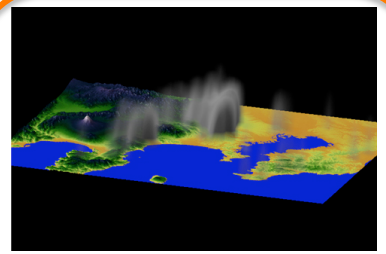
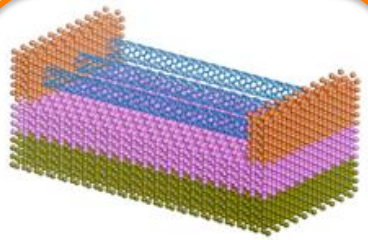
次世代スーパーコンピュータ:
アプリケーションの高性能化に向けて

2010年3月24日

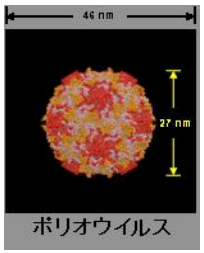
理化学研究所
次世代スーパーコンピュータ
開発実施本部　開発グループ
アプリケーション開発チーム　南



難しい大気現象の解明、また正確な台風の進路・強度の予測による気象への貢献



ポスト35nm世代のデバイス全体のシミュレーションによるエレクトロニクスへの貢献



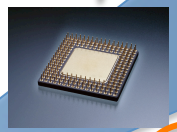
水中のウィルスの丸ごとシミュレーションによる医療への貢献

10^{21}m

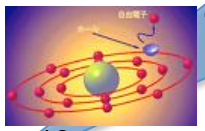
10^{7}m

10^{2}m

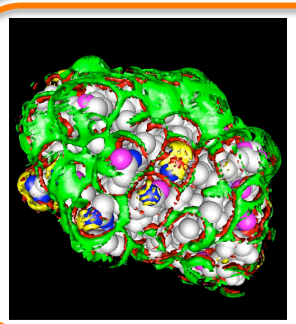
10^{0}m



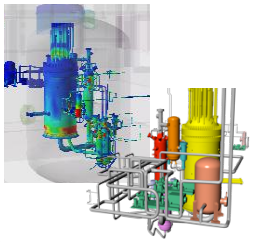
10^{-8}m



10^{-10}m



セルロース分解酵素のシミュレーションによる安価なバイオ燃料の提供等エネルギーへの貢献



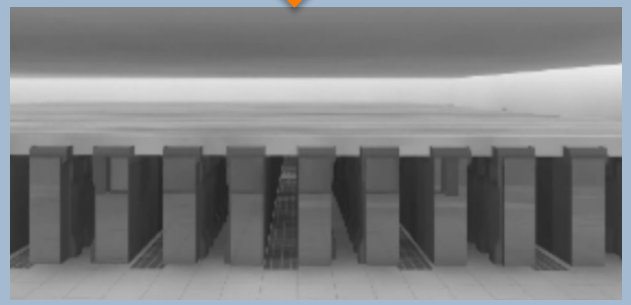
短周期地震波動の地震波シミュレーション、構造物の耐震シミュレーションを組み合わせた防災への貢献

現代のスパコン利用の難しさ

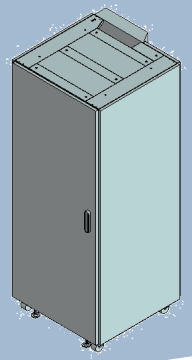
アプリケーションの
超並列性を引き出す

プロセッサの単体性能
を引き出す

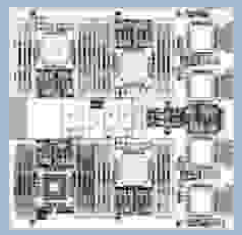
System



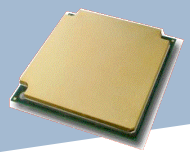
Rack



System Board

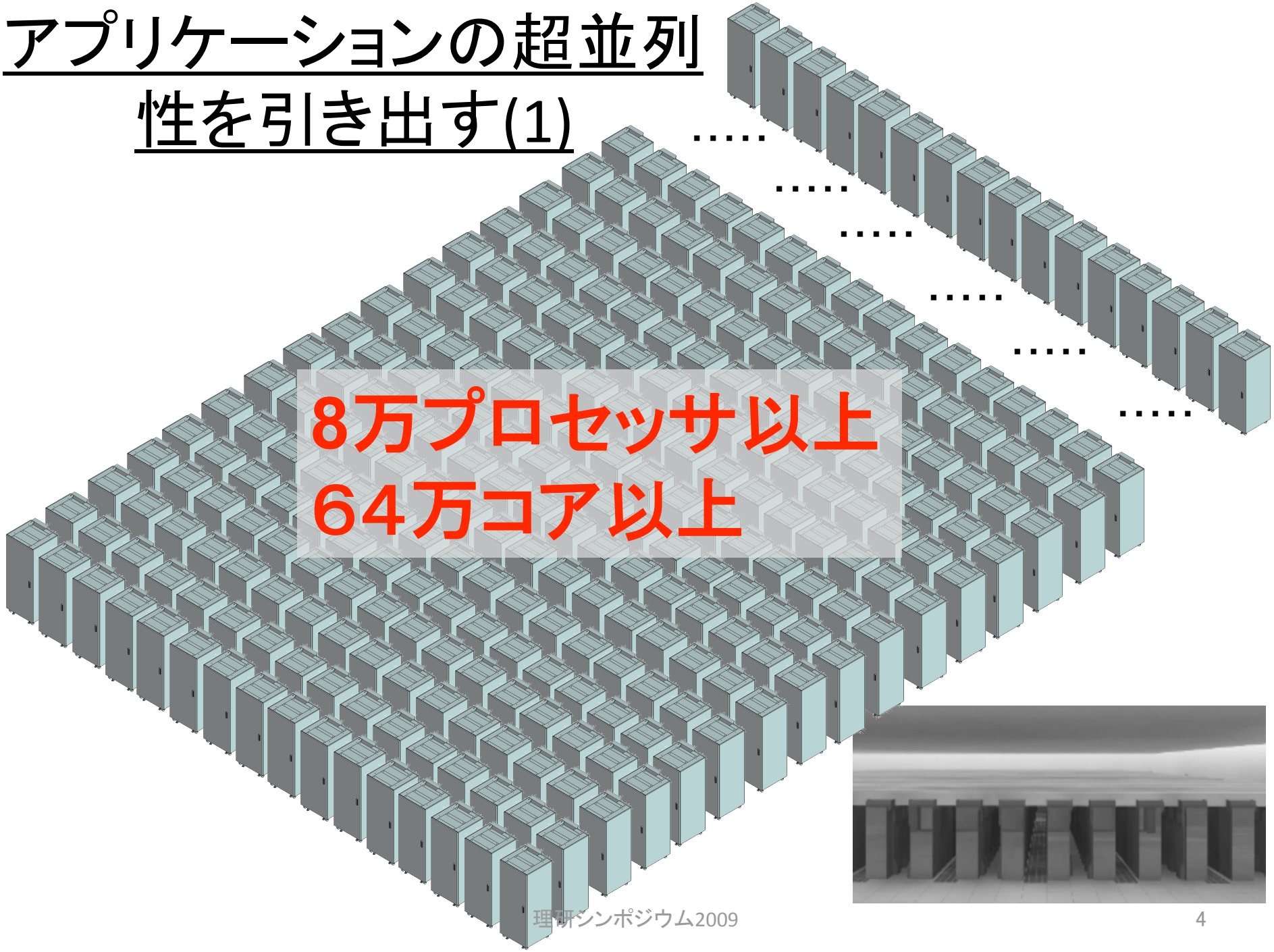


CPU



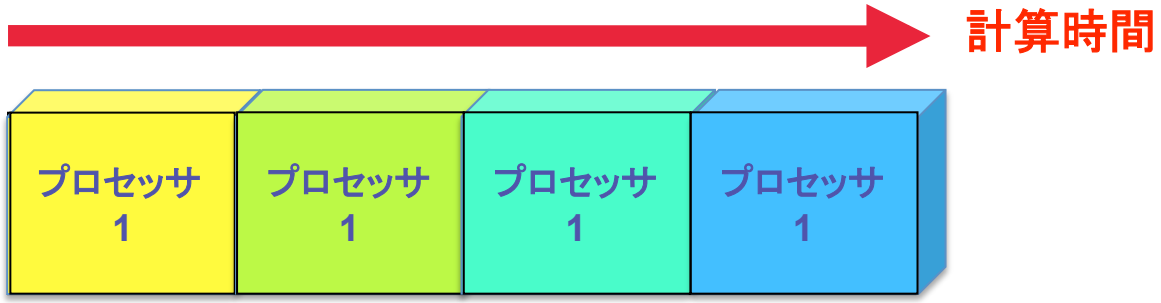
画像提供：富士通株式会社

アプリケーションの超並列性を引き出す(1)



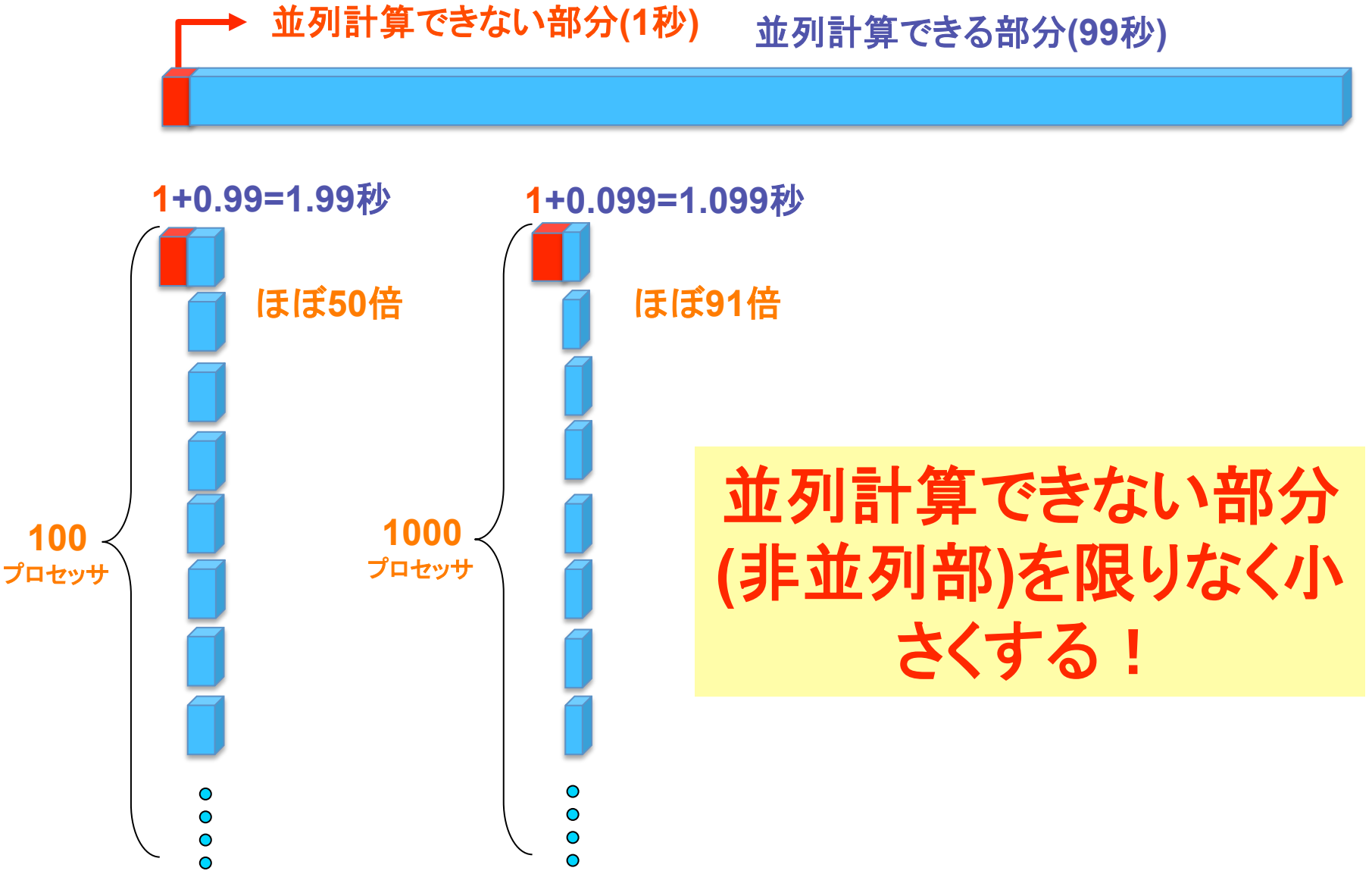
アプリケーションの超並列性を引き出す(2)

逐次計算



並列計算

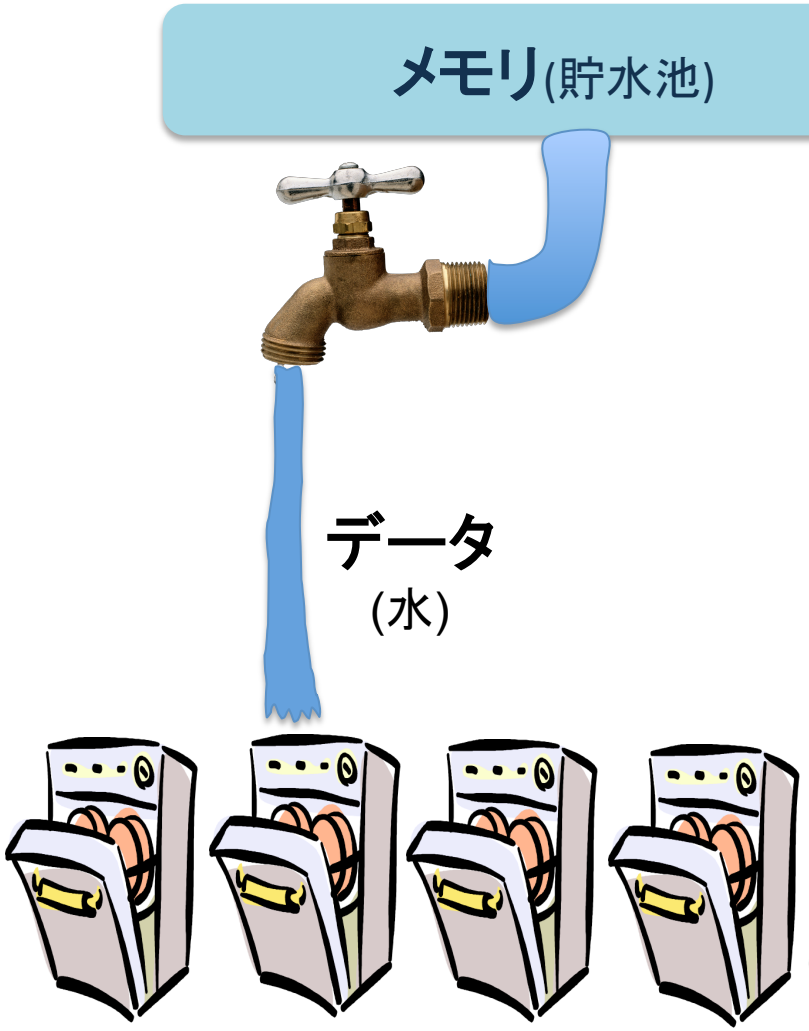
アプリケーションの超並列性を引き出す(3)



並列計算できない部分(非並列部)を限りなく小さくする！

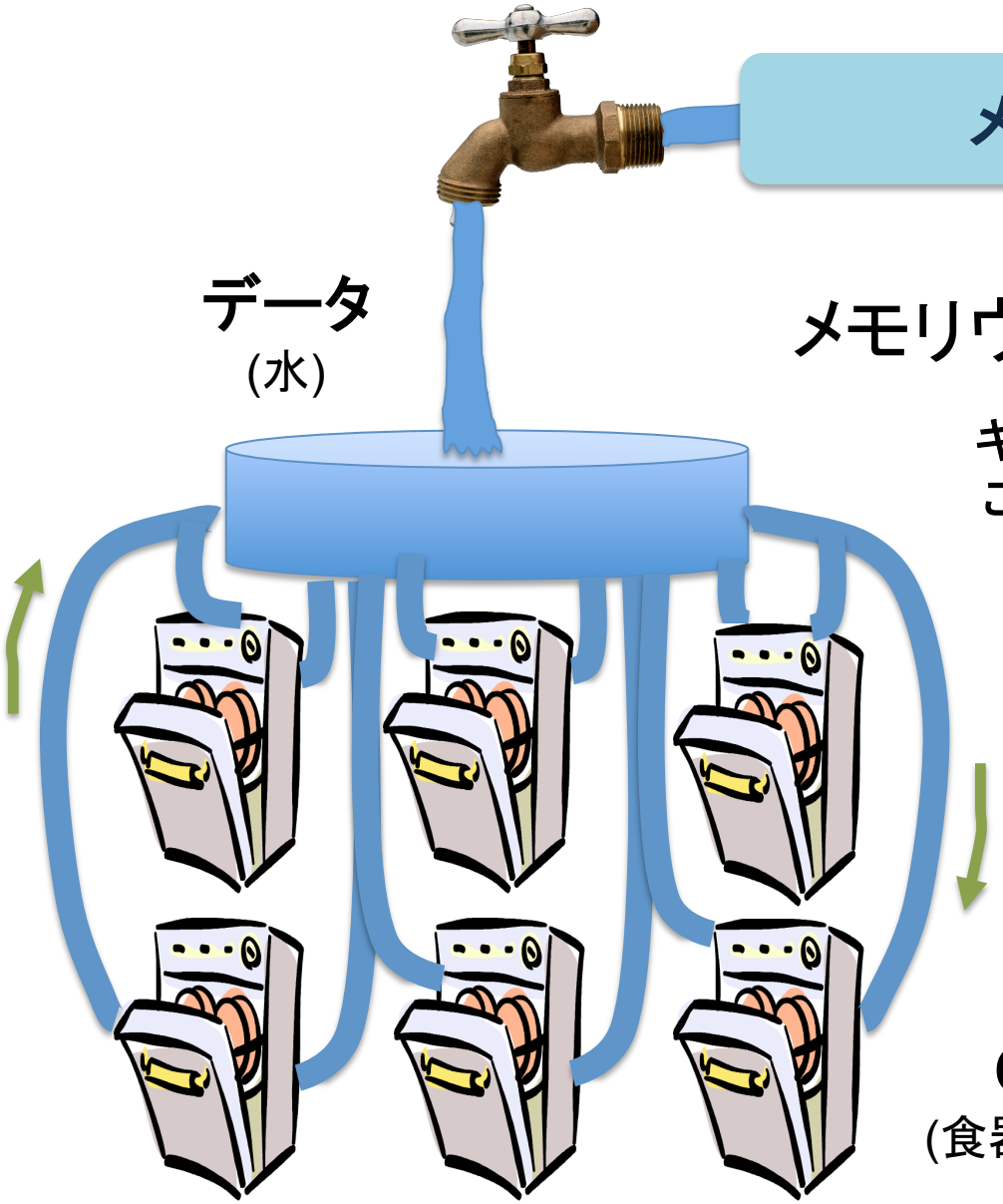
プロセッサの単体性能を引き出す(1)

代表的な問題：メモリウォール問題



- 昔の計算機はメモリのデータ供給能力と演算器の能力がバランスしていた
- 現代の計算機は演算器の能力が高くなりメモリのデータ供給能力が相対的に不足している

プロセッサの単体性能を引き出す(2)

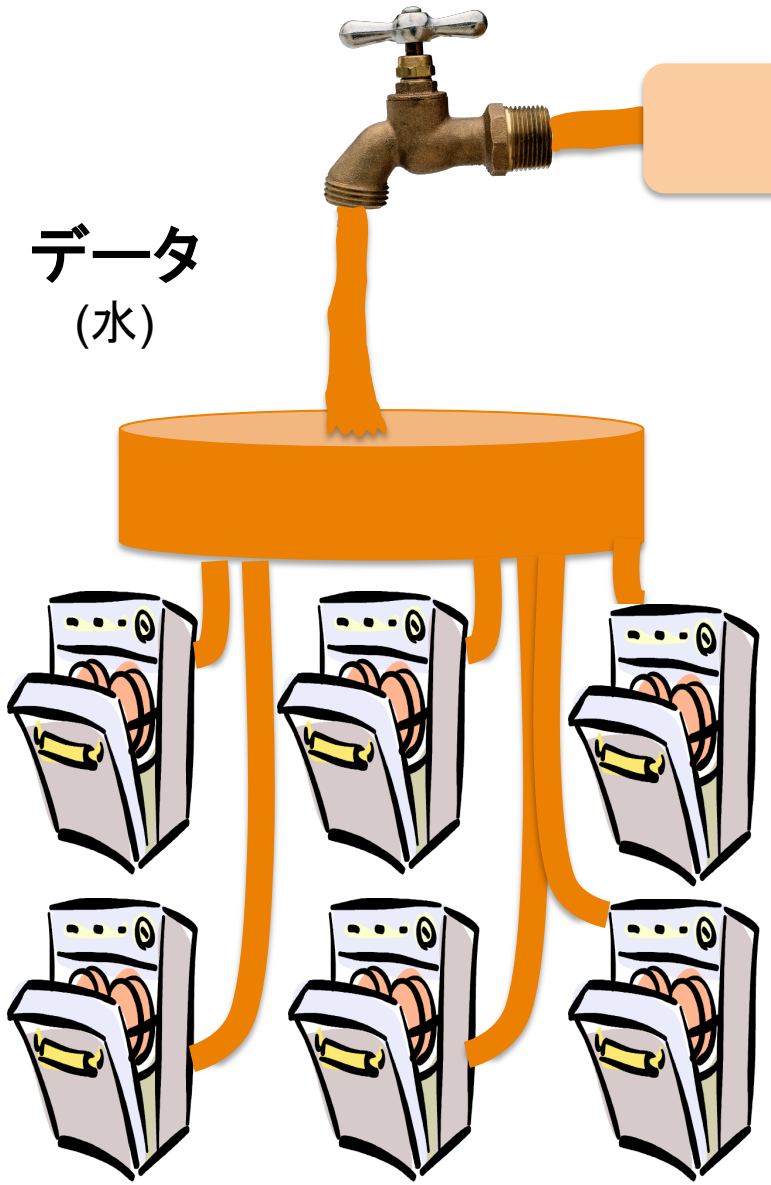


メモリウォール問題への対処

キャッシュ(一時貯水容器)を設けることによりリサイクル可能な水を使えばたくさんの食器を洗える

「アプリケーションが要求するB/F値が低い」タイプの計算と云う

プロセッサの単体性能を引き出す(3)



メモリ(貯水池)

データ
(水)

CPU
(食器洗い機)

それでも問題が

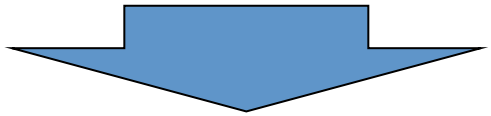
キャッシュ(一時貯水容器)を設けても
リサイクル不可能な水を使えば
水が溜まるまで食器が洗えない

「アプリケーションが要求するB/F値が高い」タイプの計算と云う

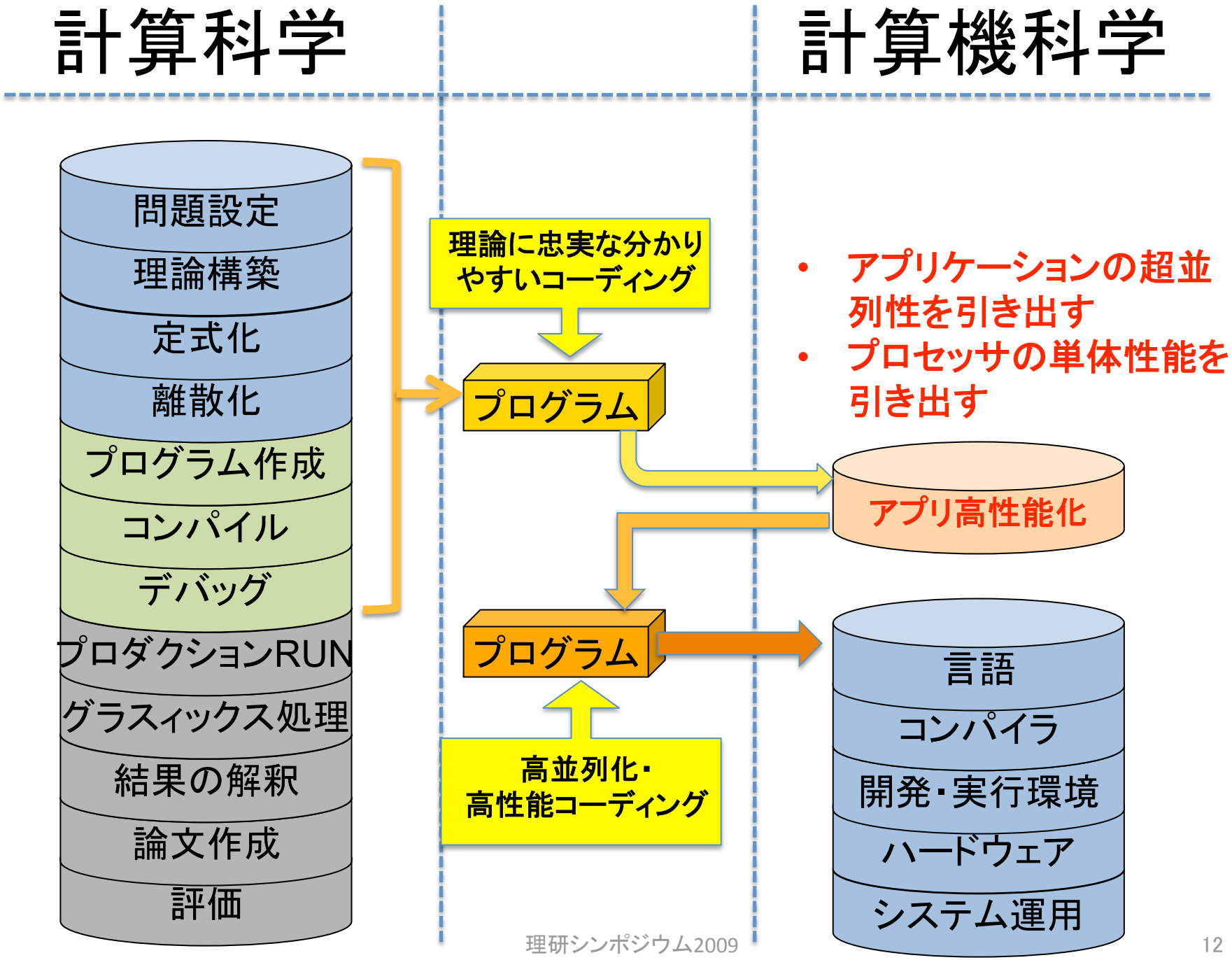
理研で進めている
アプリ高性能化について

目的

- 次世代スパコンの運用に先立ち、システム性能を実証する



- 並列性能が見込めるコード複数本を対象
- 次世代スパコンの汎用性を踏まえ分野バランスを考慮
- 次世代スパコンの汎用性を踏まえ計算特性のバランスを考慮
  - アプリケーションの要求B/F値が高い・低い
  - 並列化手法が比較的理解しやすい・かなり複雑


計算科学
計算機科学
問題設定
理論構築
定式化
離散化
プログラム作成
コンパイル
デバッグ
プロダクションRUN
グラスィックス処理
結果の解釈
論文作成
評価
理論に忠実な分かりやすいコーディング
プログラム
プログラム
高並列化・高性能コーディング
アプリ高性能化
言語
コンパイラ
開発・実行環境
ハードウェア
システム運用
• アプリケーションの超並列性を引き出す
• プロセッサの単体性能を引き出す

対象コード

ES：地球シミュレータ1

プログラム名	分野	アプリケーション概要	コードの計算科学的な特性	手法
NICAM	地球科学	全球雲解像大気大循環モデル	ESではピーク性能比40%程だがプログラミング上は**B/F性能を要求する**ように見える。次世代スパコンではキャッシュの有効利用を始め高速演算機構の活用が必須。	FDM (大気)
Seism3D	地球科学	地震波伝播・強震動シミュレーション	ESではピーク比40%程だがプログラミング上は**B/F性能を要求する**ように見える。次世代スパコンではキャッシュの有効利用を始め高速演算機構の活用が必須。	FDM (波動)
PHASE	ナノ	平面波展開第一原理分子動力学解析	**単体性能向上は主要処理の行列・行列積化により可能**との予測が立っている。しかし原子数を相当増やさないと**超高並列は難**のため高並列化の検討要。	平面波 DFT
FrontFlow/Blue	工学	Large Eddy Simulation (LES)に基づく非定常流体解析	ESではピーク比25%程だがプログラミング上は**B/F性能を要求するように見え、またリストアクセス**のため次世代スパコンではキャッシュの有効利用を始め高速演算機構の活用、データアクセスの効率化が必須。	FEM (流体)
RSDFT	ナノ	実空間第一原理分子動力学計算	**単体性能は主要処理の行列・行列積化により可能**との予測が立っている。しかしメッシュ分割数を相当増やさないと**超高並列は難**のため高並列化の検討要。	実空間 DFT
LatticeQCD	物理	格子QCDシミュレーションによる素粒子・原子核研究	通信トポロジーを意識した**高度な並列化チューニング**、また実機を意識した**単体性能向上策が必須**。	QCD

役割分担

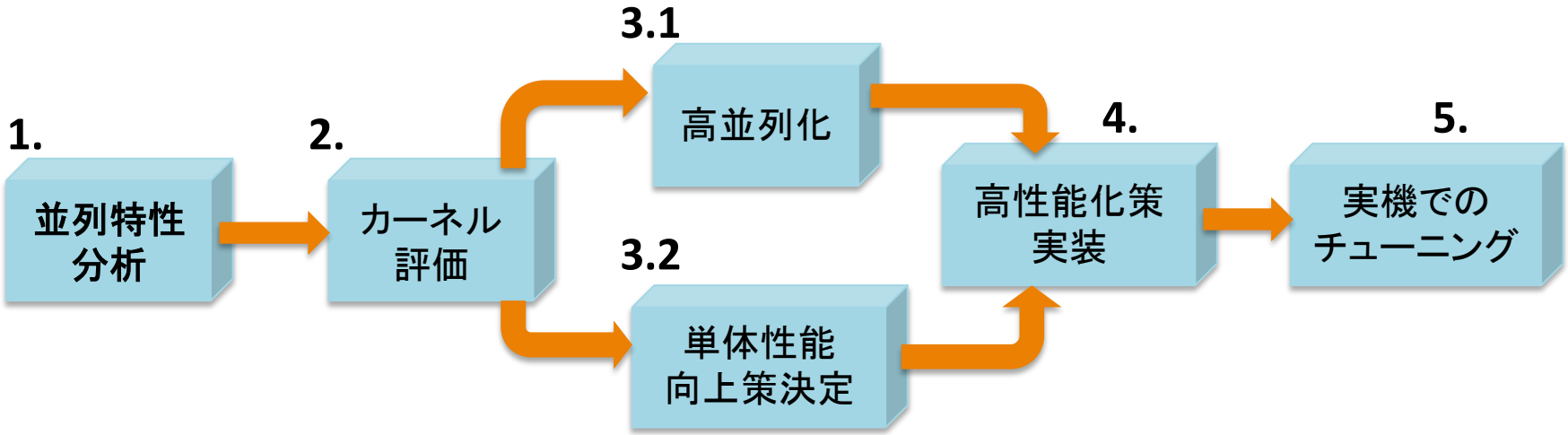
計算科学
(コード開発者)

計算機科学
(理研)

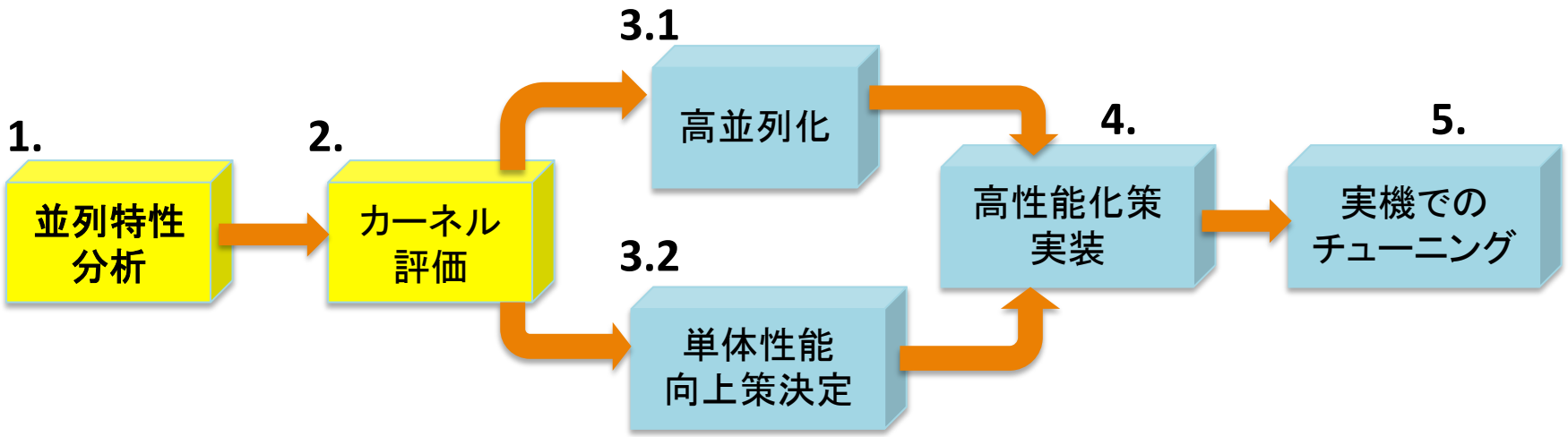
- 高並列・高性能化の検討
- 試作を元にコード全体として開発

- 高並列・高性能化の検討
- ハードウェア情報に基づく高並列化・高性能化の試作・性能評価

アプリ高性能化のステップ

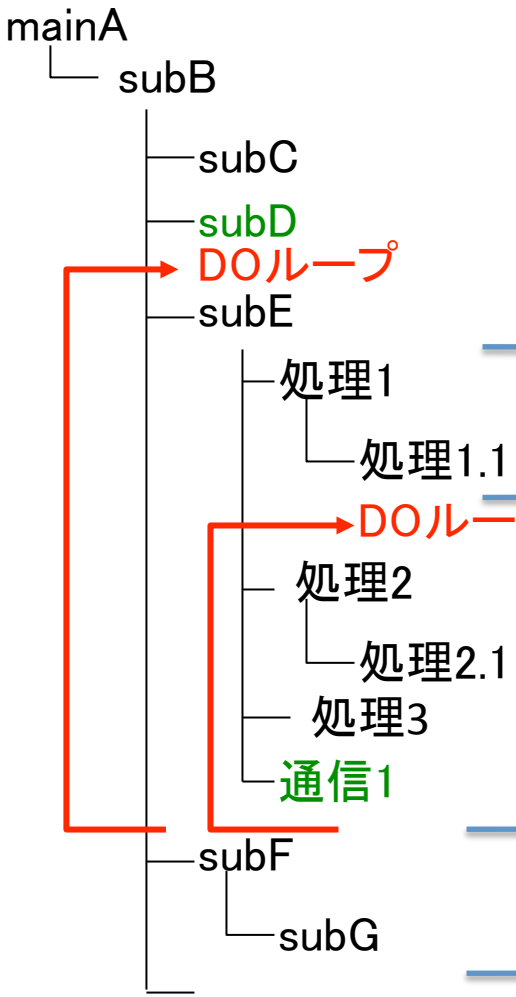


並列特性分析・
カーネル評価



1. 並列特性分析
(処理構造分析・ブロック特性分析)

(1)コードの構造を分析し物理に沿った処理ブロック(計算/通信)に分割
(2)コードの実行時間の実測
(3)プログラムソースコードの調査
(4)処理ブロックの物理的処理内容を把握
(5)計算ブロック毎の計算特性把握(非並列/完全並列/部分並列、Nに比例/N**2に比例等)
(6)通信ブロック毎の通信特性把握 (グローバル通信・隣接通信、隣接面に比例&隣接通信/体積に比例、等)



	実行時間・スケーラビリティ	物理的処理内容	演算・通信特性	演算・通信見積り	カーネル
ブロック1（計算）			部分並列	Nに比例	
ブロック2（計算）			完全並列	N**3に比例	○
（通信）			隣接通信	隣接面に比例	○
ブロック3					

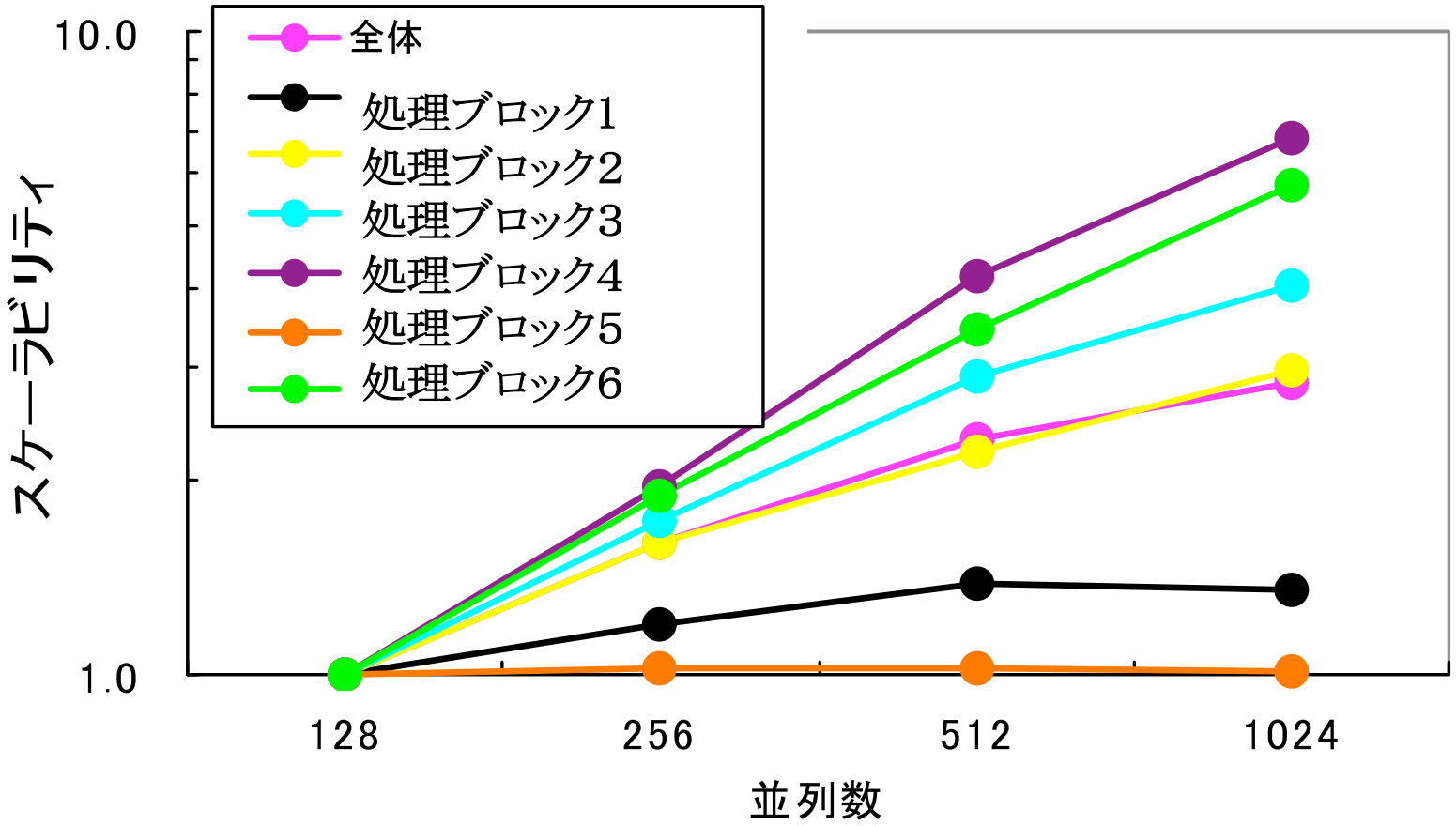
2. カーネル評価

(1)計算・通信ブロックについて物理的処理内容・コーディングの評価を行い同種の計算・通信ブロックを評価し異なる種類の計算・通信ブロックをカーネルの候補として洗い出す
(2)並列特性分析の結果から得た問題規模に対する依存性の情報を元にターゲット問題実行時に、また高並列実行時にカーネルとなる計算・通信カーネルを洗い出す

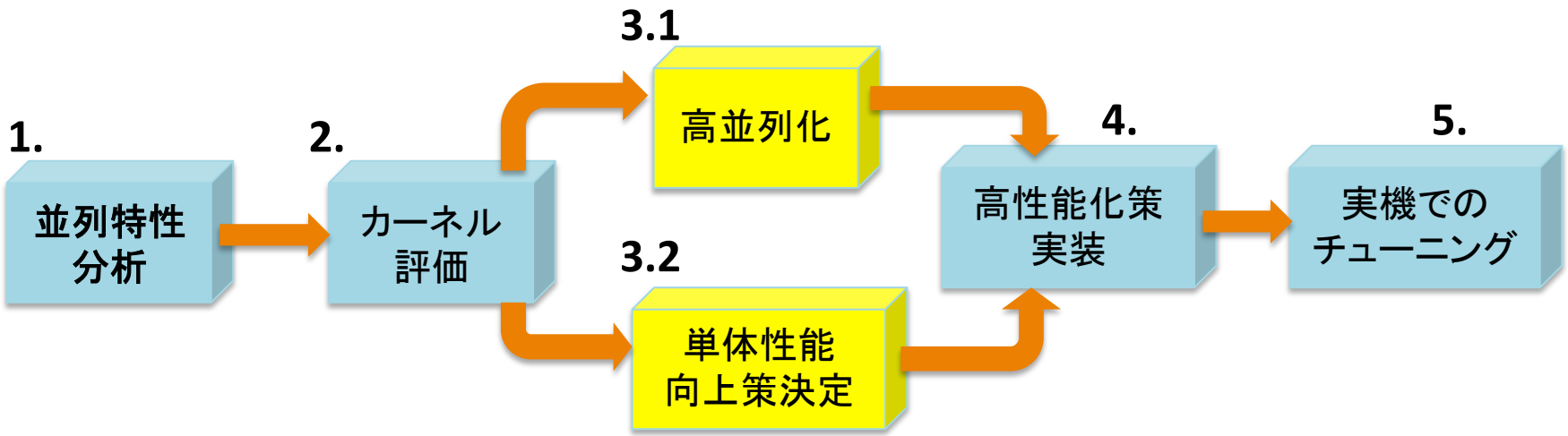
ブロック毎の実行時間とスケーラビリティ評価(例)

→従来の評価はサブルーチン毎・関数毎等の評価が多い

→サブルーチン・関数は色々な場所で呼ばれるため正しい評価ができない



高並列化・単体性能向上策決定



3.1高並列化

超高並列を目指した場合の留意点

(1)非並列部が残っていないか？残っている場合に問題ないか？
(2)ロードインバランスが超高並列時に悪化しないか？
(3)隣接通信時間が超高並列時にどれくらいの割合を占めるか？
(4)大域通信時間が超高並列時にどれくらい増大するか？

そのための準備 → これらの評価が重要

(1)ターゲット問題を決める
(2)1プロセッサの問題規模がターゲット問題と同程度になるようなテスト問題を作成する(100並列程度)
(3)このテスト問題で実行時間・ロードインバランス・隣接通信時間・大域通信時間を測定・評価する
(4)問題なければウィークスケーリングで大規模並列の挙動を測定する(次頁)
(5)問題が有ればストロングスケーリングで並列特性を測定し問題点を見つける

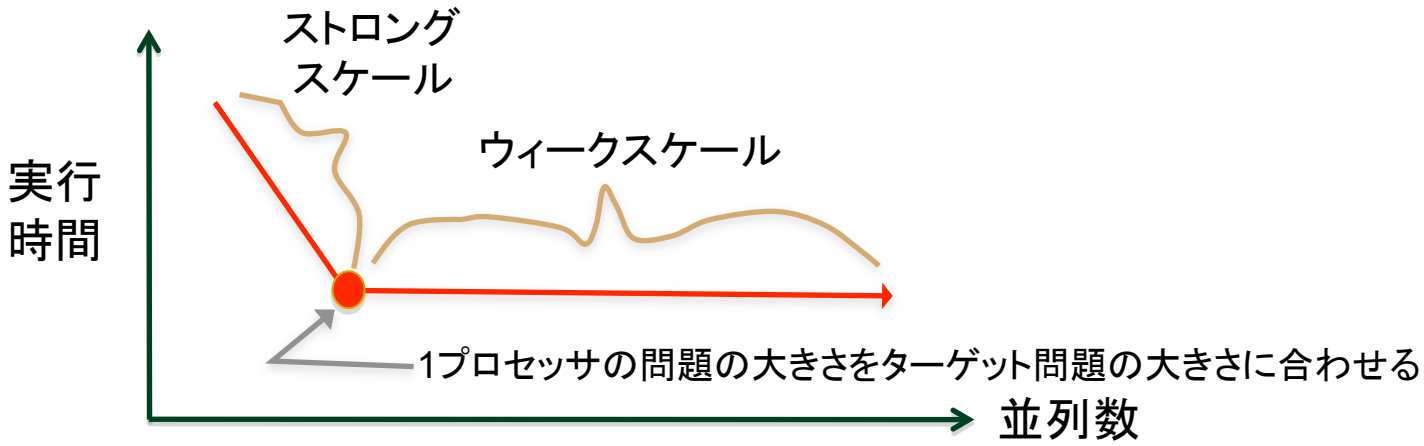
ストロングスケーリング：全体の問題規模を一定にして並列数を増やし測定する方法
ウィークスケーリング　：1プロセッサで実行する問題規模を一定にし並列数を増やし測定する方法

高並列に向けた性能測定

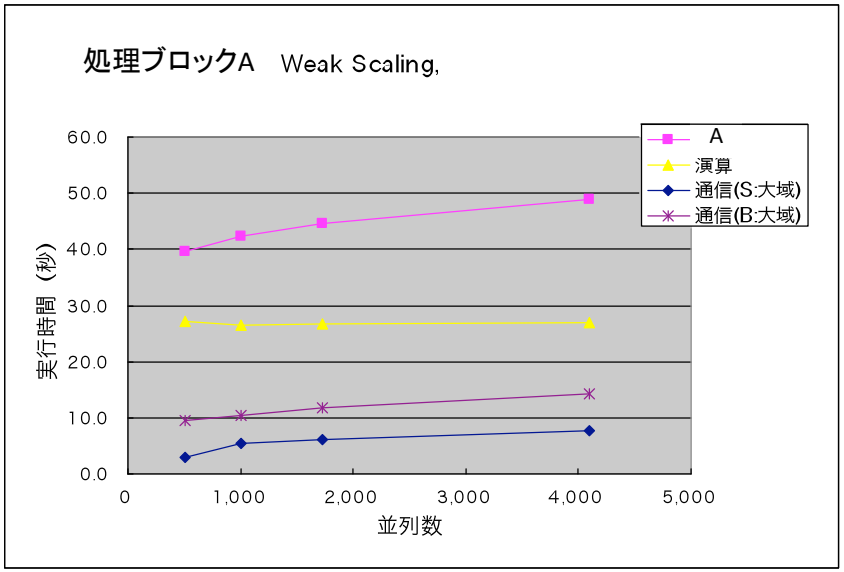
(1)現状使用可能な実行環境を使用し100程度/1000程度/数千程度と段階を追ってできるだけ高い並列度で並列性能を確認する(ウィークスケーリング測定).
(2)ウィークスケーリングが難しいものもあるが出来るだけ測定したい. 難しい場合は, 工夫が必要.
(3)高並列時の実行時間, ロードインバランス, 隣接通信時間の挙動, 大域通信時間の挙動を見る.

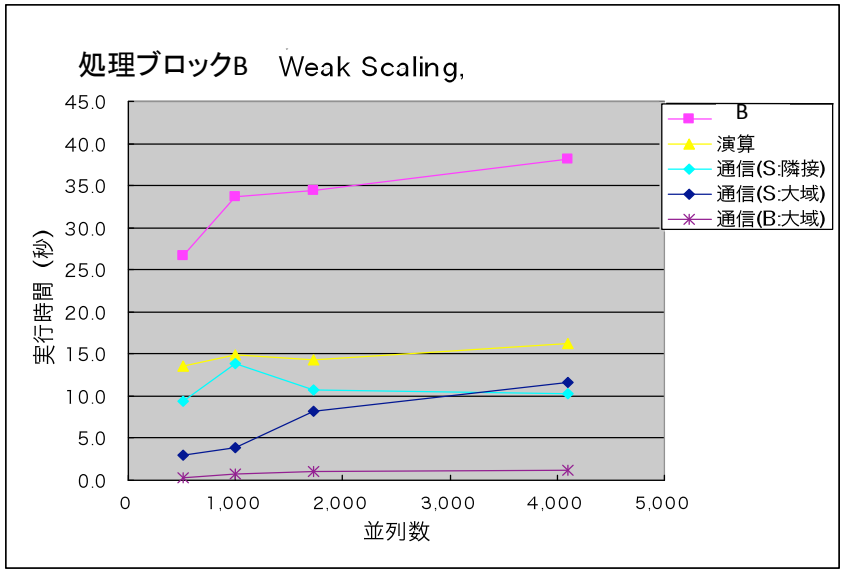
→実行時間は増大しないか？非並列部が残っていれば実行時間が増大する
→ロードインバランスが増大しないか？
→隣接通信は増大しないか？
→大域通信は一般的に増大するがその度合いは問題ないか？

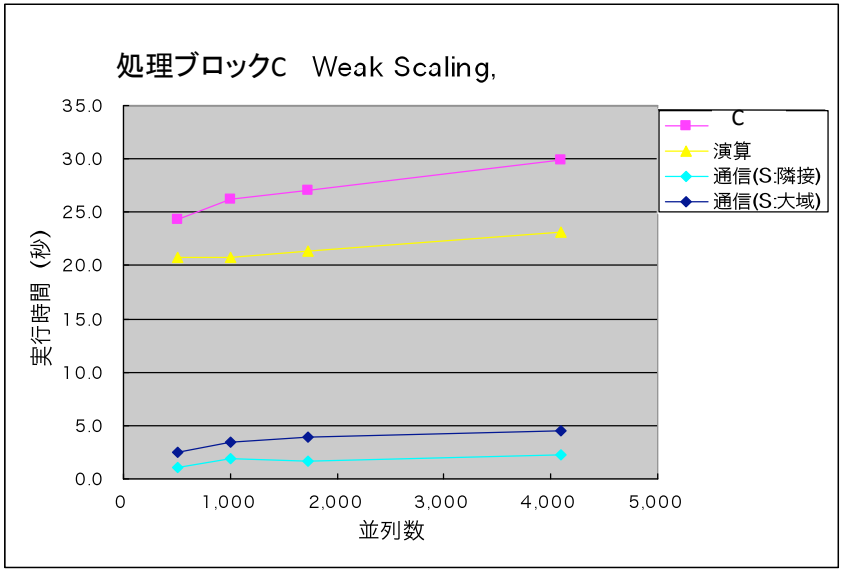
(4)(3)まで測定し通信回数・通信量等のモデル化が出来ていれば, 次世代機での通信時間を見積もる事ができる.

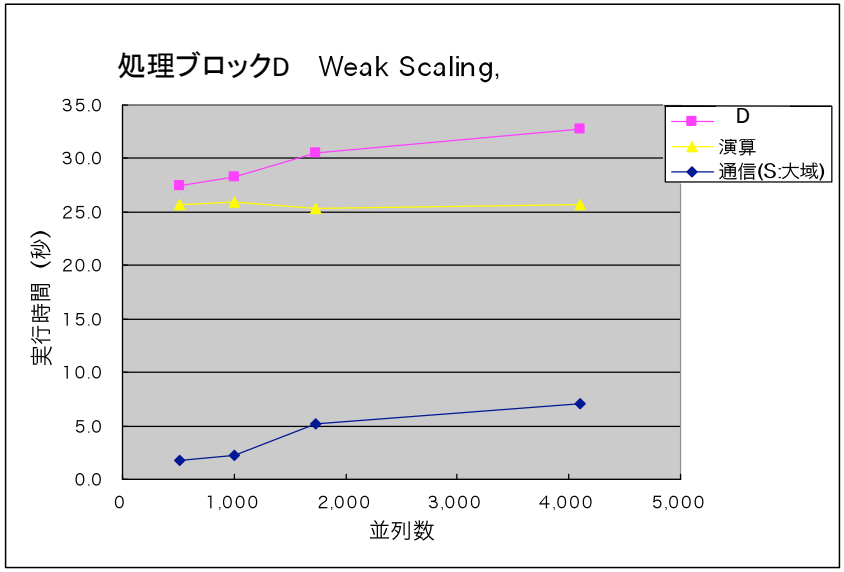


Weak Scaling測定例









3.2 単体性能向上策決定

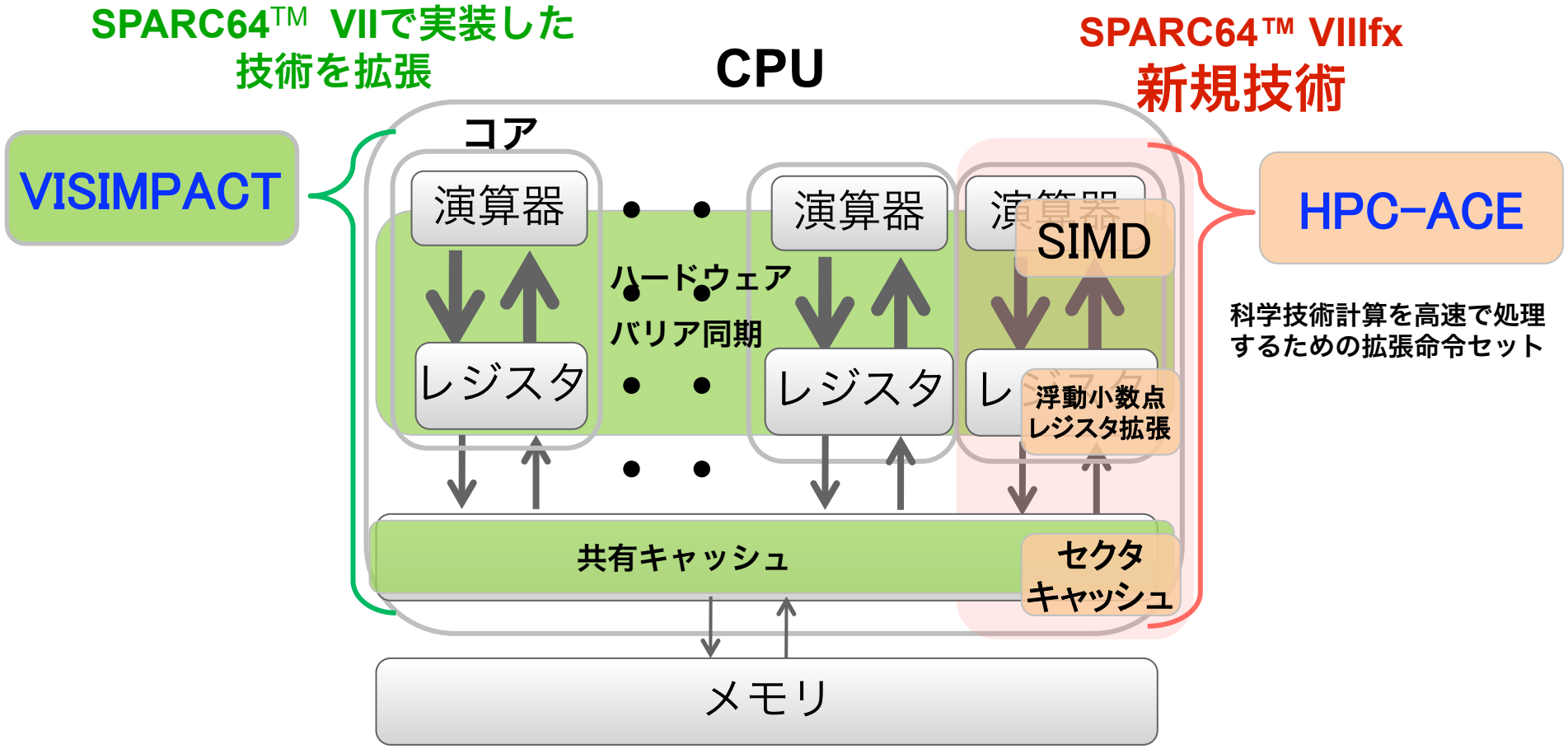
(1)計算カーネルの切り出し→計算カーネルを独立なテストプログラムとして切り出す.

(2)性能向上策の試行→切り出したテスト環境を使用し様々な性能向上策を試行する.

(3)性能向上策の評価・決定→試行結果を評価し性能向上策の案を策定する.

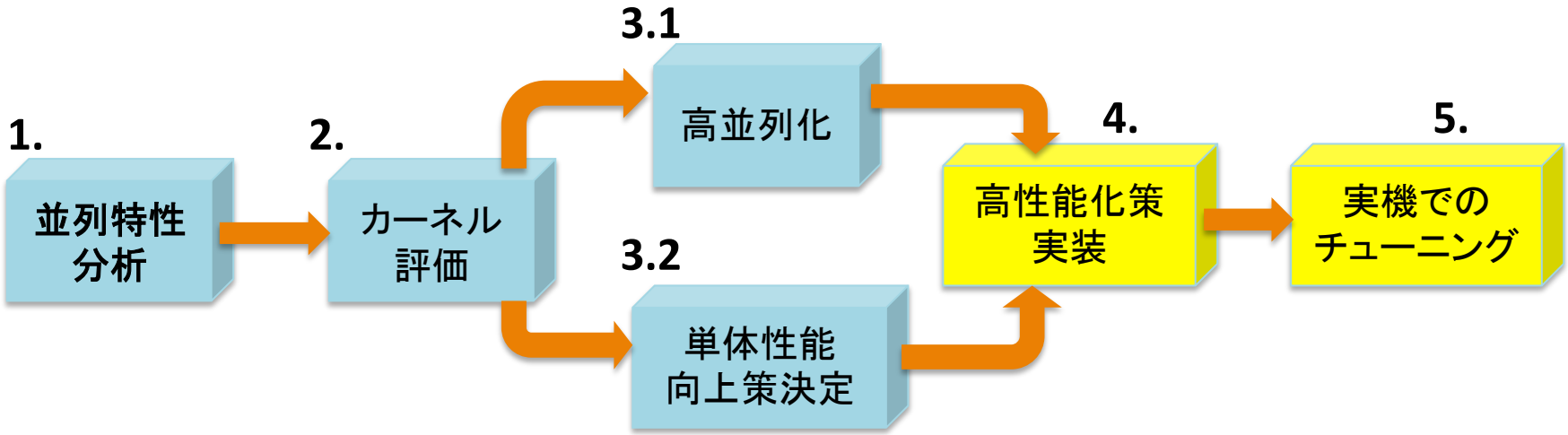
(4)作業量見積り→性能向上策を実施した場合のコード全体に影響する作業を洗い出す.それら作業を実施した場合の作業量を見積る.それらを評価し最終的に採用する案を決定する.

SPARC64™ VIIIfx　高速化技術



➔コンパイラ/アプリケーションで、いかに活用できるか/するかが高速化のキーポイント

高性能化策実装
実機でのチューニング



4. 高性能化策実装

高並列化の対策を実施したコードに対して策定した単体性能向上策を実装する

設計・プログラミング・デバッグ作業であり作業工数的には大きな作業となる

5. 実機でのチューニング

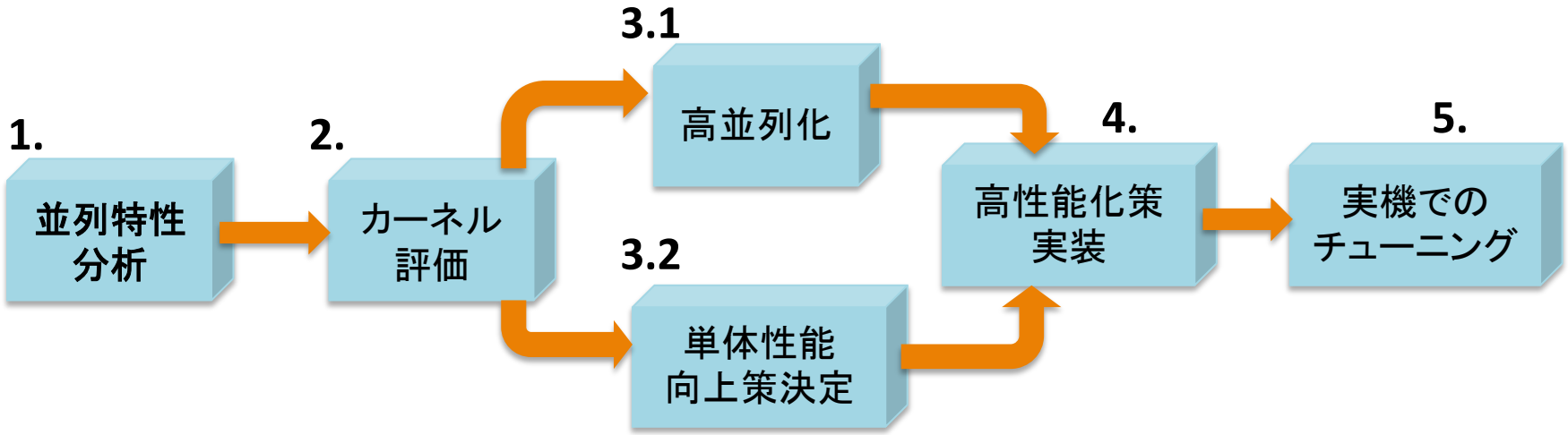
4まで実施されたコードをさらに次世代スーパーコンピュータをターゲットマシンとしてチューニングする

(1)測定
・実機上で並列性能/単体性能を測定する.

(2)問題点洗い出し
・並列性能/単体性能のそれぞれについて
問題点を洗い出す.

(3)問題点解決
・洗い出された問題点の解決策策定
・解決策の実装

まとめ



- 現在進めているアプリ高性能化作業について紹介
- 上図の手順に則り統一した方法で作業を進めている
- 数万を超える超並列への対応とプロセッサを意識した単体性能向上が重要
- 現状で可能な高並列機での特性取得と現行スカラ機上で可能な限りの単体性能向上を実施
- その上で今後実機でのチューニングを実施