‘Rightweight’ Languages

田浦健次朗
東京大学

個人的バックグラウンド

- 1993- 並列 OO 言語 ABCL on 富士通 AP1000+
- 1997- 超軽量マルチスレッディング (StackThreads/MP), 並列 C/C++ 言語
- 1996- 分散 / 並列 GC
- 2005- InTrigger cluster of clusters プラットフォーム
- 2004- 並列分散シェル GXP, 並列分散ワークフロー GXP make

テーマ: 高水準並列言語, 高水準並列処理

高水準並列言語研究の小史

1. **夢多き時代 (∼'90 前半):**
   - アーキテクチャも色々(MPP, データフロー, etc.)
   - 言語も百花繚乱 (自動並列化, 関数型, 論理型, オブジェクト指向, 融合, etc.)
2. **コモディティ時代 ('90 後半 ∼):**
   - PC クラスタ + Linux ベース + MPI という「基本フォーム」に収束. HPC 普及・産業利用に貢献
3. **現在:**
   - マルチコアの並列処理は文字通り「誰もが必要」に
   - 多コア × 多ノードの「超並列」路線
   - アーキテクチャも多様化 (ヘテロノード)

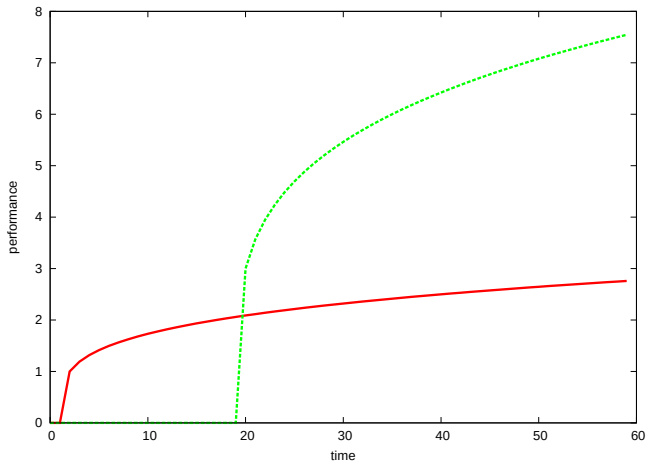
   これらを使いこなすのに高水準言語が必要だと考えるのは極めて自然

今後の並列言語研究開発の必要性

- **計算中心のいわゆる HPC**
  - ノード数増大, ハイブリッド並列, ヘテロノード, . . . 大変
  - ヘテロノードの最適化基準は猫の目, 将来不明
  - **正しい「抽象マシン」がわからない, 投資寿命読めない**
- **「大きな」並列処理** (Parallel Processing at Large)
  - 多数の逐次プログラム, 複数の並列タスクを組み合わせた並列処理
  - サイトをまたいだ並列 (主にデータ) 処理
- **「実データ処理」中心の並列処理**
  - ゲノム, 天文, 文献情報処理, etc.
  - シミュレーションデータのポストプロセシング解析 (既存 HPC でも重要)

高水準並列言語「成功」の定義

- 「C + MPI より 2 (3,4,...) 倍遅かったらダメ」はどれほど重要なのか (なぜ世の中では P* が使われているのか)?
- 本当の評価基準は「結果」を出せるか・もしくはそれを出すまでの時間 (のはず)
- 「結果」とは? 科学的な発見 > 最大の問題サイズ etc. > FLOPS
- ⇒ 時間オーバーヘッド < メモリオーバーヘッド < 生産性



「高水準」HPC 言語が直面する困難

- 私見: 問題は遅そうな事より, 「生産性・移植性」を達成できていないこと
- マシンが変わると苦労するのは, アプリもシステムソフト (含言語) も同じ,
- . . . どころか (ふつう) 「アプリの移植 << 言語の移植」
- その日速いだけの (将来どうなるかわからない) アーキテクチャへの投資は困難 (費用大, 効果小)
- ふたを開けるまで分からないマシン用の開発はできない

(まっとうな) 打開策:

- 障壁を下げて利用者を増やす
- 売れそうなマシンを作る
- マシン開発の切れ目をなくす

もう一つの HPC: 「大きな」, 「データ集約的」並列処理

軽量化・超高水準化・多様 (目的指向) 化する言語

- 既存コンポーネントを統合するための, **ワークフロー・スクリプト (glue) 言語** (Swift, GXP make, Xcrypt, . . .)
  - 10,000 並列くらいを簡単に扱えるソリューション
- **データ集約的計算のための言語**とアーキテクチャ(Hadoop, Dryad, . . .)
- **大域的な計算**を実際に行える処理系 (yet to come . . .)

戦略:

- **軽量化:** 既存プログラムの composition に徹して移植性向上, 役割分担
- **低参入障壁化, 高費用対効果化:** デスクトップや小クラスタから, シームレスにスパコンへ

研究テーマ

- **Lightweight な言語:** 「データ処理」を最適化する
- **Lightweight な file system:** 大域データを即席で「統合」する.
- 計算ノード **co-located disk** と並列ファイルシステムの, ハイブリッド・適材適所利用 (信頼性, 性能, アクセスパターン, etc.)
- co-located disk の有効利用と, jitterless なマシン運用の**両立**
- 階層的な局所性 (ローカル, サイト内, サイト間, 大陸間) を意識した**データ集約的計算の大域的最適化**

「クラウド」に学ぶべき事

現状のクラウドが HPC に使えないというのはたやすい, だが人はそれでも期待する...

- **参入障壁の低い売り方:** 「あなたのデータ」をクラウドで処理してあげます
- **自由度の高い使い方:** あれはできないこれもできない, と言われる心配がない
  - このソフトのインストールは勘弁!
  - スパコンと外をつなぎたい!? とんでもない!

研究体制について

- システムソフトウェア屋の投資基準は「将来性のある (少しは息の長そうな) アーキテクチャ」
  - オープンソース研究体制のためには必須
- 巻き返しのためには, 枕詞ではない**「次のマシン (the next machine)」**の上で研究ができなくてはならない
- 「次世代」はおろか「現世代」の大型マシンも, システム研究にはなかなか使えないのが現状 (「安定運用」oriented?)
  - HA8000 512 ノードサービスや, TSUBAME hpc キューは貴重