（講義1）
並列システム概説
理研サマースクール2013

神戸大学システム情報学研究科
小柳義夫

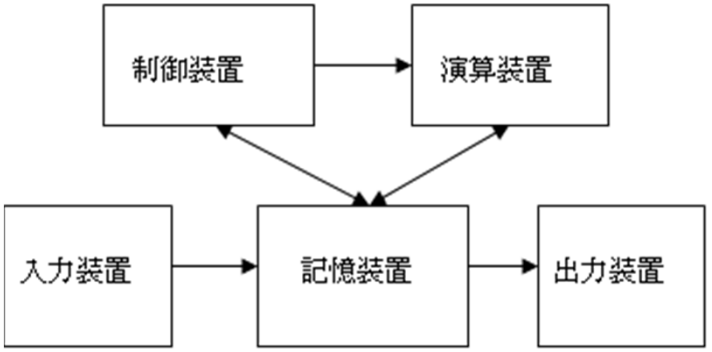
目次

岩波講座『計算科学』別巻
「スーパーコンピュータ」
に基づく
［2012年出版］

岩波講座『計算科学」第1巻
「計算の科学」(2013)

§1.1フォン・ノイマン型コンピュータ

特徴
(1) 記憶装置（メモリ、主記憶）は1個だけあり、1次元アドレスにより規定されるアドレス空間を構成する。
(2) 演算を制御する命令がデータとともに記憶装置に記憶される。
(3) 演算は命令の信号によって実行される。
(4) 命令は、次に実行すべき命令のアドレスを保持する単一のプログラムカウンタ(program counter)により、逐次的に実行される。



「プログラム内蔵型コンピュータ」
「命令駆動コンピュータ」

§ 1.2 仮想記憶

- 現在のほとんどのコンピュータは仮想記憶(virtual memory)方式を採用
- 仮想的な記憶装置を提供
  - プログラムカウンタは「仮想アドレス」を示す
  - 命令のオペランドのアドレスは「仮想アドレス」
  - 物理的な大きさに囚われず（オーバーレイ不要）
  - 不連続なメモリ領域をプログラムから連続に見せる
  - プロセス毎に別のアドレス空間（多重仮想記憶）
  - データの保護（アクセス権を設定できる）

仮想記憶のメカニズム

- プログラムやデータは補助記憶装置（ディスク）に格納される。必要に応じて主記憶にコピーし、補助記憶に戻される。
  - 仮想（論理）アドレスと実（物理）アドレス
  - 変換テーブル（ページ表）はOSが管理
    - 必要なデータが主記憶になければ、OSが補助記憶から読み出して主記憶に置く
    - 主記憶が一杯になれば、一定のアルゴリズムで補助記憶に待避する（変更がなければただ消去）
    - 変換テーブルもデータの一種（待避されることもある）

アドレス変換の高速化

- TLB(translation lookaside buffer)
  - よく使われる論理アドレスについてのページテーブルのデータを保持
  - 格納方式は後述のキャッシュと同じ
  - 保護フィールド、利用ビット、ダーティビットも持つ
  - TLBミス：変換テーブルの一部をTLBにコピー（どこかを追い出す必要）
  - アドレス変換テーブルの変更を反映
  - キャッシュと同じく多重TLBもある（京は2段）
  - 命令用とデータ用で区別することもある

アドレス変換

- ページ（例えば4KB）単位に管理
  - 論理・物理の変換はページ単位
  - 下位12ビット（ページオフセット）を除く上位アドレスを仮想ページ番号という
  - 変換テーブルは「ページテーブル」
- ページフォールト
  - OSは、一定のアルゴリズムで物理メモリのあるページを追い出し（ページアウトという）
  - 必要なページを補助記憶から読み込み
  - 物理メモリのそのページに置く。もちろん、ページテーブルも更新する。

ページサイズの選択

- ページサイズが大きい方が:
  - ページテーブル自体が小さくなる
  - 空間局所性(後述)を活用できる
  - 補助記憶とのやりとりが効率的
  - TLBミスが減る
  - しかし、容量の無駄が生じやすい
- 複数のページサイズをもつプロセッサもある
  - 京のSparc64 viiifxでは、8KB, 64KB, 512KB, 4MB, 32MB, 256MB, 2 GBを指定できる(らしい)。

§1.3 キャッシュメモリ

- 主記憶の容量は急速に増大しているが、アクセス時間はCPUの演算速度に比べて非常に大きい（数百演算に相当）。
- 1命令につき必ずメモリアクセス
  - フォン・ノイマン・ボトルネック
- キャッシュ：高速小容量のメモリ
  - ユーザから直接見えない
  - 機能的にはフォン・ノイマン型コンピュータの基本原理に準拠

キャッシュのデータ格納構造

- アクセスするデータがキャッシュ上：ヒット
  - ヒット率を上げる必要
- 主記憶のどの部分をコピーしておくか
- キャッシュ上にあるかどうかを高速に判定
  - チェック機構を単純に
- どの単位で出し入れ
  - 小さいと転送の回数が増える
  - 大きいと不必要なデータも転送することになる
  - 「ライン」を単位。例えば64B（下位6ビットに対応）

キャッシュのデータ格納構造

- フルアソシアティブ･キャッシュ:ラインより上位のアドレスそのものをキーとして検索。
  キャッシュとしては非現実的
- 実際は上位アドレスを分割する

- フレームアドレスをキーとする。
- エントリアドレスはキャッシュ内のアドレス。
  - ダイレクトマップ方式（1 frame address /1 entry address)
  - k-way set associative cache(k frame addresses/1 entry address)
- キャッシュラインの入れ替えアルゴリズム

キャッシュメモリの動作(data cache)

- メモリとキャッシュの一致をどう保つか
  - CPUがキャッシュに書き込む場合
  - メモリが共有されている場合（後述）
  - 通信(受信), I/O等で、メモリに書き込まれる場合
  - 通信（送信）, I/Oで、メモリから送る場合
- Write throughとwrite back
  - 書き込みの高速性： write back
  - Consistency: write through
- 命令キャッシュは書き込みなし（現在では）

多階層キャッシュ

プロセッサの速度向上

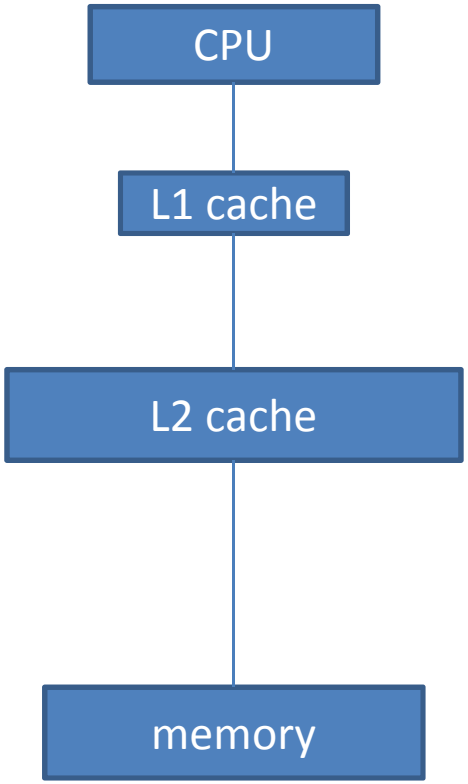
- L1: 高速、小容量。命令とデータと別
- L2: 中速、中容量。命令とデータ共通
- メモリ: 低速、大容量

キャッシュの格納構造

- キャッシュ毎に設定可能

Sparc64 viiifxの例

- Line size: 128B
- L1: 命令・データ別、2-way, 各32KB, write through, core毎
- L2: 共通、12-way, 6MB, write back, 共有



仮想記憶とキャッシュ

- 物理インデックスキャッシュと仮想インデックスキャッシュ
  - 仮想アドレス：一意でない（多重仮想空間）。プロセスの切り替えのたびに追い出す必要。
  - 物理アドレス：一意だが、TLBにより変換が必要
- L1：仮想（高速、小容量で入れ替え負担小）
  L2：物理（大容量、L1アクセスと同時にTLBを引く）
- TLBも多階層のことがある。

京のSparc64 viiifxの場合

- 命令用
  - L1 TLB: 16 entries, full associative
  - L2 TLB: 256 entries, 2-way set associativeとfull associativeから成る（らしい）
- データ用
  - L1 TLB: 16 entries, full associative
  - L2 TLB: 512 entries, 2-way set associativeとfull associativeから成る（らしい）

§ 1.4 メモリ階層と局所性

- Register—L1 cache—L2 cache—L3 cache(if any)—main memory—disk
- 昔は、演算の時間が律速
  今は、データの供給が律速
- できるだけ高速メモリを使えばよい、しかし容量は小さい
- メモリ階層を意識したプログラミングが重要
- しかし、メモリ階層は直接見えない（見たくない）

局所性(locality)

- メモリに複数回アクセスする場合の様式に関する概念
- 時間的局所性(temporal locality)
  空間的局所性(spatial locality)
- 定性的、相対的概念。定量化しにくい。
- メモリ階層を有効に活用するために重要

時間的局所性

- あるデータがアクセス（読み出しまたは書き込み）された場合、近い将来にも同じデータが再びアクセスされる可能性が高いようなアクセスの形式
- このようなデータは高速な記憶装置（レジスタやキャッシュなど）に置くことができる。
- レジスタに置いておけるか：次にアクセスされるまでの時間や、レジスタの使用状況による。
  - 割り付けは高級言語ではコンパイラの任務
- キャッシュに置いておけるか：空間局所性

空間的局所性

- あるデータがアクセスされた場合、近い将来にはその近傍（アドレスから見て）のデータだけがアクセスされる可能性が高く、遠いアドレスのデータがアクセスされないようなアクセスの様式
- キャッシュでは：ラインを単位
  - とくにダイレクトマップの場合、エントリアドレスが同一の違うデータへのアクセス→必ず追い出される
  - 典型：キャッシュサイズの間隔でアクセス
  - この意味でベストは連続アクセス（逐次的局所性ともいう）
- 仮想記憶では：ページを単位にメモリ上にある
  - 実メモリから追い出される可能性
  - TLBをミスする可能性

命令の局所性

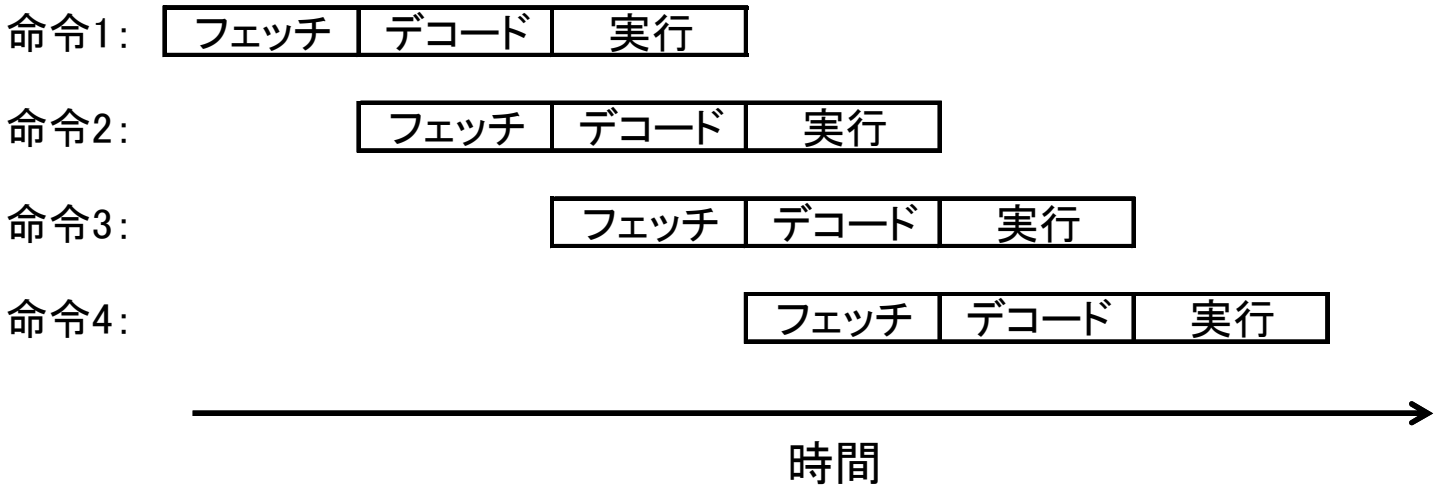
- 実行される命令の集合の様式
- 時間的局所性：小さな反復（短い命令列）を多数回実行
  - 悪い例：長い命令の列を1回だけ実行する
- 空間的局所性：命令キャッシュに収まる領域の中を走る場合
  - 悪い例：しばしば離れたアドレスにジャンプ

§1.5演算順序(高速化手法)

- 基本原理:逐次メモリから命令を取り出して実行する。完全に終わってから次の命令を取り出す
  - あまりに遅い。もっと高速に実行したい。
- 逸脱する手法が発達してきた:ただし、結果はあくまで基本原理に従って実行した場合と同一でなければならない

命令パイプライン

- パイプライン制御(流れ作業)



- 依存関係:演算の結果による分岐など
- 割り込みの処理(エラーなど)

演算パイプライン（ベクトル処理）

- 演算そのものをオーバーラップさせる
  - do i=1, n
  - c(i)=a(i)+b(i)
  - end do

演算パイプライン（ベクトル処理）

- ベクトル処理：1～3個の1次元データの各要素に対する同一の演算を、演算パイプラインにより高速に処理
- 最初に提案：D. N. Senzig and R. V. Smith (IBM), Proc. of AFIPS ‘65, 1965
  最初のベクトル演算器、2938 Array Processor (IBM), J.F.Ruggiero and D.A. Coryell, IBM Systems Journal, 8(1969)118-135

演算パイプライン（ベクトル処理）

- データの供給
  - ベクトルレジスタ（ASC, Star-100, IAP, Cyber203/205, ETA10を除く）
    - レイテンシ隠蔽、時間的局所性
  - バンクメモリ（バンド幅）
  - キャッシュは役に立たない
    - NEC SX-9にはADBという制御可能なキャッシュがある
- 多重ベクトル処理
  - 日本のベクトル計算機のお家芸

多数の命令の同時実行

- 依存性のない演算は、並行して実行できる
  - 命令レベル並列性(Instruction-level parallelism)
  - 現在のコンピュータには多数の演算器が装備
  - メモリアクセスも並行して実行
  - スーパーパイプライン
  - ただし、基本原理に従った場合と同一
- 判断
  - コンパイラ
  - CPU（ハードウェア）

SIMD方式

- 本来Flynnの分類の一つ
  - Single Instruction, Multiple Data
- SSE, VMX, AVXなど
  - 同一の命令(a+b, a*b+cなど）を複数(2～8)個のデータに対して実行
  - ベクトル処理と類似(short vector)
- GPU (graphic processing unit)
  - グラフィックコントローラから発展
  - GPGPU

Out-of-Order実行

- 命令の順序を入れ替えて実行
  - 命令のブロックをバッファに読み込み、デコードを行って待つ。
  - 入力オペランドが得られた順に実行する
  - 一種のデータ駆動計算機
- 基本原理に従って計算したのと同一の結果が得られるよう制御する

分岐予測・投機的実行

- 高速化の邪魔：分岐ー判定に時間が掛かる
  - 予測し、分岐の先まで実行を進める
  - 予測が当たれば「ラッキー」、外れたらリセット
  - 予測の精度にもよるが平均的には得
- 両方の分岐先の実行をあらかじめ並列に進めておき、判定が出た段階で正しい方を採用
  - 投機的実行(speculative execution)：結果を捨ててしまうかもしれない命令を実行すること
  - 有り余るトランジスタの利用法の一つ

§ 1.6 マルチコア

- 複数のCPUを搭載したチップ
  - 個々のCPUをコア(core)という。「磁気coreではない」（このジョークの分かる人は老人）
  - 「コアの数は18ヶ月毎に倍増する」（新しいMooreの法則）
  - チップ自体をCPUと呼ぶこともあるが、よくない。せめて「CPUチップ」と（メモリやNWと区別）。あるいは、「ダイ」「ソケット」（ニュアンスが違う）
- 現在は16コア程度が最高。
  - BlueGene/Qは18コア（1つは予備、1つはOS専用）
  - このノートパソコンはdual core
  - 大学で使っているデスクサイドPCはoctacore

メニーコア(manycore)

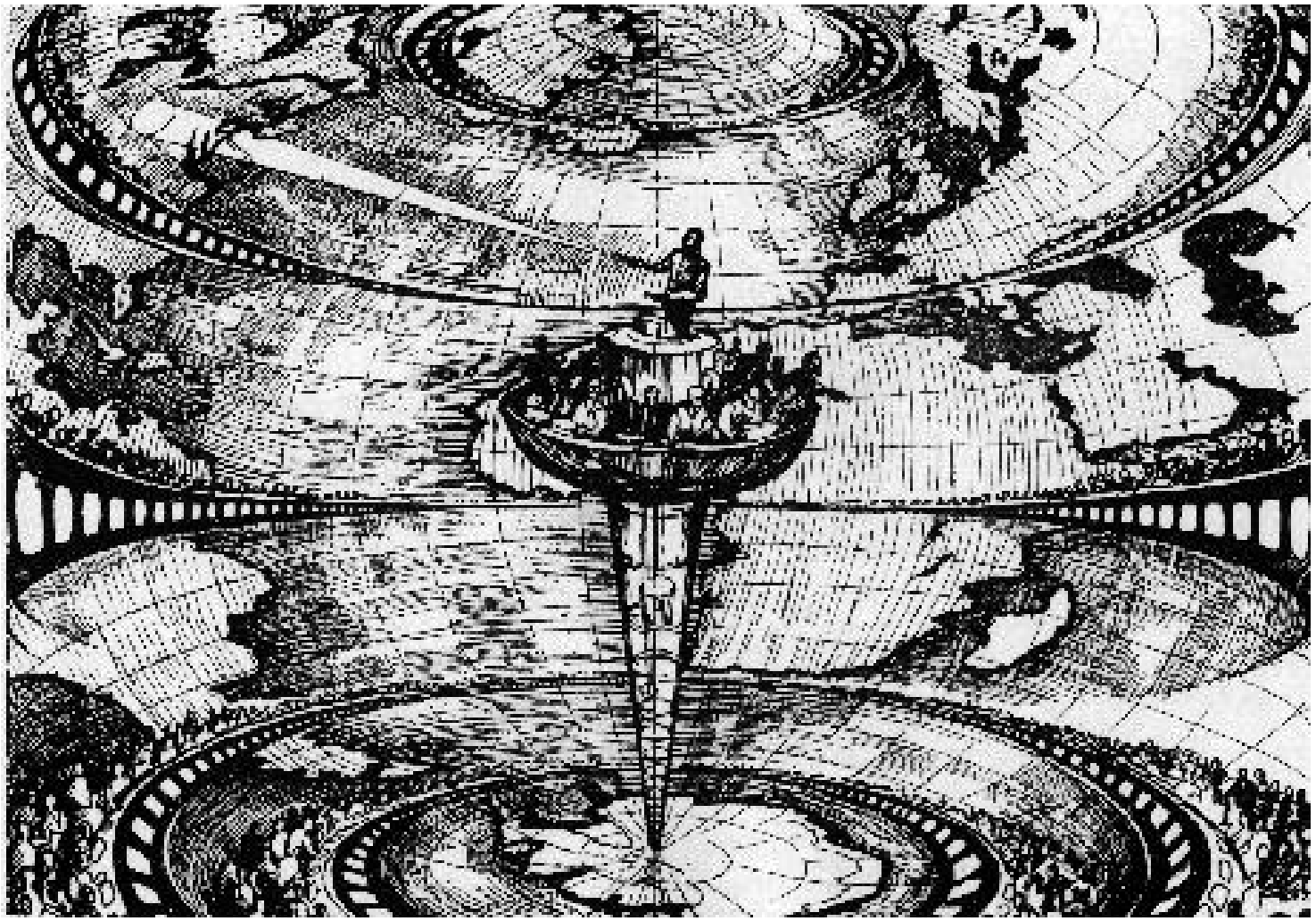
- 機能を制限したCPUを多数組み込んだチップ
  - OSの主要部が動く程度
  - GPUなどのアクセラレータは演算のみ。
  - 同一コア(MIC arch.など): Intel Xeon Phi
- 問題点
  - メモリバンド幅が相対的に減少
  - Local memoryかキャッシュか
  - 3次元チップ

§ 1.7 並列処理

- 並列処理(parallel proc.)と並行処理(concurrent proc.)
- 歴史
  - L.F.リチャードソンの夢(1922)
    - リチャードソン加速（補外）も彼による。建部賢弘は200年前に。
  - ENIACでも加減算と乗算の同時処理
  - EDVAC報告書では、非現実的
- 広義の並列処理
  - 前述のCPU内高速化技術全体も一種の並列処理
  - 狭義には複数のCPUが並列動作の場合

Richardson's Forecast Factory

http://mathsci.ucd.ie/~plynch/Dream/ForecastFactory/Lannerback.jpg



Richardson's Forecast Factory

http://mathsci.ucd.ie/~plynch/Dream/ForecastFactory/SchuitenHD3.jpg

ベクトルか並列か？

- 1960年以来の高性能計算の対立項
  - 単独（少数）の高性能ベクトルで実現（少数精鋭）
  - 比較的低い性能のプロセッサを多数（人海戦術）
- 半導体技術の進歩
  - 対立項の解消：結局「多数精鋭」でなければならない
  - さきほど、ベクトル処理をCPU内の高速化技術として位置づけた
  - 現在では、マルチコア、メニーコアの時代

§ 1.8 並列性

- 並列性：複数の処理が原理的に同時に実行可能
- 結合則、分配則によって並列性を見いだす
  - カスケード演算
    - （分散メモリの場合）全部のノードで総和がほしい
    - 浮動小数演算の丸め誤差
    - バタフライ加算
    - Sparc64 viifx/viiifxの演算器付きネットワーク
  - Recursive doubling
    - 漸化式の並列化

並列処理モデル（マクロな並列性）

- Master-workerモデル (EP, embarrassingly parallel)
  - Worker同士に通信なし
  - Load balanceが問題
- データ並列
  - 配列の各要素にほぼ同一な演算を行う
  - Loop-level parallelismとも呼ばれる
  - ベクトル処理やSIMD演算が得意
  - データ配置の問題

並列処理モデル(マクロな並列性)

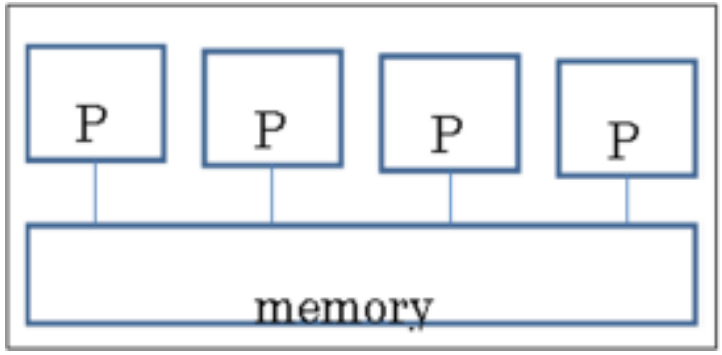
- タスク並列
  - プロセッサ毎にタスクを割り当てる
- 領域分割法
  - 偏微分方程式に対する反復法が典型
  - 依存関係は、アルゴリズムによる
  - 加法的Schwarz領域分割
  - 内点消去領域分割法
- 粒子分割法
  - 多粒子問題は領域分割でも可能

メモリ・アーキテクチャ

- 大きく二つに分類
  - 共有メモリ
    - 対称型マルチプロセッサ
    - 分散共有マルチプロセッサ
  - 分散メモリ
- プログラミング・モデルとしても意味をもつ
  - 両者は独立
  - 共有メモリモデルでプログラムして、分散メモリのコンピュータの上で走らせる（HPFなど）
  - 逆もある

対称型マルチプロセッサ

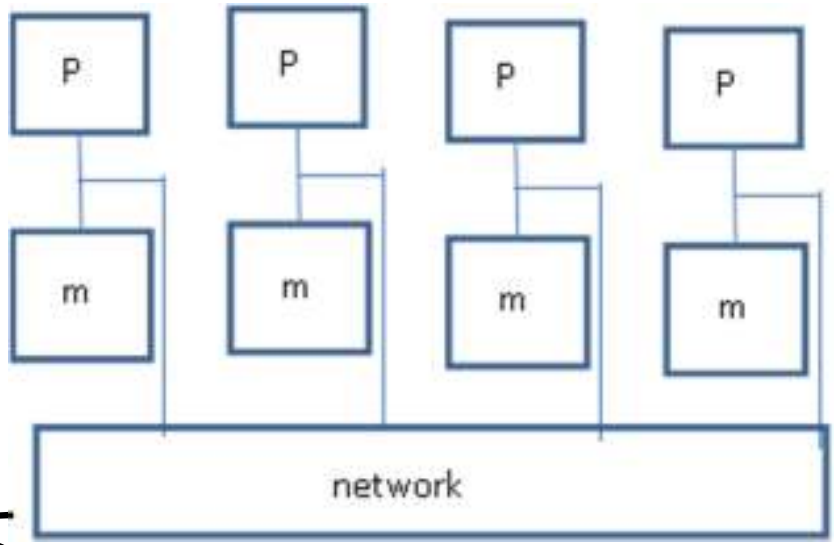
- メモリの任意の場所が、どのプロセッサからも等距離（原理的に）
- メモリはプロセッサ数の口をもつ必要
  - 排他制御が必要
  - メモリバンド幅の維持



- 現在ではSMPは、symmetric multiprocessorよりも、shared memory processorの略号として使われることの方が多い。

分散共有メモリ(Distributed Shared Memory)

- 各プロセッサは固有のメモリをもっているが、他のプロセッサからもアクセス可能。
- NUMA (Non-Uniform Memory Access)とも呼ばれる。対称型はUMA
- 現在では、対称型でも完全に同一の時間でアクセスできるわけではないので、両者の区別は曖昧に。
- 各データをどこのメモリに置くかが重要
  - First touch, memory affinity, ….

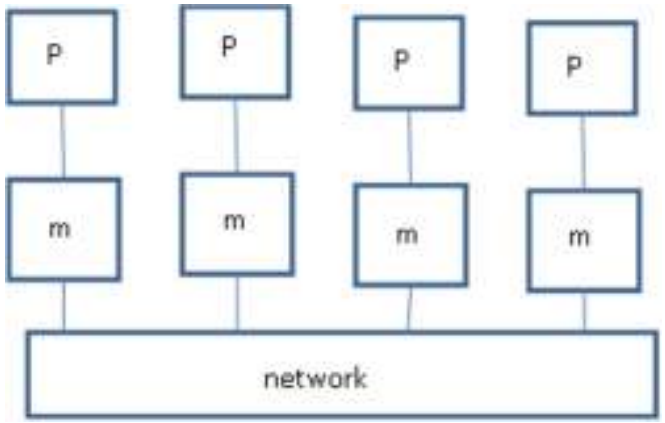


キャッシュ・コヒーレンシ

- 共有メモリ型のキャッシュ
  1. 個別のプロセッサが独占するキャッシュ
  2. 一部のプロセッサが共有するキャッシュ
  3. 全体で共有するキャッシュ
- メモリ上の同一のデータのコピーが複数のキャッシュに存在する場合
  - あるプロセッサが書き込むと、不整合
  - Write throughでもwrite backでも同様
- 不整合を起こさないこと:コヒーレンシ
  - Update方式（無駄）
  - Invalidate方式
  - バススヌーピングとディレクトリ方式
- ccNUMA

分散メモリ型並列コンピュータ

- 複数台のコンピュータをネットワーク（相互接続網）で接続
- 通信モデル
  - メッセージパシング
  - リモートDMA（ユーザレベル通信、ゼロコピー通信）
  - リダクション通信



- バンド幅、遅延、レイテンシ、倍セクションバンド幅
- ハイブリッド並列処理

§ 1.9 並列処理性能評価指標

- 速度向上率

$$S(p) = T(1)/T(p)$$

- 並列処理効率

$$E(p) = S(p)/p$$

- Amdahlの法則

$$S(p) = \frac{1}{1-\alpha+\alpha/p} < \frac{1}{1-\alpha}$$

並列処理性能評価指標

- Gustafsonの法則

$$S(p) = p - \beta(p-1)$$

- Amdahlの法則との関係

$$\beta = \frac{1-\alpha}{1-\alpha+\alpha/p}$$

- スケーリング
  - いくつかの条件を固定したとき、ある量が他の量に一定の範囲でほぼ比例する。
  - 強いスケーリングと弱いスケーリング

並列化で速度が向上しない理由

以下の理由は独立ではない

- 並列化できない処理
  - とくに、初期化や入出力
- 通信（隠蔽できない場合）
  - とくにリダクション通信
- 同期
  - できるだけ減らしてパイプライン処理
- 負荷の不均衡
- アルゴリズムの不適切

§1.10 並列コンピュータの歩み
milestoneに注目

- 1 MF: CDC6600 (1964, pk 4 MF)
  10個のfunctional units. LLNLに納入
  clock 10 MHz, add 4 clocks, mult 10 clocks(x2)
- 10 MF: CDC7600 (1969, pk 36 MF) 命令パイプライン。
  FACOM 230-75 APU (1977, pk 22 MF) NAL
- 100 MF: Cray-1 (1976, pk 160 MF)演算パイプライン
  Illiac IV (1976, pk 150 MF) SIMD並列処理(64) デコーダは1個
  HITAC S810/20, FACOM VP200など

並列コンピュータの歩み(2/5)

- 1 GF: Cray X-MP/4 (1984, pk 1.26 GF)
  Cray-2 (1985, pk 1.95 GF)
  VP-400 (1985, pk 1.14 GF)
  SX-2 (1986, pk 1.3 GF)
  SX-2はLivermore Loop No. 7 (Equation of States)で1.042 GFlopsを実測
  X-MP/4では、最大0.807 GFlops (No. 7) を達成。
- 10 GF: ETA-10 (1987, pk 10 GF) 液体窒素冷却
  ほとんど安定に作動せず
  PHI (1989, pk 10 GF)「スパコン大プロ」評価後解体
  SX-3/44R (1990, Lp 23.2 GF) NEC社内
  C916/16256 (1992, Lp 13.7 GF) Cray社内
  CM-5/1024 (1993, Lp 59.7 GF) LANL
  S-3800 (1993, Lp 27.5 GF) 東大

並列コンピュータの歩み(3/5)

- 100 GF: NWT (1993, Lp 124 GF) NAL
  Paragon XP/S140 (1993, Lp 143.4 GF) SNL
  cp-pacs (1996, Lp 368.2 GF) 筑波大

  Petaflops計画始まる(1994)
- 1 TF: ASCI Red (1997, Lp 1.068 TF→2.379) SNL
  ASCI Blue Mountain (1998, Lp 1.608 TF) LANL
  ASCI Blue Pacific (1998, Lp 2.144 TF) LLNL
  ASCI White (2000, Lp. 7.304) LLNL

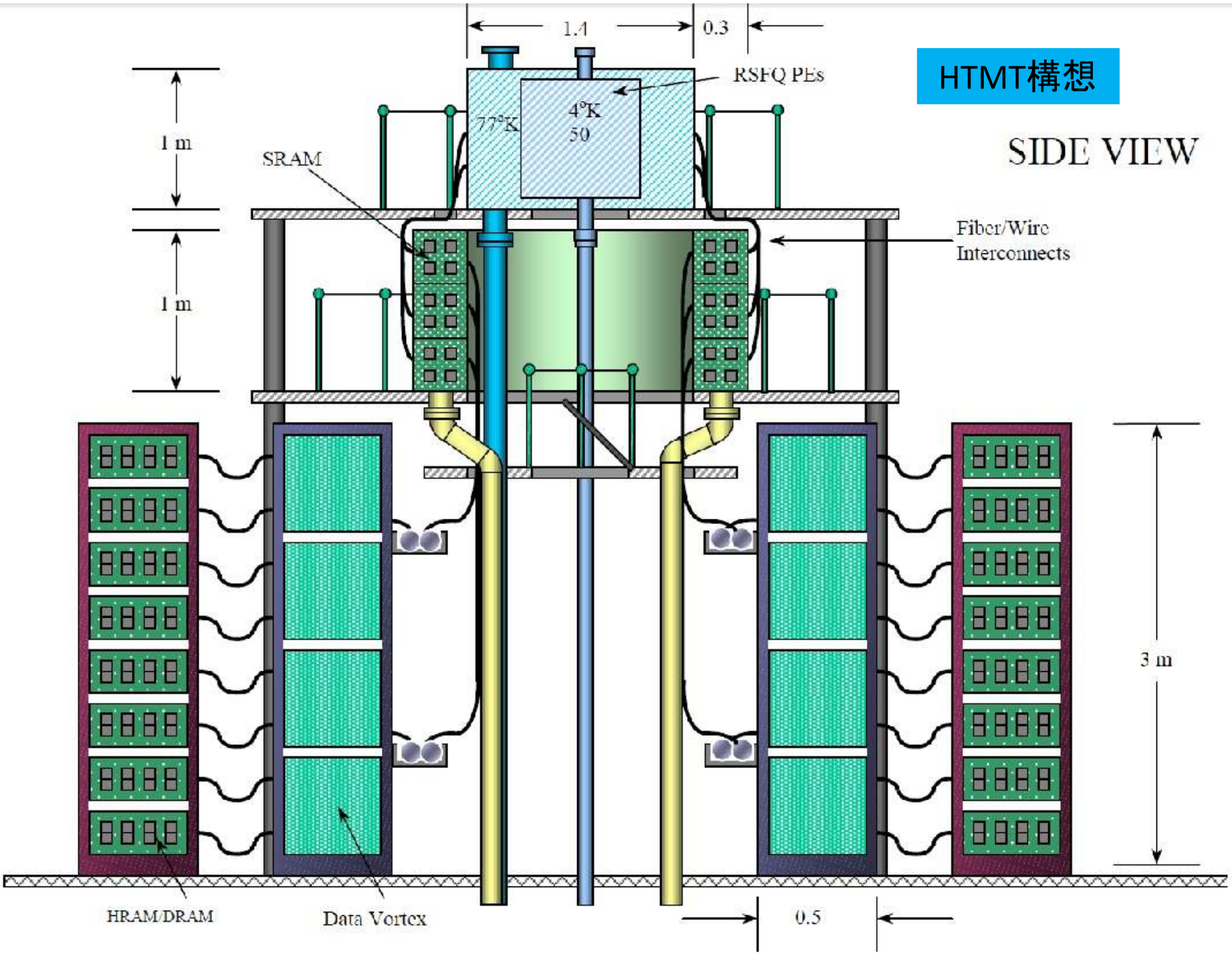
Petaflopsへの道

- 1994/2, “Enabling Technologies for Petaflops Computing” (Pasadena)
  - 当時はNWTが100 GFを越えた頃。10000倍へ！
  - Seymour Crayが基調講演（当時Cray-4を発表）
- その後、7回のWS
- SC96でPetaflops Computingのパネル
  - 2010年頃を想定
  - 応用として、実時間3D心臓モデルが例示。その他、新材料、ナノ、VR、バイオ、プラズマなど
  - 要素技術としては、latency hiding, million threads, 消費電力（1～3 GWを想定する人も。2-3 MW/PFが主流）、費用（$100M～1B）、信頼性、プログラムできるか、など。
  - 何となく今のExaの議論を彷彿させる

1999/2, “PETAFLOPS II” (Santa Barbara)

- Tilak Agerwala (IBM)
  - Powerの延長でPetaflopsができる
- Peter Kogge (Notre Dame)
  - PIM (Processor-in-Memory)ならlatencyは1/10, バンド幅は100倍
  - でもPIM間、PIM-nonPIM間の接続はどうする。
- Thomas Stirling (当時CalTech)
  - HTMT（超伝導、PIM、光接続、ホログラム記憶などを組み合わせる）
  - 超伝導なら、250nmで200GHzが実現できる

HTMT構想


SIDE VIEW
1.4
0.3
RSFQ PEs
77°K
4°K
50
SRAM
1 m
1 m
Fiber/Wire
Interconnects
3 m
HRAM/DRAM
Data Vortex
0.5

アメリカの動き

- HEC Revitalization Task Force
  - 2003/3：米国政府は国家科学技術会議（NSTC）の下に特別プロジェクトとしてこのタスクフォースを編成（ES対抗策か？）
- 2004/11：High-End Computing Revitalization Act of 2004成立 (2005fy-$50M, 2006fy-$55M, 2007fy-60M for DoE)
- 2004/10：Zettaflops Project (実際はExaFlops)が始まる
- 2005/3：High Performance Computing Revitalization Act of 2005 (H.R. 28) was approved by House Science Committee
  - 4月、House通過
  - NSFを中心に、NASA, NOAA, DoE, NIST, EPA

日本の動き

- 1990年代、アメリカはPetaflopsに向けて動いていたが、日本の動きは緩慢
- 2002年に地球シミュレータが驚異的性能
- 2004/5：文部科学省情報科学技術委員会にWGを設置し、11回の会合を行う
- 2005/8：「計算科学技術の推進方策」
- 2005/6：総合科学技術会議「科学技術基本政策策定の基本方針」に次世代スーパーコンピューティング技術を入れる

日本の動き

- 2005/7：スーパーコンピュータ推進議員連盟
  - 座長：尾身幸次、副座長：安倍晋三、事務局長：後藤茂之、100人以上集まる
  - 8月、政府に勧告を提出
- 2005/7：「最先端・高性能汎用スーパーコンピュータ開発利用」プロジェクトが正式決定
  - 2006～2012年度、総額1154億円、Lp 10 PF
  - 次世代スーパーコンピュータの開発
  - 応用ソフト（ナノ、バイオなど）の開発
  - 研究教育拠点の形成

日本の動き

- 2005年度～7年度、要素技術の研究開発
  - 総額40億円／年
  - システム相互接続（九大、富士通）
  - IPによる相互接続（東大、慶応等）
  - 低消費電力素子（日立、東大、筑波大）
  - CPUとメモリの光接続（NEC、東工大）
- 2005/8：計算科学推進WG中間報告（>10 PF)
- 2005/8：文部科学省が概算要求
- 2005/10：理研を開発主体とする
- 2005/9, 10：総合科学技術会議の評価検討会

Requirements from these Applications

1. Large-scale processing part
   - Although never explicitly stated, this part is believed to be a vector/pseudovector computer.
   - 2 PF from disaster prevention
     1 PF from drug design
     0.2-0.6 PF from various fields

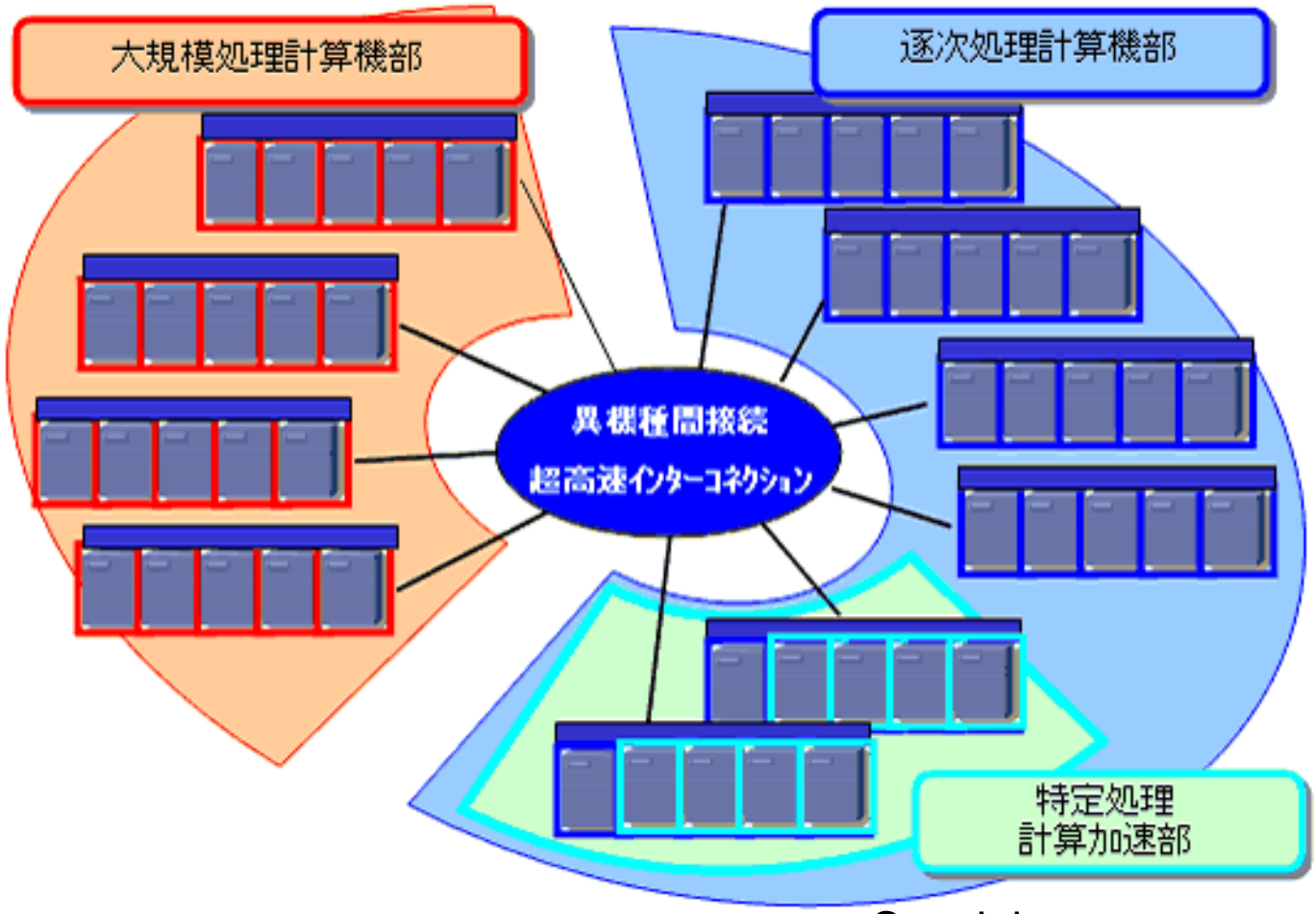
Requirements from these Applications

2. Scalar computer part
   - 4 PF from device simulation with electron correlation
   - 0.3-0.5 PF from various fields
3. Special purpose computer
   - 20 PF from drug design (MD)
   - 20 PF from astrophysics

Original Proposal

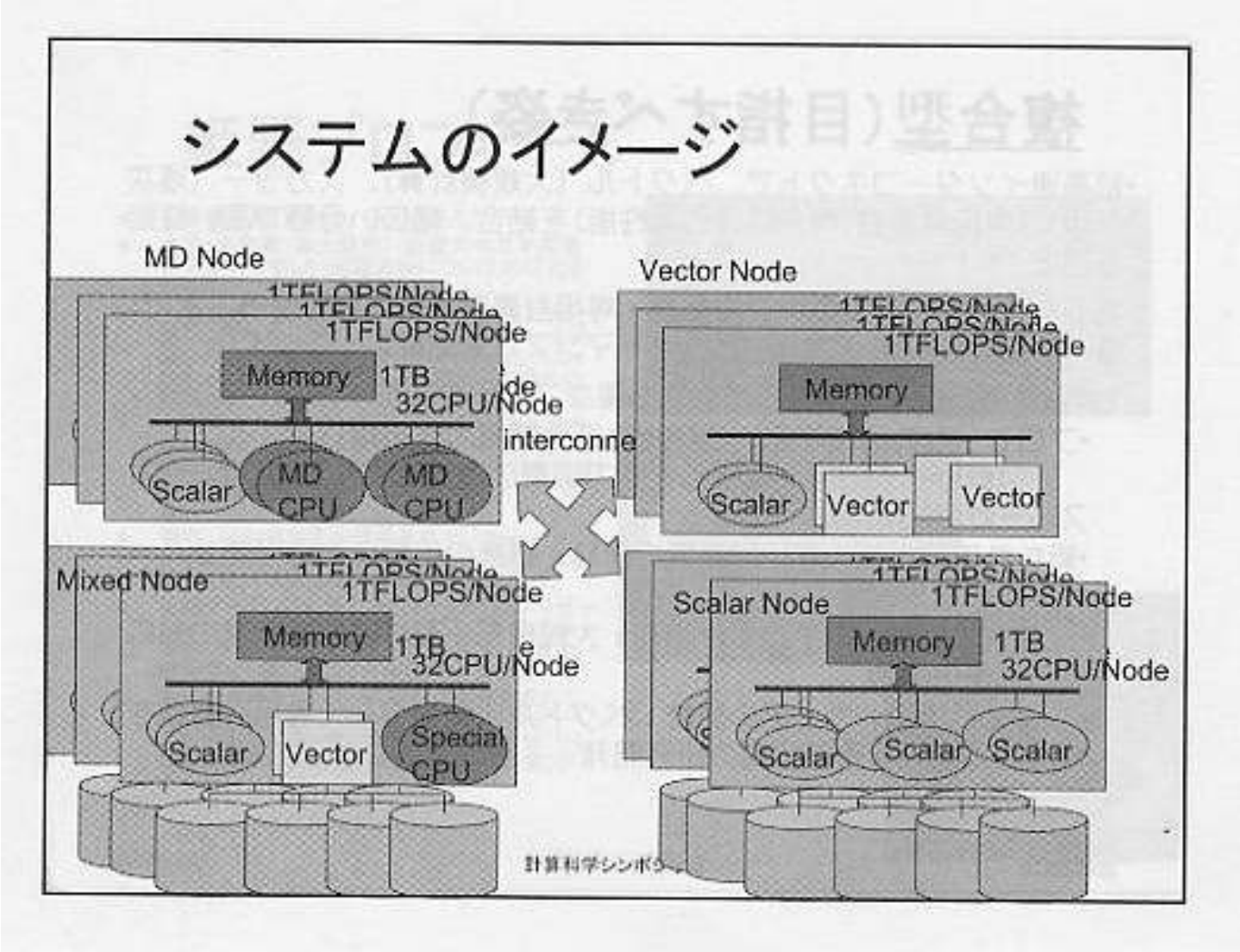
Large scale processing

Scalar computer

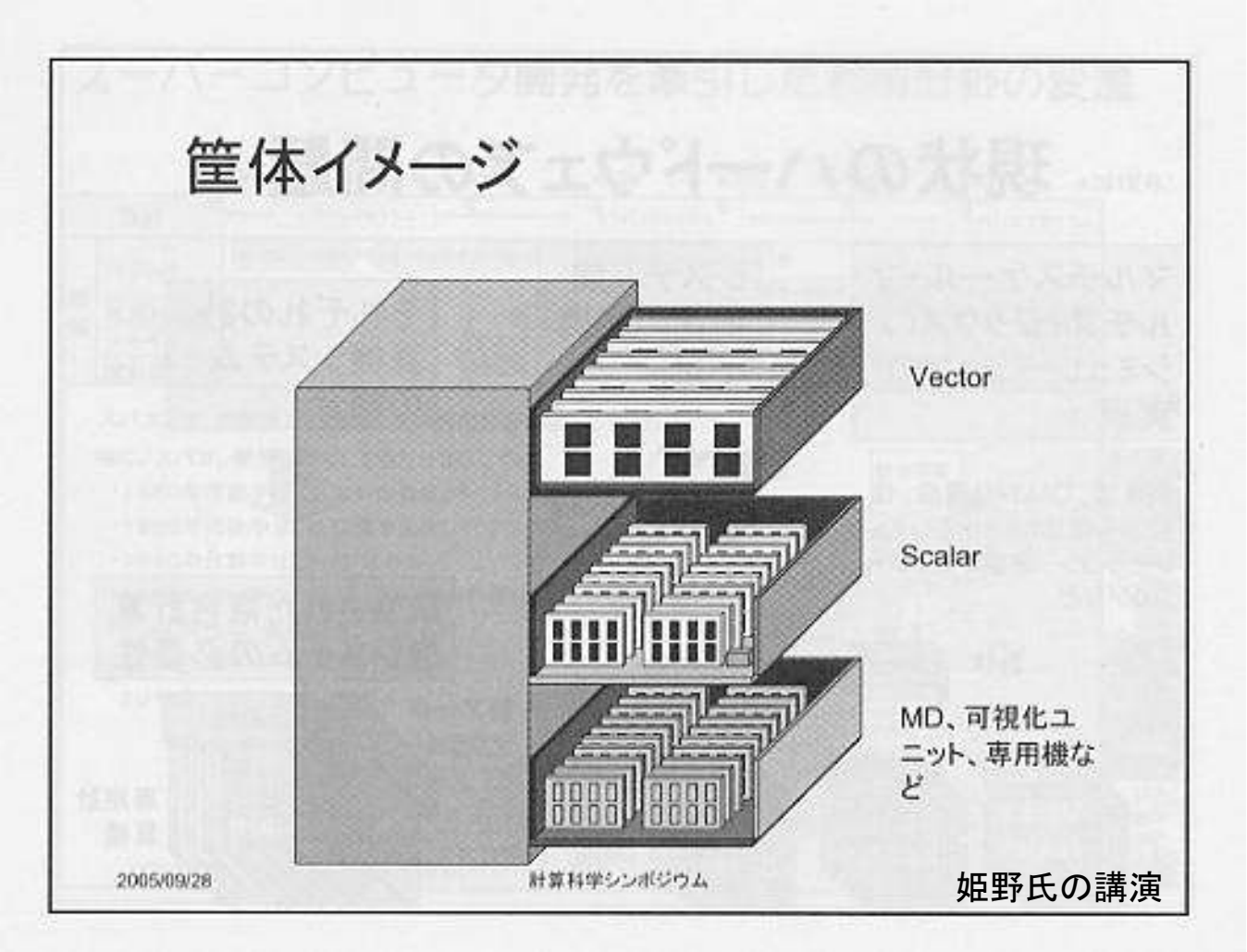


Special-purpose computer

Proposed System Image



Possible Rack Image



姫野氏の講演

日本の動き

- 2005/9-10：評価検討会
  - 「汎用性だけを目標としては使える計算機はできない」I先生や私が強調。
  - 「ぼろくそに言われた」（牧野）
  - ターゲットアプリをいくつか選定し、性能目標を設定し、それを実現するアーキ、基盤ソフト、ミドルウェアを設計する方法論が主張された
- 2005/11：総合科学技術会議で「実施することが適当」と報告
- 2006/1：文部科学省に推進本部
  - 21本のターゲットアプリを選定

日本の動き

- 2006/3：第三期科学技術基本計画、閣議決定。スーパーコンピュータを国家基幹技術と位置付け
- 2006/7：共用促進法を改正、立地委員会設置
- 2006年度：「次世代スーパーコンピュータ概念構築に関する共同研究」（10以上の提案でコンペ）
- 2007/3：神戸に決定
- 2007/4：理研からシステム構成案
  - スカラ演算部（富士通担当）、ベクトル演算部（NEC、日立担当）から成る複合システム。合計>10 PF
- 2007/9から詳細設計、2009/4から中間評価

日本の動き

- 2009/5/13にNECが製造段階への不参加。スカラ部だけの構成とする
- 2009/7：文科省、5戦略分野を決定
- 2009/11/13：行政刷新会議の「事業仕分け」第3分科会
- 見送りに近い縮減→必要な改善を行いつつ推進

2009年11月13日金曜日

行政刷新会議事業仕分け第3ワーキンググループ

「世界一になる理由は何があるんでしょうか？」

「2位じゃダメなんでしょうか？」

票決:
廃止 1
見送り 6
縮減 5

結論
限りなく見送りに近い縮減

1 PFにどうやって到達したか

- Roadrunner (2008, Lp 1.026 PF), LANL
  - Cell processorは一種のPIMといえるか？
- Jaguar (2008, Lp 1.059→1.759) ORNL
  - Opteronのhomogeneous system。在来型の延長？
- Nebulea (2010, Lp 1.271 PF) Shenzhen深圳市
  - Intel+NVIDIA GPU
- 天河1A (2010, Lp 2.566 PF) 天津
  - Intel+NVIDIA GPU
- Tsubame2.0 (2010, Lp 1.192) 東工大
  - Intel+NVIDIA GPU

日本の動き

- 2010/2：NEC、ベクトルコンピュータの開発継続を表明
- 2010/7：理研、愛称「京」を決定。AICS設立。
- 2010/11：東工大、TSUBAME2.0：1.192 PF
- 2011/3：京の一部が稼働開始
- 2011/6：京が8.162 PFでTop500の1位
- 2011/11：京、10.51 PF。Gordon-Bell賞
- 2011/4：登録機関、HPCIコンソーシアム
- 2011/5：京の利用課題募集開始
- 2011/6：京、正式に理研に引き渡し

10 PFの壁

- 京(2011, Lp 8.162→10.51) 理研
  - Sparc64 viiifx and Tofu interconnect
- Sequoia (2012, Lp 16.32→17.17 PF) LLNL
  - BG/Q
- Titan (2012, Lp 17.59 PF) ORNL
  - Opteron+Gemini interconnect
- 天河2号(2013, Lp 33.86) NUDT→広州
  - Intel Xeon+Phi
  - 2015年に100 PF（peak?)を目指す

ExaScaleへの挑戦

- Linpack 1 EFは可能か？
  - メモリ100 PBとすると、n=10^8の行列が入る。これを1 EF (10^{18} flops)で実行すると、$2/3 \times 10^{24}/10^{18} = 7 \times 10^5$秒＝7.7 日
  - 1週間安定に動作するとは思えない
  - もっと小さいnでEFが出れば別であるが。

Exascaleへの挑戦

- Zettaflops activity (SNL)
  - 1990's : Petaflops
  - 2004 : Zettaflops (extreme forward-looking focus)
- The Path to Extreme Supercomputing
  - 2004/10 Santa Fe, 30 people
  - Erik P. DeBenedictis, SNL, organizer
  - P. Jones, P. Kogge, W. Gropp, M. Frank, C. Lent
  - Panel: T. Stirling, H. Simon, T. Michalske, F. Johnson, W.Campなど
  - ちょうど地球シミュレータがトップを譲った頃

Exascaleへの挑戦

- Horst Simonの2007/10の展望
  http://www.zettaflops.org/fec07/presentations/Monday-1330-Simon7Challenges.pdf
  1. No R&D program beyond Petaflops
  2. Cannot afford EF until 2019 at current budget levels
  3. Cannot afford the power requirements
  4. Productivity for science not adequately addressed (data tsunami)
  5. Application at 100K way parallel is hard
  6. How to express parallelism
  7. System software??

Exascaleへの挑戦

- アメリカ：DARPA, NSF, DOE
- IESP (International Exascale Software Project)
  - SC08 (Austin)で発足、Santa Fe, Paris, つくば, Oxford, Mauii, San Francisco, 神戸(2010/4)
- 2010/8：DARPA's Ubiquitous HPC Program selected --NVIDIA-Cray-ORNL troika, Intel, … 終了？
- EESI (European Exascale Software Initiative)：
  - 2010/6に発足、18ヶ月、4 WG's
  - 2010/11にAmsterdamで第1回会合
- 2011/1：Obama--State of the Union Address
  - Supercomputingに重点

Exascaleへの挑戦

- 日本
  - 2008～：IESPへの参加
  - 2010/8～：SDHPC WS
  - 2011/7：WG（アプリ、システムソフト・アーキ）
    - 若手中心。老人はアドバイザとして棚上げ
  - 2012/2：次世代HPCI-WG（2年の予定）
  - 2012/3：WG報告書
    - http://www.open-supercomputer.org/workshop/sdhpc/
  - 2012：FS (アプリ、3アーキ）
  - SC12:日米システムソフトウェア共同研究
  - 2013：日米科学技術協定

Exascaleへの挑戦

- 日本（続き）
  - 2013/5：WG中間報告案、パブリックコメント
  - 2013/6/25：正式な中間報告
    - 今後のHPCI計画推進の在り方について（中間報告）
  - 2013/7/2：今後のHPCI計画推進のあり方に関する検討ワーキンググループ システム検討サブワーキンググループ
  - その他のワーキンググループ？
  - 2013/8：概算要求？

Exascaleへの挑戦

- Russia :
  - 2012/4 : T-Platforms 6, IBM 17, HP 16, others 11
  - 2010 : 自主開発への動き($37M, 2010)
  - National Exascale effort, $1.5 b over several years
    人材育成、ミドルウェア開発、応用開発、相互接続、プロセッサなどをすべて含む
  - the Russian Academy of Sciences, T-Platforms, and RosAtom, the state controlled Russian nuclear regulatory body（応用は限定的）
  - 2014/15にIntel+ NVIDIAで10+ PFを狙う
  - 2017/18に自主開発プロセッサ(?)で100 PF
  - 2020頃のExascaleは自主プロセッサ

§1.11 エクサフロップスに向けて

- メモリバンド幅の壁
  - チップ内メモリなら高速アクセス可能
  - キャッシュ（またはその変種）で済むか
  - プログラマブル・メモリ：フォン・ノイマン型コンピュータの基本原理からバイバイ
- 消費電力の壁
  - 数pj/flop　データ移動の方が電気を食う
- 故障率の壁
  - 故障しても動けるコンピュータ

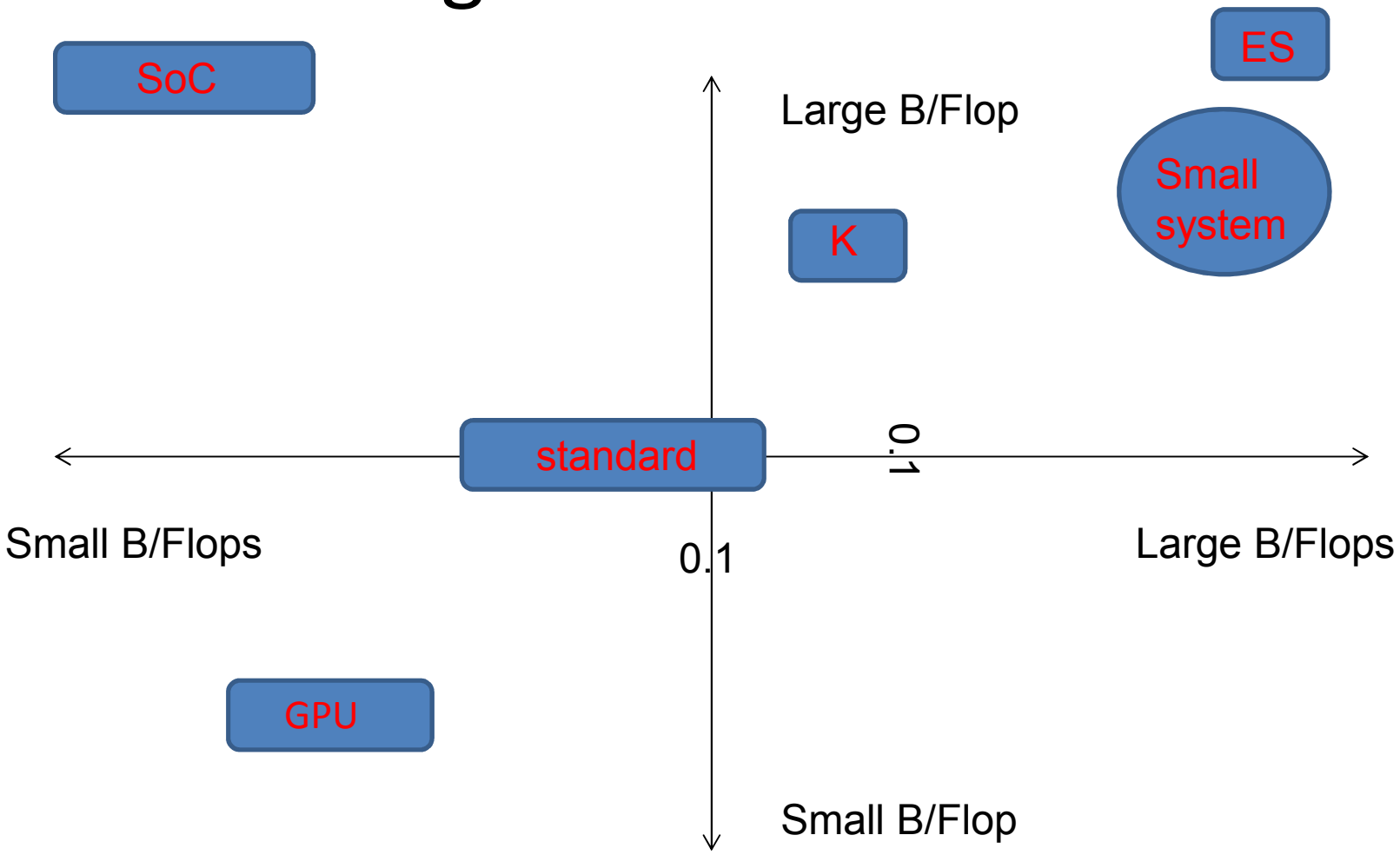
Memory and memory B/W

- 地球シミュレータ
  - 4 B/Flop and 0.25 B/Flops (10 TB for 40 Tflops)
- The K Computer (single node)
  - 64 GB/s for 128 Gflops---0.5 B/Flop
  - 16 GB for 128 Gflops---0.125 B/Flops
- Standard EXA
  - 0.1 EB/s for 1 Eflops---0.1 B/Flop
  - 10-100PB for 1 Eflops---0.01-0.1 B/Flops
- Limitation
  - Cost and power
  - Programmability

今後のHPCI技術開発に関する報告書

- 計算科学ロードマップ白書
  http://open-supercomputer.org/wp-content/uploads/2012/03/science-roadmap.pdf
- **HPCI** 技術ロードマップ白書
  http://open-supercomputer.org/wp-content/uploads/2012/03/hpci-roadmap.pdf

Memory and memory bandwidth
Big technical issue



最後に

- Von Neumannアーキテクチャにもかかわらず、並列処理はコンピュータの初期から利用されていた。
- 現在では、core内、chip内、chip間、node間の並列性が存在。
- 並列性を活用するソフトウェアは今後の発展が期待される。いつまでMPIか？
- エクサフロップスでは、10^8以上の並列性を活用する必要がある。