HPCI FS第2回全体ミーティング
ミニアプリベンチマーク

丸山直也（理研AICS）

2012年9月19日

ミニアプリベンチマーク

1. フルアプリの収集

2. フルアプリ整備
   - 担当: フルアプリ提供者

3. フルアプリのミニアプリ化
   - 担当: ミニアプリ開発者

フルアプリ収集のまえに・・・
ミニアプリの取り扱いについて

- 公開
  - プロジェクト終了後速やかにソースコードを外部公開
  - 理由 → 選出過程の透明化
  - 狙い
    - 次世代アーキテクチャ向けプログラム実装のリファレンス
    - 次世代アーキテクチャやシステムソフトウェアの研究開発における評価ベンチマーク
- ミニアプリのライセンス
  - 理想
    - ベンチマークセット全体で単一のライセンス
  - しかし、
    - フルアプリがGPL→ミニアプリもGPL
    - 全体GPLは制約が強すぎ
  - 案
    - 各ミニアプリについてGPLもしくはApacheの2択
- ミニアプリ著作権者
  - フルアプリ著作権者＋ミニアプリ作業実施者

フルアプリ実行要件案

- 対象アーキテクチャ
  - X86クラスタ
  - K/FX10
- 言語
  - C99
  - C++ (no C++0x?)
  - Fortran 2003
  - その他？
- 外部ライブラリ
  - 計算用標準
    - BLAS、LAPACK、FFTW、他？
  - I/Oライブラリ
    - NetCDF、HDF、MPI-IO、他？
  - 他？
- 並列化
  - MPI
  - OpenMP
  - 自動並列化

フルアプリ整備

1. コード整備
   - 不要コードの除去
2. ビルド手順
   - 対象アーキテクチャ上でMake等でビルドできることを確認
3. 実行手順
   - 問題サイズ（やその他実行条件）の決定
     - 2020年頃に実行したい問題設定から決定
     - 必要に応じて、評価・検証用の問題サイズも
     - 問題サイズ可変が望ましい
   - 入力ファイルの整備
     - 問題サイズに応じた入力ファイルの整備
4. 検証方法
   - 実行結果の機械的な検証
     - 「可視化結果を人間がみて判断」→ NG
   - 並列化、アーキテクチャの違いによりビット単位では結果のばらつき有り → 許容可能な検証プログラムが必要
5. ドキュメント
   - 計算する問題の概要
   - 計算スキーム
   - 2018-2020年頃の想定実行規模

ミニアプリ化

- ベンチマークとしての整備
  - いまのところ共通時間計測ルーチンの挿入
  - 最終的にはビルドの統一化も
    - パス等の環境設定の統一化
    - make一発で全ミニアプリビルド
- コードの整備
  - リファクタリング
  - 不要コードの除去
- 参照実装バージョンの作成
  - 最適化なしに素直に記述したバージョン
- 性能モデルの構築
  - 目的: 与えられたアーキテクチャにおける性能の推定
  - 手順
    1. フェイズ分割
    2. フェイズ毎の計算、データアクセスパターン解析
    3. 計算・データアクセスパターン毎の性能モデル化

SCALEのミニアプリ化

SCALE（AICS富田チーム）

- 数m-数百mの空間解像度での気象現象をシミュレーション
- 直交格子直接法差分計算
  - 倍精度
- 約3万行のFortranコード
- MPIによるノード間並列化、自動並列化コンパイラによるノード内並列化
- 力学過程
  - ステンシル
  - メモリ律速
- 物理過程
  - スカラー計算
  - 計算律速

SCALEフルアプリの整備

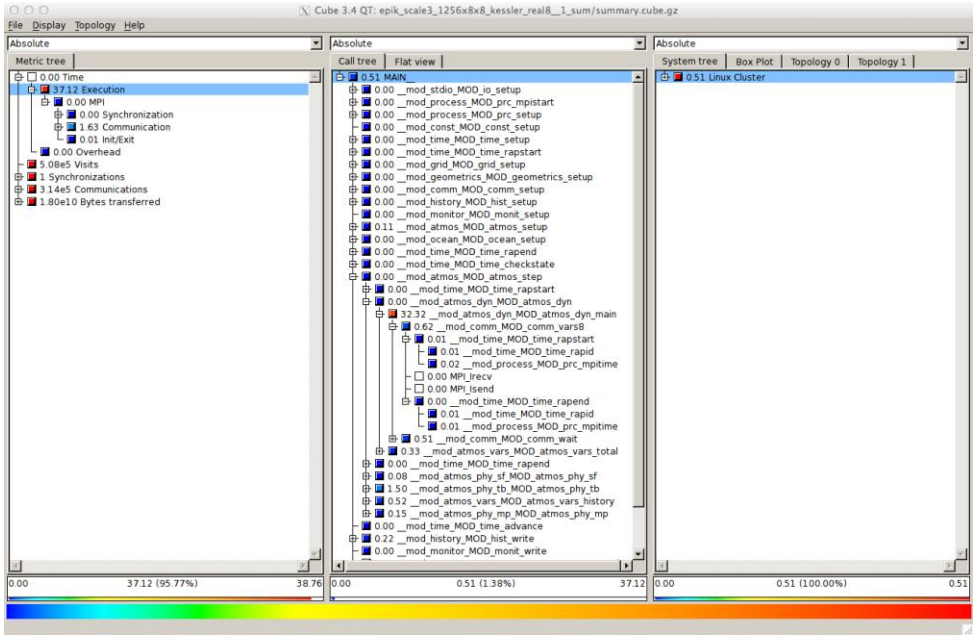
- ビルド
  - Intelクラスタ、京、FX10用Makefileもともと有り
- 実行
  - 問題サイズ
    - 2020年ごろの想定シミュレーション領域から決定
    - 各プロセスの問題サイズは系全体を均等分割
    - メッシュのサイズ調整により適宜変更化
  - 入力ファイル
    - 実行条件をパラメータとして持つ入力ファイル生成スクリプトを用意した
- 検証
  - 結果検証プログラムを用意した

SCALEのミニアプリ化

- 比較的新しいコードであるため元々それほど複雑ではない
- リファクタリング
  - 共通コードのサブルーチン化、不要コードの除去
  - 3万行 → 2万行
    - 大気部分で1万行（カーネルは主に大気だけ？）
- 最適化無しシンプルな参照実装バージョン
  - オリジナルでの最適化
    - 2次元ブロッキング
    - RDMA通信
  - それぞれコンパイル時オプションとして無効化可能

SCALEミニアプリの性能モデル: フェイズ分割

- 実行時間プロファイル取得→力学過程の時間が主
- 力学過程→内部で3つのパートに意味的に分割済み
  - SETUP, RK, FCT
  - それぞれを別個のフェイズとして定義



- やったこと
  - Scalascaによるプロファイル用領域指定コードを挿入

SCALEミニアプリの性能モデル: フェイズ毎のパターン解析

- RK(ルンゲクッタ)フェイズ
  - ステンシルの3重ループ27個
  - 演算回数、メモリアクセス回数の調査
    - 人手で数え上げ
    - 自動化ツール？
  - メモリアクセス
    - プログラムの字面上のアクセス != 実際のアクセス
    - 実効メモリアクセス = 理想的なLLCを仮定したアクセス
      - 一度読んだデータは必ずキャッシュヒット
      - ≈最適化されたメモリアクセス
  - 見た目のB/F: 4.8
  - 実効B/F: 1.9
  - 明らかにメモリボトルネック

性能モデル ➔ システムのメモリバンド幅 / 実効B/F

RKの性能モデル検証

- Intelマシンにて実験
  - Xeon E5-2670 (2.6GHz x 8 cores), 166 GFLOPS (Peak)
  - DDR3-1600 (4 channels), 51.2 GB/s (Peak)
  - STREAM TRIAD: 33 GB/s
- RK結果 (1256x8x8)
  - OpenMP指示文を挿入
  - Intelコンパイラ v12.1.0
    - -fast -xHost -opt-streaming-stores always
    - 全ステンシルループのSIMDベクトル化

```
RK    Time:                          2.371 (S)
RK    FLOPS:                        15.160 (GFLOPS)
RK    Throughput:                   72.331 (GB/S)
RK    Effective Throughput:         30.007 (GB/S)
```

だいたいSTREAMの結果に近い実効バンド幅が出てる

SCALEミニアプリ化残作業

- 力学過程他のフェイズの性能モデル構築
  - 同様にメモリバンド幅から推定
- 物理過程の調査
- 通信性能の調査
- ファイルI/Oの調査
- 電力性能への拡張
  - プロセッサ単体
    - Intel: RAPL を利用
  - システム全体
    - プログラマブルに消費電力データにアクセスできる必要あり
    - 東工大TSUBAME2では可能

今後の予定

- フルアプリの決定
  - ひとまず9月中
- システム評価会合
  - 10月1日予定
  - 他のFS3チームにも参加してもらう
  - フルアプリ紹介
  - ミニアプリ化作業ロードマップ確認

ミニアプリの選別

- システム評価の観点からの選別が必要
  - 重複の排除： 特定の計算ルーチン・パターンがボトルネックとなるミニアプリが複数→一つを代表として選択
- 選別方法
  - システム評価の観点より機械的に選別
  - 性能特徴からクラスタリングなどを検討中

FSウェブサイト（準備中）

- http://hpci-aplfs.aics.riken.jp/
- 構成
  - FSについて
  - フォーラム
  - 会議案内
    - 次回全体会議からは参加受付もウェブで
  - 資料掲載