Oakbridge-CX 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，2018 年に続く開講となる。英語名は“Parallel Numerical Computations”としている。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，理学系研究科，総合文化研究科，新領域，USTEP の学生を含め約 60 名が履修登録した。うち 3 名が USTEP 生，5 名が交換留学生であった。また 8 名は博士後期課程の学生であった。

2020 年は新型コロナウイルス感染症 COVID-19 に揺れた 1 年であり，この講義もすべてオンラインで実施することになった。さらに，4 月当初は準備が整わない学生がいる可能性を想定して，最初の 2 回は受講に必須の話をしないということになったので，並列計算ではない，いわゆる単体の高性能化について話をした。その分，GPU の詳しい話が入れられなくなったため，GPU を搭載した Reedbush-H ではなく，Oakbridge-CX を利用した。

２．講義の概要

計算機の性能向上は並列性によって実現されている。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。膨大な計算量を必要とする近年の機械学習の隆盛もあって，GPU をアクセラレータとする計算機が広く使われるようになっている。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ところである。

このような背景のもと，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したつもりである。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 と OpenMP の 2 種類のプログラミングモデルを取り上げている。また，SIMD 型の計算として，CPU の AVX，GPU の説明を短く入れた。分散メモリの MPI を最初に説明して，その後 OpenMP をやや軽めに説明している。今回は最初の 2 回を単体性能の最適化に当てたため，GPU が十分に説明できなかったのが残念である。また，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。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

令和２年度には 13 回の講義を実施した。すべて zoom によるオンライン講義で，配信の録画を復習用に学内限定で提供した（一部録画しそこなったものを除く）。この講義での録画の活用状況は確認していないが，一般的には復習ができることは好評だったという。

以下に各回の内容の概要を示す。

第 0.1 回：単体性能の高性能化（１）

２回にわけて，単体性能の高性能化について話をした。かなり以前に行っていた講義の資料を急いで英語にしたものである。性能に影響を与える要因として，最内ループ長，CPU パイプライン，レジスタ，キャッシュについて，ハードウェアの構成から話をした。また性能を向上させる可能性のある手段として，ループ交換，ループ融合・分離，ループアンローリング，ソフトウェアパイプライニング，レジスタブロッキング，ループタイリング・ストリップマイニング，データ圧縮，データ構造変換などを紹介し，実機での性能例を示した。

第 0.2 回：単体性能の高性能化（２）

前回に引き続き，性能に影響を与える要因として，キャッシュライン，ラインのアドレスへのマッピング，アソシアティビティ，ページングと TLB について，ハードウェアの構成を説明した。データの再利用性を高める手法として，ループ交換などによるストライドの変更，データ構造の工夫，パディング，一時コピーの作成などを解説した。こちらも実際の性能例を示した。

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できることを示した。

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ている。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性能オーダーを示すにとどめた。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：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ることを説明し，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８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９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 10 回：Oakbridge-CX の使い方

情報理工学系研究科所属で，「計算科学アライアンス」の特任講師である松本正晴先生に，Oakbridge-CX の使い方について説明をしていただいた。内容を非常に初等的なものにしていただき，Linux の最重要コマンド，アカウントの登録から，ログインやデータの送受信，コンパイルの方法，バッチとキューとは何か，どうやってタスクをキューに入れるか，など，丁寧に説明していただいた。また，各学生が自分の PC から接続しているので，実際にログインやテストコードのコンパイル・実行までの最小限の事項についてハンズオンをしていただいた。若干の学生がログインできないなどのトラブルが生じた。多くはその場で解決できたが，パスワードが漏れるのを防ぐために画面共有を使わなかったので，ログインができない学生がごく少数残ってしまった。

第 11 回：CUDA 入門

アクセラレータという概念，GPU のハードウェア特性を導入した。続いて，CUDA の基本的な書き方の説明をした。ホストとデバイス，ブロックやスレッド，メモリ階層や同期など，しっかり理解しなければならないことが多いため，講義 1 回で説明できるのはごく入門に限られてしまう。GPU を使うからには性能を出さなければ意味がないのだが，それについては下記のような資料配布に留めざるを得なかった。

これらに加えて，講義で説明ができなかった以下の 2 点について，ITC-LMS で資料を配布した。

資料１：CPU における SIMD

CPU における SIMD として，Intel AVX の説明をした。最初にイントロとして，CPU の SIMD

の歴史，SIMD 関連のハードウェア，SIMD 命令の概要，GPU の SIMD との比較などを説明した。次に SIMD で性能を出すためのコードの準備（データ構造やループ構造の整理，アラインメントやエイリアスなど），続いてコンパイラオプション・指示行や OpenMP 4.5 による SIMD 指示行を説明した。プリフェッチやストリーミングストアについても少し触れた。SIMD intrinsics は存在だけを紹介した。

資料２：GPU における高性能化

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。

CUDA の更新は早く，講義のたびに新しい GPU と CUDA の仕様を確認している。（今回はアカウントを発行しないものの）Reedbush に搭載されている Pascal アーキテクチャに準拠して説明をした。

４．課題

講義の最初の３回で，UNIX コマンドと C/Fortran プログラミングのごく初歩的な問題を３分で解いてもらった。「カレントディレクトリを表示するコマンドは何か」「.o で終わるファイルを全部消すコマンドを示せ」とか，「vi か emacs で，現在カーソルがある１行を消すコマンドは何か」とか，「整数変数 n と m を宣言せよ」，「Hello, World! を表示するプログラムを書け」というレベルである。2 年前は平均点が半分程度であったが，今回は 7 割ぐらいまでできていた。オンライン環境であったので，その場で調べた学生もいるかもしれないが。いずれにせよ，この 3 分テストによって，UNIX とプログラミングができないとこの講義では困るということは理解されたのではないかと思う。

この講義では全部で 3 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 Oakbridge-CX のアカウントを配布して，最後の課題を行わせた。

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。

今回は，例年に比べると極端に低い性能を報告する学生がほとんど見られなかった。これが講義の最初の 2 回で単体性能の高性能化を説明したことによるのだとすると，やはりこの部分もある程度きちんと説明することには十分な意味があるようである。

第3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 Oakbridge-CX 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。前回に引き続き，Cholesky 分解と熱伝導方程式については逐次プログラムをダウンロードできるようにした。

例年は，第2回と第3回の間に，「スパコンを使ってみよう」という課題があったが，今年は時間的な余裕がなく，それがなかったことの影響を当初心配していた。しかし，例年に比べて今年は学生たちは順調に課題をこなしたようである。

後に学内で課題とされたのだが，オンライン講義であることにより，学生たちが相互にどのぐらいの理解をしているかよくわからず，復習やレポート課題に例年以上の時間をかけていたとも言われている。

５．おわりに

並列計算の話ができる講義が2回減ってしまったが，それ以外は全体として講義はスムーズに進み，Oakbridge-CX のアカウント発行も問題なくできた。例年のことながら，アカウント発行の申請が遅れてしまう学生がいた。例年だと準備していただいている予備アカウントで足りるのだが，今回は1名分が不足してしまい，ご担当の方に急遽アカウント発行をしていただくことになった。迅速なご対応をいただきましたこと，また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。