計算科学と計算機科学
アカデミアとインダストリの
新しい融合・連携を目指して

朴　泰祐

筑波大学大学院　システム情報工学研究科／

筑波大学　計算科学研究センター

計算科学と計算機科学の連携

- 個人的経験から（その１）
  - 筑波大学計算物理学研究センター／計算科学研究センターでの成功体験（CP-PACS, FIRST, PACS-CS,...）
    - 計算科学（ニーズ）：サイエンスドライバとして、大規模・高性能計算が必要だがある程度以上のスケーラビリティと性能を得るためのスキル不足。アマチュア的方法論。
    - 計算機科学（シーズ）：テクノロジドライバとして、システム構築やプログラミングのノウハウがあるが具体的問題が見えていない。箱庭研究になる可能性あり。

    ⇒これらの融合により極めて強力で有効性のある研究チームを構築、先端的計算科学諸問題への超並列・HPC技術の適用を推進した。
  - 互いに必要なことはわかっているが「出会いの場」に乏しい
    - いわゆる「共同研究」ベースでは不十分。日常的な議論や進捗状況のチェックが不可欠。
    - アプリ側：そもそもコード開発者が少ない。PD等の若手は基本的にアプリの人しか雇っていない。
    - CS側：実アプリは敷居が高い。アプリのチューニングを手伝っても評価されない。

アカデミアとインダストリの連携

- 個人的経験から（その２）
  - CP-PACS, PACS-CS等の開発
    - アカデミアのアイデアとインダストリの技術の融合。
    - 実情：インダストリに余裕がない、自社システムのキープが手いっぱい。（HPCがビジネスとして回っていないという背景）
      実はコモディティベースでもできることは一杯ある。これすらなかなかできていない。
  - 米国と比べた日本の現状
    - 日本：ベンダー主導の導入、オペレーション。アカデミア側で技術をもった人が育たない。（悪い意味で）分業制ができてしまっている。
      ⇒システムコストやオペレーションコストとして最適化されていない
    - 米国：国研等では、大規模システムの開発から導入まで、産学連携が強力に行われている。オペレーション、問題解決に至るまで、アカデミアが人と金をかけ実施している。

    ⇒技術的にも経済的にも、両者の強力な協調関係が「HPCエコサイクルを確立できていない」のが日本の現状

大規模化するHPCシステムとアプリケーション

- 現在の性能向上は基本的にシステムスケーラビリティの向上で成り立っている
  - プロセッサ周波数の頭打ち、Byte/FLOP比の限界、ムーア則の限界
  - アプリ側の大きな誤解:「PFLOPSはTFLOPSに比べ千倍速い」
    ⇒「PFLOPSはTFLOPSの千倍のことが実行できる」
  - スケーラビリティとデータローカリティを無視したコードやアルゴリズムの性能向上は極めて難しい
- 高性能／大規模化における応用とシステムの意識の乖離
  - 従来システムでは何とかなっていたことが崩壊しつつある(例)
    - システムの規模を無視したアルゴリズム。MPIにおける集合通信の多用、不用意な同期、適切でない同期／非同期通信の組み合わせ等
    - ファイルI/Oや通信性能に関する過度の期待。大量の入出力要求、膨大な数のファイル生成、ローカルストレージとグローバルストレージの不適切な利用等
    - メモリ階層を無視したデータアクセス、マルチコアに対する過度の期待

⇒次世代でも危機的状況、ましてや次々世代では…

次世代スパコンに望ましい連携のあり方

- 計算科学／計算機科学の真の融合を目指す
  - 計算科学研究機構（神戸）
    - 組織構造から運用まで、両分野の研究者及びアプリ分野間の研究者が自由に議論し、協調できる環境を整備。両分野の研究者が「同じ釜の飯を食べ、戦友になる」。
    - 戦略機関との連携：各機関からの技術的相談に対する対応（特にシステム側研究者からのアドバイスを各機関にフィードバックできる仕組み）。
    - 次世代をやりながら次々世代を考える。
  - 戦略機関（各拠点）
    - 次々世代まで使えるアプリ開発。
    - システムを意識したマインドを持つ。コードチューニングや大規模化に関し、計算機屋と積極的に議論する。
  - CS側
    - アプリ側が努力することを期待しつつ、従来以上のシステム性能向上、アプリが使いやすいシステムやライブラリ開発を行い、現実に即した方法論やツールを開発。
    - システム屋でなく、並列数値処理の専門家も重要

アプリケーション／並列数値処理／並列処理システムの三位一体

次々世代につなげるために

- HPCエコサイクルを回す努力
  - スパコンを「消費」から「活用」へ
    - スパコンに投入したベンダーの開発費用を民生に広く活用
      ⇒実際はなかなか難しい
    - HPCに携わる人々が使いやすく、使いたくなるシステムの開発
      （例：T2K連携で行っている e-サイエンスプロジェクト）
  - システム開発から運用までを高いコスト効率で
    - ユーザの「お客さん」意識の脱却、ベンダーとの一体感を持ったシステムの利用と開発
    - 良い形の分業体制の確立。システム開発から深く関わることで、運用上の問題解決もベンダー任せにならないはず。
- 人材育成
  - 計算科学と計算機科学に跨る人材育成
    - アプリとシステムの両分野が互いをrespect、必要に応じ相互のPD雇用や綿密な共同研究を推進
    - 教育制度上でこれらを意識する⇒dual degree等
    - 「境界」⇒「複合」：これらの人材を高く評価し登用する体制
      （”half” ではなく “double” ）

計算科学と計算機科学の新しい連携を目指して

- 互いに対する respect
  - 両者の関係は give & take からさらに踏み込んだものへ
  - CS:「お手伝い」⇒「アプリは役に立つHPC研究の実験場」
    - HPCは「実学」であり、実用に供するために必要な研究ネタは一杯ころがっている
    - 大規模実験には大規模実システムが不可欠
  - アプリ:「お客さん」⇒「自分でもシステムを理解する」
    - スケーラビリティや効率が高いアルゴリズムを考え、コードを開発しないと自分自身が生き残れない
    - 人材の登用方法の再検討
  - 数値計算:両者の橋渡し
    - 「三位一体」を構成するジョイント
    - 計算モデルや基本アルゴリズムから見直さないといけないアプリは一杯あるはず
  - 論文は共著、成果を共有

- 計算科学研究機構はこういうマインドをもった運営を！