ロードマップ
（数値計算ライブラリWG）

WG代表
須田礼仁（東大）、片桐孝洋（東大）

2011年12月8日

数値計算ライブラリWG役割分担

- ロードマップ作成担当

（＊は責任者）

- 全体統括：　＊須田
- 自動チューニング：＊片桐、須田
- 数値計算ミドルウエア：＊小野、伊東
- 数値計算ライブラリ：＊高橋、岩下

- 横断的キーワード担当

- ヘテロ　　：大島
- メモリ　　：小野
- 大規模並列　：高橋
- 電力　　：須田
- 耐故障　　：須田
- 生産性（ツール）：
  片桐・岩下（AT、数値ライブラリ）、伊東・小野（数値ミドルウェア）

話の流れ（全体：25分＋10分質疑）

進行役：片桐

- イントロ（4分）：片桐
- 分野ごとのロードマップ説明
  - 数値ライブラリ（7分）：高橋
  - 数値ミドルウェア（7分）：
    小野（代理、伊東）
  - 自動チューニング（7分）：片桐

アプリケーション特性による抽象化の重要性

- 数値計算ライブラリ構築の目的
  - できるだけ多くのアプリケーション主要演算を抽象化し、その機能と高性能実装を提供する
  - プログラミング生産性を格段に高め、高性能化を実現する
- アプリケーション主演算を、たとえば以下の尺度で分類
  - 計算機ノード内におけるメモリアクセスパターン
  - 計算機ノード間の通信パターン
  - 通信回数（通信レイテンシ）と通信量（通信バンド幅）
  - I/O性能（データアクセス頻度とデータ量）
  - その他アプリケーションから抽出できる尺度
- 抽象化により、計算機ハードウェア構成方式、数値計算ライブラリや数値計算ミドルウェアのインターフェース、数値計算のための自動チューニング機能、などを構築するための指針を示すことは重要

章立て

- 各章について

1. はじめに
2. 数値計算ライブラリ分野
3. 数値計算ミドルウェア分野
4. 数値計算ライブラリのための自動チューニング分野
5. 我が国でファンディングされた課題

    5．１　はじめに
    5．２　数値計算ライブラリ分野
    5．３　数値計算ミドルウェア分野
    5．４　数値計算ライブラリのための自動チューニング分野

- 節について

x．１　はじめに
x．２　技術動向
x．３　技術項目と優先度
x．４　ロードマップ

技術項目と優先度

- 主項目と副項目に分けて技術課題を列挙
- 副項目（短期）：5年程度に達成すべき副項目
- 副項目（長期）：5年以上必要とする技術項目、もしくは、基礎研究すら開始されていない技術項目
- 必要性・重要性：エクサを達成するために必要か（必要性）、技術として重要か（重要性）、についてランキングをつけた
  - ☆、★、★★、★★★（→ 重要度が高い）
- 2011年度の研究進捗の度合いについてフェーズを記載
  - フェーズI ：これから基礎研究
  - フェーズII：基礎研究が進行中
  - フェーズIII：プロトタイプが進行中
  - フェーズIV：もうすぐ実用化メド
- 既存技術：エクサで適用可能な現存技術
- 革新技術：エクサを達成するために新規開発がいる技術（項目）

ロードマップ
（数値計算ライブラリWG：
数値計算ライブラリ分野）

文責：高橋大介（筑波大）

2011年12月8日

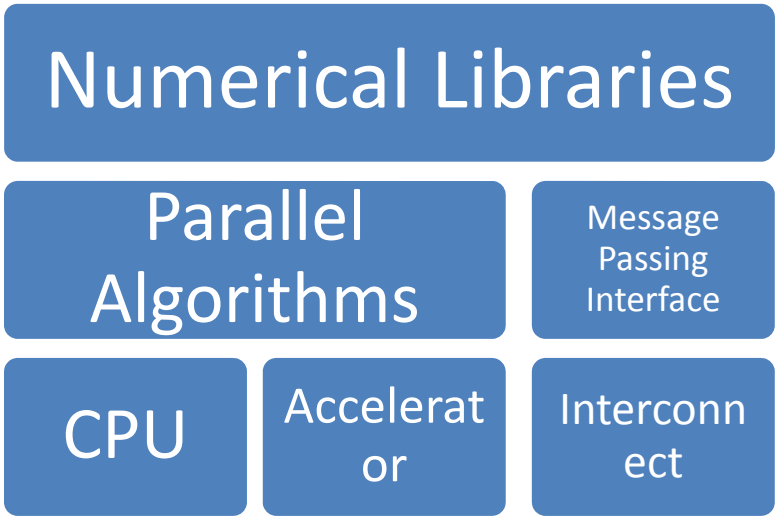
はじめに

- 数値計算ライブラリは、科学技術アプリケーションプログラムにおいて共通に使われる数値計算プログラムをライブラリの形にまとめたものである。
- 現在、大規模数値シミュレーションを行う際には、数値計算ライブラリの性能が実行時間に大きく影響する。
- したがって、エクサフロップス級マシンにおいて高い実行効率を達成できる数値計算ライブラリを実現することは重要である。
- エクサフロップス級マシンにおける数値計算ライブラリを実現する上で必要な技術項目を示す。
- エクサフロップス環境で実現する数値計算ライブラリが実現する機能
  - アプリケーション開発者および利用者からエクサフロップス級マシンの複雑なシステム構成が見えないように抽象化されたインターフェースを提供。
  - エクサフロップス級マシンの高い性能をできるだけ引き出すようなアルゴリズムおよび実装手法を用いること。

数値計算ライブラリ

- 適切に設計・実装された数値計算ライブラリを使うことで、アプリケーションプログラムの
  - 実行性能の向上
  - 開発期間の短縮
  - 開発コスト・メンテナンスコストの削減

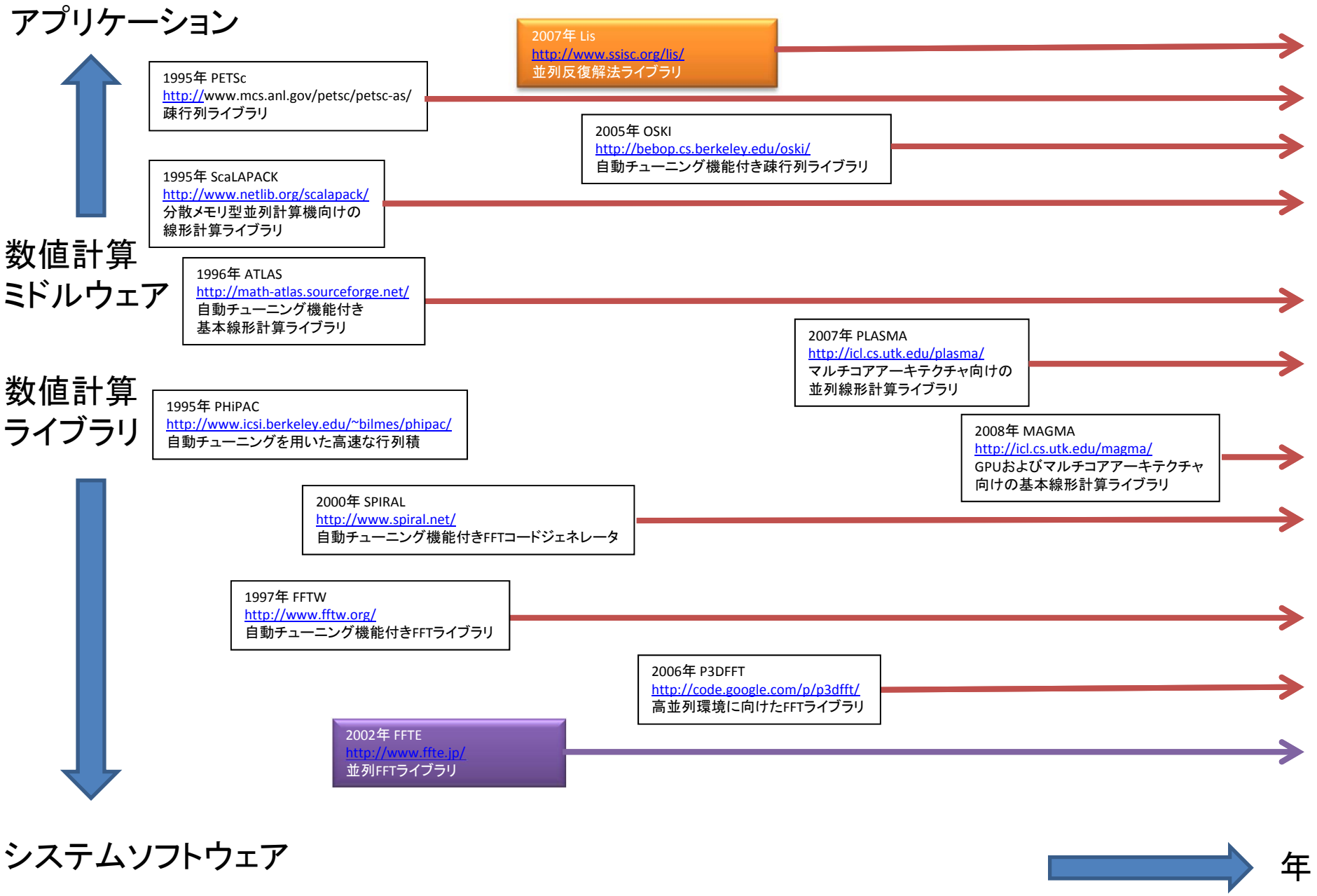
  を図ることができる.
- エクサフロップス級マシンにおいては、高い実行効率を達成するために必要なプログラミングコストが、さらに増大すると考えられる。
- 数値計算ライブラリの必要性が高くなる。



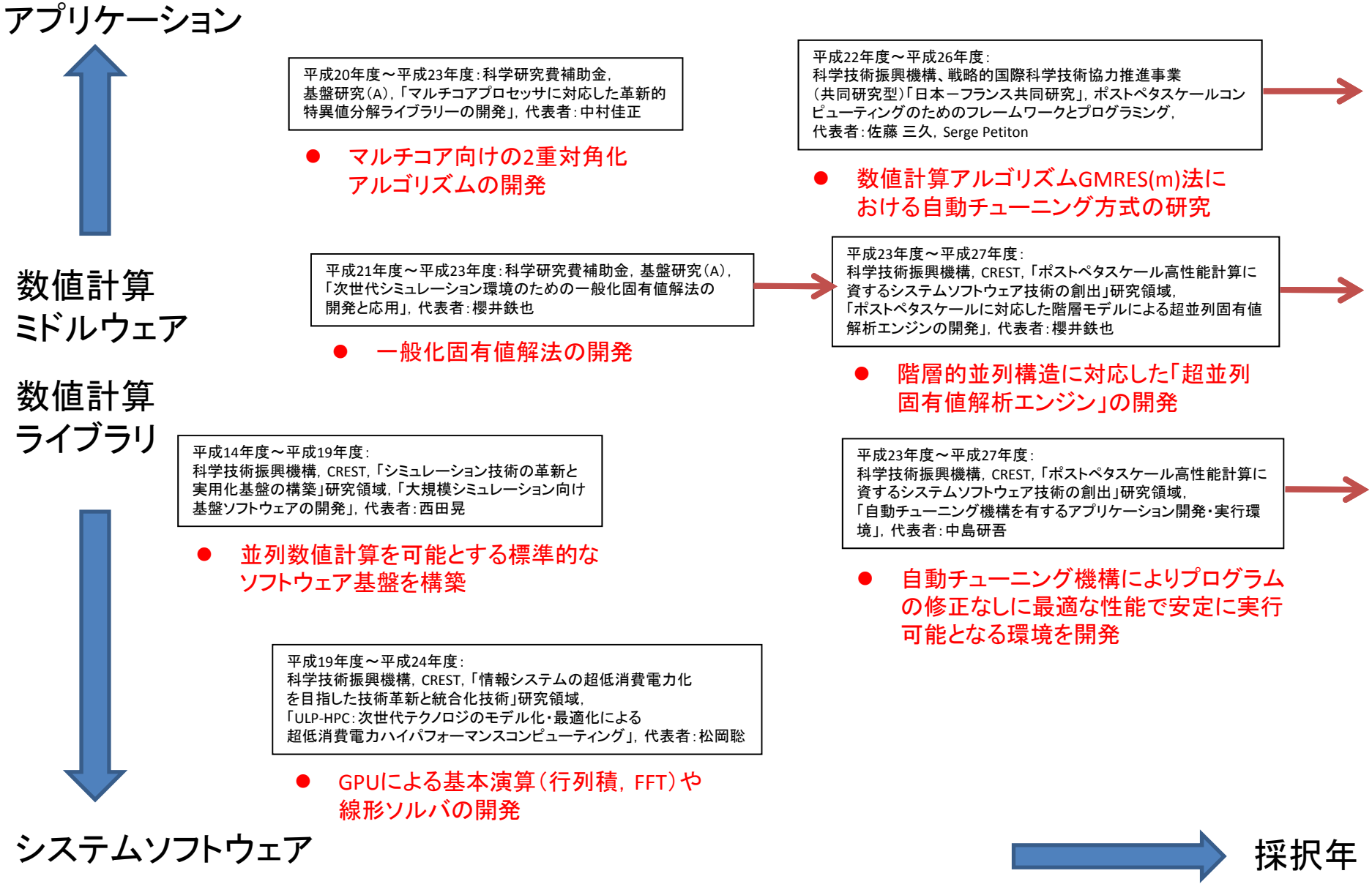
課題

- 通信量およびメモリアクセス回数の削減
  - 通信（回数またはデータ転送量，またはその両方）を削減するアルゴリズム
- 演算精度
  - 倍精度で十分か？
  - ハードウェアのByte/Flop値が小さい場合，ソフトウェアによる4倍精度演算（例えばdouble-double型）でも倍精度演算に比べてそれほど計算が遅くならない可能性がある．
  - 混合精度演算
- 耐故障性
  - 数値計算ライブラリにチェックポイント／リスタートの機能を持たせる？
- 数値計算ライブラリのインターフェース
  - 既存のアプリケーションプログラムからの移行コストを低くする必要がある．
  - 非均質プロセッサの場合でも，できればこれまでと同じAPIが望ましい．
  - ノード数の増加に伴い，最適なデータ分散が変わると，APIも変更せざるを得なくなる．

数値計算ライブラリ技術マップ



我が国でファンディングされた数値計算ライブラリ研究俯瞰図



数値計算ライブラリの技術項目 1/3

■ 通信量およびメモリアクセス回数の削減

1. 通信最適化

- 【革新技術】
  - AllgatherやAllreduce等の集合通信を避けたライブラリ（アルゴリズムの変更が必要）
  - 通信（回数またはデータ転送量，またはその両方）を最小限にしたアルゴリズム
  - 通信量を増やしてでもレイテンシの影響を削減し，通信時間を削減する通信方式

2. 演算量を増やしてでも通信量やメモリアクセス回数を削減するアルゴリズム

- 【革新技術】
  - データは他のノードから持ってくるのではなく，自ノードで計算できるものは計算する

■ 演算精度

1. 高精度計算

- 【革新技術】
  - 高精度計算のCPU／GPU最適化

2. 混合精度演算・精度保証計算

- 【革新技術】
  - 単精度演算や倍精度演算と3倍精度・4倍精度演算を組み合わせる
- 【既存技術】
  - 区間演算を用いることにより，精度を保証する

3. 精度をユーザが確認する方法

- 【革新技術】
  - verboseモード？逐次実行・演算順序保証実行モード？

数値計算ライブラリの技術項目 2/3

■ 耐故障性

1. フォールトトレラント機能

- 【革新技術】
  - システムソフトウェアのレベルではなく，数値計算ライブラリにチェックポイント／リスタートの機能を持たせる
  - 耐故障性の必要性の調査

■ 数値計算ライブラリのインターフェース

1. ライブラリ利用形態に関する検討

- 【革新技術】
  - 数値計算ライブラリがどのような階層で使用されるか，されるべきかの再検討

2. 適切なHWを選択するための情報を得る仕組み

- 【革新技術】
  - 電力効率最大HWがどれかを知るためには，電力情報をうまく利用できる仕組み（API?）が必要

■ 非均質プロセッサ対応

1. 非均質プロセッサ環境に対応するライブラリ

- 【既存技術】
  - マルチコア，メニーコア，GPUへの対応
- 【革新技術】
  - ハイブリッド並列処理対応

2. 「各HWで最大性能を達成する＆ライブラリ化」を繰り返す方法論の確立

3. 生産性の高いアプリケーション記述用言語（フレームワーク）

- 【革新技術】
  - ライブラリ自体は何で記述するのか？HW専用言語ではその後が続かない

他WGとの関係

- アーキテクチャWG
  - ヘテロジニアスアーキテクチャ
  - メモリアーキテクチャ
  - 大規模並列・ストロングスケーリング・ネットワーク
  - 耐故障・信頼性
- システムソフトウェアWG
  - ランタイム（MPIライブラリ等）
  - 耐故障
- プログラミングWG
  - 大規模並列性
  - 複雑化するメモリアーキテクチャへの対応
  - 耐故障
  - 生産性（ツール）

数値計算ライブラリ分野：優先度（主項目）

技術完成目標年度	技術項目	必要性	重要性	2011年における研究進捗
2016年	非均質プロセッサ対応	★★★	★★★	フェーズII
2014年	高精度計算	★★	★★	フェーズI
2016年	通信最適化	★★★	★★★	フェーズII
2014年	通信量を増やしてでも通信量やメモリアクセス回数を削減するアルゴリズム	★★★	★★★	フェーズII
2018年	混合精度演算・精度保証計算	★★	★★	フェーズII
2020年	フォールトトレラント機能	★★★	★★★	フェーズI

数値計算ライブラリ分野：優先度（副項目）

技術完成目標年度	技術項目	必要性	重要性	2011年における研究進捗
2014年	ハイブリッド並列処理対応	★★★	★★★	フェースII
2014年	AllgatherやAllreduce等の集合通信を避けたライブラリ	★★★	★★★	フェーズII
2014年	混合演算・精度保証による数値計算安定化	★★	★★	フェーズII
2014年	耐故障性の必要性の調査	★★★	★★★	フェーズII
2014年	高精度計算のCPU／GPU最適化	★★	★★	フェーズI
2014年	生産性の高いアプリケーション記述用言語（フレームワーク）	★★	★★	フェーズII
2016年	適切なHWを選択するための情報を得る仕組みの開発	★★	★★	フェーズI

ロードマップ（数値計算ライブラリ分野）

年	研究開発機能	備考
2011～2012	● 非均質プロセッサへの対応 ● 高精度計算のシリアル／CPU／GPU最適化 ● 通信量を増やしてでもレイテンシの影響を削減し，通信時間を削減する通信方式	● 大学センター群等ペタコン運用開始 ● 「京」供用開始（10ペタ級稼働）
2013～2014	● 非均質プロセッサへの対応 ● 通信最適化数値ライブラリ ● 高精度計算のシリアル／CPU／GPU最適化 ● 演算量を増やしてでも通信量やメモリアクセス回数を減らす数値計算アルゴリズム	● 10ペタ級スパコンの運用開始（大学センター群等）
2015～2016	● 非均質プロセッサへの対応 ● 高精度計算の並列版の実装 ● 混合精度演算・精度保証計算	● プリエクサ級が稼働
2017～2020	● フォールトトレラント機能を持つ数値計算ライブラリ ● エクサ級アプリでの利用を実現 ● エクサ級アプリでのテスト	● エクサ級が稼働

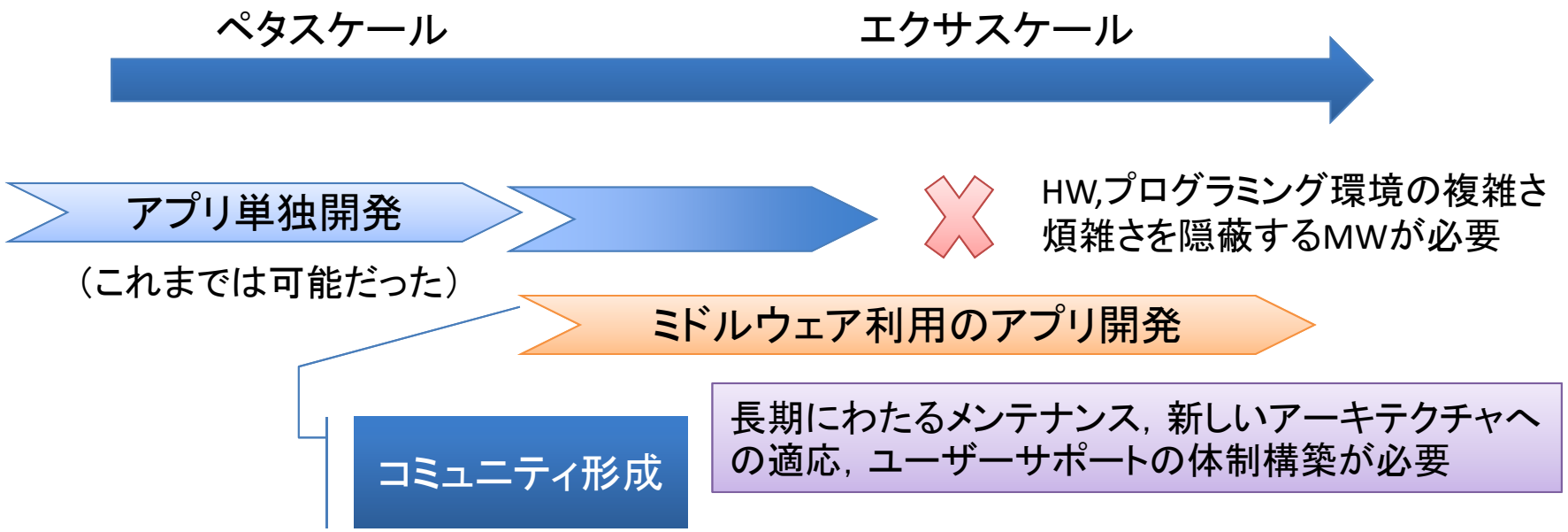
ロードマップ
数値計算ミドルウェア分野

小野謙二（東大/理研），
伊東聰（東大）

課題

- 電力をはじめとするハードウェアの制約条件
  - アクセラレータの利用
  - メモリモジュールの構成ポリシー（容量優先あるいは速度優先），深いメモリ階層
  - 非均質な帯域のネットワーク構成
  - ．．．
- エクサで性能を出せるアルゴリズム開発が必要
- アプリケーション開発のコストが増大
  - エクサスケールのプログラミングに関して，考慮すべき事項が組み合わせ的に増大

プログラム開発のパラダイムシフト



単独開発

- Pros
  - 最短で開発可
- Cons
  - 類似の開発は冗長
  - メンテとアップグレード難

ミドル利用

- Pros
  - 水平展開
  - 類似のアプリ開発効率化
- Cons
  - 設計（スコープ，共通項）
  - 開発期間

アクティブなMWとアプリの開発実績

- MW: SPHERE/V-Sphere
  - 開発者：小野（理研）
  - 利用者：グランドチャレンジ（ライフ）とVCADプロジェクトのアプリ開発者（理研，東大）
  - 稼働実績：Intel Cluster, Power, SX, Mac, Win, T2K, 京, SGI
  - 最大コア数：8,192（RICC），30,720（京）
  - 研究分野：
    - HIFU，流体構造連成，細胞内物質輸送，熱流体コード
    - エンジニアリング分野
  - 研究成果：
    - 大規模計算による高精度計算
    - 高並列アルゴリズム開発（次年度ゴードンベル候補？）
    - プロセスマッピングによる高性能通信フレームワーク
- MW：HPC-MW/HEC-MW
  - 開発者：奥田（東大）
  - 利用者：革新プロジェクト，メーカー
  - 稼働実績：Intel Cluste, AMD Cluster, SR, ES, SGI, T2K, Win
  - 最大コア数：4,092(ES)
  - 研究分野：
    - 構造，固有値，熱伝導，非圧縮/圧縮流体コード
    - FrontISTR
  - 研究成果：
    - 大型ポンプまるごと解析
    - リファイナー併用によるメッシュ細分化とマルチグリッド法による高速計算
    - 並列可視化

MW利用のアプリ開発と利用計画

- AICS
  - 統一現象解析
  - 可視化
- 戦略分野3
  - 気候モデリング
- 戦略分野4
  - 熱流体コード
  - 流体・構造・音響コード
- グランドチャレンジ
  - HIFU，流体構造連成，細胞内輸送．．

ライブラリ，フレームワーク，ミドルウェア

- ライブラリ
  - ある分野の有用かつ汎用的なサブルーチン群
  - アプリケーション構築法とは独立
  - 例:行列計算ライブラリ，FFTライブラリなど
- フレームワーク
  - ある特定分野のアプリ開発に再利用できる機能モジュール，ライブラリ群
  - アプリケーションアーキテクチャを形成する
  - 例:流体計算フレームワーク，MDフレームワークなど
- ミドルウェア
  - アプリケーションとOSのソフトウェアスタック間にあるソフトウェアパッケージの総称（ライブラリとフレームワークを含む）
  - アプリケーション・ミドルウェアは最もアプリに近い階層のミドルウェア
- 目的と役割
  - コード開発の生産性を高め，アプリケーションのプロトタイプ開発の効率化
  - 開発後の保守にも役立つコード開発環境
  - 複雑な現象・問題を迅速にコード化し，解析することを支援
  - 計算科学ソフトウェア構築に必要な共通的機能を提供する
  - 計算機アーキに対してチューニングされたサブルーチンを提供（自動チューニング機構の実装提供）

エクサ環境で実現するミドルウェア

1. エクサ向きアプリ開発を支援するフレームワーク
   - アプリ開発者，計算機科学者とのコデザイン
   - アプリに必要な共通的な機能モジュールのパッケージ化
   - 共同開発，コミュニティの醸成

2. 多様で複雑な問題を少ない労力でプロトタイピングするためのアプリケーション・ミドルウェア
   - 多くの研究者や技術者がエクサを使い，幅広い成果を創出することに貢献
     - 最高性能には届かないが，比較的高性能なレンジを狙う
   - ラピッドプロトタイピングの枠組みを構築
     - 連続体力学系が一つの出口
     - データ構造と解法を絞る必要あり
     - 優れたユーザインターフェイス

両者は，2つの独立したミドルウェアの開発ではなく，連続した継承関係にある

ミドルウェア開発

- ターゲット
  - 「何でもできる」ではなく，カスタマイズされた使いやすいものを指向
    - 分野：連続体力学系，離散力学系，．．などデータ構造や解法の分類により いくつかの特定領域のミドルウェアが考えられる
    - 物理モデリングを数式イメージで記述，迅速なプロトタイピング
      - 短時間で大規模並列計算を実施，物理モデリングの試行期間を短縮
    - 実現象を扱うので，形状や物理の複雑性に対応できる枠組みが必要
      - HPC分野に参入し成果を創出するコミュニティを拡大する
- 方針
  - フレームワークを構成する要素技術が成熟した時点で，順次採用，リファクタリング．
    - 要素技術：プログラミングモデル，言語，ライブラリ，アルゴリズム，FTなど
    - 技術的な課題以外にも，コミュニティ形成やアプリ開発サポートの課題もある

ミドルウェア開発スケジュール

1. フィージビリティスタディと開発計画の詳細化
   - アプリ開発，計算機科学，ミドルウェア開発の研究者による議論
   - 各シミュレーション分野から，エクサに向けた領域選定
     - データ構造，アルゴリズムを絞る必要あり
     - 共通化する技術項目のポートフォリオと対応プライオリティ
     - アプリ開発者との強固な連携ができること
2. プロトタイプ実装
   - クラスライブラリ群（必要な機能ライブラリとAPI）
   - アプリケーション・ミドルウェア（Mathematica+高性能・高並列バックエンド）
   - ペタスケールでの性能実証
   - 関連アプリのポーティング
3. アップデートと展開
   - 開発したミドルウェアの要素技術の開発進展に伴い，実装アップデート，機能拡張を継続的に実施
   - エクサスケールでの性能実証

他WGとの連携

- アプリWG
  - ミドルウェアを利用してアプリを書く開発者
  - ミドルウェアで用意すべき機能のリストアップ
  - エクサ向けアルゴリズム開発
- プログラミングWG
  - Domain Specificなフレームワーク構築を記述するメタな汎用的DSL
    - プログラミングWGで開発するDSLが利用可能になったら，DSLで記述．それまでは，実アプリの記述方法や移植方法の検討を先行実施
  - 並列プログラミングモデル/言語
- 数値計算ライブラリWG（当WG）
  - ミドルウェアアーキテクチャ設計
    - ある分野の具体的なアプリ構築の観点からニーズベースでアプリケーション・ミドルウェアを設計・構築
  - ライブラリ設計
    - ライブラリとして実装する機能の選定，API設計，実装

技術項目 1/2

■ 高生産性ミドルウェアの開発

1. 高性能アプリの開発・実行・メンテナンスを支援するアプリケーションプログラマ向けフレームワーク
   - 【既存技術】
     - オブジェクト指向，マルチレベルAPIをもつ軽量クラスライブラリ群
       - » 並列分割管理，データ管理，I/O，AMR，プロセスマッピング，格子生成クラスライブラリを用いたアプリ開発フレームワーク
       - » データ構造（直交等間隔，八分木，非構造，BFC，重合格子，離散点群）
   - 【革新技術】
     - 超並列アプリの問題の記述に適した並列化フレームワークの提供

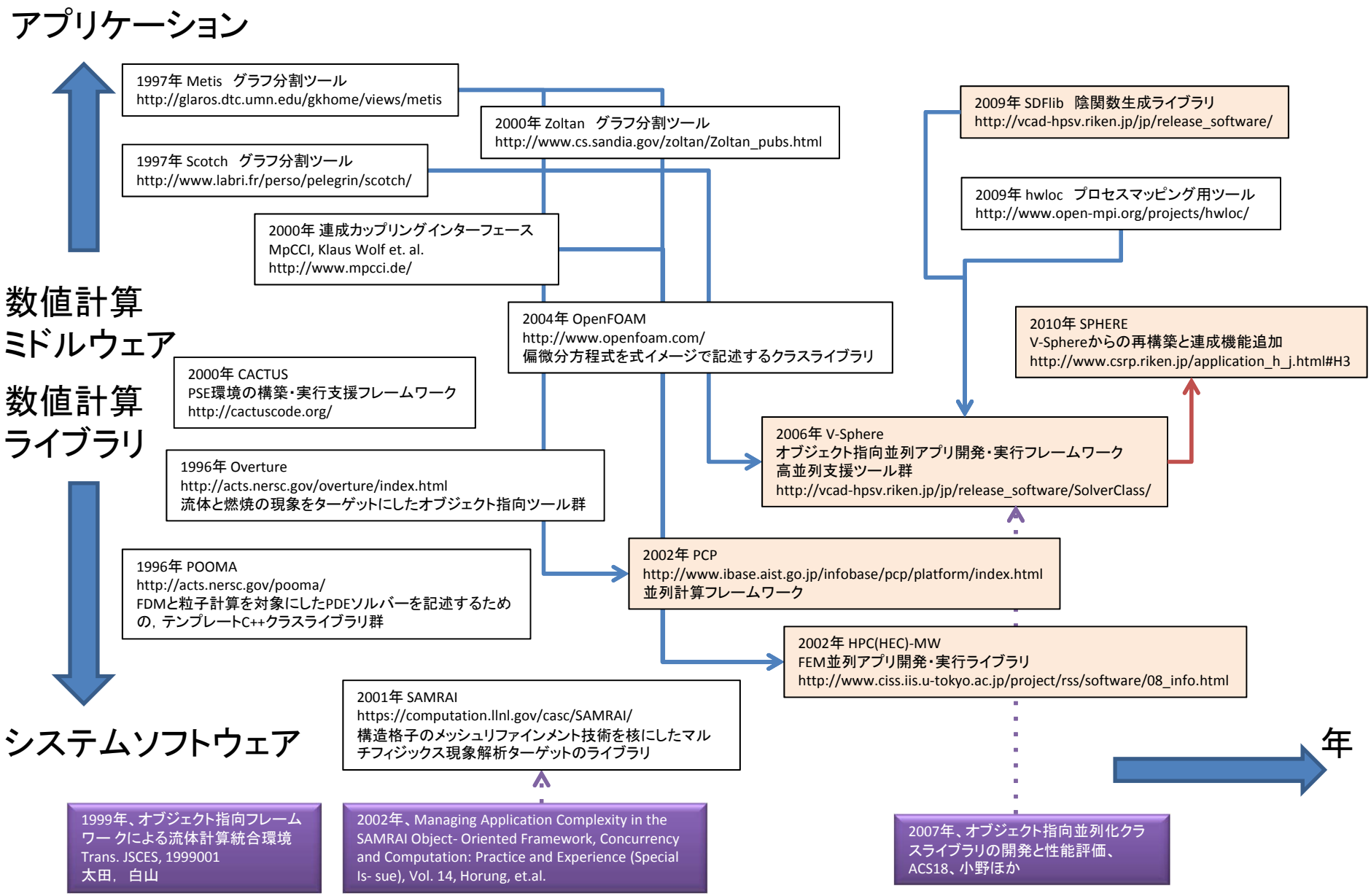
2. 記述性と実行性能の高いPDE系プログラム記述のユーザ向けアプリケーション・ミドルウェア
   - 【既存技術】
     - Mathematica
   - 【革新技術】
     - PDEを記述する直感的なユーザーインターフェイスをもち，高性能な計算をバックエンドで行うミドルウェア

技術項目 2/2

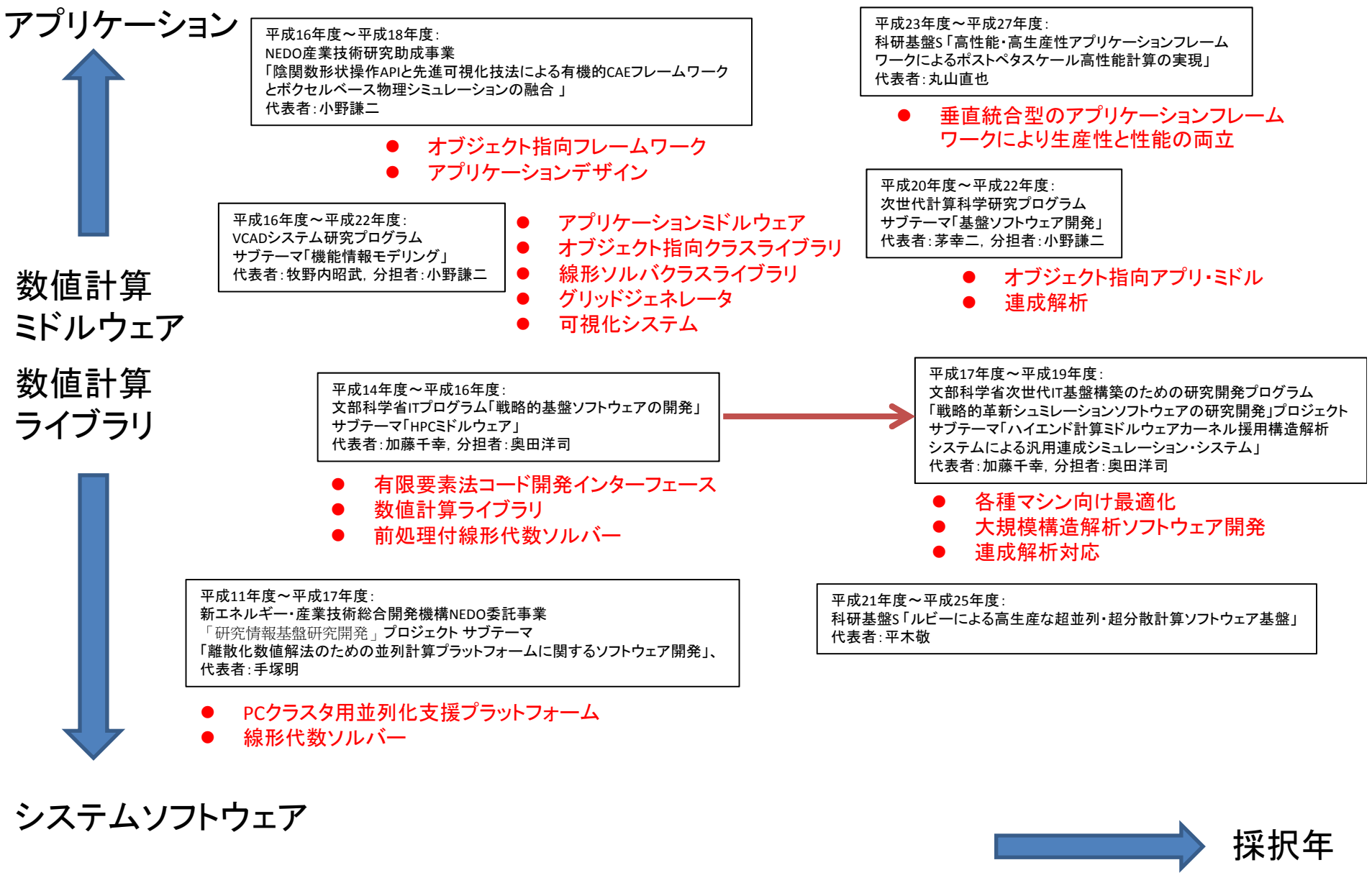
■ 高性能・高信頼アプリの要素技術開発

1. 超並列対応の領域分割と並列データ管理技術
   - 【革新技術】
     - 高いロードバランスで領域分割し，省メモリで並列領域を管理するデータ構造
2. 高性能ファイルI/O管理技術
   - 【既存技術】
     - データ圧縮を併用した高速ファイルI/O技術
   - 【革新技術】
     - 大規模・多量のファイルハンドリング技術
3. 領域分割法に対する高レイテンシ・非均質ネットワーク利用技術
   - 【既存技術】
     - 計算空間の並列性と物理コア（ネットワーク）の並列性のマッピングの最適化
     - ネットワークトポロジに対応した高速通信アルゴリズム
4. 超並列対応の負荷分散技術
   - 【既存技術】
     - 計算中に著しく負荷が変動する動的負荷分散技術，マイグレーション，
     - 動的負荷分散に伴う格子細分化ライブラリの整備
5. 故障コアに対するリカバリー対応技術
   - 【革新技術】
     - 故障発生時の継続実行をアプリレベルで制御する技術の開発
6. ヘテロジーニアスなメニーコア利用技術
   - 【革新技術】
     - CPU,GPUやそのメニーコアを利用する数値ライブラリの利用技術
7. 超並列探索・最適化アルゴリズム
   - 【革新技術】
     - 大規模な制約充足問題、最短経路問題、プランニング問題など

ミドルウェア技術マップ



我が国でファンディングされたミドルウェア研究俯瞰図



ミドルウェア分野：優先度（主項目）

技術完成目標年度	技術項目	必要性	重要性	2011年における研究進捗
2012年	高性能ファイルI/O管理技術	★★★	★★★	フェーズII
2012年	超並列対応の分割並列データ管理技術	★★	★★	フェーズII
2013年	高性能アプリの開発・実行・メンテナンスを支援するアプリケーションプログラマ向けフレームワーク	★★	★★	フェーズII
2014年	領域分割法に対する高レイテンシ・非均質ネットワーク利用技術	★★★	★★	フェーズII
2015年	超並列対応の負荷分散技術	★★★	★★★	フェーズI
2017年	記述性と実行性能の高いPDE系プログラム記述のユーザ向けアプリケーション・ミドルウェア	★★	★★★	フェーズI

ミドルウェア分野：優先度（副項目）

技術完成目標年度	技術項目	必要性	重要性	2011年における研究進捗
2012年	高性能/高機能ファイルハンドリング技術	★★	★★	フェーズII
2014年	データ圧縮を併用した高速ファイルI/O技術	★	★	フェーズI
2014年	プロセスマッピングの最適化	★	★★	フェーズII
2015年	動的負荷分散アルゴリズム開発	★	★★	フェーズII
2015年	粒度の異なる負荷分散アルゴリズム開発	★★	★	フェーズII
2016年	超並列探索・最適化アルゴリズム	★★	★	フェーズI
2017年	故障コアに対するリカバリー対応技術	★	★	フェーズI
2018年	ヘテロジーニアスな実行コアの利用技術	★★	★★	フェーズI

ロードマップ（数値ミドルウェア分野）

年	研究開発機能	備考
2011～ 2012	● フィージビリティスタディと開発計画の詳細化 1．アプリ，計算機科学，ミドル開発分野の研究者の議論 2．対象アプリ領域選定，共通機能の選定，開発プライオリティ 3．ミドルウェア実装の検討	
2012～ 2013	● フレームワークのプロトタイプ開発 1．軽量クラスライブラリ群の設計開発 2．エクサ向けの機能要素開発 • ネットワークトポロジ特性を利用した通信最適化など 3．ペタスケール環境での性能評価	● センター群 ペタコン運用 ● 「京」本格運用
2014～ 2015	● フレームワークのアップデートと展開 1．要素技術のアップデートに対応して，実装変更 2．機能拡張とエクサ向けアルゴリズムの導入 3．フレームワークのリリースと既存関連アプリの移植	
2016～ 2017	● アプリケーションミドルウェアの構築 1．開発したフレームワークを用いて高レベル記述のラピッドプロトタイピング用のミドルウェアを構築 2．フレームワークの継続的なアップデートとサポート 3．プリエクサ級環境での実証	● 10ペタ級稼働 ● プリエクサ級が稼働
2018～ 2019	● エクサスケールへの展開 1．エクサスケール環境でミドルウェアの有効性を検証 2．ミドルウェアとアプリの継続的なアップデートとサポート	● エクサ級が稼働

コミュニティ形成について

- 日米欧における差異
  - 欧米：MW利用とコミュニティ形成の土壌がすでにある
  - 日本：コードは自己開発（特にHPC分野）が主
- 差異の生じた原因は？（推測）
  - これまで（ペタ以前）は自己開発で特に問題なかった
  - 生産性のメリットが少なかった？
    - コミットする労力（アプリ，CS，MWそれぞれの開発者による議論，コーディング，テスト，etc）＞利用するメリット
  - 開発担当の将来的なサポートの問題
    - 開発とサポートが時限的（プロジェクトや有期雇用）
- エクサスケールに向けて
  - 生産性の点から必須のアプローチ
    - ミドルウェア開発と計算機科学の諸分野、アプリ開発者の協同なくしては成功しない

コミュニティ形成について

- つづき
  - 未完成のMW
    - アカデミック性と完成度の乖離
    - すべての（計画段階での）機能が実装されずに終わる
    - マニュアル，WEB，セミナー，etcの不備・不足
      ＝＞First adopterが付きにくい

解決策は…？

1. 責任を持って開発する組織を設置
2. コミュニティ形成と維持を進める
3. 何より、アプリ開発者のコミット

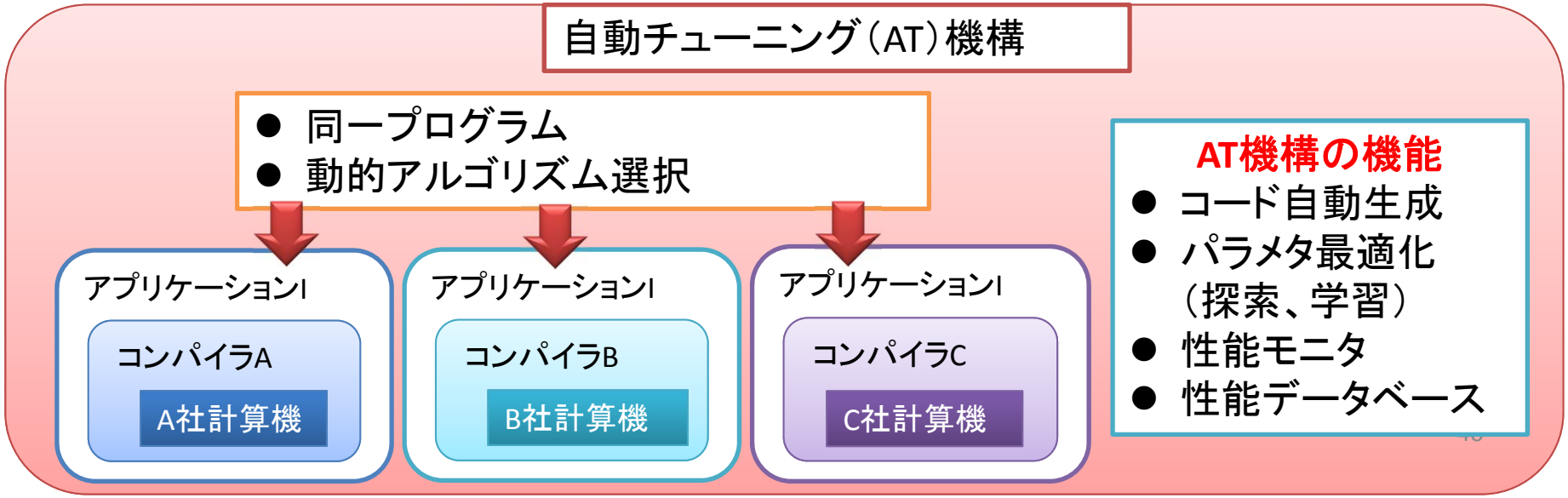
ロードマップ
（数値計算ライブラリWG：
数値計算ライブラリにおけるAT分野）

文責：片桐孝洋（東大）

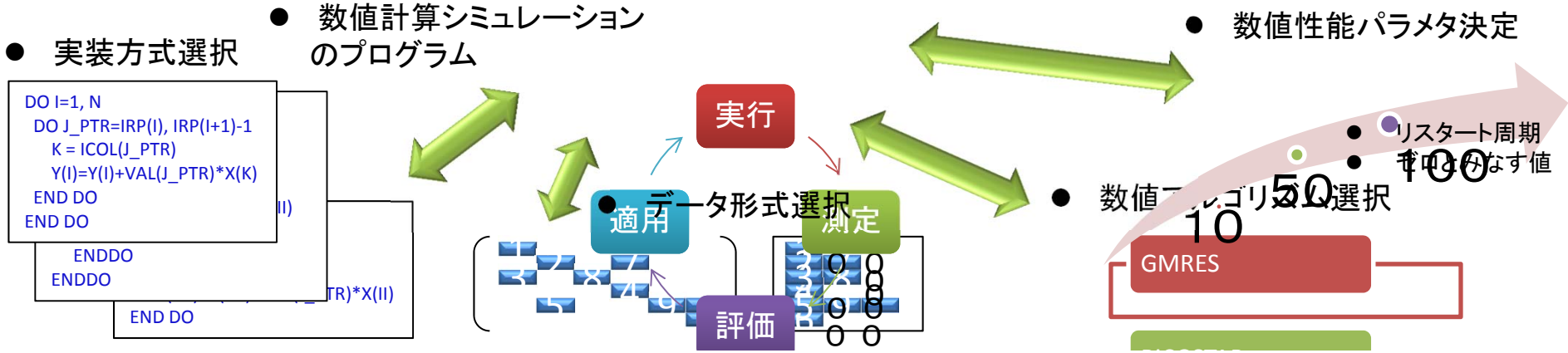
2011年12月8日

自動チューニング（AT）とは

- 複数計算機で有効な最適化の提供
  - 同一プログラムで、計算機が変わっても高性能を維持　→性能可搬性



- 実行時情報を利用した最適化の提供
  - 動的な数値特性の悪化に対応する数値アルゴリズム選択
  - ネットワーク形状、並列実行数など、実行時にしか判明しない情報を利用した最適化



はじめに

- 数値計算ライブラリのための自動チューニング（AT）技術は、数値計算ライブラリで現れる主演算において、計算機環境に自動適応し、高性能となる実装方式を自動生成できる技術
- AT技術で実現される実装方式の最適化は、単にプログラミングの仕方に留まらず、数値計算アルゴリズムも考慮し最適アルゴリズムを選ぶアルゴリズム選択も含まれる
- ＜汎用的＞に行われるAT技術について、エクサフロップスを実現する数値計算ライブラリに必要な技術項目に焦点をあて本ロードマップを作成
- エクサフロップス環境で実現するAT機能
  - エクサフロップス環境で想定される多様な計算機環境に自ら柔軟に適応すること。ソフトウェア性能を最適化できる枠組み（ソフトウェア構築法）を提供すること。このことで、より高い実効性能を実現するソフトウェアが開発可能
  - コンパイラが行うことができない最適化（コード最適化、自動並列化など）を提供すること。性能チューニングに関する工数を削減し、結果として、ソフトウェア開発の全体工数の削減
- 研究開発の背景と技術的制約
  - 非均質CPU(マルチコアCPUとGPUの混合)が普及する
  - 実行時に判明するなんらかの電力制約がある
  - 通信性能が劇的に低下する（高レイテンシになる）
  - 大規模問題実行時に演算精度が著しく劣化する
  - 高性能を達成するためにはアルゴリズム選択が必要となる
  - ハードウェア故障を考慮しないと安定実行ができなくなる

課題と機能の一例

- ATに求められる機能
  - コード自動最適化 （例：ATLAS：BLAS実装の最適化）
  - BLASを呼び出す上位部分の最適化 （例：反復解法の主反復部分）
  - 実行時の条件（ノード数、ネットワーク形状、数値特性、など）に応じた数値計算アルゴリズム（実装方式）選択
  - 対象：既存の数値計算ライブラリ、プログラムの一部（レガシーコード）
- ヘテロジニアス環境最適化
  - 性能クリティカルな場所を、問題サイズ等に応じ、CPUとGPUで切り替えて最適化する機能を提供するAT機構
- 電力最適化
  - 周波数を低くしてもあまり遅くならない演算部分を自動的に見つけ、適する周波数を自動設定するような低電力化を提供するAT機構
- 超低レイテンシ通信最適化
  - 実行時に定まるメッセージサイズやトポロジに応じ、通信のやり方、書き方を切り替えるAT機構

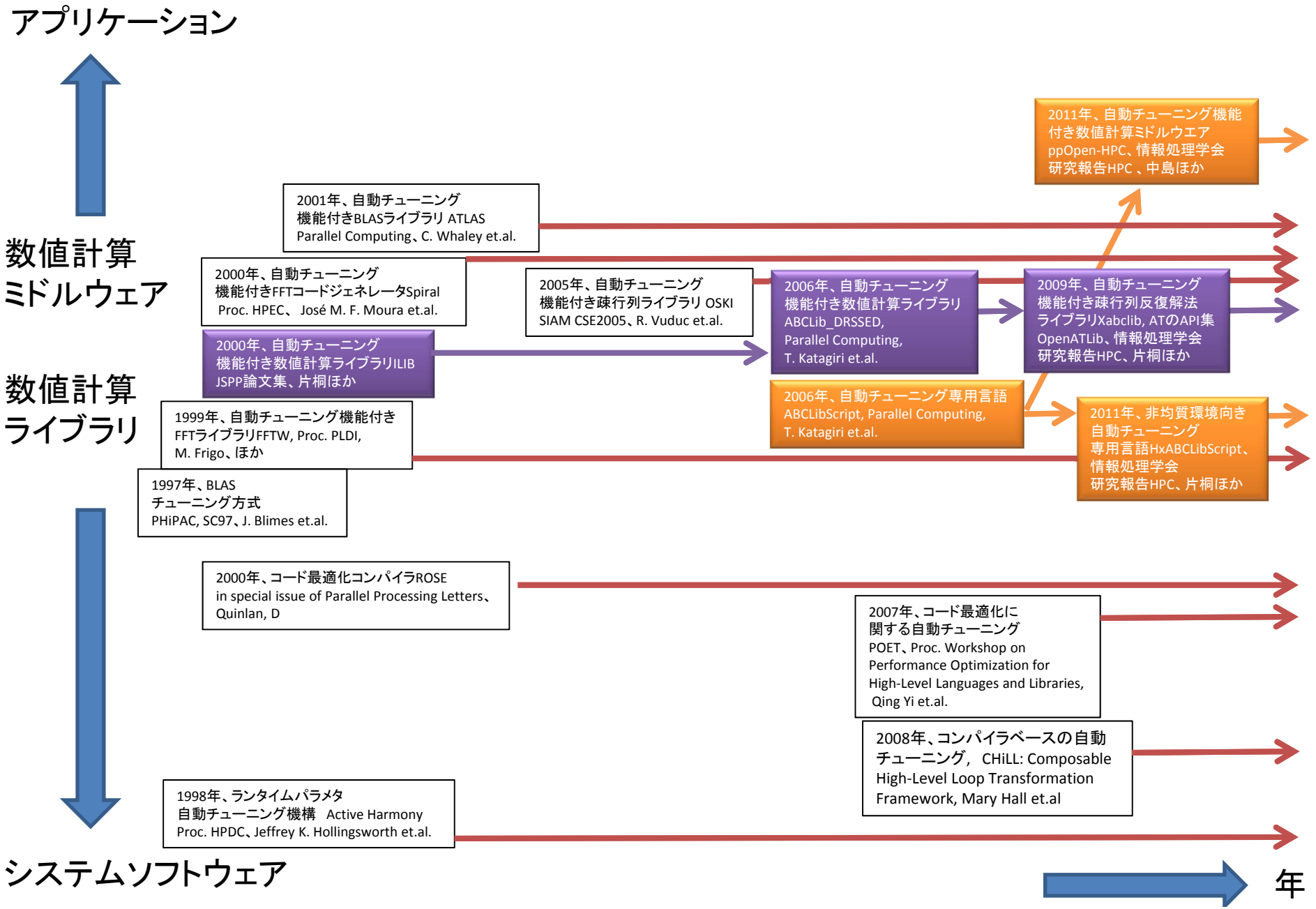
課題と機能の一例

- 数値計算アルゴリズム選択
  - 実行時に定まる問題サイズや数値特性に応じ、数値計算アルゴリズムを切り替える機能をもつAT機構
  - 低並列実行と高並列実行のアルゴリズムを実行時に切り替える機能をもつAT機構
- 混合精度・高精度演算（多倍長演算）・精度保証
  - ユーザの精度要求に応じ、単精度と倍精度の演算を反復解法中で適切に切り替える機能をもつAT機構
- 耐故障性
  - リスタートポイントを、アプリ特性に応じ、自動的に高速となる場所に入れる機能をもつAT機構
- 他WGとの連携
  - システムWG：性能プロファイリングとATの連携
    - ホットスポット同定とAT言語との連結

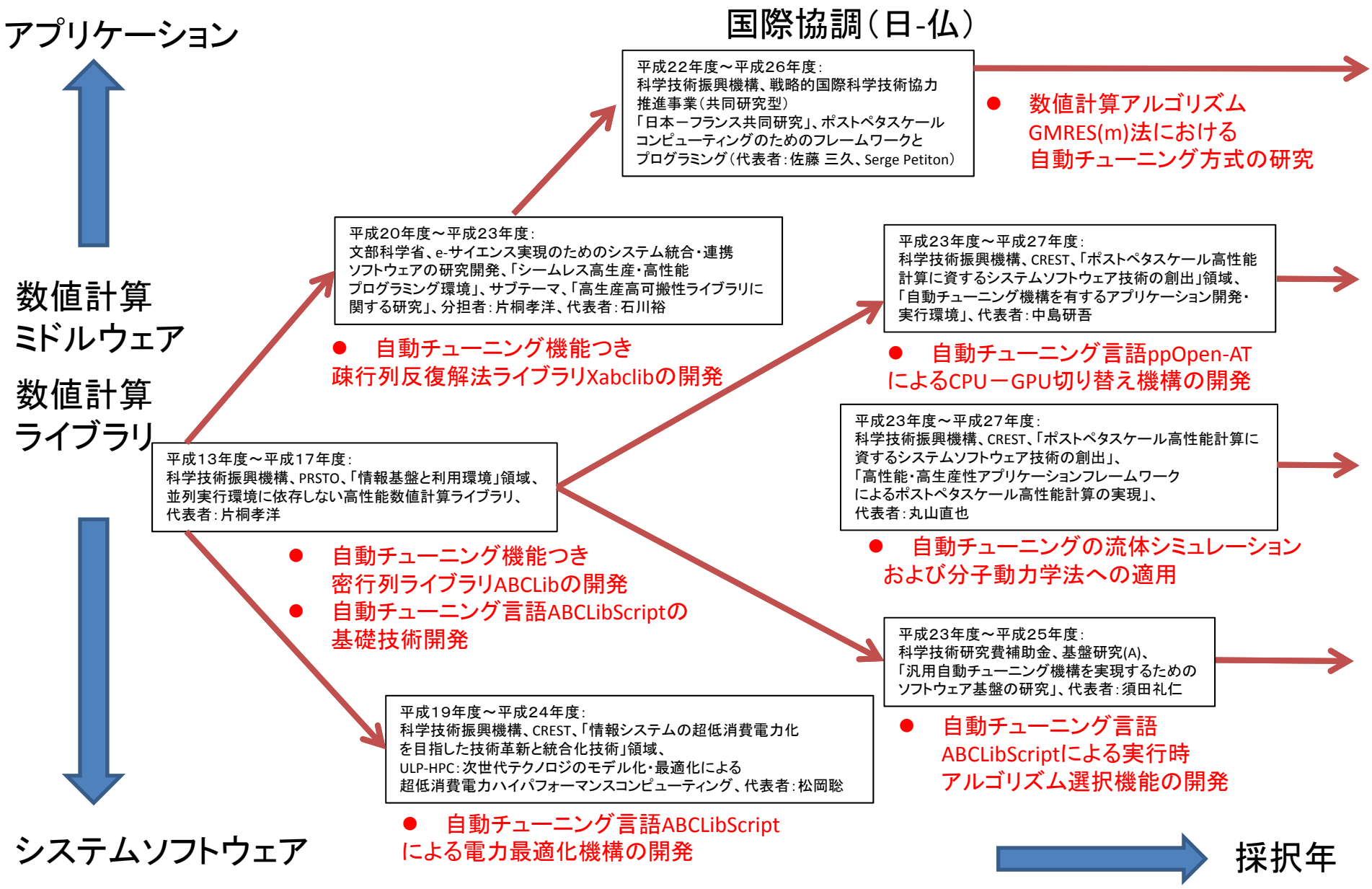
既存のATソフトウェア

- ソースコードが公開されている、もしくは何らかのサポートのあるもの
- 数値計算ライブラリ
  - 密行列BLAS
    - ATLAS (U. Tennessee)（提供：国内スパコン）
  - 疎行列-ベクトル積
    - OSKI (UC Berkeley)（提供：不明）
  - 高速フーリェ変換（FFT）
    - FFTW (MIT)（提供：国内・国外のスパコン）
    - Spiral (CMU) (提供：不明)
  - 疎行列-反復解法ライブラリ（連立一次方程式、固有値問題）
    - Xabclib（東大）（文科省E-サイエンス、提供：東大基盤センタースパコン）
- 数値計算ミドルウェア
  - 5種の数値計算法による基本機能提供
    - ppOpen-HPC（東大）（JST CREST、提供：東大基盤センタースパコン（予定）、AICS京（予定））：ppOpen-ATによる自動チューニング機能の実現
  - ステンシル計算用フレームワーク
    - Physis（東工大）（JST CREST、提供：AICS　京（？））
- コード最適化機能
  - AT専用言語
    - ABCLibScript (電通大／東大)（サポート言語：Fortran90、C）
    - ppOpen-AT（東大）（JST CREST、提供：東大基盤センタースパコン（予定）、AICS京（予定））

自動チューニング（数値計算ライブラリ、AT専用言語）技術マップ



我が国でファンディングされた
自動チューニング（数値計算ライブラリ、AT専用言語）研究俯瞰図



ATの技術項目

対象となる最適化を自動化する技術

1. **ヘテロジニアス環境最適化**
   - 目的：プログラマが意識せず計算対象ごとに最適な計算機資源を選択
   - マルチコア最適化、ハイブリッドMPI最適化のためのAT機構
   - CPU-GPU自動切り替えのためのAT機構

【既存技術】CPU-GPU切り替え機能を自動付加するAT言語
【革新技術】ヘテロジニアス環境に適用可能な性能モデル化手法

2. **電力最適化**
   - 目的：プログラマが意識せずプログラムに対する電力最適化がされる
   - ジョブレベル、演算カーネルごとに適する電力最適化（周波数の変更、電力が低いCPUの選択、等）の自動化のためのAT機構
   - どのようなポリシで電力最適化するかという「電力最適化ポリシ」と、自動チューニング専用言語の確立

【既存技術】コンパイル時の周波数切り替え
【革新技術】電力最適化のための、性能モデル、最適化ポリシ指定方式、AT言語

3. **超低レイテンシ通信最適化**
   - 目的：プログラマが意識せず実行状況に適応した通信高速化がされる
   - 完全網から3Dトーラス網などへのネットワークアーキテクチャ変更に伴う＜高レイテンシ化＞に対応する通信処理の最適化のためのAT機構
   - 通信ライブラリ（MPI）の実行時の実装方式・アルゴリズムの自動切り替えのためのAT機構
   - 実行時の物理ノード割り当て情報からの通信最適化のためのAT機構

【既存技術】インストール時のMPI実装方式切り替え
【革新技術】通信実装方式を実行時に切り替えるAT方式およびAT機構

ATの技術項目

対象となる最適化を自動化する技術

4. 数値計算アルゴリズム選択

- 目的:プログラマが意識せず問題サイズや数値条件に応じたアルゴリズムが自動選択される
- 入力データ特性(行列の数値特性や問題サイズ等)を自動抽出し最適アルゴリズムを選択するためのAT機構
- 数値シミュレーションが進むたびに変化する数値特性に応じた最適化のためのAT機構

【既存技術】アルゴリズム選択のためのAT言語

【革新技術】汎用的なアルゴリズム選択のための性能モデルとそのAT言語への適用

5. 混合演算・精度保証による数値計算安定化

- 目的:プログラマが意識せず要求に応じた高精度化や安定化がされる
- 以下の演算の導入を行うことで実現するAT機構
  - 混合精度演算、多倍長演算、区間演算の導入
  - 基本演算に対する高精度演算と混合精度演算の導入
- 反復解法中で内積演算のみ高精度化する等の混合演算のAT機構
- 「実行速度」「メモリ量」「演算精度」のうちの最適化ポリシ記述(「数値計算ポリシ」記述)
- 解の一覧を与えた上、プログラム変換により、ソルバ実装が変化してもテスト問題は正しく解けていることを保証してくれるメカニズム(アルゴリズム検証の自動化)

【既存技術】多倍長演算ライブラリ、混合演算ライブラリ、特定アルゴリズムの部分高精度化手法

【革新技術】精度保証手法とそのライブラリ、数値計算ポリシ指定方式、アルゴリズム検証方式、数値計算安定化のためのAT言語

6. 耐故障性対応

- 目的:プログラマが意識せずオーバーヘッドの低い耐故障性方式の自動選択がされる
- 数値計算アルゴリズム特有の知識記述でチェックポイント・リスタートを低レイテンシ化する技術を利用したAT機構
- 実行時間と耐故障性のトレードオフ考慮し、方式を自動選択する「耐故障性ポリシ」の確立
- 「ある確率で失敗するかもしれない計算」という部品を記述し、それをもとに100%に近い確率で成功するよう計算全体を組み立てるフレームワーク(対故障ソフトウェアの部品化)

【既存技術】アプリケーション知識を利用する低レイテンシな耐故障化

【革新技術】低レイテンシ耐故障化のためのAT方式とAT言語、耐故障性ポリシ記述方式、耐故障ソフトウェア部品化

他WGとの関係

- アーキテクチャWG
  - ATによる最適化の前提
    - ヘテロジニアス環境
    - 電力最適化
    - 数値計算アルゴリズム選択：メモリバンド幅・大規模並列
    - 超低レイテンシ通信最適化：ノード間インターコネクト
    - 耐故障性：リスタートポイントの最適化
- システムソフトウェアWG
  - ATのための 要素技術（ != AT機構 ）
    - 生産性（性能プロファイラとATの連携）
    - ランタイム（MPI集団通信の最適化）
    - ヘテロジニアス環境最適化：通信・アクセレータ間最適化
- プログラミングWG
  - ATのための言語および実行環境
    - ヘテロジニアス環境最適化：演算アクセレータプログラミング技術、
大規模並列：NUMA構成、メモリ：データ分散技術
    - 耐故障性：MPIレベルでの耐故障機能
    - 電力最適化：電力測定のためのAPI
    - ツール：生産性、性能プロファイリング　と　AT機能の連携
    - ハイレベルプログラミング：DSL　とAT機能の連携

AT分野：優先度（主項目）

技術完成目標年度	技術項目	必要性	重要性	2011年における研究進捗
2014年	非均質環境最適化	★★★	★★★	フェーズIII
2016年	電力最適化	★★	★★★	フェーズII
2016年	超低レイテンシ通信最適化	★★★	★★★	フェーズI
2014年	数値計算アルゴリズム選択	★★	★★	フェーズII
2018年	数値計算安定化	★	★★	フェーズI
2020年	耐故障性対応	★★	★★	フェーズI

AT分野：優先度（副項目、短期）

技術完成目標年度	技術項目	必要性	重要性	2011年における研究進捗
2014年	非均質環境最適化（CPU-GPU）：疎行列用のデータフォーマット切り替え	★★★	★★★	フェーズIII
2014年	非均質環境最適化（CPU-GPU）：自動チューニング専用言語への適用	★★★	★★★	フェーズIII
2014年	電力最適化：自動チューニング専用言語への適用（含、周波数切り替え、電力最適化ポリシ）	★★	★★	フェーズII
2014年	超低レイテンシ通信最適化：実行時の通信ライブラリの実装最適化	★★★	★★★	フェーズI
2014年	数値計算アルゴリズム選択：疎行列反復解法アルゴリズム選択	★★	★★	フェーズII
2014年	数値計算アルゴリズム選択：自動チューニング数理コアのライブラリ化	★★	★★	フェーズII
2016年	数値計算安定化：多倍長計算、精度保証計算の基本線形計算ライブラリへの適用	★	★★	フェーズI

AT分野：優先度（副項目、長期）

技術完成目標年度	技術項目	必要性	重要性	2011年における研究進捗
2016年	自動チューニング向け性能プロファイル（データベース化）・可視化	★	★★	フェーズI
2016年	自動チューニングプログラムのデバッグ支援、AT品質保証	★★	★★★	フェーズI
2018年	自動チューニング指向システムソフトウェア（OS、ミドルウェア、スケジューラの自動チューニング化）	★	★★	フェーズI
2020年	ATのためのコスト推定モデルの、自動推薦、自動選択、自動構築（数値計算ポリシの自動記述）	★	☆	フェーズI
2020年	耐故障性ポリシ、耐故障性ソフトウェア部品化	★★	★★	フェーズI

ロードマップ（AT分野）

年	研究開発機能	備考
2011～2012	● 現存スパコンでATの有効性検証 1. 非均質環境最適化のプロトタイピング 2. 電力最適化の基本設計 3. 通信最適化の基本設計 4. 数値計算アルゴリズム選択のプロトタイピング	● 大学センター群等ペタコン運用開始 ● 「京」供用開始（10ペタ級稼働）
2013～2014	● 10ペタ級でATの有効性検証 1. 非均質環境最適化の実現 2. 電力最適化のプロトタイピング 3. 通信最適化のプロトタイピング 4. 数値計算アルゴリズム選択の実現 5. 数値計算安定化の基本設計	● 10ペタ級スパコンの運用開始（大学センター群等）
2015～2016	● 100ペタ級でATの有効性検証 2. 電力最適化の実現 3. 通信最適化の実現 5. 数値計算安定化のプロトタイピング 6. 耐故障性対応の基本設計	● プリエクサ級が稼働
2017～2020	● エクサ級でATの有効性検証 5. 数値計算安定化の実現 6. 耐故障性対応のプロトタイピング	● エクサ級が稼働