FX10 利用講義「並列数値計算」

須田礼仁

東京大学情報理工学系研究科

１．はじめに

「並列数値計算」は情報理工学系研究科の大学院講義である。隔年で実施しており，平成 28 年度にも S（夏）学期に開講した。英語名は“Parallel Numerical Algorithms”としている（日本語と英語は微妙に対応していない)。講義はスライド・説明とも英語で実施しているため，留学生が多く参加している。今年度は情報理工学系研究科，工学系研究科のほか，新領域創成科学研究科，理学系研究科の学生を含め約 50 名が履修登録した。氏名から判断すると，そのうち半数以上が留学生とみられる。今回は交換留学で比較的短期に滞在し，講義の単位を取る必要がある学生が多く参加したことが特徴的であった。

２．講義の概要

今やマルチコアでない計算機を探す方が難しいぐらいである。ワークステーションでは 4 CPU で 96 コアといったマシンも手に入る。安価な PC でも 2 〜 4 コアあるのに加え，GPU を使えば手元でも容易に並列計算ができる。今後まだコア数は伸びてゆくと思われるので，並列計算は誰にとっても重要な技術となってきている。また，本講義の実施は昨年度（平成 28 年度）の前半であったため FX10 を用いたが，同年度後半には我が国最高の実行性能を誇る Oakforest-PACS，最新の CPU および GPU を搭載した Reedbush-U/H が設置されたことで，様々な分野で大規模並列計算が広く行われてゆくことが期待できる。一方で，計算機アーキテクチャは Intel の MIC に基づき，AVX512 を搭載する Intel Knights Landing プロセッサから構成される Oakforest-PACS，Intel Xeon を CPU，NVIDIA Pascal GPU をアクセラレータとして持つ Reedbush のように，広がりを見せている。近年はさらに FPGA の性能も向上し，解く問題によっては最新の CPU や GPU を凌駕する高い性能を達成している。

このような背景のもと，前回平成 26 年度の実施のときから，本講義では並列計算の一般論と各論とを分離し，一般論として述べられる点はできるだけ一般的な記述をするように構成した。これにより一般論が統一的な視点で理解できるようになった。ただし，一般論部分は話が抽象的になりがちで実感がわきにくくなっているかもしれず，今後とも継続的な改善を要するところである。各論においては，特に大規模並列を念頭において MPI，OpenMP，CUDA の 3 種類を取り上げている。また，分散メモリの MPI を最初に説明して OpenMP と CUDA はやや軽めに説明している。時間的には半年の講義ではこれで限界と思われるため，いくつかの重要なパラダイムが組み込めていない。ひとつは単体性能の最適化が十分に説明できていない。命令レベル並列性，SIMD 命令，キャッシュ最適化など，部分的には説明しているが，これらの HPC（ハイパフォーマンスコンピューティング）の基礎技術が十分に説明できないうらみがある。もうひとつの側面として，近年注目を浴びているタスク型の並列性，PGAS などのメモリモデル，FPGA といった新しいプロセッサについては説明できていない。後述のように FX10 での実装を課題として出している観点からこれらの話題を取り上げるのには限界があるが，残念と言わざるを得

ない。幸い，同じ情報理工学系研究科において，共有メモリ並列処理でタスク型のモデルもきちんと取り上げている講義があり，多くの学生はそちらも受講してくれているようである。しかし，情報理工学系研究科の中で FPGA による高性能計算を教えている講義はないかもしれない。FPGA は数値計算で必要とする倍精度浮動小数点数は得意としていない点がネックであるが，注目されているアーキテクチャであるので，実際に触って演習することもできる講義があればよいのだが。

３．講義の内容

平成 28 年度には 13 回の講義を実施することができた。以下に各回の内容の概要を示す。

第１回：イントロダクション

この講義で取り上げる並列計算とは何かを，類似性のある並行計算，分散計算と対比する形で定義した。特に，この講義では再現性があり予測可能な計算を取り上げ，動的な並列性は最小限とする。続いて，ハードウェア並列性の基礎概念として，複数演算器とパイプライン，SIMD と MIMD，分散メモリと共有メモリ，UMA と NUMA，およびそれらのハイブリッド，均一性・不均一性などを導入した。また，この講義では所要時間を主なメトリックとすること（スループットではない），そこから高速化率，並列化効率，理想高速化率，スーパーリニアーなどの概念を導入した。また，強スケーリングと弱スケーリング，アムダール則とグスタフソン則といった相対性能の基本法則を導入した。加えて，絶対性能として flops およびピーク性能を説明した。また，所要時間測定時の注意点についても説明しておいた。

第２回：並列性と依存性

並列性とは依存性のないこと，よって並列性の解析は依存性の解析に他ならないことからスタートした。続いて，データ依存と制御依存，RAW/WAR/WAW 依存などの依存性の類型を示した。配列，リスト，木などのデータ構造へのアクセスの並列性，ステンシル計算と超平面法・ループスキュー，木によるリダクション，カスケードによるスキャン，3 項漸化式の並列性，パイプラインなどの依存性と並列性を事例として挙げた。一方で計算順序が変えられる場合にはリオーダリングにより並列性が変わること，その表現としてカラーリングが便利であることも示した。また配列への間接アクセスがある場合を取り上げ，その並列性の抽出方法には多数ありうることを示した。

第３回：局所性

並列計算は高性能を目指すものであるが，性能という観点では局所性は避けて通れない重要事項である。まずは分散メモリでも共有メモリでも局所性が低いと高い性能が期待できないことを説明した。局所性には計算強度と粒度の２つの側面がある。まず計算強度や B/F 値について説明し，行列積を例として，プログラムの書き方次第で計算強度が変わることを示した。また行列ベクトル積のような計算強度の弱い計算もあることも注意した。リダクション，FFT，ステンシル計算の計算強度について概観し，領域分割と Owner-computes-rule が局所性の観点から利点があることを説明した。次に粒度について説明し，パイプライン計算を例に並列性と粒度のトレードオフがあることを示した。最後に単純な通信性能モデルを導入し，計算強度と粒

度が性能にどのように影響を与えるかを解析し，その単純化したモデルとしてルーフラインモデルを紹介した。

第４回：スケジューリング理論

この講義では離散最適化問題に分類される古典的なスケジューリング理論についても紹介をした。HPC 分野とはやや用語が異なるため，まずその点を注意した。そのうえで，タスクとスケジューリングの基本概念，不均一プロセッサの類別，メイクスパン，ガント図，グラハム記法などを導入した。P||Cmax に対する LPT の最悪性能比， P|prec|Cmax に対するリストスケジューリングの最悪性能比，P|intree, pj=1|Cmax に最適解を与えるレベルスケジューリング，P∞|prec|Cmax に最適解を与えるクラスタリングスケジューリングを説明した。また，クリティカルパス，タスク挿入，タスク重複などの概念を紹介した。不均一プロセッサやオンラインスケジューリングについては既知のアルゴリズムの性能を示すにとどめた。そのかわり，Divisible Load Theory およびループスケジューリングについて紹介した。

ここまでが一般論であり，このあと各論に入る。

第５回：MPI 入門

メッセージパッシング，Local View，SPMD を説明したのち，MPI の基礎の一対一通信を説明した。Eager と Rendezvous プロトコル，ブロッキング・非ブロッキングの違い，メッセージの到着順序などについて注意をしたうえで，リダクションとステンシル計算を題材に実際の簡単な MPI 並列プログラムを説明した。

第６回：集団通信

MPI の集団通信について，基本的な集団通信と，それを実現するためのいくつかのアルゴリズムを説明した。プロセッサ数とデータサイズにより，異なるアルゴリズムが最適になりうることを確認した。また Broadcast と Reduction などの双対性，Scatter と AllGather で Broadcast が実現できるなどの集団通信の組み合わせについて解説した。

第７回：FX10 の使い方

情報基盤センターの大島聡史先生に FX10 の使い方について説明をしていただいた。

第８回：共有メモリ並列性特論

インド工科大学ハイデラバード校から滞在されていた Sathya Peri 先生に，共有メモリ並列計算における計算の正しさの基礎を教えていただいた。前述のようにこの講義では主に再現性のあるアルゴリズムしか教えていないが,Peri 先生のこの講義では実行のタイミングによって結果が異なりうるような計算について，基礎的な概念を教えていただいた。

第９回：分散データ構造

まず，分散メモリ並列計算におけるデータ構造のあり方の一般論を導入した。続いて，最も単純な１次元ベクトルについて，ブロック，サイクリック，ブロックサイクリック，可変長ブロック，要素ごと指定の５種類の分散方法を示した。続いて２次元の場合には列１次元分散，

行１次元分散，２次元分散の選択があり，LU 分解を例題として利害得失を考察した。またステンシル計算における袖領域を説明し，遅延隠蔽，piggy-backing を説明した。さらに，疎行列ベクトル積の事例，および動的負荷分散（この文脈では動的データ分散）の概念を紹介した。

第 10 回：OpenMP 入門

OpenMP の入門として，global view であること，また自動並列化ではなく正しさはプログラマの責任であることを注意した。基礎的な構文を説明したのち，OpenMP で陥りがちな誤りとして，共有・私有変数の区別，レース条件，弱い一貫性とメモリ同期の必要性を説明した。あわせて，reduction 節や atomic 節も紹介した。

第 11 回：OpenMP 高性能プログラミング

OpenMP の性能低下要因とその回避・軽減策を紹介した。排他制御のための構文である atomic，critical および lock 関数の違い，ループスケジューリングの選択肢，アフィニティに関する注意などを行った。また，キャッシュ周りの性能向上手法として，パディング，タイリング，ループ交換，ループ結合，Arrays of Structures/Structures of Arrays などの手法を紹介した。また MPI + OpenMP ハイブリッド並列化の必要性について説明した。

第 12 回：CUDA 入門

アクセラレータという概念，GPU の性能特性を導入した。続いて，CUDA の基本的な書き方の説明をした。

第 13 回：GPU の高性能計算

GPU において高い性能を達成するために重要な概念を説明した。GPU のハードウェアと CUDA の並列性階層の対応を説明し，SM ごとに走るブロックの数を決定する式を示した。またスレッドを実行する機構を説明し，並列性により遅延隠蔽が可能となることを説明した。分岐命令の削減，coalesced メモリアクセスの性能，CPU-GPU 間データ転送時間について注意を促した。

４．課題

この講義では全部で５回の課題を課した。最初の２つの課題は各自が使える計算機により行い，これらの課題を提出した学生には FX10 のアカウントを配布して，残りの３つの課題を行わせた。

第 1 回：CPU モデルと計時方法の確認

各自のプロセッサのモデルを確認し，ピーク性能を計算させた。また，計時方法を適当に選ばせ，その精度を測定により求めさせた。

第 2 回：Flops への挑戦

「どんな計算でもよいので，できるだけ高い flops 値を出すプログラムを書け」という課題を出した。また，これに加えて，十分長いベクトルのコピー，内積，和の性能を測定させ，それらからメモリスループットを測らせた。

前者については，様々なプログラミング言語を用いた報告が寄せられた。最も性能が低いものは 10 Mflops 以下，最高性能は 200 Gflops を超え，最大約 30,000 倍もの性能の違いが観測された。また，第 1 回の課題で求めたピーク性能と比較することもあわせて，性能について考えずにプログラミングをすると，非常に低い性能しか出ないということを理解してもらえたと考えている。

第 3 回：FX10 を使ってみる

FX10（Oakleaf-FX）を用いて，ピンポン通信による通信性能の測定，および簡単なリダクションかステンシル計算を MPI で実行させて，FX10 の使い方を確認してもらった。

講義においては UNIX のコマンドが使えること，C 言語か Fortran のプログラミングができることを受講条件としている。それにもかかわらず，PC と FX10 でうまくデータが転送できなかったり，コンパイルと実行のスクリプトの違いを理解できなかったりする学生が出た。FX10 の使い方について，一部の学生には個別の丁寧な指導をする必要があることが明らかになった。

第 4 回：MPI プログラミング

密行列計算，偏微分方程式の数値解法，もしくは個人的に興味のある計算のいずれかを選んで，MPI でプログラミングをして FX10 で実行し，弱スケーリングと強スケーリングでの性能を報告させた。

馴染みが深いせいか LU 分解を選んだ学生が多かったが，枢軸選択の必要性やブロック化をしないと性能が出ないことなどから，苦戦した学生が多かった。

第 5 回：MPI + OpenMP ハイブリッドプログラミング

第 4 回の課題をベースとして，OpenMP 並列化を施し，MPI + OpenMP のハイブリッド並列計算を行わせた。第 4 回の課題ができなかった学生については，OpenMP のみの並列化を行わせた。

５．おわりに

全体として講義はスムーズに進み，FX10 のアカウント発行も問題なくできた。今年度は前述のように，FX10 を使えない学生が何名か出たが，ある程度その原因も理解できていて，やむを得ないものと思われる。しかしこれらの学生にも教育機会を提供する必要があるので，一部の学生には個別の指導をする必要性がある。一方で，演習課題をこなすことができた学生たちは，講義で説明した様々な高性能化手法のうち，自分の課題にどれをどのように適用すればよいか比較的スムーズに判断できたようである。ただし実装にはてこずっていた。

次回からは Reedbush-H を用いれば GPU の演習も可能になるので，特に演習の部分について再検討をはかってゆきたいと考えている。

最後になりましたが，FX10 の使い方を説明していただいた大島先生，アカウントを発行する作業をしてくださった方々をはじめ，スーパーコンピュータを用いた講義という貴重な機会を提供していただいた情報基盤センターの皆様方に感謝を申し上げます。