリンクを通過する. □

補題2では c を切捨てて整数にしたが, 同様に切上げて整数化し, $d_0(l)$ を求めることもできる. この場合,

$$d_0(l) = 2^{c_c-2}(2(c_c-1)+s_c)+1 \tag{13}$$

ここで

$$c_c = \left\lceil \frac{1}{2}\left\{\sqrt{8l+1}-1\right\} \right\rceil \tag{14}$$

$$s_c = l - \frac{1}{2}c_c(c_c+1) \quad (0 \leq -s_c < c_c) \tag{15}$$

となる.

表8に1次元SRTの レベル l, c, 経由レベル $l-c$, レベル間距離 $d_0(l)(=$ 直径$)$ の関係を示す.

定理1より, ノード数 $N=2^n$ から成る1D-SRTの直径 $D(N)$ は, $l=\log_2 N$ として表8の $d_0(l)$ の項となる. 1D-SRTの直径のオーダーは, 補題2より,

$$D(N) = d_0(\log_2 N) \sim 2^{\sqrt{2\log_2 N}}\sqrt{2\log_2 N}. \tag{16}$$

従って1D-SRTの直径のオーダーは,
$\mathcal{O}\left(2^{\sqrt{2\log_2 N}}\sqrt{\log_2 N}\right)$ である.

表 8: レベル l, c, 経由レベル $l-c$, レベル間距離 $d_0(l)$(直径) の関係

l	c_f	s_f	$l-c_f$	c_c	s_c	$l-c_c$	$d_0(l)$
1							1
2							2
3	2	0	1	2	0	1	3
4	2	1	2	3	-2	1	5
5	2	2	3	3	-1	2	7
6	3	0	3	3	0	3	9
7	3	1	4	4	-3	3	13
8	3	2	5	4	-2	4	17
9	3	3	6	4	-1	5	21
10	4	0	6	4	0	6	25
11	4	1	7	5	-4	6	33
12	4	2	8	5	-3	7	41
13	4	3	9	5	-2	8	49
14	4	4	10	5	-1	9	57
15	5	0	10	5	0	10	65
16	5	1	11	6	-5	10	81

4.2.3 ルーティングアルゴリズム

1D-SRT のルーティングアルゴリズムを図 7 に示す. ルーティングの始点ノードを x_s, 終点ノードを x_d とする. 関数 $FindMLevel$ によって, ルーティングに使用するリンクの最大レベル l_r を求め, 関数 $FindNearestNodes$ で, このレベルのノード x_s^r, x_d^r を探す. 始点ノード x_s と x_s^r, x_d^r 終点 x_d との間で, 手続きを再帰的に呼び出し, 経路を求める. この方針に基づくルーティングを再帰ルーティングと呼ぶ. 関数 $RecursiveRouting$ によるルーティングの結果は, 経路が通過するノードリストとして得られ, これを経路リストと呼ぶ. 32 ノードから成る 1D-SRT における, ノード 0 ($x_s = 0$) からノード 15 ($x_d = 15$) までのルーティング概念を, 図 8 に示す. 以下, 図 8 を例にして, ルーティングについて詳細に述べる.

手順 1. 最大レベルのリンクの決定 ルーティングで使用するリンクのうち, 最大のレベル l_r を決定する. $FindMLevel$ で, 補題 2 に基づき, レベル l_r を計算する. つまり, $l = \log_2 |x_d - x_s|$ から, 式 (11) により c_f を計算し, $l_r = l - c_f$ を

```
RecursiveRouting(x_s, x_d){
  if( x_s = x_d )      return( φ );
  if( x_s ≤ x_d )      dir = +1;
  else                 dir = −1;
  l_r = FindMedLevel(x_s, x_d);
  (x_s^r, x_d^r) = FindNearestNodes(l_r, x_s, x_d);
  l_u = FindUprLevel(x_s, x_d);
  (x_s^u, x_d^u) = FindNearestNodes(l_u, x_s, x_d);
  if( |x_s − x_s^u| + |x_d − x_d^u| < |x_s − x_s^r| + |x_d − x_d^r| ){
    l_r = l_u;
    x_s^r = x_s^u;      x_d^r = x_d^u;
  }
  RoutingList = RecursiveRouting(x_s, x_s^r);
  while( x_s^r ≠ x_d^r ){
    addlist(RoutingList, x_s^r);
    x_s^r = x_s^r + dir * 2^{l_r};
  }
  addlist(RoutingList, RecursiveRouting(x_d^r, x_d));
  return( RoutingList );
}
```

図 7: 1 次元 SRT のルーティングアルゴリズム.

求める. 図 8 のノード 0 から 15 への経路では, $\log_2|15-0|$ を切り上げ, $l=5$ を得る. 式 (11) より, $c_f=2$ を計算して, $l_r=l-c_f=3$ を得る.

手順 2. バイパスリンクの最近隣端点の探索 *FindNearestNodes* で, レベル l_r のノードのうち, x_s, x_d に最も近いノードを探索する. ノード番号が x より大きい, x に最近隣のレベル l のノード番号は次式で与えられる.

$$nlnear\_f(x,l)=\left\lceil \frac{x-2^{l-1}}{2^l} \right\rceil \cdot 2^l + 2^{l-1} \tag{17}$$

同様に, ノード番号が x より小さい, x に最近隣のレベル l のノード番号は次

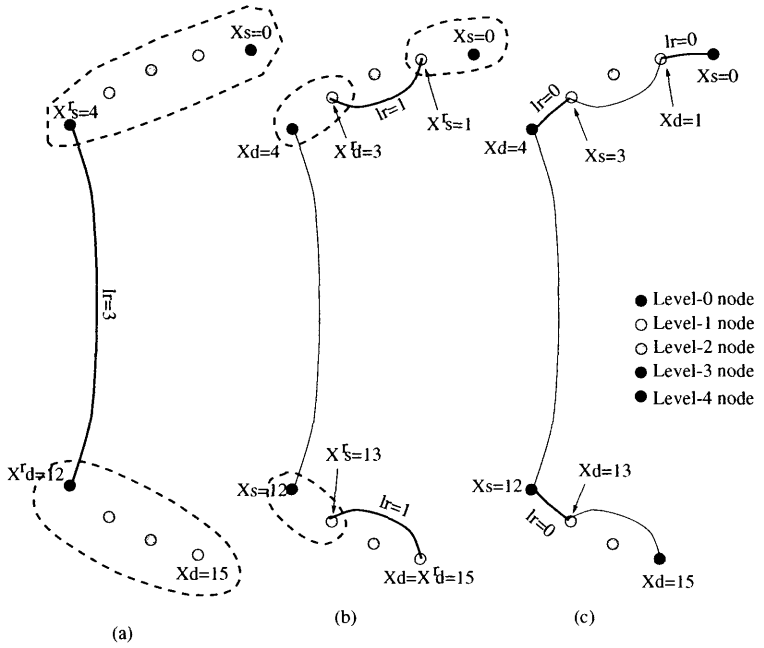


図 8: 1 次元 SRT のルーティングの概念図

式で求まる.

$$nlnear\_b(x,l) = \left\lfloor \frac{x - 2^{l-1}}{2^l} \right\rfloor \cdot 2^l + 2^{l-1} \tag{18}$$

最近隣ノードの探索時, x_s に最近隣のレベル l_r のノードは, x_d 側だけに限られない. そこで, 通信相手と反対側の逆方向のレベル l_r のノードも探索し, より近いノードを決定する. 決定された x_s 側の最近隣ノードを x_s^r, x_d 側の最近隣ノードを x_d^r とする. 図 8(a) で, $l_r = 3$ なので, $x_s^r = 4$, $x_d^r = 12$ となる.

手順 3. l_r **の補正**

(手順 3-1) 手順 1 によるレベルの決定は, ノードの位置を考慮していないため, 上位レベルのリンクを使用した経路の方が明かに短い場合でも, 上位レベルのリンクを使用しないという問題点がある. 例えば図 6 でノード 8 から 24 へのルーティングの場合, 式 (11) を用いたレベル決定では $l_r = 3$ となるが, 実際には $l_r = 4$ のリンクを使用した方が良い. そこで, $FindUprLevel$ で x_s から x_d 間のリンク中の最大のレベル l_u を取り出す. l_u は,

$$l_u \;=\; log_2(|x_d - x_s|);$$

と計算できる. 例では $l_u = 3$ である.

(手順 3-2) 手順 2 と同様に, レベル l_u のノードのうち, x_s, x_d に最も近いノードを探索する. 例では, $l_u = 3$ なので, $x_s^u = 4$, $x_d^u = 12$ となる.

(手順 3-3) l_r をそのまま使用するか, l_u を使用した経路にするかを決定する. x_s^r, x_d^r よりも x_s^u, x_d^u の方が x_s, x_d に近い場合は, レベル l_r の代りに l_u を使用する. 例では, $l_r = l_u$ なので, $l_r = 3$ を使用する.

手順 4. 低位リンクに対する再帰ルーティング x_s と x_s^r 間で, $x_s = x_s^r$ となるまで再帰的に *RecursiveRouting* を呼び出し, x_s, x_s^r 間の経路リストを得る.

次に x_s^r から x_d^r 間にあるレベル l_r のノードのリストを経路リストに加える. ここでは経路リストに $\{4, 12\}$ が加えられる. x_d^r と x_d 間の経路も x_s, x_s^r 間と同様に求め, 経路リストの末尾に付加する

図 8(b) のように, 2 回目の再帰呼び出し時に, 前回不明だったノード 0 から 4 の間の経路を求める. $x_s = 0, x_d = 4$ として, $l_r = 1$, 経路リスト $\{1, 2\}$ を得る. 更に図 8(c) に示すように, 3 回目の再帰呼び出しで, $x_s = 0, x_d = 1$ として, $l_r = 0$, 経路リスト $\{0, 1\}$, 及び $x_s = 3, x_d = 4$ として, $l_r = 0$, 経路リスト $\{3, 4\}$ を得る. 2,3 回目の経路リストを継げ, 現在の経路リストの先頭に加える ($\{0, 1, 3, 4, 12\}$). 同様にノード 12 から 15 についての経路 ($\{12, 13, 15\}$) を, 経路リストの末尾に追加する.

1D-SRT はトーラス結合を基本としているため, $|x_d - x_s| > 2^{l_{max}-1}$ の場合には, x_s から逆方向に経路を求める.

4.2.4 1D-SRT のネットワーク性能

1D-SRT の平均距離を表 9 に示す. 表中 Optimal Routing は可能な経路を全探索し, 最短経路を使用するルーティング方式である. これに対し, 1D-SRT の再帰ルーティングでは 10%前後平均距離が長くなる. この平均距離の増加は, ノード数が大きい場合に大きくなる. ノード数が大きくなると, 距離を短縮する上位レベル l_u と補題 2 で計算されるレベル l_r の差が大きくなる. レベル l のノード数は, $N/2^l$ なので, 距離の短縮に有効な l_u のノードの数が l_r のノードの数に比べて少なくなる. このため, レベル l_u のリンクを使用しにくくなり, 距離の短縮が有効に行なわれない. 再帰ルーティングによる直径は, 1D-SRT の理論直径である $D(N) \equiv d_0(\log_2 N)$ に一致する.

表 10 に 1D-SRT や他の結合網の直径と次数を示す. Chordal Ring は, 次数が小さくノード数が少ない時は 1D-SRT よりも直径が小さい. しかしノード数が多くなっ

表 9: 1 次元 SRT の平均距離

Number of node	2^8	2^{10}	2^{12}	2^{14}
Optimal Routing	7.0	11.5	17.7	-
Recursive Routing	7.4	12.4	20.0	30.2

表 10: 1 次元 SRT と他の結合網の直径 (次数) の比較

Number of node	2^8	2^{10}	2^{12}	2^{14}	2^{16}
1D-SRT	17(4)	25(4)	41(4)	57(4)	81(4)
Chordal Ring	15(3)	31(3)	63(3)	127(3)	255(3)
Barrel Shifter	4(15)	5(19)	6(23)	6(27)	8(31)

てくると, 急速に直径が増大する. Barrel Shifter はノード数が多い場合でも直径が非常に小さいが, 次数が $\mathcal{O}(\log_2 N)$ であり, ノード数が大きくなるとノードの次数が非常に大きくなり, 実装性が低下する. 1D-SRT は次数が Chordal Ring とあまり変らないにもかかわらず, ノード数が増加しても直径が急激に増加しない. 1D-SRT は, 階層構造により次数や網の構成を変化させることなくノード数を増加できるので, スケーラビリティに優れる. 1 次元 PEC 網は, 1D-SRT がトーラス結合を基にしているのに対し, 線型結合を基にしているので, 直径は 1D-SRT よりも 1 段大きくなる.

4.3 2 次元 SRT

4.3.1 2D-SRT への拡張

本章では, 1 次元 SRT を 2 次元に拡張した 2D-SRT について詳しく議論する. $N \times N (N = 2^n)$ のノードから成る 2D-SRT は, x 方向に直線状に配置した N ノードから成る 1D-SRT を y 方向に積み重ねることにより構成される. 2D-SRT は, x 方向とともに, y 方向にも 1D-SRT が構成可能でなくてはならない. そこで, 積み重ねる段ごとに, 1D-SRT の原点ノード (ノード 0) の位置をシフトする必要がある. シフトの幅を変化させることにより, 様々な 2D-SRT が構成可能である. 最初に 2D-SRT の一般形について定義する.

定義 3 $N \times N(N = 2^n)$ のノードから成るトーラス上で, アドレスを左下から順に与え, (x, y) と表記する. *1* 次元 *SRT* と同様に, トーラス上のノードに上位レベルを割り当てる. レベル l のノード (x_l, y_l) は次の式を満すノードである.

$$(x_l + s_x \cdot y_l) \ mod \ 2^l \ = \ 2^{l-1} \tag{19}$$

ここで s_x は x 方向のシフト幅である. ノード (x_l, y_l) は隣接する *4* ノード $((x_l \pm 1) \ mod \ N, (y_l \pm 1) \ mod \ N)$ と, 2^l 離れた *4* ノード $((x_l \pm 2^l) \ mod \ N, (y_l \pm 2^l) \ mod \ N)$ と接続される. □

x 方向に 1D-SRT を構成すると同時に, y 方向にも 1D-SRT が構成されるためには, シフト幅 s_x が以下の条件を満す必要がある.

1. 原点ノードが各行・列に 1 つだけ存在する
2. x と y が転置関係にある式が定義できること.
   つまり, 式 (19) に対して,
   $(y_l + s_y \cdot x_l) \ mod \ 2^l = 2^{l-1}$ が得られる.

原点ノードとは, 1D-SRT のノード番号が 0 のノードのことである.

各レベルの通信特性を考えると, レベルが低いノードは近距離の通信に適し, レベルが高いノードは遠距離の通信に適している. 従って, 全体の通信性能を向上するためには, レベルが高いノードを平面内に均等に配置することが有効である. $N \times N$ ノードから成る 2D-SRT は, N 個の 1D-SRT を積み重ねて構成されるため, N 個の原点ノードを持つ. この N 個のノードを $N \times N$ の平面内に均等に割り当てると, 原点ノード同士は間隔が $\sqrt{N}$ の格子状に配置される. この格子配置に基づいて, 定義 3 の条件を満たすために, y 方向に進むに従い格子から x 方向に 1 列ずらした千鳥型 2D-SRT を定義する.

定義 4 (千鳥型 2D-SRT) 千鳥型 *2D-SRT* は, 一般型 *2D-SRT* で

$$s_x = \pm \left(2^{\lceil (l_{max}-1)/2 \rceil} \pm 1\right)$$

の時の構成である. 複号は独立である. □

千鳥型 2D-SRT では, 符号によって s_x が 4 通りあり, 式 (19) に当てはめると, 次の 4 つの式は, どれもが千鳥型 2D-SRT の定義式となり得る.

$$\left(x_l - \left(2^{\lceil (l_{max}-1)/2 \rceil} + 1\right) y_l\right) \ mod \ 2^l \ = \ 2^{l-1} \tag{20}$$

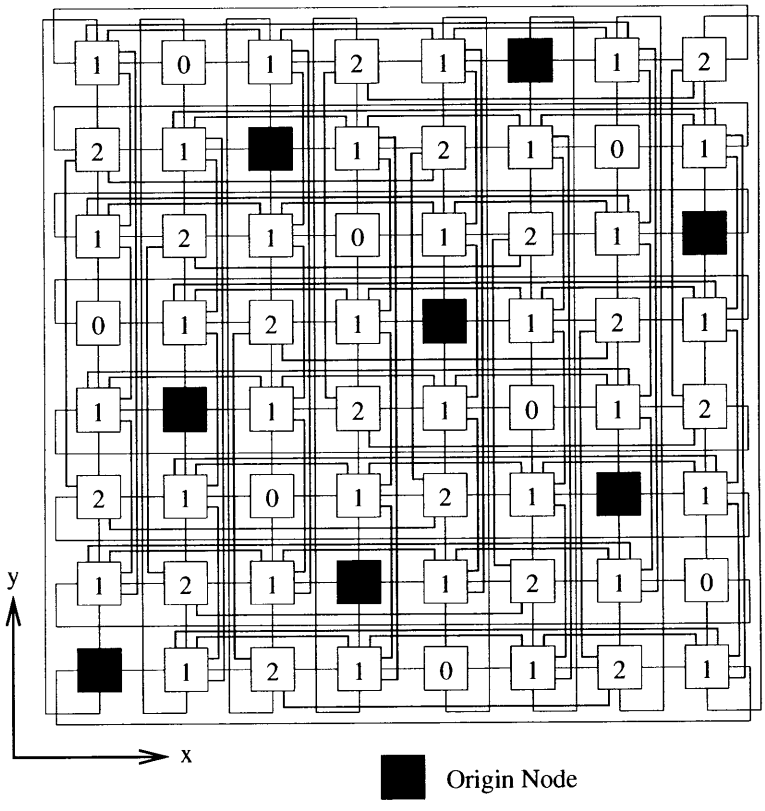


図 9: 8×8 ノードから成る千鳥型 2D-SRT.

$$\left(x_l + \left(2^{\lceil (l_{max}-1)/2 \rceil} - 1\right) y_l\right) \ mod\ 2^l \ = \ 2^{l-1} \tag{21}$$

$$\left(x_l - \left(2^{\lceil (l_{max}-1)/2 \rceil} - 1\right) y_l\right) \ mod\ 2^l \ = \ 2^{l-1} \tag{22}$$

$$\left(x_l + \left(2^{\lceil (l_{max}-1)/2 \rceil} + 1\right) y_l\right) \ mod\ 2^l \ = \ 2^{l-1} \tag{23}$$

ここで, $l_{max} = n$ である.

ノード (x_l, y_l) は, 隣接する 4 ノード $((x_l \pm 1)\ mod\ N, (y_l \pm 1)\ mod\ N)$ と, 2^l 離れた 4 ノード $((x_l \pm 2^l)\ mod\ N, (y_l \pm 2^l)\ mod\ N)$ に接続される.

式 (20) から式 (23) により定義された SRT は, x 方向に関して 1D-SRT を形成することは明らかである. また, それぞれ転置関係にある式が定義できるので, y 方向に関しても 1D-SRT を形成する. 式 (20) と式 (21), 式 (22) と式 (23) は転置の関係, 式 (20) と式 (23), 式 (21) と式 (22) は対称の関係となる. 図 9 に, 8×8 ノードから成る式 (22) に基づく千鳥型 2D-SRT のレベル配置と結合の様子を示す. 図中の数字はノードのレベル番号を示す.

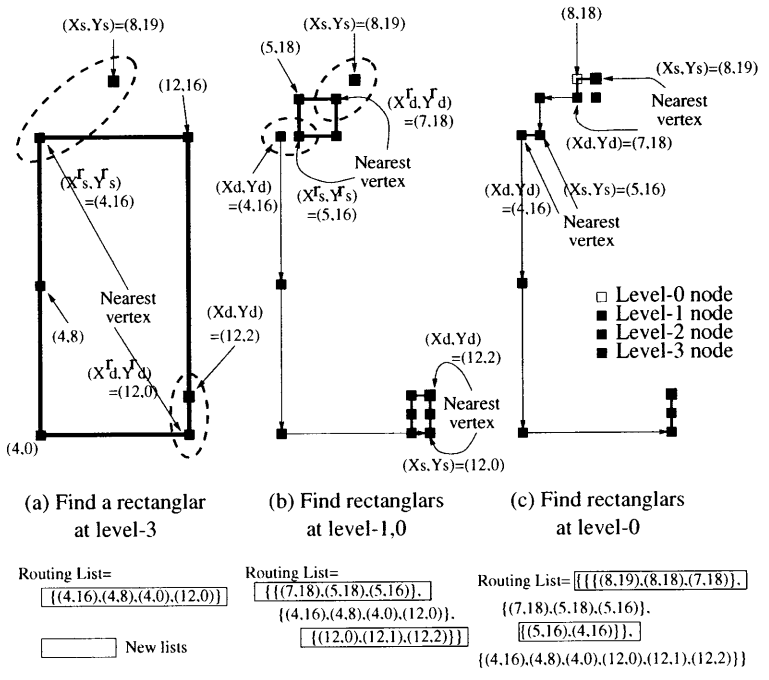


図 10: 2D-SRT のルーティングの概念図.

4.3.2 2D-SRT のルーティング

2次元SRTのルーティングの基本方針は, 1D-SRTのルーティングと同様に, ルーティングに使用する最大のレベルを求め, このレベルのノードと始点・終点ノードとの間で再帰的に手続きを繰り返し経路のリストを作成する. 始点ノードを (x_s, y_s), 終点ノードを (x_d, y_d) として, 経路リストを作成する. 2次元SRTのルーティングの概念図を図10に示す. 図に示した $(8,19) \rightarrow (12,2)$ へのルーティングを例として, 以下に手続きの流れについて述べる.

手順1. 最大レベルのリンクの決定 ルーティングで使用するリンクのうち, 最大のレベル l_r を求める. $x_s \rightarrow x_d$ と $y_s \rightarrow y_d$ について, それぞれ1D-SRTと同様にレベルを求め, 大きいレベルを l_r とする. 図10では, $x_s = 8 \rightarrow x_d = 12$ から $l_{rx} = 2$, $y_s = 19 \rightarrow y_d = 2$ から $l_{ry} = 3$ となるので, y 方向のレベルが大きく, $l_r = 3$ とする.

手順2. バイパスリンクの最近隣端点の探索 (x_s, y_s), (x_d, y_d) に近い, レベル l_r のノードを探索する. 複数あるレベル l_r トーラスのノードのうち, (x_s, y_s), (x_d, y_d) に最も近いノードを基準とする. このレベル l_r トーラスに含まれるノードのうち, (x_s, y_s) の最近隣ノードを (x_s^r, y_s^r), (x_d, y_d) の最近隣ノー

ドを (x_d^r, y_d^r) とする．図 10(a) では，$l_r = 3$(グリッド間隔が 8) のトーラスのうち，(x_d, y_d) に近い $(12,0)$ を基準としたレベル 3 のトーラスを選択し，$(x_s^r, y_s^r) = (4,16)$, $(x_d^r, y_d^r) = (12,0)$ をとる．

手順 3. 低位リンクに対する再帰ルーティング (x_s, y_s) と (x_s^r, y_s^r) 間で，$(x_s, y_s) = (x_s^r, y_s^r)$ となるまで，本手続きを再帰的に行い，$(x_s, y_s), (x_s^r, y_s^r)$ 間の経路リストを得る．次に (x_s^r, y_s^r) から (x_d^r, y_d^r) 間にあるレベル l_r のノードのリストを経路リストに加える．(x_d^r, y_d^r) と (x_d, y_d) 間の経路も $(x_s, y_s), (x_s^r, y_s^r)$ 間と同様に求め，経路リストの末尾に付加する．

図 10(b) で，$(8,19) \rightarrow (4,16)$ について再帰的に経路を求めると，2 回目の再帰呼び出しで $l_r = 1$ のトーラスを決め，経路リスト $\{(7,18),(5,18),(5,6)\}$ を得る．図 10(c) で，同様に 3 回目の再帰呼び出しを行ない，経路リスト $\{(8,19),(8,18),(7,18)\}$ と $\{(5,16),(4,16)\}$ を得る．これらを継げ，レベル 3 のトーラス上の経路リスト $\{(4,16),(4,8),(4,0),(12,0)\}$ の前に加える．同様に $(12,0) \rightarrow (12,2)$ についての経路を求め，経路リストの末尾に追加する．

4.3.3 2D-SRT の直径

$N \times N$ $(N = 2^n)$ ノードから成る 2D-SRT の直径は，単純に $(x_s, y_s) \rightarrow (x_d, y_s) \rightarrow (x_d, y_d)$ という経路をとれば $2 \cdot D_{1D-SRT}(n)$ を越えないことは明かである．2D-SRT のルーティングでは，経路途中で現在の移動方向と直交する方向の移動も含まれるが，補題 1 から，上位レベルに対する下位レベルのノード配置が同じであるので，移動途中で直交する方向への移動が加わっても，直交方向移動後の下位レベルのノード配置は変化しない．例えば図 10(a) で，y 方向 $((4,16) \rightarrow (4,0))$ に移動後，x 方向 $((4,0) \rightarrow (12,0))$ に移動するが，y 方向の下位レベルのノード配置は x 方向の移動前 $(4,0)$ も移動後 $(12,0)$ も変らない．従って 2D-SRT の直径も N ノードから成る 1D-SRT の直径の 2 倍をを越えない．このため，2D-SRT の直径のオーダーは，$\mathcal{O}(2^{\sqrt{\log_2 N}}\sqrt{\log_2 N})$ を越えない．

4.3.4 2D-SRT のネットワーク性能

2D-SRT と他の相互結合網の平均距離，直径及び次数を表 11 に示す．2D-SRT の平均距離は，1D-SRT と同様の理由で，Optimal Routing(表中 Opt.) に比べ，再帰ルーティング (表中 Rec.) の平均距離が長くなる．2 次元では，“1 次元方向の平均距離の増加” の 2 乗となるため，Optimal Routing との差が 1D-SRT よりも大きく，

表 11: 2 次元 SRT と他の結合網の平均距離, 直径, 次数の比較

Number of node	2^8	2^{10}	2^{12}	2^{14}	2^{16}
Mean distance 2D-SRT and other networks					
2D-SRT (Opt.)	3.6	4.8	6.3	7.9	10.1
2D-SRT (Rec.)	4.2	5.7	7.8	10.4	13.3
2D Torus	4.0	8.0	16.0	32.0	64.0
RDT(2,4,1)/α	-	5.6	6.7	7.8	9.0
Diameter (Degree) of 2D-SRT and other networks					
2D-SRT (Opt.)	6(8)	8(8)	11(8)	13(8)	16(8)
2D Torus	16(4)	32(4)	64(4)	128(4)	256(4)
3D Torus	10(6)	16(6)	24(6)	40(6)	64(6)
RDT(2,4,1)/α	-	7(8)	8(8)	10(8)	12(8)
2D-DTN[13]	8(8)	16(8)	32(8)	64(8)	128(8)
HyperCube	8(8)	10(10)	12(12)	14(14)	16(16)
CCC	-	14(3)	18(3)	-	24(3)
HyperNet[14]	16(4)	-	19(5)	-	23(6)
DeBruijn[15]	4(8)	5(8)	6(8)	7(8)	8(8)
Pladhan	7(4)	9(4)	11(4)	13(4)	15(4)

- 未定数

10 ～ 30% 程となる. RDT(2,4,1)/α の平均距離と比べると, 殆ど劣らない性能を持つことが分る.

2D-SRT の直径と次数を他の結合網と比較すると, 2D-SRT は少ない次数でハイパーキューブに近い性能を持ち, 他の超並列計算機向きの結合網と比べ, 遜色ない性能を持つことが分る. 2D-SRT は, トーラスだけを階層的に用いて構成されるので, 配線が容易で, 故障回避アーキテクチャなどのインプリメントが可能 [11] という特徴を有している.

4.4 まとめ

本章では, グリッドの大きさが異なるトーラス結合網を, 互いに重ならないように再帰シフト構造に配置した, 超並列計算機のための新しい相互結合網 Shifted Recuresive Torus (SRT) 網を提案した. 最初に 1 次元の SRT(1D-SRT) の構成法を

定義し, その直径を求める手法を示した. 次に, 1D-SRT の上位のレベルから再帰的に経路を求める再帰ルーティングアルゴリズムを提案し, 最適ルーティングに近い性能を与えることを明かにした. また, 1D-SRT を他の 1 次元の相互結合網と比べた結果, 1D-SRT が少ない次数で小さな直径を持つことを明かにした.

1 次元 SRT 網を拡張して, x, y の両方向で SRT が構成可能な条件を示し, 2 次元 SRT 網を提案した. 従来の超並列計算機用相互結合網と, 平均距離, 直径及び次数について比較した結果, 2D-SRT はトーラスだけで構成される非常に簡単な構造でありながら, 従来の相互結合網に殆ど劣らない性能を有することが分った.

5　3次元実装と放熱手法

5.1　3次元実装

LSIの微細加工技術は，近年急速に進展しており，2000年では0.18μm ルールのプロセスが実用のものとなっている．これに伴ない，LISの集積化も高度に進展しており，0.18μm ルールのLSIでは数億トランジスタが1チップに集積でき，2005年には0.10μm ルール，10億トランジスタ，2010年には0.05μm ルール，100億トランジスタが1つのLSIに集積可能と予想されている．

その一方，現在の超高速計算機システムでは，このような超高集積かつ超高速のCPUを用いているが，CPUチップと主メモリのチップは依然としてPCBボード上の配線に頼っており，チップ内の信号伝達速度に比べて，チップ間の速度の遅さがシステム全体の性能向上の大きな問題となってきている．更に，本センターのRS/6000 SP システムのような超並列システムでは，本来非常に高速な通信が要求されるPE間結合にPCIバスを介してネットワークインターフェースを設けており，CPUとの信号伝達速度は主メモリよりも更に一段遅い速度となってきている．PE間の信号伝達速度を高速にすることが，システム全体の性能向上のための大きな課題である．

LSI内の信号伝達速度についても同様に，現在ではLSI内の速度を決定づける主たる要因は各トランジスタのスイッチング速度である．しかしながら，LSIの高集積化が進展すると，トランジスタのスイッチング速度は比例縮小によって高速化できるが，集積する回路の増加に従って，平均の配線長が増加し，配線による遅延が処理速度を決定する大きな要因となってくる．

高度に集積される回路量と配線による遅延を軽減するため，システム全体を1チップに集積する “System on Sillicon” の技術が提案され，これを3次元的に集積して超高並列，超高集積マルチプロセッサシステムを構築することが提案されている．この基礎技術として，3次元メモリシステムの3次元IC技術[16][17]が発展しており，3次元コンピュータが研究されている．Littleらは，5個のウェーハスタックとして組織された32 × 32のセルラアレイを開発した[18]．このスタックは，アキュムレータとシフタの二種類のウェーハにより構成されている．ダイの大きさはおよそ1立方インチで，10メガヘルツでのスループットは約600MOPSである．縦方向結線については，Campbell[19]らや，Carson[16]による研究が報告されている．近年では，栗野ら[20]がさらに高度な3次元構築技術について議論している．

3次元コンピュータの構築にあたって重大な障害となるのは，縦方向の結線のた

めのエリアコストである．縦方向の配線を少なくする相互結合網として，前節で再帰シフトトーラスについて述べた．3次元実装でのもう一方の重要な課題は，放熱手法である．次節では，3次元ウェーハスタック実装の放熱の問題について取り組む．

5.2 放熱に関する問題

大規模な超並列システムを実装するための手段として，1枚のウェーハ上に多数のプロセッシング要素(PE)を搭載するウェーハスケールインテグレーション(WSI)や，WSIを3次元的に構築するウェーハスタックシステム[21]が注目されている．超並列システムをWSIやウェーハスタック実装により構築した場合，PE間の相互結合網が全てチップ内部で可能となるため，クロックの遅延が少なくなり，システムの小型化，高速化，小電力化が期待できる．しかしながら，大口径ウェーハ上に発生する欠陥は現在の集積技術では避けられない問題であるため，WSI技術を用いて超並列システムを構築するためには，予備のPEなどを配置するなどの欠陥PEの救済手段が不可欠である．さらにウェーハスタック実装においては，三次元構造内部にあるPEの発生する熱をどのように放熱するかが，欠陥PE救済に加えて重要な問題となっている．各PEの消費電力はクロックの高速化に比例して増加するため，効率の良い冷却が可能になれば，大規模なシステムの構築が可能になると同時に，高速で発熱量の大きなPEを用いることができる．

欠陥PEの救済は，大規模科学技術計算に適する格子結合網について，これまでに多くの研究がなされている[22, 23]．これらは，初期に動作するものと仮定するPEの周囲に，予備の冗長なPEを配置し，欠陥PEの機能を冗長PEで置き換えることにより，論理的に欠陥の無い格子結合網を得ている．予め用意しておく冗長PEの数は，網の再構成を行なうために，予想される欠陥PEの数よりも十分多い必要があるが，その結果，使用されない冗長PEがウェーハ上に残る．欠陥PEや使用されない冗長PEは電源をカットすることにより，発熱しないようにできるが，配置場所によっては，発熱する動作PEの放熱の妨げとなる．

そこでウェーハスタックの放熱性能を向上させることを目的として，動作PEを放熱性能に優れるウェーハ周辺部，休止PEを放熱の妨げにならない中心部に移動しながら，網の再構成を行なうアルゴリズムについて議論する．放熱を考慮した再構成方式としては，冗長PEの初期配置とシフト方向の重み付けによって冷却性能を向上させる手法が既に提案されているが[24]，本論文では，より高い冷却性能を得る方法として，複数の再構成を行ない，それぞれの冷却性能を予測し，最も高い

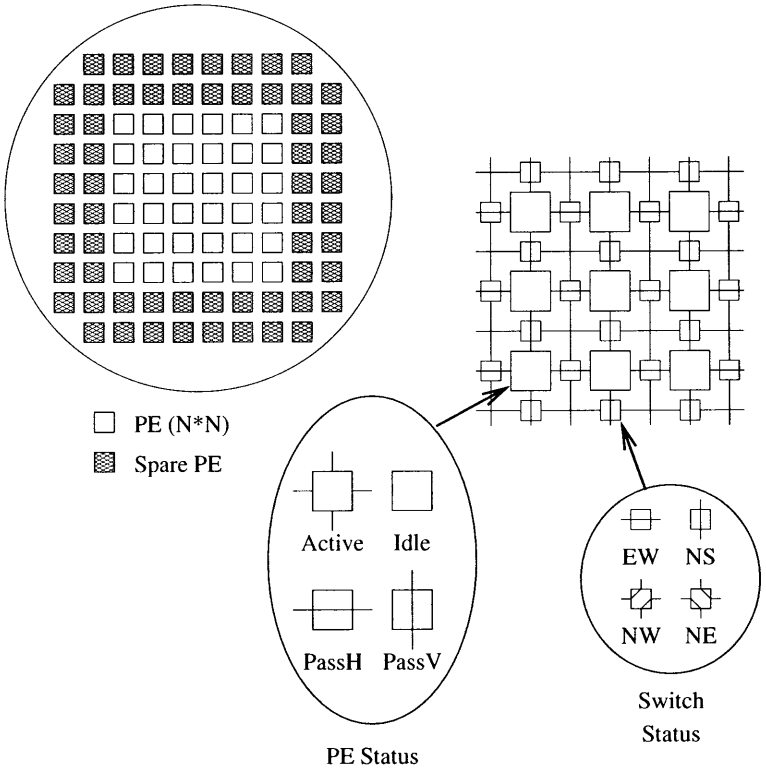


図 11: 格子結合網の欠陥回避アーキテクチャ

冷却性能を持つ再構成を発見し選択する方法を提案する．ウェーハスタックの冷却に関する他の従来研究として，スタック内を縦貫する銅の熱伝導シャフトを設け，これにより放熱を行なう手法 [25] が試みられているが，放熱性能が十分ではなく，試作システムでは直径が 2〜3 インチと小規模なものとなっている．

本章では，科学技術計算に適する格子結合網のウェーハスタック実装について，放熱を考慮した欠陥回避アルゴリズムによって，網の再構成とウェーハスタックの放熱を同時に行なうことを試みる．提案した欠陥回避方式のウェーハスタックシステム全体の歩留まり，スタック内部最高温度，および冷却の安定性について熱伝導シミュレーションによる評価を行なう．

なお，ウェーハスタックは三次元構造を持つため，本来であれば三次元での解析を行なうべきであるが，熱伝導シミュレーションにおける計算量が膨大で 3 次元での解析が困難なため，中心の 1 枚のウェーハを取り出して二次元的に解析を行なう．

5.3 欠陥回避アーキテクチャ

図11に，格子結合型マルチプロセッサシステムの欠陥回避可能なアーキテクチャを示す．中央に$N \times N$のPEアレイが置かれ，この周囲に冗長PEが配置される．中央のPEが欠陥である場合，この機能は周囲の冗長PEによって代行される．各PEは，動作状態(Active)，休止状態(Idle)，機能せず信号のみが通過する状態(PassH, PassV)の4つの状態をとることができるが，これらは欠陥PEが動作状態となれないことを除いて，PEの欠陥の有無とは関係なく設定できる．各PEの周囲には，バイパスリンクとスイッチが置かれる．スイッチはバイパスリンクとともにPE間の接続の方向を切り変えるために用いられ，PEの接続を通常の上下左右への接続から斜め上/下のPEへの接続に切りかえることができる．この様子を図12に示す．このように，欠陥PEを避けて隣接するPEに機能を代替することをシフトと呼ぶ．図12の例に示すように，欠陥PEが2つ続く場合でも，2段階のシフト操作を行なうことにより，欠陥を回避することができる．シフトは，図13の(B)の手続きによって行なわれる．Idle状態のPEに行き当たるまでシフトを繰り返すが，その過程で前方のPEが既に横にシフトしている場合は戻し，また横のPEが逆方向にシフトしている場合，これを順方向に戻す．シフトが完了すればSucceed, シフトが不可能な場合はFailを返す．

上下左右どちらの方向にシフトするかを決定するアルゴリズムは，WSI全体の歩留まりに大きく影響し，いくつかの方法が提案されているが，Kungらはグラフ理論を用いた方法[22]，Numataらはローカル情報のみを用いてヒューリスティックに欠陥回避を行なう方法(HS法)[23]を提案している．このHS法は簡単なローカル情報のみで，高いシステム歩留まりを得ることができる．図13(A)にHS法による欠陥回避のための全体の手続きを示す．最初に，欠陥していて使用状態にあるPEを順に検索する．次に，この欠陥したPEをどの方向にシフトするかを決定する．HS法では，乱数を用いて，上下左右一様な確率でシフトしている．シフトの結果，全てのPEの欠陥が回避されれば，再構成が成功，一定の回数内に全てのPEの欠陥が回避が不可能な場合は，この欠陥パターンについての再構成は不可能である．

5.4 ウェーハスタックの冷却

5.4.1 ウェーハスタックの構成法

大規模な格子結合型マルチプロセッサシステムを構成する場合，複数のウェーハ上にシステムの一部を実装し，このウェーハを層状に重ねることにより，ウェーハスタックによる実装が考えられる．この概念図を図14に示す．ウェーハ上には実装

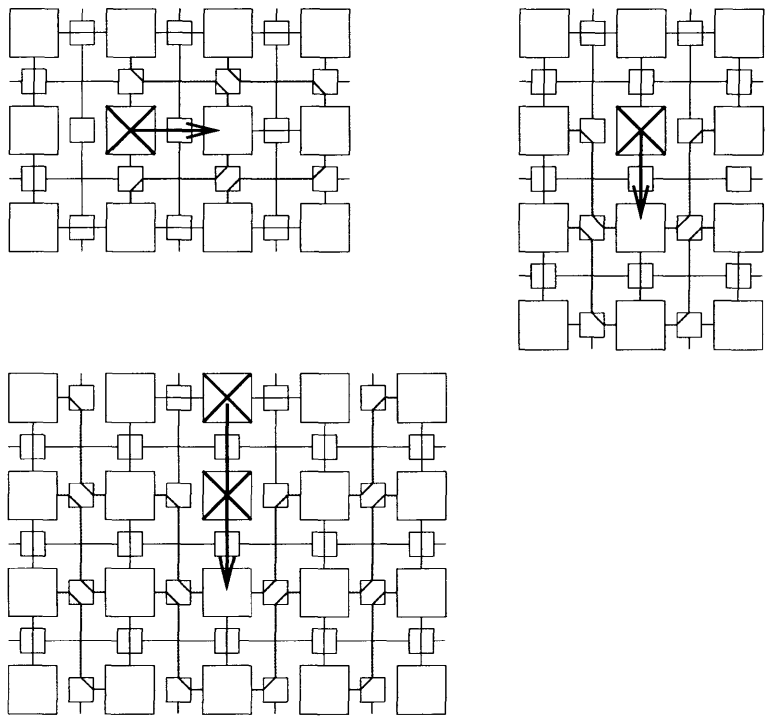


図 12: シフトの例

すべき格子結合網のサブセットに加え，冗長 PE を配置しておく．ウェーハ上の欠陥 PE は，図 13 の手続きによってウェーハ内部で再構成され，各ウェーハは，論理的に欠陥の無い格子結合網のサブセットを構成する．これらの論理的に欠陥の無いウェーハを縦方向に接続し，大規模な格子結合網を得ることができる．ウェーハ間の結合においてはバイパスリンクを設けることが困難なため，欠陥回避のためのシフトはウェーハ内で完結し，ウェーハ間にまたがるシフトは行わないものとする．

5.4.2 ウェーハスタックの放熱モデル

ここでウェーハスタックの冷却について考える．ウェーハ上の PE は，もし動作状態ならば発熱し，休止状態の場合は発熱しない．放熱をウェーハスタックの周囲から行なうとすると，スタック内の発熱量が同じ場合，発熱部分がなるべく周囲に存在する方が，発熱部分が中心部に集中する場合よりも，スタック内の最高温度を低くすることができる [24].

図 15 (I) にウェーハスタックの発熱モデルを示す．単位時間当りに単位面積を通過する熱量 $\dot{q}$ は，λ を熱伝導率として

$$\dot{q} = -\lambda \frac{dT}{dx}. \tag{24}$$

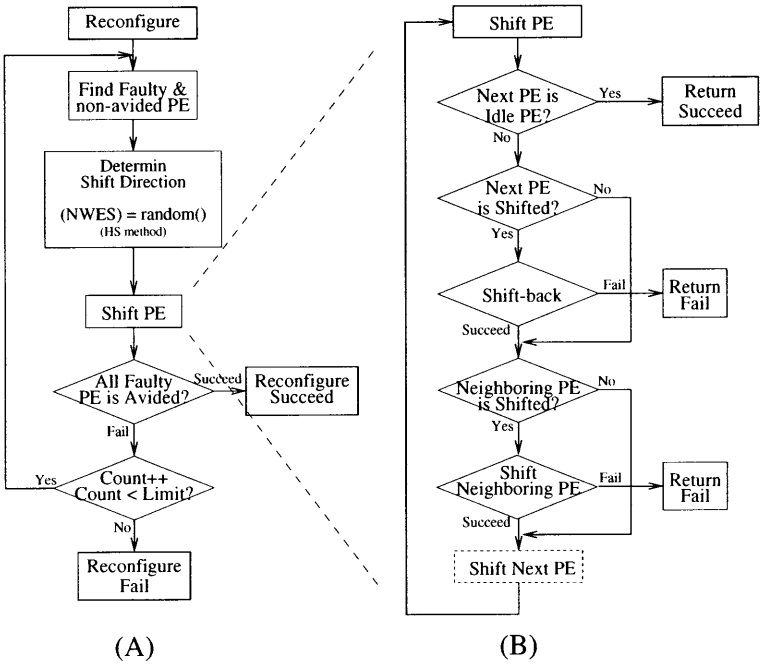


図 13: 再構成の手続き

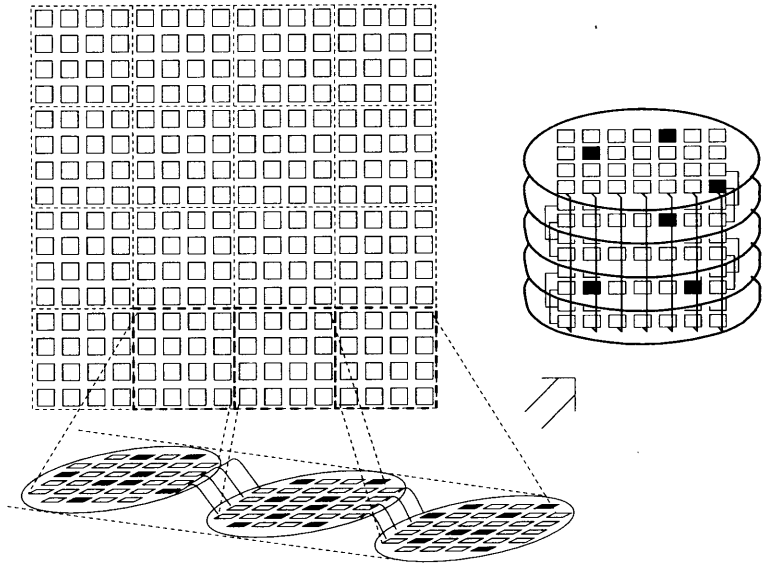


図 14: 格子結合型マルチプロセッサのウェーハスタック実装

表せる．すると，ある領域 (x, y, z) の側面からの熱の流入は，

$$dQ_x = -\lambda \frac{\partial T}{\partial x} dydzd\tau \tag{25}$$

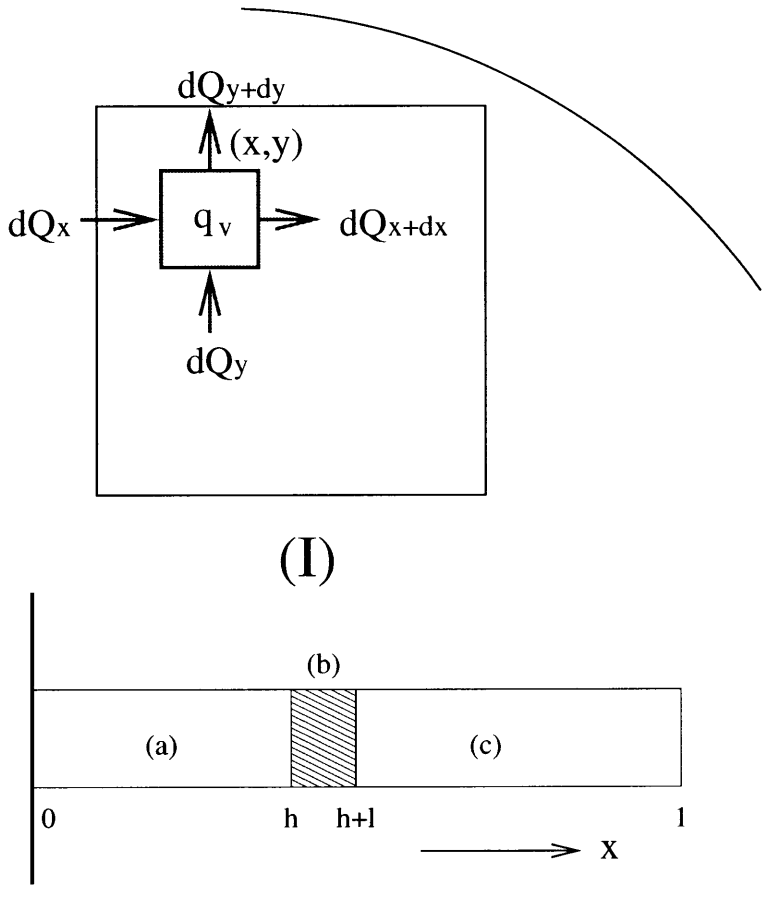


図 15: ウェーハスタックの放熱モデル

$(y, x$ も同様) であるから，領域に残る正味の熱量は次式で表せる．

$$dQ_x - dQ_{x+dx}. \tag{26}$$

また，この領域の内部発熱量は，$\dot{q}_v$ を単位体積当りの発熱量として，

$$\dot{q}_v \cdot dxdydz \cdot d\tau. \tag{27}$$

ここで，ウェーハの状態から $\dot{q}_v$ を考えると，

$$\dot{q}_v(x,y) = \begin{cases} 0 & if\ PE = idle \\ P & if\ PE = active \end{cases} \tag{28}$$

である．また，材料の比熱を c, 密度を ρ とすれば，内部エネルギの増分は，

$$c\rho dT \cdot dxdydz \tag{29}$$

式 (25) から (27) が式 (29) に等しいとすると，

$$c\rho\frac{\partial T}{\partial \tau} = \frac{\partial}{\partial x}\left(\lambda\frac{\partial T}{\partial x}\right) + \frac{\partial}{\partial y}\left(\lambda\frac{\partial T}{\partial y}\right) + \frac{\partial}{\partial z}\left(\lambda\frac{\partial T}{\partial z}\right) + \dot{q}_v. \tag{30}$$

熱伝導率を一定とするならば，

$$c\rho\frac{\partial T}{\partial \tau} = \lambda\nabla^2 T + \dot{q}_v \tag{31}$$

となる．4 章で熱伝導シミュレーションを行なうが，これは式 (31) を数値積分して行なう．

次に，PE の位置が内部温度にどのように影響するかについて考察する．ここでは単純化するために一次元で考える．図 15 (II) に一次元での熱伝導モデルを示す．ここで，領域 (a), (c) は発熱領域であり，具体的には動作 PE に相当する．領域 (b) は非発熱領域であり，休止 PE に相当する．ここで左端をウェーハ中心と仮定し，ここからは熱の入出力が無いものとする．また，右端をウェーハの外縁と仮定し，内部の発熱は全て右端から放熱される．また境界条件は，温度一定の条件とする．単位時間当りに単位面積を通過する熱量は式 (24) となるから，ウェーハ中心 (左端) の温度は，非発熱領域の位置を h として以下のように書ける．

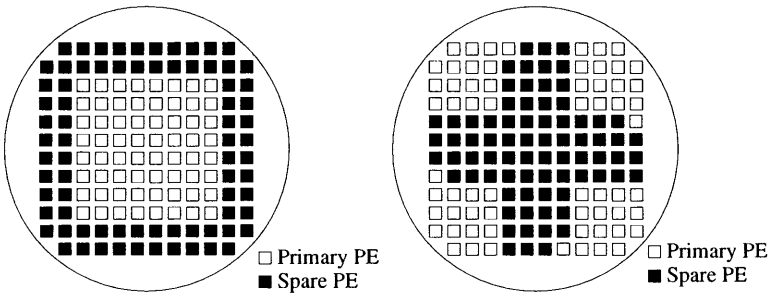
$$T_{x=0} = \frac{\dot{q}_v}{\lambda}\left\{l(h+\frac{1}{2}l) + \frac{1}{2} - l\right\} + T_0, \tag{32}$$

ここで T_0 はウェーハ外縁 (右端) の温度，また l は非発熱領域の大きさ，つまり PE の一辺に相当する．この式より，ウェーハ中心の温度を低下させるためには，h を小さく，つまり非発熱領域をウェーハ中心に持ってくれば良いことが分る．

各 PE は，動作状態，欠陥状態，及び休止状態の 3 つの状態を取ることができる．PE の歩留まりが高い場合，殆どの初期 PE が動作状態，冗長 PE は休止状態となる．冷却の観点から考えると，休止 PE をできるだけウェーハ中心に持ってくるようなシフトを行なえば良いことが分る．次に休止 PE をできるだけウェーハ中心に持ってくるような戦略について述べる．

5.4.3 冗長 PE の集中配置

欠陥回避を開始する前の PE の初期配置について，冗長 PE を周囲に配置する方式と中心部に配置する方式が考えられる．この配置方式の違いについて，システム全体の歩留まりとウェーハスタック内の最高温度について考察する．



(a) 周囲配置　　(b) 集中配置

図 16: 冗長 PE の初期配置

冗長 PE をウェーハの周囲に配置する場合 (図 16(a)) は，動作すると仮定している PE が配置されるウェーハ中心部の表面状態が良く，欠陥回避の際にどの方向にもシフト可能なので，欠陥回避後のウェーハとしての歩留まりが高いことが期待できる．しかしながら，欠陥 PE が少なく PE の配置が初期配置からあまり変化しない場合，動作 PE がウェーハ中心部に集中するため，中心部の温度は高くなる．

一方，冗長 PE を中心に配置した場合 (図 16(b)) は，発熱する PE が周囲に配置されるため，シフトがあまり行なわれない場合，中心部の冗長 PE が休止 PE として残され，ウェーハの内部温度は低くできる．その一方で，欠陥回避の際シフトの方向が中心方向に限定されるため，ウェーハの歩留りは低下することが報告されている [24].

5.4.4 シフト方向の重み付け

次に欠陥 PE を回避する際のシフト方向について考える．欠陥回避の際のシフト方向を決定するためには，様々なアルゴリズムが提案されている．HS 法 [23] は非常に高いシステム歩留まりを得ることができるが，シフトの方向は乱数を用いて決定されており，方向は均質である．

そこで，冷却効率を高めるために，重み付けシフトが提案された [24]．ウェーハの周囲から放熱するために，シフトをなるべくウェーハの外側に向って行なうように，シフト方向を決定する乱数に重み付けを行なう．欠陥 PE が外側に向ってシフトする確率を P_O, 内側にシフトする確率を P_I, それと直交する方向 (円周方向) にシフトする確率を P_R, P_L とすると，この確率は次式で与えられる．

$$P_O = (1.0 - 4 \cdot d \cdot \beta) \times 0.25 \tag{33}$$

$$P_I = (1.0 - 2 \cdot d \cdot \beta) \times 0.25 \tag{34}$$

```
/* 再構成後のPE配置マップを返す */
pemap heuristic_replace(W)
wafer  W;   /* 欠陥PEを含む再構成前のウェーハ */
{
  for( i = 0; i < N; i++ ){
    M_i = Reconfigure(W, seed(i));
    /* 異なる乱数のseedで複数の再構成を実行 */
    /* 再構成マップ M_i を得る */
  }
  /* 温度予測値 T_i が最高となる再構成マップを選ぶ. */
  maxT = 0;
  for( i = 0; i < N; i++ ){
    T(M_i) = Σ_j^{all active PEs} |d(^{M_i}PE_j)|^2;
    if( T(M_i) > maxT )      M_m = M_i;
  }
  return( M_m );
}
```

図 17: ヒューリスティック置換の手順

$$P_R \quad = \quad P_L = (1.0 - d \cdot \beta) \times 0.25 \tag{35}$$

ここで d は欠陥PEのウェーハ中心からの距離，β はシフト重み $(0 \leq \beta \leq 0.5)$ である．ウェーハの中心部にあるPEは，どちらに移動してもスタックの放熱性能への影響が少ないので，シフトの偏り (P_O と P_I の比) は小さい．しかしウェーハの周辺部にあるPEは，もし内側にシフトすると放熱性能を大きく損なうので，外側にシフトする確率 P_O が大きくなる．

5.4.5 ヒューリスティック置換方式

本節では，スタック内のウェーハ温度をより一層低下させる新たな手法として，事前にウェーハ内温度を予測しながら最適なPE構成を発見する，ヒューリスティック置換方式を提案する．

再構成を行なう場合，欠陥を含むあるウェーハ1枚に対して，多くの場合，複数の再構成が可能である．あるPEを回避するのに，上下左右どちらにも回避可能な

場合，ある欠陥PEを回避するために，上へのシフトにより再構成が可能となったとしても，別の解として右にシフトしても再構成が可能であることは，非常に高い確率で期待できる．そこで，ヒューリスティック置換方式では，再構成を複数回試行し，各PEの状態を保持した再構成マップを集める．それぞれの再構成マップの放熱性能を予測し，最も良い放熱性能を持つ再構成マップを選択することにより，ウェーハ内の温度を低下させることを狙いとする．

再構成マップの放熱性能を評価するための方法としてはいくつもの方法が考えられるが，熱伝導シミュレーションが最も正確である．しかし，このためには非常に多くの計算機資源が必要となる．例えば，20×20 PE のウェーハの熱伝導シミュレーションには，1回当り 128CPU の T3E を用いて約1時間の計算時間が必要である．

そこで，実用上十分な速度で再構成マップの放熱性能を評価するために，放熱性能を予測する関数を導入する．この関数に求められる特性として，動作PEをウェーハ周辺に持って行きたいのであるから，動作PEがウェーハ周辺に多い場合に高い得点が付き，中心部に動作PEが集まると得点が低くなるようにすれば良い．そこで，この関数を次にように仮定する．

$$T(M_i) = \sum_{j}^{all\ \text{active}\ PEs} |d({}^{M_i}PE_j)|^2 \tag{36}$$

ここで は，M_i はある乱数の seed によって得られる再構成結果のマップ，$d({}^{M_i}PE_j)$ は，マップ M_i における j 番目の動作PEのウェーハ中心からの距離，$T(M_i)$ はその再構成マップ M_i によって得られる得点とする．この関数を用いると，周辺に動作PEが多い時，高い得点が得られ，概ね初期の要求に合致するものであると考えることができる．

この評価関数によって最も高い得点を得た再構成マップを選択する方式を，ヒューリスティック置換方式と呼ぶ．この手続きを図17に示す．最初に異なる乱数の seed を用いて N 回の再構成を繰り返し，再構成マップごとの放熱性能を評価する．この中で放熱性能が最高となるPE構成を選び，最終的な再構成マップとして採用する．

5.5 歩留まりとスタック内最高温度

5.5.1 システム歩留まり

一枚のウェーハ上に置くPEアレイのサイズが $(16+4) \times (16+4)$ の場合のシステムの歩留まりを図18に示す．ここで，PE歩留まりとは全PE中の正常なPEの割合，またシステム歩留まりとは，欠陥を含むウェーハを用いても，システムとし

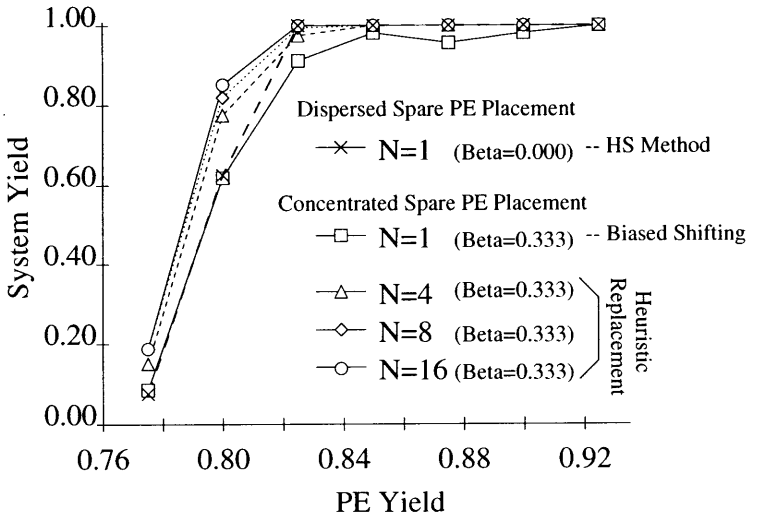


図 18: システム歩留まり $(16+4)^2$

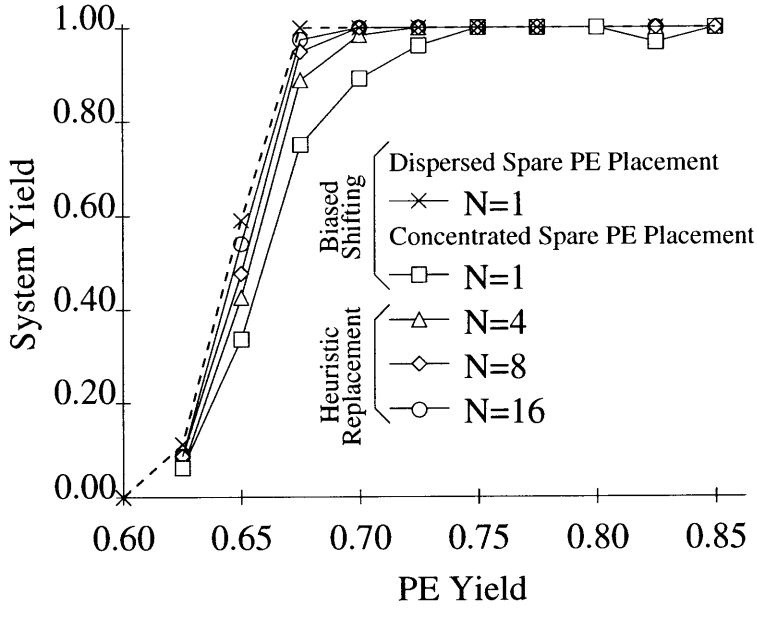


図 19: システム歩留まり $(10+4)^2, \beta = 0.333$

て利用可能な状態に再構成が成功する割合である．冗長 PE が周囲配置 (Dispersed Spare PE Placement) で重み $\beta = 0.000$, かつ試行する再構成の回数 $N = 1$ の場合，HS 法 [23] と全く同一の条件となる．また，$\beta > 0$ で試行回数 $N = 1$ の場合，文献 [24] の重み付けシフトと同一の条件となる．$N > 1$ の場合がヒューリスティック置換であり，重み付けシフト ($\beta > 0$) と併用することが可能である．

文献 [24] の結果では，冗長 PE を中心部に集中して配置 (Concentrated Spare PE

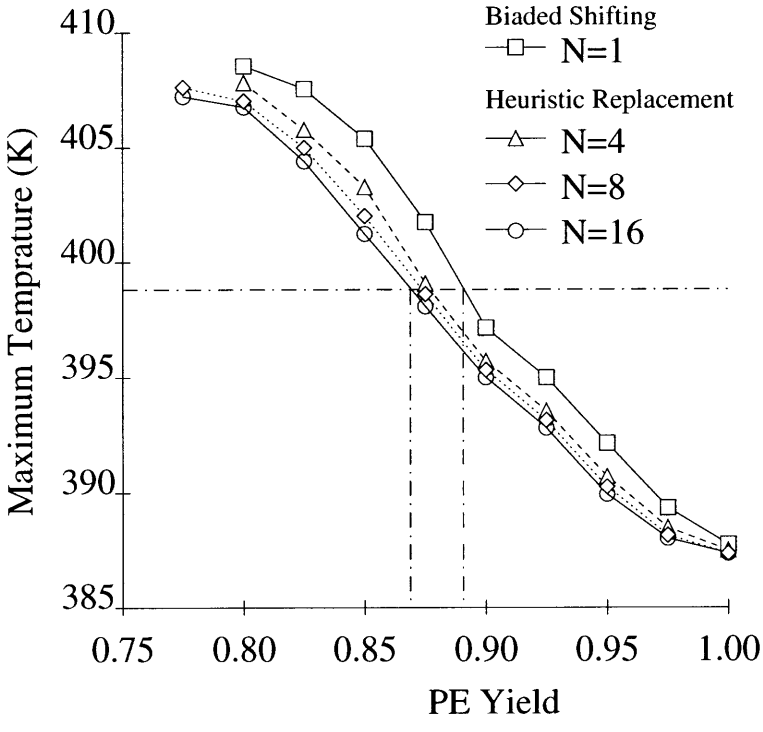


図 20: スタック内最高温度 $(16+4)^2, \beta = 0.333$

Placement) する場合，スタック内最高温度は低下できる反面，PE 歩留まりが比較的高い状態 (PE Yield $= 0.85 \sim 0.90$) でもシステム歩留まりが 1.0 にならなかったが，ヒューリスティック置換では，再構成を複数回試行するため，システム歩留まりが向上する．おおむね 4 回以上試行することにより，HS 法と同等以上のシステム歩留まりを得ることができた．

図 19 に，より小さなアレイサイズの場合として，$(10+4) \times (10+4)$ でシフト重み $\beta = 0.333$ の場合のシステム歩留まりを示す．アレイサイズが $(16+4) \times (16+4)$ の場合と同様に，再構成の試行回数の増加につれ，システム歩留まりは向上する．

5.5.2 スタック内最高温度

次に，スタック内の最高温度を熱伝導シミュレーションにより評価した．シミュレーションの条件として，ウェーハの素材は Si とし，熱伝導率などの物理定数は Si と同じ値を用いた．つまり，$\lambda = 168.0(W \cdot m^{-1} \cdot K^{-1}), \rho = 2.34(g \cdot cm^{-3}), c = 22.1(J \cdot K^{-1} \cdot mol^{-1}\ at\ 400K)$ である．また，周囲温度 (放熱器温度) は 50 ℃ $(323K)$，1PE 当りの発熱量を 0.5W とした．PE アレイサイズが $(10+4) \times (10+4)$ の時，1 つの PE の面積を $10mm \times 10mm$，ウェーハの直径は $195mm$ とし，$(16+4) \times (16+4)$

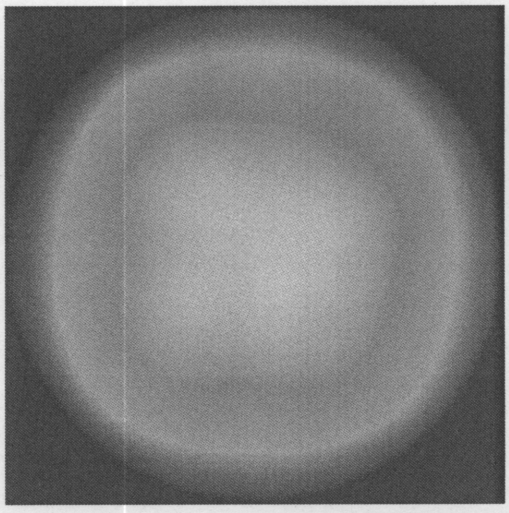


図 21: ウェーハ内温度分布の例

の時，1 つの PE の面積を $5mm \times 5mm$，ウェーハの直径は $140mm$ とする．また，冗長 PE は集中配置とした．

図 20 に，$(16+4) \times (16+4)$ の PE アレイにおけるスタック内最高温度を示す．また，図 21 に，この時のウェーハ内の温度分布の一例を示す．冗長 PE が集中して配置してあるので，PE の歩留まりが高いと，冗長 PE が使われないまま中心に残るため，動作 PE の冷却の妨げにならず，スタック内最高温度は低くなる．一方，欠陥 PE が増加 (PE 歩留まりが低下) すると，動作 PE が内部の冗長 PE の方にシフトし，これらが発熱するようになるので，冷却効率が低下する．

図から分かるように，ヒューリスティック置換では，再構成の試行回数が増加するにつれ，同じ PE 歩留まりの場合でも，より低いスタック内最高温度が得られる．シリコン半導体の動作温度の上限は，一般には 125 ℃ (398K) 前後であるが，例えば $(16+4) \times (16+4)$ の場合 (図 20) 重み付けシフトでは，PE 歩留まりが 0.89 以上のウェーハでないと，実用システムとして稼働させることは難しい．しかしながら，試行回数を増加させた $N = 16$ の場合では，PE 歩留まりが 0.865 程度のウェーハも利用することが可能となる．

図 22 に，より小規模な $(10+4) \times (10+4)$ の PE アレイのスタック内最高温度を示す．発熱する PE 数が少ないのでスタック内最高温度は全般に低いが，やはり試行回数 N を増加させることにより，同じ PE 歩留まりでの温度は低下させることができる．

図 23 に，重み付けシフト方式とヒューリスティック置換方式のスタック内最高温

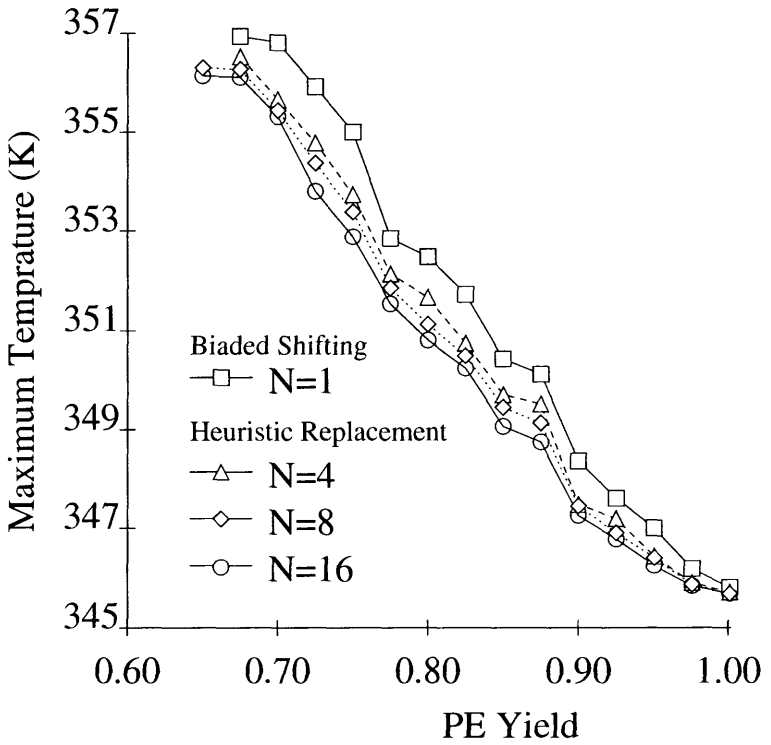


図 22: スタック内最高温度 $(10+4)^2, \beta = 0.333$

度を示す．重み付けシフト方式でシフト重みを大きくしても，スタック内の最高温度を下げることができるが，比較的大き目のシフト重み ($\beta = 0.333$) の場合でも，ヒューリスティック置換方式の方が，スタック内最高温度の低下に有効である．

5.5.3 冷却の安定性

熱伝導シミュレーションでは，さまざまな欠陥パターンを持つウェーハを多数生成し，それぞれの温度評価を行なった．その場合，欠陥PEの個数が同じであっても，欠陥PEの場所や再構成手法によっては，スタック内最高温度にかなりのばらつきが生じる．PE歩留まりが同じならば，どのような欠陥PEの配置でも安定して冷却が可能な方式である方が優れている．

そこで，PE歩留まりを固定しつつ欠陥PEの場所を変えたウェーハを多数生成して，スタック内最高温度時の標準偏差を求めた時の，PE歩留まりごとの標準偏差の平均値と最大値を表12に示す．なお，冗長PEの初期配置は集中配置である．表より，アレイサイズにかかわらず，試行回数 N を増加させると，標準偏差が減少している，つまりどのような欠陥PEの配置であっても，安定して冷却性能を得ることが可能であることが分かる．試行回数について見ると $N = 4$ 程度で十分小

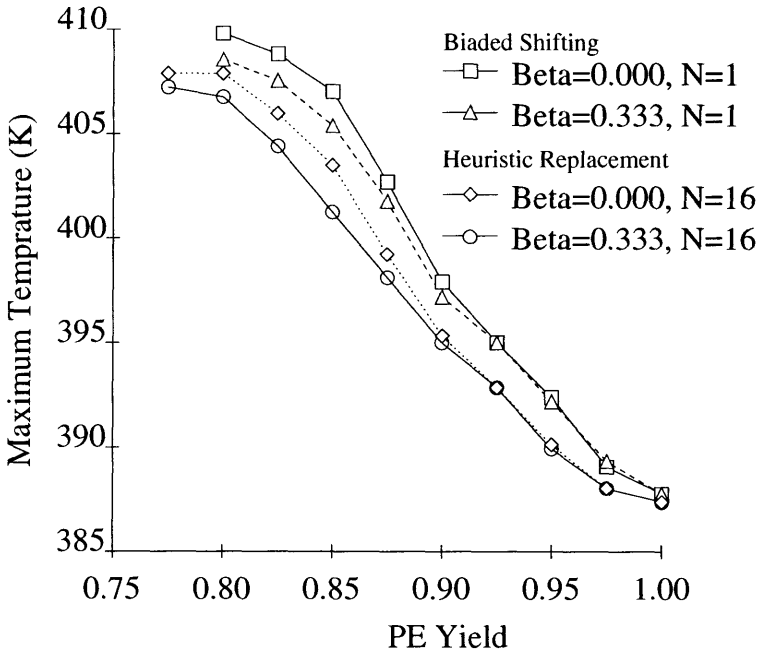


図 23: スタック内最高温度 $(16+4)^2$

さな標準偏差が得られており，それ以上試行を繰り返しても冷却性能の安定性に対する寄与は小さい．シフト重みを増加させることによっても冷却の安定性を高めることができるが，シフト重みを上限まで ($\beta = 0.500$) 高めた場合でも，試行回数を増やした場合の安定性の方が高い結果となった．

図24に，冗長PEの集中配置の場合 (アレイサイズ $(16+4)\times(16+4)$) のPE歩留まりとスタック内最高温度の標準偏差を示す．おおまかな傾向として，PE歩留まりが中くらい ($0.85 \sim 0.95$) の場合に，標準偏差の減少が大きいことが分かる．PE歩留まりが非常に低い場合 (0.85 以下) は，欠陥回避が可能な再構成が殆んど一意に定まり，置換方式による差が出にくい．また，PE歩留まりが十分大きいと，殆んどのPEを動かすことなく欠陥回避が可能であり，やはり置換方式による最高温度のばらつきの差が小さくなると考えられる．また，表12から推測されるように，PE歩留まりの良否にかかわらず，シフト重み β による安定性向上の効果よりも，試行回数の増加による安定性向上の効果が大きいことも分かった．

5.6 結論

本論文では，格子結合型マルチプロセッサシステムのウェーハスタック実装について，網の再構成方式を改良することによって冷却性能を高める方法を議論した．放熱を効果的に行なうために，欠陥回避用の冗長PEを中心に配置し，シフト方向

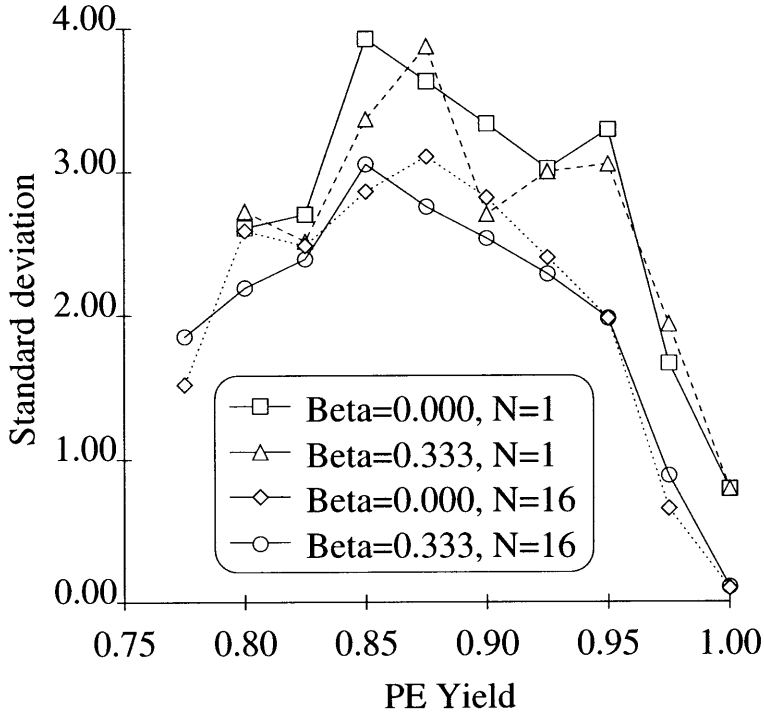


図 24: スタック内最高温度の標準偏差 $(16+4)^2$

の重み付けを行なう方式に対して，複数の再構成を繰り返し，その中から冷却性能を予測し，最も冷却性能の高い再構成マップを選択するヒューリスティック置換方法を提案した．システム全体の歩留まりとスタック内の最高温度をシミュレーションによって求めたところ，従来の重み付けシフトでシステム歩留まりが低下していた領域においても，高いシステム歩留まりを得ることが可能となった．また，熱伝導シミュレーションによるスタック内最高温度は，ヒューリスティック置換方式を用いると，重み付けシフトよりも，一層低い温度を得ることが可能であった．ヒューリスティック方式と重み付けシフトは併用可能であり，両者を用いる場合が最も低い温度となる．更に冷却の安定性について統計をとったところ，比較的少ない試行回数でも，スタック内最高温度の標準偏差を重み付けシフトよりも大幅に小さくできた．

表 12: スタック内最高温度の標準偏差 (冗長 PE 集中配置)

PE Array Size	β	# of Reconfig	Avr. σ	Max. σ
$(10+4)^2$	0.000	1	1.67	2.40
$(10+4)^2$	0.000	4	1.37	2.17
$(10+4)^2$	0.000	8	1.26	2.07
$(10+4)^2$	0.000	16	1.25	2.17
$(10+4)^2$	0.333	1	1.65	2.70
$(10+4)^2$	0.333	4	1.33	2.22
$(10+4)^2$	0.333	8	1.33	2.11
$(10+4)^2$	0.333	16	1.21	1.99
$(10+4)^2$	0.500	1	1.48	2.42
$(16+4)^2$	0.000	1	2.90	3.92
$(16+4)^2$	0.000	4	2.39	3.39
$(16+4)^2$	0.000	8	2.36	3.17
$(16+4)^2$	0.000	16	2.23	3.10
$(16+4)^2$	0.333	1	2.77	3.87
$(16+4)^2$	0.333	4	2.32	2.84
$(16+4)^2$	0.333	8	2.17	3.00
$(16+4)^2$	0.333	16	2.13	3.05
$(16+4)^2$	0.500	1	2.76	3.68

6 おわりに

本論文では，北陸先端科学技術大学院大学 情報科学センターの超並列計算サーバを通じて，学術目的の情報科学センターが将来提供すべき超高速計算サーバに必要とされる技術とその解決方法について検討した．

最初に同センターにこれまで導入された超並列システムについて検討し，これらのシステムの特徴，利点，課題となっている点について議論した．同センターの超並列システムを用いて，相互結合網の通信性能がシステム全体の性能に与える影響について調べたところ，PE 間通信のバンド幅が重要であり，特に通信遅延がシステム全体に大きく影響することが分かった．

相互結合網のトポロジについて検討したところ，同センターが導入しているような二次元や三次元のトーラス結合網が科学技術計算に適していることが示された．三次元トーラス網のシステムは同センターで既に稼働しており，高い計算性能を有していることが示されているが，将来の数百万のプロセッサを有するシステムへの

拡張のためには，再帰的な結合網が有効であることが分かった．再帰的なトーラス網として「再帰シフトトーラス相互結合網」について詳しく議論した結果，本相互結合網は従来の相互結合網と遜色ない通信性能を有しながら，非常に高い拡張性があることが分かった．このため，再帰シフトトーラス相互結合網は将来の情報センターがサービスすべき超並列システムの基幹となる相互結合網の有力な候補であると言える．

また，実装手法について議論し，システム全体の高速化のためには，システムを高度に集積することが可能な三次元実装が重要であることを示した．三次元実装では放熱が重要な課題であるが，放熱を考慮して予備PEを選択する「WSIスタックのヒューリスティック配置方式」を用いることにより，ウェーハスタックのスタック内最高温度を大幅に冷却できることが分かった．将来の超高速計算サーバを実現する重要な課題が解決できる．

これらの知見を踏まえて，情報科学センターで既存のシステムを単に購入するだけでなく，本センターの研究成果が反映された先進的なシステムを開発し，本学の教育研究活動に資することを，本研究の結論とする．

参考文献

[1] 佐藤理史(編). "JAISTにおける超並列関連研究(1992年度–1993年度)". Technical Report IS-TM-94-000, 北陸先端科学技術大学院大学, Mar. 1994.

[2] 佐藤理史(編). "JAISTにおける超並列関連研究(1994年度–1996年度)". Technical Report IS-TM-97-3, 北陸先端科学技術大学院大学, May 1997.

[3] 佐藤理史(編). "JAISTにおける超並列関連研究(1997年度)". Technical Report IS-TM-98-1, 北陸先端科学技術大学院大学, Apl. 1998.

[4] 沼田一成, 堀口進. "格子結合型マルチプロセッサシステムのWSI構成法". 信学論, Vol. J77-D-I, No. 2, pp. 121–129, 1994.

[5] 楊愚魯, 天野英晴, 柴村英智, 末吉敏則. "超並列計算機に向き結合網:RDT". 信学論, Vol. J78-D-I, No. 2, pp. 118–128, 1995.

[6] W. Worth Kirkman and Donna Quammen. "Packed Expornential Connections — A Hierarchy of 2D-Meshes". In *Proceeding of the Fifth International Parallel Processing Symposium*, pp. 464–470, Apr. 1991.

[7] Cho-Chin Lin and Viktor K. Prasanna. "Bounds on the diameter of one-dimenstional PEC networks". *Journal of Parallel and Distributed Computing*, Vol. 29, No. 1, pp. 1–16, 1995.

[8] 井口寧, 堀口進. "超並列計算機向きプロセッサ結合網 SRT". 信学技報, Vol. 95, No. 327, CPSY95-69, pp. 25–30, Oct. 1995.

[9] Yasushi Inoguchi and Susumu Horiguchi. "Shifted Recursive Torus Iinterconnection for High Performance Computing". In *High Performance Computing on the Information Superhighway*, pp. 61–66. IEEE Computer Society Press, Apr. 1997.

[10] Yasushi Inoguchi, Teruo Matsuzawa, and Susumu Horiguchi. "SRT Iinterconnection Network on 3D Stacked Implementation by Considering Thermo-Radiation". In *International Conference on Innovative Systems in Sillicon*, pp. 41–51. IEEE Computer Society Press, Oct. 1997.

[11] 川井雅之, 井口寧, 堀口進. "超並列計算機向き相互結合網SRTのデッドロックフリールーティング". 情処論, Vol. 40, No. 5, pp. 1977–1984, 1999.

[12] 川井雅之, 井口寧, 堀口進. "超並列計算機向き相互結合網SRTにおける適応型ルーティング". 情処論, Vol. 41, No. 7, pp. 2010–2017, 2000.

[13] SangHo Chae, Jong Kim, DongSeung Kim, SungeJe Hong, and Sunggu Lee. "DTN: A New Partitionable Torus Topology". In *Proceeding of International Conference on Parallel Processing*, Vol. I, pp. 84–91, 1995.

[14] Kai Hwang and Joydeep Ghosh. "Hypernet: A Communication-Efficient Architecture for Constructing Massively Parallel Computers". *IEEE Trans. on Computers*, Vol. C-36, No. 12, pp. 1450–1466, 1987.

[15] J. C. Bermond and C. Peyret. "DeBruijn and Kautz Network: A Competitor for the Hypercube". In *Hypercube and Distributed Computers*, pp. 279–293. North-Hollaud, 1989.

[16] J.Carson. The Emergence of Stacked 3D Silicon and Impacts on Microelectronics Systems Integration. In *IEEE Int'l Conf. on Innovative Systems in Silicon*, pp. 1–8, 1996.

[17] S.Horiguchi. Wafer Scale Integration. In *Proc. 6th International Microelectronics Conference*, pp. 51–58, 1990.

[18] M.J Little, J.Grinberg, S.P.Laub, J.G.Nash, and M.W.Yung. The 3-D Computer. In *IEEE Int'l Conf. on Wafer Scale Integration*, pp. 55–64, 1989.

[19] Michael L.Campbell, Scott T.Toborg, and Scott L.Taylor. 3-D Wafer Stack Neurocomputing. In *IEEE Int'l Conf. on Wafer Scale Integration*, pp. 67–74, 1993.

[20] H.Kurino, T.Matsumoto, K.H.Yu, N.Miyakawa, H.Tsukamoto, and M.Koyanagi. Three-dimensional Integration Technology for Real Time Micro-vision Systems. In *IEEE Int'l Conf. on Innovative Systems in Silicon*, pp. 203–212, 1997.

[21] Michael J. Little and Jan Grinberg. "The 3-D Computer: An Intergrated Stack of WSI Wafers". *Wafer Scale Integration*, pp. 253–318, 1989.

[22] S. Y. Kung, S. N. Jean, and C. W. Chan. "Fault-Tolerant Array Processors Using Single-Track Switches". *IEEE Trans. on Computers*, Vol. 38, No. 4, Apr. 1989.

[23] I. Numata and S. Horiguchi. "Efficient Reconfiguration Scheme for Mesh-Connected Network: The Recursive Shift Approach". *Proc. of the Parallel Architectures, Algorithms and Networks*, pp. 221–227, June 1996.

[24] 井口寧, 松澤照男, 堀口進. "重み付けシフトによる格子結合型ウェーハスタック実装の放熱と再構成". 情処学研報, 2001-HPC-85, pp. 13–18, Mar. 2001.

[25] J.M.Kallis, L.B.Duncan, S.P.Laub, M.J.Little, L.M.Miani, and D.C.Sandkulla. "Reliability of the 3-D Computer under Stress of Mechanical Vibration and Thermal Cycling". In *International Conference on Innovative Systems in Silicon*, pp. 65–72. IEEE Computer Society Press, Oct. 1989.